Integrator-Chain Multiplier

Hsu Liang Ho and Kenneth C. Smith

Department of Electrical Engineering
University of Toronto
Toronto, Canada

Abstract

Switched-capacitor (SC) signal processing, being a sampled-data, analog-circuit technique, has a natural affinity to multiple-valued logic (MVL). Its simplicity in creating and manipulating signed-signals (in the form of voltage or charge) in any desired radix makes the SC topology a unique circuit technique for MVL implementation, especially when the base value is relatively high. In this paper, an MVL multiplier circuit scheme called the Integrator-Chain method is presented. This circuit scheme is particularly suited to the SC topology. The feasibility of circuit implementations of the Integrator-Chain Multiplier design in various number systems, including the sign-bit number representation, the R's and (R-1)'s complement representations and the signed-digit number (SDN) system, is discussed. In particular, it is demonstrated by simulation that a 4x4 multiplier circuit operating in a radix-7 SDN system has a size comparable to the simplest binary counterpart.

1. INTRODUCTION

MVL signals are relatively analog-like in comparison to binary ones in the sense that there are many legitimate intermediate values between 0 and 1. On the other hand, they can be viewed as digital because their signals are discretized, nevertheless. MVL circuits may, therefore, possess analog or analog-like elements as well as binary-like decision-making ones. SC, being a sampled-data analog topology, is easily capable of providing both extremes. The discretized charge transfer caused by the fixed capacitance values used in SC circuits, and the simple mechanisms available for the implementation of comparison are quite evident. The natural relationship of SC to MVL is unmistakable!

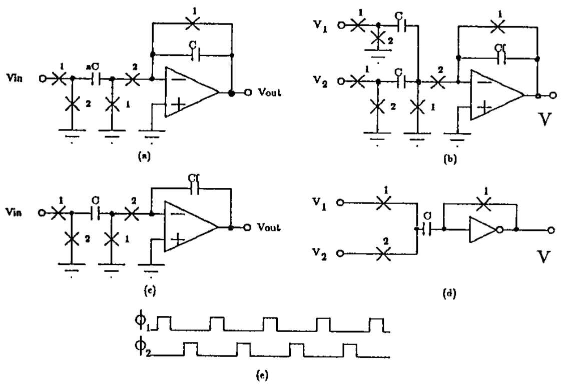


FIG 1 (a) Gain Stage, (b) Adder, (c)Integrator, (d) Comparator, (e) Clock Signals

SC technology has been employed in many areas of application such as filtering [1], A/D conversion [2], and other circuit-module designs [3][4][5]. Correspondingly there are a few well-developed basic SC building blocks (figure 1): they are integrators, adders(or subtractors), gain stages(or buffers), and comparators. In the diagrams to follow, an "X" represents a switch with the number beside it indicating the clock phase during which the switch is closed. These SC functional blocks (with certain specializing modifications applied where necessary) will also be used as the backbone structure for the SC Integrator-Chain multiplier design. The integrators are used to create MVL-threshold references, and MVL signals; comparators are used for MVL-signal detection; adders are there to create and provide outputs.

2. THE CONCEPT AND MOTIVATION OF THE INTEGRATOR-CHAIN METHOD

The Integrator-Chain Method has been proposed and discussed in references [6] and [7]. Here, this particular scheme is further modified to operate in other number systems. First, the general concept of this circuit scheme is illustrated using the simple block diagram shown in figure 2.

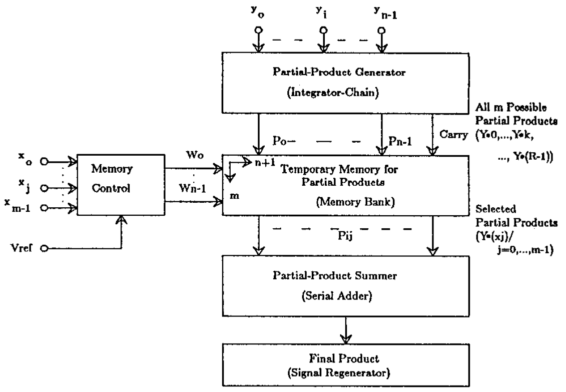


FIG 2 Block Diagram of a m x n Digit Multiple-Valued Multiplier in Radix-R

Whereas most multiplier circuits generate only the needed partial products, which are later summed to produce the final product, the Integrator-Chain method introduced here generates all possible partial products in a sequence, Yx0, Yx1,.....Yxk,...Yx(R-1), from which the needed ones are selected at an appropriate time. (Here Y is the multiplier and R is the base value as shown in the diagram). Referring to the diagram, each digit of the multiplicand, x_j, is compared to a voltage reference that supplies a step-up output sequence (V_{ref}) having R values separated by 1 and beginning at 0.5, namely 0.5, 1.5,...,R-0.5. When the stepping voltage reference first exceeds x_j, at k+0.5 (meaning that x_j is equal to k), the corresponding comparator provides a single pulse, and the needed partial product Yxk is strobed into the respective memory. One partial product could be, at

CH2886-0/90/0000/0363$01.00 © 1990 IEEE

a single time, strobed into more than a single memory location, if more than one digit of X has the same value. The comparators must provide only one WRITE (W_j) pulse between Reset signals for proper operation. All the required partial products are generated, and stored into the partial-product memory, in R cycles. The content of the whole memory of partial-products will be summed to produce the final product.

