Chapter 1

Advances in CMOS Circuits

Advances in CMOS Circuits

This section contains selected papers describing the advances being made in CMOS circuits and logic. It starts with a review paper in which Akira Masaki argues in favor of deep sub-micron CMOS over the use of bipolar ECL logic. The paper presents the main arguments explaining why high-speed logic should be using room-temperature CMOS instead of ECL. The paper was written in 1992 and its assertion that CMOS would become a dominant logic technology has proven true today.

At the time CMOS was introduced, however, it was not considered a viable technology for computer logic, whereas bipolar and n-MOS (in a later stage) were dominating the processor and microprocessor development. First consideration as a logic technology was given to CMOS with the introduction of Belmac-32 from Bell Laboratories. Belmac-32 was also the first 32-bit microprocessor on a single chip. The CMOS circuits and technology used in this processor were quite different from the ordinary CMOS known earlier. Even after Belmac-32 was introduced, a debate raged as to what kind of CMOS would be a suitable replacement for n-MOS in computer logic. The evolution of CMOS is described in the next seven papers. It is interesting that some of those circuit techniques were dormant for a while only to find a use in modern high-performance processors today.

The second paper (by Krambeck, Lee and Law from Bell Laboratories) is on CMOS-Domino circuits. This paper was a first serious departure from the static CMOS known in its classical sense as a circuit consisting of a p-type switching network using a function f and an n-type implementation of a complementary function $\bar{f}$ shown in Fig.1.1.

As shown in Fig. 1.1, the CMOS circuit requires twice as many transistors as n-MOS in order to implement the same function. To make this comparison even worse for CMOS, the switching time of the p-transistor is twice as long as that of the n-transistor owing to the different carrier mobility of n- and p-type carriers in silicon. Therefore, the transition form *low-to-high* output voltage (in CMOS) takes twice as long as the corresponding *high-to-low* transition. In order to make the transition times symmetric, the p-transistor switching network is usually built from devices that are twice as large as the n-type transistors. In such a case, we are dealing with a circuit that either has non-symmetric transition times or is considerably larger than if the same function f was implemented in n-MOS logic.

CMOS Domino Logic

To alleviate the CMOS problems, researchers from AT&T Bell Laboratories suggested building CMOS logic that contains only n-type transistors implementing the switching function f. This logic is a dynamic type because two clock-phases are necessary for its proper operation. First, there is a *precharge phase* during which the clock is low, followed by the *evaluation phase* during which the clock is high, as shown in Fig. 1.2. During the precharge phase, all of the nodes N are precharged via the p-type transistor Q_1. The nodes N will stay charged if there is no discharge path in the switching function f during the evaluation phase when the transistor Q_2 is conducting. However, if the nodes N were taken directly as outputs, thus driving the inputs of the next logic blocks, all of the subsequent blocks would start to discharge immediately following the precharge phase, simply because all of the outputs are at the *logic one*. Therefore, the final state of the functional blocks would be undetermined. Domino Logic resolves this problem by passing all of the nodes N through the regular CMOS inverters. Only the output of an inverter can drive the next logic block. In Domino Logic, all of the outputs are at *logic zero* immediately following the precharge stage. Therefore, no discharge can exist in the logic blocks that are themselves driven by other Domino Logic blocks. The evaluation phase starts with the logic blocks being driven by the *primary inputs*. Some of the blocks will be selectively discharged, if a path to ground is established by the logic function represented by that particular block. This would change their outputs

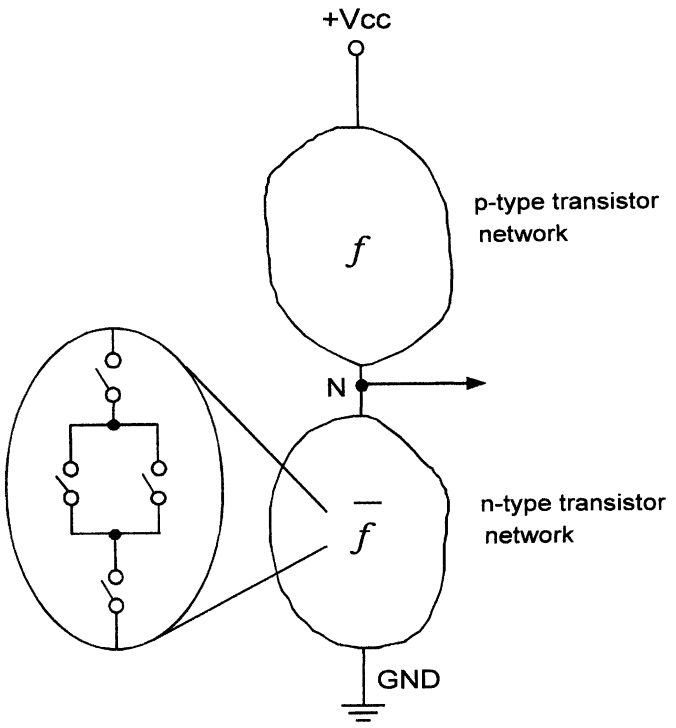


Fig. 1.1. Switch representation of static CMOS circuit implementing a function f.

from logic zero to *one* driving the inputs of subsequent logic blocks. Now if this change in turn creates a discharge path in the subsequent logic blocks, they will discharge, changing their outputs from *zero* to *one*. This change would further propagate through the logic from the primary inputs to the primary outputs like falling dominos. This is where the logic obtained its name: *Domino Logic.*

The benefits of CMOS Domino Logic were not obtained without a cost. First, part of the cycle used for precharge is essentially lost because no logic operation is possible during this time. The ability to rapidly discharge the output node N via the n-type MOS transistor switching network was supposed to offset this loss. This feature is dependent on implementation and is not always true. Furthermore, CMOS Domino is inherently *noninverting* logic which represents difficulties. (E.g., XOR gate implementation with Domino as described by Krambeck, Lee, and Law is not possible). This inability is to some extent compensated by using dual polarity latches at the inputs.

Yet another feature plagued CMOS Domino Logic: the *charge redistribution problem* described in the paper by Oklobdzija et al. The charge redistribution mechanism is illustrated in Fig.1.3. Charge redistribution is manifested by the loss of charge from the node N, owing to the creation of paths leading into the previously discharged parts of the transistor switching network f, but not to the ground node. The output of the Domino Logic block was to stay at logic zero level, because there is no path in the switching network f leading to ground. However, owing to the distribution of charge from the node N into the various nodes in the switching network, the voltage of the node N will drop. This voltage drop might exceed the threshold voltage of the inverter INV, and its output will assume the logic one. The logic one value of the output might in turn discharge the next block (though it was not supposed to do so) leading to the erroneous value. A techniques used to alleviate the *charge redistribution* employs a small feedback p-type transistor Q_f connected to the node N whose function is to replenish the charge lost in the redistribution and return of the node N to the *logic one* value. If the amount of redistributed charge was not large, this might be sufficient. However, depending on the amount of charge lost from the node N, a negative voltage *glitch* of a different magnitude will result on the node N. This glitch may be amplified in the inverter (INV), and it might be sufficient to discharge the next logic block..

The problem posed by the noninverting property of the CMOS Domino Logic is treated in the paper by Goncalves and DeMan in which they propose a new type of Domino Logic, termed NORA, consisting of p-type as well as n-type switching networks that are employed alternatively. The operation of NORA logic is illustrated in Fig. 1.4. Both n-type logic and p-type logic are in the precharge phase where the output node of the n-type switching function is precharged to logic one, while the output of the p-type switching function is discharged to logic *zero*. This is achieved by clocking the n-type and p-type switching functions with the two opposite clock phases. During the evaluation phase, n-type logic blocks will discharge their output to logic zero, if there exists a path in the n-transistor switching function connecting the output to ground. In turn, this may create a path to V_{cc} in a p-type transistor switching network of the next block bringing the output to logic one.

The change would propagate from the primary inputs to the primary outputs alternating between the n-type and p-type switching block. If a particular output is destined to be an input in the next logic block of the same type, the inverters make its operation very similar to Domino circuits which consist of n-type and p-type switching networks. A tristate buffer at the output of a logic section holds its logic value during the precharge phase, thus enabling pipelining of the logic section. NORA logic is also sensitive to charge redistribution, and the large p-type switching transistor blocks do not add to its performance. Though a very interesting concept, even today this type of logic has not found many applications.

The CMOS Domino family was further enhanced by Heller, Griffin, and Thoma of IBM, which resulted in CVSL logic. The IBM authors developed two types of CMOS logic which they called Cascode Voltage Switch: *static* and *dynamic,* (see Fig. 1.5). The first of these types of CMOS logic: static CVSL, consists of two n-type transistor switching functions f and $\bar{f}$, which are connected to V_{cc} via a cross-coupled p-type transistor combination. The advantages of this logic over regular CMOS are obvious. CVSL implements two functions f and $\bar{f}$, as CMOS does except that in CMOS f is implemented with a p-type transistor switching network. Given the inferior speed of p-type transistors, the p-type transistor switching function will usually end up being twice as large as the one implemented with the n-type transistors. In terms of the transistor numbers, CVSL contains two p-type transistors in the cross-coupled combination

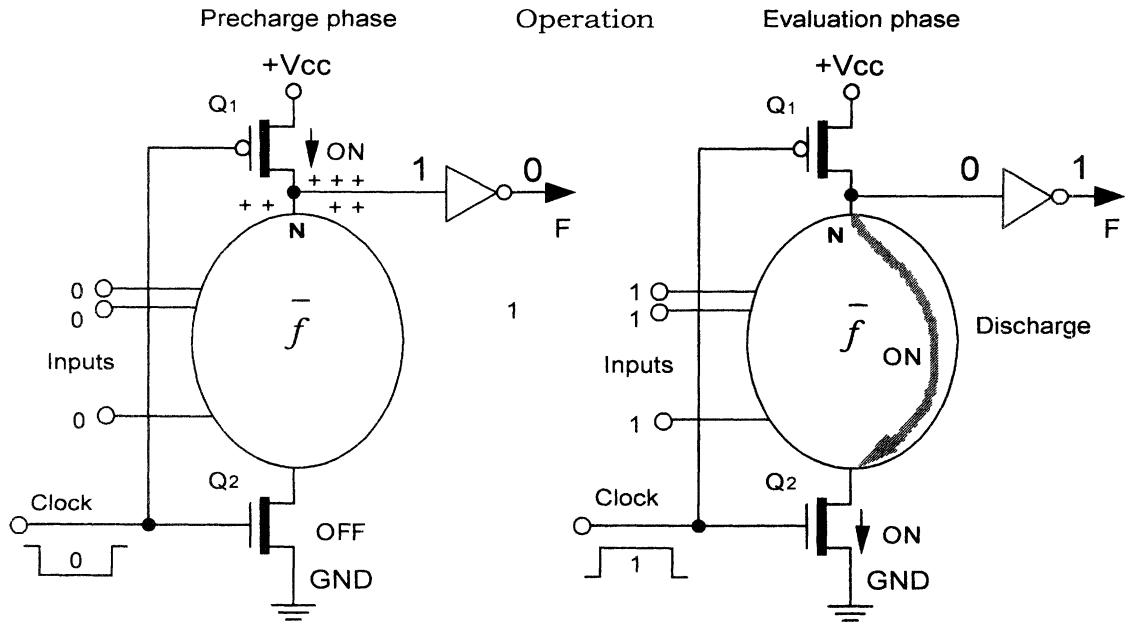


Fig. 1.2. CMOS Domino Logic [Krambeck, Lee, and Law].

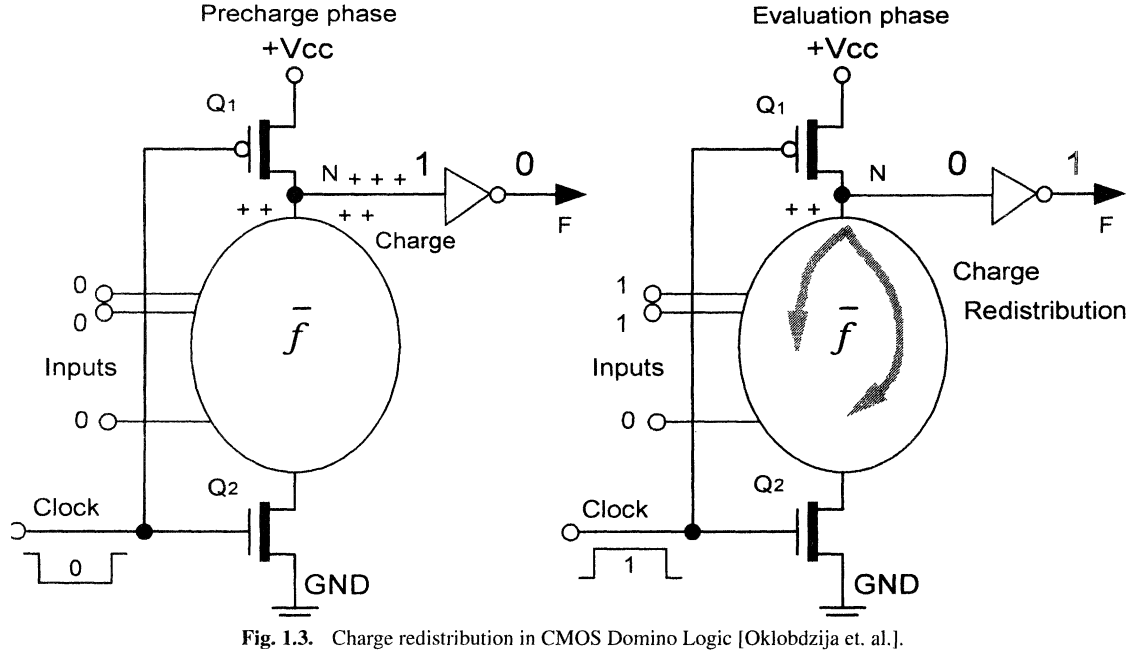


Fig. 1.3. Charge redistribution in CMOS Domino Logic [Oklobdzija et. al.].

that are in excess of the total number of transistors used by regular CMOS. In terms of size, CVSL is not larger than regular CMOS, although this is dependent on the complexity of the logic function implemented. Certainly, an operation involving only the n-type transistor switching function is faster. Furthermore, Heller, Griffin, Davis, and Thoma showed that implementation of f and $\bar{f}$ does not necessarily mean duplication. Therefore a number of transistors in the switching functions f and $\bar{f}$ can be shared, setting duplication only as an upper limit. Creation of CVSL circuits was supported by an automated synthesis tool (one of the first of that kind) that would create optimal and shared n-type transistor switching functions f and $\bar{f}$ as shown in the example in Fig. 1.5*a*.

The second type of the new CMOS logic, *dynamic CVSL*, is a clocked version of static CVSL, and in essence it represents a dual-output CMOS Domino Logic. Transistor *Qf* is added to ensure more robust operation of the dynamic CVSL circuit. If charge redistribution occurs, resulting in the loss of charge at the inverter's input, the output voltage will drop toward zero. This will in turn activate the transistor *Qf* and the charge will be restored, pulling the output back to logic one. In order to discharge this node, the pass-transistor network

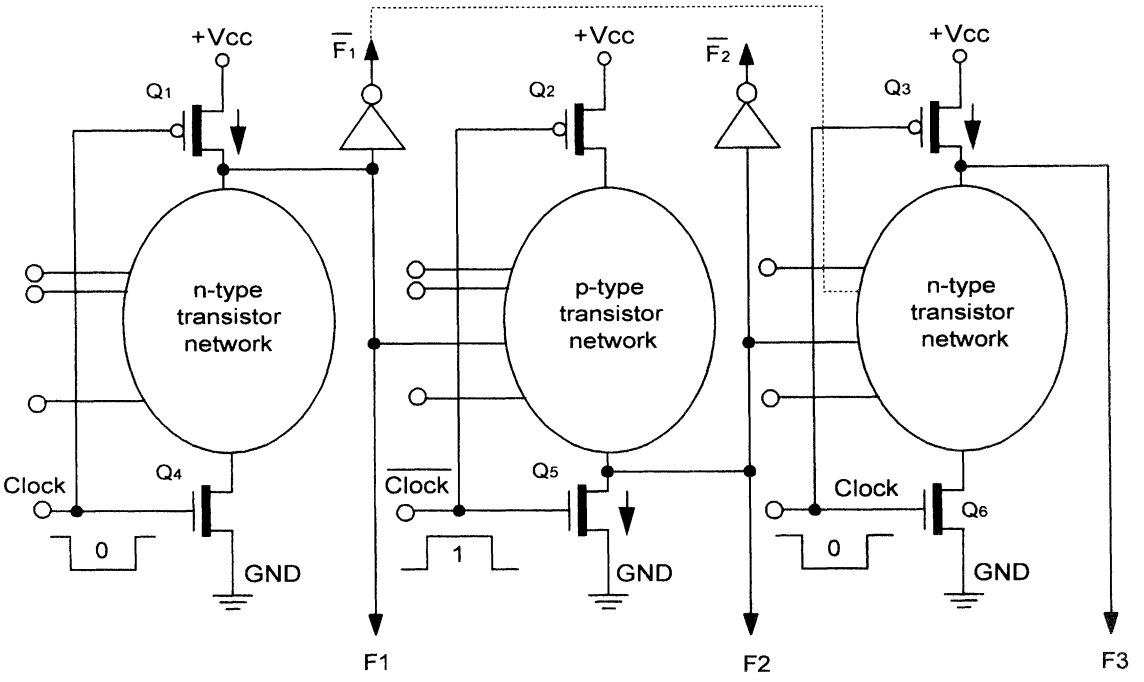


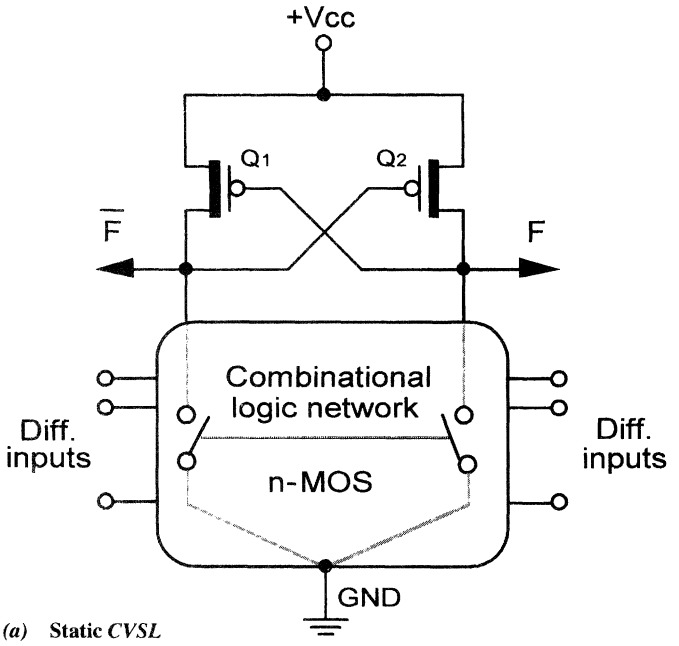
Fig. 1.4. Operation of NORA logic [Goncalves, DeMan].

implementing the function $\bar{f}$ has to overcome the transistor *Qf*. *Qf* is made to be a weak transistor; thus, its dimensions are smaller than the rest of the transistors implementing the switching function.

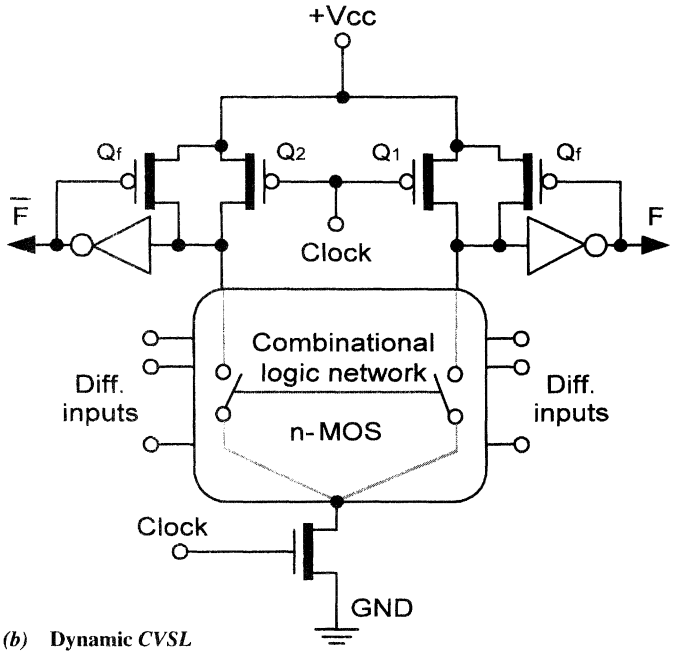
An interesting new concept is presented in the paper by Leo Pfennings on Differential Split-Level (DSL) CMOS Logic. This is essentially a very clever enhancement of CVSL, which allowed the use of one-micron devices at that time, and under a reduced voltage operation. The logic transistors were made of $L_{\text{eff}} = 1\mu$ devices, operating at 2.5 V, while the entire circuit operates on the commonly used $V_{DD} = 5$ V. The logic swing of the interconnection lines was reduced to 2.5 V, allowing for a subnanosecond speed (in 1985). The essence of DSL logic can be best understood from Fig.1.6. The difference between DSL and CVSL is in introduction of the two n-MOS transistors Q_3 and Q_4 between the load transistors Q_1 and Q_2 acting as source followers. Therefore, the impedance seen from the side of the logic is low, permitting a fast change of state, while their output resistance is high, allowing for the use of small devices Q_1 and Q_2 representing the *load* transistors. This enables a fast change of the logic state, which is further facilitated by the reduced voltage swing of 1/2 V_{DD}. The load transistors Q_1 and Q_2 are not completely turned off because their gates are being driven from the lower voltage potential. This makes an active loop capable of switching the state in a rapid manner. However, it also represents a power problem because DSL contains a static power component.

A comparative analysis of conventional CMOS versus Differential Cascode Voltage Switch Logic (which includes CVSL, DSL, and NORA), both static and dynamic, is presented in the paper by Chu and Pulfrey. The analysis is somewhat limited because it involves a section of a full adder only. Their conclusion is that DCVS logic is faster than conventional CMOS, although this advantage is counterbalanced by a somewhat larger circuit and more active power consumption. The fastest logic technique seems to be DSL, but the problems with DSL involve static power dissipation and increased sensitivity on circuit parameters in order to make the operation reliable. In dynamic operation, differential logic seems to have a speed advantage over single-ended logic.

Comparative studies similar to that of Chu and Pulfrey were conducted extensively at IBM in the early 1980s in order to reach an important decision regarding which CMOS family should be adopted for future products. These studies involved several large pieces of the control logic besides the parts from the data path as test examples. The findings, which were not published, were in some agreement with those of Chu and Pulfrey. Although they showed that CVSL has an advantage over regular CMOS, this advantage was not sufficient to justify extensive changes in the computer-aided design tools and methodology, test generation in particular. Therefore, regular CMOS was chosen, but the real deciding factor was the status of the existing CAD tools, not the performance of the logic families compared. In spite of its advantages, CVSL failed to become a technology of the future. The choice is often influenced by other factors, such as CAD and testability as was the case. Those issues are covered in later chapters of this book.



(a) **Static *CVSL***



(b) **Dynamic *CVSL***

Fig. 1.5. *CVSL* Circuit. [Heller et. al.].

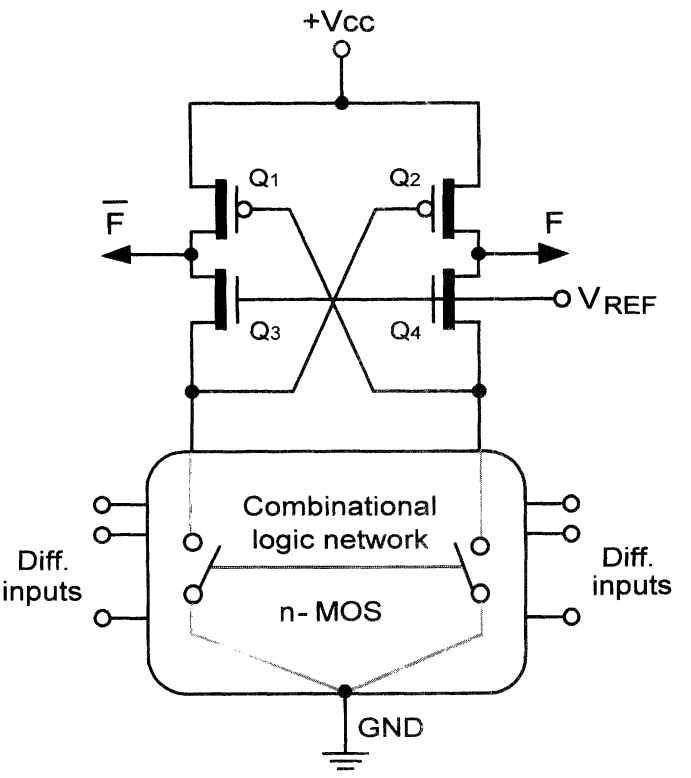


Fig. 1.6. DSL logic circuitm [Pfennings et al.].

Pass-Transistor Circuits

As the technology improved and started moving into the deep submicron region, use of regular CMOS reached its limits. The problems associated with ever-increasing requirements for performance and a higher level of integration required that other types of logic be examined. Extensive consideration was given to the use of pass-transistors' design style resurrected in several recent papers. Several experimental prototypes were built at the industrial research institutions that possessed the state-of-the-art technology.

This section begins with a relatively old paper by Whitaker, published in 1983, which shows how the logic functions can be efficiently and optimally built using pass-transistor logic. This paper contains some fundamental pass transistor building blocks that became very popular in VLSI design practices. In addition it established a methodology for synthesis of pass-transistor functions and presents a modified Karnaugh map that employs the pass variables, as well, aside from logic zeroes and ones.

In 1990, researchers from Hitachi Research Center in Japan published the structure known as Complementary Pass-Transistor Logic (CPL). The CPL structure was significant because it brought back the pass-transistor efficiency in implementation of logic circuits. The logic function built from the pass-transistors not only efficiently utilized the space on silicon, but also resulted in a very fast logic, which is also characterized by low-power consumption.

Yano and his co-workers went one step further than CVSL by decoupling the p-transistor latch combination and replacing it with two separate inverters. The logic is still dual-ended, which means that both the function f and its complement $\bar{f}$ are present at the outputs; in addition, the logic was built from the pass-transistor networks, as shown in Fig. 1.7. If we are to implement an AND gate, a NAND output will be readily available. There-

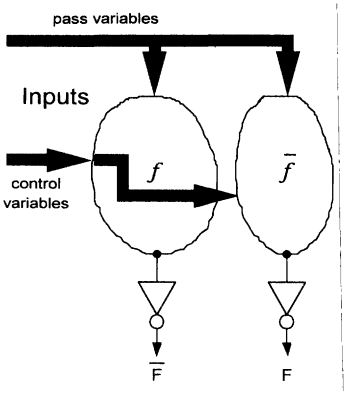


Fig. 1.7. CPL logic structure [Yano et al.].

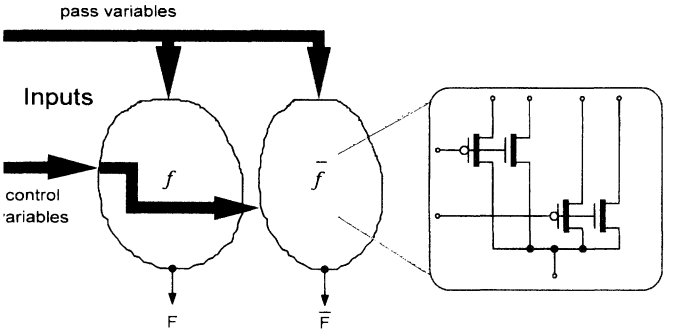


Fig. 1.8. DPL logic structure [Ohkubo].

fore, complementation consists of the proper choice of the signals only, given that both polarities are available. A family of gates is implemented in this fashion, including the XOR/XNOR combination as well as the multiplexer.

CPL logic proved to be not only very efficient but also very fast, yielding an 3.8 nS 16 × 16-b multiplier in double metal 0.5 μ CMOS technology. However, CPL suffers from one problem. When passed through a series of pass-transistors, the signal voltage is degraded by one V_T (*threshold voltage drop*). This brings both transistors in the output inverter to the conducting region, causing static current to flow from V_{cc} to GND and thus resulting in static power dissipation. To alleviate this problem, Hitachi researchers used two types of transistors: *logic transistors* (with $V_T = 0$ V) and *inverter transistors* (with $V_T = 0.4$ V and -0.4 V). Although this reduced static power dissipation and delay, it increased the process complexity and the sensitivity to noise. In the new version of CPL, the problem of the "threshold drop" was alleviated by using a special type of inverter which has the ability to restore the voltage level to its full potential.

Another pass-transistor logic, which evolved from the same group of researchers, is Double Pass-transistor Logic (DPL), Shown in Fig. 1.8. DPL was originally developed to overcome the threshold drop problems of CPL and to provide an alternative pass-transistor logic. In creating the switching network *f*, DPL uses both n-MOS and p-MOS transistors in parallel. This eliminates the problem of the threshold drop, and the use of inverters after each logic block is not necessary, thus enhancing the speed of DPL. Hitachi has shown two very fast implementations using DPL: one a 1.5 nS 32-b ALU and the other a 4.4 nS 54 × 54-b parallel multiplier.

Oklobdzija and Duchene have taken DPL a step further in developing the logic family, which they call DVL (Dual Value Logic). The new logic family was obtained from DPL by eliminating the redundant branches and rearrangement of signals. These simplifications still preserve full-swing operation of DPL and improve its speed. The speed improvement is a direct result of eliminating of one branch containing one transistor. This minimizes the capacitive load and the number of inputs applied to the previous gate.

The new logic family is achieved in three steps:

a. Elimination of redundant branches in DPL
b. Elimination of branches via signal rearrangement
c. Combination of (a) and (b) using two faster halves

The process is illustrated in Fig. 1.9 *a, b, c.*

A faster half was chosen from (a) and (b), resulting in a complete gate with both output polarities (c). Fortunately (a) produces a faster NAND, while (b) produces a faster AND, which makes a complete gate shown in Fig. 1.9c. The resulting DVL gate contains a total of six transistors (three *p*-transistors and three *n*-transistors) compared to four transistors of each type in DPL. There is a total of 9 inputs in DVL versus 12 in DPL, resulting in a smaller capacitive load of DVL gates. Of those inputs, three are connected to the transistor source and six to the gate (three to *p*-type and three to *n*-type). (In DPL four are connected to the source, four to *p*-type, and four to *n*-type transistors.) The total DVL gate area (taking resizing into account) is only 5 percent larger as compared to DPL. The speed advantage is 20 percent in favor of DVL.

The comparison between NAND/AND DPL gate and NAND/AND DVL shows

- 20 percent speed improvement, utilizing 75 percent of the transistors used in DPL.
- 25 percent fewer connections and wires as compared to a DPL gate.

A similar method is used to build the NOR/OR gates.

Further development of the differential CMOS family is presented in the paper by Lai and Hwang. They introduced pass-transistor logic in the CVS logic tree in order to eliminate the problem of current spikes. Lai and Hwang modified CVSL by using pass-variables instead of the ground connection at the opposite end of the pass-transistor block implementing the function *f*, while leaving the cross-coupled *p*-transistor latch at the opposite end.

The resulting logic (CVSL-PG) showed better performance than that of CVSL. This was demonstrated by implementation of a 2 nS 64-bit adder in 0.5 μ CMOS technology.

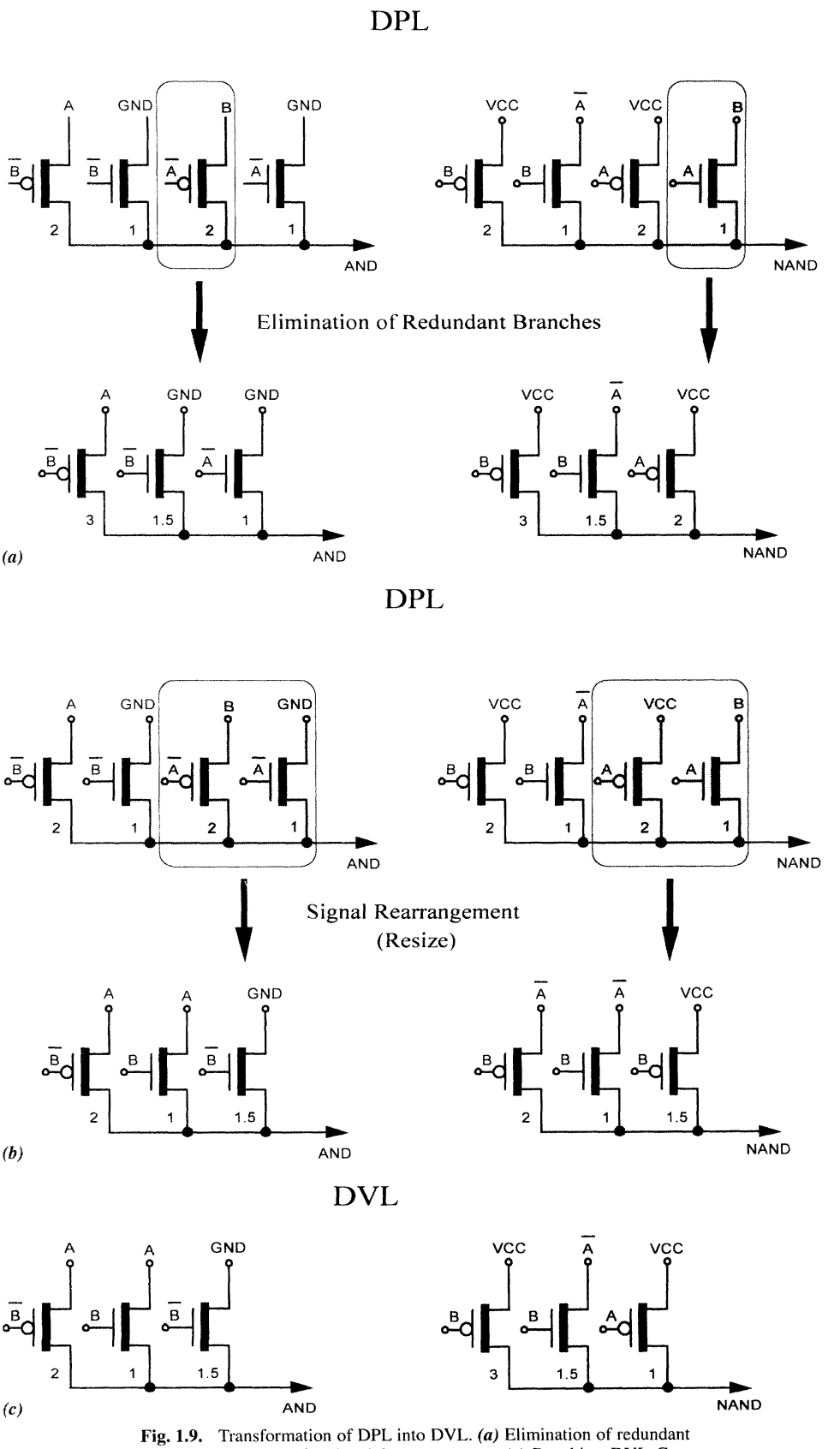


Fig. 1.9. Transformation of DPL into DVL. ***(a)*** Elimination of redundant branches, ***(b)*** Signal Rearrangement, ***(c)*** Resulting DVL Gate [Oklobdzija, Duchene].

Researchers from Toshiba Corporation developed their version of differential CMOS pass-transistor logic which does not suffer from degraded pull-down performance which they named Swing Restored Pass-Transistor Logic (SRPL). In SRPL the generic gate consists of a pass-transistor logic constructed of n-MOS transistors (similar to CPL) and a latch-type swing restoring circuit consisting of two cross-coupled CMOS inverters (Fig. 1.10). The n-MOS transistor logic network implements any Boolean logic function, while the complementary outputs of the pass-transistor logic are restored to full swing by the cross-coupled combination at the circuit output. In this way, SRPL solves a major problem associated with CPL logic. However, the input variable may be connected to a long chain through several gates, thus making the total output capacitance of the circuit quite large. Toshiba has built an experimental MAC (Multiply Accumulator) in a 0.4 μ CMOS technology achieving a 150 MHz speed at 3.3 V supply voltage.

Comparisons of full adder circuits implemented with CMOS, CPL, DPL, DCVSPG, and SRPL showed CPL to be the fastest, followed by SRPL and DCVSPG logic. However, SRPL had the best power-delay product, which amounted to 21 percent of that of CMOS.

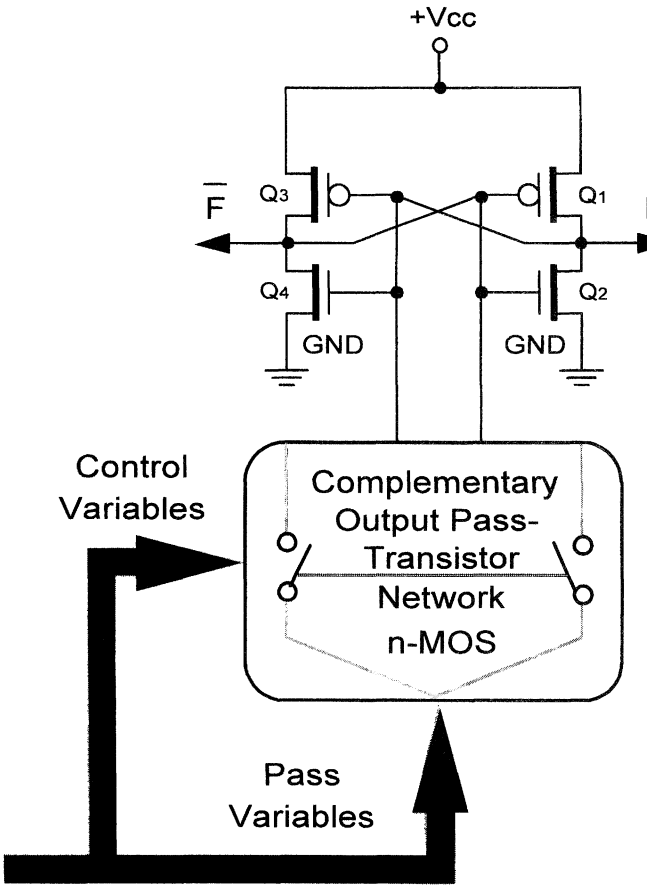
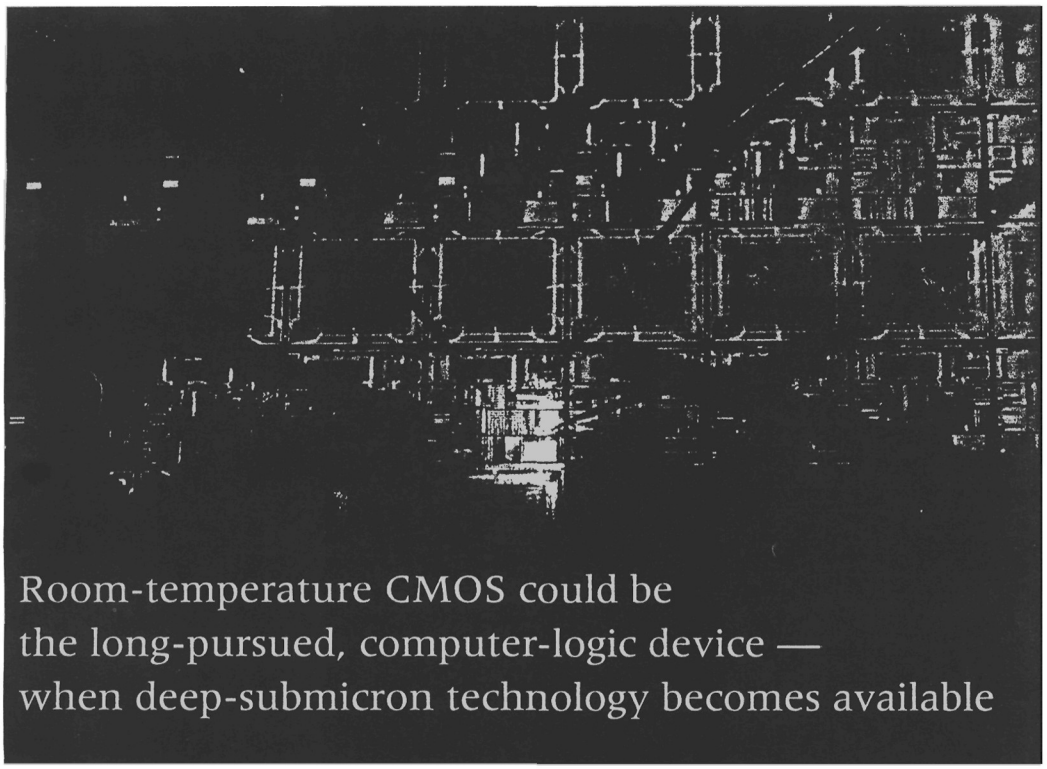


Fig. 1.10. Generic SRPL Gate [Parameswar et al.].

Deep-Submicron CMOS Warms up to High-Speed Logic*

AKIRA MASAKI



L. Manning/Westlight

Silicon bipolar devices, particularly those in emitter-coupled logic (ECL), have been accepted as the most useful of the various high-speed technologies since ICs were first introduced into computers. But the possibilities for improving ECL will eventually reach their limits. In anticipation of that time, many alternative technologies have been proposed, including low-temperature CMOS, GaAs, HEMT, and even Josephson junction devices. It is difficult to compare these technologies and predict which technology will be ECL's successor.

Despite the predictive difficulties, it will be indicated in this article that room-temperature CMOS, which heretofore has not been widely recognized as producing very-high-speed devices, can be the

*This article is based on "Possibilities of CMOS Mainframe and its Impact on Technology R&D," an invited paper presented by the author at the 1991 VLSI Symposium on Technology.

Reprinted from *IEEE Circuits and Devices Magazine*, pp. 18-24, November 1992.

Table 1 Logic Function Capability (4-bit ALU)			
	CMOS (NAND)	ECL	Ratio
Total circuit count	98	57	1.7
Circuit stages in signal paths (avg.)	6.1	2.9	2.1

long-pursued post-ECL technology. If it becomes possible to implement high-speed computer logic with room-temperature CMOS, we will no longer need to worry about comparing technologies. Even if equivalent system-performance can be obtained by other technologies, CMOS will win industrial favor because the advanced semiconductor technology established through development of DRAM is directly applicable to CMOS. In addition, the technological compatibility with personal computers, workstations, and small computers benefits all aspects of R&D in high-speed CMOS computers. The ability to scale CMOS down will also allow circuit designs to be inherited through design generations. Last but not least, CMOS's low power dissipation cannot be matched by other technologies, even when applied to high-speed computer logic. As a consequence of the preceding factors, the integration scale achievable with CMOS is the highest of any of the candidate technologies.

Integrating High-Speed ICs

The scale of integration is often regarded as a non-critical parameter for high-speed computer logic because circuit speed is the primary concern. But integration scale has steadily increased by ten times every five years over the past quarter century (Fig. 1). This rate of increase is the same as that for DRAMs, which is usually stated as four times every three years. This clearly tells us that increases in integration scale have been the source of decreasing cost-performance ratio in computers.

It is not yet known whether this trend will, or should, continue. If it does continue, the gate count per chip will reach two million in the year 2000. Novel high-speed devices will not be able to cope with this complexity in a practical way, and it will not easy for conventional ECL either. Even if individual gate-circuit power is as little as 2 mW, a 2 megagate chip will consume 4 kW — an unmanageably large power.

Fortunately, silicon technology itself should be capable of realizing the required scale of integration since there is no sign that the rate of increase of DRAM-chip integration is slowing. Therefore, in addition to meeting performance requirements, we should make a conscious effort to increase the integration of logic chips.

CMOS can realize the integration but its speed has been insufficient, at least at room temperature. Recently, a fairly fast circuit speed was obtained by decreasing the MOSFETs gate-length to approximately 0.2 μm [1,2]. However, it is not yet clear that very-high-speed computer logic can be made with such CMOS devices. By referring to extensive theoretical developments, we hope to shed some light on this question throughout the remainder of this article [3-9].

Estimating Logic Performance

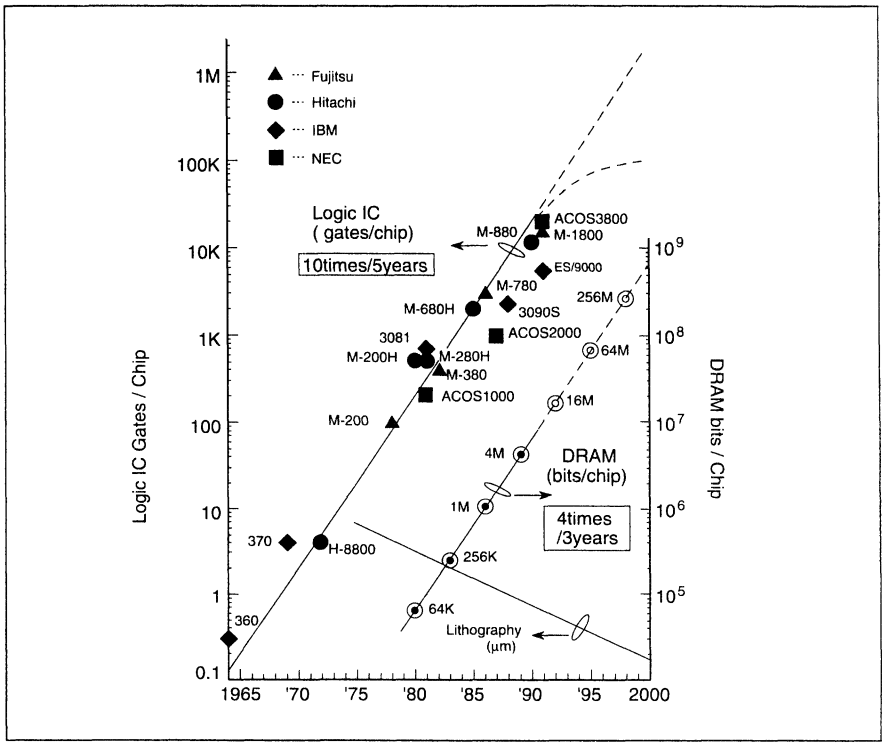
Device performance is usually demonstrated with data from a very lightly loaded ring oscillator, but such data does not directly relate to system performance. While most experts agree when they estimate memory performance, estimating the performance of logic systems has always been much more controversial. Various factors should be considered in such estimates, including the logic-function capabilities of basic circuits, the wire length of logic signal nets, and the power of CMOS circuits in system environments.

Logic Function Capability

In assessing the differences in logic-function capabilities among circuits [3,5,6], the essential questions are:

- How many circuits are required in all?
- In each technology, how many circuit stages are required for the critical signal path in a specific functional block?

The real potential of a technology can be grasped only after the second question is fully answered, because it is the power and delay of the functional block, not the unit circuit, that should be minimized. Unfortunately, it is difficult to obtain a general solution to this problem because the results of a design depend strongly upon the logical characteristics of the functional block, as well as upon the differences in the individual abilities of designers. Nonetheless, differences in the logic-func-



1. Although the integration scale of very-high-speed computer-logic chips is often regarded as non-critical, it has been increasing at the same rate as DRAMs for a quarter of a century.

tion capabilities of various circuits cannot be neglected.

We can demonstrate the point by comparing the total circuits and the average number of circuit stages in the signal paths of carefully designed 4-bit arithmetic and logical units built with ECL and simple CMOS NAND (Table 1). The circuit count and number of circuit stages required to implement a specific logic function are smaller for ECL. Results obtained in various case studies [5,6] lead us to conclude that simple CMOS NAND requires roughly twice as many total circuits and circuit stages as ECL.

Wire Length of Logic Signal Nets

Delays caused by wire capacitance significantly affect circuit performance. Estimating the wire length of logic signal nets is therefore indispensable when evaluating performance in system environments.

A simplified structure for a logic-circuit cell array, which could be an LSI chip, a module substrate, or a printed circuit board, provides a basis for wire-length estimates (Fig. 2). The wire length of the signal net, L_w, is $n_{pp} \times l_{pp} \times p$, where n_{pp} is the number of pin-to-pin (or terminal-to-terminal) wires per signal net, and l_{pp} is the terminal-to-terminal wire length expressed in terms of the average center-to-center spacing of the cells, p.

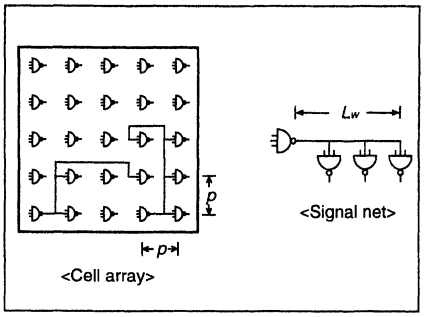
An equation for estimating l_{pp} [10], which is applicable to two-dimensional square arrays, was derived from the Rent's Rule equation [11]. By extending this work, we obtained equations for estimating various types of two- and three-dimensional packaging structures [7]. One of these is extremely useful:

$$l_{pp} = \frac{1 - 2^{2(r-1)}}{6\,(1 - 2^{2(r-1)L})} \cdot \left\{ \frac{7\,(2^{(2r-1)L} - 1}{2^{2r-1} - 1} - \frac{1 - 2^{(2r-3)L}}{1 - 2^{2r-3}} \right\}$$

where L is the base 2 logarithm of the number of circuit cells in the x and y directions, and r is the exponential coefficient in the Rent's Rule equation.

Power Consumption of CMOS Circuits

A CMOS circuit consumes energy only during switching. Estimating its power consumption in a realistic system environment therefore requires us to obtain the circuit's switching frequency in actual systems. To help us analyze the problem, let's introduce a quantity (k) that is defined as the average switching period (T) of the circuits in a system, divided by the circuit delay in the system (t_{sd}) [4,5]. The definition of k is similar to that of the CMOS switching factor defined in [12].



2. The signal net model for a logic-circuit cell array provides a convenient basis for wire-length estimates.

The value of k is independent of the hardware technology; it depends solely upon the logical structure of the system as long as the performance potential of the circuit is fully utilized. If sophisticated logic is used, k becomes smaller; that is, switching occurs more frequently.

There are several methods for obtaining the value of k [4,5]:

(1) Investigate the logical structure of the system.

(2) Measure the switching frequency of logic signals in system operation.

(3) Measure power dissipation in systems built with CMOS.

(4) Count the number of times the circuits switch by simulating the logical operation of the system.

Experimental values obtained with methods 2 and 3 are in good agreement with the estimated values from methods 1 and 4 (Table 2). We can therefore conclude that k is within the range of 20 to 200 for most computer systems. We will assume k to be 40.

Load-driving Capability and Wire Delay

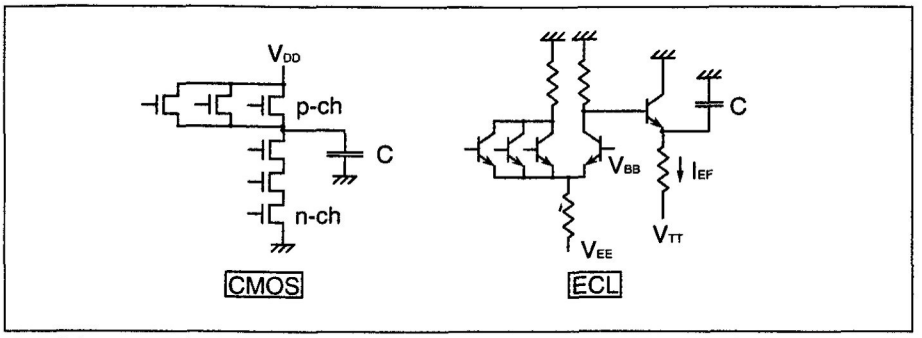
Load-driving capability is one of the most important characteristics of computer-logic

Table 2: K Values (T/tsd)

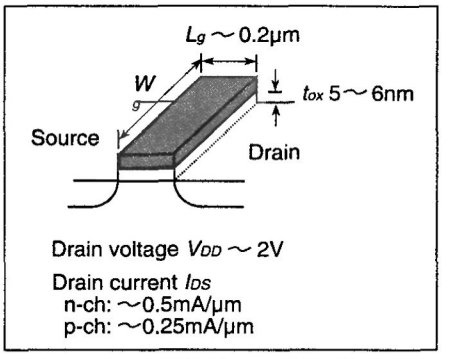
	Minicomputer		Large-scale computer
	A	B	
(1) Analysis of logic Structure	180	120	60 ~ 20
(2) Measurement of switching frequency	200		
(3) Measurement of power of CMOS computer		170 ~130	
(4) Logic Simulation			80 ~ 30

Table 3. Device Parameters for Very-High-Performance CMOS Logic

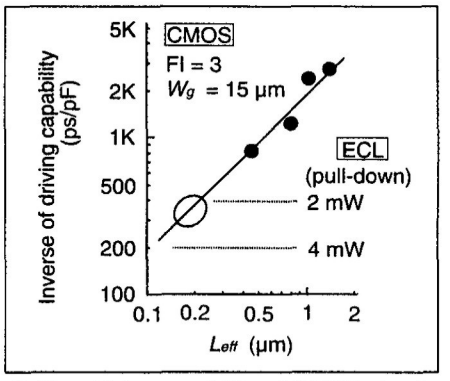
	Intrablock 3NAND	Interblock buffer
Drain voltage V_{DD}	2V	
Gate length L_g	0.2 μm	
Gate oxide thickness t_{ox}	5 ~ 6 nm	
Gate width W_g	15 μm	75 μm
Drain current I_{DS} (n-ch)	7 mA	35 mA
(p-ch)	3.5 mA	18 mA
Input capacitance C_{in}	0.05 pF	0.25 pF
Wire capacitance	0.2 pF/mm	0.2 pF/mm
Wire Resistance	100Ω/mm	6Ω/mm
Switching energy (circuit)	0.5 pJ	2.5 pJ
(load)	3.0 pJ/pF	3.0 pJ/pF
Driving capability	400 ps/pF	60 ps/pF
Circuit delay (F0=1, C_w=0 pF)	80 ps	40 ps
(F0=3, C_w=0.2 pF)	200 ps	



3. Driving capability is directly proportional to signal current divided by the signal voltage. CMOS's signal voltage is as large as 5 V and its driving current is small; ECL's signal voltage is as small as 0.5 V and the technology can drive a large current. The resulting load-driving capability is one reason for ECL's popularity.



