Full systolic binary multiplier

J. Arechabala
E.I. Boemo
J. Meneses
F. Moreno
C. Lopez Barrio

Indexing terms: Array processing, Very large scale integration

Abstract: The paper describes the architecture of a binary multiplier that, because of its intrinsic regularity and simplicity, may be extended for any number of bits. It is a modification of the array of full adders scheme, which allows the operation with two's complement numbers in systolic mode. The design has been validated and implemented in the SOLO 1400 environment, using ES2 standard cells. Laboratory prototypes perform a multiplication every 24 ns. Finally, a preliminary evaluation of a full custom version is presented.

1 Introduction

The binary multiplication is completely systolisable. One of the first studies on array multipliers was published by J.A. Rajchman [1] in 1943. A comprehensive review of binary multipliers can be found in Reference 2. Most of the recent technical papers describe partially systolic architectures with different trade-offs between performance and regularity. In References 3 and 4, some new schemes are presented. The performance of a full custom design is shown in Reference 5.

This paper describes the architecture, implementation and results of an eight-bit full systolic multiplier with emphasis on practical aspects such as a data synchronism and parallel I/O interface cost. The principal characteristics of the topology are its regularity, simplicity and easy-to-route connections, all suitable for VLSI implementation. To obtain high through-put rate from this architecture, large input data streams are required to compensate for its high latency.

This work served as a training design exercise at the undergraduate VLSI program of the Engineering Electronic Dpt. ETSIT, Technical University of Madrid. The course involved two professors and 60 h of lectures. The design presented in this paper was implemented by students in about 200 h of terminal time. The objective proposed was to obtain a multiplication module for fine-grain systolic FIR architectures.

2 Multiplier architecture

The operations involved in the multiplication of positive numbers are shown in Fig. 1, which represents the simplified case of a 4 × 4 multiplier.

Paper 8732G (E3, E10), first received 8th October 1991 and in revised form 29th January 1992

The authors are with the Dpto. de Igenieria Electrónica, ETSI Telecomunicación, Universidad Politécnica de Madrid, Ciudad Universitaria, 28040 Madrid, Spain

$$
\begin{array}{cccccccc}
 & & & & B_3 & B_2 & B_1 & B_0 \\
 & & & & A_3 & A_2 & A_1 & A_0 \\
\hline
 & & & & A_0 \cdot B_3 & A_0 \cdot B_2 & A_0 B_1 & A_0 B_0 \\
 & & & A_1 \cdot B_3 & A_1 \cdot B_2 & A1 \cdot B_1 & A_1 \cdot B_0 & \\
 & & A_2 \cdot B_3 & A_2 \cdot B_2 & A_2 \cdot B_1 & A_2 \cdot B_0 & & \\
 & A_3 \cdot B_3 & A_3 \cdot B_2 & A_3 B_1 & A_3 \cdot B_0 & & & \\
\hline
O_7 & O_6 & O_5 & O_4 & O_3 & O_2 & O_1 & O_0
\end{array}
$$

Fig. 1 *4 × 4 binary multiplication*

It can be observed that partial multiplications *AiBi* may be done simultaneously. However, it is necessary to impose an ordering in the operations because the carries produced after adding a column must be passed to the next column addition. Fig. 2 shows the mapping of the multiplication process into a systolic architecture. Note that for the sake of clearness, all cell inputs and outputs that do not pass data are not drawn. In the implementation, they are grounded and dangling, respectively. To simplify the model, we have used just one kind of cell, whose structure is shown in Fig. 3. However, it would be possible to simplify the EPs that do not compute add with carry, do not perform the AND operation or do not pass data to neighbour nodes.

At every clock beat, each EP receives data from the South, *Ai*, the West, *Bi* and *Ci*, and the South-west, *Si*. Then, it computes the *Ai* × *Bj* + *Ci* + *Si* operation and passes the input data *Ai*, *Bj* and the result, *So* and *Co*, to the next EPs, maintaining the data flow directions.

To maintain proper synchronisation, the delay of diagonal channels must be equal to the delay of vertical plus horizontal channels. This is the reason why a double register has been included for *So*.

The following operations are computed to complete a 4 × 4 bit multiplication:

t_0: $R_{03} = A_0 \times B_3$
t_1: $R_{02} = A_0 \times B_2$; $R_{13} = A_1 \times B_3$
t_2: $R_{01} = A_0 \times B_1$; $R_{12} = A_1 \times B_2 + S_{03}$;
$R_{23} = A_2 \times B_3$
t_3: $R_{00} = A_0 \times B_0$; $R_{11} = A_1 \times B_1 + S_{02}$;
$R_{22} = A_2 \times B_2 + C_{12} + S_{13}$; $R_{33} = A_3 \times B_3$
t_4: $R_{10} = A_1 \times B_0 + S_0$;
$R_{21} = A_2 \times B_1 + C_{11} + S_{12}$;
$R_{32} = A_3 \times B_2 + C_{22} + S_{20}$
t_5: $R_{20} = A_2 \times B_0 + C_{10} + S_{11}$;
$R_{31} = A_3 \times B_1 + C_{21} + S_{22}$; $R_{42} = C_{32} + S_{33}$
t_6: $R_{30} = A_3 \times B_0 + C_{20} + S_{21}$; $R_{41} = C_{31} + S_{32}$;
$R_{52} = C_{42}$
t_7: $R_{40} = C_{30} + S_{31}$; $R_{51} = C_{41} + S_{42}$
t_8: $R_{50} = C_{40} + S_{41}$; $R_{61} = C_{51} + S_{52}$
t_9: $R_{60} = C_{50} + S_{51}$
t_{10}: $R_{70} = C_{60} + S_{61}$