The motives underlying this Integrator-Chain topology are several:

Firstly, there is no time penalty in generating all possible partial products, than otherwise, provided the base value is equal to, or less than, the number of digits in the multiplicand. For example, if a single partial-product is generated in one cycle for each digit of the multiplicand, as is true in general, m steps or m cycles are required for completion of partial-product generation in the case of an mxn multiplication. Where n is the length of the multiplier. On the other hand, the Integrator-Chain method only requires R cycles, independent of the length of the multiplicand and of the multiplier. Moreover, it is much simpler to produce partial products in such a sequence, rather than at random. This is particularly true for implementation using SC technology.

Secondly, the memory required is basically simple and low-cost. Each memory cell is composed of a capacitor and several associated CMOS switches. They, working as a buffer between partial-product generation and summation processes, allow the use of serial addition (rather than parallel) for summation and, therefore, introduce a potential saving of space. Because SC is a hybrid-mode technology, SC circuits can accept charge inputs as well as voltage ones, and consequently an extra charge-sensing amplifier is not needed for reading the content of the memory.

Thirdly, this topology is chosen particularly with the goal of demonstrating the unique strengths of SC in comparison to other technologies. Here, the low output impedance of an SC integrator is well-utilized, as the integrator chain might at any time feed several memory locations without additional circuits intervening. This is in marked contrast to the requirement for copy circuits in CCDs, for example.

In figure 2, multiple-valued digits y_i must be available as parallel voltage signals because they are used to feed (in parallel) the inputs of the SC integrators in the integrator chain for several cycles. On the other hand, x_i can be presented either in parallel or in serial form. Partial-product summation could also employ either a parallel or serial structure. However, parallel addition is very costly, and provides little speed advantage here, while serial addition is less expensive, and more consistent with the original design goal, that of size minimization.

>From an application point-of-view, this topology is most suited for adaptive-coefficient multipliers. For this situation, Y, being a coefficient, can be supplied internally, thereby being automatically restored. This is to be contrasted to the general case, where Y must be restored before feeding the integrator chain, and fast restorers are expensive.

Note that in usual binary logic, there is no distinction made between an adaptive-coefficient multiplier and a general multiplier. However, here in MVL design, the cost difference implied by this distinction is quite obvious. Nevertheless, by limiting its application area, the utility of the topology does not necessarily diminish. In fact, many digital-signal-processing systems have a number of coefficient multipliers as their major components. Therefore, henceforth, the discussion following will be directed toward producing efficient designs for MVL adaptive-coefficient multipliers.

3. VARIOUS NUMBER SYSTEMS

The brief description of the Integrator-Chain Method given in the previous section provides only the most basic aspects of the design. Note that direct implementation of the circuit blocks according to the brief description given will produce a multiplier that is capable of handling only positive numbers. However, many recursive digital filters require subtraction, as well as addition. Thus, the multipliers used in such designs must be able to handle both negative and positive numbers. Therefore, the SC MVL multiplier demanded must be modified for application in general digital-filter designs. There are several choices for these modifications, including the sign-bit number representation (or in other words, the sign and absolute-value representation), 1′s and 2′s (or (R−1)′s and R′s) complement number representations, and the signed-digit number (SDN) representation.

3.1 SIGN–BIT NUMBER REPRESENTATION

The first and most direct representation is probably that of the sign-bit number representation. The sign of the final product would be simply the Exclusive NOR function of the sign bits of the input number and of the coefficient. This method requires almost no change to the original circuit topology. As well, it is simple conceptually. However, it shifts the burden to subsequent additions or subtractions of the products in an overall filter design.

Using the sign-bit number representation, comparison of magnitudes must be performed before adding 2 numbers with opposite signs (or subtracting 2 numbers with the same sign). Comparison is not always simple, particularly in dealing with MVL numbers represented by multiple-valued physical entities. The comparison might involve many steps. For instance, it is possible to arrange that the most significant digits each from the corresponding number are compared simultaneously to a step-by-step reference. If, at any time, the reference exceeds only one of the 2 digits, the region of inequality has been reached. If the reference exceeds the 2 digits during the same step (indicating equal digit values), comparison must continue with the following digits until an inequality is detected or the comparison of least significant digits is complete. Obviously, in the worse case, the operation must be performed through the length of the whole word.