4. In a deep-submicron CMOS device structure, drain current is increased by substantially decreasing gate length, L_g, and gate-oxide thickness, t_{ox}, thus increasing driving capability.



5. The driving capability of CMOS circuits has been improving steadily. The estimated driving capability of the 0.2-μm-gate CMOS is derived from the device characteristics for deep-submicron CMOS reported so far [1,2].

circuits and devices. ECL has been favored because of its large driving capability; conventional CMOS is weak in this area.

Driving capability is directly proportional to signal current divided by the signal voltage. CMOS's signal voltage is as large as 5 V and its driving current is small. On the other hand, ECL can drive a large current and its signal voltage is as small as 0.5 V (Fig. 3).

But things are changing. The drain current of a CMOS device can be increased by decreasing gate length L_g and gate-oxide thickness t_{ox} (Fig. 4). A fairly large drain current has been reported at a drain voltage as low as 2 V when gate length is decreased to approximately 0.2 μm [1,2]. Such "deep-submicron" devices represent the latest in a steady stream of improvements in the driving capability of CMOS circuits (Fig. 5).

The driving capability of ECL circuits, on the other hand, is mainly determined by the current flowing through the emitter-follower pull-down resistor when the output signal changes from high to low. Therefore, the circuit's driving capability is determined primarily by power dissipation and cannot benefit from device miniaturization. Since ECL circuit power cannot be increased in the future, the driving capability of CMOS will be equivalent to that of ECL when deep-submicron devices become available. The driving capability of ECL may still be improved by adopting active pull-down circuit techniques, but CMOS is attractive because its device structure and circuit configuration are very simple.

The delay caused by long signal nets on printed circuit boards and module substrates often determines system performance. Driving such long nets has been a strong point of ECL. Usually, such signal nets are treated as controlled-impedance-terminated (CIT) nets, in which case signals reach the far end of the net with minimum delay. ECL is suitable for driving CIT nets.

In CMOS VLSI, long nets are implemented on the chip. Wire resistance is large, and resistance-capacitance (RC) nets are used. Delay per unit length of RC nets is larger than that of CIT nets, but wire length is shorter for CMOS VLSI so total delay can be competitive.

Case Study

Device Requirements and VLSI Chip Model

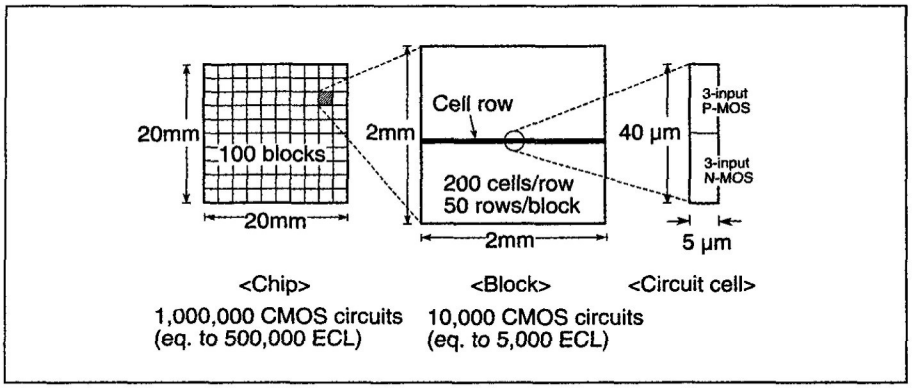
Based on our experience and the data reported so far, we expect certain basic characteristics for 0.2-μm gate-length, room-temperature CMOS (Table 3)[1,2]. To evaluate the possibilities of CMOS, we assume a particular VLSI chip model (Fig. 6). In actuality, almost half of the chip area will be occupied by memory circuits, but in this case study we assume that the chip consists of logic only. As a result, we are considering the worst-case wire length.

In the model, we assume a CMOS chip area of 20 mm x 20 mm, a reasonable assumption considering that a 15 mm x 15 mm ECL chip is used today [13]. The chip is hypothetically divided into 100 identical blocks. Actually, the block size will not be identical, especially when standard-cell design methodology is applied, but the assumption is valid for evaluating performance and chip area. Ten thousand CMOS circuits can be integrated in a 2-mm x 2-mm block. This leaves sufficient area for interblock buffers and chip input-output (I/O) circuits, because a 3-input CMOS NAND using 15-μm-gate-width FETs can

Table 4: Packaging Density of Future CMOS* VLSI Chips and Modern** ECL Modules

		Packaging density	Wire Length
1st level	CMOS (block)	1,250/mm²	
	ECL (chip)	100/mm²	
	Ratio	12.5	1/3.5
2nd level	CMOS (chip)	125,000/cm²	
	ECL (module)	4,000/cm²	
	Ratio	30	1/5.5

*deep-submicron
**early 1990's



6. A chip model for future CMOS VLSI computer logic.

be placed in a 40 μm x 5 μm area when 0.3-μm lithography is used. We assume the gate width of 15 μm, which would be unreasonably large for a microprocessor design, to obtain a large driving capability equivalent to ECL.

In our model, one million CMOS circuits with large driving capability are implemented on a chip. The chip is equivalent to 500,000 ECL circuits in terms of its logic-function capabilities. Implementing the same logic by the most modern ECL technology would result in a much larger module (Fig. 7). Let's compare the packaging density and wire length of the ECL module with the CMOS VLSI (Table 4). Since the integration scale of a modern ECL chip is 5,000 to 20,000 gates [14], 25 to 100 ECL chips would be necessary.

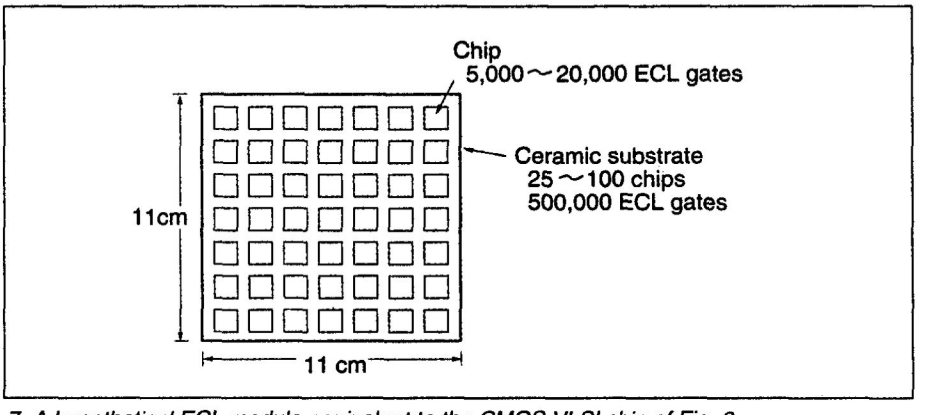
One CMOS block on the VLSI chip is equivalent to one ECL chip in terms of gate count. Since the density of the modern ECL is about 100 gate/mm^2 [15-17], the CMOS is 12.5 times denser. Therefore, the wire length of the CMOS is 1/3.5 that used in ECL. One CMOS chip is equivalent to one ECL module. Since the density of the modern ECL module is 3,500-4,000 gate/cm^2 [14], an 11 cm x 11 cm ceramic substrate is necessary. The CMOS is 30 times denser, and its wire length is 1/5.5 that used in ECL.

The estimated wire length of the CMOS VLSI chip is shown in Table 5 using the equations reported in [7]. A Rent's Rule coefficient of r = 2/3 is used. Two cases are shown for the intrablock nets. The first case (left column) assumes that CMOS circuits are uniformly distributed in the block, and applies to estimating total wire length. The second case (right column) applies when four CMOS circuits are used as a cluster to implement one ECL function, and applies to estimating critical path delay.

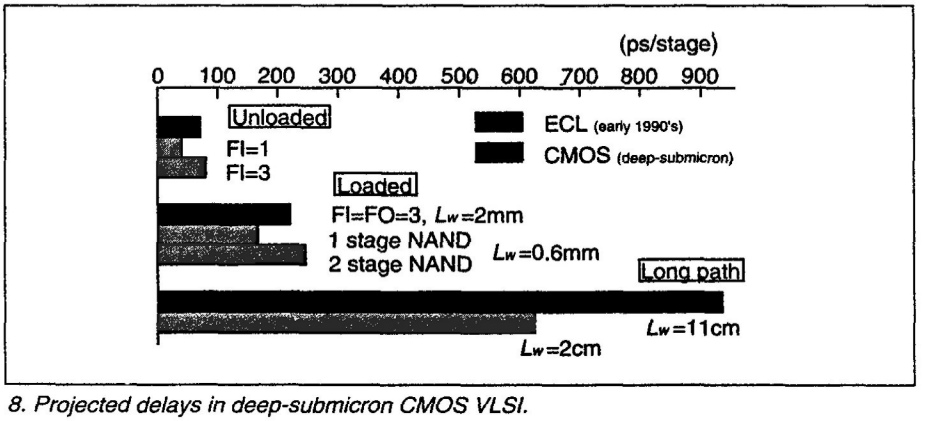
If the average number of terminal-to-terminal wires per signal net, n_{pp}, is 2, that is, FO = 2, wire length per signal net is about 240 μm for intrablock and 9 mm for interblock. Therefore, the total wire length in a block is about 2.4 m since each block has 10,000 signal nets. If wire channel pitch is 1.5 μm (that is, width = spacing = 0.75 μm), two signal layers are required for the intrablock wiring because approximately 50 percent of the total channel length is usable. As for the interblock wiring, the total wire length in a chip is about 100 m since the number of output terminals of a 5,000-ECL equivalent-circuit block is estimated to be 100 - 110. Since the interblock wire resistance causes a significant delay, a wire channel pitch as large as 6 μm is assumed for estimating performance. In this case, three signal layers are required for the interblock wiring. The required number of metal layers is at least six, including power and ground layers. This requirement is not easy to meet, but is not unrealistic for the future since 4-metal-layer chips are already used today [14].

Delay and Power Consumption of the CMOS VLSI Chip

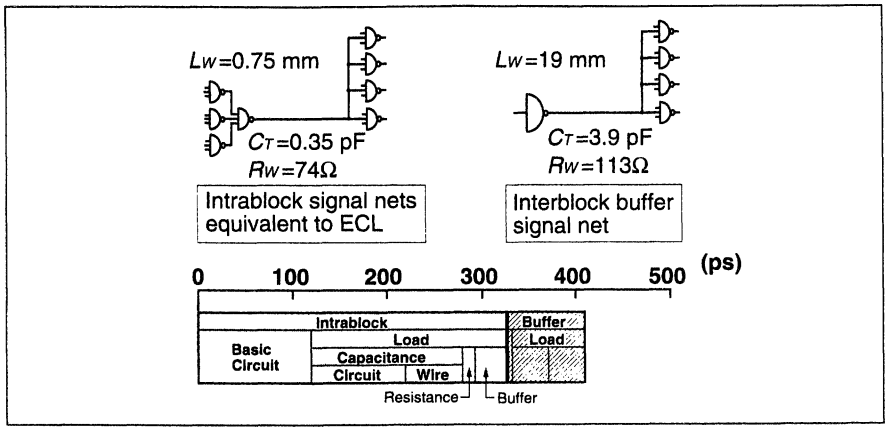
Let's compare the delays of the modern ECL and the deep-submicron CMOS (Fig. 8). The lightly-loaded circuit delay of ECL is 70 ps [14]. The corresponding delay of CMOS is 40 ps for a 1-input inverter and 80 ps for a 3-input NAND gate, as shown in Table 3. The delay of the ECL with 3 fan-outs and 2-mm wire is 220 ps [17]. The corresponding wire length for CMOS is 0.6 mm (from Table 4). Using Table 3, the delay of



7. A hypothetical ECL module equivalent to the CMOS VLSI chip of Fig. 6.



8. Projected delays in deep-submicron CMOS VLSI.



9. Critical-path delay of the CMOS VLSI.

the CMOS is estimated to be 170 ps for a 1-stage NAND. If a 2-stage circuit is needed to obtain logic-function capability equivalent to that of an ECL, the delay is 250 ps.

In the case of ECL, if a signal net is as long as one side of the module substrate, the transmission delay is about 800 ps because the dielectric constant of the substrate is 5.7 - 5.9 [14]. Assuming that the delay of a buffer circuit used for driving the wire is twice the lightly loaded circuit delay, the total delay of the net would be about 940 ps. The corresponding delay of the CMOS would be about 620 ps, assuming a buffer circuit consisting of two stages of inverters for driving the 2-cm wire. The gate widths of the transistors are 15 μm for the first stage and 75 μm for the second stage. The delay of the buffer circuit is estimated as 140 ps. The wire and load capacitance C_T would cause a delay of 240 ps, and the delay caused by wire resistance RW is estimated as 240 ps ($0.5\ R_W C_T$).

Although a larger coefficient for the worst case should be assumed for CMOS than for ECL, delays of the CMOS are comparable to the ECL in the various paths in Fig. 8. These results indicate that system performance obtained by the deep-submicron CMOS will be roughly equivalent to the most modern ECL systems.

Various paths were used in the preceding comparison of ECL and CMOS delays, but a more comprehensive discussion of system performance requires a model of typical critical paths. Figure 9 shows one example of such models. The intrablock circuit model consists of two CMOS NAND stages with three inputs and four fan-outs for the second stage, which has associated wires. This is approximately equivalent to one ECL stage in critical paths. Total capacitance of this net is estimated to be 0.35 pF, using the wire length in Table 5 and the circuit characteristics in Table 3. The length of this wire is 0.75 mm and its resistance is 74 Ω. The net's total delay is about 290 ps.

Using interblock buffer circuits decreases the total delay in communicating with other blocks. The buffer circuit shown in Fig. 9 is a simple CMOS inverter with 75-μm-gate-width FETs. Fan-outs of four and associated wires are assumed. Total capacitance of this net is estimated as 3.9 pF, using the wire length in Table 5 and the circuit characteristics in Table 3. The length of this wire is 19 mm and its resistance is 113 Ω. This net's total delay is about 490 ps. Since the input capacitance of this circuit is large and the wire connecting the circuit with an intrablock circuit could be as long as 1 mm, an additional delay of about 200 ps should be assumed when driving this circuit with an intrablock circuit. Thus, total delay for the buffer circuit is estimated as 690 ps.

To obtain CMOS delays corresponding to ECL critical paths, a portion of the buffer delay has to be added to the intrablock delay. From the data reported in [18] the number of ECL equivalent circuit stages is estimated to be about six, considering that the block is equivalent to 5,000 ECL circuits. To obtain the "system delay," one sixth of the buffer-circuit delay should be added to the intrablock circuit delay.

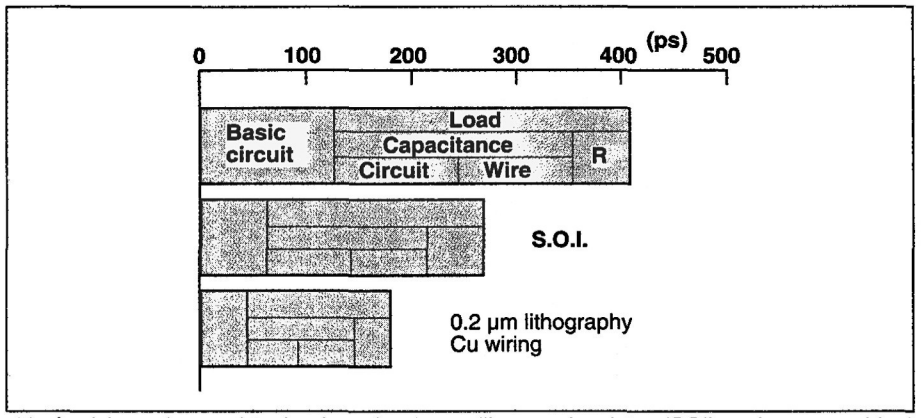
The delay is about 400 picoseconds under typical process parameters and system environments. Assuming a coefficient of 1.5 for the worst case, the value for estimating the system performance is about 600 ps. Since the chip integration is very large, almost all portions of the critical

Table 5. Estimated Wire Lengths – CMOS VLSI Chip

	Intrablock		Interblock
Circuits/cell	1	4	
Area (mm x mm)	2 x 2	2 x 2	20 x 20
No. of cells or blocks/area	200 x 50	50 x 50	10 x 10
Wire length *lpp* x *p* (μm)	121	189	4,660
Total wire length (mm)	2,415		97,860
Channel pitch (μm)	1.5		6.0
Number of layers required	2		3

Table 6. Estimated Device Parameters of the CMOS VLSI Chip

	Intrablock	Interblock
Wire length/net (mm)	0.24	9.3
Fan-outs/net	2	2
Capacitance/net (pF)		
Wire	0.048	1.86
Load Circuits	0.10	0.10
Net Total	0.148	1.96
Switching energy/net (pJ)	0.94	8.4
/chip (μJ)	0.94	0.088
Avg. Switching Period (ns)	25	25
Power/net (μW)	38	340
/chip (W)	38	3.6



10. Applying advanced technology (such as silicon-on-insulator (SOI) and copper wiring) to deep-micron CMOS VLSI would substantially reduce delays.

paths can be implemented on one chip. The delay we've obtained can thus be directly compared to the system delay of ECL machines, which consists of on-chip circuit and loading delay, chip I/O delay, and wire delay on module substrates and/or printed circuit boards.

The switching period is estimated to be about 25 ns, using k = 40 and the system delay a little larger than 600 ps. The power of the CMOS chip can be estimated using the switching period, the device and basic circuit characteristics in Table 3, and the wire lengths in Table 5. As shown in Table 6, the total power of the intrablock circuits and interblock buffers are 38 W and 4 W, respectively. Total chip power will be about 50 W, including chip I/O circuits. If equivalent logic is implemented by using 2-mW ECL circuits, the total power would be more than 20 times larger.

Discussion

If the deep-submicron device in Table 3 is realized, we will be able to obtain system performance from CMOS computer-logic circuits equivalent to today's most modern ECL systems. In addition CMOS has other possibilities.

The effects of applying advanced process and device technology are shown in Fig. 10. The uppermost bar shows the delay breakdown of the CMOS VLSI described above. The second bar shows the effects of applying fully depleted silicon on insulator devices. Better device performance is obtainable by realizing fully depleted SOI MOSFETs [19, 20]. If basic circuit speed is doubled (at x 1.5 original drain current) using the same level of lithography, the system delay decreases by one third. If further miniaturization is realized and low-resistance material such as copper is used for wiring, the delay will decrease to less than one half of the uppermost bar.

In the case of CMOS, system performance increases almost directly with improvements in device performance. This should provide strong motivation for developing advanced deep-submicron devices, including the technologies we've mentioned. The development of such devices should increase the possibility that room-temperature CMOS will become the long-pursued post-ECL high-speed technology. **CD**

Akira Masaki [SM] is with the Device Development Center, Hitachi, Ltd., Tokyo, Japan.

References

1. B. Davari et al., "A high performance 0.25 μm CMOS technology," *1988 International Electron Device Meeting Technical Digest*, pp. 56-59, December 1988.

2. M. Miyake, T. Kobayashi, and Y. Okazaki, "Subquarter-micrometer gate-length p-channel and n-channel MOSFET's with extremely shallow source-drain junctions," *IEEE Trans. on Electron Devices*, vol. 36, no .2, pp. 392-398, February 1989.

3. A. Masaki, Y. Harada, and T. Chiba, "200-gate ECL masterslice LSI," *1974 International Solid-State Circuits Conference Dig. Tech. Papers*, pp. 62-63, 230, February 1974.

4. A. Masaki *et al.*, "Comparison of MOS basic logic circuits," *IECE Trans.*, vol.58-D, no.7, pp. 397-404, July 1975.

5. A. Masaki and T. Chiba, "Design aspects of VLSI for computer logic," *IEEE Trans. Electron Devices*, vol. ED-29, no.4, pp. 751-756, April 1982.

6. A. Masaki et al., "Perspectives on hardware technologies for very-high-performance computers," *Proc. IEEE International Conference on Computer Design*, pp. 561-564, October 1984.

7. A. Masaki and M. Yamada, "Equations for estimating wire length in various types of 2-D and 3-D system packaging structures," *IEEE Transactions on Components, Hybrids, and Manufacturing Tech.*, vol. CHMT-10, no. 2, pp. 190-198, June 1987.

8. A. Masaki, "Electrical resistance as a limiting factor for high performance computer packaging," *IEEE Circuits and Devices Magazine*, vol. 5, no. 3, pp. 22-26, May 1989.

9. A. Masaki, "Possibilities of CMOS mainframe and its impact on technology R&D," *1991 VLSI Symposium on Technology Dig. Tech. Papers*, pp. 1-4, May 1991.

10. W.E. Donath, "Placement and average interconnection length of computer logic," *IEEE Trans. Circuits Syst.*, vol. CAS-26, no. 4, pp. 272-277, April 1979.

11. B.S. Landman and R.L. Russo, "On a pin versus block relationship for partitions of logic graphs," *IEEE Trans. Comput.*, vol. C-20, no. 12, pp. 1469-1479, December 1971.

12. P.W. Cook, D.L. Critchlow, and L.M. Terman, "Comparison of MOSFET logic circuits," *IEEE J. Solid-State Circuits*, vol. SC-8, no. 5, pp. 348-355, October 1973.

13. H. Tokuda et al., "A 100k-gate ECL standard-cell LSI with layout system," 1990 *International Solid-State Circuits Conference Dig. Tech. Papers*, pp. 94-95,272, February 1990.

14. Nikkei Microdevices, no. 65, pp. 145-151, November 1990.

15. Nikkei Microdevices, no. 64, p. 79, October 1990.

16. Nikkei Electronics, no. 509, pp. 111-113, September 17, 1990.

17. Nikkei Microdevices, no. 48, pp. 38-61, June 1989.

18. M. Nakagawa and M. Yamada, "A study on the relationship between integration scale and number of circuit stages in computer logic," 1981 IECE, paper 402, book 2, p.167.

19. D. Hisamoto et al., "A fully depleted lean-channel transistor (DELTA) — a novel vertical ultra thin SOI MOSFET —," 1989 *International Electron Device Meeting Technical Digest*, pp. 833-836, December 1989.

20. G. Shahidi *et al.*, "Fabrication of CMOS on ultrathin SOI obtained by epitaxial lateral overgrowth and chemical-mechanical polishing," *1990 International Electron Device Meeting Technical Digest*, pp. 587-590, December 1990.

High-Speed Compact Circuits with CMOS

R. H. KRAMBECK, MEMBER, IEEE, CHARLES M. LEE, AND HUNG-FAI STEPHEN LAW, MEMBER. IEEE

Abstract—Characteristics of various CMOS and NMOS circuit techniques are described, along with the shortcomings of each. Then a new circuit type, the CMOS domino circuit, will be described. This involves the connection of dynamic CMOS gates in such a way that a single clock edge can be used to turn on all gates in the circuit at once. As a result, complex clocking schemes are not needed and the full inherent speed of the dynamic gate can be utilized. The circuit is most valuable where gates are complex and have high fan-out such as in arithmetic units. Examples are shown of the use of domino circuits in an 8-bit ALU, where simulations indicate a speed advantage of 1.5 to 2 over traditional circuits, and in a 32-bit ALU where a worst case add in 124 ns was projected and a time less than 100 ns was achieved.

I. INTRODUCTION

THIS paper will describe some new design techniques which can substantially reduce area and increase speed for circuits made with CMOS technology. These techniques combine, in a unique way, the speed and power advantages of dynamic circuits with the stability and ease of use of static circuits.

In a fully complementary CMOS circuit the logic function of each gate is implemented twice. For example, a combinational gate that does the AND/OR invert (AOI) function for one 3-input AND, and one 2-input AND (32 AOI), is shown in Fig. 1. The five n-channel transistors have all the information needed to implement the function and so do the five p-channel transistors. The advantage of having both arrays is that except for the very brief period when the output or the inputs are making transitions no current flows and no power is consumed.

The problem with this fully complementary approach is that for complex gates of the type shown in Fig. 1, substantial amounts of area can be wasted. For example, the same function could be made with six transistors in static NMOS or pseudo-NMOS as shown in Fig. 2. (Pseudo-NMOS refers to a design technique which gives circuits identical to NMOS circuits except for the use of a p-channel transistor as the load instead of an n-channel transistor.)

As a result of the extra area and extra transistors, the capacitive load on gates of a fully complementary circuit are considerably higher than the loads on a pseudo-NMOS or NMOS circuit. Each output goes to both a p-channel and an n-channel transistor in every gate it drives. P-channels are generally twice the size of n-channels to obtain more balanced rise and fall times [1]. As a result, the total gate load on each output will be three times higher. Parasitics do not increase that much but overall capacitance is at least a factor of two higher.

Manuscript received March 10, 1981; revised November 6, 1981.
The authors are with Bell Laboratories, Murray Hill, NJ 07974.

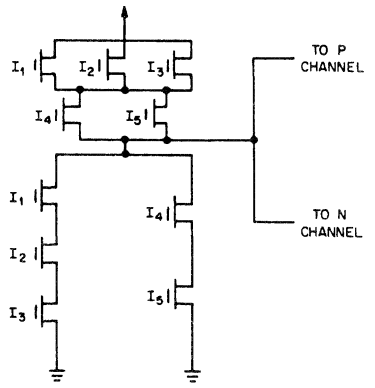


Fig. 1. Fully complementary MOS 32AOI gate. No static power but high-output capacitance and area.

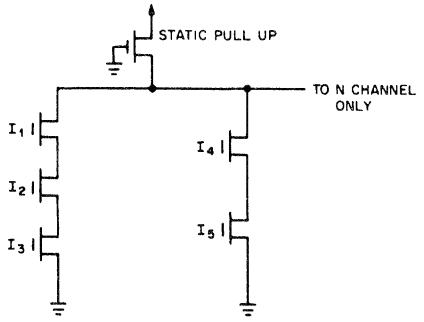


Fig. 2. Pseudo-NMOS 32AOI gate. Low output capacitance and area, but static pull-up current consumes power and slows pull-down.

It would appear from this that pseudo-NMOS or NMOS would be much faster than CMOS but this is not the case. The problem is that pull-up current always flows in the pseudo-NMOS circuit even if the gate is pulling down. This slows the pull-down. Making the pull-up current very small does not solve this problem because then the pull-up would be very slow. In fact minimization of the sum of rise time and fall time occurs when pull-up current is one half the pull-down. Thus, at most only one-half as much current is available in a pseudo-NMOS circuit as there is in a CMOS circuit using the same size transistors. In actual circuits the sum of rise and fall time is somewhat worse than this for pseudo-NMOS because for noise immunity the pull-up is usually chosen somewhat smaller than half the pull-down.

As a result, the speed of CMOS and pseudo-NMOS are very close. The CMOS has twice the capacitance but also twice the available current. The tradeoff in choosing one or the other is between the low power of the CMOS and the low area of the pseudo-NMOS.

The remainder of this paper will show first how dynamic circuits have combined both low-capacitance and high-current

Reprinted from *IEEE Journal of Solid-State Circuits*, Vol. SC-17, No. 3, pp. 614-619, June 1982.

capability, but at a cost in circuit stability and operational complexity. Next, new techniques will be described which maintain the above advantage of dynamic circuits while still keeping the stability and simplicity of static circuits. Finally, some specific examples will be presented.

II. Dynamic Circuits

Many dynamic circuit schemes have been described [2], but they all show some basic features in common. Basically, they involve precharging the output node to a particular level (usually high for NMOS), while the current path to the other level (ground for NMOS) is turned off. Changing of inputs to the gate must occur during this precharge phase. At the completion of precharge, the path to the high level is turned off by a clock and the path to ground is turned on. Then depending on the state of the inputs, the output will either float at the high level or will be pulled down. Fig. 3 illustrates how this is done for the 32 AOI gate described earlier. The advantage of a dynamic circuit is that the load capacitance is comparable to static pseudo-NMOS but the full pull-down current is available. Therefore, the gate should respond roughly twice as fast as either pseudo-NMOS or full CMOS. In addition, there is no static current path so power would be much closer to CMOS than to static pseudo-NMOS. (There is still some power penalty compared to CMOS because each gate must be precharged high every cycle even if its output is to continue low.)

However, there are serious problems involved in realizing these apparent speed advantages in real circuits. This happens because useful circuits generally have several logic gates in series and in the dynamic approach; no gate can be activated until its inputs have stabilized. There are many ways to clock the gates so that this occurs, and an example is shown in Fig. 4. A detailed description of the operation of this circuit is given in [2] and will not be repeated here. Basically, each gate goes through a precharge when transistors A and B are on, an evaluation when transistors B are on and A is off, and a hold period when transistors B are off. It is required that when a gate is in the evaluation mode, the gate driving it must be in the hold mode. There are four types of gates distinguished by the phase in which evaluation occurs. The one shown is type 3. This means that gate type 2 can drive either type 3 or type 4 but not type 1. Similar restrictions apply to each circuit type. This requires some additional care in design but is not a major problem. There are two reasons why the speed of this circuit will not be double that of a static circuit. First, each gate has two additional transistors in the pull-down path which reduces the available current considerably. For a 1- or 2-input gate this could easily be a factor of two. Second, the time allowed for a gate to stabilize must be chosen so that even the gate with the longest delay can settle down. This can cause substantial time waste on the faster gates because they must be allocated a full time slot. In addition, the difficulties of generating the four clocks and synchronizing them throughout the circuit to a small fraction of a gate delay are formidable. In practice considerably more than one gate delay would be needed between successive edges to assure a full gate delay in worst case. Overall then, in a circuit of reasonable complexity, the dynamic approach would not be any faster than static though it would have power advantages compared to pseudo-NMOS or NMOS.

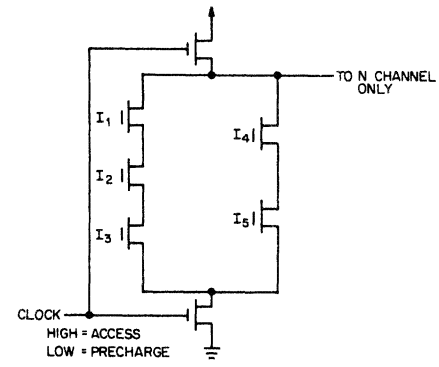


Fig. 3. Dynamic pseudo-NMOS gate. Low output capacitance and no static pull-up, but inputs must be valid before access begins.

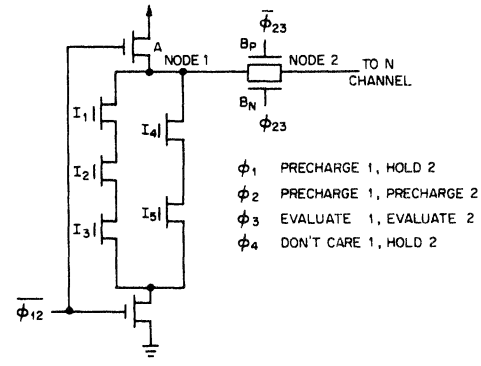


Fig. 4. Four phase dynamic pseudo-NMOS. The shortest clock phase must be long enough so that the slowest gate in the circuit can complete its evaluation. This results in considerable dead time.

III. CMOS Domino Circuit

The CMOS domino circuit shares some characteristics with dynamic circuits. In particular, each output is precharged high while the path to ground is opened and the precharge is stopped while the path to ground is activated. The critical difference is that the transition from precharge to evaluation is accomplished by means of a single clock edge applied simultaneously to all gates in the circuit. This greatly simplifies clocking and permits utilization of the full inherent speed of the gates.

A single domino circuit gate is shown in Fig. 5. It consists of two parts. The first looks like a dynamic pseudo-NMOS gate and is clocked in the same way as such a gate, with a precharge phase followed by an evaluation phase. The second part is a static CMOS buffer. Only the output of the static buffer is fed to other gates of the circuit; the output of the dynamic gate goes only to the buffer. During precharge, the dynamic gate has a high output so the buffer output is low. This means that during precharge, all circuit nodes which connect the output of one domino gate to the input of another are low, and therefore the transistors they drive are off. In addition, during evaluation a domino gate can make only a single transition, namely from a low to high. Because of the nature of the dynamic gate which drives it, it is impossible for the buffer to go from high to low during evaluation. (Since the dynamic gate cannot go high, the buffer cannot go low.) As a result there

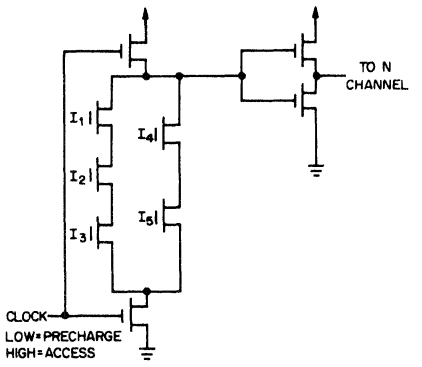


Fig. 5. Domino CMOS circuit. No static power, low area, with simple single edge clocking for all gates in the circuit.

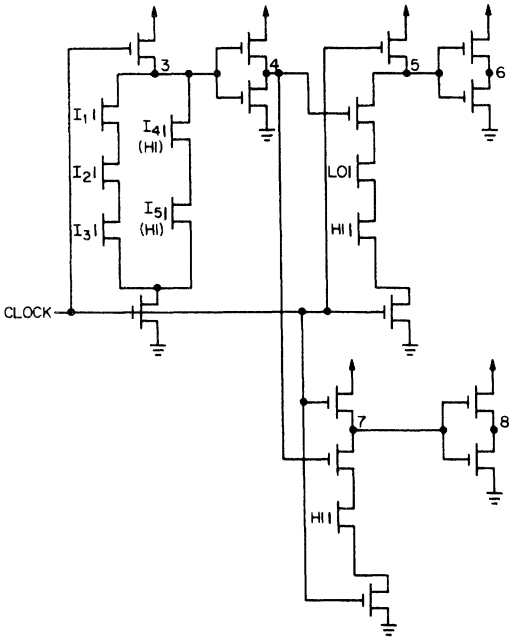


Fig. 6. An example of a domino CMOS circuit showing how a single clock activates all clocks simultaneously.

can be no glitches at any nodes in this circuit. All nodes can make at most only a single transition and then must stay there until the next precharge. This is reminiscent of the behavior of a row of dominos toppling into one another, and hence the proposed name.

Since there is no need to worry about glitches and since during precharge all domino outputs turn off the transistor they drive, all gates may be switched from precharge to evaluate with the same clock edge. An example of how this works is shown in Fig. 6. During precharge, nodes 3, 5, and 7 are all high so nodes 4, 6, and 8 are low. When precharge ends node 4 goes high which causes node 8 to go high. Node 6 remains low during evaluation.

As will be described in more detail in the next section many types of circuits when made with domino gates can be significantly faster than a corresponding circuit made with other techniques. The circuit has the low power of a dynamic circuit since there is never a dc path to ground. Also, the full pull-down current is available to drive the output nodes. At the same time the load capacitance is much smaller than for

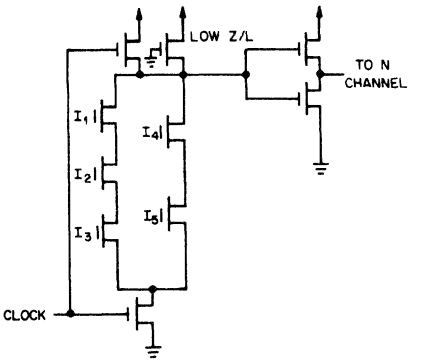


Fig. 7. Domino CMOS circuit with an additional pull-up device to permit static or low-frequency operation.

CMOS because most of the p-channel transistors have been eliminated from the load. Meanwhile, the use of a single clock edge to activate the circuit provides simple operation and full utilization of the speed of each gate. (There is no dead time between output valid and operation of the next gate in the circuit.)

One limitation of this circuit technique is that all of the gates are noninverting. This may seem serious since an XOR is not possible, but actually very complex circuits can be implemented including an arithmetic logic unit (ALU) with two levels of carry look ahead (to be described later). This is feasible because the domino gate is fully compatible with standard CMOS gates and the needed CMOS XOR can be driven by the last domino circuit.

Another limitation is that each gate must be buffered. This has not been a problem in the circuits designed so far because buffers would have been needed anyway to achieve maximum speed. The need for buffers indicates that this circuit technique is most valuable in logic involving many gates with high fan-out.

IV. Static Domino Circuit

In some applications it is desirable to have a static capability to allow lower frequency operation or to avoid the risk of storing data on floating nodes. This can be obtained in a domino circuit by the addition of a low current pull-up transistor as shown in Fig. 7. This functions as a means of removing charge which accumulates on the output node as a result of leakage or noise.

This transistor would be chosen small enough so there is no significant impact on pull-down current and so the power consumed during the evaluation phase is tolerable. A value of 10 μA is reasonable. This would require a p-channel transistor that is 20 μm long and 4 μm wide. For a chip with 2000 pull-up devices at 5 V, power consumption during evaluation would be 100 mW, if all gates are being pulled down. Average power would depend on the application but would be significantly less.

Another way to implement the static circuit is to include the static pull-up transistor shown in Fig. 7, but to have no clocked precharge transistor. This can be done if the time between evaluation phases is relatively long so precharge can be accomplished by the weak static pull-up transistor.

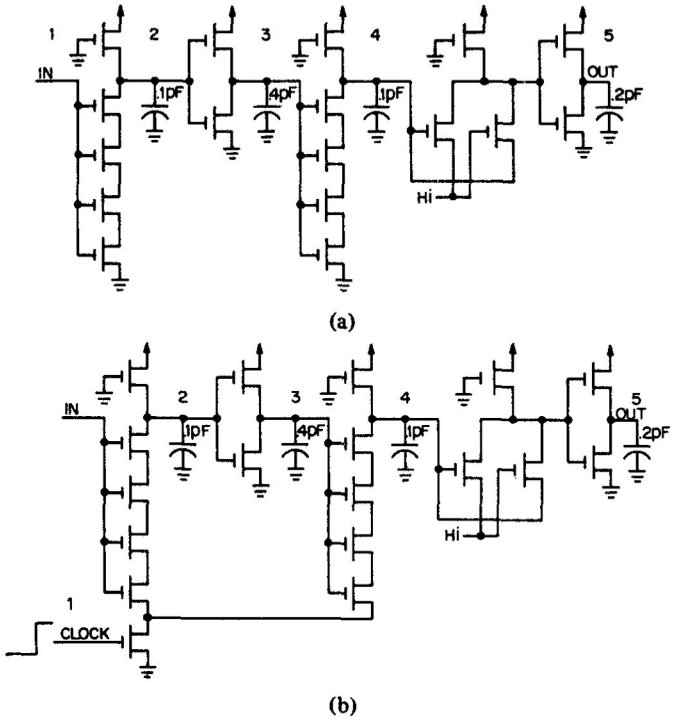


Fig. 8. (a) Part of an 8-bit ALU critical path using static pseudo-NMOS. (b) Same critical path with domino clocking.

V. An 8-Bit ALU

The first use of the domino circuit was on an 8-bit arithmetic logic unit (ALU) of an 8-bit microprocessor [3]. This happened because simulations indicated that adequate performance could not be obtained with a pseudo-NMOS circuit, while full CMOS was too area-consuming. The circuit of part of the critical path of ALU using pseudo-NMOS is shown in Fig. 8(a). The ALU in domino CMOS uses 690 transistors, and with a 15 μm pitch for metal and polysilicon, the area is 6000 mils^2. A similar transistor density in full CMOS would have required an additional 3000 mils^2 which was not available. A photograph of the ALU is shown in Fig. 9. The large structure on the lower left is the clocked ground switch which turns on the ALU when it is the evaluate phase. In a strip along the right side are the p-channel static load devices.

A SPICE [4] simulation of the simple pseudo-NMOS critical path predicted a worst case propagation delay of 450 ns which exceeded the chip requirement of 250 ns. A SPICE simulation of the domino circuit which is shown in Fig. 8(b) predicted 215 ns and so the design was made this way. Note that this circuit is like the one in Fig. 7 except that the clocked pull-up transistor has been eliminated and only the static one remains. This was done because the time between accesses of this ALU are so long that the low Z/L static transistor is sufficient to do the precharge. Table I shows propagation delays predicted by the simulation for both pseudo-NMOS and domino CMOS. The very slow pull-up time dominates the pseudo-NMOS CMOS delay. This happens because even though optimum speed is obtained with pull-up current equal to one-half pull-down current, noise margins forced a smaller ratio resulting in slow pull-ups. A histogram of measured propagation delay for 116 circuits that were fabricated is shown in Fig. 10. This

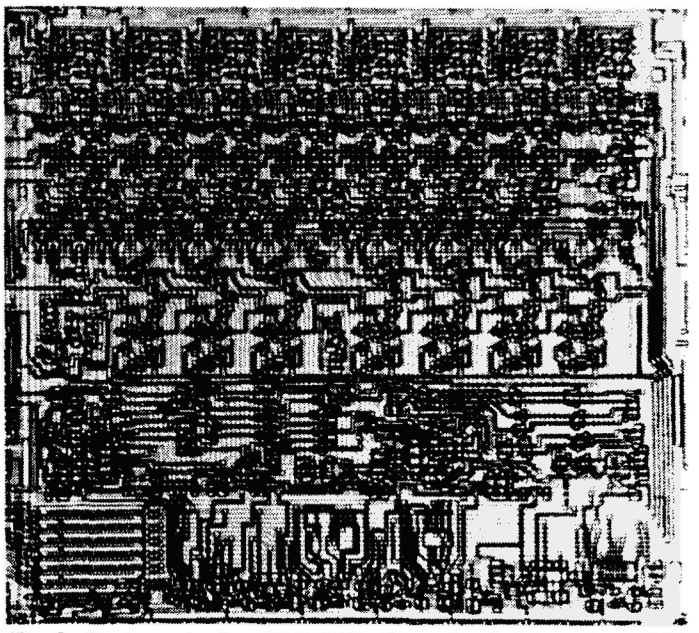


Fig. 9. Photograph of an 8-bit ALU. Ground switch is on lower left.

TABLE I
Worst Case Delays in 8-Bit ALU

	Static Circuit Delay (nSec)		Domino Circuit Delay (nSec)	
Node	In Goes High	In Goes Low	In High	In Low
1	0	0	0	0
2	40	270	100	0
3	10	10	25	0
4	40	80	25	0
5	50	50	65	0
TOTAL	140	410	215	0

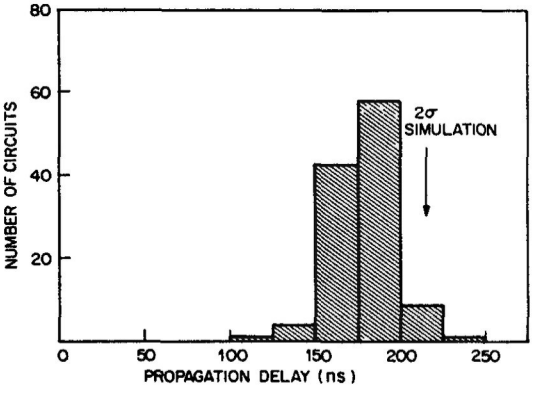


Fig. 10. Histogram showing distribution of delays for 116 ALU circuits.

histogram confirmed the high-speed predictions made by the simulation and verified the operation of the domino CMOS circuit.

VI. A 32-Bit ALU

For a more complex example, a critical path in a 32-bit ALU [5] will now be discussed. This circuit uses 3300 transistors and does a 32-bit add as well as other arithmetic and logic

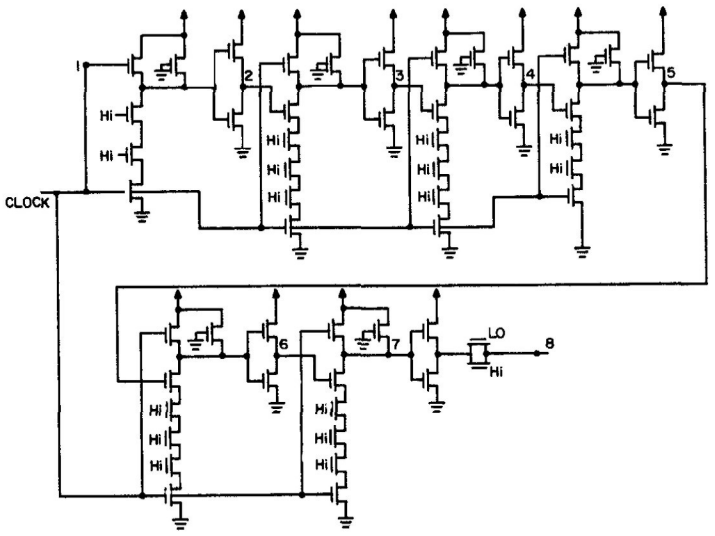


Fig. 11. Critical path through a 32-bit ALU.

TABLE II
WORST CASE DELAYS IN 32-BIT ALU

Node	Delay (nSec)
2	13
3	29
4	16
5	22
6	16
7	21
8	7
TOTAL	124

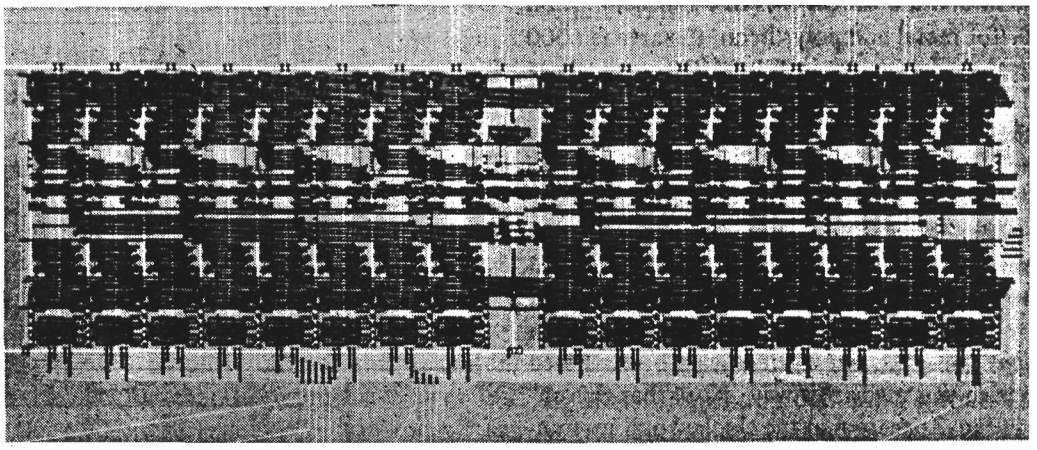


Fig. 12. Photograph of a 32-bit ALU.

function. The critical path in the domino CMOS path is shown in Fig. 11. Simulations have been made for this path and Table II gives propagation delays at various nodes on the critical path. The predicted worst case total propagation delay is 124 ns for $V_{DD} = 4.75$ V and a junction temperature of 105°C. This circuit was fabricated and a photograph of it is shown in Fig. 12. Process parameters of test transistors on the wafer were measured and using these a propagation delay of 104 ns was predicted. The actual delay was 97 ns.

VII. SUMMARY

A new compact, high-performance circuit design technique has been described for use with CMOS technology. This domino CMOS technique gives circuits with areas comparable to static NMOS or pseudo-NMOS, but gives a speed improvement of a factor of 1.5 to 2. This is achieved without resorting to any multiphase clocks and the static stability of the circuit can be maintained.

REFERENCES

[1] S. M. Kang, "A design of CMOS polycells for LSI circuits," *IEEE Trans. Circuits Syst.*, vol. CAS-28, pp. 838–843, Aug. 1981.

[2] W. M. Pensey and L. Lau, *MOS Integrated Circuits*. New York: Van Nostrand, 1972, pp. 260–282.

[3] J. A. Cooper, J. A. Copeland, R. H. Krambeck, D. C. Stanzione, and L. C. Thomas, "A CMOS microprocessor for telecommunications applications," in *Dig. ISSCC*, Feb. 1977.

[4] L. W. Nagel and D. O. Pederson, "Simulation program with integrated circuit emphasis," in *Proc. 16th Midwest Symp. Circuit Theory*," Waterloo, Ont., Canada, Apr. 1973.

[5] B. T. Murphy, R. Edwards, L. C. Thomas, and J. J. Molinelli, "A CMOS 32-bit single chip microprocessor," in *Dig. ISSCC*, Feb. 1981.

Design-Performance Trade-Offs in CMOS-Domino Logic

VOJIN G. OKLOBDZIJA AND ROBERT K. MONTOYE, MEMBER, IEEE

Abstract —**This paper is a study of the charge-sharing problem and its effect on the performance of CMOS-Domino logic. Several solutions to the charge-sharing problem are examined, and the results are verified by simulation. Thus the charge-sharing problem in CMOS-Domino logic was identified and alternate approaches were evaluated.**

I. Introduction

With increased interest in CMOS, Domino-type logic has been gaining favor due to its n-MOS-like performance (i.e., n-channel dominant delay) and CMOS-like power consumption [1], [2], and favorable testability relative to CMOS [3]. However, Domino logic presents a charge-redistribution problem that continues to impair its usability.

This logic family was developed during the course of implementation of BELMAC-32 microprocessor and the first paper on Domino logic was published by the authors from Bell Laboratories [2]. However, the logic originally published had several drawbacks. For example, inversion was not possible, making the implementation of EXCLUSIVE OR (XOR) function difficult. The original circuit implementation was very sensitive to the charge-redistribution problem, causing spurious results.

The authors from IBM developed a logic family, Cascode Voltage Switch (CVS), which further advanced the status of Domino logic [1]. This is a complete logic family because both polarities of each function output are available at every stage. In fact, this is a two-rail logic that offers a self-checking feature at no extra cost. Their logic family comes in two versions: static and dynamic. The dynamic version can be treated as part of the CMOS-Domino logic family. The static version of the CVS logic implements the p-MOS latches at the output nodes, which triggers a regenerative action to bring the nodes to their full logic one and logic zero values. An extension of this technique is Differential Split Level (DSL) logic, which claims a performance improvement of ten times over regular static CMOS, but consumes more power as reported [4].

In this paper, we will consider CMOS Domino and dynamic CVS version for the purpose of analysis and identification of the charge-redistribution problem.

II. Operation

Principles of operation of Domino-type logic are outlined by the circuit example shown in Fig. 1. This logic family evolved from the dynamic n-MOS (p-MOS) circuits and therefore retained two phases of operation: "precharge" and "evaluate" (designated PCHG and EVAL, respectively, in this paper). The basic logic function implemented with this type of logic consists of: clock circuitry (transistors Q_2, Q_{p1}), n-MOS transistor network SWF implementing given Boolean function f, and inverter. During the PCHG phase of the clock, the p-MOS transistor Q_{p1} is ON while the n-MOS transistor Q_2 is OFF. Node $N4$ is charged to V_{dd} and the output from the inverter is at the voltage level close to 0 v. This situation occurs at that time at every logic block including those whose outputs are connected to the inputs X_i of this particular block. Registers are designed in the same way so that all of their outputs are logic zero value during the PCHG phase. As a consequence, all of the inputs X_i of the particular block, and all of the other blocks, are close to 0 v during the PCHG-phase. Therefore during the PCHG phase there is no electrical path from the "top" node (N_4) to the "bottom" node (N_1) and only the "top" node (N_4) is storing charge. When the clock turns to the EVAL phase, transistor Q_2 is ON creating the path from the node $N4$ through the switching network SWF to the ground. If the condition for the existence of an electrical path in the network SWF (between the nodes $N4$ and $N2$) is established by the signal values of the inputs to the SWF, node $N4$ is discharged to ground, which in turn makes the output of the inverter F the logic ONE. This value is the input to the subsequent logic block(s) and can cause the output of the block(s) to switch to ONE. This signal change is propagated in the "domino" fashion.