R_{ij} is the result obtained in the EP_{ij}, with C_{ij} being the MSB and S_{ij} the LSB. $O_k = R_{ko}$ is the output of the multiplier. The first valid data appear at the output 11 clock cycles after the operands were entered into the array. The rest of the results flow each clock bit. The maximum output data frequency is determined by the delay of one EP plus the routing delay.

Both operands must enter the array at a rate of one bit per clock cycle (first A_0 and B_n, then A_1 and B_{n-1}, and so on), and the result also emerges in the same fashion (first O_0, then O_1, and so on). To produce a parallel I/O, it is necessary to add a chain of registers of different lengths for each bit position. Fig. 4 shows the topology with parallel data I/O.

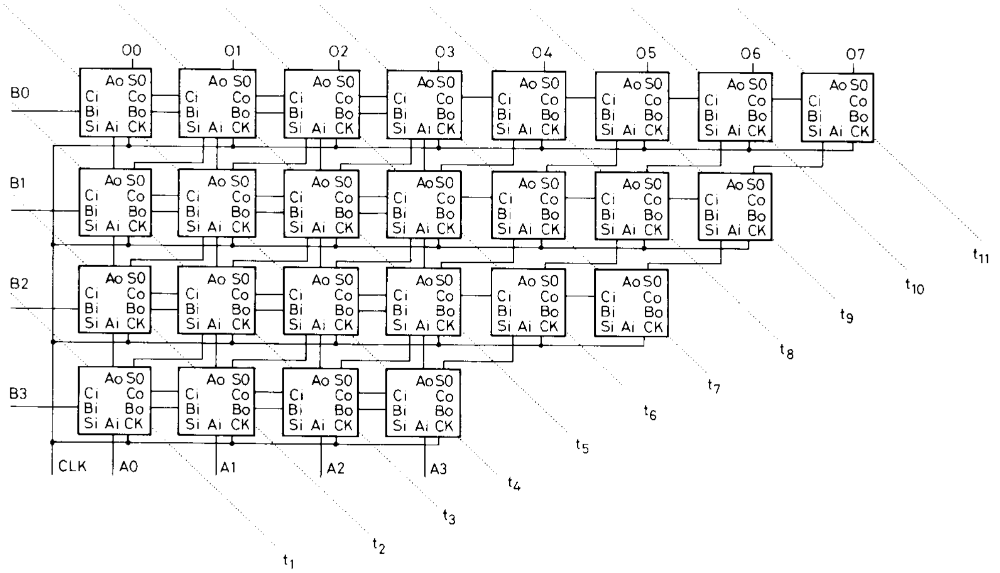

Fig. 2 *Full systolic binary multiplier architecture*

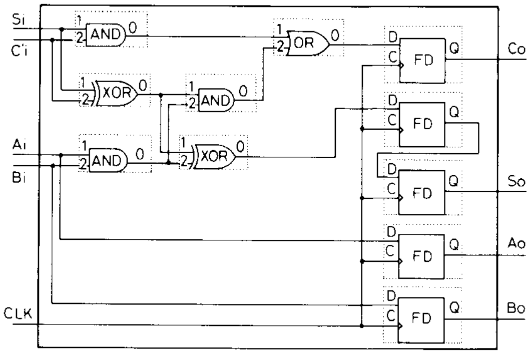

Fig. 3 *Elemental processor*

For an n-bit multiplier, the parameters that characterise this architecture are

Number of EPs:

$$NPE(n) = \frac{3n^2 + n}{2} - 1$$

Number of additional registers for a parallel I/O:

$$NR(n) = 3n^2 - 2n$$

Finally, for a parallel I/O, the number of clock cycles for the first true datum at the output after the filling the pipeline is

$$TP(n) = 3n - 1$$

After this time, there is a valid $2n$-bit word every clock cycle.

The previous scheme is used to multiply two unsigned numbers. Let us focus now on the multiplication of signed numbers. A signed number is usually represented in sign + module, one's complement or two's complement formats.

Taking into account the following properties of the multiplication:

$$\text{mod}\ (AB) = \text{mod}\ (A)\ \text{mod}\ (B)$$

$$\text{sgn}\ (AB) = \text{sgn}\ (A) \oplus \text{sgn}\ (B)$$

it can be deduced that the sign + modulus format is the best suited to adapt the described architecture for signed multiplication. Consequently, it is necessary to convert the last two formats (one's complement and two's complement) to this one. The one's complement format can be converted to sign + modulus by simply doing an XOR operation between the MSB and the rest of the bits, and keeping the MSB as the sign bit.

