A Fast 16 Bit NMOS Parallel Multiplier

CLAUDE P. LEROUGE, PIERRE GIRARD, AND JOËL S. COLARDELLE

Abstract—**A new architecture for a fast parallel array multiplier is described. Using a 3 μm E/D NMOS process, a 16×16 bit trial circuit has been designed. A multiply time of 120 ns has been achieved with a power dissipation of 200 mW and a silicon area of 5 mm². This new architecture concept greatly reduces the logical depth of the array by rearranging internal delays. It remains applicable in principle to any MOS, CMOS, GaAs, or bipolar technology, provided to keep a suitable approach for the adder cell design.**

I. Introduction

HIGH-SPEED multiplication is of prime importance in signal processing systems. Although faster architectures are known, the most widely used approach in LSI multipliers is the "carry–save" adder array, mainly because of its modular and repetitive layout which enables a compact and error-free design (see [1] and [4]). A reference diagram is given in Fig. 1 for an 8×8 carry–save array.

The structure of an $N\times N$ bit multiplier comprises $N\times(N-1)$ adder cells in its implementation. If T_a is the logic delay time for an adder cell, the propagation delay time through the combinatorial array is of $(2N-2)\times T_a$. Hence, the speed of the adder dictates the multiply time.

In a carry–save array, there are two main directions for signal propagation.

1) The carry propagation direction which runs from the least significant product bit to the most significant product bit.

2) The sum propagation direction in which each partial sum result must run in a line of fixed binary weight.

These two directions are crossing in each adder cell, with a 45° angle among the array.

However, in most full adder cell designs, the carry output signal is obtained faster than the sum output signal, but this fact is not profitable to the overall multiply delay time because of the step-by-step propagation inherent to the carry–save structure.

The new architecture, as proposed in this paper, keeps most advantages of the carry–save array, but significantly reduces the number of logic gate delays required to perform a complete multiplication.

II. Main Structural Characteristics

The new array keeps the same carry propagation as for a standard carry–save multiplier. On the contrary, the sum propagation is speeded up by transferring each intermediate sum result from three to three cells, which is equivalent to having three lines of adder cells working in parallel. A specific adder cell design is used in which the carry output signal is obtained with the shortest delay time called T_c. The sum output signal may reach a delay time T_s equal to $3T_c$. The total multiply time is close to $2N\times T_c$ and *not* to $2N\times T_s$ or $2N\times T_a$ as for a standard carry–save multiplier.

In this trial circuit, it has been decided to work with unsigned numbers in order to keep the testing job easier. However, there would be no problem in converting the design to two's complement numbers without any change in performance.

The circuit will be described as an array of 19 lines numbered from 0 to 18, and of 18 rows numbered from -2, -1, 0, to 15.

Figs. 2–5 give details on the interconnection patterns which will be explained in Sections IV and V. The different symbols used for the adder cells are recalled in Figs. 6–9 where detailed schematics are shown, and they will be explained hereunder.

III. Adder Cells

The N bit operands are X and Y. Bits X_0-X_{15} are applied inputs to rows 0–15. Bits Y_0-Y_{15} are applied inputs to lines 0–15.

The basic adder cells are shown in Figs. 6 and 7, which are used, respectively, for even and odd lines.

The three input bits are A, B, C_{in}; the output bits are S and C_{out}. The first input bit A is the product of two operands bit $A=X_i\cdot Y_j$. First and second summands A and B are combined to produce an intermediate sum M and its complement $\overline{M}$ defined by $M=A\oplus B$. These intermediate signals M and $\overline{M}$ are used to control both the carry-in to carry-out propagation (or generation) and to build up the final sum output S.

On alternating lines of adder cells, two slightly different designs are used:

Fig. 6 { inputs $\overline{A}, \overline{B}, C_{in}$; outputs $\overline{S}$, $\overline{C}_{out}$ } line number 4, 6, 8, 10, 12, 14, 16

Fig. 7 { inputs $\overline{A}, \overline{B}, \overline{C}_{in}$; outputs $\overline{S}$, C_{out}. } line number 5, 7, 9, 11, 13, 15

On line number 17, a supplementary inverter per carry output is needed, and there is no local product $X\cdot Y$, so the

Manuscript received November 3, 1983; revised January 23, 1984.
The authors are with the Laboratoire Central de Télécommunications, F-78141 Vélizy Villacoublay Cedex, France.