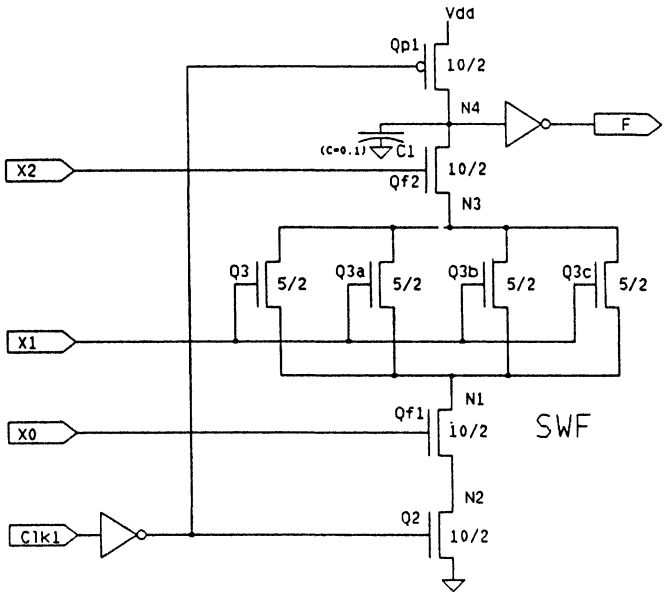


Fig. 1. Domino circuit. A p-channel transistor is indicated by a circle at the gate. W/L ratios are indicated next to the transistor. Waveforms on the internal nodes $N1$, $N3$, and $N4$ are shown in Fig. 2.

A. Charge-Redistribution Problem

From the operation of the Domino logic, it is clear that the charge is stored only at the "top" node ($N4$ in Fig. 1.) and the nodes N_1, N_2, N_3 are not charged during the PCHG phase. They might have been discharged during the previous cycle and thus have no charge. Therefore, during the evaluation phase, there may be an electrical path to several discharged nodes (causing charge redistribution) without an electrical path to ground. If there is sufficient charge redistribution (i.e., the ratio of the capacitance at the top node of the tree C_t) the uncharged capacitance internal to the tree C_i reduces the voltage below the inverter threshold I_{th}

$$V_{dd} \times \frac{C_t}{C_i + C_t} \leqslant I_{th}.$$

This charge redistribution will cause the inverter at the output of the tree to falsely switch, thus placing the incorrect value on the line causing other groups to discharge falsely. One such example (shown in Fig. 2) is generated by simulation of the single Domino-logic stage using the Toggle circuit simulation package [5]. We are observing in this case the behavior of the logic block during the period of two full cycles: PCHG–EVAL, PCHG–EVAL. During the first cycle inputs $X0$, $X1$, and $X2$ are set to

Manuscript received September 9, 1985; revised December 20, 1985.
The authors are with the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598.
IEEE Log Number 8607667.

Reprinted from *IEEE Journal of Solid-State Circuits*, Vol. SC-21, No. 2, pp. 304-306, April 1986.

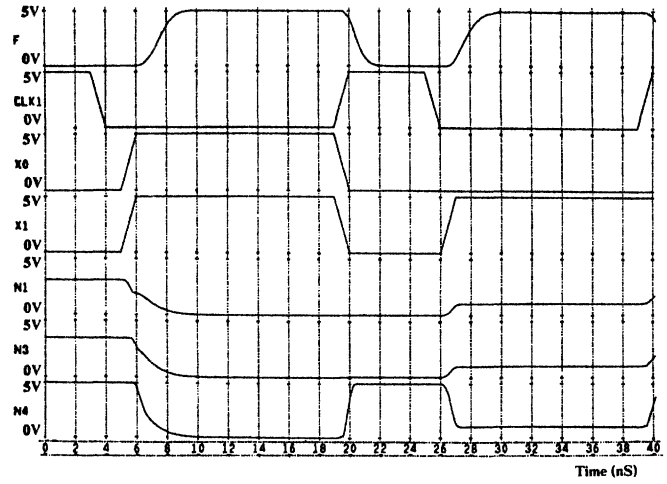


Fig. 2. Waveforms from the Domino circuit example in Fig. 1. The effect of charge redistribution is seen on the node $N4$.

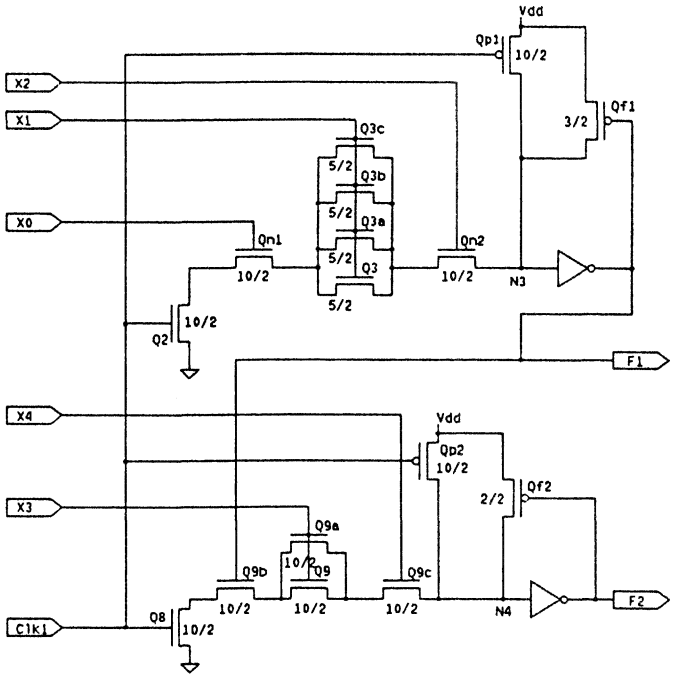


Fig. 3. An example of CVS logic which was used to simulate charge redistribution. p-channel transistors are distinguished by the circle associated with the gate symbol. Voltages on the nodes $N3$ and $N4$ are shown in Fig. 4

logic one causing the discharge of the entire network during the EVAL phase. Following the PCHG phase in the second cycle, the node $N4$ is precharged to the value of 5.000 V. In the second cycle, the inputs are set to $X0 = 0$, $X1 = 1$, and $X2 = 1$ creating an electrical path between the nodes $N4$, $N3$, and $N1$, but stopping short of the node $N2$ which is connected to ground during the EVAL phase. This situation causes redistribution of the charge between the nodes $N4$, $N3$, and $N1$. From the waveforms in Fig. 2, we can observe that the voltage on the node $N4$ falls to 1.0016 V. The voltage on node $N1$ has risen to 1.0005 V and the voltage on node $N3$ has risen to 1.0009 V. The input combination ($X_0 = 0$, $X_1 = 1$, and $X_2 = 1$) is supposed to produce the logic ZERO value at the output F_1. However, because of the redistribution of charge between the node N_4 and the nodes N_1, N_2, N_3 the voltage at node N_4 is only 1.0016 V. This produces an erroneous value of logic ONE at the output F_1.

III. Problem Elimination Methods

In this section we examine the techniques used to alleviate the problem caused by charge redistribution and evaluate the trade-offs in reliability and circuit performance.

Two methods of reducing the charge-sharing problem are addressed. The first of these, incorporating feedback into the tree, reduces the charge-sharing problem by injecting charge into the tree during evaluation. The second method selectively increases the storage capacity of the precharge node in proportion to the number of nodes to which the charge can be redistributed.

A. Feedback Transistor (Dynamic CVS Logic ONE)

One method used by dynamic CVS logic to alleviate the charge redistribution problem is to place an additional p-MOS transistor Qf in parallel with the precharge transistor. The gate of this transistor is connected to the output of the inverter so that feedback from the output is obtained (Fig. 3). In this way the inverter-transistor combination acts as a "latch" that locks on the state where the output of the inverter F is at the ZERO logic value. Because the output $F = 0$ is "latched", it takes more current from the node $N4$ to pull the node to ground since the charge is being continuously replenished by the device Q_f. When charge redistribution occurs, transistor Q_f serves the purpose of replenishing the charge lost in the process of redistribution to the other nodes. We can distinguish two cases.

1) During the charge redistribution, the total sum of the currents to node N_4 is such that node N_4 will recover the charge lost by redistribution. The current from the node N_4 which is due to redistribution of charge, decreases exponentially in time. The current to the node N_4 through the transistor Q_f will also exponentially decrease in time but have a larger time constant.

As a result, the current calculated with reference to the node N_4 is negative at the beginning and is equal to the charge taken from the node and distributed to other nodes. Later, this current becomes positive, bringing the charge lost in redistribution back to the node N_4. It decreases exponentially to ZERO. This produces a voltage "spike" or "glitch" at the node $N4$ which is of the amplitude that never exceeds the threshold for logic ONE at the inverter input.

2) In the second case, the amount of charge lost during the initial period is such that produces the voltage spike of sufficient amplitude to change the output value to logical ONE. This in turn cuts off the transistor Qf preventing it from replenishing the lost charge to the node $N4$. In this case, the fault is permanent and the output will stay at the erroneous value of logic ONE instead of ZERO.

Let us consider the first case and the consequences of the voltage "spike." The voltage "spike" at the node $N4$ is propagated through the inverter producing a positive voltage "spike" at the output F. The effect of this positive voltage "spike" can be twofold.

1) It can cause complete discharge in the next logic block creating the erroneous value to appear at its output. This value is propagated further in a "domino" fashion.

2) The voltage "spike" can cause a similar "spike" at the output of the next logic block. This spike is further propagated, in which case it can be:

a) of the smaller amplitude and therefore dissipated in the logic;
b) amplified through the consequent stages and therefore eventually resulting in a permanent error being propagated (much like the case 1);
c) fanned out in different directions, in which some will dissipate the spike and some will amplify it and distribute the erroneous reading.

These cases are illustrated in Fig. 4.

However, feedback transistor Q_f serves as a load transistor when the node $N4$ is forced to ground. Therefore the operation is now that of a ratioed circuit, and as such the transistor Q_f serves as a load and its L/W ratio has to be adjusted to the cumulative (L/W) eff ratio of the maximum length electrical path to ground of the switching block SWF. L/W ratio represents resistance of the feedback transistor in the circuit.

Let us define β to be the ratio of $(L/W)f/(L/W)$eff. This ratio represents the resistance of the feedback device compared to the resis-

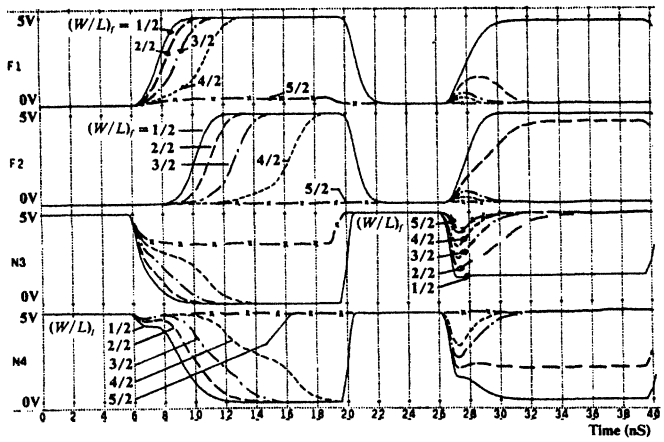


Fig. 4. Signals at the nodes $F1$, $F2$, $N3$, and $N4$ for the circuit in Fig. 3, as a function of time for various sizes $(W/L)_f$ of the feedback transistors Q_{f1} and Q_{f2}.

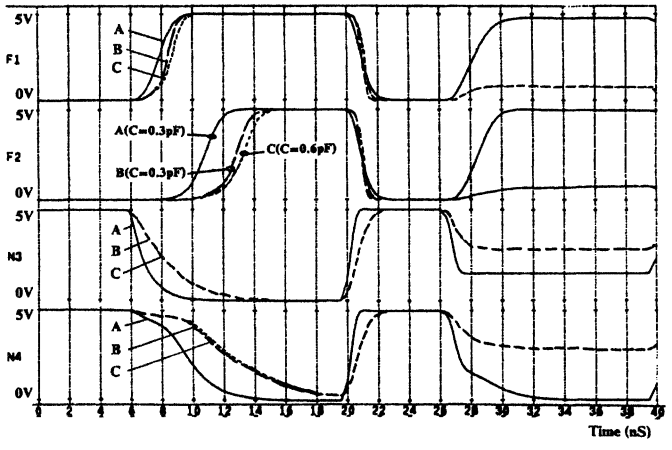


Fig. 5. Waveforms on the nodes $F1$, $F2$, $N3$, and $N4$ for the circuit in Fig. 3. without feedback transistors Q_{f1} and Q_{f2}. The size of the output inverters is varied: A—small inverter; B—output inverter size increased 7 times over case A; C—output inverter of the size in case B with the output load on the nodes $F1$ and $F2$ doubled.

tance of the switching network SWF. We can distinguish three cases:

1) β too large, in which case the feedback device is not very effective except for very small "glitches;"
2) β in the range on 0.9, which was determined to be the optimal value with respect to adequate "glitch" protection; and
3) β too small, in which case the output F acts as being stuck at zero, because the SWF block is too weak to pull the node $N4$ to ground.

However, the "safe" or "glitch-free" operation is dependent on the proper choice for β. The range of β values in which the circuit is effective in "glitch" suppression imposes the restriction on the maximal length of the possible electrical paths in the SWF network. Additionally, a restriction is placed on the number of nodes in the tree that can be at 0 V and cause redistribution.

Given the example shown, in which a SWF with three devices in a series and a total of six devices can provide glitch immunity only with careful feedback device tuning, it is clear that the range of effectiveness of the feedback device is very limited.

Another impact of the feedback device is on performance. This is visible in Fig. 4, as the delay increases markedly as a larger feedback device (smaller β) is used, until, as previously mentioned, the circuit fails to operate β below 0.6. Thus the reduction in glitch sensitivity is paid for by a degradation in performance. This difficulty becomes apparent as larger SWF's are used, since the path to ground involves more active devices, and there are more total devices in the circuit.

B. Charge Storage on Output Inverters

One method of reducing the glitch sensitivity is to increase the capacitance of the precharge node. This method forces the charge to be drained proportionally to the number of nodes in the SWF. This increase in capacitance allows the charge stored to be distributed over the available nonprecharged drains. This capacitance can be increased by making the size of the transistors in the output inverter larger. This method has the major advantage of increasing the drive capability of the circuit, thus reducing its sensitivity to output loading. The waveform A in Fig. 5 shows the result of using a small output inverter, i.e., charge redistribution. The waveform B results from increasing the output inverter by a factor of 7, which removes the glitch in the output at the expense of a 25-percent increase in delay. This is because the additional capacitance must be discharged if the circuit is to switch. However, the increase allows much greater insensitivity to the effects of increasing the output load, since the output driver is much larger. Notice that there is both charge redistribution for the first set of inputs and along delay for the true switching signal. The final waveform C shows the result of the circuit with the larger output inverter and double the expected loading, i.e., 0.6 pf. Note that the delay for this circuit is only slightly larger than for the circuit having the output inverter driving the smaller load.

The technique of increasing the output inverter size to reduce the effects of charge sharing is limited in range to acceptable output inverter sizes and delays. It has the potential to reduce charge sharing in cases where the problem is not too severe. However, the internal switching delay may grow significantly with larger trees. The additional side effect of increasing the switching speed of the output load may offset some of this weakness.

IV. Conclusion

The charge-sharing problem may be combatted using several methods. Two solutions to this problem were described and analyzed. It was concluded that feedback devices are helpful in a limited range and increasing output inverter sizes are additionally helpful to reduce the problems of charge redistribution. In practice, these two methods can be used to produce a large family of glitch-free trees. Further study is required to make Domino logic viable for a wider range of switching functions.

References

[1] L. G. Heller, W. R. Griffin, J. W. Davis, and N. G. Thoma, "Cascode voltage switch logic: A differential CMOS logic family," in *Proc. ISSCC* (San Francisco, CA), Feb. 22–24, 1984.
[2] R. H. Krambeck *et al.*, "High-speed compact circuits with CMOS," *IEEE J. Solid-State Circuits*, vol. SC-17, no. 3, June 1982.
[3] V. G. Oklobdzija and P. G. Kovijanic, "On testability of CMOS-Domino logic," in *Proc. 14th Int. Conf. Fault-Tolerant Computing* (Orlando, FL), June 20–22, 1984.
[4] L. C. M. G. Pfennings *et al.*, "Differential split-level logic for sub-nanosecond speeds," in *Proc. ISSCC 1985* (New York, NY), Feb. 14, 1985.
[5] J. F. Beetem *et al.*, "A large-scale MOSFET circuit analyzer based on waveform relaxation," in *Proc. ICCD'84* (Port Chester, NY), Oct. 8–11, 1984.

NORA: A Racefree Dynamic CMOS Technique for Pipelined Logic Structures

NELSON F. GONCALVES, STUDENT MEMBER, IEEE, AND HUGO J. DE MAN, SENIOR MEMBER, IEEE

Abstract—This paper describes a new dynamic CMOS technique which is fully racefree, yet has high logic flexibility. The circuits operate racefree from two clocks ϕ and $\bar{\phi}$ regardless of their overlap time. In contrast to the critical clock skew specification in the conventional CMOS pipelined circuits, the proposed technique imposes no restriction to the amount of clock skew. The main building blocks of the NORA technique are dynamic CMOS and C^2MOS logic functions. Static CMOS functions can also be employed. Logic composition rules to mix dynamic CMOS, C^2MOS, and conventional CMOS will be presented. Different from Domino technique, logic inversion is also provided. This means higher logic flexibility and less transistors for the same function. The effects of charge redistribution, noise margin, and leakage in the dynamic CMOS blocks are also analyzed. Experimental results show the feasibility of the principles discussed.

I. INTRODUCTION

IN the conventional CMOS technique there is an inherent redundancy of information. For each n-type device there is a corresponding p-type device. In fact, a complete logic function is built with the n devices and repeated with the p devices. As a consequence of this approach, substantial amounts of silicon are wasted, especially for complex logic. Also, power dissipation and speed are degraded by the extra area and extra transistors.

Another important problem of CMOS technique is clock races in pipelined circuits. To latch the information between two pipelined sections, transmission gates are usually employed. In CMOS logic, these transmission gates are generally implemented with p-n gates in parallel and controlled by clocks ϕ and $\bar{\phi}$, as shown in Fig. 1. The use of single gates (p- or n-type) is to be avoided in CMOS due to power dissipation and low noise margin as a result of clock feedthrough and bulk effect. CMOS p-n transmission gates, controlled by clocks ϕ and $\bar{\phi}$, suffer from signal races. As depicted in Fig. 1, this results from unavoidable overlap of the clock phases during the clock transitions. During the phase overlaps, all the transmission gates are switched on, which may cause illegal flow of information, depending on the ratio between the gate delay and the clock skew.

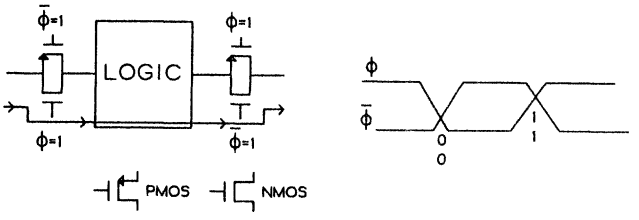


Fig. 1. Signal races in CMOS pipelined circuits.

This race problem is usually bypassed by a careful synchronization of the two clock phases within a small fraction of the gate delay (a few nanoseconds). This skew clock control is extremely difficult, especially for high speed technologies, for unmatched clock loads or for distributed clock VLSI circuits [1]. This leads to highly critical and untestable designs. A possible solution to the clock race is the use of four clock phases which, however, requires too much silicon area.

To overcome the redundancy of information in the conventional CMOS, dynamic circuit schemes have been proposed in

Manuscript received November 8, 1982; revised January 18, 1983. This work was supported in part by Fundaçao de Amparo a Pesquisa do Estado de São Paulo, Brazil.

The authors are with the Department Elektrotechniek-ESAT, Katholieke Universiteit Leuven, B 3030-Heverlee, Belgium.

Reprinted from *IEEE Journal of Solid-State Circuits*, Vol. SC-18, No. 3, pp. 261-266, June 1983.

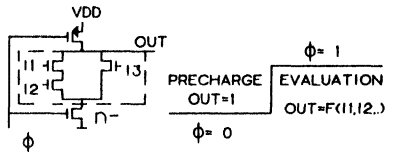


Fig. 2. An n-type dynamic CMOS logic block.

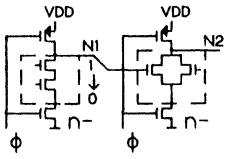


Fig. 3. Internal delay race problem.

the literature [2]-[4]. Fig. 2 shows the dynamic CMOS building block. The desired logic function is implemented using only n-type devices. The logic tree is connected to V_{DD} and groundthrough clocked transistors. There are two modes of operation. First, for phase $\phi = 0$ the output node is precharged to a high level while the current path to ground is turned off. Then, for phase $\phi = 1$, the path to the high level is turned off by the clock and the path to ground is turned on. Therefore, depending on the state of the inputs, the output node will either float at the high level or will be pulled down.

A clear advantage of this CMOS dynamic block is the reduced silicon area. Whereas there are $2n$ transistors in a conventional n-input CMOS gate, the dynamic configuration needs only $n + 2$. Also due to the smaller area and consequently smaller capacitances, power dissipation and speed are, in principle, improved by the dynamic approach.

A strong limitation of this dynamic structure is the impossibility of cascading the logic blocks for implementing complex logic. Consider, for instance, the circuit in Fig. 3. During the precharge phase, nodes $N1$ and $N2$ are set up to the high level "1." In the evaluation phase ($\phi = 1$), internal delay in block 1, associated with a "1" → "0" transition of node $N1$, can cause an incorrect discharge of node $N2$. This occurs because, during the evaluation phase and while node $N1$ is still "1," there is a direct path between node $N2$ and ground. When this path is eliminated by the effective transition of node $N1$ to "0," the precharge information of node $N2$ could already be gone. We define such a race as the "internal delay problem."

In the Domino technique, Krambeck *et al.* [4] have solved the internal race by placing a static inverter after every dynamic block, as indicated in Fig. 4. During the precharge phase, the outputs of all the static inverters are set up to a low level. Consequently, all the n-type transistors driven by these inputs are set up to an OFF condition. Now, during the evaluation phase, internal delays cannot incorrectly discharge the dynamic storage nodes since during the entire delay period the path to ground is turned off.

A limitation of the Domino technique is the lack of inverted signals. The combination of the dynamic block with the static inverter gives a noninverted signal. This decreases logic flexibility and, therefore, usually requires more transistors for a given *logic function. Besides this inconvenience*, no provisions are made to overcome the clock race problem.

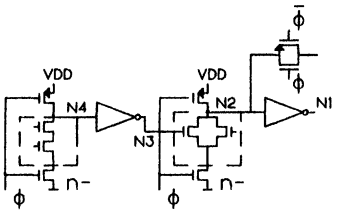


Fig. 4. Domino circuit.

In the next section, a new technique called NORA is presented which overcomes the above deficiencies. In Section III, the properties of the NORA CMOS technique are analyzed and proved. Logic composition rules to mix dynamic, static, and C²MOS [5] logic functions are also derived. The dynamic CMOS limitations are described in Section IV. Experimental results and major conclusions are presented in Sections V and VI, respectively.

II. NORA CMOS Technique

The main building blocks of NORA technique are shown in Fig. 5. The logic functions are implemented using n-type and p-type dynamic CMOS and C²MOS blocks. Conventional (static) CMOS function blocks can also be eventually employed. Logic composition rules to combine these functions, preserving the racefree properties, will be presented in Section III. As it will further be shown, to guarantee a fully racefree operation in pipelined circuits, the storage of information must always be performed by a C²MOS function block (C²MOS latch stage). In a previous paper [6], the NORA (*NO RA*ce) technique was called n-p-CMOS, due to the possible employment of n- and p-dynamic blocks. We decided to change the name because the p-dynamic block is not essential to the racefree principle; it is only used to increase the logic flexibility.

The pipelined circuit in Fig. 5 is defined as a ϕ-section. For phase $\phi = 0$ $\overline{\phi} = 1$, the ϕ-section is in the precharge phase. The outputs of all the n- and p-dynamic blocks are precharged to "1" and "0," respectively. Also during this phase, the ϕ-section inputs are in a sampling mode, i.e, these inputs are set up.

For phase $\phi = 1$ $\overline{\phi} = 0$, the ϕ-section is in the evaluation phase. The ϕ-section inputs are held constant, and the outputs of all the dynamic blocks are evaluated as a function of the ϕ-section inputs and of the internal inputs[1]. From these output results, those which must be transferred to the next pipelined section are stored in C²MOS latch stages.

In the circuit of Fig. 5, notice the following characteristics (see Section III).

1) Inverted and noninverted signals are provided. When direct coupling between dynamic blocks is desired, the logic function is implemented by alternating p- and n-logic blocks. If the inverter is required, a Domino like connection is employed, i.e, sequences of the same block type are used (n-

[1]For convenience, the inputs of a dynamic block have been separated into section inputs and internal inputs. The section inputs are set up during the precharge phase. The internal inputs are set up during the evaluation phase. For instance, in Fig. 5:

IN–section inputs
N 2, *N* 3–internal inputs.

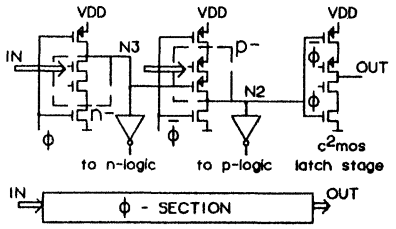


Fig. 5. NORA-CMOS pipelined circuit—ϕ-section.

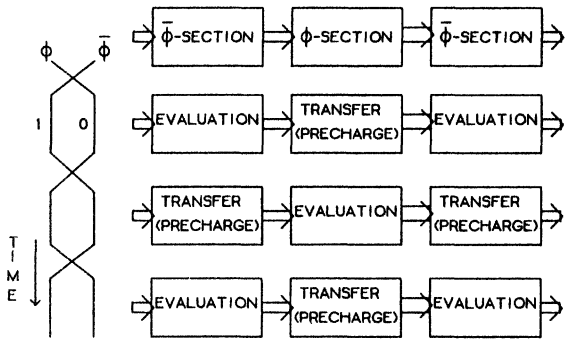


Fig. 6. Pipelined system.

inverter-n or p-inverter-p). Compared with the Domino technique, this means higher logic flexibility and less transistors for the same function.

2) n-p as well as p-n sequences are possible and the sequences can be of arbitrary logical depth. Therefore, many logic levels can be operated in only half a clock period.

By interchanging ϕ and $\bar{\phi}$ in the circuit of Fig. 5, a $\bar{\phi}$-section is obtained. A sequence of ϕ- and $\bar{\phi}$-sections makes a pipelined system, as shown in Fig. 6. For phase $\phi = 0$ $\bar{\phi} = 1$, the ϕ-sections are precharged while the $\bar{\phi}$-sections are in the evaluation phase. The ϕ-section outputs are held constant by the C^2MOS latch stages. Then, for phase $\phi = 1$ $\bar{\phi} = 0$, the ϕ-sections are in the evaluation phase and the $\bar{\phi}$-sections are precharged. Now, the $\bar{\phi}$-section outputs, evaluated in the previous phase, are held constant in such a way that the ϕ-sections can use this information to compute the corresponding results. In this way, there is a complete flow of information; with the information travelling from one $\bar{\phi}$-section to the next ϕ-section, from this to the next $\bar{\phi}$-section, and so on.

III. NORA Racefree Properties and Logic Composition Rules

In this section the racefree properties of the NORA technique will be carefully analyzed. Logic composition rules to combine dynamic, conventional, and C^2MOS function blocks will also be derived.

A. Internal Delay Racefree Property

The internal delay racefree property is defined as the capability of the dynamic block to keep its precharge signal during the delay time of the previous blocks to set up the internal inputs. It is easy to prove that a dynamic block will have the internal delay racefree property if the following conditions occur:

1) During the precharge phase, the internal inputs are set up in such a way they cut off their corresponding transistors.

2) During the evaluation phase, the internal inputs are glitch-free, i.e, these inputs can make only one transition.

From the above conditions the following results can be derived:

a) When the number of "static" inversions between two dynamic blocks is even, complementary type of logic blocks must be used for these two blocks (n-p or p-n). For instance in Fig. 5, this corresponds to alternate p- and n-logic blocks when the direct coupling between dynamic blocks is desired.

b) The same type of dynamic blocks (n-n or p-p) must be used when the number of "static" inversions is odd. In Fig. 5, this corresponds to Domino-like connections: n-inverter-n or p-inverter-p.

c) Normally, after mixing dynamic blocks with static CMOS, the circuit should be kept static up to the C^2MOS latch stage. Static functions can also be used after the C^2MOS stage. This should be done because in general "static" functions driven by dynamic blocks are not glitch-free. (Exceptions are the inverter and in some cases the NAND, NOR . . . .)

These logic composition rules can easily be implemented in a CAD system like Dialog [7] for automatic checking of the logic design consistency.

B. Clock Racefree Properties

As indicated in Fig. 6, to have a working pipelined system the results generated during the evaluation phase must be held constant until the end of the transfer phase. The latched information should not be altered by the precharge signal or by input variations. It will now be proven that after the evaluation phase a NORA pipelined section keeps its output results in spite of high-high or low-low clock overlaps (clock skew).

For simplicity, let us initially consider that all the circuits in the pipelined section are built only with dynamic blocks; the two exceptions being the C^2MOS latch stage and the static inverter for connecting complementary dynamic blocks. For this circuit, two possible cases should be analyzed.

Case I—Precharge Racefree

During the evaluation phase, the dynamic block which precedes the C^2MOS latch stage has its precharge signal modified by the inputs. Such a situation is indicated in Fig. 7 for an n-type and a p-type dynamic block.

As indicated in Fig. 7, the alteration of the output information is controlled by only one of the phases ϕ or $\bar{\phi}$. Therefore, these outputs are not influenced by the other phase. The outputs are, for instance, completely immune to the overlap of the phases. This kind of output latch control by only one phase (ϕ or $\bar{\phi}$) is completely different from the conventional case with transmission gates, where the output latch is controlled simultaneously by the two phases ϕ and $\bar{\phi}$. In contrast to the critical clock skew specification of the conventional transmission gates (few nanoseconds), the NORA technique imposes no restriction to the amount of clock skew.

Note: Although the NORA circuit is immune to the overlap

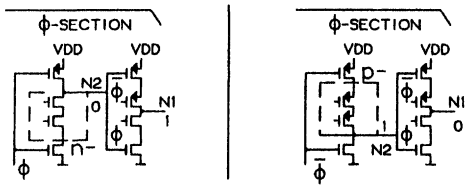


Fig. 7. Precharge racefree–precharge signal altered by the inputs:

n-type

$N1: 1 \rightarrow 0$ [$\phi = 1 \longrightarrow$ impossible*
$N2: 0 \rightarrow 1$ [$\phi = 0$]

p-type

$N1: 0 \rightarrow 1$ [$\bar{\phi} = 0 \longrightarrow$ impossible*.
$N2: 1 \rightarrow 0$ [$\bar{\phi} = 1$]

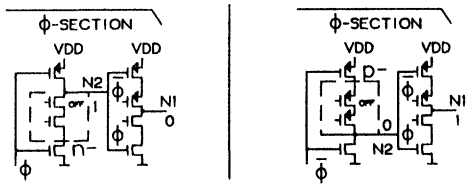


Fig. 8. Input variation racefree–precharge signal kept by the inputs.

of the clock phases, there could still be signal races for clock signals with very slow rise and fall times (10 to 20 times the gate delay). In contrast to the clock overlap race this kind of race is easily eliminated since it does not require control of the clock skew and, also, the available time margin is much larger.

Case II–Input Variation Racefree

The other possible case, i.e, when the dynamic block keeps the precharge signal, is illustrated in Fig. 8.

If the dynamic block keeps the precharge signal, at least one of the logic transistor should be driven off. If this transistor is controlled by an internal input, the dynamic block which generates this input has also kept its precharge signal. This occurs because the internal inputs are precharged in such a way that the corresponding driven transistors are off. Therefore, there must be at least one sequence of dynamic blocks with precharge signals preserved. Fig. 9 depicts this sequence. Again, as shown in Fig. 9, the alteration of the output information is controlled by only one of the phases ϕ or $\bar{\phi}$. Therefore, they are not influenced by the overlap of the phases.

For the case being analyzed, the racefree property has been derived from the interelation between a dynamic CMOS block and a C^2MOS latch stage. Let us now show that the input variation racefree property can also be derived by the action of two C^2MOS stages: "A NORA pipelined circuit is input variation racefree if the total number of inversions (static and dynamic) between two C^2MOS latch stages is even." The proof is indicated in Fig. 10. This racefree property can also be used to solve the clock race condition of some conventional CMOS circuits. An important circuit which can be built using the above property is the shift register.

Combining the racefree properties derived from two C^2MOS *functions and from* C^2*MOS with dynamic block, the following* result can be proven.

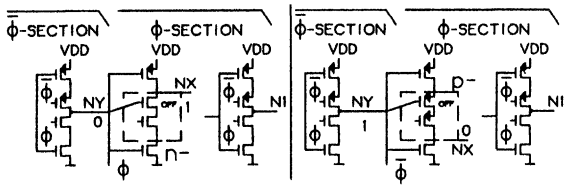


Fig. 9. Input variation racefree–sequence of dynamic blocks with precharge signals kept by the inputs:

n-type

$NX: 1 \rightarrow 0$ [$\phi = 1 \longrightarrow$ impossible*
$NY: 0 \rightarrow 1$ [$\phi = 0$]

p-type

$NX: 0 \rightarrow 1$ [$\bar{\phi} = 0 \longrightarrow$ impossible*.
$NY: 1 \rightarrow 0$ [$\bar{\phi} = 1$]

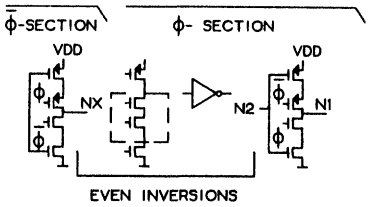


Fig. 10. Input variation racefree–even inversions between two C^2MOS latch stages:

"1" modification

$N1: 1 \rightarrow 0$ [$\phi = 1 \longrightarrow$ impossible*
$N2, NX: 0 \rightarrow 1$ [$\phi = 0$]

"0" modification

$N1: 0 \rightarrow 1$ [$\bar{\phi} = 0 \longrightarrow$ impossible*.
$N2, NX: 1 \rightarrow 0$ [$\bar{\phi} = 1$]

Consider a NORA pipelined section, built with dynamic, conventional, and C^2MOS function blocks. Consider all the chains of function blocks of this pipelined section, starting in a C^2MOS input stage (C^2MOS latch stage of the previous pipelined section) and ending in a C^2MOS output latch stage. The NORA pipelined section is clock racefree if, for every chain, the following conditions are satisfied:

1) Precharge racefree:

a) There is an even number of inversions between the C^2MOS output stage and the last dynamic block (see Fig. 11).

2) Input variation racefree:

b1) There is a dynamic block in such a way that there is an even number of inversions between this dynamic block and the C^2MOS input stage (see Fig. 12); or

b2) the total number of inversions between the two (input, output) C^2MOS stages is even (see Fig. 10).

If the pipelined section does not satisfy the clock race conditions, generally, circuit modifications can be easily included. By way of example, consider the nonracefree pipelined section indicated in Fig. 13(a). For this example, the following circuit modifications would eliminate the race condition:

1) conversion of one static function to dynamic function [see Fig. 13(b)];

2) conversion of one static function to C^2MOS function [see Fig. 13(c)];

3) placement of one static function after the C^2MOS latch

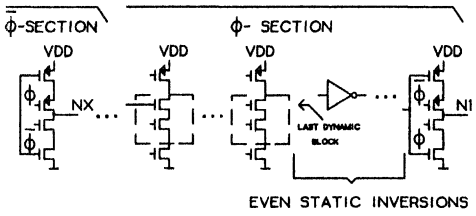


Fig. 11. Precharge racefree–even inversions between the C^2MOS output stage and the last dynamic block.

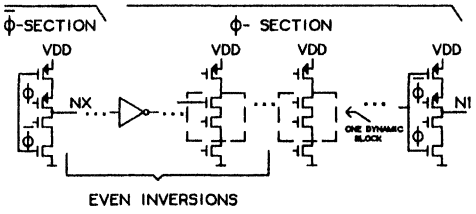


Fig. 12. Input variation racefree–even inversions between the C^2MOS input stage and one dynamic block.

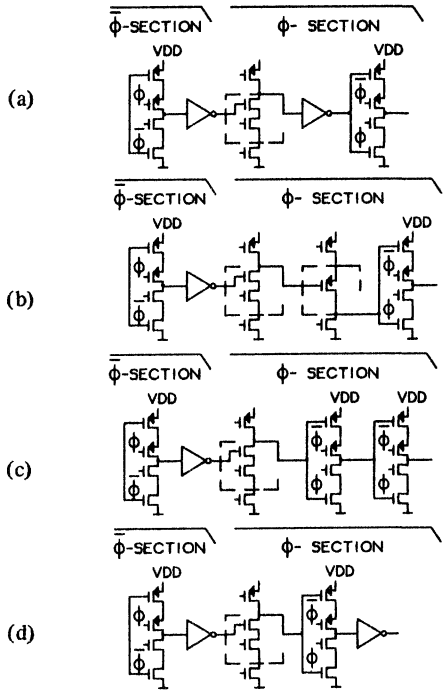


Fig. 13. Elimination of signal races. (a) Circuit with race of signals. (b) Conversion to dynamic CMOS function. (c) Conversion to C^2MOS function. (d) Placement after the C^2MOS output stage.

stage, provided that the racefree property of the next pipelined section would not be destroyed [see Fig. 13(d)].

IV. Dynamic CMOS Limitations

In this section the limitations of the NORA technique will be presented. These limitations are directly related to the dynamic storage of information and, therefore, they are common to all the dynamic techniques.

A. Charge Redistribution

The output signal of the dynamic blocks relies on storage nodes. As indicated in Fig. 14, by commutation of an OFF transistor to an ON state, a charge redistribution effect may appear between the output capacitance and the parasitic logic tree capacitances.

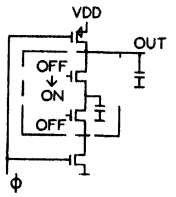


Fig. 14. Charge redistribution in dynamic blocks.

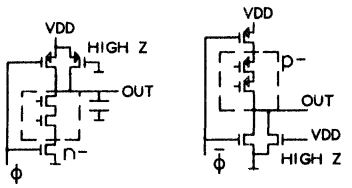


Fig. 15. Dynamic CMOS for low operating frequency.

Normally, there will be no charge redistribution between the precharged node and the logic tree nodes controlled by section inputs. This occurs because these inputs are set up during the precharge phase and, therefore, the logic tree nodes will also be precharged. Yet, some charge redistribution effect will exist, if the precharge period after input set up is too small. This extra period of precharge generally does not result in speed limitation for the pipelined system due to the small capacitances of the logic trees.

For the internal inputs, such attenuation of the charge redistribution does not exist, since these inputs are set up only after the precharge period. In this case, the charge redistribution must be minimized by layout and by proper logic tree arrangement. The transistors driven by internal inputs must be placed as far as possible from the output storage node.

B. Leakage and Noise Margin

Another limitation of the dynamic CMOS techniques is the leakage of the storage nodes. Due to clock feedthrough, power supply variation, noise, etc., the inputs of the dynamic block can be altered from the ideal zero and V_{DD} values. Consequently, the logic transistors are driven to weak inversion. This leakage effect imposes a limit to the lowest operating frequency and to the noise margin of the circuit.

For lower frequency applications, a possible solution [8] is the addition of a high impedance transistor, as shown in Fig. 15.

V. Results

In Fig. 16, a microphotograph of the chip designed to characterize the NORA technique is shown. It contains serial full adders, subtractors, shift registers, a 4-bit serial-parallel multiplier and some special structures to analyze charge redistribution, leakage, and clock feedthrough. Fig. 17 shows a NORA serial full adder containing only 32 transistors. The pipelined output sum is generated using only 20 transistors, compared with 28 if conventional CMOS is employed. The circuit area is $130 \times 318\ \mu m^2$, giving a density of 770 transistors/mm^2 in a 5 μm technology, which compare favorably with an NMOS solution. The threshold levels of the n-type and p-type devices

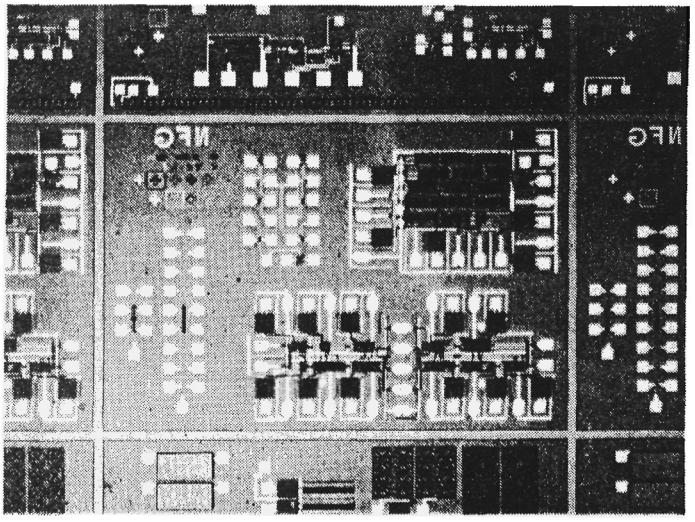


Fig. 16. Microphotograph of the chip to characterize the NORA-CMOS technique.

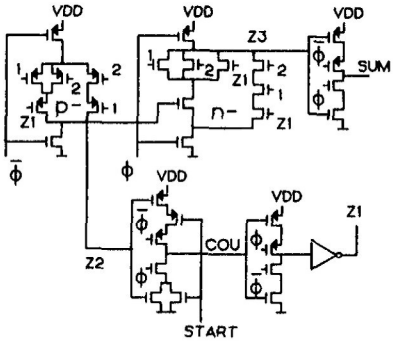


Fig. 17. NORA-CMOS serial full-adder—circuit diagram.

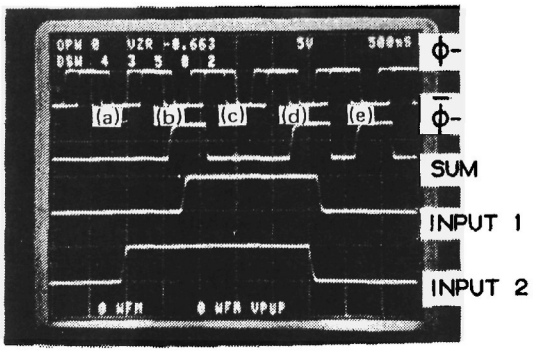


Fig. 18. NORA-CMOS serial full-adder—experimental results for very large clock skew. (a) 0 + 0 + 0. (b) 1 + 0 + 0. (c) 1 + 1 + 0. (d) 1 + 1 + 1. (e) 0 + 0 + 1.

are +1 V and -1 V, respectively. Fig. 18 shows experimental results for very large clock skew = 150 ns at 1 MHz clock frequency, without disturbing the circuit operation. Notice that during the evaluation phase $\phi = 1$ $\bar{\phi} = 0$, the results are obtained and then hold constant until the end of the transfer phase $\phi = 0$ $\bar{\phi} = 1$. Also from experimental results, the minimum working frequency was less than 1 kHz at room temperature, indicating that the current leakage due to weak inversion is not a critical limitation. The measured power disssipation of the serial full-adder was 17 μW/MHz at a supply voltage of 5 V. The circuits were designed for a maximum operating frequency of 14 MHz, and the devices have been tested on the wafer probe up to 10 MHz. More careful measurements about speed and noise margin are under investigation and will be presented in a later publication.

VI. Conclusion

A new dynamic CMOS technique has been presented. The NORA technique provides high logic flexibility, high speed circuits, and compact chip areas. A new concept of latch control by only one clock phase was theoretically and experimentally demonstrated. By this concept the critical clock skew specification of the conventional CMOS technique is completely eliminated. This simplifies the design and greatly increases the reliability, feasibility, and testability of CMOS circuits, The NORA technique also provides very high density layouts, which compare favorably with NMOS solutions.

References

[1] M. Shoji, "Electrical design of BELLMAC-32A microprocessor," in *Proc. IEEE Int. Conf. Circuits Comput.*, 1982, pp. 112–115.

[2] W. M. Pensey and L. Lau, *MOS Integrated Circuits.* New York: Van Nostrand, 1972, pp. 260–282.

[3] E. Hebenstreit and K. Horninger, "High-speed programmable logic arrays in ESFI SOS technology," *IEEE J. Solid-State Circuits*, vol. SC-11, pp. 370–374, June 1976.

[4] R. H. Krambeck, C. M. Lee, and H. S. Law, "High-speed compact circuits with CMOS," *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 614–619, June 1982.

[5] Y. Suzuki, K. Odagawa, and T. Abe, "Clocked CMOS calculator circuitry," *IEEE J. Solid-State Circuits*, vol. SC-8, pp. 462–469, Dec. 1973.

[6] N. F. Goncalves and H. De Man, "n-p-CMOS: A racefree dynamic CMOS technique for pipelined logic structures," in *ESSCIRC Dig. Tech. Papers*, Sept. 1982, pp. 141–144.

[7] H. De Man, D. Dumlugol, P. Stevens, G. Schrooten, and I. Bolsens, "Logmos: A transistor oriented logic simulator with assignable delays," in *Proc. IEEE Int. Conf. Circuits Comput.*, 1982, pp. 42–45.

[8] R. G. Stewart, "High density CMOS ROM arrays," *IEEE J. Solid-State Circuits*, vol. SC-12, pp. 502–506, Oct. 1977.

Cascode Voltage Switch Logic: A Differential CMOS Logic Family

LAWRENCE G. HELLER, WILLIAM R. GRIFFIN
IBM GENERAL TECHNOLOGY DIVISION, ESSEX JUNCTION, VT
JAMES W. DAVIS, NANDOR G. THOMA
IBM SYSTEM PRODUCTS DIVISION, BOCA RATON, FL

IMPORTANT CRITERIA for choosing a suitable VLSI logic family include power, delay, logic circuit density, device/process complexity, and compatibility with design automation tools. This paper will describe a differential CMOS logic family – Cascode Voltage Switch Logic (CVSL).

Logic design leverage is achieved in CVSL by cascoding differential pairs of MOS devices into powerful combinational logic tree networks capable of processing complex Boolean logic functions within a single circuit delay. Logic trees with N-high cascoding of differential pairs of NMOS devices are capable of processing Boolean functions with up to (2^N-1) input variables. CVSL has been found to offer a performance advantage of up to 4X compared to CMOS/NMOS primitive NAND/NOR logic families, while maintaining the expected low power characteristics of CMOS circuitry. Potentially, CVSL is twice as dense as primitive NAND/NOR logic, and is compatible with existing design automation tools. Combinational logic trees can be designed in cascoded high-peformance NMOS devices with unstacked PMOS devices used sparingly as pull-up devices in load and buffer circuitry. Optimization of the PMOS devices and the criticality of the PMOS to NMOS spacing can therefore be relaxed, relieving the device/process complexity burden for CVSL designs.

The CVSL circuit concept, in its differential form, is illustrated in Figure 1. Depending on the differential inputs, either node N1 or node N2 is pulled down by the NMOS combinational logic tree network. Regenerative action sets the PMOS latch to static outputs $Q,\overline{Q}$ of full differential V_H and ground logic levels. The logic trees are free of direct current after the latch sets. Since the inputs drive only the NMOS tree devices, input gate capacitance loading is typically a factor of 3X smaller than CMOS circuits that require complementary N-channel and P-channel devices to be driven.

The logic trees networks can be designed automatically using existing logic minimization algorithms[1]. An example of the efficiency of a differential CVSL circuit, requiring 12 devices, is shown in Figure 2. Device redundancy is naturally reduced by the functional power of the differential logic trees. Implementation of the same Boolean function in primitive CMOS NAND gates, as indicated in Figure 3, required 5 NAND gates and 28 devices, not counting the additional inverters to provide the complementary inputs. It will be noted that performance leverage in CVSL is enhanced by a reduction in the number of circuit delays compared to primitive logic. The Boolean function Q can also be implemented with 16 devices in a cascoded fully CMOS circuit.

The differential version of Figure 2, however, requires 6 fewer large P-channel devices and has considerably less input capacitance.

An experimental masterslice chip, personalized at the metal and contact levels, has been designed. The chip contains 10,880 *brickwalled* NMOS differential pairs, forming 1088 CVSL trees. The image design is compatible with existing automatic placement[2] and wiring algorithms. A photomicrograph of the chip, implemented in a 2μm CMOS technology, is shown in Figure 4. The macro in the upper right-hand corner contains 150 CVSL logic trees, which were designed top-down and automatically placed and wired from a high level language description. Other experiments include a 4b carry look-ahead ALU and several ring oscillators. Delays in the ring oscillators loaded with 0.3pF of wiring capacitance were measured in the 1 to 2ns range.

Performance of the circuit of Figure 1 is limited by the set time of the PMOS latch. A high-performance clocked CVSL circuit is illustrated in Figure 5. The outputs $Q,\overline{Q}$ are precharged low when the clock phase PC is low and data are propagated in a *domino* mode[3] when PC goes high. Feedback devices T1, T2 hold the internal nodes N1, N2 statically high prior to switching within the logic tree. The feedback devices reduce charge sharing noise within the tree and improve the noise margin, with only a small sacrifice in performance. During switching either N1 or N2 is pulled down and either device T1 or T2 is shut off. No direct current flows after switching. The logic invert function is implicit in this clocked differential CVSL circuit, a clear advantage over other incomplete domino type logic families. All logic functions can be implemented, including, for example, the XOR. A 4-way clocked XOR is illustrated in Figure 6.

Acknowledgments

The authors would like to acknowledge the work of R.S. Chelemer and J.J. Preli for their automatic placement and wiring support, the logic synthesis effort of R.D. Kilmoyer, and many others for helpful discussions.

---

[1] Brayton, R.K. and McMullen, C., "Decomposition and Factorization of Boolean Expressions," *Proc. IEEE ISCA*, Rome, Italy; May, 1982.

[2] Kirkpatrick, S., Gelatt, Jr., C.D. and Vecchi, N.P., "Optimization by Simulated Annealing", *Science*; May 13, 1983.

[3] Krambeck, R.H., Lee, C.M. and Law, H.S., "High-Speed Compact Circuits with CMOS," *IEEE J. Solid State Circuits*, Vol. SC-17, No. 3; June, 1982.

Reprinted from *Digest of Technical Papers, IEEE International Solid-State Circuits Conference*, pp. 16-17, February 1984.

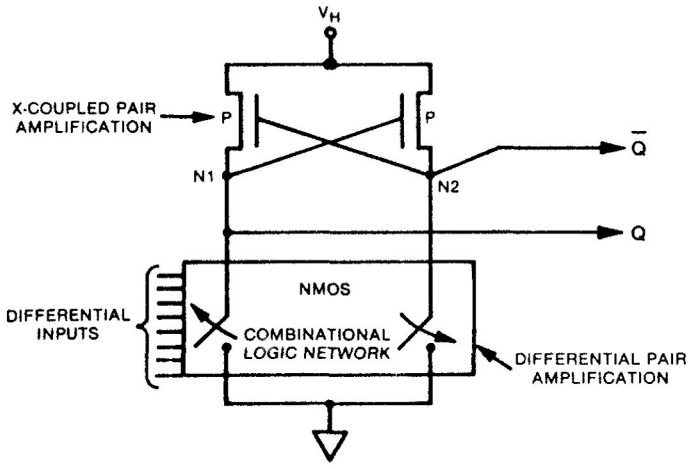


FIGURE 1–Basic CVSL circuit.

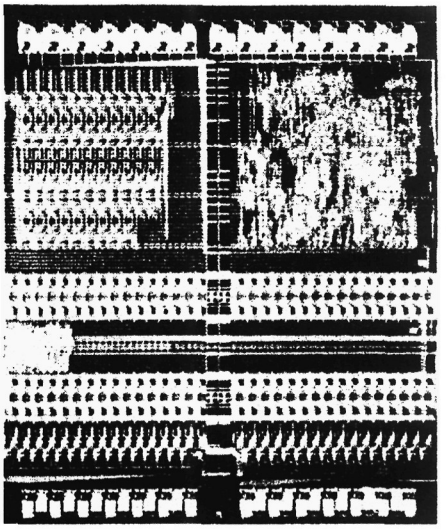


FIGURE 4–Photomicrograph of masterslice chip.

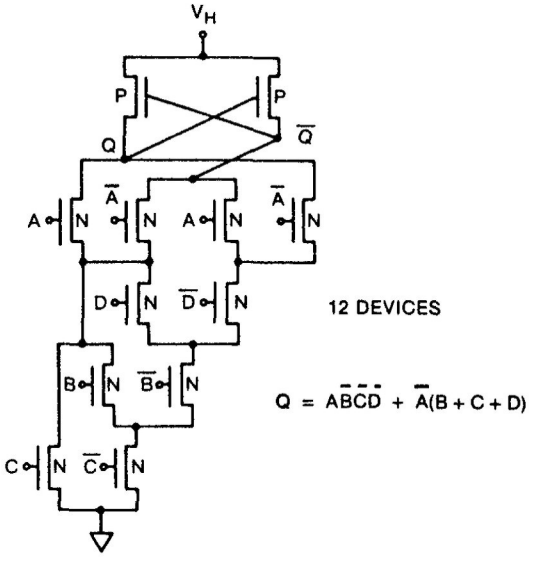


FIGURE 2–CVSL implementation of Q.