Note that the apparently straightforward direct (analog) comparison between the 2 digit magnitudes will not provide the correct outcome because physical entities representing the same value are likely to be some what different (no matter how slightly). Direct comparison will indicate that one is larger than the other, while it is not, from a quantified multiple-valued point of view.

3.2 (R−1)′S AND R′S COMPLEMENT REPRESENTATION

The second choice is to adapt the 1′s or 2′s complement number representation from binary logic [8]. An additional digit is used to indicate the sign of the number, with the assignment of R-1 to a negative number and 0 to a positive one. In case of negative numbers, each of the other digits must be subtracted from R-1 to obtain the complement form.

Let us discuss the (R−1)′s complement first. Since the input is in (R−1)′s complement representation, the memory-control circuitry has to include an additional (MVL) down-counter as a voltage reference, in addition to the up-counter, and, as well, the corresponding comparators. The down-counter, providing a logic reference in the sequence of (R-1.5), (R-2.5),...., complements the input to reflect the correct magnitude in case the sign of the number is detected to be negative.

The coefficient Y is most conveniently represented in the sign-bit number representation independent of what scheme the input number and the final product use. This stipulation allows the integrator-chain to produce, directly, partial products with correct magnitudes. The integrator-chain always supplies the partial products in the definite order, Yx0, Yx1,..... If the coefficient is a negative number in (R−1)′s complement representation, padding needs to be done before partial-product generation. In this case, the length of the integrator chain must be the same as the length of the final product.

When the (R−1)′s complement input is negative, and the down-counter is used as the logic reference, a digit value of R-2 causes the partial product Yx1 to be strobed into the memory, where 1 is the correct absolute value for a digit of value R-2 in a negative (R−1)′s number. Therefore, with the above modifications, the partial-products, and thus the final product, will have the correct absolute value. Should the signs of the input and the coefficient be different (meaning that the final product is negative), the (R−1)′s complement of the final product can be easily obtained by subtracting each digit from R-1 .

A problem with the (R−1)′s complement number operation occurs in the addition of 2 numbers of opposite sign. Addition of 2 positive numbers is done directly. Addition of 2 negative numbers is just as simple, and only requires a 1 to be added to the least significant digit, independent of the outcome (of which, indeed there is only one possible). Unfortunately, addition of a positive and a negative number could have one of two possible outcomes. If the sum is negative, nothing needs to be done. However, if the sum is positive, a 1 must be added to the least significant digit. This may (in the worst case) cause a carry to propagate through the entire word. Many extra clock cycles must be allocated to accommodate this possibility. This makes the addition required in general filter design a potentially long process, if (R−1)′s complement is used.

Whereas the (R−1)′s complement number representation causes a certain difficulty in addition, the R′s complement representation poses no problem. The sum is correctly obtained by direct addition independent of the signs of the numbers. However, now, the problem lies not in addition but rather in multiplication. The fact is that the input to the multiplier cannot be handled as conveniently as in the case of (R−1)′s complement. In particular, the least significant digit must be treated differently if the input is a negative number. The direct comparison between the down-counter output and the least significant digit only reflects the (R−1)′s complement absolute value. A one-cycle delay must be added to the memory control for this particular digit. For example, recall that the integrator-chain produces partial products in the sequence Yx0, Yx1, Yx2,...... If the least significant digit (LSD) is R-2, the WRITE pulse will arrive 2 cycles after the Reset. With a one-cycle delay, this WRITE signal will cause the intended partial product Yx2 (instead of Yx1 in the case of a (R−1)′s complement number) to be strobed into the memory.

Another difference between (R−1)′s and R′s complement operation is in complementing the final product. Should the R′s final product be negative (which could be detected before the partial product summation starts), one extra unit of charge must be added to the LSD upon the (R−1)′s complementing function. Any means to provide addition of 1 to the LSD could be used in this context.

3.3 SIGNED-DIGIT NUMBER REPRESENTATION

The third alternative is the signed-digit number (SDN) system [9] in which a number may contain negative as well as positive digits. The redundancy in the SDN representation limits carry propagation to one position, and thus allows fast parallel addition. Taking radix-7 addition as an example, an SDN can be represented by a symmetrical set:

$$L = (-4, -3, -2, -1, 0, 1, 2, 3, 4)$$

An integer X can be coded as a sequence of x_i from set L

$$X = (x_{n-1} \cdots x_i \cdots x_0) = \sum_{i=0}^{n-1} x_i 7^i \qquad (1)$$

The addition of two numbers X and Y is performed in three steps:

$$z_i = x_i + y_i \qquad (2)$$

$$7c_i + w_i = z_i \qquad (3)$$

$$s_i = w_i + c_{i-1} \qquad (4)$$

Where z_i is a linear sum, w_i an intermediate sum, c_i a carry and s_i a final sum.