0018-9200/84/0600-0338$01.00 ©1984 IEEE

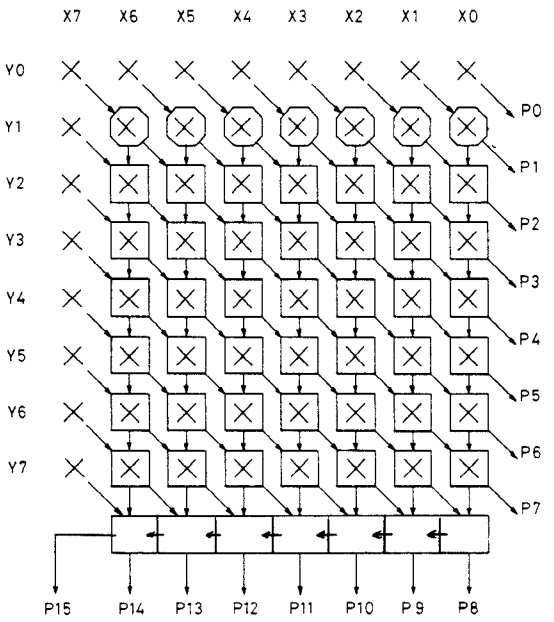

Fig. 1. 8×8 carry–save array.

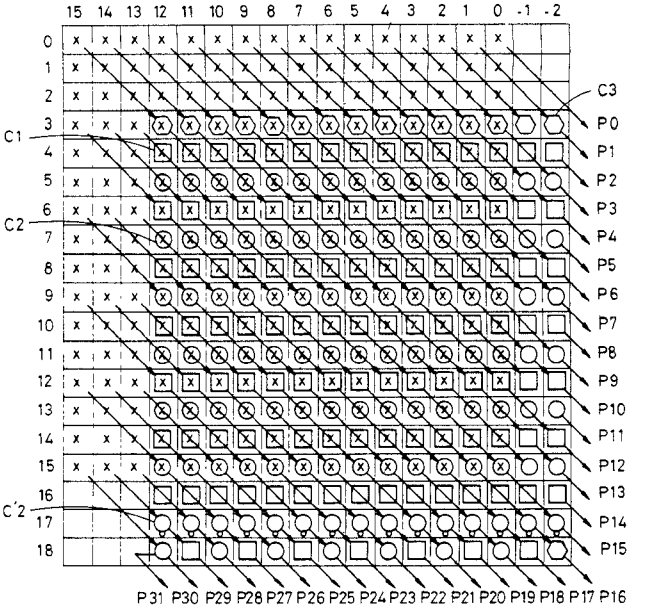

Fig. 3. First group of sum paths.

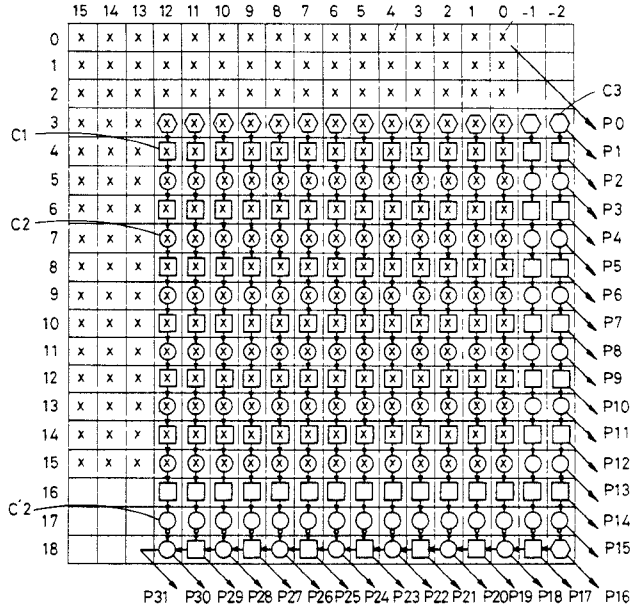

Fig. 2. 16×16 array/carry paths.