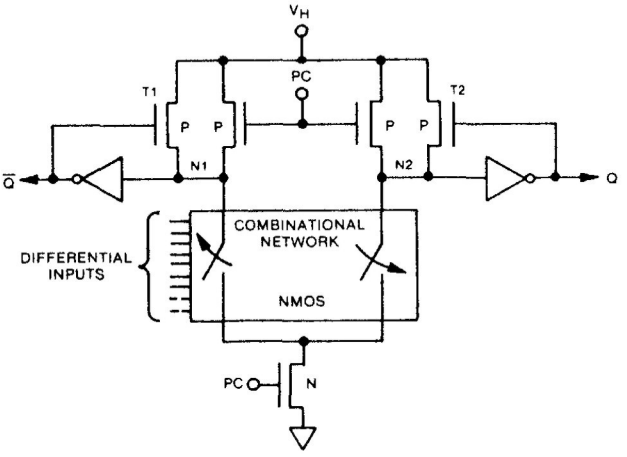


FIGURE 5–Clock CVSL.

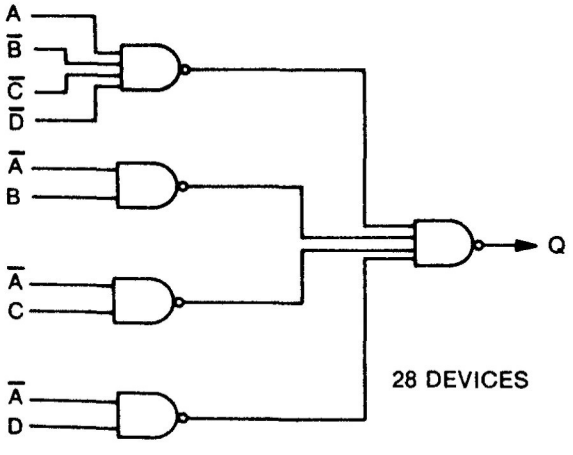


FIGURE 3–CMOS NAND implementation of Q.

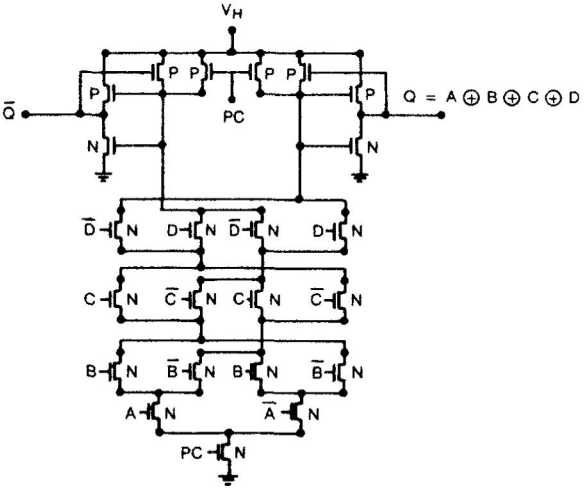


FIGURE 6–Clocked CVSL 4-way XOR.

Differential Split-Level CMOS Logic for Subnanosecond Speeds

LEO C. M. G. PFENNINGS, WIM G. J. MOL, JOSEPH J. J. BASTIAENS, AND JAN M. F. VAN DIJK

Abstract—**Subnanosecond gate delays (0.8 n) have been measured on complex logic gates (e.g., sum functions of a full adder) designed in the differential split-level (DSL) CMOS circuit technique. This high speed has been achieved by reducing the logic swing (2.4 V) on interconnect lines between logic gates, by using current controlled cascoded cross-coupled NMOS–PMOS loads, by using combined open NMOS drains as outputs, and by employing shorter channel lengths ($L_{eff}=1\mu m$) for the NMOS devices in the logic trees with reduced maximum drain–source voltages to avoid reliability problems. Extra ion implantation protects these transistors from punchthrough.**

I. Introduction

THE GENERAL rule: "Enhancement Depletion NMOS is faster than CMOS" was a challenge to explore speed improvements in CMOS circuit techniques. Differential split-level (DSL) [1] CMOS logic makes a compromise between the static power dissipation of *E/D* NMOS and the dynamic power dissipation of CMOS.

This paper will describe the switching behavior and the speed improvements owing to the DSL circuit technique. This circuit technique shows similarity with differential cascode voltage switch logic (CVSL) [2] but the electrical behavior of the DSL circuit technique is essentially different.

The DSL circuit technique was implemented in a double-metal 2.5-μm CMOS n-well process with conventional 2.5-μm projection lithography, except for the polysilicon. In this mask only the gate length of the n-channel transistors in the logic trees is decreased to 1.5 μm ($L_{eff}=1$ μm), while the other polysilicon details and pitches remain constant. Therefore a stepper exposure is used for the polysilicon definition. An extra deep boron implant protects the short-channel NMOS transistors from punchthrough.

DSL incorporates a reduced maximum drain–source voltage $V_{DS\,max}$ in the logic trees resulting in less hot-electron-induced degradation of these devices. Decreasing $V_{DS\,max}$ reduces the lateral electric field near the drain edge of these NMOS transistors [3], [4]. This allows the use of shorter channel lengths without lifetime reduction.

Manuscript received March 28, 1985; revised May 28, 1985.
The authors are with Philips Research Laboratories, Prof. Holstlaan, P.O. Box 80.000, 5600 JA Eindhoven, The Netherlands.

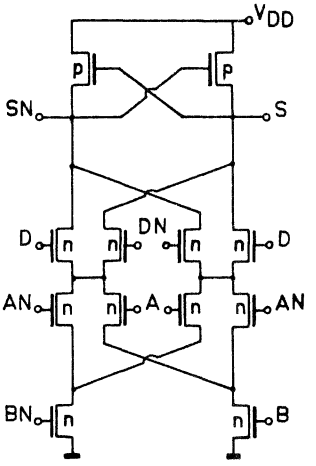


Fig. 1. Sum part of a differential CVSL full adder.

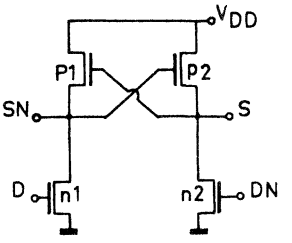


Fig. 2. Differential CVSL basic circuit.

II. Short Summary of CVSL

Fig. 1 shows the circuit diagram of the sum function of a full adder designed in differential cascode voltage switch logic. The cross-coupled PMOS loads and the differential logic NMOS trees are typical for this circuit technique. To explain the switching behavior of the CVSL circuit technique we replace the differential logic NMOS trees by two NMOS transistors as shown in Fig. 2.

Now suppose we switch input *D* from a low to a high level, starting with input *D* low and input *DN* high. Then node *SN* is at a high level of V_{DD} and node *S* at a low level of 0 V so PMOS *P*1 is on and *P*2 is off. If we now switch the inputs *D* and *DN* then NMOS *N*1 turns on and *N*2 turns off. This is ratioed logic because transistor *N*1 has to discharge node *SN*, while *P*1 is still on. *P*1 switches off, after *P*2 has switched on and node *S* has reached a high level. So during switching both *N*1 and *P*1 (or *N*2 and *P*2 depending on the input transition) conduct, causing relatively large current spikes and additional delay.

Reprinted from *IEEE Journal of Solid-State Circuits*, Vol. SC-20, No. 5, pp. 1050-1055, October 1985.

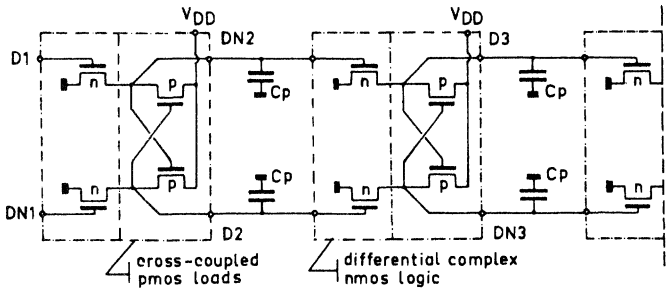


Fig. 3. Serial chain of differential CVSL circuits.

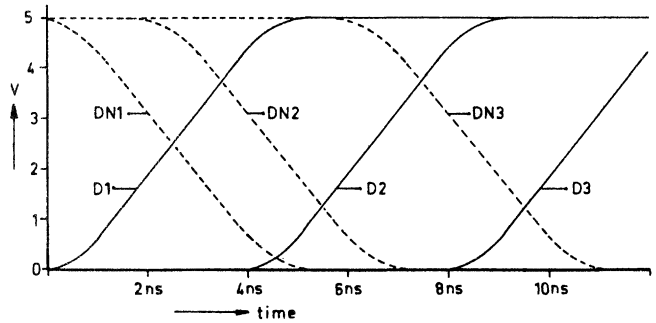


Fig. 4. Switching behavior of the differential CVSL serial chains.

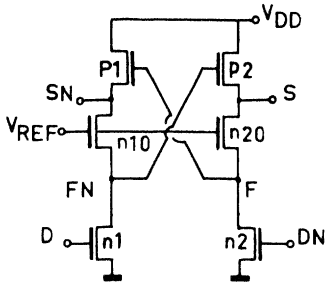


Fig. 5. Differential split-level circuit principle.

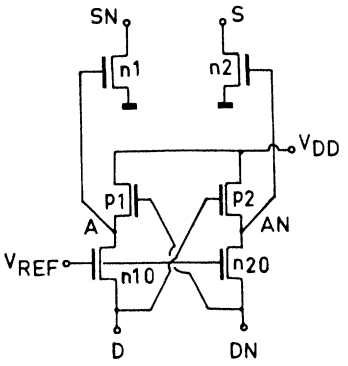


Fig. 6. DSL basic circuit.

Fig. 3 shows a part of a chain of differential CVSL circuits. The two NMOS transistors of the NMOS logic segments represent the CVS tree combinatorial logic network, i.e., full adder. The capacitances C_p represent the parasitic capacitance between the circuits.

Fig. 4 shows the switching behavior of this chain. It shows that although there is a simultaneous switching of the inputs $D1$ and $DN1$, there is a skew between the complementary output nodes $DN2$ and $D2$, and $DN3$ and $D3$, respectively.

III. Differential Split-Level (DSL) CMOS Logic

The DSL principle is shown in Fig. 5. Two extra NMOS transistors $N10$ and $N20$ are placed between the PMOS part and the logic NMOS part. Their gates are controlled by a reference voltage, which must be equal to half V_{DD} plus the threshold voltage of the NMOS transistors to guarantee optimum circuit operation (see below). The gates of the PMOS transistors $P1$ and $P2$ are now connected to F and FN instead of the outputs S and SN. Now we again switch input D from a low to a high-level starting with input D low and DN high. Then node SN has a high level of V_{DD} and node S has a low level. The reference voltage determines the high-level at node FN to be half V_{DD}. Node F has a low level of about 100 mV, because PMOS $P2$ is weakly on. This causes static power dissipation. The NMOS transistor $N10$ is cut off, which causes a high impedance to V_{DD} for node FN while there is a low impedance to V_{DD} for node SN. If we now switch the inputs D and DN then NMOS $N1$ turns on and $N2$ turns off.

The half V_{DD} level at node FN will immediately be discharged and PMOS $P2$ turns more on to its high drive state. At the same time nodes S and F start rising because PMOS $P2$ was already partly on, causing PMOS $P1$ to switch faster to its low drive state.

Now let us consider ways to improve further the switching speed of this circuit technique. The maximum drain–source voltage of only half V_{DD} on nodes FN and F allows the channel length of the NMOS logic transistors to be reduced. Because of the low voltage swing on nodes FN and F, it is preferable to use these nodes as outputs and inputs of the gates, thereby reducing the delay due to wiring capacitance. This technique of separating gates is also used in some bipolar techniques.

Fig. 6 shows the reconfiguration of the DSL circuit. At the inputs D and DN, we now have current controlled cascoded cross-coupled NMOS–PMOS loads and the outputs SN and S are the open drains of the logic NMOS transistors. The internal gate signals A and AN of this figure correspond with the outputs S and SN of Fig 5.

Application of the differential split-level logic technique in a complete full adder is shown in Fig. 7. At the bottom of this circuit diagram we have three complementary input circuits consisting of cascoded cross-coupled NMOS–PMOS loads. The interconnections between nodes D, DN, A, AN and B, BN of the input circuits and the gates of the logic NMOS trees are not drawn in this diagram.

Fig. 8 shows simulation results for the full adder of Fig. 7. Shown are the inputs AI and ANI, the internal gate signals A and AN, and the open drain outputs S and SN. The inputs and outputs, respectively, are driven and loaded by other full adders. The effective gate lengths of the p-channel transistors are 2.1 μm and of the n-channel transistors 2 μm for the reference transistors and 1 μm for the logic transistors. The delay between ANI, AI and SN,

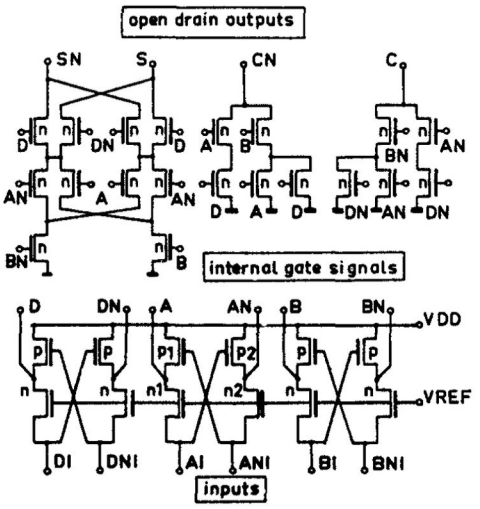


Fig. 7. Complete full adder in DSL.

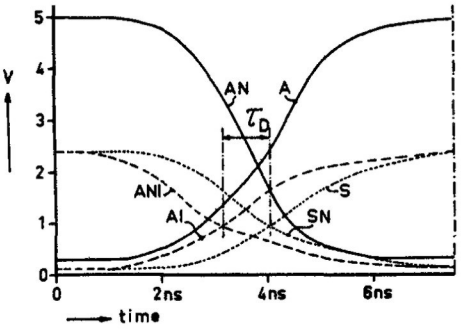


Fig. 8. Timing diagram of a DSL full adder.

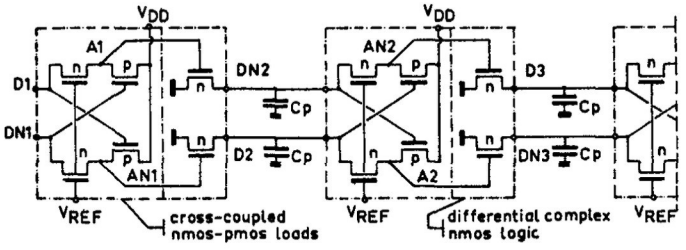


Fig. 9. Serial chain of DSL circuits.

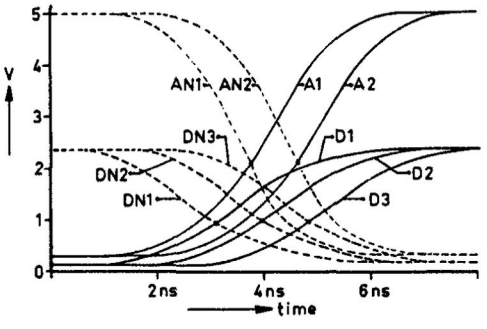


Fig. 10. Switching behavior of the DSL serial chain.

S equals 0.8 ns. It can be seen that complementary signals switch simultaneously.

Fig. 9 shows a part of a DSL CMOS gate chain. The two NMOS transistors of the NMOS logic segments represent the combinatorial logic network, i.e., full adder of Fig. 7. Simulation results of this chain are shown in Fig. 10.

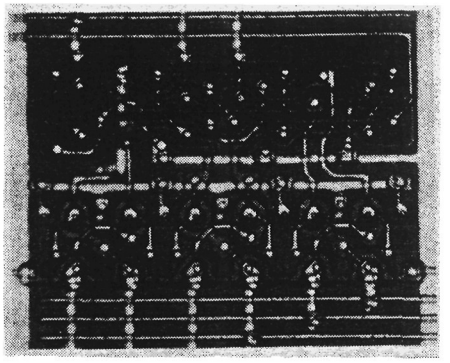


Fig. 11. Photomicrograph of a DSL full adder.

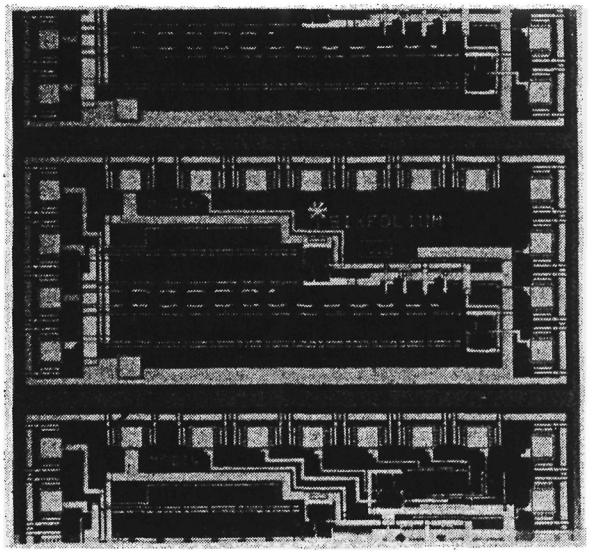


Fig. 12. Photomicrograph of a two-chain test structure.

$DN1$, $D2$, $DN2$ and $D3$, $DN3$ are the low-voltage-swing output and input node signals. $A1$, $AN1$ and $A2$, $AN2$ are the internal gate signals with a full V_{DD} swing.

IV. Experimental and Simulation Results

Fig. 11 shows a photomicrograph of a full adder. The lower part of the layout shows the three complementary input circuits with the NMOS reference transistors and the PMOS cross-coupled loads. The upper part shows the NMOS logic trees. The size of the full adder is 142 μm $\times$ 90 μm. The full adder is processed in a 2.5-μm double-metal n-well CMOS process.

Two full adder chains have been implemented on the testchip shown in Fig. 12. This test chip consists of one chain of six and one of twelve full adders in series. Both chains have the same input and output buffer. Since the delay of a full adder is not the same for every input, six of these two-chain test structures have been implemented on a chip to measure the various delays. A differential measurement between the outputs of a two-chain test structure gives the total delay of six full adders.

Fig. 13 shows an oscillogram of this measurement for the sum delay. The output signals are measured with probes with an attenuation factor of 10; the horizontal scale is 5

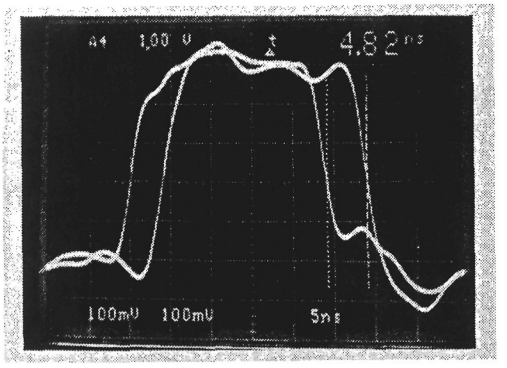


Fig. 13. Sum delay of six full adders.

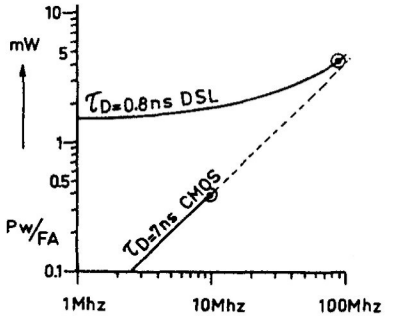


Fig. 14. Power as a function of frequency for DSL and CMOS.

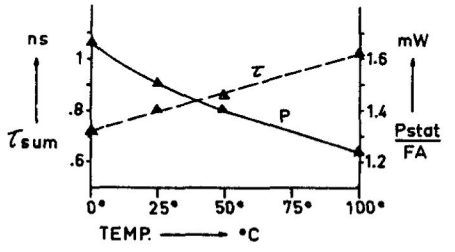


Fig. 15. Sum delay and static power as a function of the temperature.

ns/div. A difference of 4.8 ns between the outputs of the two chains was measured on the test structure, where the S and SN nodes are connected to the AI and ANI inputs, see Fig. 7. So a propagation delay of 0.8 ns for the sum function of a full adder is obtained. The static power dissipation was 1.5 mW for each full adder at 5 V V_{DD}. Similar test chains have been implemented with conventional static CMOS circuitry.

Fig. 14 shows the simulation results of a comparison of the power dissipation as a function of the frequency for conventional static CMOS and DSL fabricated in the same CMOS process. The power dissipation per full adder is plotted in milliwatts versus the frequency in megahertz. The dots indicate the maximum oscillation frequency of a seven-stage full adder ring oscillator, with both the sum and the carry switching. These maximum operating frequencies are 10 and 89 MHz, respectively, for conventional CMOS and for DSL CMOS. The propagation delay of the DSL full adder is about one tenth that of the conventional CMOS full adder, 0.8–7 ns. The simulated results shown in this graph are in agreement with the measured values. The dynamic power dissipation of the conventional CMOS full adder is 40 μW/MHz while for the DSL full adder 30

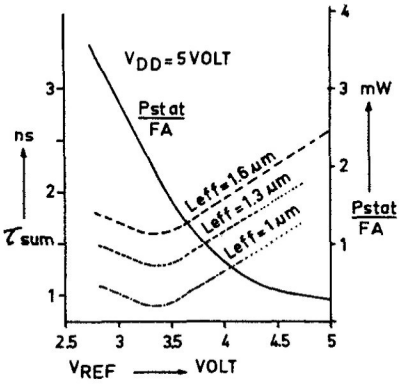


Fig. 16. Influence of V_{ref} on T_{sum} and P_{stat}.

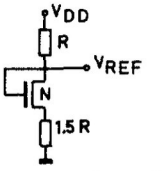


Fig. 17. Reference voltage circuit diagram.

μW/MHz is obtained. The dynamic power dissipation for the differential CVSL full adder was 80 μW/MHz with a propagation delay of 4 ns (not shown in the graph).

However, a disadvantage of the DSL full adder is its static power dissipation of 1.5 mW. Both the static power dissipation and the sum propagation delay depend on the operating temperature. Fig. 15 shows these simulated relations.

Fig. 16 shows the measured influence of the reference voltage on the sum propagation delay and the static power dissipation of the full adder, at a fixed $V_{DD} = 5$ V. The channel length of the NMOS transistors of the differential logic trees affects neither the static power dissipation nor the optimum reference voltage, which equals half the V_{DD} voltage plus the threshold voltage of the NMOS reference transistor. The static power dissipation is proportional to $(V_{DD} - V_{ref} + V_{tn} + V_{tp})^2$. Fig. 17 shows a simple example of a V_{ref} circuit for a 5 V $\pm$ 20-percent V_{DD}. The resistors can be polysilicon resistors. The cross talk from the inputs on the reference NMOS transistors does not affect the V_{ref}, because the inputs are complementary.

Experiments and simulations show that the decrease of the propagation delay by a factor of ten is the result of several steps: a factor of two is obtained by the use of differential logic trees (NMOS mostly) while application of the DSL circuit technique yields another factor of two. A factor of 2.5 is further obtained by the shorter channel length of the NMOS transistors in the logic trees.

V. Technology

The DSL circuits have been fabricated in a 2.5-μm n-well double-metal CMOS process (Table I). A -2.5-V back-bias voltage was used to improve the performance of the 1-μm L_{eff} logic NMOS transistors.

Fig. 18 shows measurements of the punchthrough voltage (VPT) [5], [6] of the n-channel transistors as a function

TABLE I

SUMMARY OF PROCESS—RELATED INFORMATION

• LITHOGRAPHY:	CONVENTIONAL EXPOSURE EXCEPT POLYSILICON STEPPER EXPOSURE
• TECHNOLOGY	2.5 μm DOUBLE METAL N-WELL CMOS
• GATE LENGTH:	
P-CHANNEL	3 μm. Leff. = 2.1 μm
N-CHANNEL	2.5 μm. Leff. = 2.0 μm
N-CHANNEL (LOGIC TREES)	1.5 μm. Leff. = 1.0 μm
• GATE OXIDE	500 A°
• FIELD OXIDE	6000 A°
• THRESHOLD VOLTAGES:	
V_{TP} 20/3	−1.1 V.
V_{TN} 20/2.5	1.0 V. (V_{BS} = −2.5 V.)
• DESIGN RULE PITCHES	INCLUDING CONTACTS
METAL I	7.5 μm
METAL II	8 μm
POLYSILICON	8.75 μm

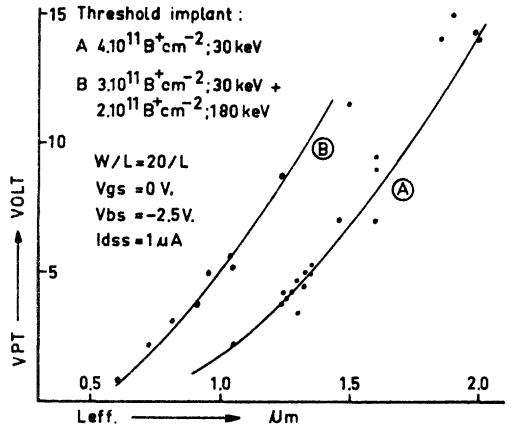


Fig. 18. Punchthrough voltage versus channel length.

of the effective channel length for two different threshold correction implants. The punchthrough voltage of the NMOS transistors is defined here as the drain–source voltage at which the drain source current of a MOSFET with grounded gate equals 1 μA ($W/L = 20/L$). Such a leakage current is allowed in the DSL circuit technique. For our 2.5-μm n-well CMOS process the threshold correction implant is 4.10^{11} B^+ cm^{-2} at 30 keV, see curve *A*. Curve *B* shows the result of a modified implant. The deep implant at 180 keV suppresses the bulk punchthrough and the increased total dose will also increase the surface concentration and thereby suppresses the surface punchthrough as well. The 1-μm device from curve *B* has a VPT of 5 V which corresponds with a maximum power supply voltage of 10 V in the circuitry.

It is well known that parameters like threshold voltage and transconductance of short-channel transistors may degrade because of hot-electron effects [7], [3], [4]. This degradation depends on V_{ds} and V_{gs}. The worst-case operating condition for this degradation occurs at $V_{gs} = 1/2$ V_{ds}.

Fig. 19 shows the measured lifetime of such a 1-μm NMOS transistor as a function of the drain–source voltage. The lifetime is defined as the time until a threshold shift of 100 mV appears. This curve shows that a transistor with a $L_{\text{eff}} = 1$ μm used in the differential logic trees will not suffer from hot-electron degradation. Even at power-

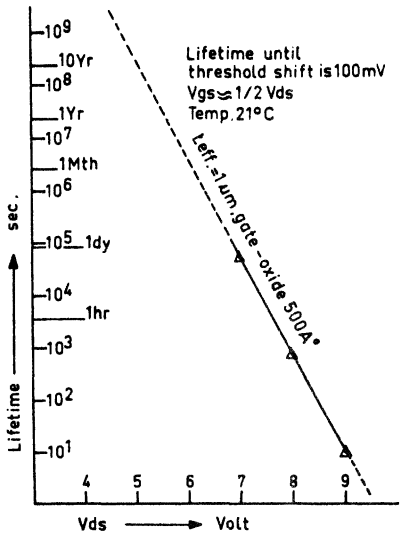


Fig. 19. Lifetime NMOS with $L_{\text{eff}} = 1$ μm.

supply voltages of 10 V the maximum V_{ds} will still be only 5 V resulting in a lifetime in excess of 10 years.

VI. Applications

The DSL circuit technique is advantageous in high-speed complex logic circuits, i.e., multipliers, comparators, error-correcting circuits etc. In applications where the data rate is high, standard CMOS has a high dynamic power dissipation, see Fig. 14. In that case DSL compares favorably with its power-delay product.

The conversion from standard CMOS logic circuits into DSL and vice versa is straight forward. The gates of the NMOS logic transistors of the first DSL gate can simply be driven by the full V_{DD} swing of standard CMOS. For the conversion back to standard CMOS the internal node of the last DSL gate is used to drive a standard CMOS gate, because this node has a full V_{DD} swing.

Shorter channel lengths of the NMOS transistors in the logic trees, possible because of the reduced drain source voltage, can be used in every CMOS process, down to the limits of the lithography. Fig. 16 illustrates the achievable gain for different channel lengths.

In the near future electron beam exposure could be used for the submicrometer gate length in a stepper process after down scaling.

VII. Conclusion

The differential split-level (DSL) CMOS circuit technique has about 10× shorter propagation delay as compared to conventional CMOS in the same process, but at the cost of static power dissipation. DSL can be used in a conventional process in combination with conventional as well as other CMOS techniques on the same chip to combine fast subnanosecond and slower circuitry.

Differential split-level CMOS logic is a circuit technique which allows, in any given technology, shorter channel length owing to a reduced drain–source voltage.

ACKNOWLEDGMENT

The authors would like to thank A. T. Van Zanten and H. J. M. Veendrick for their contributions to this subject.

REFERENCES

[1] L. C. M. G. Pfennings, W. G. J. Mol, J. J. J. Bastiaens, and J. M. F. Van Dijk, "Differential split-level CMOS for sub-nanosecond speeds," in *ISSCC Dig. Tech. Pap.*, vol. XXVIII, 1985, pp. 212–213, 351.

[2] L. G. Heller and J. W. Davis, "Cascode voltage switch logic," in *ISSCC Dig. Tech. Pap.*, vol. XXVII, 1984, pp. 16, 17.

[3] C. Hu, "Hot-electron effects in MOSFETs," in *Tech. Dig. IEDM*, 1983, pp. 176–181.

[4] C. Hu, S. C. Tam, F.-C. Hsu, P.-K KD, T.-Y. Chan, and K. W. Terrill, "Hot electron-induced MOSFET degradation-model, monitor, and improvement," *IEEE Trans. Electron Devices*, vol. ED-32, p. 375, 1985.

[5] F. M. Klaassen, "Design and performance of micro-size devices," *Solid-State Electron.* vol. 21, pp. 565–571, 1978.

[6] B. Eitan and D. Frohman-Bentchkowsky, "Surface conduction in short channel MOS devices as a limitation to VLSI scaling," *IEEE Trans. Electron Devices*, vol. ED-29, p. 154, 1982.

[7] ——, "Hot electron injection into the oxide in n-channel MOS devices," *IEEE Trans. Electron Devices*, vol. ED-28, p. 328, 1981.

A Comparison of CMOS Circuit Techniques: Differential Cascode Voltage Switch Logic Versus Conventional Logic

KAN M. CHU AND DAVID L. PULFREY, MEMBER, IEEE

Abstract—**Differential cascode voltage switch (DCVS) logic is a CMOS circuit technique which has potential advantages over conventional NAND/NOR logic in terms of circuit delay, layout density, power dissipation, and logic flexibility. In this paper a detailed comparison of DCVS logic and conventional logic is carried out by simulation, using SPICE, of the performance of full adders designed using the different circuit techniques. Specifically, comparisons are made between a static full CMOS design and two different implementations of static DCVS circuits, and, in the dynamic case, between two conventional NORA implementations and DCVS forms of both NORA and DOMINO logic. The parameters compared are: input gate capacitance, number of transistors required, propagation delay time, and average power dissipation. In the static case, DCVS appears to be superior to full CMOS in regards to input capacitance and device count but inferior in regards to power dissipation. The speeds of the two technologies are similar. In the dynamic case, DCVS can be faster than more conventional CMOS dynamic logic, but only at the expense of increased device count and power dissipation.**

I. INTRODUCTION

DIFFERENTIAL cascode voltage switch (DCVS) logic is a recently proposed CMOS circuit technique which is claimed to have advantages over traditional NAND/NOR circuit techniques in terms of circuit delay, power dissipation, layout area, and logic flexibility [1]. DCVS also has an inherent self-testing property which can provide coverage of both stuck-at and dynamic faults [2]. A further attraction of DCVS circuits is the fact that they can be readily designed using straightforward procedures based on Karnaugh maps (K-maps) and tabular methods [3].

All these worthwhile features would appear to make DCVS logic a very promising CMOS circuit technique. To investigate this possibility, we have compared DCVS logic and more conventional CMOS logic forms using the full-adder circuit as a test vehicle. The full adder is suited to this purpose as it is a common, yet reasonably complex, building block in digital circuits. The comparison reported here uses SPICE simulations to assess the performance parameters of area, input loading, speed, and power dissipation. Area is represented by the number of transistors needed to implement the adder and loading is quantified in terms of the input gate capacitance. Speed is assessed by simulating the worst-case propagation time. Power dissipation is computed at the maximum frequency of operation of each circuit.

Manuscript received August 29, 1986; revised January 26, 1987. This work was supported by the Natural Sciences and Engineering Research Council of Canada.

The authors are with the Electrical Engineering Department, University of British Columbia, Vancouver, B.C. V6T 1W5, Canada.

IEEE Log Number 8714872.

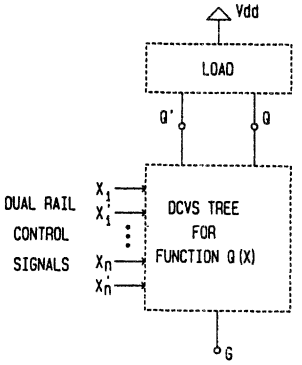


Fig. 1. Block diagram of a DCVS circuit. The load circuitry is connected to nodes Q and Q'.

II. CIRCUIT TECHNIQUES FOR DCVS LOGIC

The basic DCVS circuit comprises two parts: a binary decision tree and a load (see Fig. 1). The tree is specified such that:

1) when the input vector $x = (x_1, \cdots, x_n)$ is the true vector of the switching function $Q(x)$, then the output Q is disconnected from node G and the node Q' is connected to G; and
2) when $x = (x_1, \cdots, x_n)$ is the false vector of $Q(x)$, then the reverse holds.

There are two trees required to implement a full adder, one to perform the sum and one to perform the carry function (see Fig. 2). These circuits, which were designed using the K-map procedure described in [3], are used as the tree circuits for all the DCVS circuit forms examined in this paper. The various DCVS forms differ in their load circuitry, as is now described.

The load for a static DCVS circuit is the simple latch shown in Fig. 3. Depending on the differential inputs, either node Q or Q' is pulled down by the DCVS tree network. Regenerative action sets the PMOS latch to static outputs Q and Q' of V_{DD} and ground or vice versa. The

Reprinted from *IEEE Journal of Solid-State Circuits*, Vol. SC-22, No. 4, pp. 528-532, August 1987.

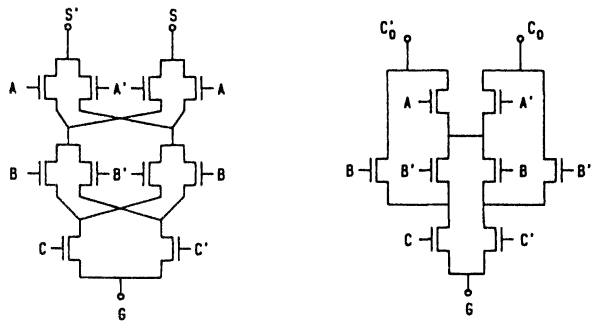


Fig. 2. The DCVS trees for a full adder. (a) The circuit providing the sum, $S(A,B,C) = A + B + C$. (b) The circuit yielding the carry, $C_o(A,B,C) = AB + BC + CA$.

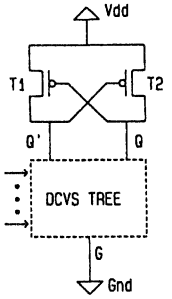


Fig. 3. The load for a static DCVS circuit.

logic trees do not pass any direct current after the latch sets.

A variation of this static DCVS circuit is the differential split-level (DSL) logic circuit [4] shown in Fig. 4. Two n-transistors $T3$ and $T4$ with their gates connected to a reference voltage V_{REF} are added to reduce the logic swing at nodes Q and Q'. If V_{REF} is set to $V_{DD}/2 + V_{th}$, where V_{th} is the threshold voltage of the n device, then the nodes Q and Q' are clamped at $V_{DD}/2$. Suppose node Q is pulled down from 2.5 V (i.e., assume $V_{DD} = 5$ V) to a low level. $T1$ switches from its low-current state to its high-current drive state very quickly, because $T4$ is initially OFF. The voltage on node f' goes up to 5 V because $T1$ is fully ON. Node Q' is raised up to 2.5 V until $T3$ is in the cutoff mode. DSL circuits would be expected to be about two times faster than standard DCVS circuits on account of the need for logic swings of only half the rail-to-rail voltage difference. This should result in a reduction by two times of the charges needed to be manipulated in the circuit.

Turning now to dynamic operation of DCVS circuits, consider first the DOMINO [5] configuration of Fig. 5 [1]. Nodes Q and Q' are precharged to high during the precharge phase ($\phi = 0$) and either node Q (node f) or Q' (f') discharges to low during the evaluation phase ($\phi = 1$). Transistor $T1$ (or $T2$) is a high impedance p transistor which serves as the feedback device to maintain the high logic level at node Q' (or Q), where charges may be lost due to charge sharing [6].

For dynamic operation of pipelined architectures, NORA (NO RACE) techniques [7] are suitable for implementing logical functions. In its original form the NORA structure consists of n- and p-logic gates to enhance logic flexibility. The p-logic gates usually cause long delay times and consume large areas. Using DCVS logic in the NORA technique will eliminate p-logic gates because of the inherent availability of complementary signals. The general structure of a DCVS NORA pipelined section consisting of only one dynamic gate is shown in Fig. 6. This type of circuit technique is suitable for use in a heavily pipelined logic design, as in the case, for example, of a newly developed 8×8 pipelined multiplier [8].

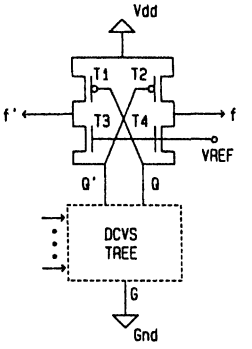


Fig. 4. The load for a static DSL circuit.

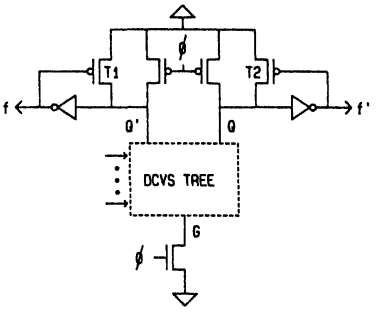


Fig. 5. The load and circuit arrangement for a DCVS DOMINO circuit.

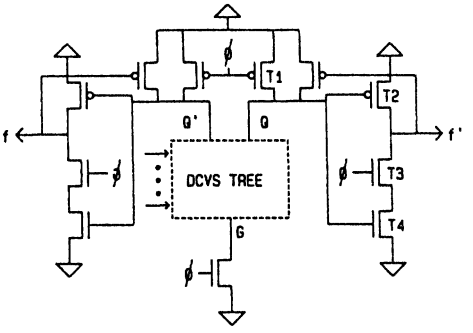


Fig. 6. The load and circuit arrangement for a DCVS NORA pipelined section.

As Fig. 6 indicates, the load circuitry is symmetrical, and thus, for analysis purposes, only one side of it need be considered. During the evaluation phase ($\phi = 1$), node Q is either floating or discharged depending on the inputs. The output register acts as a clocked inverter, and the output can be either high or low. During the precharge phase ($\phi = 0$), the ground path of the register is blocked. If the output resulting from the previous evaluation is high, then

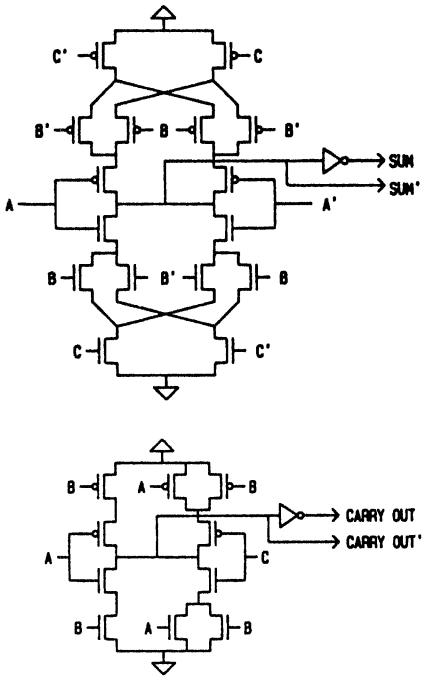


Fig. 7. Circuits for a static CMOS full adder.

the output continues to be high regardless of the voltage of Q. If the output is low (i.e., node Q has never been discharged) and transistor $T1$ is ON, then the output continues to be low because no charges can be added through $T2$. Thus for a ϕ section of a pipeline, the output changes freely when ϕ is high and is latched at the falling edge of ϕ.

III. Conventional CMOS Circuit Techniques

To provide a basis for comparison of the DCVS circuits described in Section II, conventional CMOS designs operating under static and dynamic conditions need to be considered.

The circuit used here for a static CMOS full adder is shown in Fig. 7. Two subcircuits are identified, one to generate the sum signal and one to generate the carry out signal. The three-way EXCLUSIVE-OR gate in the sum circuit has the highest stack level and largest parasitic capacitance, and thus determines the worst-case delay time of the adder. This circuit is relatively fast compared to other possible static full CMOS implementations because the complemented outputs are obtained through only one gate delay from the complementary inputs.

Two versions of conventional approaches to dynamic CMOS full-adder design were studied. One, a conventional NORA adder with serial n- and p-logic blocks, is shown in Fig. 8. The other circuit, a modified NORA adder [9], is shown in Fig. 9. It contains a special three-way XOR gate to generate the sum signal.

IV. Comparison of the Full Adders

To compare the performance of the various forms of full adders, each of the circuits described in Sections II and III was simulated using SPICE. The conventional CMOS circuits simulated were those shown in Figs. 7–9. Four different DCVS circuits, two static and two dynamic, were generated by connecting the full-adder tree of Fig. 2 to the load circuits shown in Figs. 3–6. The DCVS circuits were laid out on a Metheus λ700 workstation in accordance with design rules for the single-metal 3-μm CMOS process of Northern Telecom, Ottawa, Canada [10]. The areas occupied by the DCVS circuits were about 2.2×10^{-4} and 3.5×10^{-4} cm^2 for the static and dynamic versions, respectively. The results of SPICE simulations from the schematics of all the circuits are summarized in Table I.

The input gate capacitance gives a measure of the input loading of the circuit. This parameter is, for the case of transistors of fixed length (3 μm in this case), determined by the number of transistors and their widths. A general guideline used in the first iteration of a design was to size the transistors in a tree network such that the equivalent conductance of any single discharging path was the same as the conductance of a minimum-size ($W = 3$ μm in our

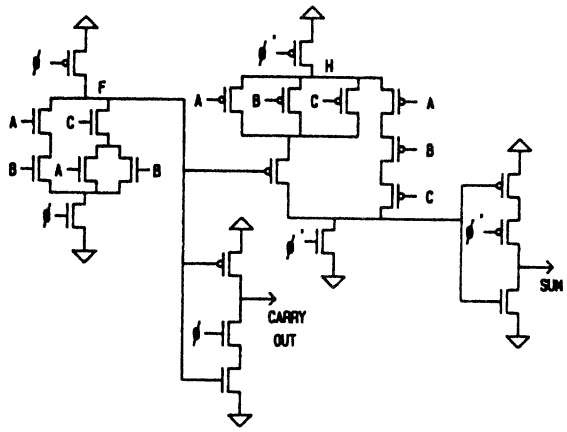


Fig. 8. Circuit for a conventional NORA full adder.

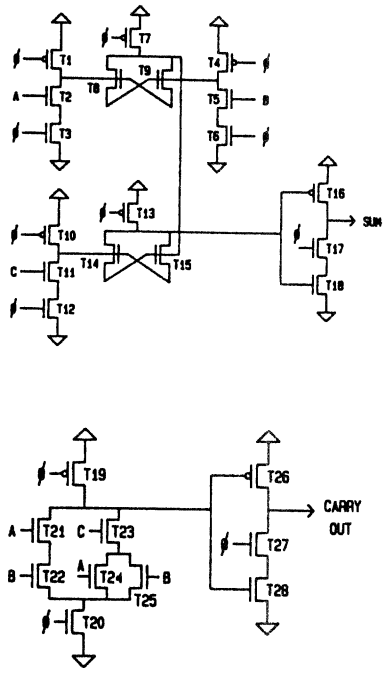


Fig. 9. Circuit for a modified NORA full adder (from [9]).

TABLE I
COMPARISON OF SIMULATION RESULTS FOR DIFFERENT TYPES OF FULL ADDERS

PROPERTY / CIRCUIT TECHNIQUE	INPUT GATE CAPACITANCE (fF)	OUTPUT LOAD CAPACITANCE (fF)	# OF P-DEVICES / # OF N-DEVICES	WORST CASE DELAY TIME (ns)	AVERAGE POWER DISSIPATION AT MAX. FREQ. (mW)	NORMALIZED POWER-DELAY PRODUCT
STATIC FULL CMOS	155	155	15/15	20	0.58	1.00
STATIC DCVS	85	85	4/18	22	1.11	2.11
STATIC DSL	85	85	4/22	14	1.35	1.63
NORA	110	220	12/10	18	0.83	1.29
MODIFIED NORA	45	90	8/20	10	1.24	1.06
DCVS NORA	85	170	12/28	10	1.55	1.34
DCVS DOMINO	85	170	12/24	9	1.75	1.36

case) n transistor. For example, a path with four serially connected transistors requires each transistor contained in that path to be 12 μm ($=4\times3$ μm) wide. Consider the right half of the network in Fig. 2(b); that the number of transistors (or stack level) contained in path $A'BC'$ is three implies that each transistor in this path should be 9 μm wide. If the width of transistor C' is 9 μm, then the width of B' in path $B'C'$ can be estimated as 4.5 μm. Similar principles can be applied to the sizing of transistors in the charging or discharging paths in other circuits. The final form of a design was arrived at by making adjustments, usually small, to the widths of the transistors on the basis of minimizing the circuit delay time as predicted by SPICE simulations.

The worst-case delay times quoted in Table I refer to situations where the input signals are such that the circuit operation is likely to be slowest. For example, in the conventional NORA circuit of Fig. 8, the speed performance will be poorest when $A=B=$ HI and $C=$ LO. In this case, during the evaluation phase ($\phi=1$), node F needs to be pulled down, in order to turn on the p-channel transistor through which node H, via transistor C, is connected to the output stage to render the sum signal LO.

Power dissipation was computed using the procedure described by Kang [11]. The figures quoted in Table I refer to average power dissipation at the maximum frequency of operation of each circuit, i.e., as determined by the worst-case delay times. The power-delay product, normalized to the static full CMOS case, is also shown in Table I.

The output load capacitances used in the simulations are meant to represent typical load conditions. A fan-out of two was used for the dynamic designs as these circuits are buffered and would be expected to be able to drive larger loads than the static gates.

V. DISCUSSION

Considering, first, the static designs, it appears that the DSL technique yields a significantly faster circuit than do the other two techniques. This is to be expected due to the need for logic swings which are only one-half of the rail-to-rail value. The significant differences between the other two static designs are the increased power dissipation and the reduced device count and input gate capacitance of the static DCVS circuit. The number of devices is less because the DCVS implementation uses only p-channel transistors, as opposed to both p- and n-channel devices, as pull-ups in the load and buffer circuitry. The input gate capacitance loading in the DCVS circuit is typically a factor of 2 or 3 times smaller than conventional CMOS circuits which require complementary n- and p-channel devices to be driven, since the inputs drive only n-channel tree devices.

The static DCVS circuit consumes more power than the conventional static CMOS circuit because the charging and discharging times of nodes Q and Q' in Fig. 3 depend on the turn-on and turn-off paths within the DCVS tree and these are, generally, not symmetrical. An asymmetry in the rise and fall times of the potential at nodes Q and Q' will prolong the period of current flow through the latch during the transient state, thus increasing the power dissipation.

The apparent attractiveness of the static DSL circuit in regards to speed is negated somewhat by three possible problems which may arise when using this technique. For example, with reference to Fig. 4, if node Q' is at 2.5 V, then $T2$ is partially ON and it is possible to destroy the low logic level that would otherwise have appeared on node f. Although reducing the size of the p device alleviates this problem, it decreases the output drive capability and results in longer delay. Thus a trade-off should be considered when the sizes of $T1$ and $T2$ are chosen. Another problem is due to the body effect existing in $T3$ and $T4$. Although the threshold voltage V_{th} is equal to 0.8 V in the Northern Telecom 3-μm CMOS process [10], SPICE simulations show that it is necessary to set V_{REF} equal to 4.2 V in order to clamp either of the nodes Q or Q' to 2.5 V. Also the clamped logic swing is sensitive to the stack level of the DCVS tree for a fixed V_{REF}. The third problem is that this circuit exhibits static power dissipation. There is a direct current path to ground through transistors $T1$ and $T3$ when Q' is low or through $T2$ and $T4$ when Q is low.

Turning now to the dynamic circuits, all the designs have a similar power-delay product. The DCVS circuits appear to have a speed advantage, but this is achieved at the expense of an increased device count. The conventional NORA circuit, Fig. 8, is characterized by a large input gate capacitance due to the wide transistors in the p-logic block, and a slow speed due to the use of two levels of gate delay and because half of the logic is performed by p transistors. Considerable improvement in these two areas is achieved by the modified NORA adder of Fig. 9. This circuit has two times smaller input gate capacitance and is nearly twice as fast as the serial NORA adder. The disadvantage of this circuit is that accidental discharge due to races is possible under certain conditions. For example, if $A=0$, $B=1$, and $C=1$, the gate of $T14$ (or source of $T15$) and the gate of $T15$ (or source of $T14$) are pulled down. If the drain nodes of $T7$ and $T10$ do not pull down at similar

rates so that a voltage difference of more than one threshold is developed across the gate nodes of $T14$ and $T15$, the drain node of $T13$ discharges accidentally. To avoid this requires careful sizing of the transistors along the discharging paths so that the conductance to ground and the capacitive load associated with each of the pull-down paths is equal. Tight process control and detailed simulation through circuit extraction are needed if this circuit is to be successfully implemented.

The DCVS NORA (Fig. 6) adder has smaller input gate capacitance and delay time than the conventional NORA adder, although the area consumed is larger. The large area stems from the symmetrical buffer circuits used to provide complementary outputs. The DCVS NORA circuit is as fast as the modified NORA adder, but only at the expense of a higher device count and increased input gate capacitance. However, the DCVS version of NORA is superior to the modified conventional version, in terms of circuit flexibility, due to its provision of complementary outputs, and reliability, due to the fact that accidental discharge cannot occur. The DCVS DOMINO (Fig. 5) adder is similar to the DCVS NORA adder in all the parameters evaluated in this comparison. It is the only kind of full-adder circuit which can be included in a DOMINO chain without causing race problems.

VI. Conclusions

The main conclusion to be drawn from this work is that DCVS logic offers opportunities for realizing faster circuits than are possible with conventional forms of CMOS logic, but this speed advantage is often gained at the expense of circuit area and active power consumption.

The fastest static logic technique investigated was the differential split-level (DSL) version of DCVS logic. The worst-case delay time for this implementation was 14 ns, while that of a conventional CMOS circuit was 20 ns. However, DSL may have some problems in terms of static power dissipation, security of charge storage, and sensitivity of the logic swing to the number of input signals.

In dynamic operation, DCVS versions of NORA and DOMINO circuits appear to be a few nanoseconds faster (9–10 versus 10–18) than their conventional counterparts. Further, DCVS logic may overcome the problem of accidental discharge, which appears to be a concern with one of the conventional NORA techniques evaluated in this study.

References

[1] L. G. Heller, W. R. Griffin, J. W. Davis, and N. G. Thoma, "Cascode voltage switch logic: A differential CMOS logic family," in *ISSCC Dig. Tech. Papers*, 1984, pp. 16–17.

[2] R. K. Montoye, "Testing scheme for differential cascode voltage switch circuits," *IBM Tech. Disc. Bull.*, vol. 27, pp. 6148–6152, 1985.

[3] K. M. Chu and D. L. Pulfrey, "Design procedures for differential cascode voltage switch circuits," *IEEE J. Solid-State Circuits*, vol. SC-21, pp. 1082–1087, Dec. 1986.

[4] L. C. Pfennings, W. G. J. Mol, J. J. J. Bastiaens, and J. M. F. Van Dijk, "Differential split-level CMOS logic for subnanosecond speeds," in *ISSCC Dig. Tech. Papers*, 1985, pp. 212–213; also *IEEE J. Solid-State Circuits*, vol. SC-20, pp. 1050–1055, Oct. 1985.

[5] R. H. Krambeck, C. M. Lee, and H. Law, "High-speed compact circuits with CMOS," *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 614–619, June 1982.

[6] L. G. Heller, "Stabilizing cascode voltage switch logic," *IBM Tech. Disc. Bull.*, vol. 27, p. 6015, 1985.

[7] N. F. Goncalves and H. J. De Man, "NORA: A racefree dynamic CMOS technique for pipelined logic structure," *IEEE J. Solid-State Circuits*, vol. SC-18, pp. 261–266, June 1983.