$$z_i \in (-8, -7, \ldots, 0, \ldots, 7, 8)$$

$$w_i \in (-3, -2, -1, 0, 1, 2, 3)$$

$$c_i \in (-1, 0, 1)$$

$$s_i \in L$$

The intermediate sum w_i and carry c_i can be obtained as follows:

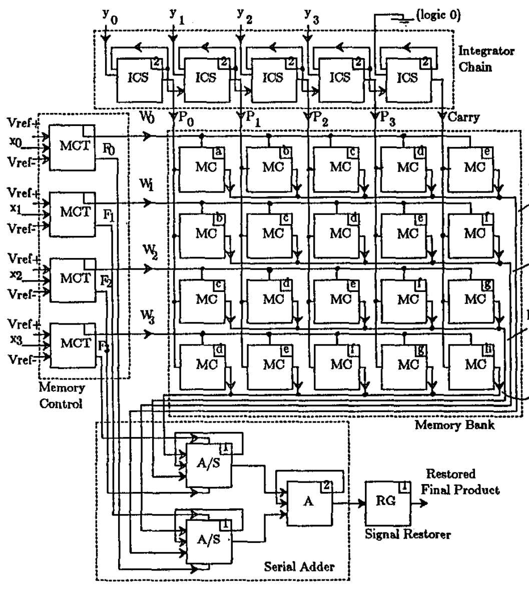
$$w_i = z_i - 7,\ c_i = 1 \ \text{ if } z_i > 3 \qquad (5)$$

$$w_i = z_i,\quad c_i = 0 \ \text{ if } -3 \le z_i \le 3 \qquad (6)$$

$$w_i = z_i + 7,\ c_i = -1 \ \text{ if } z_i < -3 \qquad (7)$$

The SDN system is usually used to increase the speed of multiplication through a process of parallel addition, made possible by redundancy in the SDN representation. However, here, SDN is employed to serve a different purpose, namely negative-number handling. In practice, SDN operations are highly suited to SC MVL implementation. The signed digits can be represented by positive or negative voltages or charges. Neither generation nor addition of such physical entities demands complex procedures on a local scale.

However, overall,compared to the two preceding methods, this one requires some extensive circuit modifications. Thus, immediately, for an effective SDN design, two power supplies (V+, V-) are required (or at least convenient), as shown in figure 3, the Memory Control circuit must be doubled now, since the input digits are simultaneously compared to 2 references, which are Vref+ generated by an up-counter and Vref- generated by a down-counter respectively. In the diagram, the flag signals (F_j) indicate the signs of the corresponding x_j. When x_j is negative, the respective row of partial products in the memory bank will be subtracted rather than added during partial-product summation.



ICS------Integrator-Chain Stage
MC------Memory Cell
MCT----Memory Control
A/S------2-input Adder/Subtractor
A--------2-input Adder

FIG 3 4x4 Radix-7 SDN Multiplier

The number or letter in the box at the right upper corner of each circuit block indicates the clock phase during which the output of the block is valid. Blank indicates that the output is valid at all times.

Circuit details for each functional part, including clock signals will be described in the following:

3.3.1 Partial–Product Generation

The name Integrator-Chain multiplier comes from the fact that partial products are generated by a chain of integrator stages. The integrator-chain consists of N full SC integrator stages (ICS) and one simple SC integrator, where N is the length of the multiplier (or most likely of the coefficient). While a full SC integrator-chain stage requires circuitry for the functions of both input provision and carry generation, a simple SC integrator can be used to provide the most significant digit of a partial product by accumulating the most significant carries. In figure 3, an ICS is used instead of a simple SC integrator to make the diagram more uniform.

One SDN integrator-chain stage is shown in figure 4 (a). It consists of 3 functional parts: Part 1 is the addition of the input y_i (provided by the binary to MVL (B/M) converter) and the carry correction signals C_i and C_{i-1} (through the gain boxes, whose gains are 7 and 1 respectively); Part 2 is the integrator (I) whose step size is determined by the result of the above addition; Part 3 is a comparator (C) which is used to detect and generate a carry. The value of the binary signal b_2 determines whether R/2 or -R/2 is to be used as the carry-generating threshold, and therefore the sign of the carry.

At a circuit level, the SC circuits that perform the functions of part1 and part2 can be easily combined to become a single one that has a well defined boundary, as shown in figure 4(b).

Each SDN coefficient can be conveniently represented in a binary code. In figure 4(b), each SDN digit, positive or negative, is in binary sign-bit representation. The sign bit (b_2) designates that either the positive or negative power supply is used to charge up an input capacitor while the remaining binary digits (b_1, b_2) encode the absolute value of the MVL digit.