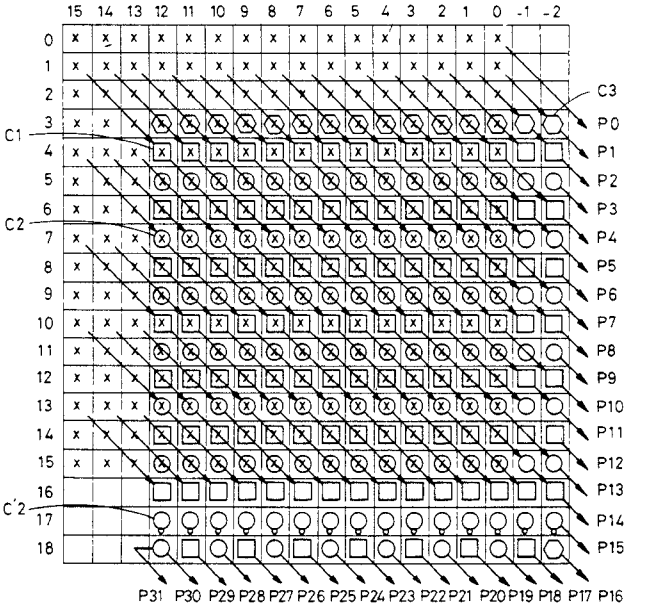

Fig. 4. Second group of sum paths.

corresponding NAND gate is removed. The resulting cell is shown in Fig. 8.

On line number 3, only two summands are to be added instead of three. This leads to a simplified adder cell shown in Fig. 9.

On cells used at the matrix periphery, that is, on lines 16, 17, and 18 and on rows -1 and -2, the local NAND gates for $X \cdot Y$ are removed.

On the opposite side, cells used for lines 0, 1, and 2 and rows 13, 14, and 15 have NAND gates only, without adders.

The need for those different cells will be better understood in Sections V and VI.

IV. Carry Signals Propagation

Fig. 2 gives the carry interconnection pattern as drawn on the chip. It is quite similar to that of a carry–save design. In each cell, the transit of the carry signal includes only one logic inverter and one pass transistor.

This approach is usually called a "Manchester" chain, and it is used throughout in the summands array (see [5] and [6]). Moreover, a strong geometrical dimensioning of the transistor devices of the carry chain has been chosen in order to strengthen the effect.

The intrinsic features of that design may be summarized as follows:

1) it minimizes the propagation delay and the number of transistor devices,

2) it allows working with alternatively true and complement carry signals,

3) it provides, nevertheless, both carry-in and $\overline{\text{carry-in}}$ information needed in each cell.

The result expected is really a very minimum carry-prop-

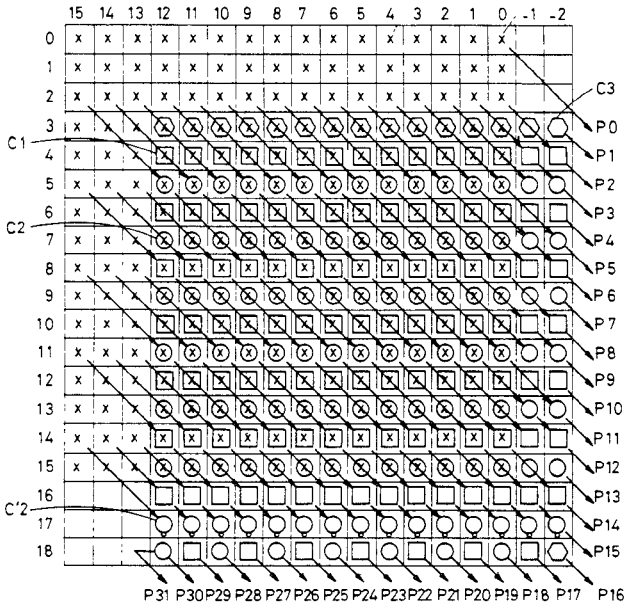

Fig. 5. Third group of sum paths.