[8] K. M. Chu, "Cascode voltage switch logic circuits," M.A.Sc. thesis, Univ. of British Columbia, Vancouver, Canada, 1986.

[9] A. H. C. Park, "CMOS LSI design of a high-throughput digital filter," M.Sc. thesis, Mass. Inst. of Technol., Cambridge, ch. 4, 1984.

[10] G. Puukila, "Canadian Microelectronics Corporation guide for designers using the Northern Telecom CMOS3 Process," Canadian Microelectronics Corp., Kingston, Ont., Rep. IC 85-6, 1985.

[11] S. M. Kang, "Accurate simulation of power dissipation in VLSI circuits," *IEEE J. Solid-State Circuits*, vol. SC-21, pp. 889–891, Oct. 1986.

Pass-transistor networks optimize n-MOS logic

Formal methods for transfer-gate logic design achieve minimum area, delay, and power for complex circuits

by Sterling Whitaker, *American Microsystems Inc., Santa Clara, Calif.*

□ Conventional methods of designing logic chips employ blocks of discrete and small-scale integrated circuits—an approach that is now crumbling under the impact of very large-scale integration. Designers who set out to create high-performance cost-effective VLSI chips must minimize the power, delay, and area of MOS ICs. But traditional logic design, with its black-box representation of Boolean functions, does not shrink them.

However, these three parameters can be minimized by experimenting with the many combinatorial logic circuits, transfer gates, and MOS pass transistors that affect an IC's power consumption, speed, and size. Systematically designed pass-transistor networks can reduce complex functions to highly regular structures that operate more quickly than conventional n-channel MOS logic, fill only one third as much space, and consume only one eighth as much power.

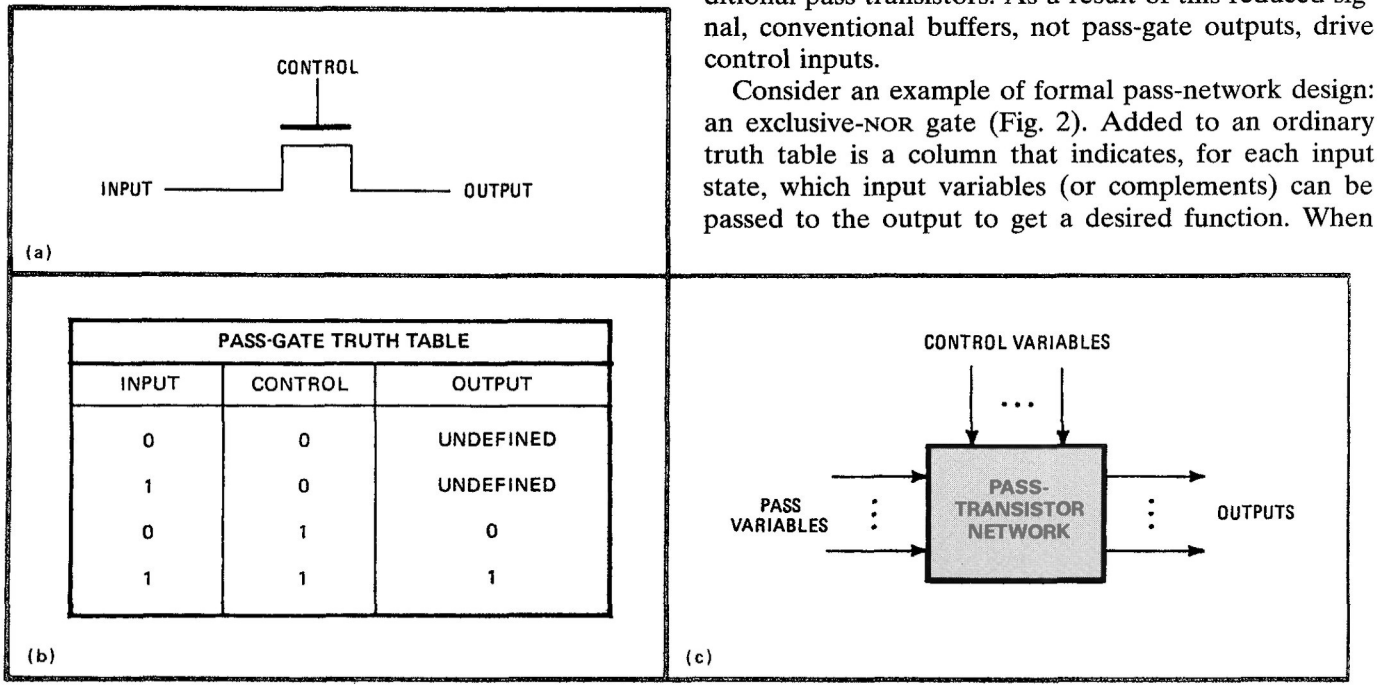
Designing pass-transistor networks has been more a craft than a science. Early masters—Carver Mead and Lynn Conway among them—popularized pass transistors in latches, flip-flops, multiplexers, and some combinatorial logic structures [*Electronics*, Oct. 20, 1981, p. 102]. The more formal design techniques presented here extend the benefits of pass transistors to networks too complex to be realized intuitively, and they also permit informally designed pass networks to be verified in a systematic way.

The basics

Figure 1 shows the pass gate and its logical function. When control inputs are high the transistor conducts, passing the logic level at its input to its output. When the control input is low the transistor is off, leaving the output in a high-impedance (or undefined) state. Designers form pass-gate networks (see Fig. 1c) by joining together the outputs of pass transistors to feed the inputs of succeeding transistors.

A high signal level at a pass transistor's input is reduced by a threshold voltage at the output. This reduced level is passed, without further degradation, through additional pass transistors. As a result of this reduced signal, conventional buffers, not pass-gate outputs, drive control inputs.

Consider an example of formal pass-network design: an exclusive-NOR gate (Fig. 2). Added to an ordinary truth table is a column that indicates, for each input state, which input variables (or complements) can be passed to the output to get a desired function. When

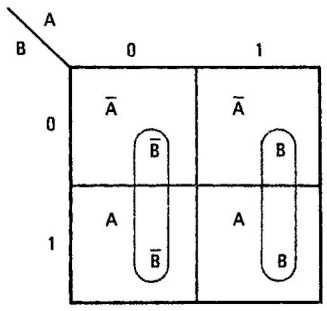


1. Pass gate. When an MOS transistor is operated as a pass gate, as shown in (a), the device passes the signal at its drain to its source. As the truth table given in (b) demonstrates, the output is in a high-impedance, or undefined, state when the transistor is shut off. Networks of pass gates make pass and control variables flow at right angles (c).

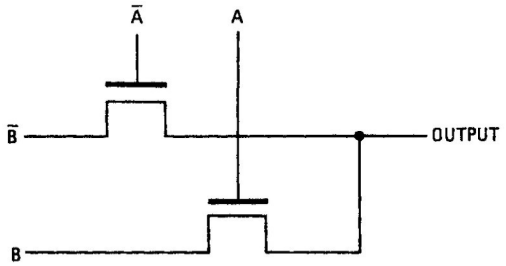
Reprinted with permission from *Electronics* - S. Whitaker, "Pass-transistor networks optimize n-MOS logic," Vol. 56, No. 19, pp. 144-148, September 1983.

EXCLUSIVE NOR TRUTH TABLE			
A	B	OUTPUT	PASS FUNCTION
0	0	1	$\overline{A} + \overline{B}$
0	1	0	$A + \overline{B}$
1	0	0	$\overline{A} + B$
1	1	1	$A + B$

(a)



(b)



(c)

2. Straightforward logic. With the aid of a truth table (a) that includes pass functions, it is possible to modify conventional logic design for pass transistors. The pass functions are entered in a Karnaugh map (b) with pass variables looped together. The resulting exclusive-NOR gate needs just two devices (c).

both A and B equal 0 the output should be 1. Either $\overline{A}$ or $\overline{B}$ can be passed to the output. $\overline{A} + \overline{B}$ is called the pass function.

For each state of the input variables, pass functions are then entered into corresponding cells of a conventional Karnaugh map (Fig. 2b). To steer the pass variables to the output, they are grouped to produce the control functions that drive the pass-gate control inputs.The left loop in Fig. 2b groups $\overline{B}$ under the control of $\overline{A}$; the other one groups B under the control of A.

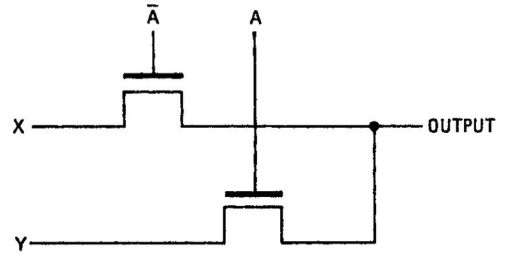
Figure 2c shows the resulting pass-transistor network, which has one pass-transistor delay and comprises two devices. The network's steady-state power dissipation—consisting only of leakage currents between the substrate and the transistors' source and drain regions—is negligible. Figure 3 shows the other basic logic functions—AND, NAND, OR, NOR, and exclusive-OR—that can be derived in the manner of exclusive-NOR.

Contrast this simplicity with the traditional n-MOS version, which comprises five transistors, is plagued by pull-down or pull-up delays, and has one node that dissipates steady-state power. Complementary-MOS gates do not dissipate steady-state power, but they do comprise eight transistors and have a pull-up or -down delay.

Synthesizing pass networks

Logic functions that are more complex can be approached as the simple gates are: with a truth table that includes the pass functions and with a modified Karnaugh map to derive control functions. The control-function groupings in the Karnaugh map for a pass network can be reduced with techniques that resemble minterm reduction in classical logic design. But the reduction rules differ in three ways from those of conventional Karnaugh maps.

For one thing, a pass transistor's output is not defined when its control gate is not asserted, so a variable must be passed in every map state for which the output has been defined. (In a conventional map, only those states with true outputs must be looped.) Then, too, more than one variable may be passed in one state because the pass

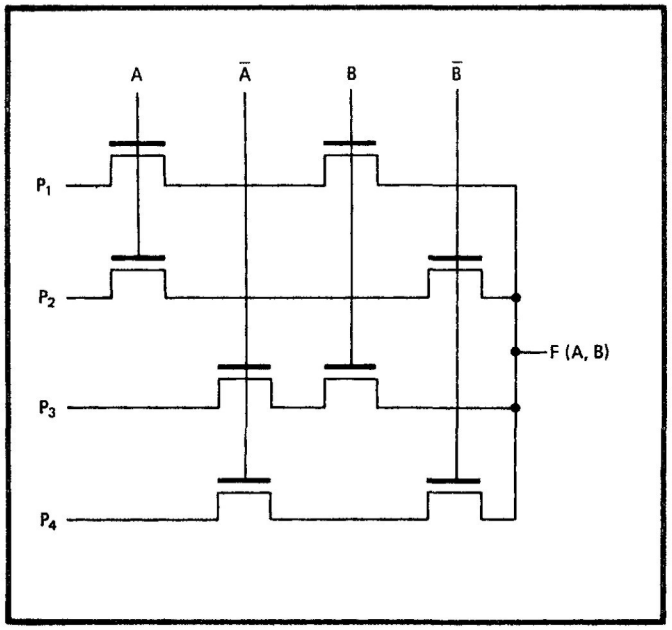


(a)

OUTPUT	X	Y
AB	0	B
$\overline{AB}$	1	$\overline{B}$
$A + B$	B	1
$\overline{A + B}$	$\overline{B}$	0
$A \oplus B$	B	$\overline{B}$

(b)

3. Fundamentals. Like the exclusive-NOR gate of Fig. 2, the five other basic logic functions can be implemented with the assistance of just two transistors. The X and Y inputs in (a) take the values shown in the table (b) to produce the logic functions.



4. Function generator. Each state of the input word AB connects just one of the four pass lines, P_1 through P_4, to the output. Any function of A and B can therefore be implemented with the proper choice of logic values for the pass lines.

functions in the map ensure that the passed variables are all at the same logic level. Finally, once a "don't care" state has been included in a loop, it acquires a pass function determined by the variable being looped.

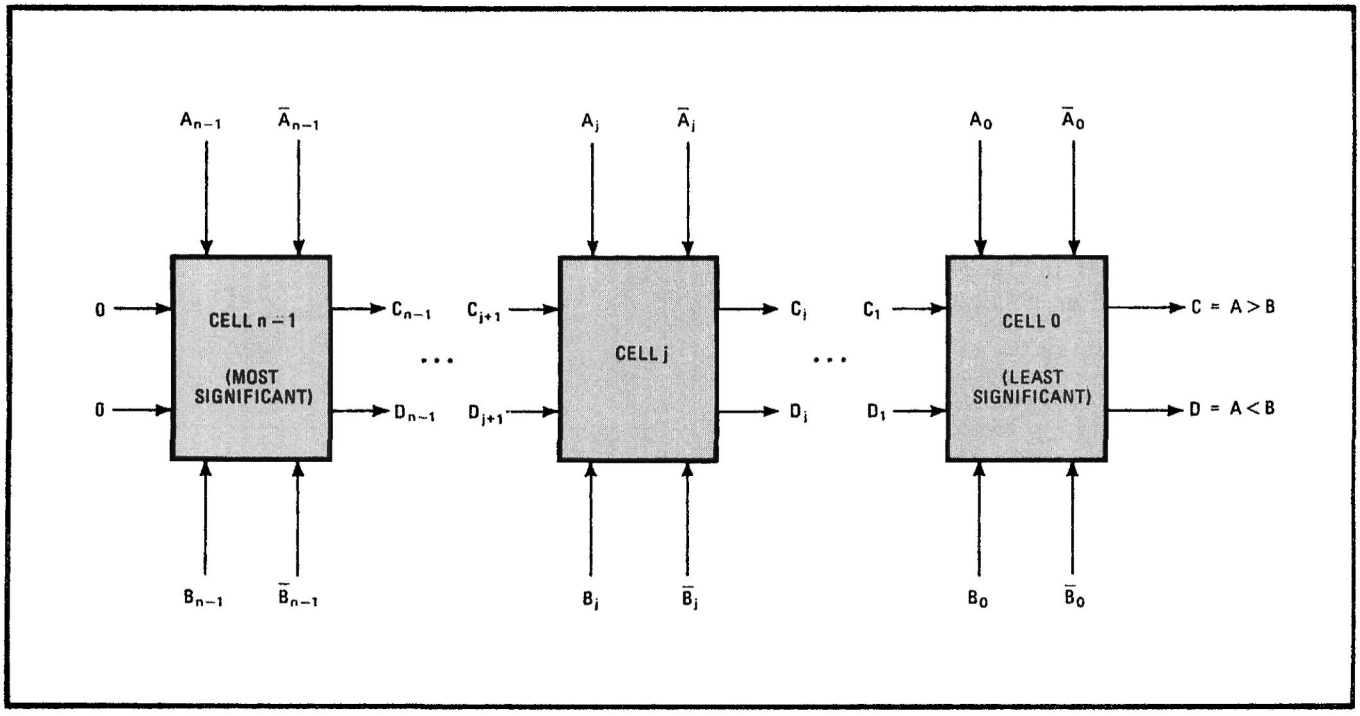
The input variables must be divided between two sets—pass variables and control variables—for modified Karnaugh maps to be used in pass-network design. Maps do not always make it clear whether or not such a division is worth making, especially with more than a few variables. One thing can help: understanding the connection between pass networks and the canonical sum-of-products form of Boolean equations.

All possible functions of variables can be synthesized in a single pass-transistor network. Mead and Conway discussed such a function generator for an arithmetic and logic unit. Consider Fig. 4, for example. In each of four possible states of control variables A and B, one of pass lines P_1 through P_4 is connected to the output. To implement the function "A exclusive-OR B," for example, input 0110 is applied to the pass lines.

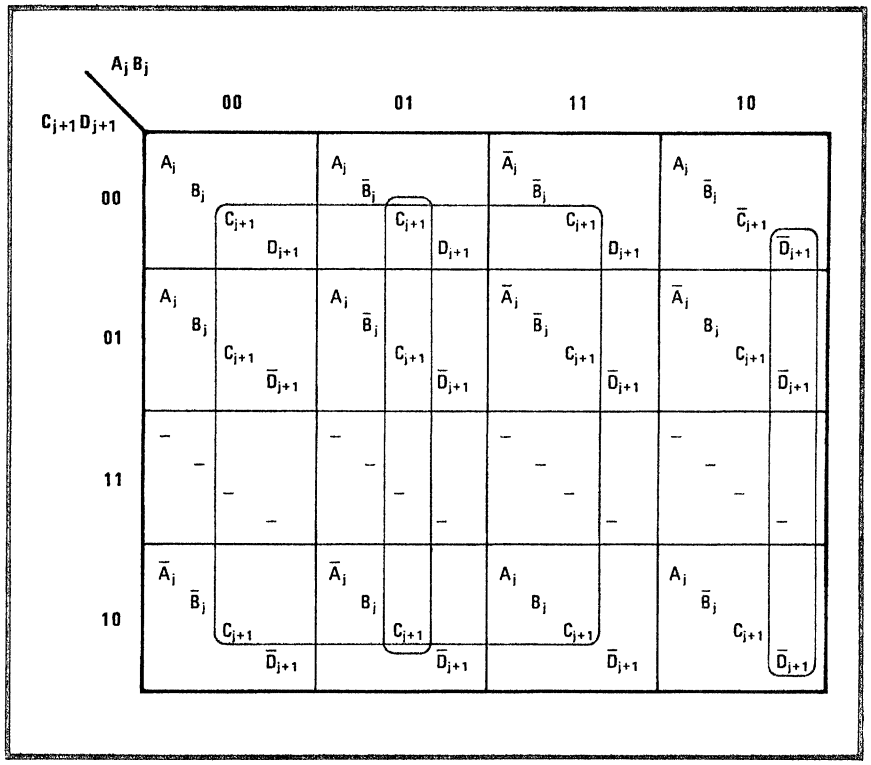
Input determines function

This circuit is useful because different functions can be implemented just by placing the proper input on the pass lines. In fact, the scheme can be extended to implement all the functions of N variables, which require 2^N pass lines and $N2^N$ transistors. Many random-logic designs do not need this flexibility. But they do need to implement the function with the least possible area, power, delay—and effort.

In the network shown in Fig. 4, properly assigning a third variable—C—or its complement, or the constants 1 or 0 to the pass lines achieves any particular function of three variables. (This claim can readily be verified with the canonical sum-of-products form for Boolean logic functions.) Of the eight minterms that can be formed from three variables, four at most enter any particular function. A fifth minterm can always be reduced in combination with one of the others. Each pass line of the network implements a three-term product. (The first is ABP_1.) Since every possible state of inputs



5. Comparator in operation. Bit by bit, an iterative array compares two digital words A and B. The comparison of the most significant bits (left) provides an intermediate result that passes to the right and is used in comparing the next bits. Each cell performs the same operations, and the final result is available from the least significant cell.



6. Map. For cell j of the comparator shown in Fig. 5, output C_j is mapped as a function of the corresponding bits of A and B and of outputs C_{j+1} and D_{j+1} from the previous cell. The loops (color) show that $\overline{A}_j+B_j$ passes C_{j+1}, while $A_j\overline{B}_j$ passes $\overline{D}_{j+1}$.

A and B leads to the selection of some pass line, the output is always defined.

Moreover, a particular function of N variables can always be implemented with $2(N-1)$ control lines, 2^{N-1} pass lines, and $N2^{N-1}$ transistors. This, however, is a worst case. Fewer than the maximum number of minterms may be present, and some can be reduced in combination.

When written in what might be called the "pass canonical form," Boolean equations can generally be translated straight into pass-transistor networks. Pass networks, as their structures show, directly implement a Boolean equation of the form:

$$F = P_1F_1(C_1, \ldots, C_m) + \ldots + P_nF_n(C_1, \ldots, C_m)$$

where the F_i are the control functions formed with series and parallel combinations of pass transistors, the C_i are the control variables that drive the gates of those transistors, and the P_i are the pass variables.

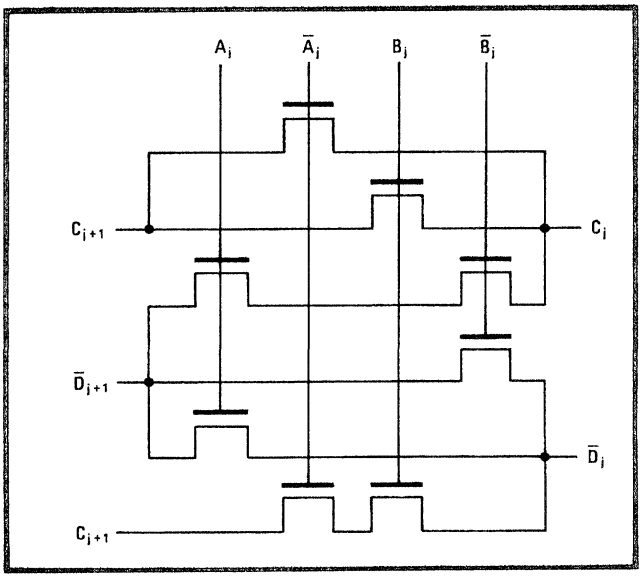
The F_i are a complete and disjoint set: their logical sum is 1, and the logical product of any two is 0. These conditions guarantee that every state of the control variables selects one and only one pass variable. They are easy to meet, for on a Karnaugh map of the C_i the F_i are nonintersecting loops that cover the whole map. Of course, when a high-impedance output state is desired, the corresponding control function can deliberately be left out of the network.

As the above equation shows, no pass variable enters any control function, so an equation does not of necessity allow more than one pass variable. Several pass variables, however, usually produce a denser and faster network. A simple algorithm can be used to calculate the maximum number of pass variables for a Boolean equation written in reduced minterm form.

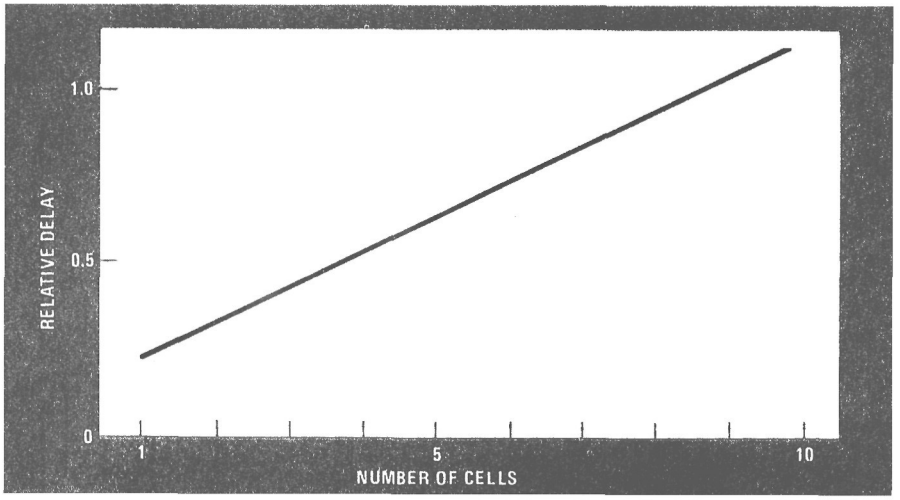
Simple logic

For simple logic functions, pass networks do significantly improve on the traditional implementation, but performance requirements limit their size and usefulness. A signal passing to a network's output travels through several device channels, each with on-resistance and capacitances to gate and substrate. Delay through the network therefore increases with the square of the number of pass transistors. Conventional logic gates merely add delays in turn. The resistance-capacitor time constant associated with the channel of a single pass transistor is typically about 0.2 nanosecond. The delay of a conventional logic stage is about 2 ns.

Conventional NAND or NOR gates can implement any function in three levels of logic. Pass networks, however, may require conventional buffer stages at the output to restore logic levels and drive further stages. For a fair comparison, one logic-gate delay should therefore be added to the pass-network delay.



7. Iteration. One cell of the comparator takes just eight pass transistors. The layout occupies an area of 3,024 square micrometers, only a third of the area of a conventional n-channel MOS implementation, and the cell dissipates almost no standby power.



8. Delay comparison. The propagation delay of a pass-transistor implementation of the magnitude comparator increases with the number of stages. For more than eight stages, it turns out that conventional buffers are required with pass gates.

A signal can pass through five pass transistors and a conventional buffer stage in a time about equal to the delay involved in passing through three conventional logic stages. Those five transistors correspond to five control variables, which can pass at least one and possibly several pass variables. Functions of six or fewer variables are thus generated more quickly in pass networks than in conventional logic; more complex functions are not always suited to them.

Like conventional NAND logic, programmable logic arrays incur only three logic-stage delays, however many variables may team up in the function that is implemented. Unlike pass networks, which basically are wired-OR functions, both PLAs and conventional NAND logic independently form each minterm of a function, permitting a single minterm to be used in more than one output function. Several are often derived from one set of input variables, so cutting the number of transistors in PLAs and conventional NAND logic often offsets the additional area both need for ground connections and load devices and the larger number of transistors per function.

Pass-transistor networks are not the best solution for all problems in combinatorial logic; their unique features suit particular applications—fairly simple random-logic functions, for instance. They also implement a disabled output's high-impedance state naturally and thus make great sense for bus drivers and multiplexers.

Iterative arrays

Pass-transistor networks are usually the best choice for another important class of logic functions: iterative arrays. Iterative logic circuits, often used in magnitude comparators and adders, form a series of intermediate results along the way to the final output. Serial processing is needed in all logic implementations, so pass-transistor designs can always have lower delays than conventional ones.

Consider the iterative magnitude comparator in Fig. 5. A 1-bit cell in it compares the most significant bit of A and the most significant bit of B. Intermediate results C and D are then passed to the next-most significant cell, where they are used with the next bits of A and B to complete the comparison. The process continues until the final result is available from the least significant cell.

For cell j, the output C_j is 1 if A_j is greater than B_j and D_{j+1} is 0, or if C_{j+1} is 1. Output D_j is 1 if D_{j+1} is 1, or if A_j is less than B_j and C_{j+1} is 0. These conditions make the final output, C, high if A is greater than B. D is high if B is greater than A. If both C and D are low, A and B are equal.

Truth tables are constructed for C_j and D_j (with the methods described earlier) and the pass functions are entered in Karnaugh maps. The map for C_j (Fig. 6) shows how the pass variable $\overline{C}_{j+1}$ is looped under the control function $\overline{A}_j+B_{jj}$, while $\overline{D}_{j+1}$ is looped under the function $A_j\overline{B}_j$. The map for D_j is completed in a similar manner.

The resulting pass-network implementation of one cell of the magnitude comparator (Fig. 7) occupies 3,024 square micrometers—only 34% of the 8,840 mm^2 conventional logic designs need. Simulations of the two versions using parameters extracted from the layout are employed to compare circuit delays.

As mentioned previously, pass-transistor networks form RC delay lines, and signals on them obey a diffusion equation. The delay takes the form $An^2 + B$, where n is the number of cells in the comparator and A and B are constants. Delays through the conventional logic network take the form Cn. Constant C is larger than A, in part because pass networks cut capacitance by cutting the number of transistors needed to implement a function. This reduction increases the circuit's regularity, cuts the amount of wiring and gate capacitance, and completely eliminates depletion-load devices and their gate capacitance.

Figure 8 compares the delay of a pass network with that of a conventional circuit by plotting them as functions of the number of bits in the comparator. For 8 or fewer bits, pass networks are faster than conventional ones. For more bits, they can still be faster if conventional buffer stages are inserted when the delay from an additional pass-gate stage would exceed the delay through a buffer followed by the pass stage.

With 8 or fewer bits, the pass-transistor network's power consumption is negligible. For more than 8 bits, the conventional buffers added to the circuit contain two power-dissipating nodes. The conventional logic design has two power-consuming nodes in each cell, so its total power consumption is eight times that of the pass network. □

A 3.8-ns CMOS 16×16-b Multiplier Using Complementary Pass-Transistor Logic

KAZUO YANO, MEMBER, IEEE, TOSHIAKI YAMANAKA, MEMBER, IEEE, TAKASHI NISHIDA, MASAYOSHI SAITO, MEMBER, IEEE, KATSUHIRO SHIMOHIGASHI, MEMBER, IEEE, AND AKIHIRO SHIMIZU

Abstract—**A 3.8-ns 257-mW CMOS 16×16-b multiplier with a supply voltage of 4 V is described. A complementary pass-transistor logic (CPL) is proposed and applied to almost the entire critical path. The CPL consists of complementary inputs/outputs, an nMOS pass-transistor logic network, and CMOS output inverters. The CPL is twice as fast as conventional CMOS due to lower input capacitance and higher logic functionality. Its multiplication time is the fastest ever reported, including bipolar and GaAs IC's, and it can be enhanced further to 2.6 ns with 60 mW at 77 K.**

I. Introduction

THE SPEED of CMOS devices, which were used mainly in low-power high-density LSI's, has increased drastically with the rapid progress in miniaturization. CMOS speed is getting close to that of Si bipolar technology; for example, with submicrometer CMOS technology, a 9-ns 1-Mb SRAM [1] and a 7.4-ns 16×16-b multiplier [2] have been reported. However, the recent progress in fast engineering workstations and real-time digital-signal processing requires faster CMOS speed.

A multiplier is an essential element in any digital-signal processing circuit and constitutes the critical path in DSP and FPU LSI's. Recently, versatile microprocessor chips also have begun to contain multipliers, which is made possible by the rapid progress in integration technology. Therefore, the demand for improved multiplier performance is increasing. In addition, multipliers are designed and fabricated as benchmarks for demonstrating various high-speed technologies, e.g., Si bipolar [3], GaAs [4], and Josephson junction devices.

This paper describes a fast 3.8-ns 0.5-μm CMOS 16×16-b multiplier that is implemented as a test vehicle for investigating a new circuit technique for high-speed CMOS-based logic circuits. A new family of advanced differential CMOS logic, called complementary pass-transistor logic (CPL), is proposed and fully utilized on almost the entire critical path to achieve very high speeds. The circuit techniques and the technology used in this multiplier have the potential for 100-MHz operation of 32- to 64-b floating-point multiplication, thus enabling very fast DSP's, FPU's and ASIC's. First, the concept of CPL is introduced in Section II, and then CPL implementation of the multiplier is described in Section III. Device fabrication is described in Section IV, and finally the performance of the fabricated multiplier is shown in Section V.

Manuscript received September 1, 1989; revised December 4, 1989.

K. Yano, T. Yamanaka, T. Nishida, M. Saito, and K. Shimohigashi are with the Central Research Laboratory, Hitachi Ltd., Kokubunji, Tokyo 185, Japan.

A. Shimizu is with the Hitachi VLSI Engineering Corporation, Tokyo, Japan.

IEEE Log Number 8934072.

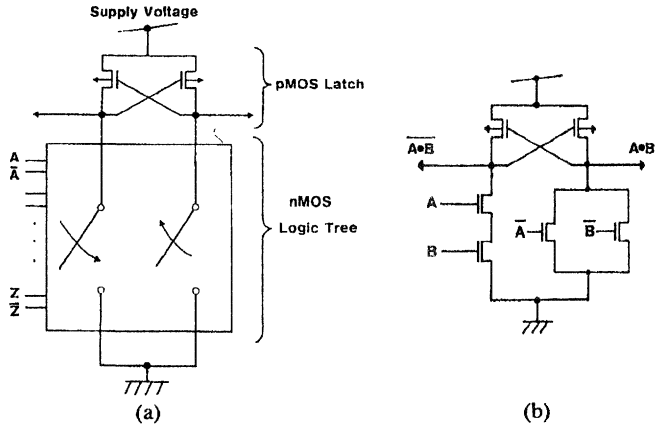


Fig. 1. Basic CVSL circuit. (a) Schematic structure. (b) AND/NAND circuit.

II. CPL: Concept and Examples

Several differential CMOS logic families such as cascode voltage switch logic (CVSL) [5] (Fig. 1) and differential split-level logic (DSL) [6] have been proposed for CMOS circuit speed improvement. These have the common features of complementary data inputs/outputs, an nMOS logic tree, and a pMOS cross-coupled load, which together can reduce input capacitance, increase logic functionality, and sometimes eliminate inverter circuits. Therefore, these logic families can increase speed. However, the actual advantage of CVSL circuits is less than that anticipated in the original paper, as clarified in [7]. This is because the pMOS cross-coupled latch cannot easily be inverted due to the regenerative property of the latch. High-speed inversion of the pMOS latch is possible only when the gate width of the pMOS is sufficiently small. However, a small

Reprinted from *IEEE Journal of Solid-State Circuits*, Vol. 25, No. 2, pp. 388-394, April 1990.

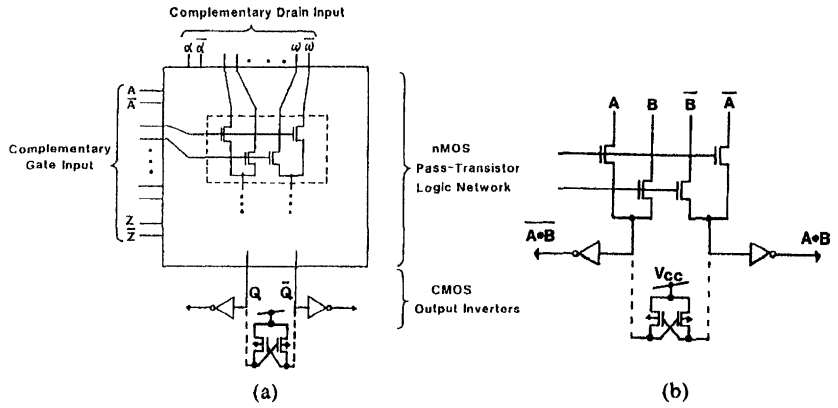


Fig. 2. Basic CPL circuit. (a) Schematic structure. (b) AND/NAND circuit.

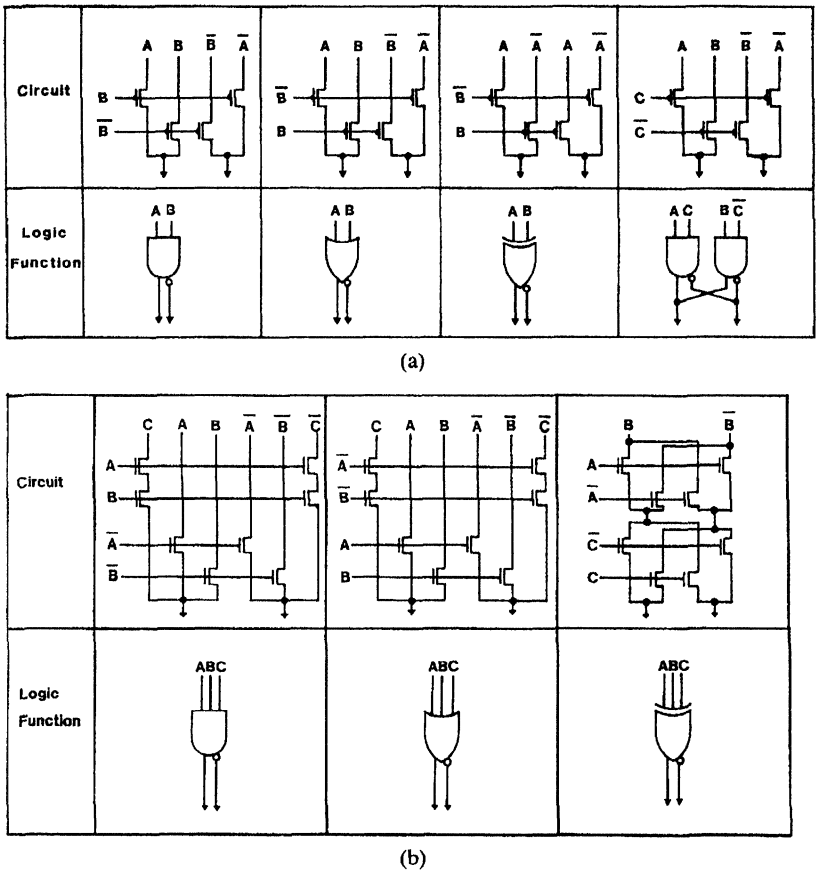


Fig. 3. CPL circuit modules. (a) Two-way logic. (b) Three-way logic.

gate width severely degrades the pull-up transit time. DSL is faster than conventional CMOS, however at the expense of static power consumption [7].

The main concept behind CPL is the use of an nMOS pass-transistor network for logic organization, and elimination of the pMOS latch, as shown in Fig. 2. CPL consists of complementary inputs/outputs, an nMOS pass transistor logic network, and CMOS output inverters. The pass transistors function as pull-down and pull-up devices. Thus the pMOS latch can be eliminated, allowing the advantage of the differential circuits to be fully utilized. Because the high level of the pass-transistor outputs (nodes Q and $\overline{Q}$) is lower than the supply voltage level by the threshold voltage of the pass transistors, the signals have to be amplified by the output inverters. At the same time, the CMOS output inverters shift the logic threshold voltage and drive the capacitive load. The logic threshold shift is necessary because the logic threshold voltage of the output inverter is lower than half the supply voltage, due to the lowering of the high signal level.

A pMOS latch can also be added to CPL, as shown in Fig. 2, to decrease static power consumption, as opposed to the conventional pull-up function. In this case, the pMOS gate width can be designed to be minimum, as long

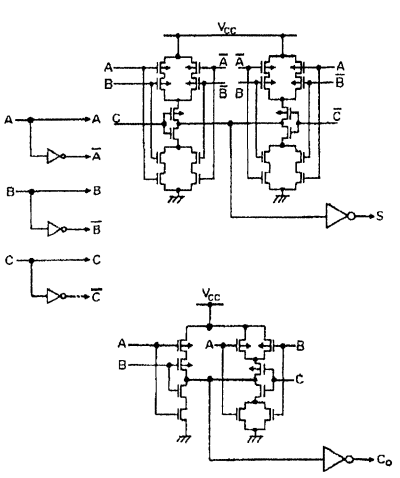
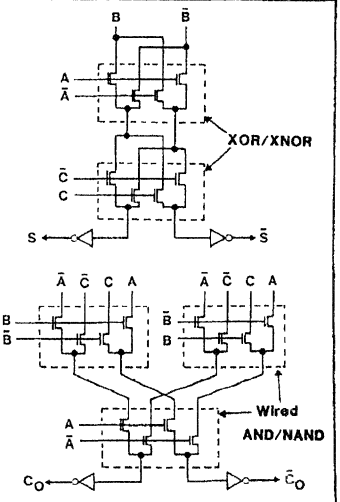
	CMOS	CPL
Full Adder Circuit		
Transistor Count	40	28
Area	4730μm^2	4218 μm^2
Delay (4V)	0.63ns	0.26ns
Power (100MHz)	1.2mW	0.86mW

Fig. 4. Comparison of CMOS full adder with CPL full-adder circuit.

as the pull-up function is completed in the given cycle time.

Arbitrary Boolean functions can be constructed from the pass-transistor network by combining four basic circuit modules: an AND/NAND module, an OR/NOR module, an XOR/XNOR module, and a wired-AND/NAND module. These are shown in Fig. 3(a) in which the XOR/XNOR module is used once [8]. One attractive feature of CPL is that the complementary outputs are produced by the simple four-transistor circuits. Because inverters are unnecessary in CPL circuits, the number of critical-path gate stages can be reduced. Note that these various functions are produced by an indentical circuit configuration with only a change of input configuration. This property of CPL is apparently suitable for masterslice design. Logic functions for three and greater inputs can also be easily constructed similarly to two-way logic. Examples of three-way input logics are shown in Fig. 3(b).

As an example of more complex logic circuits, a CPL full adder is shown in Fig. 4. A conventional CMOS full adder [9] is also shown in Fig. 4 for comparison. In the CPL full adder, both the sum logic and carry logic are structured on the CPL concept by combining the basic modules in Fig. 3. The sum logic comprises two XOR/XNOR modules, whereas the carry logic comprises three wired-AND/NAND modules. The output inverters are "overhead," in the sense that they are needed whether the circuit has one, two, or many inputs. Therefore designing with complex logic functions in a gate is adopted to minimize the overall device count and delay time.

Since pMOS can be eliminated in logic construction in the CPL, the input capacitance is about half that of the conventional CMOS configuration, thus achieving higher speed and lower power dissipation. Moreover, the powerful logic functionality of CPL due to the multilevel pass-transistor network realizes complex Boolean functions efficiently with a small number of MOS transistors, thus further reducing area and delay time. In fact, the transistor count in the CPL full adder is 28, whereas in the CMOS it is 40. The areas required for these full adders based on the half-micrometer design rule are 4218 μm^2 (37×114 μm^2) for CPL and 4730 μm^2 (55×86 μm^2) for conventional CMOS. The actual area reduction rate is smaller than the transistor-count reduction rate, because the interconnection area is not proportional to the transistor count. Further, the complementary input/output function eliminates the internal inverter to provide XOR input in the full adder, thus reducing the number of critical-path gate stages.

To compare the full-adder performance between CPL and CMOS, circuit simulations are performed using the half-micrometer device parameters at a supply voltage of 4 V. The 4-V supply was chosen because it is the maximum voltage at which half-micrometer CMOS devices are immune to hot-carrier degradation. The simulated worst-case delay time (which refers to situations where the input signals are such that circuit operation is slowest) of the CPL full adder is as short as 0.26 ns, which is 2.5 times faster than the conventional CMOS.

The simulated power dissipation as a function of supply voltage is shown in Fig. 5. The CPL consumes 30% less power than CMOS with a 4-V supply mainly due to smaller input capacitance. The effect of power reduction in CPL is more significant at lower supply voltages. This is because the logic swing of the pass-transistor outputs

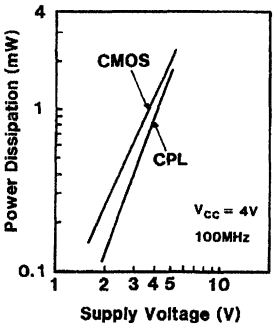


Fig. 5. Simulated full-adder power dissipation versus supply voltage.

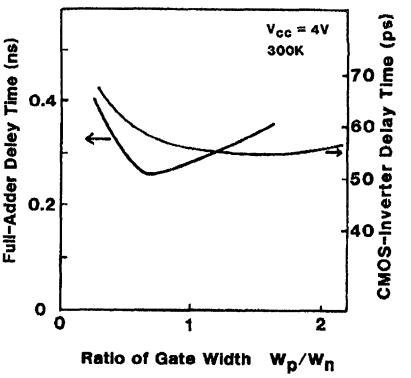


Fig. 6. Simulated full-adder delay time versus gate width ratio W_P/W_N. Delay dependence of a simple CMOS inverter on the gate width ratio is also shown.

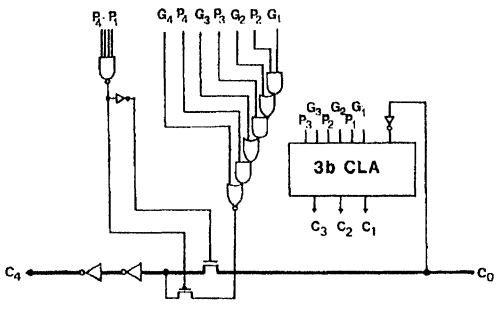
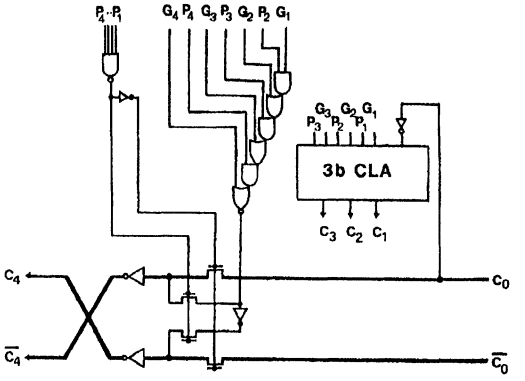
	Single-Ended Pass Transistor Logic	CPL
CLA Circuit		
Transistor Count	82	86
Delay ($C_0 \rightarrow C_4$)	0.26ns	0.15ns

Fig. 7. Comparison of 4-b carry-lookahead circuits between a CPL circuit and its single-ended counterpart. C_j is carry signal, G_j is carry-generate signal, and P_j is carry-propagate signal.

(nodes Q and $\overline{Q}$ in Fig. 2) is smaller than the supply voltage level. The dynamic power resulting from charging and discharging the capacitances in CPL circuits is given as

$$P = C_{\mathrm{OUT}} \cdot V_{CC}^2 \cdot f + C_{INT} \cdot V_{CC}(V_{CC} - V_{TN}) \cdot f \qquad (1)$$

where P is the dynamic power in the CPL circuit, C_{OUT} is the output capacitance driven by the output inverters, C_{INT} is the internal capacitance (at nodes Q and $\overline{Q}$ in Fig. 2), V_{CC} is the supply voltage, V_{TN} is the threshold voltage of the nMOS pass transistors, and f is the operating frequency. On the other hand, as is well known, the power dependence on supply voltage in CMOS circuits is described by squared dependence:

$$P' = C'_{\mathrm{OUT}} \cdot V_{CC}^2 \cdot f \qquad (2)$$

where P' is the dynamic power in the CMOS circuit, and C'_{OUT} is the total capacitance in the CMOS circuit. In the full-adder circuits in Fig. 4, the sum of C_{OUT} and C_{INT} is 20% smaller than C_{OUT}'. In addition, at low supply voltages the threshold voltage effect of the second term in (1) further reduces the dynamic power dissipation.

The following are key points in CPL circuit design:

1) controlling the threshold voltage of the nMOS pass transistors to a lower value than the threshold voltage of pMOS; and
2) designing the logic threshold voltage of the output CMOS inverter to a lower value than $V_{CC}/2$.

These points improve the speed, static power dissipation, and noise margins of CPL circuits by offsetting the lowering of the high signal level at the pass-transistor output nodes. The full-adder delay time as a function of gate width ratio between pMOS and nMOS in output inverters W_P/W_N has a minimum as shown in Fig. 6. The optimum gate width ratio exists at the ratio of about 0.75, which is much less than that of the ordinary CMOS inverters (1.5–2).

The CPL circuit can also be applied to carry-lookahead logic, which is indispensible for fast ALU's and multipliers. The CPL 4-b carry-propagate circuit and its single-ended counterpart are shown in Fig. 7. The logic functions

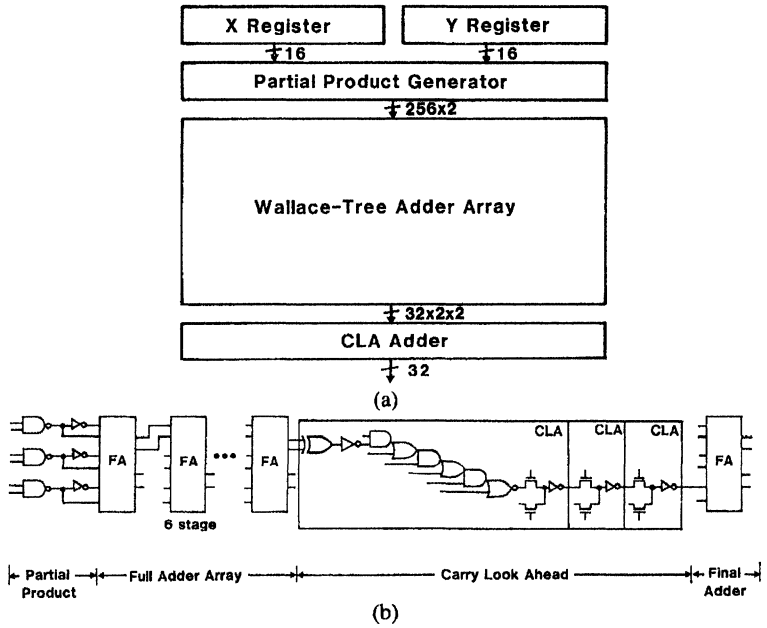


Fig. 8. (a) Block diagram of the 16×16-b multiplier. (b) Critical path of the 16×16-b multiplier.

of these carry-propagate circuits are expressed by

$$C_4 = G_4 + P_4 \cdot [G_3 + P_3 \cdot (G_2 + P_2 \cdot G_1)] + P_4 \cdot P_3 \cdot P_2 \cdot P_1 \cdot C_0 \quad (3)$$

$$C_4 \equiv A + B \cdot C_0 \quad (4)$$

where C_0 is the carry input from the lower carry-lookahead unit, C_4 is the carry output to the higher carry-lookahead unit, P_j's ($j = 1$–4) are the carry-propagate signals, and G_j's are the carry-generate signals. The OR logic in (4) can be expressed by the simple pass-transistor circuits in Fig. 7 by considering the following relationship:

$$A \cdot B = 0. \quad (5)$$

Within a single inverter delay time, the CPL carry-lookahead unit can quickly transfer the carry output to the upper unit (C_4 and $\overline{C_4}$) after receiving the carry input from the lower unit (C_0 and $\overline{C_0}$). Complementary carry data in the CPL unit can be inverted simply by twisting the carry lines. The simulated delay time is only 0.15 ns/4 b. By contrast, the single-ended counterpart requires two-stage inverter delay. Because in actual adders and multipliers these units are connected in series (for a 64-b adder, 16 units have to be connected) and constitute the critical path, the delay reduction of a factor of 2 significantly reduces the total adder delay time.

III. Multiplier Architecture

The 16×16-b multiplier was designed using a parallel multiplication architecture, as shown in Fig. 8(a). A Wallace-tree adder array and a CLA adder were used to minimize the critical-path gate stages. There are a total of 8500 transistors in an active area of 1.3×3.1 mm^2, whereas the area including bonding pads is 1.6×4.5 mm^2. The transistor count is less than that of a full CMOS counterpart [2], mainly because the transistor count in the full adder is less than that of the CMOS full adder. The critical path consists of a partial-product generator, six full adders, lookahead carry logic, three carry-propagate circuits, and a final full adder as shown in Fig. 8(b).

IV. Device Fabrication

The multiplier and the ten-stage full-adder chains are fabricated with double-level-metal 0.5-μm CMOS technology. The minimum feature size is 0.5 μm, and the gate oxide thickness is 12.5 nm. In this fabricated 0.5-μm CMOS device optimized for CPL, the nMOS pass transistors are designed to have a threshold voltage of 0 V, whereas the other nMOS and pMOS have a threshold voltage of 0.4 and −0.4 V, respectively. This threshold control reduces the static power dissipation and delay time. The drain saturation current per 10-μm width is 4.6 mA for nMOS and −2.6 mA for pMOS. The interconnection metal consists of first-level W and second level Al. The W was adopted for its high immunity to electromigration.

V. Performance Results

A microphotograph of the multiplier and the full-adder chains is shown in Fig. 9. Ten-stage full-adder chains using CPL full adders and CMOS full adders were designed and fabricated to compare the actual performance. The measured worst delay time as a function of supply voltage is shown in Fig. 10.

The delay dependences are similar for CMOS and CPL. Thus CPL can be used at supply voltages at least as low as those for CMOS. The results agree well with the simula-

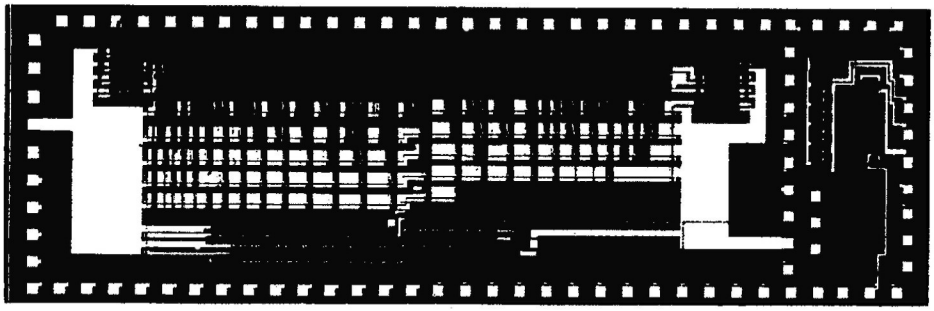


Fig. 9. Microphotograph of the 16×16-b multiplier chip (left) and ten-stage full-adder chains (right).

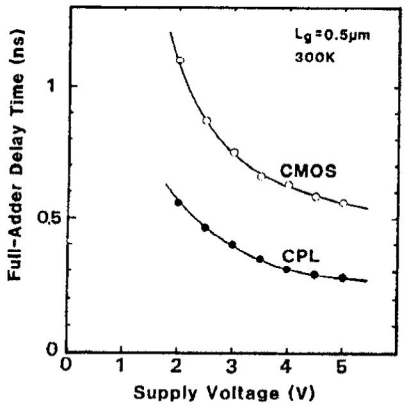


Fig. 10. Measured full-adder delay time versus supply voltage.

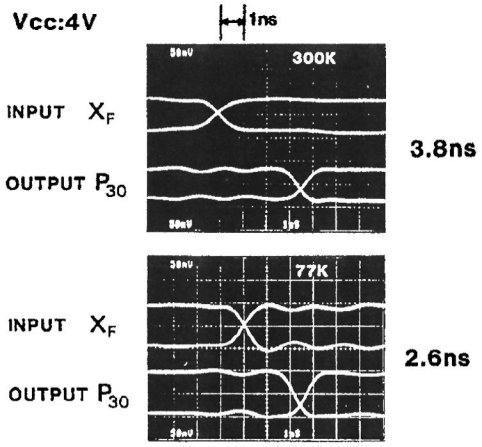


Fig. 11. Measured waveforms of the multiplier.