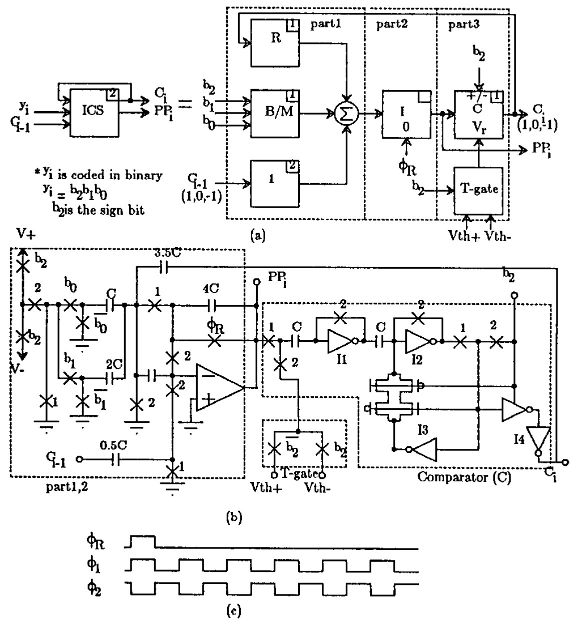


FIG 4 (a) Symbol for an Integrator-Chain Stage
(b) Radix-7 SDN Integrator-Chain Stage
(c) Clock Signals

Two threshold references V_{th+} and V_{th-} corresponding to ±R/2 (for an odd base) are used. However, only one is connected to the carry-generating comparator at a time, depending on the value of the corresponding sign bit b_2.

In the case of a positive digit, the sign bit (b_2) is low, and V+ is used to provide step-up integration. As the output of the integrator exceeds R/2, R units of charge are subtracted from the carry-generating stage directly, and one unit of charge is added to the next stage in the next phase. In the case of a negative digit, the sign bit (b_2) is high, and V- is used to provide step-down integration. As the output of the integrator becomes lower than -R/2, R units of charge are added to the carry-generating stage directly, and one unit is subtracted from the next stage.

Note that b_2 also selects which type, P- or N-channel, of MOS transistor is used to control the feedback path around the comparator. If a P-channel device is chosen (as b_2 is high), the feedback path is only activated when the output of the integrator is lower than V_{th-}, and is open otherwise. This asymmetrical property allows the integrator to step down from above to below V_{th-} in the same clock phase during which comparison is performed. Therefore, step-decrementing, carry-generating, charge-adding (when carry is -1) can be done in a single clock phase, and the maximum output value of the integrator can be limited to the base value. A larger logic separation and thus better noise immunity will result. Similar arguments hold when b_2 is low.

Because the comparator feedback path is activated either below or above (but never on both sides of) the threshold of a normal CMOS inverter, the rate of change of the output of the comparator is more gradual for one transition, and more abrupt for the other. To avoid ambiguity in interpreting the output of the comparator, a special inverter with a variable threshold falling in the region of abrupt change should be used to detect the comparator's output. Such an inverter and its symbol are shown in figure 5. Referring to the diagram, the threshold of the inverter shifts down when the switching signal is high, and shifts up when the switching signal is low.

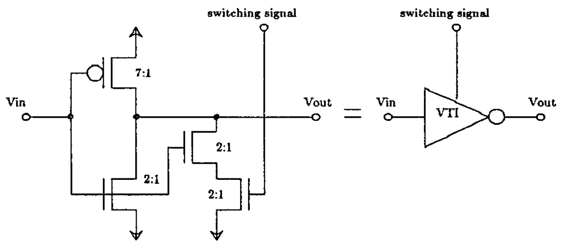


FIG 5 Variable-Threshold Inverter and its Symbol

3.3.2 Memory Control and Memory

Regarding the integrator-chain stage in figure 4, it can be observed that the integrator-chain so modified, still produces only those partial products corresponding to the positive input digits. Such a partial product sequence is Yx0, Yx1, ..., Yxk,..., Yx(R-1). Inversion of these products must be done to correspond to negative digits. There are 2 convenient ways of obtaining the inversion. Method (1) is to use a number of negative unity-gain stages to directly invert the partial products which are to be strobed into the memory, as the respective digits are detected to be negative. Method (2) is to provide a flag to indicate that the respective memory content is to be subtracted rather than added (as shown in figure 3). In method (2) the actual inversion is performed by the partial-product summer, through subtraction. Method(2) requires fewer additional opamps and is therefore preferred.