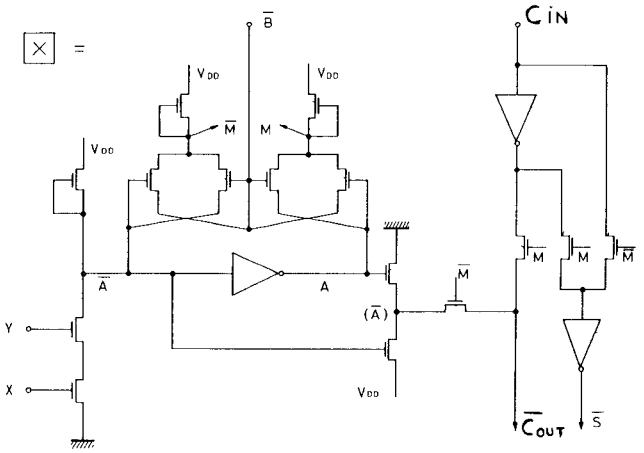

Fig. 6. Adder cell for even lines.

agation delay close to the optimum allowed by the technology.

V. Sum Signals Propagation

In the basic adder cell, a big part of the efficiency has been devoted to improve the carry-propagation speed; hence, the sum propagation is quite slow in comparison. The second aspect of the structure design aims to improve the speed of sum propagation, and to keep it compatible with a short multiply time.

This is obtained by splitting the entire array of adders in three groups working in parallel for the sum signal transmission. The sum signals interconnection patterns of those three groups are shown in Figs. 3–5, respectively. One group of adders is made of lines 0, 3, 6, 9, etc., each three lines of the matrix. Similarly, the second group includes lines 1, 4, 7, $\cdots$, and the third group comprises lines 2, 5, 8, $\cdots$.

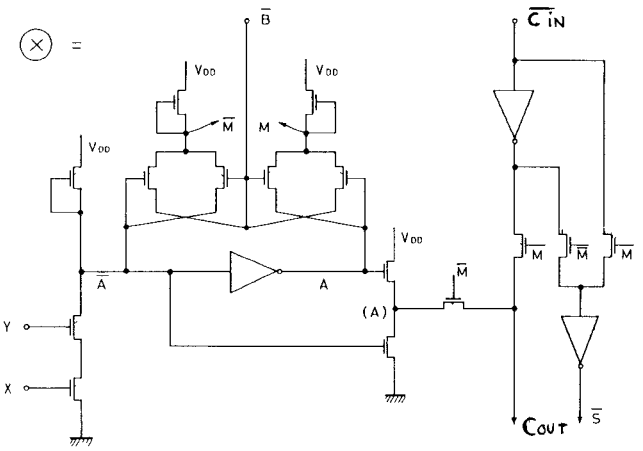

Fig. 7. Adder cell for odd lines.

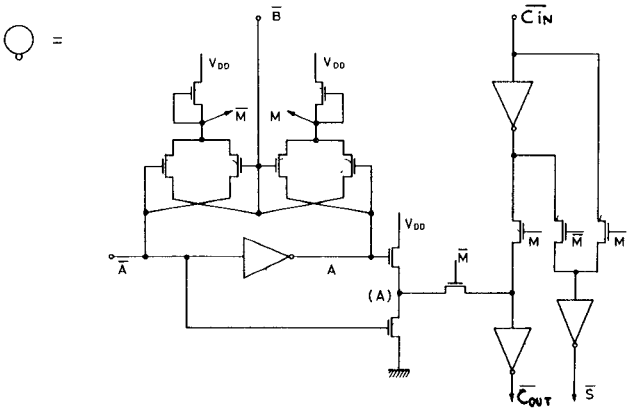

Fig. 8. Modified adder cell.

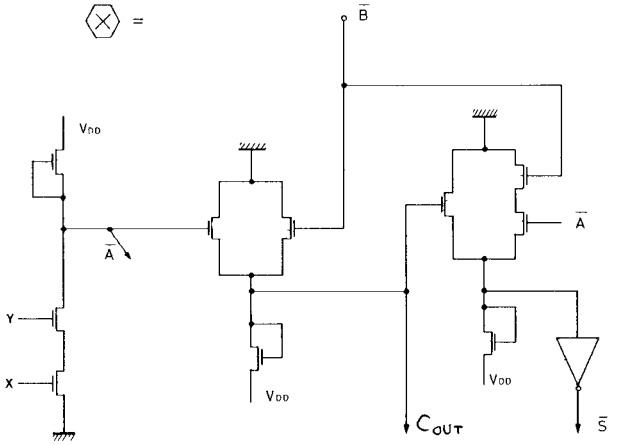

Fig 9. 2,2 adder cell.