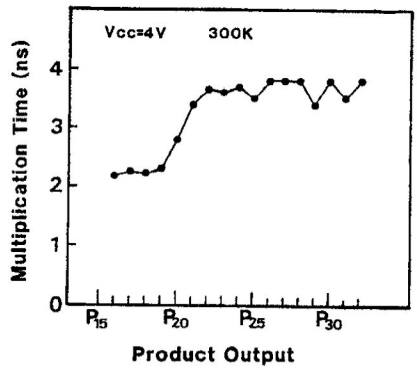


Fig. 12. Measured multiplication time versus product output.

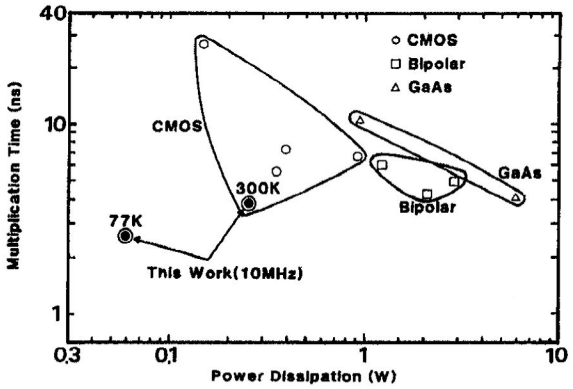


Fig. 13. Comparison of delay and power dissipation of high-speed 16×16-b multipliers.

tion results, experimentally verifying the advantage of CPL circuits.

The completed multiplier chips were probe tested at wafer level. The fully functional chips were mounted on the 68-pin pin-grid-array ceramic package, followed by waveform observation of the clock inputs and the product outputs through source-follower circuits on the chip. The test was performed up to the maximum frequency limit (250 MHz) of the pulse generator. Speed performance was measured using the worst-case pattern, FFFF×8001−7FFF×8001. Circuit simulation was also performed to confirm that this pattern consists of a series of worst-case operations of the full adder and the CLA. The multiplication time was measured at both room and liquid-nitrogen temperatures. The latter was considered for high-end applications, such as supercomputers [10], [11]. The maximum multiplication times were 3.8 and 2.6 ns at room and liquid-nitrogen temperatures, respectively. The measured waveforms are shown in Fig. 11. The observed signal amplitude is larger at lower temperatures because the on-chip source-follower gain is larger. The multiplication time versus product output is shown in Fig. 12. P_{30} gives the longest delay time as expected from our simulations. There is very little variation on the product output from P_{22} to P_{32}, which means the carry propagates very quickly in the CPL carry-lookahead circuit. Power dissipation was 257 and 60 mW at 300 and 77 K, respectively, for 10-MHz operation with a pattern of FFFF×FFFF−0000×FFFF.

The multiplication times at both room and liquid-nitrogen temperatures are compared with the published 16×16-b multipliers in Fig. 13. The minimum multiplication time before this work was 4.1 ns with a power dissipation of 6.2 W and was realized by GaAs high electron mobility

TABLE I
FEATURES OF THE 16×16-b MULTIPLIER

Architecture	Wallace Tree + CLA
Technology	0.5-μm CMOS
Gate Length	0.5 μm
Gate Oxide Thickness	12.5 nm
Metal Line/Space W	0.8/0.8
Al	1.0/1.0
Active Area	1.3×3.1 mm^2
Transistor Count	8500
Multiplication Time(4 V)	3.8 ns (300 K), 2.6 ns (77 K)
Power Dissipation(10 MHz)	257 mW (300 K), 60 mW (77 K)

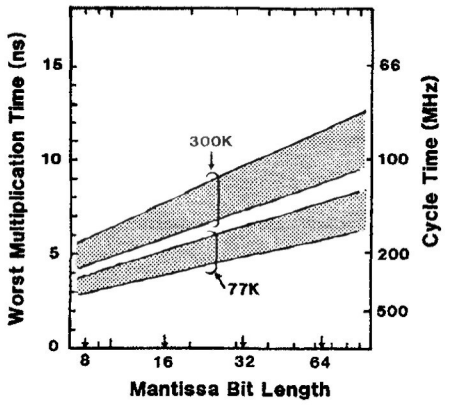


Fig. 14. Calculated floating-point multiplication time versus mantissa bit length.

transistors (HEMT's). Previously, CMOS multipliers dissipated power one order of magnitude lower than the others. However, the multiplication time was 2–3 times larger than those of fast multipliers. By contrast, the present CPL's multiplication times at both room and liquid-nitrogen temperatures are faster than those of any other devices. In addition, the power dissipation is much lower than those of the other devices. The features of this multiplier are summarized in Table I.

The estimated performance of a multiplier using CPL and the half-micrometer CMOS devices is shown in Fig. 14, considering that the bit length is enlarged and floating-point multiplication functions are included. The architecture is assumed to be a combination of Booth's algorithm and Wallace-tree adder array. The bit length and the depth of the carry-lookahead unit are assumed to be optimized for the mantissa bit length. The variations of process, supply voltage, and temperature are considered simply by multiplying the empirical factor by the typical delay time. The multiplication time increases with increasing mantissa bit length. However, the 64-b floating-point multiplication time is estimated to be 10 ns at room temperature and 6 ns at liquid-nitrogen temperature. Thus, operation at over 100 MHz is possible even at room temperature if the multiplier architecture and the carry-lookahead circuit configurations are carefully optimized for high-speed operation.

VI. CONCLUSIONS

This paper described a fast 16×16-b multiplier using a new differential CMOS logic family, CPL. In CPL, differential logic is constructed without pMOS latching load, enabling a speed more than twice as fast as conventional CMOS. The power dissipation is also smaller due to smaller input capacitance. The multiplier is the fastest ever reported at both 300 K and 77 K, proving that the half-micrometer CMOS technology fully utilizing CPL has a speed which is at least competitive with those of other fast devices with a much smaller power dissipation at room temperature, and is faster at liquid-nitrogen temperature. These results also demonstrate that half-micrometer CMOS devices fully utilizing CPL have a performance potential of a 100-MHz repetition rate for floating-point multiplication by carefully optimizing the multiplier architecture for high-speed operation. Therefore, very high-speed MPU's, DSP's, FPU's, and ASIC's are possible.

ACKNOWLEDGMENT

The authors wish to thank Y. Sakai and O. Minato from Hitachi Semiconductor Development Center and T. Masuhara, T. Nakagawa, T. Baji, K. Kaneko, T. Sawase, and K. Ishibashi from Hitachi Central Research Laboratory for their useful suggestions and discussions. The authors are also greatly indebted to K. Yagi and the device processing staff members from Hitachi Central Research Laboratory for their support throughout the sample fabrication, and to A. Kawamata from Hitachi VLSI Engineering Corporation and K. Ueda from Hitachi Central Research Laboratory for their layout design.

REFERENCES

[1] K. Sasaki *et al.*, "A 9ns 1Mb CMOS SRAM," in *ISSCC Dig. Tech. Papers*, 1989, pp. 34–35.
[2] Y. Oowaki *et al.*, "A 7.4ns CMOS 16×16 multiplier," in *ISSCC Dig. Tech. Papers*, 1987, pp. 52–53.
[3] M. Suzuki, M. Hirata, and S. Konaka, "43ps/5GHz bipolar macrocell array LSIs," in *ISSCC Dig. Tech. Papers*, 1988, pp. 70–71.
[4] K. Kajii *et al.*, "A 40 ps high electron mobility transistor 4.1K gate array," in *IEEE 1987 Custom Integrated Circuit Conf.*, pp. 199–202.
[5] L. G. Heller, W. R. Griffin, J. W. Davis, and N. G. Thoma, "Cascode voltage switch logic: A differential CMOS logic family," in *ISSCC Dig. Tech. Papers*, 1984, pp. 16–17.
[6] L. C. M. G. Pfennings, W. G. J. Mol, J. J. J. Bastiaens, and J. M. F. van Dijk, "Differential split-level CMOS logic for sub-nanosecond speed," in *ISSCC Dig. Tech. Papers*, 1985, pp. 212–213.
[7] K. M. Chu and D. L. Pulfrey, "A comparison of CMOS circuit techniques: Differential cascode voltage switch logic versus conventional logic," *IEEE J. Solid-State Circuits*, vol. SC-22, pp. 528–532, 1987.
[8] T. Kengaku, Y. Shimazu, T. Tokuda and O. Tomisawa, *IECE Japan*, 2-83, 1987.
[9] M. Uya, K. Kaneko, and J. Yasui, "A CMOS floating point multiplier," in *ISSCC Dig. Tech. Papers*, 1984, pp. 90–91.
[10] S. Hanamura *et al.*, "Operation of bulk CMOS devices at very low temperatures," in *1983 Symp. VLSI Tech.*, pp. 46–47.
[11] T. Vacca *et al.*, "A cryogenically cooled CMOS VLSI supercomputer," *VLSI Syst. Design*, vol. VIII, no. 7, pp. 80–88, 1987.

Lean Integration: Achieving a Quantum Leap in Performance and Cost of Logic LSIs

KAZUO YANO, YASUHIKO SASAKI, KUNIHITO RIKINO* AND KOICHI SEKI

ULSI Research Center, Hitachi Central Research Laboratory, Kokubunji, Tokyo 185, Japan
*Hitachi Device Engineering, Kokubunji, Tokyo 185, Japan

ABSTRACT

Lean integration aims at a fundamental change in top-down design by following the path from CISC to RISC. The central idea is a lean cell, which has a tree-shaped nMOS network with input ports placed at the end of an every branch of the tree. A lean cell has flexibility of transistor-level circuit design and full compatibility with conventional cell-based design. An extremely simple lean-cell library with only 7 cells and a synthesis tool called "Circuit Inventor," which uses the lean cells, are developed and they are compared with the conventional "complex" CMOS library that has over 60 cells. The results show that the area, the delay, and the power dissipation are improved by lean integration and performance-cost ratio is improved by a factor of three.

INTRODUCTION

Due to recent progress in top-down LSI design using logic synthesis and HDL, the cell-library design is becoming a key factor in achieving high performance ASICs and MPUs. This is quite natural if we recall that the cell library corresponds to the "instruction sets" if we compare the LSI design to the CPU design. Recently even the evaluation method of cell-library quality has been seriously considered [1]. A conventional CMOS cell library usually has over 60 cells even if we limit the cells to the combinational logic. We postulate that this complex library is, compared to a "CISC," causing unnecessary silicon and engineering costs, and a much leaner LSI design method just like RISC should be conceived. In fact, creating and maintaining the library is becoming such a serious burden in LSI design that it is creating opportunities for companies to provide library-design services.

In this work, we propose lean integration, a completely new cell-library architecture which achieves a quantum leap in performance and cost for logic LSIs. We investigate the impact of this method by comparing overall figures with those of CMOS.

LEAN CELL

The proposed lean-cell library is compared with a conventional CMOS library in Table 1 and Fig. 1. The lean-cell library has only 7 cells, which is far smaller than a conventional library. The number of essential logic cells is also less and it is only 3, Y1, Y2, Y3 (Fig. 1). The other 4 cells are simple inverters.

Another feature of lean cells is that they are defined by the *transistor network topology*, a binary tree-shaped nMOS network, rather than the Boolean function. The name "lean" came from the fact that the situation is just like transistors are directly connected to the cell ports of the cell. Although the cell works as a multiplexer, the essential advantage of the lean cell is that transistor-level circuit engineering is possible by using this cell. In fact the transistors act as pass-transistors, or source-grounded configuration, or source-follower configuration depending on the input configuration. By contrast, the conventional cell is defined by its Boolean function, which is often chosen based on previous case studies, and the inner circuit configuration is simply the means to meet the function requirement.

The advantage of a lean cell is that it changes the function by changing the configuration of the cell input ports (Fig. 3). The drain port, the end of a branch of the tree, has the freedom to be connected with the output of another cell, or a power supply line, or a ground line. Different input configurations correspond to different Boolean functions. Note that a very complex logic function is achieved by a single cell. This functionality came from that the transistor-level circuit engineering already described.

Despite this transistor-level flexibility of the lean cell, it is fully compatible with the framework of cell-based design. As a result the delay of the cell can be defined as a function of the load capacitance. This is made possible by the output inverter, which separates the inputs from the output. The feedback inverter and the pull-up pMOS, both consisting of minimum-size MOSFETs, are added to avoid DC leakage current in the CMOS inverter.

Because preparing and updating the lean-cell library requires only small engineering cost, it is much easier to adopt the state-of-the-art process technology even in a tight schedule constraint. Therefore, the lean cell encourages concurrent interaction between logic designers and process engineers. By contrast, major revision of the conventional library, which includes cell-layout data, logic-synthesizer data, and automatic place and router data for more than 60 cells, requires much more engineering effort and is sometimes unrealistic.

The area of a logic block is reduced by using lean cells. The logic area is given by the following equation.

$$\text{Logic area} = \text{Net count} \times \frac{1}{\text{Nets per cell}} \times \text{Cell area} \quad (1)$$

where net count is mainly determined by the logic synthesis algorithm, nets per cell (NPC) is mainly determined by the cell architecture, and cell area is mainly determined by the

Reprinted from *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 26.5.2 - 26.5.4, May 1994.

technology level. This relation corresponds to the well-known relation used in CPU design:

$$\text{CPU time} = \text{Instruction count} \times \frac{1}{\text{Instructions per cycle}} \times \text{Cycle time} \quad (2)$$

In Eq.(1) a logic function is considered to be a box which reduces the number of nets, or nodes. The lean cell, which has a large NPC without increasing the cell area, has a high capability of reducing nets. Therefore the number of cells required to build a logic block is smaller than that of the CMOS, resulting in smaller block area. A similar argument holds for power consumption, which leads to lower power consumption.

The delay of the logic block is also reduced by the lean cells. Very complex logic functions, which are not included in conventional CMOS libraries, can be achieved by using only a single lean cell. Therefore, the number of critical-path cells is reduced. In addition, complex CMOS gates with large parasitic capacitance are slow and have low current drive capability. By contrast, parasitic capacitance of lean cells is small and high current drive capability is possible due to the output inverter.

The transition from a CMOS library to the lean-cell library is just like the transition from CISC to RISC (Table. 2). A lean-cell library corresponds to the small instruction set of RISCs. The instructions of the RISC were not convenient for designers who were accustomed to the conventional orthogonal instructions of CISCs. However, this has been overcome by using an optimized compiler. A lean cell has a similar characteristic. Because it is somewhat like directly controlling the transistor behavior from outside of the cell, logic designers, who are not familiar with the details of individual circuit may become reluctant to deal with them. However, logic synthesis based on lean cells solves this problem. The high performance of RISCs is explained by the large number of instructions per cycle. The high performance-cost ratio of lean integration is explained by its larger nets per cell. The simple instruction set of RISCs and the especially good expectation of a relation between the performance and the instruction sequence help the compiler to provide highly efficient instructions. The lean cells also help the synthesizer to provide area- and delay-effective net lists.

We also developed a logic synthesis tool called "Circuit Inventor", which fully utilizes the lean cell characteristics. Circuit Inventor accepts an HDL description, creates net lists based on lean cells and gives those data to the layout tool. It expresses the required logic function in a compact form by using a reduced BDD (Binary Decision Diagram) [2] and conducts various optimizations. BDD has the same network topology as lean cells and efficient mapping to cells is possible. Automatic insertion of optimized inverters into heavily-loaded nodes is possible. The details of the internal algorithm of Circuit Inventor will be described elsewhere.

EXPERIMENTS AND DISCUSSION

The performance of lean cells is compared with that of CMOS cells. Two types of benchmark logic are chosen. One is a 4-b adder/subtracter, which represents arithmetic logic, and the other is 7-input 4-output random logic, which is created by assigning random numbers to the output of the truth table. CMOS logic is synthesized by using a popular commercial logic synthesis tool. 0.5-μm process with 3-level metal is assumed and poly-cell-type layout style is used. Metal 1 is assigned to the intra-cell wiring, metal 2 is assigned to Y-direction inter-cell wiring, and metal 3 is assigned to X-direction inter-cell wiring. The cell input/output ports are formed as through-holes between metal 1 and metal 2. The critical path and the wiring load was extracted from the layout data (Fig. 4) and the delay was obtained by using circuit simulation. The power consumption was determined by using circuit simulation of the total circuit.

The results are dramatic (Table. 3). The lean cells show higher figures in all respects including area, delay, and power consumption for either benchmark. If we define the performance-cost ratio of a cell architecture by using the product of the area, delay, and power, the lean cells has 3-4 times learger ratio.

The average NPC is actually boosted in the lean integration from 2.8 to 4.7 (Table 4), which is the major contributor of the area reduction. The delay per cell is smaller in the lean cells, which contributes the delay reduction.

The dependence of the delay on supply-voltage is another important aspect of the technology choice. The lean cells become slower than CMOS cells at supply voltages below the critical value, because of the influence of threshold voltage of the nMOS. However, under practical conditions, where the extrapolated threshold voltage is smaller than $V_{CC}/2.7$, the lean cells are always faster than the CMOS cells as shown in Fig. 5.

CONCLUSIONS

Lean integration, which provides a quantum leap in performance and cost of ASICs and MPUs is proposed. This new design method goes far beyond marginal improvements and reaches 3-4 times improvement in performance-cost ratio and gives much higher competitiveness in "lean LSIs" and in systems that use LSI chips.

ACKNOWLEDGEMENTS

The authors would like to thank E. Takeda, K. Uchiyama, S. Narita, T. Noguchi, N. Kageyama, M. Tonomura of Hitachi Central Research Laboratory for their valuable discussions.

[1] H. Harvey-Horn, "User-defined benchmarks help evaluate IC physical libraries," Electronics Design, Oct., 80 (1993)

[2] R. E. Bryant, "Graph-based algorithm for Boolean function manipulation," IEEE Computers, Vol-C35 (8), 677(1986)

Table 1 Cell lists of coventional and proposed lean-cell library

Conventional CMOS Library (61 cells)			Lean-Cell Library (7 cells)
INVERTER	4AND	2OR/2AND	Y1
INVERTER_P2	4AND_P2	2OR/2AND_P2	Y2
INVERTER_P4	2OR	3OR/2AND	Y3
INVERTER_P8	2OR_P2	3OR/2AND_P2	INVERTER
2NAND	3OR	2ANDx2/2OR	INVERTER_P2
3NAND	3OR_P2	2ANDx2/2OR_P2	INVERTER_P4
2NOR	4OR	3ANDx2/2OR	INVERTER_P8
2NOR_P2	4OR_P2	3ANDx2/2OR_P2	
3NOR	2AND/2NOR	2ANDx3/3OR	
3NOR_P2	2ANDx2/2NOR	2ANDx3/3OR_P2	
4NOR	3AND/3NOR	2ANDx2/3OR	
4NOR_P2	2OR/2NAND	2ANDx2/3OR_P2	
2XOR	2OR/3NAND	2ORx2/2AND	
2XOR_P2	3OR/2NAND	2ORx2/2AND_P2	
2XNOR	2AND/2OR	2ANDx4/4OR	
2XNOR_P2	2AND/2OR_P2	2ANDx4/4OR_P2	
2AND	3AND/2OR	8NAND	
2AND_P2	3AND/2OR_P2	8NAND_P2	
3AND	2AND/3OR	8AND	
3AND_P2	2AND/3OR_P2	8AND_P2	
		8OR	

(_P2, _P4, _P8 represent x2, x4, x8 powered cells)

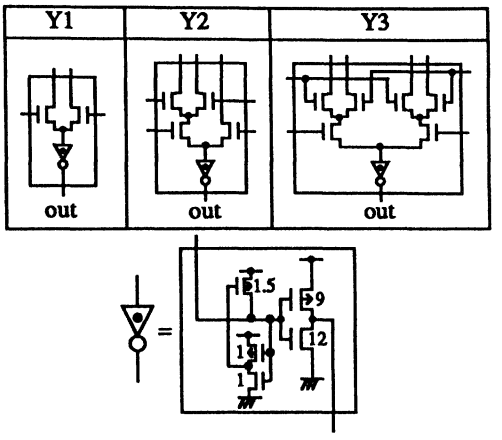


Fig. 1 Circuit diagram of lean cells and the output inverters

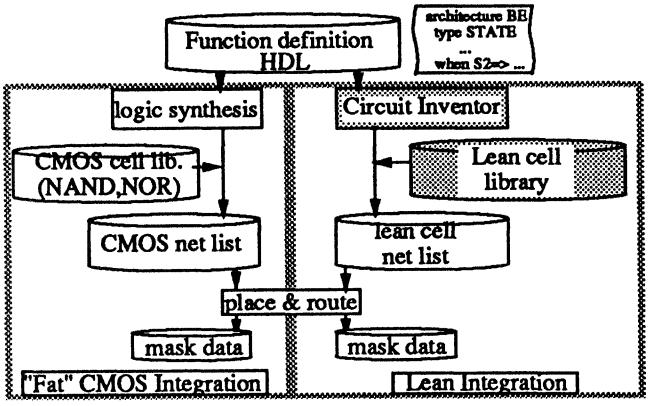


Fig. 2 Conventional "fat" CMOS integration vs. lean integration

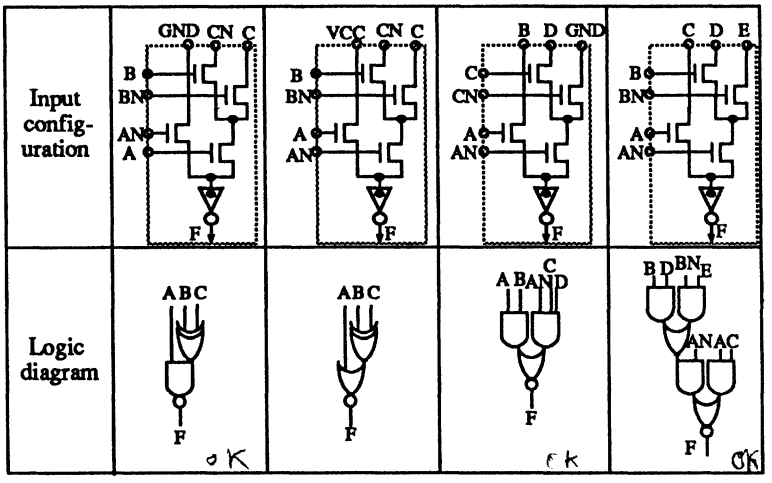


Fig. 3 Various logic fuctions of the lean cell "Y2"

Table 2 Transition from CMOS "fat" library to lean-cell library is compared to "CISC->RISC" transition

	CISC	RISC		"fat" CMOS	lean
No. instructions	many instructions	fewer instructions	No. cells	many cells	fewer cells
instruction fuction	orthogonal self-contained instructions	Load/Store architecture	Cell logic function	self-contained logic(NAND,NOR)	Select/Amplify
Instructions/cycle	low(~1/3)	high(~1)	Nets/cell	low(~2.8)	high(~4.7)
programming	assembler	high-level language	design	schematic	HDL

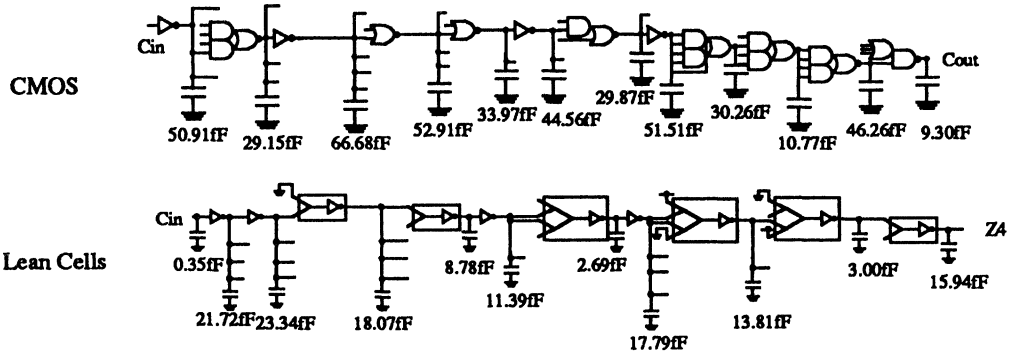


Fig. 4 synthesized critical path circuit of the 4-b adder/subtractor

Table 3 Summary of benchmark design

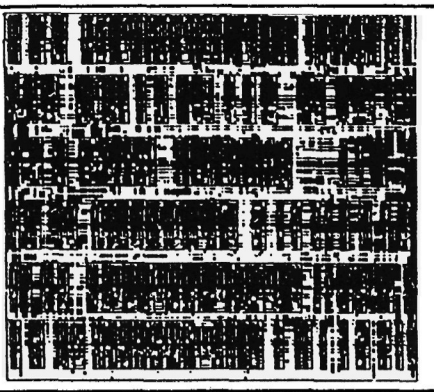
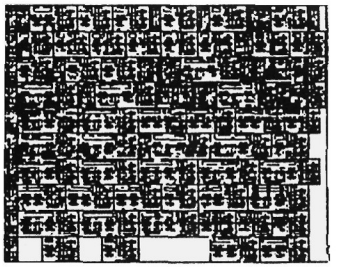
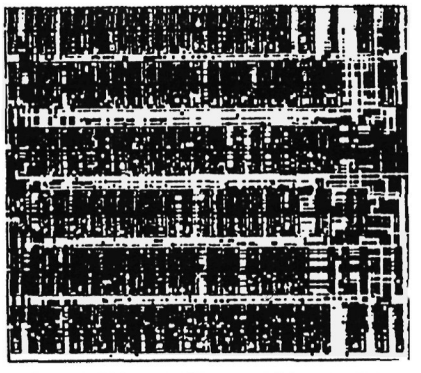
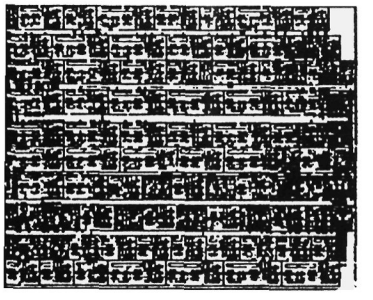
	CMOS	Lean
	4-bit ADD-SUB	
Layout		
AREA	84288.6μm^2(1.0)	48589.2μm^2(0.55)
Delay Time	3.620ns (1.0)	2.691ns (0.74)
Tr.Count	828 (1.0)	545 (0.66)
Gate Width	7583μm (1.0)	3091μm (0.41)
Net Count	386 (1.0)	400 (1.04)
Power	6.08mW/MHz (1.0)	3.84mW/MHz (0.63)
Cell Count	133 (1.0)	85 (0.64)
Critical Path	12 (1.0)	10 (0.83)
	7in Random Logic	
Layout		
AREA	86786.04μm^2(1.0)	60819.44μm^2(0.70)
Delay Time	2.284ns (1.0)	1.590ns (0.70)
Tr.Count	800 (1.0)	644 (0.81)
Gate Width	7530μm (1.0)	3741μm (0.50)
Net Count	385 (1.0)	473 (1.23)
Power	5.87mW/MHz (1.0)	3.58mW/MHz (0.61)
Cell Count	136 (1.0)	98 (0.72)
Critical Path	10 (1.0)	10 (1.0)

Table 4 Comparison of figures per cell

	4-bit ADD-SUB		7-in Random Logic	
	CMOS	Lean	CMOS	Lean
Tr. Count /Cell	6.22	6.41	5.88	6.57
Gate Width / Cell	57μm	36μm	55μm	38.2μm
Net Count / Cell	2.9	4.7	2.8	4.8
Area / Cell	633μm^2	571μm^2	638μm^2	620μm^2
Delay Time / Cell (Average wire length)	0.302ns (351μm)	0.269ns (133μm)	0.286ns (334μm)	0.197ns (206.5μm)
Power / Cell	45.7μW/MHz	45.2μW/MHz	43.2μW/MHz	36.5μW/MHz

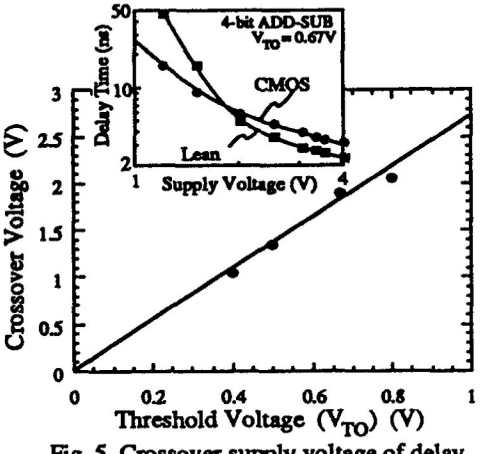


Fig. 5 Crossover supply voltage of delay between CMOS and lean cells

A 1.5-ns 32-b CMOS ALU in Double Pass-Transistor Logic

Makoto Suzuki, *Member, IEEE,* Norio Ohkubo, Toshinobu Shinbo, Toshiaki Yamanaka, *Member, IEEE,* Akihiro Shimizu, Katsuro Sasaki, *Member, IEEE,* and Yoshinobu Nakagome, *Member, IEEE*

Abstract—**This paper describes circuit techniques for fabricating a high-speed adder using pass-transistor logic. Double pass-transistor logic (DPL) is shown to improve circuit performance at reduced supply voltage. Its symmetrical arrangement and double-transmission characteristics improve the gate speed without increasing the input capacitance. A carry propagation circuit technique called conditional carry selection (CCS) is shown to resolve the problem of series-connected pass transistors in the carry propagation path. By combining these techniques, the addition time of a 32-b ALU can be reduced by 30% from that of an ordinary CMOS ALU. A 32-b ALU test chip is fabricated in 0.25-μm CMOS technology using these circuit techniques and is capable of an addition time of 1.5 ns at a supply voltage of 2.5 V.**

I. Introduction

ENHANCING the performance of macros is essential to the construction of high-performance microprocessors where macrointensive design is used to achieve a high MIPS performance. Of the many data path macros, ALU's, or adders, are the key components in processor chips for ALU's in execution units, floating-point adders, and final carry propagation adders in floating-point multipliers and digital signal processing units. A number of fast adder architectures have been proposed in the long history of computer arithmetic [1]–[7], some of which use pass-transistor logic for carry propagation [6], [7]. Pass-transistor logics gain their speed advantage over CMOS due to their high logic functionality. However, a problem with this architecture is the series connection of the pass transistors in the carry propagation path.

This paper describes circuit techniques for realizing a faster adder using pass-transistor logic. A carry propagation technique called conditional carry selection (CCS) has been developed to solve the series connection problem, and double pass-transistor logic (DPL) has been developed to improve circuit performance at reduced supply voltage. A symmetrical arrangement and the double-transmission characteristics of the DPL gate compensate for the speed degradation due to the usage of both PMOS and NMOS pass transistors. Applying these circuit techniques, the addition time of a 32-b ALU can be reduced by 30% from that of an ordinary CMOS ALU. A 1.5-ns 32-b ALU has been developed using 0.25-μm CMOS technology and these circuit techniques [8].

Double pass-transistor logic and its characteristics are discussed in Section II. The conditional carry-selection circuit is described in Section III. Section IV describes the architecture and simulated results of a fabricated 32-b ALU test chip. Some experimental results are shown in Section V, and the conclusions are summarized in Section VI.

Manuscript received May 18, 1993; revised August 5, 1993.

M. Suzuki, N. Ohkubo, T. Yamanaka, and Y. Nakagome are with the Central Research Laboratory, Hitachi Ltd., Tokyo 185, Japan.

T. Shinbo and A. Shimizu are with Hitachi VLSI Engineering Corporation, Tokyo 187, Japan.

K. Sasaki is with the R&D Division, Hitachi America Ltd., Brisbane, CA 94005-1819.

IEEE Log Number 9212552.

II. Double Pass-Transistor Logic

Several pass-transistor logic families for macrocell design have been proposed for improving the performance of CMOS circuits. Complementary pass-transistor logic (CPL) [9] is one example; it has been applied to the full adders in multiplier circuits and has been shown to result in high speed due to its low input capacitance and high logic functionality. However, when implementing CPL, particularly in reduced supply voltage designs, it is important to take into account the problems of noise margins and speed degradation. These are caused by mismatches between the input signal levels and the logic threshold voltage of the CMOS inverters, which fluctuates with process variations. DPL is a modified version of CPL that meets the requirement of reduced supply voltage designs.

A. DPL Gate

A basic circuit diagram of a DPL gate is shown in Fig. 1. By simply exchanging the input nodes, two-input AND/NAND, OR/NOR, XOR/XNOR gates and multiplexers can be constructed. The DPL gate consists of complementary inputs/outputs and is thus a dual rail logic like CPL. Dual rail logic has been widely used for other logic families, such as clocked CVSL [10], and for self-timed logic [11] and bipolar DCS logic [12]. DPL gates consist of both NMOS and PMOS pass transistors, in contrast to CPL gates, where only NMOS pass transistors are used.

Fig. 2 compares the construction of XOR gates in CPL, CMOS, and DPL pass-transistor logics. The CPL gate consists only of NMOS transistors, resulting in low input capacitance and high-speed operation. However, the above-mentioned problems are caused by the high output signal level being lower than the supply voltage V_{CC} by the NMOS threshold voltage V_{th}. The usual way to avoid this is to use CMOS pass-transistor logic. Full-swing operation is attained by simply adding PMOS transistors in parallel with the NMOS transistors. However, this addition results in increased input capacitance.

Reprinted from *IEEE Journal of Solid-State Circuits,* Vol. 28, No. 11, pp. 1145-1150, November 1993.

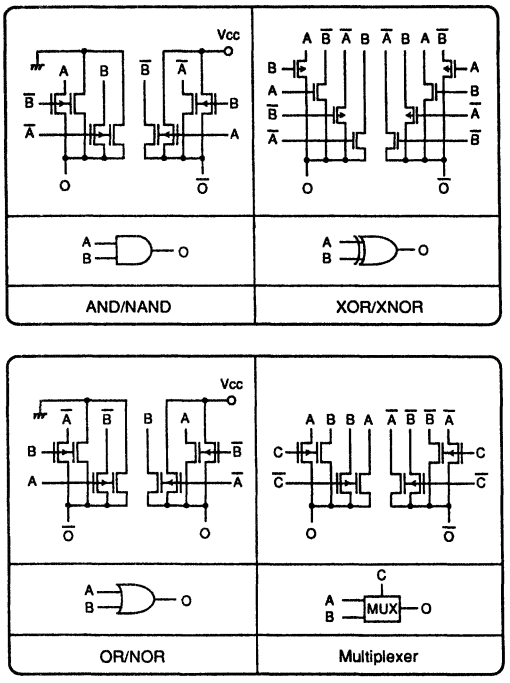


Fig. 1. Double pass-transistor logic (DPL) gates.

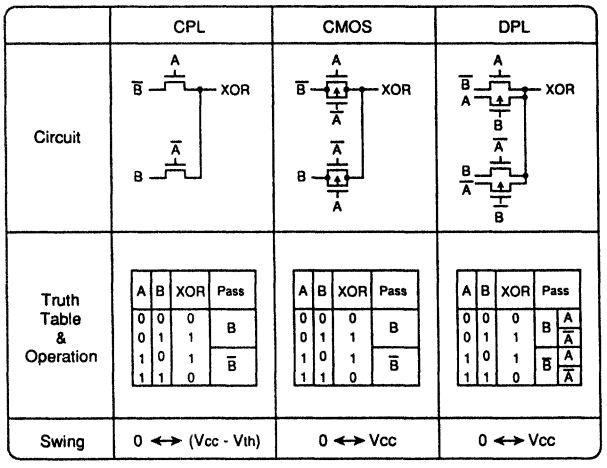


Fig. 2. Comparison of CPL, CMOS, and DPL pass-transistor logics for XOR gates.

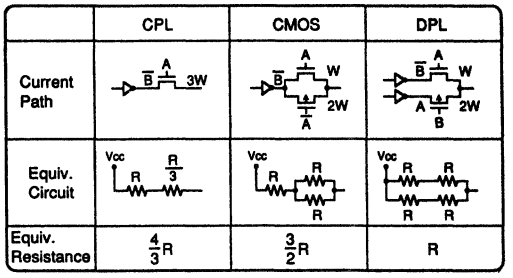


Fig. 3. Comparison of equivalent resistance for CPL, CMOS, and DPL pass-transistor logics.

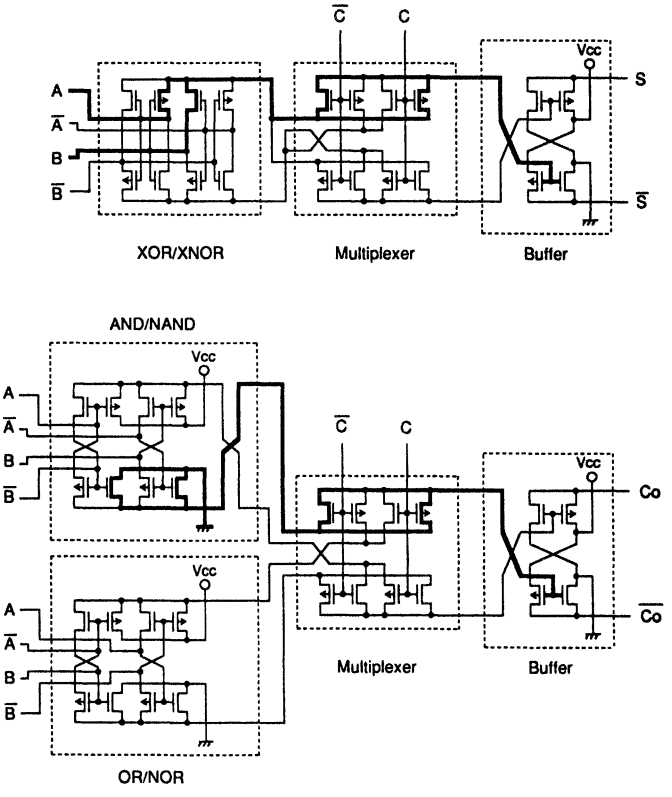


Fig. 4. DPL full adder.

In the DPL gate, the inputs to the gates of the PMOS transistors are changed from A to B. This arrangement compensates for the speed degradation of CMOS pass-transistors in two ways. First, it is a symmetrical arrangement whereby any input is connected to the gate of one MOSFET and the source of another. In the case of the XOR/XNOR, as can be seen in Fig. 1, it is perfectly symmetrical. Any of the inputs A, $\overline{A}$, B, and $\overline{B}$ is connected to the gates of the NMOS and PMOS and to the sources of the NMOS and PMOS. This results in a balanced input capacitance and reduces the dependence of the delay time on data.

Secondly, it has double-transmission characteristics. The truth tables in Fig. 2 show how the pass transistors operate for the XOR function. In this table, the column labeled Pass shows which signals are passed and performs the XOR function. For example, in the DPL gate, both A and B are passed when A and B are low. In both the CPL and CMOS implementations, the gate input A or $\overline{A}$ controls the pass transistors. When A is low, B is passed, and $\overline{B}$ is passed when A is high. In the DPL gate, on the other hand, there are two types of pass transistors: one is controlled by A and the other by B. The A-controlled pass transistors operate in the same way as CPL and CMOS. For the B-controlled pass transistors, when B is low, A is passed, and $\overline{A}$ is passed when B is high. As a result, there are always two current paths driving the buffer stage.

Fig. 3 compares the equivalent resistance of the pass transistors. In order to compare the driving source impedance, the equivalent resistance includes that of the CMOS buffer with the same input capacitance. This comparison is rather simplified, but it qualitatively illustrates the double-transmission property. In the DPL design, the widths of the NMOS and PMOS pass transistors are one-third and two-thirds, respectively, of the NMOS pass transistor in the CPL gate, so the input capacitance and the gate area are nearly the same for all these architectures. As shown in Fig. 3, the resistance, including that of the CMOS buffer of the previous stage, is smallest for the DPL gate due to its double-transmission property.

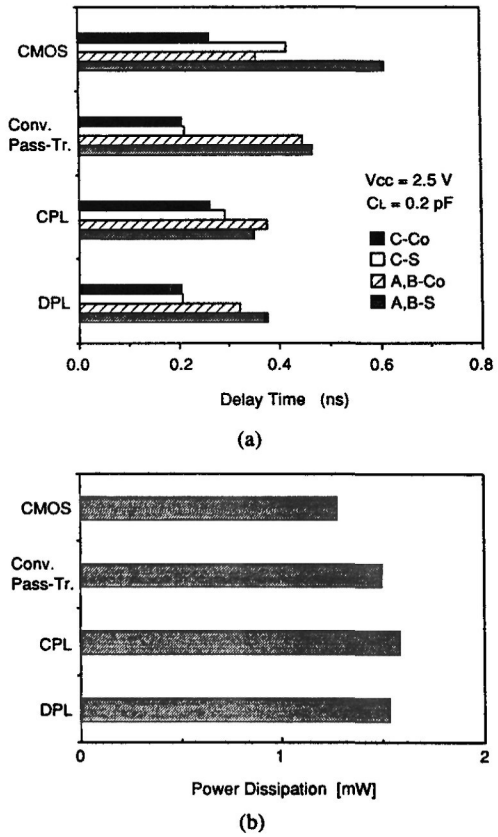


Fig. 5. Comparison of four types of full adders. (a) Delay times. (b) Power dissipation.

B. DPL Full Adder

We evaluated the speed advantage of the DPL gate using a full adder as an example. Fig. 4 shows the circuit of this full adder. The sum output portion consists of XOR/XNOR gates, a multiplexer, and a CMOS output buffer. The carry output portion consists of AND/NAND gates, OR/NOR gates, a multiplexer, and a CMOS output buffer. The current paths for the $\overline{S}$ and $\overline{Co}$ outputs when A, B, and C are all low, for example, are shown by the bold lines. These current paths include two pass transistors, and there are two current paths for each output, as discussed above.

Fig. 5 compares the simulated delay times and power dissipation of four kinds of full adders: CMOS [13], conventional CMOS pass-transistor logic [7] arranged in a dual rail structure, CPL [9], and DPL, with a load capacitance of 0.2 pF. For the slowest path that determines the speed of, for example, a multiplier, the DPL full adder is as fast as CPL, 18% faster than the conventional pass-transistor logic, and 37% faster than CMOS. As for the carry output delays ($C - Co$ and $A - Co$) that determine the ALU speed, the DPL full adder is the fastest of all. The power dissipation is simulated for 250-MHz operation with 0.2 pF loaded on each output regardless of whether it is single or dual rail logic. Under these conditions, the pass-transistor architectures show slightly higher power dissipation than CMOS because they have dual rail structure and double the load capacitance. The load capacitance determines which architecture dissipates the least power, and at lower load capacitance the dual rail pass-transistor architectures dissipate less power than CMOS.

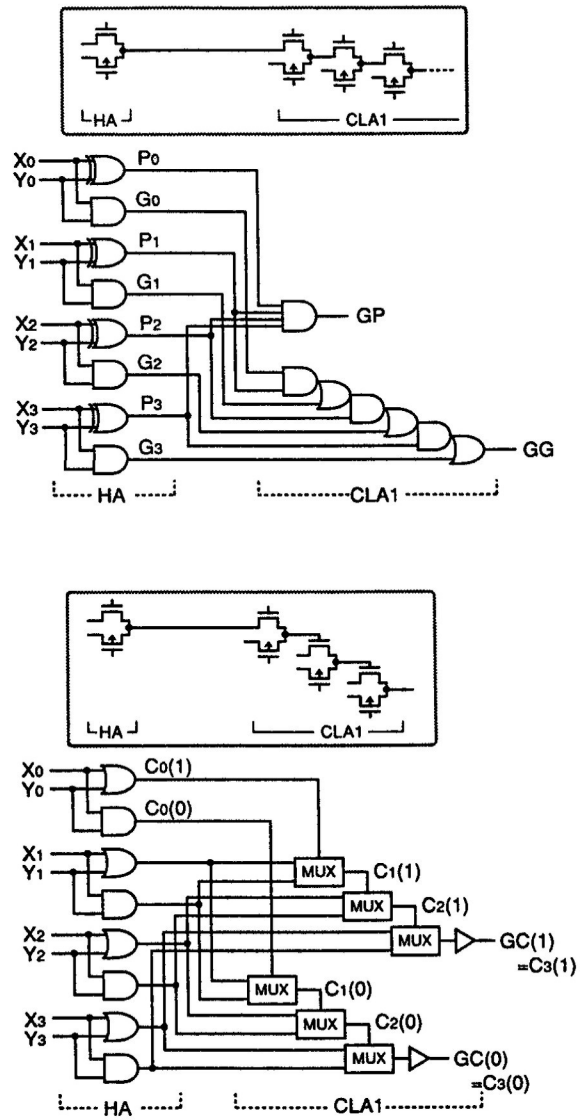


Fig. 6. 4-b carry look-ahead circuits. (a) Conventional AND–OR carry look-ahead circuit. (b) Conditional carry select (CCS) circuit.

III. Conditional Carry Selection Circuit

The most important component of high-speed ALU's is a look-ahead carry circuit. We have developed a new look-ahead carry scheme, called conditional carry selection (CCS). Fig. 6 compares a 4-b implementation of this scheme with a conventional AND-OR carry look-ahead circuit. In the conventional circuit [Fig. 6(a)], generated carry signals (G_j) are propagated (P_j) through an AND-OR circuit chain to form a group-generate (GG) signal, which is expressed as

$$\begin{aligned} C_3 &= G_3 + P_3 \cdot [G_2 + P_2 \cdot (G_1 + P_1 \cdot G_0)] \\ &\quad + P_3 \cdot P_2 \cdot P_1 \cdot P_0 \cdot C_{-1} \\ &= GG + GP \cdot C_{-1} \qquad (1) \\ GG &= G_3 + P_3 \cdot [G_2 + P_2 \cdot (G_1 + P_1 \cdot G_0)] \qquad (2) \\ GP &= P_3 \cdot P_2 \cdot P_1 \cdot P_0. \qquad (3) \end{aligned}$$

Thus, the 4-b carry look-ahead circuit involves three AND-OR circuits in its critical path. Furthermore, using pass-transistor

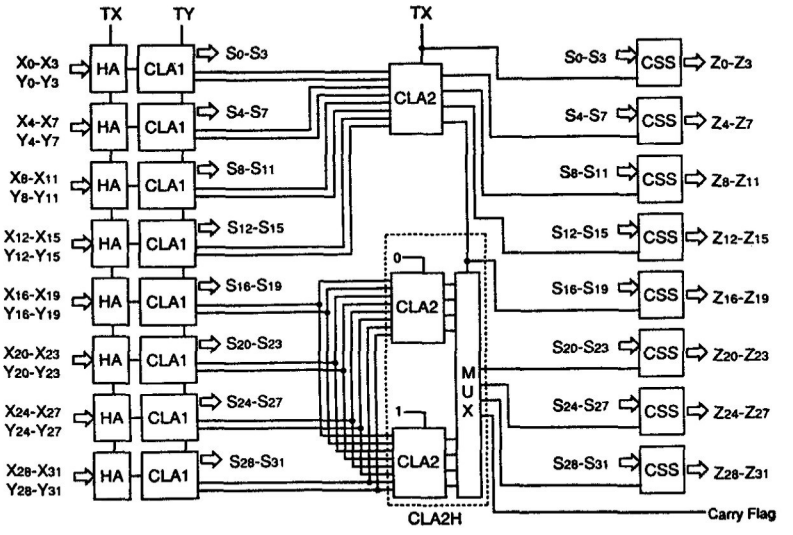


Fig. 7. Block diagram of the 32-b ALU.

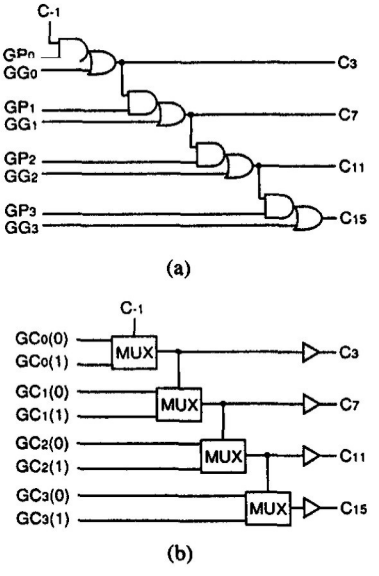


Fig. 8. Four-block carry look-ahead circuit CLA2. (a) Conventional AND-OR carry look-ahead circuit. (b) Conditional carry select (CCS) circuit.

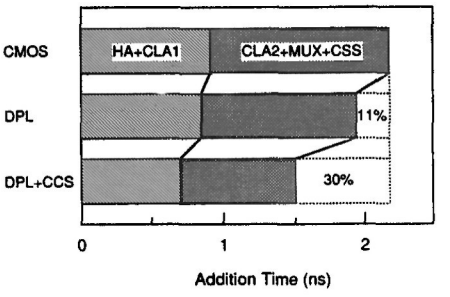


Fig. 9. Simulated comparison of 32-b ALU addition times.

logic, there are seven pass transistors connected in series, as shown in the inset figure above.

On the other hand, in the CCS architecture conditional carry signals for each bit $C_j(0)$ (assuming an incoming group carry of 0) or $C_j(1)$ (assuming an incoming group carry of 1) are selected by the multiplexers depending on the conditional carry signals of the previous bit, $C_{j-1}(0)$ or $C_{j-1}(1)$, as expressed by

$$C_j(k) = G_j + P_j \cdot C_{j-1}(k)$$
$$= G_j = X_j \cdot Y_j \quad (\text{if } C_{j-1}(k) = 0) \tag{4}$$
$$= G_j + P_j = X_j + Y_j \quad (\text{if } C_{j-1}(k) = 1) \tag{5}$$
$$k = 0 \text{ or } 1.$$

This conditional carry selection procedure finally forms the conditional group carries $GC(0)$and $GC(1)$. In this way, the critical carry propagation path can be constructed by three multiplexers instead of three AND-OR gates. As shown in Fig. 6(b), the CCS architecture also avoids the series connection of pass transistors in the multiplexers, as used in a transmission-gate conditional-sum adder [7], Manchester carry chain [14], or the AND-OR carry look-ahead circuit of Fig. 6(a).

IV. 32-b ALU Architecture

Fig. 7 shows a block diagram of the 32-b ALU based on carry select architecture [3]. DPL gates are used for all the circuits from the half adders (HA) to the final conditional-sum selection (CSS) circuits. The CCS architecture is applied not only to the 4-b carry look-ahead circuit CLA1 but also to the block carry look-ahead circuit CLA2, where four AND-OR circuits can be replaced with four multiplexers, as shown in Fig. 8. The CSS circuit consists of a multiplexer that selects the conditional sums, $S_j(0)$ or $S_j(1)$, according to the incoming block carry signal. In the carry look-ahead circuit, the upper 16 bits are processed by a conditional carry selection method whereby block carry signals are generated by CLA2, assuming the carry of the lower 16 bits C_{15} to be 0 or 1, and are then selected by the multiplexer according to the incoming true carry. This architecture enhances parallelism and results in fast operation. This is because the carry signals of the upper 16 bits are calculated in parallel with those of the lower 16 bits, and the carry signals of the upper 16 bits are generated after the delay time of a single multiplexer. The TX and TY signals select the function of the ALU.

Fig. 9 compares the simulated addition times of 32-b ALU's. An ordinary CMOS ALU uses the carry look-ahead circuits

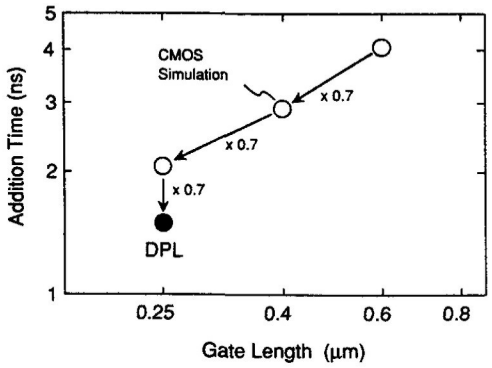


Fig. 10. Simulated addition time improvement with decreasing device dimensions.

TABLE I
PROCESS TECHNOLOGY

Technology	0.25-μm CMOS Triple Metal
MOSFET	
Gate Length	0.25 μm
Gate Oxide	6.5 nm
Contact/Via 1	0.3 μm × 0.3 μm
Via 2	0.6 μm × 0.6 μm
First Metal Width/Space	0.5 μm / 0.4 μm
Second Metal Width/Space	0.5 μm / 0.4 μm
Third Metal Width/Space	0.7 μm / 0.6 μm

of Fig. 6(a) and Fig. 8(a) and the architecture of Fig. 7, with the CMOS combinational gates like a four-input AND-OR-NOT gate. The combination of DPL with a conventional AND-OR carry look-ahead circuit reduces the addition time by 11% from that of an ordinary CMOS ALU, and the combination of DPL with a CCS carry look-ahead circuit reduces the addition time by 30%. Fig. 10 shows the simulated reduction of addition time with decreasing device dimensions. The CMOS simulation was done for an ordinary CMOS ALU with decreasing supply voltages for each generation—5 V, 3.3 V, and 2.5 V, respectively. The addition time is reduced by 30% for each generation. Therefore, the 30% improvement of the DPL and CCS architecture corresponds to a one-generation advance in process technology.