Since there is no inverted partial product available using this method, only one WRITE signal is needed for each input digit. The memory control circuit for each input digit is shown in figure 6. It

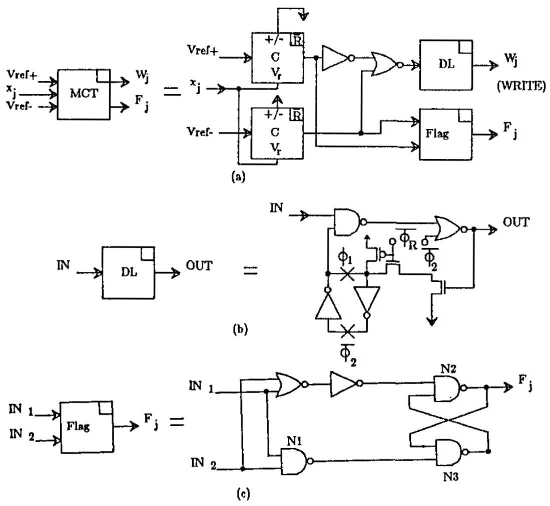


FIG 6 (a) Memory-Control Symbol and Circuit
(b) Dynamic-Latch Symbol and Circuit
(c) Flag-Indicator Symbol and Circuit

consists mainly of 2 comparators, one dynamic latch and a flag circuit. The input is simultaneously compared to V_{ref+}, a positive voltage reference stepping upward in the sequence 0.5, 1.5,..., and V_{ref-}, a negative voltage reference providing the sequence -0.5, -1.5,... The outputs (with one being inverted) of the two comparators are connected to an NOR gate whose output changes from low to high when both comparators have changed state (indicating that the absolute value of the input digit equals k). The transition of the NOR gate is detected by the dynamic latch used to provide the desired half-cycle pulse between Reset signals needed to strobe the desired partial product into the memory.

For this scheme, one flag is required for each input digit to indicate the sign of the input digit. Correspondingly, the respective partial product will be subtracted (instead of being added) during partial-product summation. To set the flag, the case of a positive input digit must be differentiated from that of the negative. It can be observed that the outputs of the two comparators are different most of the time, except for the transition interval, during which one comparator has changed state and the other has not. If the comparator with V_{ref+} changes state first, indicating a negative input, both comparator outputs would be high. If the comparator with V_{ref-} flips first, both outputs would be low during the transition time and the input is positive. In the special case of zero value input, the transition could provide either one of the above two outcomes. Fortunately, zero can be positive or negative, and the result is still correct.

In figure 6(c), the flag is set to high when both inputs are low and set to low when both inputs are high. The state of the flag will remain unchanged when the inputs differ.

As illustrated in figure 3, the memory bank for a 4x4 multiplication contains 4 rows by 5 columns of memory cells (the fifth column being for storage of carries). One such array is shown in figure 7(b). Each memory cell is simply a capacitor with some switches(figure 7(a)). When W_j is high, the input is stored into the memory cell. The content is valid at the output (as a charge signal) when the pulse z arrives. One memory bank is used if the system allows the partial-product-generation process to start after the previous final product is produced, and after the memory is relinquished. Otherwise, two arrays would be required and partial-product generation and summation would be performed in parallel. The clock signals can be viewed as delayed versions of the Reset pulse which can be obtained by using a binary shift register.

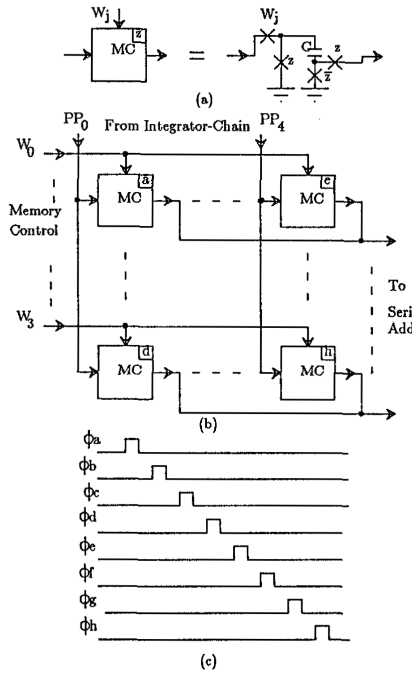


FIG 7 (a) Memory-Cell Symbol and Circuit
(b) Memory Bank for 4x4 Multiplication
(c) Multiplexing Clock Signals

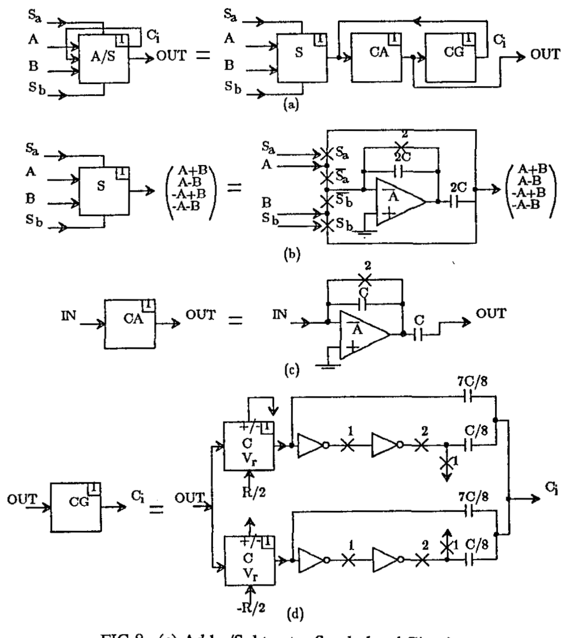


FIG 8 (a) Adder/Subtractor Symbol and Circuit
(b) Selector Symbol and Circuit
(c) Charge-Adder Symbol and Circuit
(d) Carry-Generator Symbol and Circuit