In effect, three logic sum variables are propagated in parallel instead of one per diagonal. Each intermediate sum result skips over two cells, before being logically combined and delayed by the next one. Nevertheless, there is only one adder cell layout, which includes two supplementary metal tracks skipping in diagonal over it and not related to it. The effect of additional capacitive loading due to these relatively short metal tracks is negligible.

Using the above interconnection patterns of the sum signals, the delay T_S achieved for transmission of the sum

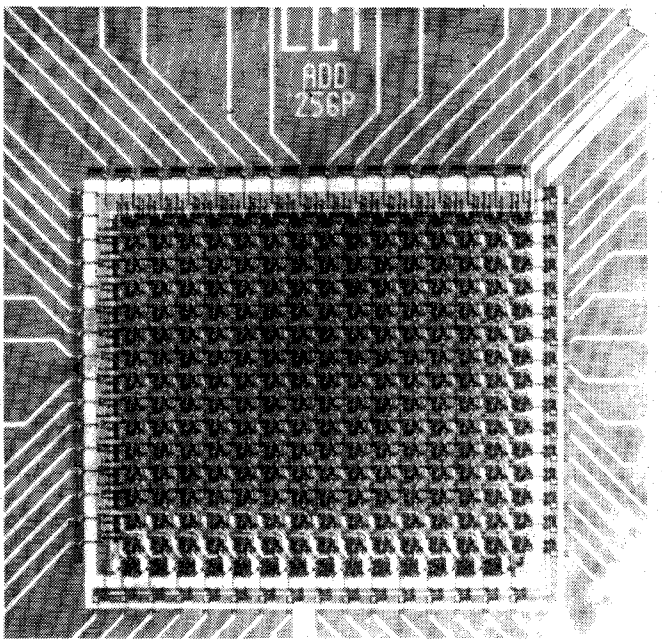

Fig. 10. Microphotograph of the chip.

information is matched to the delay of three lines of carry propagation, e.g., 3 T_c. In other terms, the relative sum delay per cell is equal to $T_s/3$.

As a result, no supplementary delay is introduced in the matrix due to the slow response time of the adder cell concerning the sum output signal, and there is no need to boost the corresponding logic gates which represent 80 percent of the circuit.

VI. Overall Array Features

The matrix is completed by a "gutter" collecting the intermediate results coming out of the three summing groups and combining them into a single bit of the final product. The associated adder cells need not use the local NAND product. On the opposite sides, the first three lines and the first three columns are not equipped with adder cells, but only with NAND gates.

Hence, the total number of NAND product is still $N \times N = 256$, and the total number of adder cells is still $N \times (N - 1) = 240$, as it would be in a carry–save array.

VII. Chip Characteristics

A 16×16 trial multiplier circuit has been implemented, processed, and tested. It uses a 3 μm enhancement/depletion NMOS technology produced at Intermetall, Freiburg, Germany. A peculiar adder cell design is adopted in which up to three enhancement devices are conducting in series. This is functionally not different from the design of a three-input NAND gate; the simulation on computer has shown acceptable figures for logic levels and noise margins.

The use of four different adder cells does not really impair the regularity and modularity of the array. In fact, one single basic adder layout was drawn, which has two slightly different variants of exactly the same dimensions. These variants are applied for odd and even lines. The same rule applies for rows -2 and -1 and for lines 16, 17, and 18.

The complete array does not look much different from what a true carry–save array would be.

A chip photograph is shown in Fig. 10.

The multiply array has no input or output registers and works on a purely combinatorial basis. Digit inputs are directly tied to the matrix, and product outputs are implemented with noninverting buffers in order to provide comfortable testing possibilities. The delays and power consumption of these buffers have been measured separately, and are not included in the figures which concern the 16×16 array only.

The main characteristics of the chip are

- $16 + 16$ input pads (multiplicand + multiplicator)
- 32 output pads (complete product)
- 5 V power supply
- power consumption 200 mW (buffers excluded)
- chip area 5 mm^2 (5000 transistors)
- adder cell area 0.018 mm^2 (21 transistors)
- response time for the longest multiplication: 120 ns (buffers excluded).

VIII. Performance Comparison

Three multiplying arrays will now be compared.

1) An 8×8 standard carry–save array actually implemented and processed prior to the new described approach.

2) A 16×16 standard carry–save array *not* processed, but extrapolated from the previous 8×8 array.