V. EXPERIMENTAL RESULTS

The 32-b ALU test chip described above was fabricated using 0.25-μm triple-metal CMOS technology. The major process parameters are summarized in Table I. Actually, this test chip was fabricated on the same wafer as an SRAM test chip [15]. An i-line stepper was used for all layers. The first metal is tungsten, and the second and third metals are aluminum. A micrograph of the test chip is shown in Fig. 11. It measures 1.58 mm × 0.38 mm (0.6 mm^2). We also designed an ordinary CMOS ALU as discussed in Section IV that measures 2.1 mm × 0.26 mm (0.55 mm^2). The area penalty of the DPL ALU, compared with the CMOS ALU, is thus 10%. This device is capable of performing 32-b additions in 1.5 ns at a supply voltage of 2.5 V, as shown in the

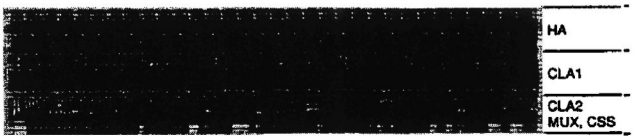


Fig. 11. Micrograph of the 32-b ALU test chip.

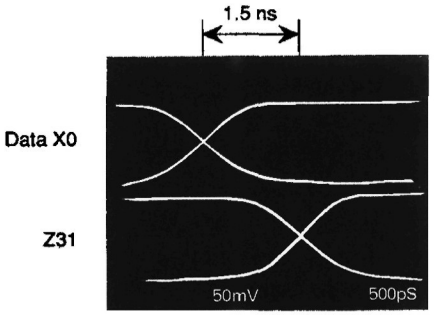


Fig. 12. Measured waveforms of the 32-b ALU test chip (1.58 mm × 0.38 mm).

TABLE II
CHARACTERISTICS OF THE 32-b ALU TEST CHIP

Organization	32-b ALU
Architecture	Carry Select Addition
Circuit	DPL
	CCS CLA
Addition Time	1.5 ns
Power Dissipation	8 mW (at 50 MHz)
Supply Voltage	2.5 V
Chip Size	0.60 mm^2

waveforms of Fig. 12. The power dissipation was measured at the maximum frequency limit (50 MHz) of the pattern generator and was found to be 8 mW with about 12% gate activity. This extrapolates to 107 mW at a 1.5-ns cycle time. The characteristics of this 32-b ALU test chip are summarized in Table II. Fig. 13 shows how the addition time depends on the supply voltage. The solid lines show the simulated results for CMOS and DPL ALU's, and the circles show the results for the DPL ALU measured at room temperature. The DPL ALU has an excellent low-voltage performance and agrees well with the results of circuit simulation. DPL AND/NAND and OR/NOR ring oscillators have also been fabricated, showing speed improvements of 15% and 30% compared with CMOS NAND and NOR ring oscillators, respectively.

VI. CONCLUSION

Double pass-transistor logic (DPL) has been developed to improve circuit performance at reduced supply voltage. Its symmetrical arrangement and double-transmission characteristics compensate for the speed degradation arising from the use of PMOS and NMOS pass transistors. A carry propagation circuit technique called conditional carry selection (CCS) has been developed to solve the problem of series-connected pass transistors in the carry propagation path. By combining these

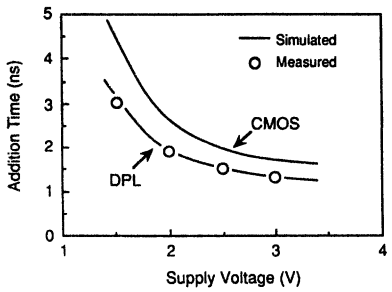


Fig. 13. Variation of the addition time of 32-b ALU's with supply voltage.

circuit techniques, the addition time of the 32-b ALU can be reduced by a substantial 30% from that of an ordinary CMOS ALU. A 1.5-ns 32-bit ALU has been developed using 0.25-μm CMOS technology and these circuit techniques. It should also be possible to apply the proposed DPL gates and CCS adder architecture to other data path macros, such as floating-point units, resulting in processing units with very high performance.

ACKNOWLEDGMENT

The authors wish to thank Dr. K. Shimohigashi, Dr. T. Nishimukai, Dr. T. Nagano, and Dr. M. Hiraki for their useful discussions, and K. Ueda and K. Takasugi for their assistance and support. The authors are also greatly indebted to T. Nishida, N. Hashimoto, N. Ohki, and H. Ishida for their assistance with the process technology and device fabrication.

REFERENCES

[1] J. Sklansky, "An evaluation of several two-summand binary adders," *IRE Trans. Electron. Comput.*, vol. EC-9, pp. 213–226, June 1960.
[2] J. Sklansky, "Conditional-sum addition logic," *IRE Trans. Electron. Comput.*, vol. EC-9, pp. 226–231, June 1960.
[3] O. J. Bedrij, "Carry-select adder," *IRE Trans. Electron. Comput.*, vol. EC-11, pp. 340–346, June 1962.
[4] C. L. Chen, "2.5-V bipolar/CMOS circuits for 0.25-μm BiCMOS technology," *IEEE J. Solid-State Circuits*, vol. 27, pp. 485–491, Apr. 1992.
[5] K. Yano *et al.*, "3.3-V BiCMOS circuit techniques for 250-MHz RISC arithmetic modules," *IEEE J. Solid-State Circuits*, vol. 27, pp. 373–381, Mar. 1992.
[6] H. Hara *et al.*, "0.5-μm 3.3-V BiCMOS standard cells with 32-kilobyte cache and ten-port register file," *IEEE J. Solid-State Circuits*, vol. 27, pp. 1579–1584, Nov. 1992.
[7] A. Rothermel *et al.*, "Realization of transmission-gate conditional-sum (TGCS) adders with low latency time," *IEEE J. Solid-State Circuits*, vol. 24, pp. 558–561, June 1989.
[8] M. Suzuki *et al.*, "A 1.5 ns 32 b CMOS ALU in double pass-transistor logic," *ISSCC Dig. Tech. Papers*, pp. 90–91, Feb. 1993.
[9] K. Yano *et al.*, "A 3.8-ns CMOS 16 × 16-b multiplier using complementary pass-transistor logic," *IEEE J. Solid-State Circuits*, vol. 25, pp. 388–395, Apr. 1990.
[10] L. G. Heller *et al.*, "Cascode voltage switch logic: A differential CMOS logic family," *ISSCC Dig. Tech. Papers*, pp. 16–17, Feb. 1984.
[11] J. Yetter *et al.*, "A 100 MHz superscalar PA-RISC CPU/coprocessor chip," in *1992 Symp. VLSI Circuits Dig. Tech. Papers*, pp. 12–13, June 1992.
[12] E. B. Eichelberger *et al.*, "Differential current switch—high performance at low power," *IBM J. Res. Develop.*, vol. 35, no. 3, pp. 313–320, May 1991.
[13] M. Uya *et al.*, "A CMOS floating point multiplier," *IEEE J. Solid-State Circuits*, vol. SC-19, pp. 697–702, Oct. 1984.
[14] N. Weste and K. Eshragian, *Principles of CMOS VLSI Design: A Systems Perspective*. Reading, MA: Addison-Wesley, 1988.
[15] T. Yamanaka *et al.*, "A 2.3 μm^2, single-bit-line SRAM cell with high soft-error-immune structure," presented at 1993 Symp. VLSI Technology, May 1993.

A 4.4-ns CMOS 54X54-b Multiplier Using Pass-transistor Multiplexer

Norio Ohkubo, Makoto Suzuki, *Toshinobu Shinbo, Toshiaki Yamanaka,
*Akihiro Shimizu, **Katsuro Sasaki, and Yoshinobu Nakagome

Central Research Laboratory, Hitachi Ltd.,
1-280 Higashi-Koigakubo, Kokubunji, Tokyo 185, Japan.

*Hitachi VLSI Engineering Corporation, Kodaira, Tokyo 187, Japan.
**R&D Division, Hitachi America Ltd., Brisbane, CA 94005-1819

Abstract

A 54 X 54-b multiplier using pass-transistor multiplexer has been fabricated by 0.25-μm CMOS technology. To enhance the speed performance, a new 4-2 compressor and a carry look-ahead adder (CLA) both featuring the use of pass-transistor multiplexers have been developed. The new circuits have a speed advantage over conventional CMOS circuits because the number of critical-path gate stages is minimized due to the high logic functionality of pass-transistor multiplexers. The active size of the 54 X 54-b multiplier is 3.77 mm X 3.41 mm. The multiplication time is 4.4 ns at 2.5 V power supply.

Introduction

Enhancing the performance of floating point operation is indispensable for current high-performance microprocessors. In particular, high speed multiplication operation is becoming one of the keys in RISCs, DSPs, graphics accelerators and so on, because of increasing demand from multimedia applications. Recent high-end microprocessors call for an operation frequency of 200 MHz or over, and a multiplier will be required to operate in one clock cycle. However, no 54 X 54-b multiplier with a delay time less than 5 ns has yet been reported [1][2]

This paper describes a 54 X 54-b multiplier macro developed for the mantissa multiplication of two double-precision numbers as outlined in the IEEE standard. To reduce the multiplication time, a new 4-2 compressor and a carry look-ahead adder (CLA) featuring pass-transistor multiplexers have been developed. The new circuits gain a speed advantage over conventional CMOS circuits because the number of critical-path gate stages is minimized due to the high logic functionality of pass-transistor multiplexers. The 54 X 54-b multiplier was fabricated by triple-metal 0.25-μm CMOS technology.

Architecture

The block diagram of the 54 X 54-b multiplier is shown in Fig. 1. We used Booth's algorithm, Wallace's tree and a conditional carry-selection (CCS) adder [3]. The number of partial products is halved by Booth's algorithm. Without propagation of the carry, partial products are summed by Wallace's tree. The summed results are added by CCS adder with high-speed carry propagation.

Wallace's tree used the 4-2 compressor, which has five inputs and three outputs. Carry-out (Co) is connected to the next 4-2 compressor's carry-in (Ci), as shown in Fig. 1. Without propagating the carry to the higher bit, the 4-2 compressor can add four partial products because the carry-out (Co) does not depend on the carry-in (Ci). By using the 4-2 compressor, only four addition stages are needed for Wallace's tree as shown in Fig. 1.

Circuit and Layout Design

The 4-2 compressor circuits using pass-transistor multiplexers are shown in Fig. 2. Since the pass-transistor multiplexer circuit, as shown in Fig. 3, has high logic functionality, a full adder circuit is constructed by three pass-transistor multiplexers. The 4-2 compressor is constructed of two full adders, and there are four critical-path gate stages, as shown in Fig. 2(a). This circuit is faster than the conventional CMOS circuit. For further speed improvement, we used an improved 4-2 compressor. The number of critical-path gate stages for this circuit becomes

Reprinted from *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 26.4.1-26.4.4, May 1994.

three by exploiting parallelism, as shown in Fig. 2(b). The simulated delay comparison for these 4-2 compressor circuits is shown in Fig. 4. The proposed circuit reduces the propagation delay time by 18% from that of full-adder-based circuit. The construction of Wallace's tree is shown in Fig. 5. By using the 4-2 compressor, the construction of Wallace's tree can be simplified.

The carry look-ahead adder (CLA) in the final adder also uses pass-transistor multiplexers, as shown in Fig. 6. We have already reported a new look-ahead carry scheme called conditional carry-selection (CCS) [3]. The 4-bit CLA is constructed by three multiplexers and is faster than the conventional pass-transistor-based design by avoiding series-connected pass-transistors in the carry propagation path. To apply this scheme to the final 108-bit adder, the 4-bit CLA is modified to an 8-bit CLA, as shown in Fig. 6. The new 8-bit CLA achieves four critical-path gate stages by exploiting the parallelism. It reduces the 108-bit addition time to 1.52 ns.

Fabrication

The chip was fabricated by triple-metal 0.25-μm CMOS technology. Table 1 shows the process technology. The 1st metal is tungsten, and the 2nd and 3rd metals are aluminum. It operates from supply voltage of 2.5 V. Figure 7 shows a micrograph of the chip. 100,200 transistors are integrated in the active area of 3.77 mm X 3.41mm.

Evaluation

The simulated multiplication time of the 54 X 54-b multiplier is shown in Fig. 8. The multiplication time is 4.4 ns with a 2.5 V power supply. It shows excellent characteristics at such a low voltage because of the pass-transistor-multiplexer-based design where both NMOS and PMOS are turned on. The Characteristics of this 54 X 54-b multiplier test chip are summarized in Table 2. The measured waveforms of Wallace's tree are shown in Fig. 9. The measured delay was almost the same as that of the simulated value.

Figure 10 shows the multiplication time plotted against the device dimensions. The multiplication time with the full-adder-based circuit is estimated to be 5.1 ns. Therefore, this multiplier achieves 14% improvement in multiplication time due to the use of the new circuits with pass-transistor multiplexers.

Conclusions

A new 4-2 compressor and CLA using pass-transistor multiplexers have been developed to shorten multiplication time. The multiplication time of the 54 X 54-b multiplier is reduced by 14% due to the reduction of the critical-path gate stages by using pass-transistor multiplexers. A 4.4-ns multiplication time was achieved with 0.25-μm CMOS technology.

Acknowledgments

The authors wish to thank Dr. K. Shimohigashi, Dr. T. Nishimukai, Dr. E. Takeda, and Dr. T. Nagano for their useful discussions, and K. Ueda and K. Takasugi for their assistance and support. The authors are also greatly indebted to T. Nishida, A. Fukami, N. Ohki, and H. Ishida for their assistance with the process technology and device fabrication.

References

[1] G. Goto *et al.*, "A 54 X 54-b regularly structured tree multiplier," *IEEE J. Solid-State Circuits*, vol. 27, pp. 1229-1236, September 1992.

[2] J. Mori *et al.*, "A 10-ns 54 X 54-b parallel structured full array multiplier with 0.5-μm CMOS technology," *IEEE J. Solid-State Circuits*, vol. 26, pp. 600-606, April 1991.

[3] M. Suzuki *et al.*, "A 1.5ns 32b CMOS ALU in double pass-transistor logic," *in 1993 ISSCC Dig. Tech. Papers*, pp. 90-91, February 1993.

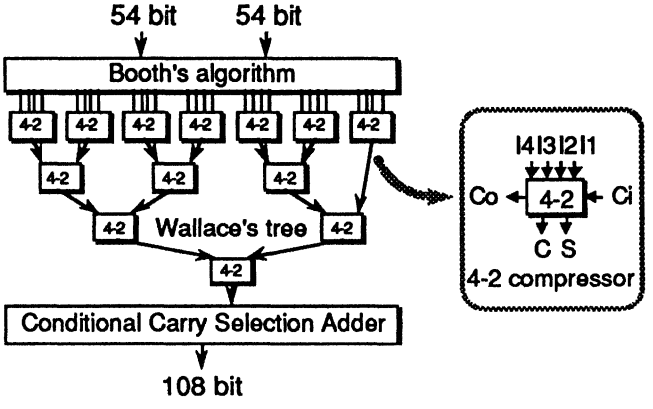


Fig. 1. Block diagram of the 54 X 54-b multiplier using pass-transistor multiplexer.

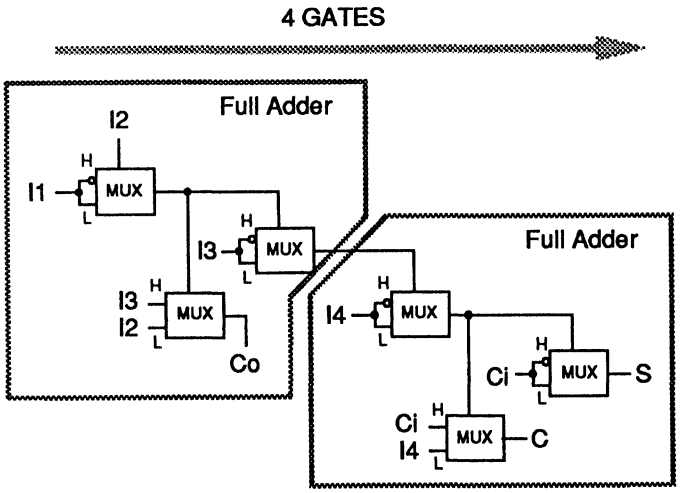


Fig. 2. 4-2 compressor circuits using pass-transistor multiplexer: (a) full-adder-based construction (b) proposed construction.

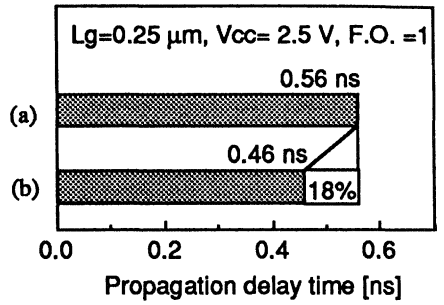


Fig. 4. Simulated comparison of 4-2 compressor circuits: (a) full-adder-based construction (b) proposed construction.

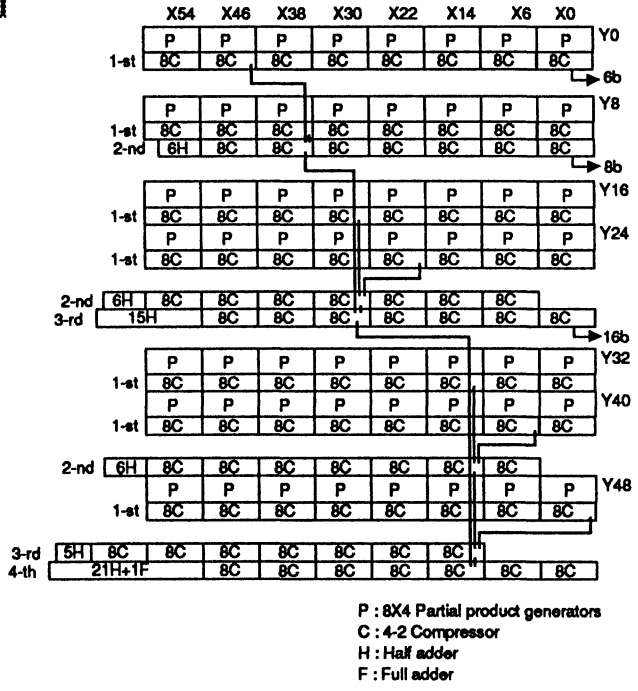


Fig. 5. Construction of Wallace's tree.

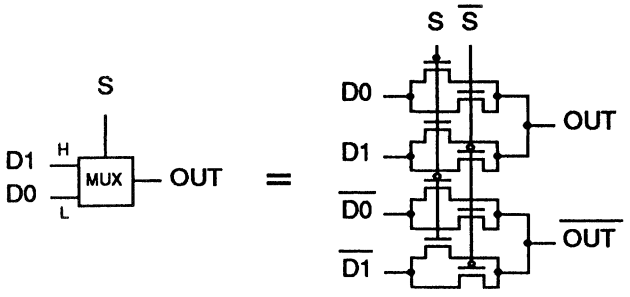


Fig. 3. Pass-transistor multiplexer circuit.

Table 1. Process technology

Technology	0.25-μm CMOS Triple metal
Gate length	0.25 μm
Gate oxide	6.5 nm
1st Metal Width/Space	0.5 μm / 0.4 μm
2nd Metal Width/Space	0.5 μm / 0.4 μm
3rd Metal Width/Space	0.7 μm / 0.6 μm

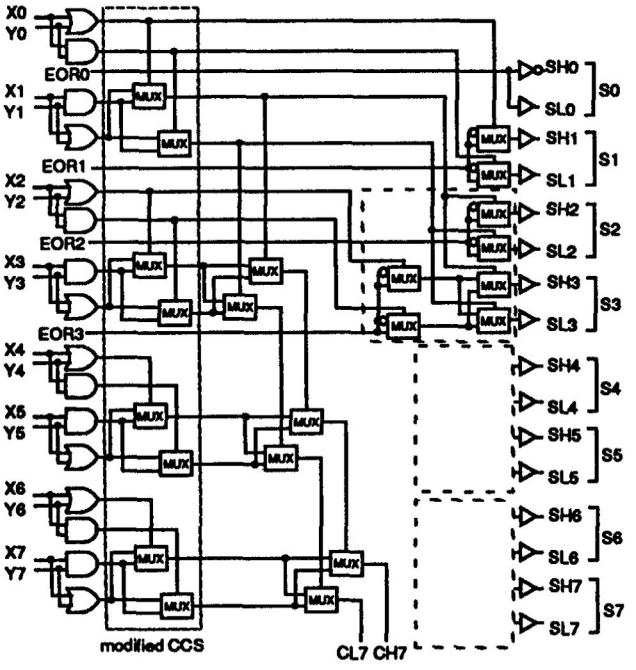


Fig. 6. 8-bit CLA using pass-transistor multiplexer.

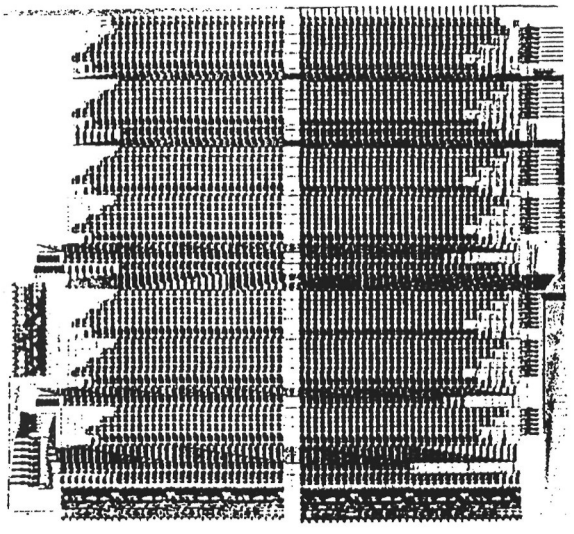


Fig. 7. Micrograph of the 54X54-b multiplier.

Table 2. Characteristics of the 54X54-b multiplier

Organization	54X54-b multiplier
Multiplication time	4.4 ns
Active Area	3.77 X 3.41 mm
Transistors	100200

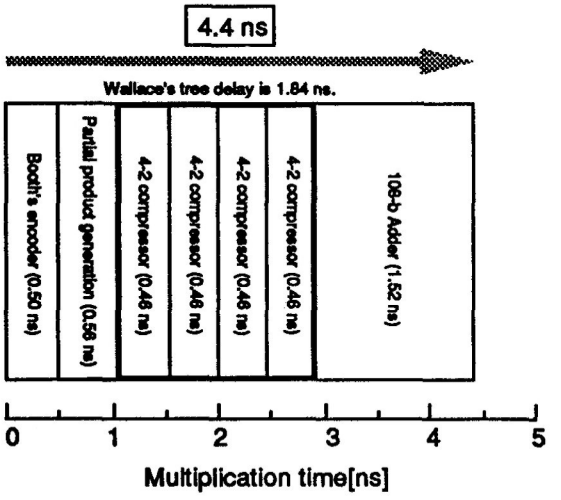


Fig. 8. Multiplication time of 54X54-b multiplier.

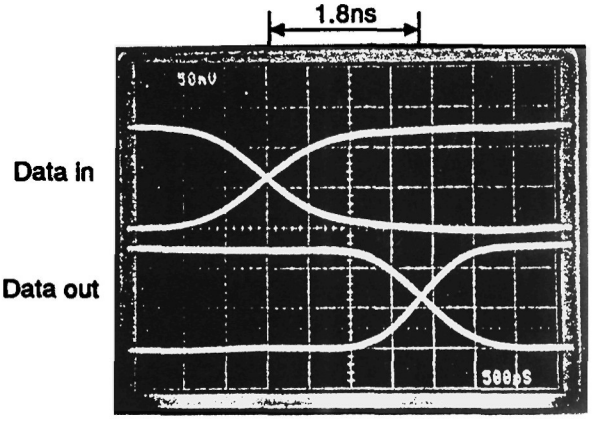


Fig. 9. Measured waveforms of the Wallace's tree.

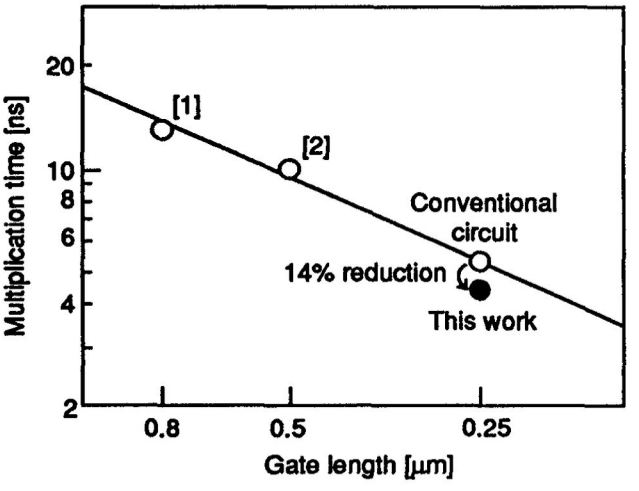


Fig. 10. 54X54-b multiplication time versus device dimension.

Pass-Transistor Dual Value Logic for Low-Power CMOS

Vojin G. Oklobdzija, B. Duchêne*
Advanced Computer System Engineering Laboratory,
Electrical and Computer Engineering Department
University of California Davis
(916) 752-5634
vojin@ece.ucdavis.edu
*Ecole Superieure d'Ingenieurs en Electrotechnique et Electronique
93162 Noisy le Grand CEDEX FRANCE

ABSTRACT

This paper presents new pass-transistor logic termed DVL which contains fewer transistors than its counterpart DPL yet maintaing comparable performance. A method for synthesis of such networks is also developed and demonstrated in this paper. The new logic is characterized by good speed and low power. The simulations and tests were performed using 1-μm CMOS.

I. INTRODUCTION

New logic CMOS families using pass-transistor circuit techniques have recently been proposed with the objective of improving speed and power consumption [1-6]. This logic (in most cases) passes the charge between the nodes rather than charging the nodes from V_{CC} and then discharging them to GND. This feature contributes to less power being used as compared to the regular CMOS. The Double Pass-Transistor Logic (DPL), developed by Hitachi demonstrated an 1.5nS 32-bit ALU and 4.4nS 54-bit multiplier in 0.25 μm technology [4,5]. However, DPL has not yet been fully adopted because of its high transistor count. The objective of the new logic gates and the synthesis method developed for pass-transistor logic is to minimize the number of transistors used in DPL and preserve the speed of the logic.

II. NEW LOGIC GATES

The new logic gate represents an improvement over DPL family achieved by the elimination of the redundant branches and rearrangement of signals. This simplification, illustrated in Fig. 1, 2 and 3, preserves the advantages of DPL gates which are:

a) Compensation of speed degradation due to the use of pMOS transistors.

b) Straightforward full swing operation.

This simplification is achieved by in three steps:

A. *Elimination of the redundant branches*

This simplification is achieved by eliminating the redunant branches (shown in shaded area) from DPL. Most of the pull up and pull down transition times, in the resulting configuration, surpass those of the DPL gates. However, the improved gate has some undesirable input configurations in which the current path is supplied by a single transistor instead of a double pass-transistor (as in the case in DPL), making this transition time worse. To avoid degradation of delay due to the use of just one pMOS transistor, the particular transistor width is increased. The elimination of redundant branches is illustrated in Fig. 1. The resulting two halves (which constitute the gate) are not of the same speed. The faster half is NAND (60pS) and the slower is AND (70pS), which is still being faster than DPL (75pS).

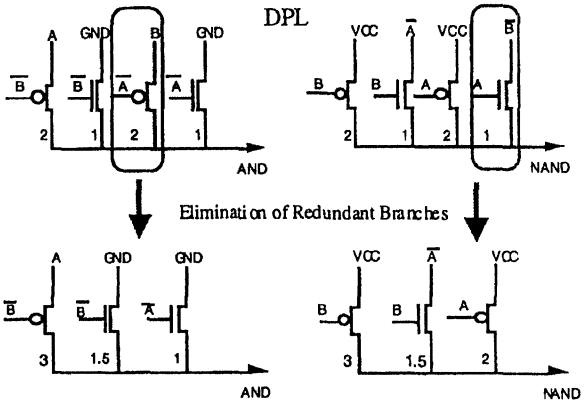


Fig. 1. Elimination of redundant branches

B. *SiMrrangement*

The use of two parallel pMOS transistors is avoided by simple signal re-arrangement because the two pMOS transistors contribute more to the delay than one pMOS transistor in parallel with one nMOS transistor. This is especially true in a pull up operation. The current is always provided by one nMOS transistor alone or by one nMOS and one pMOS in parallel. The AND/NOR DPL gate (in Fig. 2.) is obtained from NOR/OR DPL configuration whose inputs are simply inverted. Signal rearrangement applied to AND/NOR DPL gate results

Reprinted from *Proceedings of the 1995 International Symposium on VLSI Technology*, pp. 341-344, May/June 1995.

in an AND gate configuration which is faster than DPL (60pS vs. 75pS), where AND is a faster half.

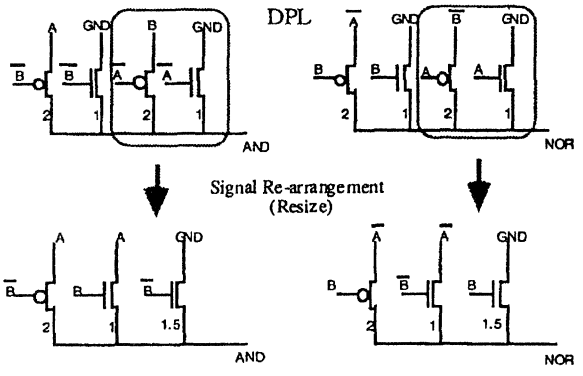


Fig.2. Signal Re-arrangement

C. *Selection of the faster halves*

Finally we take a faster half from Fig.1. and form Fig.2. The resulting AND/NAND complementary logic gate (shown in Fig.3.) is obtained by elimination of the redundant branches for the NAND and rearrangement of signals for the AND gates respectively. We named this logic: DVL (Dual Value Logic). The resulting AND/NAND DVL gate contains a total of 6 transistors as compared to DPL consisting of 4 transistors of each type. There is a total of 9 inputs in DVL versus 12 in DPL resulting in a smaller capacitive load of DVL gate. In DVL 3 inputs are connected to the transistor source and 6 to the gate (3 to p-type and 3 to n-type). In DPL 4 inputs are connected to the source and 8 to the gate (4 to p-type and 4 to n-type transistors).

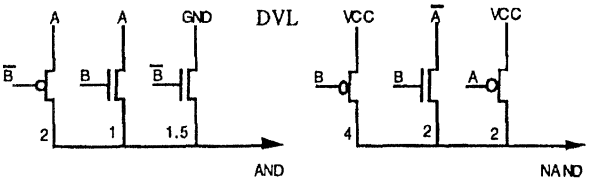


Fig. 3. Resulting DVL Gate

The comparison between NAND/AND DPL gate and DVL shows:

- *20% speed improvement, using 75% of transistors used in DPL gate.*
- *25% less connections and wires than in DPL gate. The 4% area penalty comparison to DPL is quite negligible.*

Similar arguments can be used to build the NOR/OR gates from DPL gates.

III. SYNTHESIS METHOD

The synthesis method for DVL is based on the method used to create the logic gates described before. At present, there is no known algorithm to find minimal multi-stage logic circuits. The new proposed method for synthesis of DVL, is based on transistors instead of logic gates. In place of conjointly assembling several basic gates, functions are synthesized at the transistor level. In addition, the programming of this method has been developed to prove the efficiency of the theory presented.

The key point of DVL synthesis consists of employing Karnaugh-Map at the transistor level. Thus, we are not cascading several logic gate levels (NAND/AND or NOR/OR) but building functions by directly using several transistor levels in series. However, the choice of *pseudo Karnaugh-Map* in the programming for less than 8 inputs was done because its explanation is simple, but a *pseudo Quine McCluskey* technique could have been adopted instead.

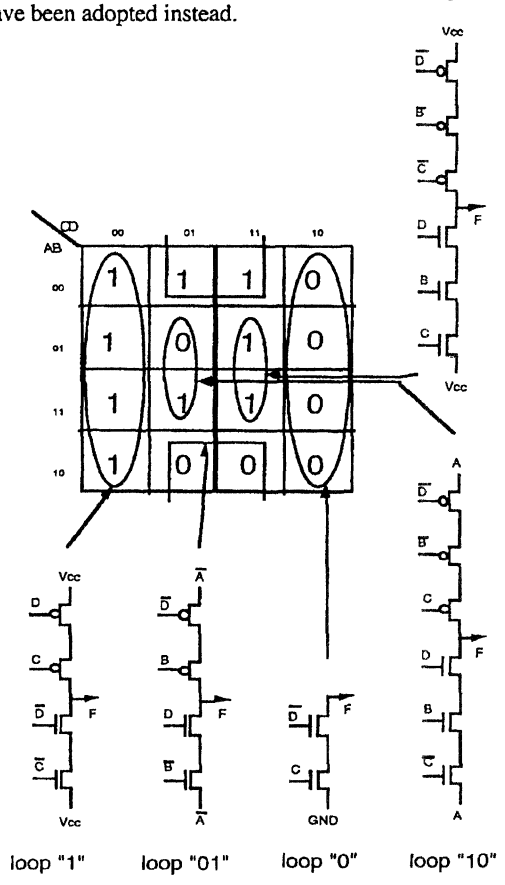


Fig. 4. Loops allowed for 4 inputs

Usually, the general Karnaugh-Map is covered by loops of "0" or "1", in such a way that a minimized Sum of Products form is obtained. In our case, four classes of loops are allowed (as illustrated in Fig. 4) to directly synthesize a part of the final circuit. Accumulating all loops necessary to cover the Karnaugh-Map yields the resulting circuit.

IV. RESULTS

The best way to compare efficiency of the presented algorithm is via synthesizing and simulating circuits obtained using our automated algorithm, and comparing it to circuits produced by CPL and DPL using the concept of logic gates.

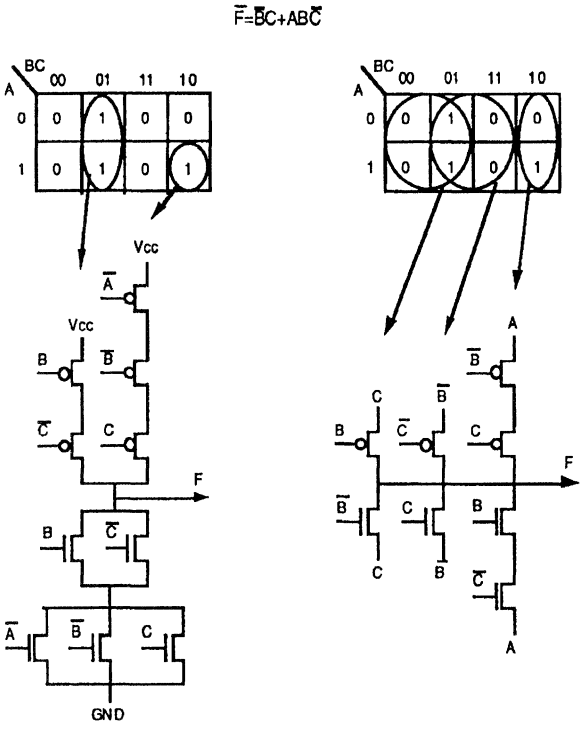


Fig. 5. Example showing implementation of the function F in DVL and conventinal CMOS

The DVL synthesis was compared not only to the Conventional CMOS, but also to DPL circuits [4] and CPL circuits using lean cells [6]. An example of DVL synthesis versus Conventional CMOS synthesis (given in Fig. 5.) shows the improvement in global size, the number of transistors and the delay of the circuit. In each circuit, the global size of DVL is smaller compared to other circuits.

The comparison with AND/NAND DPL gate shows 25% less transistors resulting in 25% less connections and wires in an equivalent DVL gate, keeping the total transistor area constant. Similar methods can be used to build the NOR/OR gates.

A. *Comparison with CMOS*

A comparison between DVL and conventional CMOS is given in Table I, while the simulation results (for the given function $F2 = \overline{\overline{B}C + AB\overline{C}}$) are shown in Fig. 6.

TABLE I.
COMPARISON BETWEEN DVL AND CONVENTIONAL CMOS

Function F2	CMOS	DVL	Savings
No. of Transistors	10nMOS 10pMOS	8pMOS 8nMOS	20%
No. of Levels	3 gate levels	2 transistor levels	
Global size	44	36	18%
Delay @ 50% of Vout	430pS	245pS	43%
Transitor ratio	Wp/Wn= 2	Wp/Wn=2	

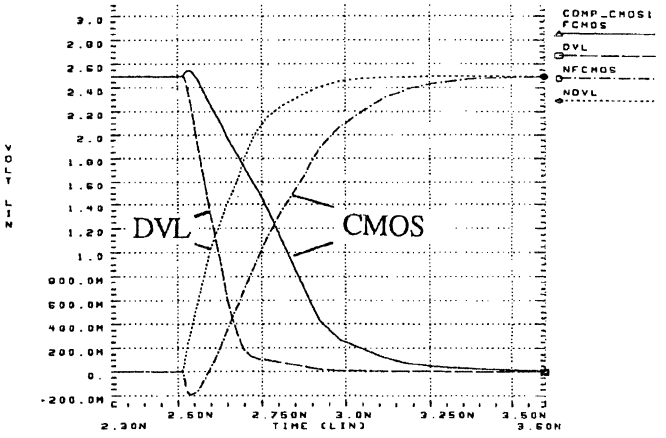


Fig. 6. Delay Comparison between DVL and Conventional CMOS for a 3 inputs function F2

B. *Comparison with DPL*

The function used for comparison is a three variable function $F_2(A,B,C)$, where $F2 = \overline{\overline{B}C + AB\overline{C}}$ and $\overline{F2} = \overline{B}C + AB\overline{C}$. This function was implemented using 4 DPL gates in two logic levels. Afterwards this circuit was built using DVL. The load applied to the output is a standard load of two gate inputs. The comparison results are shown in Table II and timing simulation for the function F_2 are shown in Fig. 7.

TABLE II.
COMPARISON BETWEEN DPL AND DVL FOR A 3 INPUTS FUNCTION F2

Function F2	DPL	DVL	Savings
No. of Transistors	16pMOS 16nMOS	8pMOS 8nMOS	50%
No. of Levels	2 gate levels	2 transistor levels	
Global size	48	30	37.5%
Delay @50%	290pS	120pS	58.6%
@80%	350pS	240pS	31.4%
Transistor ratio	Wp/Wn=2	Wp/Wn=2 2 nMOS=1.5	

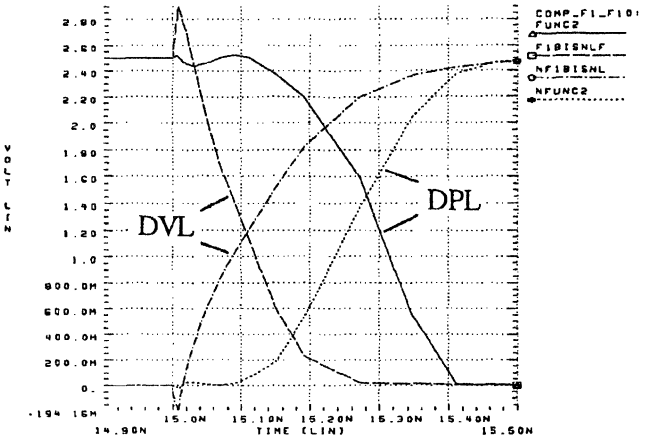
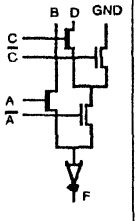


Fig. 7. Simulated delays for the 3 inputs DPL function F2 and DVL implementation of F2

C. *Comparison with CPL*

The function F published by Yano in [6] was synthesized for DVL and compared to CPL which uses lean cells and special inverters [6]. Delays were measured for 1 cell, 2 cells and 3 cells cascaded. DVL circuit is made with conventional inverters. The comparisons were made using the output load of 15 FF in both cases.

TABLE III.
COMPARISON BETWEEN CPL AND DVL FOR A 4 INPUTS FUNCTION.

	1 cell	2 cells	3 cells	Cell size
	375pS (100%)	760pS (100%)	1,150p S (100%)	105µ (100%)
Circuit F in DVL	380pS (101%)	660pS (86%)	950pS (82%)	108µ (103%)

The comparison between CPL and DVL is shown in Table III and the delays of the two cascaded CPL and DVL cells are compared in Fig. 8.

CONCLUSION

DVL logic family has been developed which has advantages over standard CMOS as well as new pass-transistor families such as DPL and CPL. However, the exact speed improvement is dependent on each particular circuit. The power consumption is also reduced for 30-50% over conventional CMOS. Generation of DVL is supported by an automated synthesis tool based on the algorithm developed in the course of this work.

ACKNOWLEDGMENT

The authors acknowledge the contributions of P. Lee, W. Wong and H. Yu.

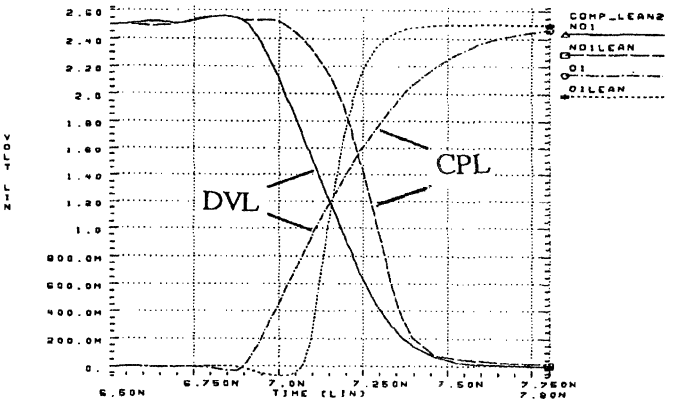


Fig. 8. Delays comparison between 2 cascaded DVL and CPL cell.

REFERENCES

[1] F.S. Lai and W. Hang, "Differential Cascode Voltage Switch with Pass Gate Logic Tree for High Performance CMOS Digital Systems", *1993 International Symposium on VLSI Technology, Systems and Applications*, pp358-362, May 1993.

[2] Yano, K, et al, "A 3.8 ns CMOS 16X16-b Multiplier Using Complementary Pass-Transistor Logic", *IEEE J. Solid State Circuits*, vol 25, p388-395, April 1990.

[3] Akilesh Parameswar, et al, "A Swing Restored Pass-Transistor Logic Based Multiply and Accumulate Circuit for Multimedia Applications", *Proceedings of the IEEE 1994 Custom Integrated Circuit Conference, San Diego, California, May 1-4, 1994.*

[4] Makoto Suzuki, et al, "A 1.5 ns 32 b CMOS ALU in Double Pass-Transistor Logic", *1993 ISSCC Dig. Tech. Papers*, pp90-91, February 1993.

[5] Ohkubo, N., et al, "A 4.4nS CMOS 54x54-b Multiplier Using Pass-Transistor Multiplexer",*Proceedings of the IEEE 1994 Custom Integrated Circuit Conference, San Diego, California, May 1-4, 1994.*

[6] Yano, K, et al, "Lean Integration : Achieving a Quantum Leap in Performance and Cost of Logic LSIs", *Proceedings of the IEEE 1994 Custom Integrated Circuit Conference, San Diego, California, May 1-4, 1994.*

Differential Cascode Voltage Switch with the Pass-Gate (DCVSPG) Logic Tree for High Performance CMOS Digital Systems

F.S. LAI AND W. HWANG
IBM T.J. WATSON RESEARCH CENTER
P.O. BOX 218, YORKTOWN HEIGHTS, NEW YORK 10598

Abstract

A new circuit configuration, the differential cascode voltage switch with the pass–gate logic tree (DCVSPG), is presented. In this circuit family, we use the pass–gate logic tree to replace the nMOS logic tree in the conventional DCVS circuit in order to eliminate the floating–node problem. By eliminating the floating–node, the DCVSPG shows superior performance, silicon area and power consumption. Moreover, the dynamic DCVSPG also provides the leverage of relieving the charge redistribution concern and reinforces the signal integrity in the typical pre–charge dynamic circuits.

The principle of operation of the DCVSPG is explained. A simple synthesis technique of the pass–gate logic tree is discussed. Finally, a 64–bit carry look–ahead adder is designed by using the static DCVSPG circuit. A nominal cycle time (T_a = 22°C and power supply of 2.5 V) of 2.0 ns is obtained by using a 0.5μm CMOS technology.

I. Introduction

The conventional cascode voltage switch (DCVS) is claimed to have advantages over the traditional static CMOS NAND/NOR design in terms of circuit delay, layout area, logic flexibility and power dissipation [1,2]. For the dynamic implementation, DCVS also shows the superior logic implementation flexibility over the standard domino logic which is suffered from the lacking of inverting gates [3]. However, the conventional DCVS can cause floating–node in both legs of their nMOS logic tree. This floating–node causes the static DCVS to become ratioed logic which in turn creates current spikes and additional delay [4]. Although the ratioed logic problem can be solved in dynamic DCVS by using the pre–charging scheme. Unfortunately, the floating–node problem still exist in the dynamic DCVS, and it will trigger another problem such as the charge redistribution. The result of charge redistribution might develop a false logic evaluation. This makes the dynamic circuit very un–reliable. Complementary pass–transistor logic (CPL) [5] was developed to solve this floating–node problem. The loading in the CPL was chosen using static inverters instead of a cross–coupled pMOS latch in conventional DCVS to restore the signal. This results in a mismatch problem between the input signal level and the logic threshold voltage of the static CMOS inverter. It also caused the CPL to have poorer noise margin and speed degradation. Recently, the double pass–transistor logic (DPL) [6] was developed to solve the CPL problem at the expense of double transistor counts and silicon area. In this paper, a new DCVSPG circuit family is developed [1] to overcome the above mentioned drawbacks in DCVS and CPL. The regeneration problem in DCVS caused by the floating–node is solved by the pass–gate network.

In the following sections, the operation principles of static and dynamic DCVSPG circuits will be presented first. Then the comparison of sum circuits among DCVS, DCVSPG and static CMOS are discussed. Finally, the implementation of a 64–bit carry look–ahead adder by using a 0.5μm CMOS technology will be described. Conclusions will be given in the last section.

II. DCVSPG Circuit Operation Principle

In this section, the basic operation of static and dynamic DCVSPG will be presented. The synthesis of pass–gate logic tree will also be described.

(i) Static Circuit

Fig. 1 shows the typical static DCVSPG to evaluate the AND function of $q = ab$. The an and bn are the complementary signals of input variables a and b respectively. Initially, we assume both of a and b signals are low, the N2 and N4 transistors all turn OFF. However, the an and bn signals are all high which in turn switch the N1 and N3 transistors ON. It shows that node q is low and node qn is high. This leads to the cross–coupled pMOS transistor P1 is ON and P2 is OFF. When both of a and b signals swing from low to high, the node q is instantly charged up to high through the N4 transistor. This makes the P1 transistor turn OFF while the node qn is discharging through N2 transistor. This is great contrast to the conventional DCVS circuit shown in Fig. 2. In the transition period, the node q kept low momentarily which let the P1 transistor ON while the node qn is discharging in the conventional DCVS circuit. This floating–node problem causes the conventional DCVS to have larger power consumption and propagation delay. Besides that, for the DCVSPG logic, the pMOS is only used as a load to bring up the full–swing signal, it is not the critical device in the pull–up operation. Therefore, the device size can be small. In the conventional DCVS, however, pMOS has to be twice as large as the nMOS device in order to get a comparable pull–up operation. This leads to a larger silicon area consumption.

Figs. 3 and 4 show the ASTAP simulation results of logic function AND of $q = ab$ for the conventional DCVS and DCVSPG with the function of the pMOS device width. In those simulations, we set the nMOS device width constant (W_n = 20 μm). It is very interesting to note that the rise time is a very strong function of the pMOS width for the DCVS circuit. It is obvious that the rise time decreases when pMOS device width increases. However, the rise time increases again when the pMOS device width increases beyond 10–20 μm. This indicates that the conventional DCVS has the ratioed circuit problem. An optimum pMOS device has to be carefully chosen. On the other hand, the rise time of DCVSPG is almost constant due to the pull–up behavior is mainly done by the nMOS instead of the pMOS. The overall performance of the DCVSPG is much superior to that of the DCVS at any pMOS width and loading.

(ii) Dynamic Circuit

Fig. 5 shows the AND logic function implementation of $q=ab$ for conventional dynamic DCVS and Fig. 6 for the dynamic DCVSPG. When clock signal ϕ is low, both of q and qn nodes are charged up to V_{dd} high level signal for both of these circuits. Both of circuits start to evaluate the logic function when ϕ is high. For the dynamic DCVS, the node qn is starting to discharge when signal a and b swing from low level 0 to high level V_{dd}. The stored charge in node qn will flow through the transistors N3, N4 and N5 to ground. Node q is presumably staying in V_{dd} voltage level due to the transistors N1 and N2 turn OFF. However, for the dynamic DCVSPG, the node qn is discharged through transistors N2 and N6 to node bn which is in the ground state. The node q is charged up through transistors N1 and N4 by node b which is in the V_{dd} state.

There are several distinct characteristics in DCVSPG in comparison with the conventional DCVS circuit. The advantages are no floating–node generation in both of logic tree legs, symmetrical logic topol-

Reprinted from *Proceedings of the 1993 International Symposium on VLSI Technology*, pp. 358-362, June 1995.

ogy and shorter logic stack height. Elimination of the floating–node prevents the static circuit from suffering the ratioed logic problem and the dynamic circuit from the charge redistribution problem. The symmetrical logic topology in the logic tree and the shorter logic stack height improve the circuit performance and power consumption [3].

(iii) Synthesis of Pass–Gate Logic Tree

The synthesis of conventional n–channel logic tree for DCVS had been discussed by using the modified Karnaugh map or the modified Quine–McCluskey tabular method [4]. The synthesis of pass–gate logic had also been explained [7]. However, the synthesis algorithm they showed is either not quite clear or too simple. Let us show a synthesis procedure by using a recursive minimization with the Karnaugh map.

In order to synthesis the logic function $F = \bar{a}\,\bar{b}\,\bar{c}\,\bar{d} + a(b + c + d)$, where $\bar{a}\,\bar{b}\,\bar{c}\,\bar{d}$ are *an, bn, cn, dn* in our figures, into the pass–gate logic, the Karnaugh map result is shown in Fig. 7. The *ab* is assumed to be the control variables and *cd* is the input function variables. By grouping the same output function pattern together, the pass–gate logic can be minimized as $F = \bar{a}\,\bar{b}\,(x_1) + b(x_2) + a(x_2)$ as shown in Fig. 7. The x_1 pattern is [1010] and x_2 pattern is [0011]. These two patterns can be continuously minimized as $x_1 = c(d) + \bar{c}(\bar{d})$ and $x_2 = c(1) + \bar{c}(0)$, where 1 is the V_{dd} state and 0 is the ground state. The final pass–gate logic function is then $F = \bar{a}\,\bar{b}\,[c(d) + \bar{c}(\bar{d})] + b[c(1) + \bar{c}(0)] + a[c(1) + \bar{c}(0)]$. The DCVSPG circuit is shown in Fig. 7(b). This circuit is much simpler and faster than the pure static CMOS implementation.

III. Comparison of The sum Circuits

In order to compare the circuit performance of various design techniques, the sum circuit of the full adder is being simulated using ASTAP. Fig. 8 shows the static and dynamic versions of the conventional DCVS design technique. The static and dynamic implementation of the DCVSPG is shown in Fig. 9. The static CMOS design of the sum circuit is also shown in Fig. 10.

The device width is designed following a basic rule that the conductance of all the dischargirg path are assumed to be the same as a conductance of a minimum size ($W_n = 3\mu m$) nMOS transistor. For example, in the conventional DCVS static circuit shown in Fig. 8(a), three transistors are seriesly connected along the discharge path. The transistor width is then chosen as $3\ \mu m \times 3 = 9\ \mu m$. For 8(b), however, the device size is then increased up to $3\ \mu m \times 4 = 12\ \mu m$ in the dynamic DCVS configuration. The pMOS device size is chosen as twice larger than that of the nMOS device.

The overall simulation results are shown in Table I. The load output capacitance is assumed that the circuit drives a chain of the similar circuits. With a fan–out of one for the static circuit, it is obvious that the output capacitance is the same with the input capacitance. The dynamic circuits, however, are buffered by the C^2MOS latch and would be expected to be able to drive larger loads than that of the static gate. A fan–out of two was then used for the dynamic designs.

Considering, first, the static design, it appears that the static DCVSPG yields the best power–delay product. The DCVSPG has the lowest logic tree stack height such that its transistor size and input capacitance are the smallest. And yet its best performance is solely due to the pull–up and pull–down are all done by the high–performance nMOS transistor. The lowest power consumption of the static DCVSPG is also due to the very symmetrical charging and discharging times of nodes *q* and *qn* in Fig. 9(a). The asymmetrical charging and discharging periods, however, of the conventional static DCVS causes a prolong transient time when the transistor switches and thus dissipates more power. The DCVS and DCVSPG have the area advantages over the conventional static CMOS circuit due to the redundant pMOS transistors are reduced dramatically in the DCVS configuration.

For the dynamic circuits, it is interesting to note that the power–delay product of the dynamic DCVSPG is the same with the static counterpart in the sum circuit. Of course, its speed is almost twice faster than that of the static DCVSPG at the expense of the larger device count and silicon area.