3.3.3 Partial–Product Summation

The contents in the memory array are summed by a serial adder to produce the final product. The serial adder consists of two 2-input adder/subtractor (A/S) modules and one 2-input adder(A) as shown in figure 3.

The adder/subtractor module is composed of 3 parts (figure 8) which are the selector, charge adder and the carry-generating circuit. The function of the selector is to provide an output A+B, A-B, -A+B, or -A-B corresponding to S_a high and S_b high, S_a high and S_b low, S_a low and S_b high, or S_a low and S_b low respectively. The charge adder adds the output of the selector and the carry correction signals. The carry-generating circuit of course is used to detect and generate carrys.

The adder (figure 9) is simply composed of an SC adder and a carry-generating circuit.

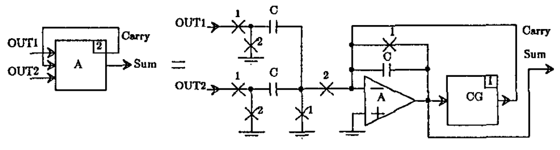


FIG 9 2-Input-Adder Symbol and Circuit

The final product is produced one-digit-per-cycle in sequence with the least-significant digit first. Restoration of the final product can be done with any suitable restorer which has a satisfactory speed. One example is the 'rough' and 'fine' restorer scheme implemented by SC circuits proposed in [6][7].

3.3.4 Remarks

In the integrator-chain, when a non-zero carry is generated, subtraction (or addition) of 7 units of charge from (or to) the carry-generating stage is done in the same phase as comparison is performed. However, addition (or subtraction) of one unit of charge to (or from) the next stage is performed in the next phase. This arrangement avoids the need to allocate time in a single clock phase for carry propagation.

Another advantage of using SDN is that the accumulated errors in the partial products are reduced simply because the number of cycles to complete partial-product generation is reduced to half. Restoration can now be done only at the final-product point instead of (possibly) at the outputs of the integrator-chain. The reduction in the number of restorers used is expected to well offset the requirement of additional circuitry for this SDN design.

4. RESULTS AND CONCLUSION

SPICE circuit simulations have been performed on all major circuit blocks to verify the feasibility of the design scheme. The clock signals used for simulation have a period of 800ns. The technology used is 5μ CMOS. Since the requirements on the opamps employed in MVL circuits are much less strict than those used in analog designs, a simple opamp configuration consisting of 2 stages is selected. Each opamp used typically contains 8 MOS transistors, 1 biasing resistor and 1 compensation capacitor. All the switches employ mininum-sized MOS transistors(L/W ratio of 2/1 for N-type and 5/1 for P-type). A unit capacitance of 0.3pF is found to be sufficiently large for a 4x4 radix-7 SDN multiplier design. Too small a unit capacitance leads to incorrect computation due to errors produced by parasitics and clock feedthrough. On the other hand, too large a unit capacitance increases overall cost beyond more than can be justified.

Simulation results come quite close to those expected since the circuit blocks used are indeed well-developed in other applications. For simulation detail, one may refer to reference [7].

Table 1
(Approximate) Device-Count Comparison
Based on Similar Number Spans

4x4 Radix-7 SDN Multiplier		10x10 Serial-in Binary Multiplier	
Integrator-Chain	385	10 Half Adders	180
Memory-Control	280	8 Full Adders	288
Memory-Bank	140	19 Flip-Flops	646
Restorer	110	19 Selectors	266
Serial-Adder	320		
Total	1235	Total	1380

Regarding the size of the 4x4 radix-7 SDN multiplier, Table 1 provides an approximate device count for the complete circuit together with counts for a 10x10 serial-in binary multiplier (of which the circuit is shown in figure 10). This comparison was chosen because 4 digits of SDN in radix-7 have about the same number span as 10 digits in binary.