3) The new 16×16 design described in the present paper.

The following chart gives a summary of performance assuming the same 3 μm NMOS technology.

	8×8 C.Save	16×16 C.Save	16×16 New	
Nb Adders	56	240	240	
Area	1.25	5	5	mm^2
Power	50	200	200	mW
Time	120	240	120	ns
		3 μm E/D NMOS		

As may be seen, a decisive advantage in power–speed product is performed by the new concept. This is due to the combined effect of two complementary tricks:

1) the design of a special adder cell, allowing fast carry chains with a very short propagation time per cell,

2) the splitting of the array into several groups working in parallel for sum chains, thus compensating for a longer propagation time per cell.

The choice of the number of groups to put in parallel is a tradeoff which involves most technological factors. Half a dozen adder cells were tested in simulation before making the decision to build up with three groups.

Different technologies and different adder cell designs might lead to two groups or four groups for the optimum implementation.

IX. Conclusion

A lot of effort has been devoted by numerous authors to speeding up the arithmetic process of multiplication (see [5]–[9]). If $2N$ is the number of product bits and if T_c is the fastest achievable carry-propagation time across one single adder cell, the intrinsic lowest limit for the multiplication time may be expressed as

$$2N \times T_c.$$

This formula represents the necessary time to propagate the carry information from the lowest significant bit to the highest significant bit of the product. In other words, the ultimate speed for an $N \times N$ multiplying array cannot be better than the speed of a fast parallel adder of $2N + 2N$ bits, based on the same technology and design rules.

The new architecture concept as proposed in this paper is a significant step towards that limit. It also has the advantage of being independent of technology. Finally, this new concept could be applicable to other interesting approaches (such as the modified Booth's algorithm), with the result of improving their speed without an increase in power or in silicon area.

Acknowledgment

The authors wish to acknowledge the help of Dr. C. Obermeier from Intermetall for technological assistance and silicon processing.

References

[1] S. Waser, "High speed monolithic multipliers for real-time digital signal processing," *Computer*, pp. 19–30, Oct. 1978.
[2] E. E. Swartlander Jr., *Computer Arithmetic*. Stroudsburg, PA: Dowden, Hutchinson, Ross.
[3] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Electron. Comput.*, pp. 14–17, Feb. 1964.
[4] N. F. Benschop and L. C. M. Pfennings, "Compact NMOS array multipliers with inverting full adders," *Philips J. Res.*, vol. 36, pp. 173–194, 1981.
[5] S. Ong and D. E. Atkins, "A comparison of ALU structures for VLSI technology," in *Proc. Arith 6*, Aarhus, Denmark, June 1983.
[6] L. Conway and C. Mead, *Introduction to VLSI Systems*. Reading, MA: Addison-Wesley, 1980.
[7] W. K. Luk, "Silicon compilation of a fast parallel multiplier," thèse de 3ème cycle, Univ. Paris, Paris, France, 1983.
[8] R. P. Brent and H. T. Kung, "A regular layout for parallel adders," Dep. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU.CS 79.131, June 1979.
[9] J. Vuillemin, "A very fast multiplication algorithm for VLSI implementation," INRIA, France, Res. Rep. 183, 1983.

Claude P. Lerouge was born in Lille, France, on November 27, 1932. He graduated as an engineer of the Institut Industriel du Nord and as a Bachelor of Science of the Université de Lille in 1954 and 1955, respectively.

In 1960 he joined the Laboratoire Central de Télécommunications, Paris, where he worked on the development of PCM carrier systems. Since 1967 he has been the head of a team working in the field of full electronic telephone exchanges. Since 1973 this team has been deeply involved in custom LSI circuit design.

Pierre Girard was born in Paris, France, in 1930. He studied electronics at the Conservatoire National des Arts et Métiers.

In 1955 he joined LCT where he worked on the development of electronic switching systems (analog) for the French Navy, digital switching systems (encoder and PCM repeaters) for the French Army, and the new Unimat electronic PABX. He is now involved in custom LSI design.

Joël S. Colardelle was born in Issy les Moulineaux, France, on September 22, 1941.

Since 1974 he has been with the LSI Design Center, Laboratoire Central de Télécommunications, Vélizy, France, where he works in the area of large-scale integrated circuits, especially in circuit design.