IV. Static 64–bit Adder Architecture

The whole adder core is shown in Fig. 11. This architecture is implemented by the binary carry look–ahead algorithm [8,9]. There are total 12 rows shown here. The first row is the PG circuit to generate the p (propagation) and g (generate) signals. The last row is the sum circuit. There are total 10 rows to generate the carry signal. Inside the 10 rows of carry chain, the white rectangular and triangular are the buffer circuit and driver circuit respectively. The black rectangular is the merge circuit. Some of white rectangular cell can also route the signal from top to its left to feed into the black merge circuits. The sum bit 0 comes from the right hand side.

From the carry look–ahead theory, the sum bit can be written as

$$s_i = a_i \oplus b_i \oplus c_{i-1} \qquad (1)$$

The generation and propagation bits are defined as

$$g_i = a_i b_i \qquad (2)$$

$$p_i = a_i \oplus b_i \qquad (3)$$

The merge bit can be explained as

$$G_i = g_i + G_{i-1} P_i \qquad (4)$$

$$P_i = p_i P_{i-1} \qquad (5)$$

According to Fig. 11, the merge circuit takes the signals from the top cell and the right–hand signal and outputs signals into the bottom cell. so the g_i and p_i are the signals come from the top cell. However, the G_{i-1} and P_{i-1} are the signals from the right–hand white rectangular cell. By assuming $G_{-1} = c_{-1}$ and $P_{-1} = 0$, we can easily demonstrate that the G_i signal is actually the carry signal c_i with the following definition

$$c_i = g_i + p_i c_{i-1} \qquad (6)$$

With the definition of equations (2) and (3), the PG circuit is shown in Fig. 12. The merge circuit of equations (4) and (5) is shown in Fig. 13. At this circuit, $G_{i-1}n$ is the complementary signal of G_{i-1}. The sum circuit of equation (1) is shown in Fig. 14.

The ASTAP simulation results are shown in Fig. 15 by using a 0.5 μm CMOS technology. All the results are simulated at the nominal condition with $T_a = 22^{\circ}C$ and power supply of 2.5 V. The propagation delay is around 2 ns. The total rows of Fig. 11 in actual ASTAP simulation is 15 stages. This includes 1 driver stage to drive a 0.3 pF capacitive load. This driver stage costs roughly 150 ps delay. Fig. 16 shows the 64–bit adder circuit performance in the function of power supply voltage.

V. Conclusions

The DCVSPG circuit family has been developed. It is shown to have superior performance to that of a conventional DCVS approach. By using the pass–gate logic tree instead of the conventional nMOS logic tree, the floating–node problem is eliminated. This leads to no ratioed logic problem, symmetrical logic topology and shorter logic stack height. A 2 ns 64–bit CMOS adder is achieved by using the static DCVSPG circuit family.

VI. References

[1] F. S. Lai and W. Hwang, to be published
[2] L. G. Heller et. al., Dig. of ISSCC, pp. 16–17, 1984.
[3] L. M. Chu and D. I. Pulfrey, IEEE JSSC, pp. 528–532, 1987.
[4] L. C. Pfennings et. al., IEEE JSSC, pp. 1050–1055, 1985.
[5] K. Yano et. al., IEEE JSSC, pp. 388–395, 1990.
[6] M. Suzuki et. al., Dig. of ISSCC , Paper 5.4, 1993.
[7] D. Radhakrishnan et. al., IEEE JSSC, pp. 531–536, 1985.
[8] R. P. Brent and H. T, Kung, IEEE Comp., pp. 260–264, 1982.
[9] B.W. Wei et. al., Rep. UCB 86/252, UC Berkeley, 1985.

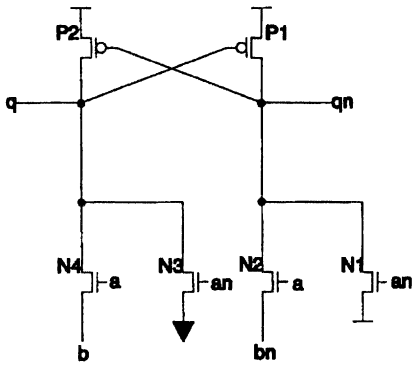


Figure 1. Static DCVSPG

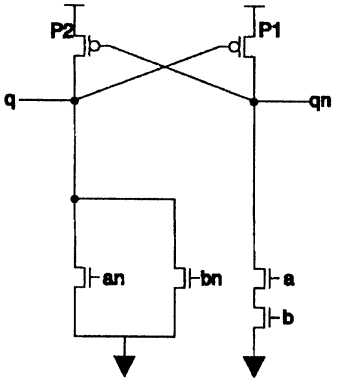


Figure 2. Static DCVS

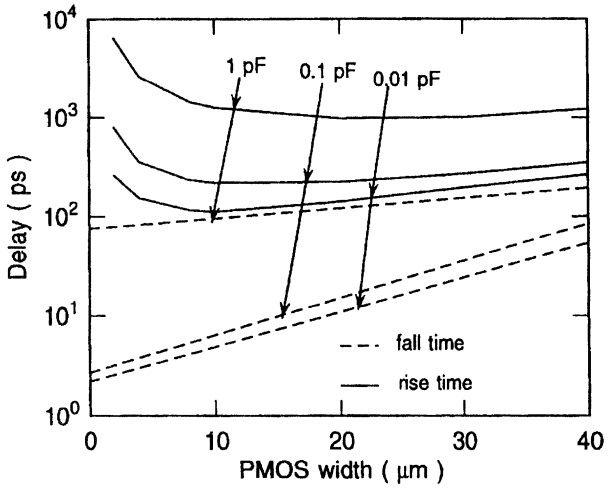


Figure 3. ASTAP simulation results of static DCVS

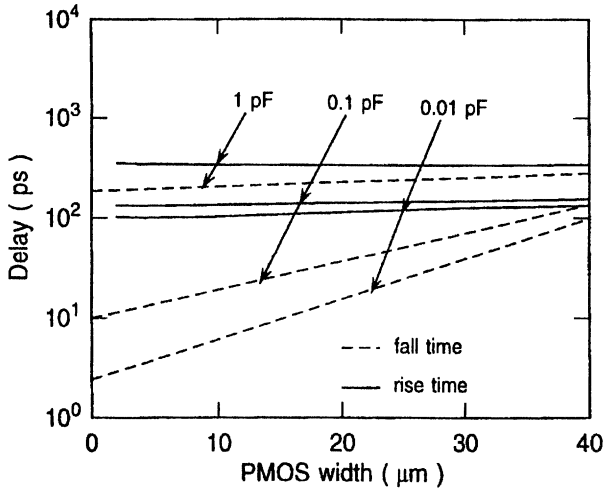


Figure 4. ASTAP simulation results of static DCVSPG

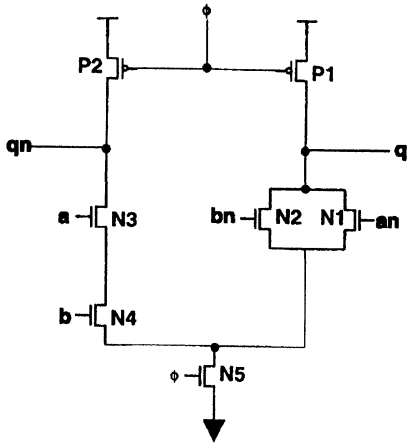


Figure 5. Dynamic DCVS

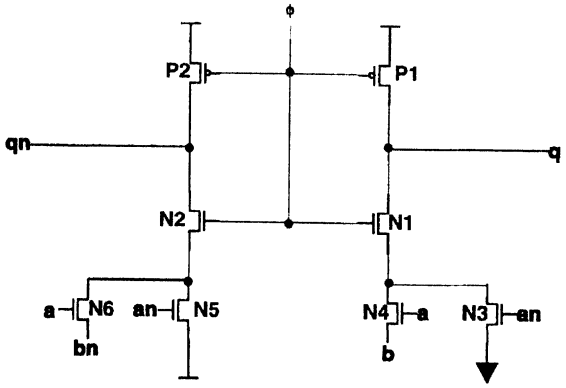


Figure 6. Dynamic DCVSPG

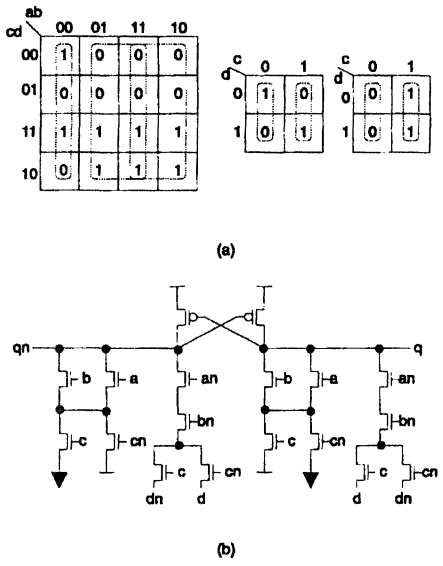


Figure 7. Synthesis of pass–gate logic

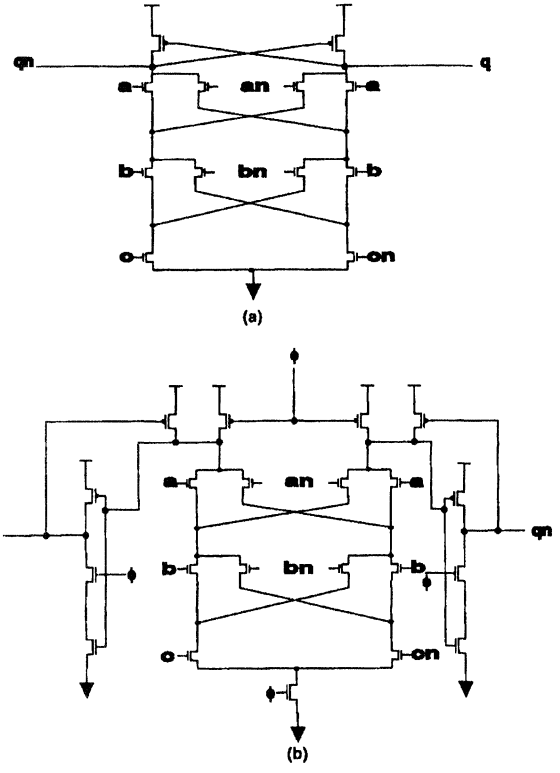


Figure 8. (a) Static DCVS (b) Dynamic DCVS

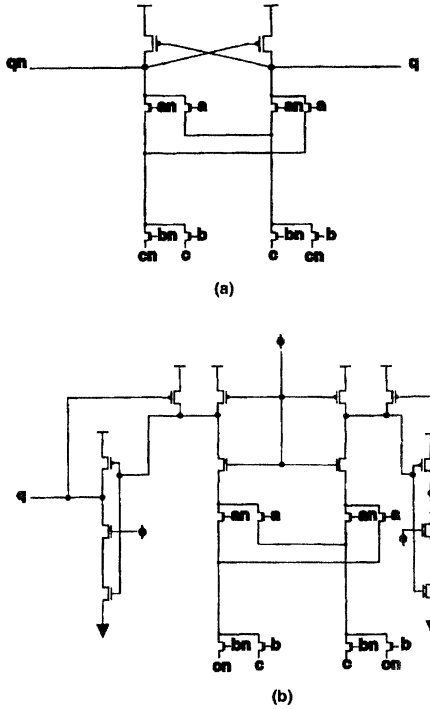


Figure 9. (a) Static DCVSPG (b) Dynamic DCVSPG

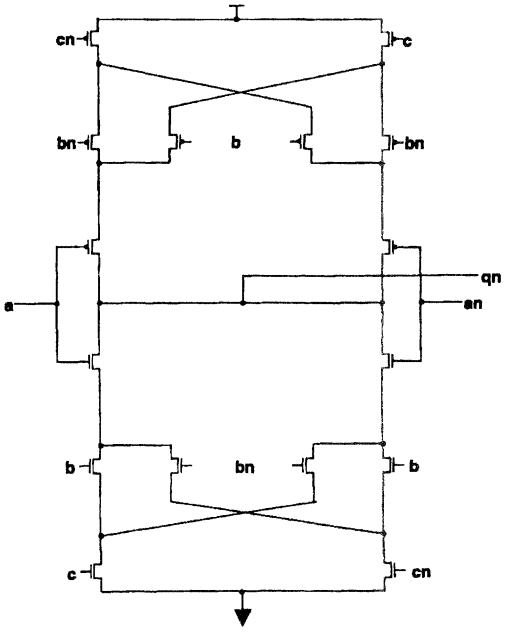


Figure 10. Static CMOS circuit

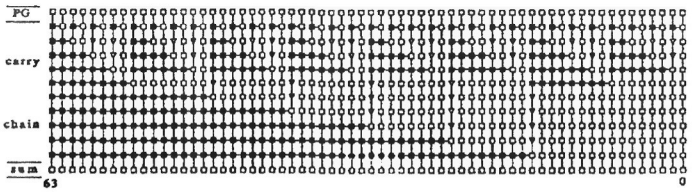


Figure 11. Adder core with the detailed construction

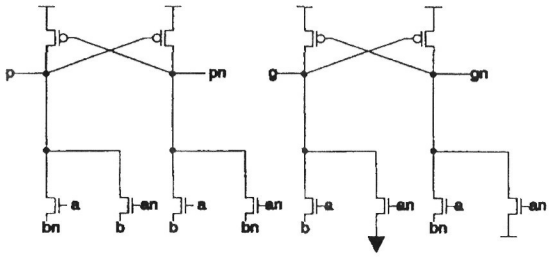


Figure 12. PG circuit implemented with DCVSPG logic family

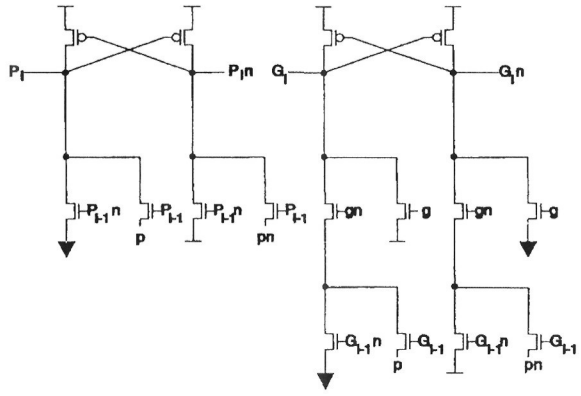


Figure 13. Merge circuit implemented with DCVSPG circuit

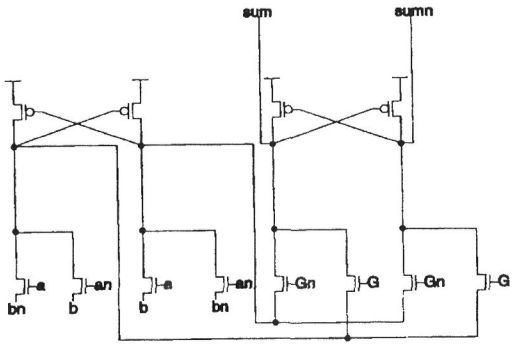


Figure 14. Sum circuit implemented with DCVSPG circuit

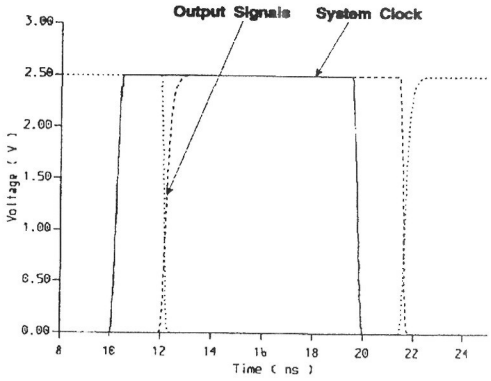


Figure 15. ASTAP simulation results

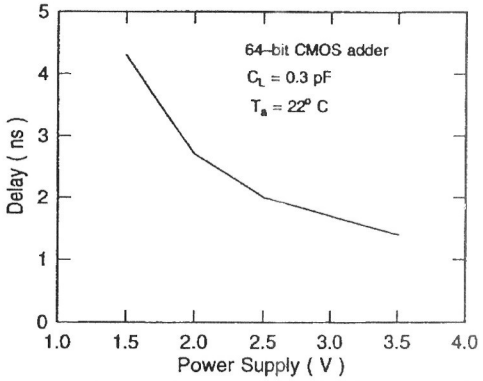


Figure 16. Circuit performance in function of power supply

Table I. Comparison of the sum circuit

	Gate Input Capacitance (fF)	Load Output Capacitance (fF)	P / N	Normalized Area	Delay (ps)	Power (μW)	Normalized Power-Delay Product
Static CMOS	108	108	8/8	1.00	327	117	1.00
Static DCVS	36	36	2/10	0.82	336	189	1.66
Static DCVSPG	24	24	2/8	0.51	210	48	0.26
Dynamic DCVS	48	96	6/15	1.37	145	118	0.44
Dynamic DCVSPG	36	72	6/14	0.86	122	80	0.26

A High Speed, Low Power, Swing Restored Pass-Transistor Logic Based Multiply and Accumulate Circuit for Multimedia Applications

AKILESH PARAMESWAR, HIROYUKI HARA, TAKAYASU SAKURAI
TOSHIBA CORPORATION
1, KOMUKAI TOSHIBA-CHO, SAIWAI-KU, KAWASAKI, JAPAN 210

Abstract

Swing Restored Pass-transistor Logic (SRPL), a high speed, low power logic circuit technique for VLSI applications is described. By the use of a pass-transistor network to perform logic evaluation, and a latch type swing restoring circuit to drive gate outputs, this technique renders highly competitive circuit performance. An SRPL based Multiply and Accumulate Circuit for multimedia applications is implemented in double metal 0.4μm CMOS technology.

Introduction

To date, the most widely used VLSI circuit design technique has been full CMOS. It has been attractive because it makes it easy to implement reliable circuits that have excellent noise margins. However, the continuing push for higher performance systems has, in recent years, brought the disadvantages of full CMOS to the fore, and a number of researchers have proposed alternative logic techniques (1-3). The majority of these have been static techniques because dynamic logic styles still suffer from charge sharing and noise margin problems, and difficulties in design and design for testability.

Complimentary Pass-transistor Logic (CPL) (1), uses a complimentary output pass-transistor logic network to perform logic evaluation, and CMOS inverters for driving of the outputs. This arrangement however suffers from leakage current through the inverter. Double Pass-transistor Logic (DPL) (2), uses both pMOS and nMOS devices in the pass-transistor network to avoid non-full swing problems, but it has high area and high power drawbacks. As the name suggests, Differential Cascode Voltage Switch with Pass Gate (DCVSPG) (3) is the same as the cascode voltage switch logic proposed in (4), but uses a pass-transistor network for logic evaluation. This logic style suffers from degraded pull down performance when used in a long chain.

In this paper, we propose a high speed, low power logic circuit technique that attempts to overcome these problems.

Swing Restored Pass-transistor Logic

Basic Circuit

The generic Swing Restored Pass-transistor Logic (SRPL) gate consists of two main parts as shown in Fig. 1. A complimentary output pass-transistor logic network that is constructed of n channel devices, and a latch type swing restoring circuit consisting of two cross coupled CMOS inverters. The gate inputs are of two types, pass variables that are connected to the drains of the logic network transistors, and control variables that are connected to the gates of the transistors. The logic network has the ability to implement any random Boolean logic function. Fig.2, for instance, shows the implementation of an SRPL full adder. The complimentary outputs of the pass-transistor logic network are restored to full swing by the swing restoration circuit.

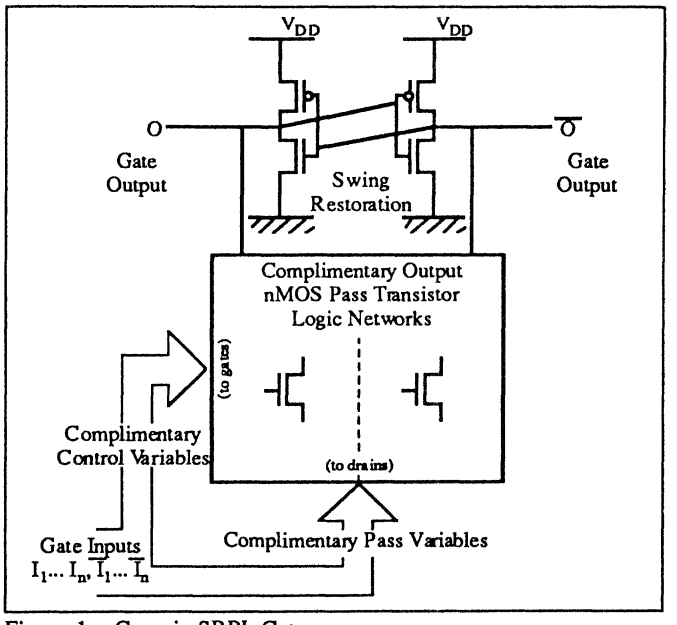


Figure 1 : Generic SRPL Gate

Gate Optimization

We have found that in the interests of speed, the nMOS transistors of the logic network farther away from the output should have larger drivability (i.e. size) than those closer to the output. This is because the transistors closer to the output pass smaller swing high signals due to the voltage drop across the transistors farther away from the output. The precise values for a given circuit depend on layout and other circuit considerations, so they must be determined by case by case simulation. Typical values are indicated in Fig. 2.

Reprinted from *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 12.5.1-12.5.4, May 1994.

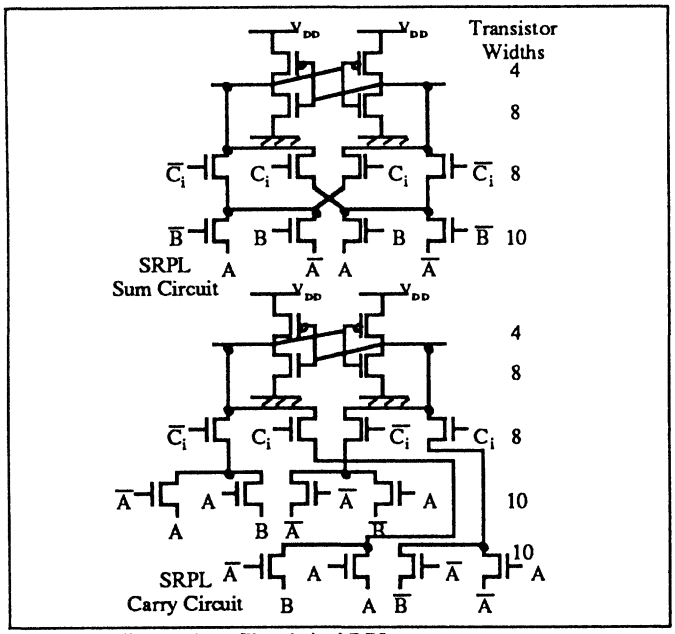


Figure 2 : Full Adder Circuit in SRPL

The optimization of the swing restoring latch is an important determinant of overall gate speed. If high speed latch inversion is required, the pMOS transistors should not be made too large. However, a large pMOS transistor size means that faster driving of the load is possible. Hence a trade-off exists, which is qualitatively demonstrated by the graph in Fig. 3. Simulations were performed on identical cascaded SRPL gates, each with a fan-out of 2, and assuming typical pass network transistor sizes shown in Fig 2. Simulations were done with SPICE, using parameters of Toshiba's 0.4μm CMOS process. The x-axis of Fig. 3 plots the ratio of the size of the pMOS transistor to the size of the topmost pass network nMOS transistor, while the y-axis plots the delay from the $0.5V_{DD}$ mark of a pass input of the gate to the $0.5V_{DD}$ mark of the output of the *subsequent* gate in the cascade.

For very small values of the $p_{latch}/n_{network}$ ratio, the gate output load becomes too large for the pMOS to be able to drive efficiently, and for very large values, the latch requires an inordinate amount of time to flip, reaching infinity (i.e. doesn't invert) over a certain limit. There exists a further dimension to the optimization, in that the curve of Fig. 3 moves up for very small or very large values of the $n_{latch}/n_{network}$ ratio. If the latch nMOS device is too small, discharging is penalized, whereas if it is too large, it introduces undue capacitive loading.

On the whole, the graph shows that though there is a trade off in determining the size of the pMOS transistor in the latch, there exists substantial design margin, making it easy to design circuits in SRPL. This design margin also means that SRPL circuits are quite robust against process variations, which might cause the threshold voltages of the transistors to fluctuate.

Performance Comparison with Competing Techniques

Full adders in CMOS, CPL, DPL, DCVSPG and SRPL were constructed, and simulated in the cascaded conditions shown in Fig. 4. Again, 0.4μm CMOS process parameters were used to perform SPICE simulations. The worst case waveforms for each of the full adders are shown in Fig. 5. Other performance values are recorded in Tab. 1.

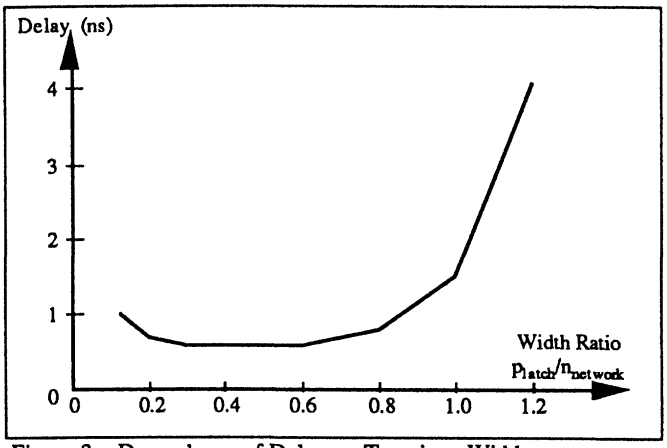


Figure 3 : Dependence of Delay on Transistor Widths

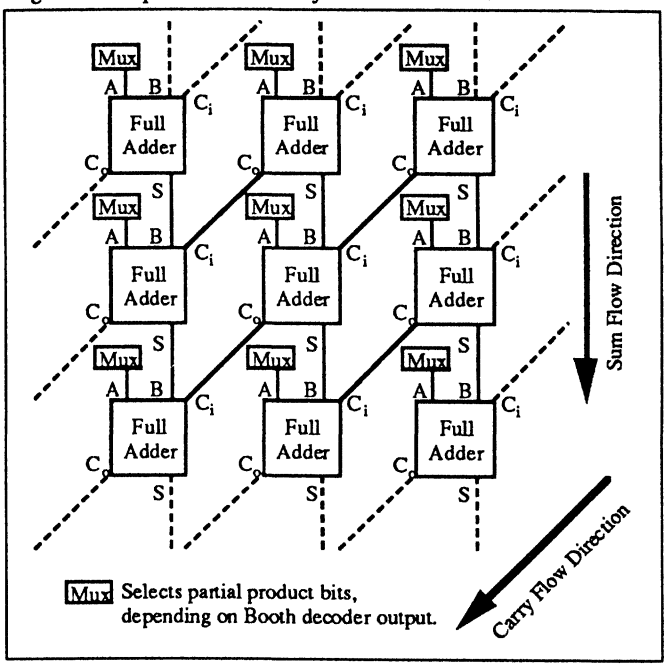


Figure 4 : Carry Save Addition of Partial Products

As Fig. 5 shows, CMOS has the slowest speed. Moreover, power consumption is quite high. The main reason for these poor performance figures is that the inefficient pMOS network of CMOS leads to a higher transistor count, larger gate area, and larger input capacitances due to the poor drivability of the pMOS transistor. DPL proves to be about 30% faster than CMOS, but this is at the expense of a higher transistor count, and more power consumed. DCVSPG is much faster than CMOS, but suffers from the problem that it cannot be used in the array structure of Fig. 4. The reason for this is that there is no pull down mechanism other than that through the pass-transistor networks. Thus for long chains of cascaded gates, the pull down becomes severely degraded as shown by the dotted line of Fig. 5.

CPL, as Fig. 5 clearly shows is the fastest of the five techniques. However, this is achieved at the expense of high power consumption. Furthermore, CPL suffers from the major drawback that it is a non-full swing technique. The non-full swing signals

at the inputs of the inverters mean poor noise margins, particularly as the inverter threshold is susceptible to process variations. Moreover, CPL circuits consume static power because of the leakage current that is always flowing through one of the inverters of a gate. The inverter output never quite reaches V_{SS} as the curve of Fig 5. shows. Because a V_{DD} of 3.3V is high relative to a channel width of 0.4μm, the speed degrading effects of the leakage are not prominent. However, when V_{th} is a significant fraction of V_{DD}, as it will certainly be in the future, the fall time of the output lengthens, and CPL becomes slower than SRPL.

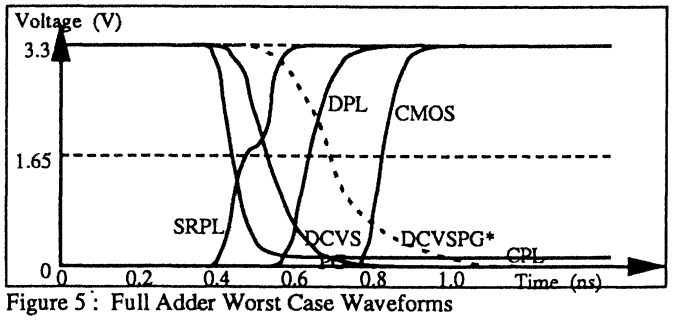


Figure 5 : Full Adder Worst Case Waveforms

SRPL has good speed performance. In the simulated conditions of Fig. 4, each SRPL circuit within the full adder fans out to only two other similarly sized circuits (carry and sum). This implies relatively light loading conditions, much less than the usual CMOS stage ratio of 3.5 or 4. It is important to note that this condition is not restricted to the simulated case. Low fanout is a very common occurrence in the design of VLSI circuits, particularly in data paths. In such conditions, it makes sense to connect the pass-transistor network output to the gate output, and to restore the swing with the cross coupled pair of inverters. The initial rise in voltage caused by the pass network output takes the gate output voltage a good margin above the $V_{th,n}$ of the transistors of the following gate, speedily setting up the correct logical path. Also because of the relatively light loading conditions, the inversion of the latch is faster, and so the $p_{latch}/n_{network}$ ratio can be made slightly larger. Thus a good pull up time through the *a priori* set up logical path of the following gate is achieved.

Table 1 : Comparison of Full Adder Circuits

	CMOS	CPL	DPL	DCVS PG	SRPL
Speed (ns)	0.82	0.44	0.63	0.53	0.48
Power at 100Mz (mW)	0.52	0.42	0.58	0.3	0.19
Power-Delay Product (normalized)	1.0	0.43	0.86	0.37	0.21
Transistor Count	40	28	48	24	28

As Tab. 1 shows, SRPL has the lowest power consumption and the lowest power-delay product of the different techniques. The main reasons for the low power are the low transistor count and the low input capacitance. Also, the fast inversion action of the latch quickly cuts off any d.c. path through the pass network.

In summary, SRPL shows itself to be a very competitive low power, high speed circuit technology. SRPL circuits will also occupy less area because of the lower transistor count. Particularly because the number of p channel devices is small, less area will be wasted on well boundary separation. The pMOS transistors are also smaller than, for instance, the CPL case, leading to slightly better area performance. This promising logic technique was used to construct a Multiply and Accumulate Circuit (MAC) for multimedia applications.

Multiply and Accumulate Circuit

The multiply and accumulate operation is crucial to a wide range of signal processing applications. With the increasing level of integration of processors dedicated to multimedia, it has become essential that high speed MAC macrocells be provided on chip. However, high speed is not the sole imperative. System portability is also a key issue, and hence low power is also very important. The MAC presented in this paper was designed with these requirements foremost in mind.

MAC Architecture

The overall circuit is shown in Fig. 6. The multiplier and multiplicand are 16-bit wide, whereas the accumulated result has a bit width of 32. A pipelined scheme was not implemented because the frequency of operation was expected to be more than sufficient to cover even the most advanced multimedia applications. Furthermore, pipelining introduces problems of complicated control and timing, and extra area and power required by the pipeline registers.

A Booth decoding scheme was used to obtain 8 partial products, which are added in a carry save manner as shown in Fig. 4. Each full adder row receives a running sum and carry from the row above. The very top adder of each column of the summation receives one of its inputs from the accumulated total of the previous cycle, which is fed back as shown in Fig. 6. A Wallace tree architecture for partial product addition was not used because the such an architecture would lead to larger power consumption due to the larger area and wiring requirements. Each of the full adders in the partial product summation array is constructed using the SRPL technique described above.

The final CLA adder cum register to which the partial product summation array outputs its carry saved result uses the same design as that of (5), where a dynamic sense amplifying scheme is used to perform both carry propagation, and latching of the final result. This design is ideally suited to the MAC design because of the high speed addition followed by the instantaneous latching. The complimentary outputs of the SRPL based summation array perfectly match the complimentary input requirements of the sense amplifying technique used by the final adder. It should be noted though that the dynamic sense amplifying technique used in (5) is completely different from the static swing restoring technique proposed in this paper.

Performance

The MAC was fabricated using a double metal 0.4μm process as summarized in Tab. 2. The chip photomicrograph is shown in Fig. 7. As Tab. 2. shows, the MAC operates at a maximum frequency of 150MHz, which more than sufficient for multimedia applications. Moreover, the power consumed is only 34mW at

this frequency, satisfying the other important multimedia requirement. The 150MHz operating frequency translates to a one cycle delay time of 6.7ns. For comparison, the MAC was simulated with a CPL partial product addition array. The simulated delay time was 6.3ns.

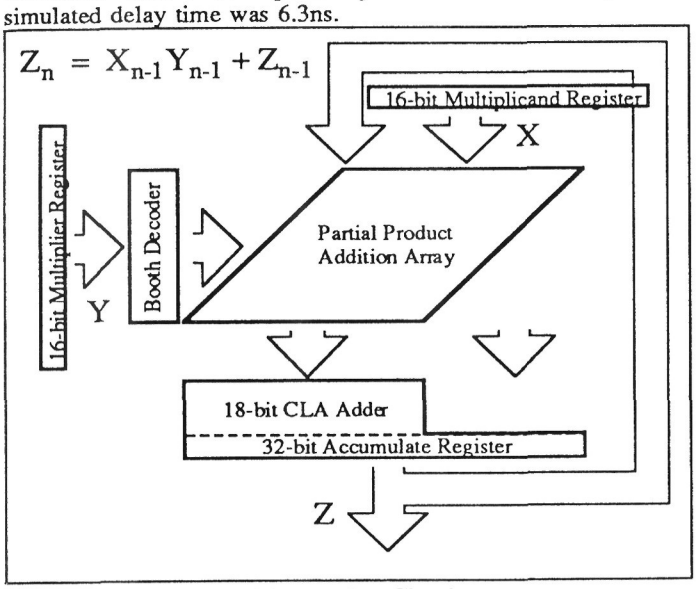


Figure 6 : Multiply and Accumulate Circuit

Though the SRPL MAC is 0.4ns slower than the CPL version, it should be remembered that CPL is the fastest technique ever reported, being nearly twice as fast as CMOS. Moreover, the power consumed by the CPL version was estimated to be more than twice that consumed by the SRPL MAC. In addition, as has been mentioned, CPL suffers from margin problems that will be exacerbated by the future reduction of the supply voltage, and this reduction in V_{DD} will also lead to speed degradation.

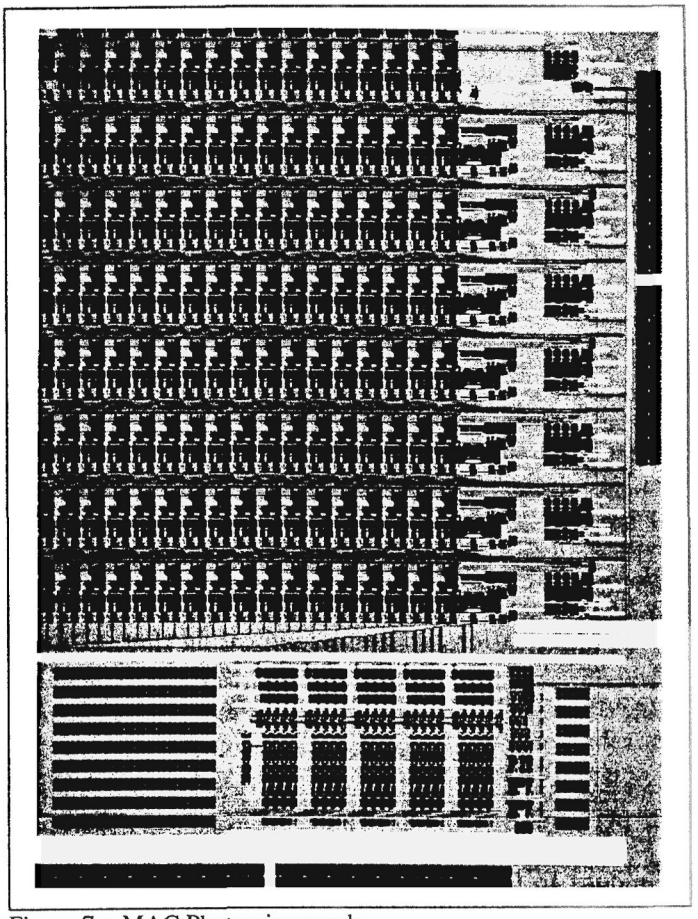


Figure 7 : MAC Photomicrograph

Conclusion

A new high speed, low power logic circuit technology was proposed, and used to implement a multiply and accumulate circuit in double metal 0.4μm CMOS. The MAC achieves a frequency of 150MHz, and 34mW and shows much promise for multimedia applications.

Table 2 : MAC Characteristics

Technology	CMOS process
n Channel Length	0.4μm (Eff. 0.39μm)
p Channel Length	0.5μm (Eff. 0.47μm)
Gate Oxide Thickness	9 nm
No. of Metal Layers	2
Power Supply Voltage	3.3 Volts
Operating Frequency	150MHz
Latency	0 cycles
Power Consumed at 150MHz	34 mW
Active Area	0.98 mm^2

Acknowledgements

The authors would like to gratefully thank the assistance and encouragement of Fumihiko Sano, Yoshinori Watanabe, Masataka Matsui, Hidetoshi Koike, Fumitomo Matsuoka, Masakazu Kakumu and Kenji Maeguchi

References

(1) K. Yano *et al* , "A 3.8ns CMOS 16x16 Multiplier Using Complimentary Pass-transistor Logic," vol. 25, no. 2, pp.388-395, April 1990.

(2) M. Suzuki *et al* , "A 1.5ns 32bit CMOS ALU in Double Pass-transistor Logic," 1993 IEEE International Solid-State Circuits Conference, pp. 90-91.

(3) F.S. Lai and W. Hwang, "Differential Cascode Voltage Switch with Pass Gate Logic Tree for High Performance CMOS Digital Systems," 1993 International Symposium on VLSI Technology, Systems and Applications, pp. 358-362.

(4) L.G. Heller, W.R. Griffin, J.W. Davis and N.G. Thoma, "Cascode Voltage Switch Logic : A Differential CMOS Logic Family," 1984 IEEE International Solid-State Circuits Conference, pp. 16-17

(5) M. Matsui *et al* , "Sense-amplifying pipeline flip-flop scheme for 200MHz video de/compression macrocells," in press (1994 IEEE International Solid-State Circuits Conference).

0.5V SOI CMOS Pass-Gate Logic

TSUNEAKI FUSE, YUKIHITO OOWAKI, MAMORU TERAUCHI,SHIGEYOSHI WATANABE,
MAKOTO YOSHIMI, KAZUNORI OHUCHI, AND JUN'ICHI MATSUNAGA
ULSI RESEARCH LABORATORIES
TOSHIBA CORPORATION, KAWASAKI, JAPAN

Demand for low-power ULSIs for mobile electronic equipment is increasing rapidly. To reduce power consumption, lower operating voltage and minimized device size (or count) is essential. To lower the actual threshold voltage and lower the operation voltage, SOI MOSFET with gate-body connection is proposed [1]. However, the circuit architecture that affords the maximum advantage of the body controlled SOI MOSFET is not reported. This SOI CMOS pass-gate logic offers the lowest operation voltage and reduced transistor dimensions.

Figure 1 shows conventional and proposed pass-gate logic. In the conventional complementary pass-gate logic (CPL, Figure 1a), the high-level signal of the pass-gate network is less than the supply voltage, Vcc [2]. This is because the pass-gate turns off when the source voltage reaches Vcc-Vt, where Vt is the threshold voltage of pass-gate which is increased by the body-effect. The drive capability of the network is degraded due to the channel resistance of pass-gates, so the output signal from the pass-gate network is amplified by using the buffer. In SOI CMOS pass-gate logic (Figure 1b, c), the body of SOI pass-gate is connected to the input signal given to the gate. Low threshold voltage for the on-state pass-gate and high threshold voltage for the off-state pass-gate is realized, and the increase in the threshold voltage due to the body-effect is suppressed. Two types of buffer suitable for the SOI pass-gate logic are examined. The buffer used in the Type A logic is composed of two CMOS inverters and a pMOS latch circuit, as shown in Figure 1b. The body of the MOSFET is connected to the gate (gate-body connection, GBC scheme). For the buffer used in the Type B logic, pull-up pMOSFETs are cross-coupled [3]. The body of the cross-coupled pull-up pMOSFET is connected to the buffer input, (input-body connection, IBC scheme), as shown in Figure 1c. Figure 2 shows the full-adder delay versus supply voltage. For SOI pMOS / nMOSFETs, the absolute value of the threshold voltage is 0.4V at 0V body-bias, and is 0.17V at 0.5V body-bias. Due to the low threshold voltage for the on-state MOSFET in the GBC scheme, the Type A full-adder reduced the delay to 1/3 of that of the conventional SOI CPL at 0.5V. Lowest operation voltage, V_{CCmin}, is improved by 0.17V by the GBC scheme, where V_{CCmin} is defined as the supply voltage which gives 2ns delay.

Transistor dimension are optimized for Type A (GBC) and Type B (IBC) pass-gate logic. The major difference between Type A and Type B logics is that the pass-gate network drives only two nMOSFETs in the Type B logic, while the pass-gate network drives two nMOSFETs and two pMOSFETs in the Type A logic. Figure 3 shows the full-adder delay versus the gate-width of the pass-gate network. By use of optimized buffer dimensions (Wp/Wn=0.6, Wu/Wn=0.4 for the Type B logic, and Wp/Wn=2.0, Wu/Wn=1.1 for the Type A logic), the optimum pass-gate width of the Type B logic is 0.6Wn, while that for the Type A logic is 1.3Wn. As a result, the total transistor dimension of the Type B logic is less than half that of the Type A logic.

The buffer using cross-coupled pull-up pMOSFETs reduces total transistor dimensions. There are two design options to control the body-bias. Figure 4 shows two types of buffer chain using the cross-coupled pull-up pMOSFET. One type uses the GBC scheme, and the other uses the IBC scheme (Figure 4). In the buffer using the GBC scheme, the on-state pull-up pMOSFET keeps high threshold voltage until the output node responds. This is because the body of the pull-up pMOSFET is connected to the output node. In the buffer using the IBC scheme, on the other hand, the threshold voltage of the on-state pull-up pMOSFET decreases before the output node responds. As a result, the buffer using the IBC scheme operates with high-speed and small short-circuit current, compared with the buffer using the GBC scheme.

A 40-stage buffer chain is used to measure the speed advantage of the buffer using the IBC scheme, in the Type B logic. A micrograph of the test chip is shown in Figure 5. Figure 6 shows the ratio of the buffer delay with the IBC scheme to that with the GBC scheme versus supply voltage. Measured threshold voltage of the SOI MOSFET is 0.58V at the body-bias of 0V, and 0.35V at the body-bias of 0.5V, respectively. The IBC scheme is 36% faster than the GBC scheme at 0.5V, and the V_{CCmin} is improved by 0.08V. Type B logic using the IBC scheme is 10 times faster than the CPL, and the minimum operation voltage is improved by 0.25V.

Multiplication is useful for estimating the logic performance. In the pass-gate full-adder using the buffer with the IBC scheme, the dissipation is reduced by 0.5V operation and reduced transistor dimensions. For a 16x16b multiplier using a Wallace-tree adder and CLA adder, the simulated multiplication time is 18ns at 0.5V. And the power-delay product is 70pJ including 50pF I/O, was more than an order of magnitude improvement for the CPL (Figure 7).

Acknowledgments:

The authors are grateful to H. Tago, T. Mizuno, and Y. Ushiku for helpful discussions and thank T. Arikado, A. Hojo, and H. Hara for encouragement.

References:

[1] Assaderaghi, F., et al., "A Dynamic Threshold Voltage MOSFET (DTMOS) for Ultra-Low Voltage Operation," IEDM Techical Digest, pp. 809-812, Dec., 1994.

[2] Yano, K., et al., "A 3.8-ns CMOS 16μ 16-b Multiplier Using Complementary Pass-Transistor Logic," IEEE J. Solid-State Circuits, vol. 25, pp. 388-395, April, 1990.

[3] Heller, L. G., et al., "Cascade Voltage Logic: A Differential CMOS Logic Family," ISSCC Digest of Techical Papers, pp. 16-17, Feb., 1984.

Reprinted from *Digest of Technical Papers, IEEE Journal of Solid-State Circuits Conference*, pp. 88-90, February 1996.

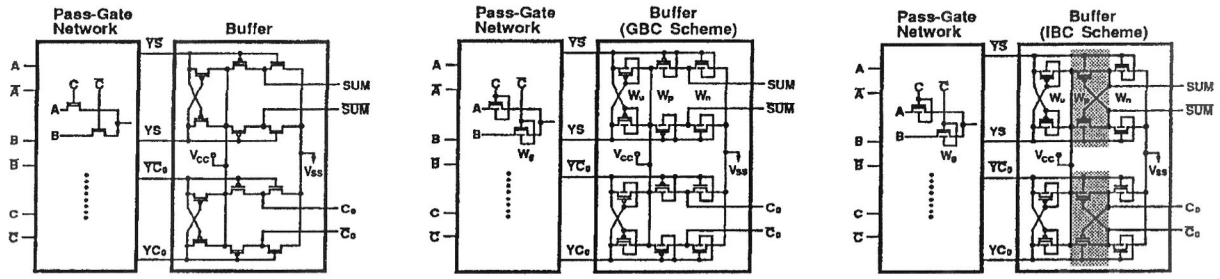


Figure 1: Conventional and proposed pass-gate logic. (a) CPL, (b) Type A, (c) Type B.

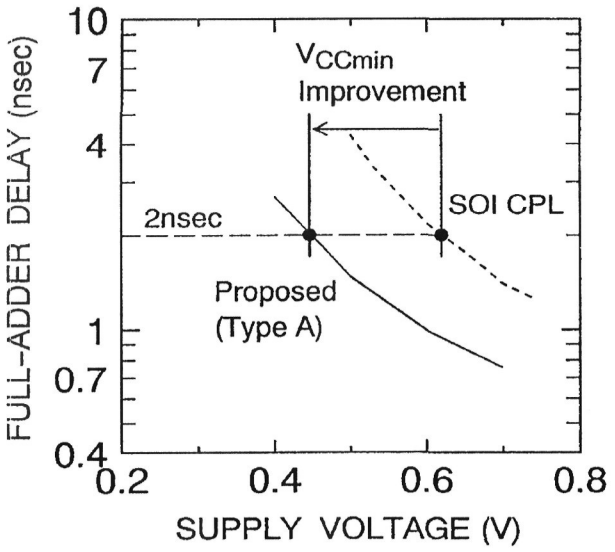


Figure 2: Simulated full-adder delay vs. supply voltage.

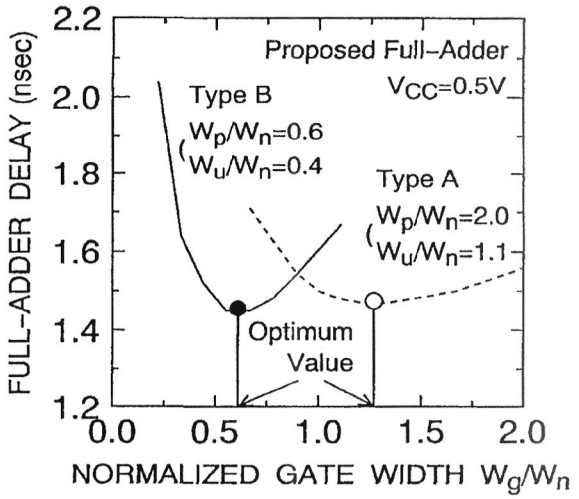


Figure 3: Gate-width optimization for pass-gate logic.
Figures 4 and 5: See page 424.

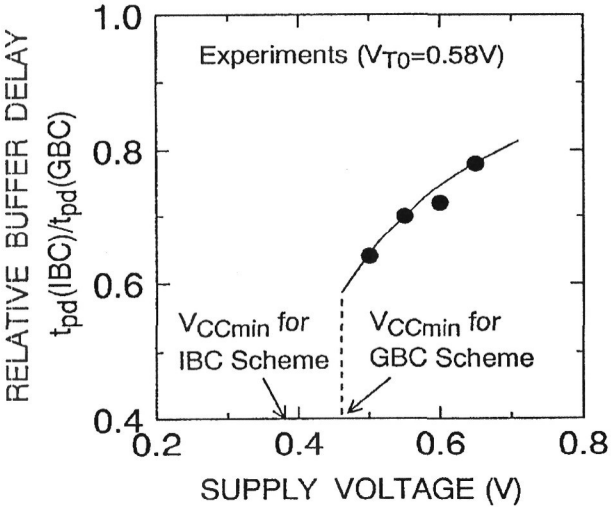


Figure 6: Measured buffer delay vs. supply voltage.

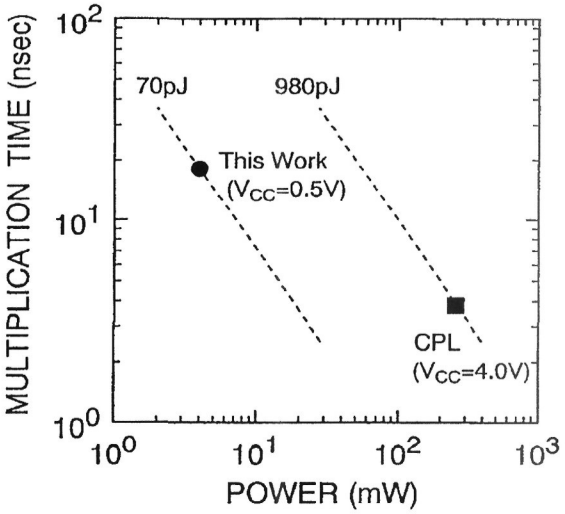


Figure 7: 16x16b multiplier power-delay product.

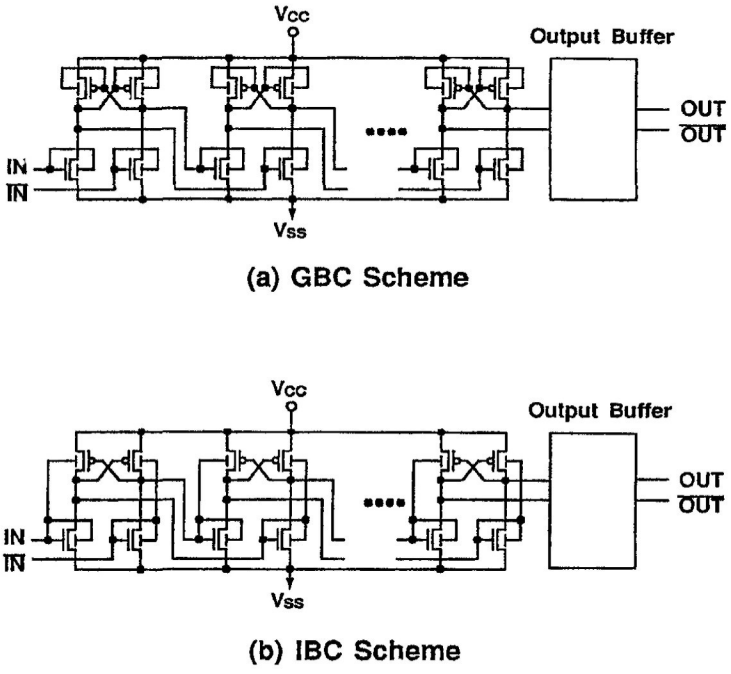


Figure 4: Buffer chain schematics: (a) GBC (b) IBC .

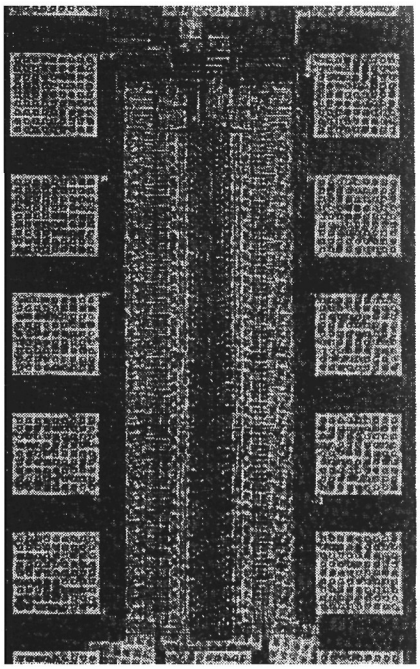


Figure 5: Chip micrograph.