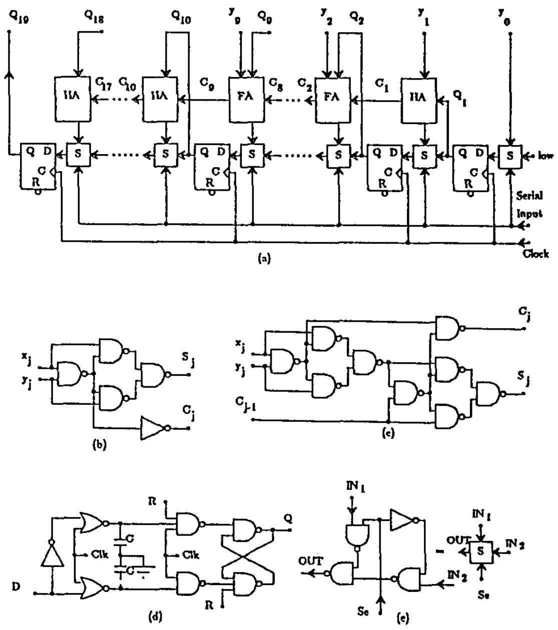


FIG 10 (a) 10x10 Binary Multiplier with Serial Input
(b) Half Adder (HA)
(c) Full Adder (FA)
(d) Rising-Edge-Triggered D Flip-Flop
(e) Selector (S)

In counting the devices of the SDN multiplier, a unit capacitor is considered to be equivalent to 2.5 devices. Thus, for example, in the SC circuits presented, the capacitors denoted by 1C or 2C would contribute 2.5 or 5.0, respectively, to the total device count. Reference generators and clock signal generators are not included in the total device count because they can be shared among multipliers. For derivation detail, see reference [7].

As one can see from table 1, the resulting SDN multiplier circuit has a size comparable to a simple binary counterpart in terms of equivalent device counts(1235 compared to 1380).

Note that this SC MVL-multiplier-circuit scheme imposes no direct restriction on the base value used. Therefore, if technology allows higher precision in fabrication, higher base values, such as 15, could actually be used. For this extension, the multiplier circuit described demands only a few minor modifications. They include changes to 1) the number of clock cycles between Reset signals, 2) the V_{th} values for carry generation, and 3) the ratios of capacitors. Therefore, a 4x4 radix-15 SDN multiplier would have a similar device count as the one in radix-7 mentioned above but a larger number span equivalent to that of a 15x15 binary multiplier. As shown in Table 2, the equivalent binary multiplier would require 2130 device count while the SDN design remains approximately at 1235.

In fact, since a shorter Integrator-Chain, and a correspondingly smaller memory bank and memory-control circuit, are used for the same number span in a higher radix, further reduction in circuit size is evidently possible using a high-radix switched-capacitor approach for large multipliers.

Table 2
(approximate) Device-Count Comparison
Based on Similar Number Spans

4x4 Radix-16 SDN Multiplier		15x15 Serial-in Binary Multiplier	
Integrator-Chain	385	15 Half Adder	270
Memory-Control	280	13 Full Adder	468
Memory-Bank	140	29 Flip-Flop	986
Restorer	110	29 Selector	406
Serial-Adder	320		
Total	1235	Total	2130

REFERENCES

1. R. Gregorian, K. W. Martin, and G. C. Temes, "Switched- Capacitor Circuit Design," *Proceedings of the IEEE*, vol.71, no.8, Aug.1983, p941-966.

2. C. C. Shih, P. R. Gray, "Reference Refreshing Cyclic Analog-to- Digital and Digital-to-Analog Converters," *IEEE, J.of Solid-State Circuits*, vol.sc-21, no.4, Aug. 1986, p544-554.

3. K. Watanabe and G. C. Termes, "A Switched Capacitor Multiplier/ Divider With Digital and Analog Outputs," *IEEE, Transactions on Circuits and Systems*, vol.31, no.9, Sept, 1984, p796-800

4. K. Nagarai, "A Novel Switched-Capacitor Rectifier," *Int. J. of Electronics*, vol.59, no.1, Jul. 1985, p91-95.

5. R. P. Colbeck, "A CMOS Low-Distortion Switched-Capacitor Oscillator with Instantaneous Start-Up," *IEEE, J.of Solid-State Circuits*, vol.sc-19, no.6, Dec.1984, p996-998.

6. H. L. Ho and K. C. Smith,"Switched-Capacitor Circuits In the Implementation of Multiple-Valued Logic," *IEEE, 19th Int. Symp. on MVL*, Kwangchow, China, May 1989, p.202-209.

7. H. L. Ho, "Switched-Capacitor Circuits In the Implementation of Multiple-Valued Logic," *M.A.Sc. Thesis 1989*, Electrical Engineering Department, U.of T.

8. I. Flores, "The Logic of Computer Arithmetic," *Prentic-Hall Publication* 1963

9. S. Kawahito, M. Kameyama, T. Higuchi and H. Yamada, "A High-Speed Compact Multiplier Based on Multiple-Valued Bi-Directional Current-Mode Circuits," *IEEE,Int. Symp. on MVL*, Boston, Massachusetts, USA, May 1987, p172-180.