One way to convert the two complement format is shown on Fig. 5. However, it has the inconvenience of a slow computation because of the serial addition. A systolic solution is depicted in Fig. 6. It makes use of the chains of registers in the two operations: the conversion of two's complement to sign + module, and the time-shift of bits at the input of the multiplier.

In this situation, the modulus of the result corresponds to the multiplier outputs, and its sign is obtained from an XOR operation between input signs. If another format for the result (e.g. two complement) is required, a new conversion must be done. In this case, a new cell is needed, operating in the following way, depending on the sign obtained:

If result is positive (sgn = '0'): result = mod (result).
If result is negative (sgn = '1'): result = 0 − mod (result).

This operation is performed by the circuit in Fig. 7.

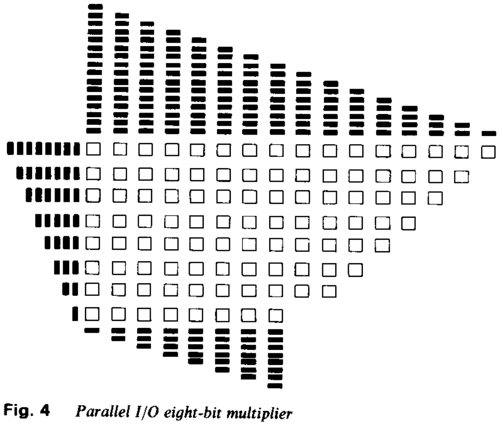

Fig. 4 *Parallel I/O eight-bit multiplier*

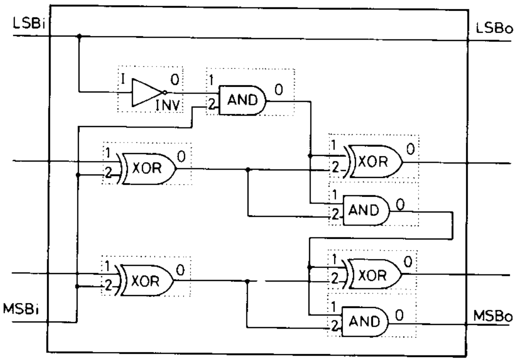

Fig. 5 *Two's complement to M + S four-bit converter*

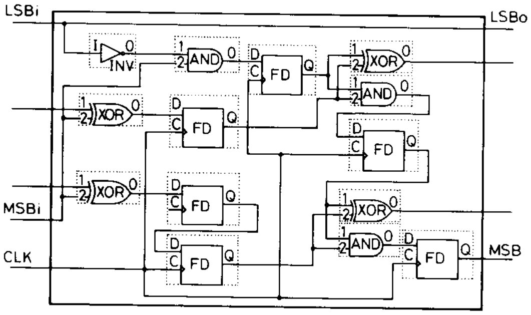

Fig. 6 *Systolic two's complement to M + S four-bit converter*

3 Architecture evaluation

The topology described above has been applied to the design of an eight-bit two's complement systolic multiplier. It has been implemented using standard cell technology of 1.5 μ from ES2 source. Some cells have been simplified to reduce silicon area by removing redundant logic. Using a two-phase clock scheme, the maximum frequency of operation of the prototypes is 40 MHz, and the silicon area used by the array multiplier is around 9 mm^2.

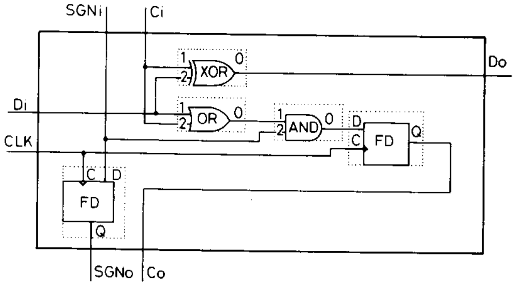

Fig. 7 *M + S to two's complement converter cell*

The relatively poor performance obtained comes from the use of automatic placement and routing strategy and standard cells, since it breaks the inherent regularity of systolic arrays. We expect to reduce the delay by redesigning the array in the CADENCE environment with manual placement strategy. A big reduction in area is also expected because of the substitution of D-registers by registers using the capacitive effect, which are composed of a pass gate plus an inverter (notice that the circuit contains more than 500 registers).

Although the use of standard cells is useful to validate the design, only a full custom implementation would exploit the potential characteristics of the architecture. To evaluate the area reduction, an eight-bit two's complement multiplier was designed in 1.5 μ single metal layer. Preliminary results shows that the area required is about 2 mm^2.

4 Acknowledgment

The programme on VLSI design in the Electronic Engineering Dept, ETSI Telecommunication, Technical University of Madrid, is supported by Eurochip, an Esprit VLSI design action of the European Community, which provided platforms, CAD tools and funded silicon samples. The authors also are grateful to J. Faura and F. Barbero for their assistance in the testing of the prototypes and to C. Carreras for suggesting improvements to this paper.

5 References

1 OBERMAN, R.M.M.: 'Digital circuits for binary arithmetic' (The Macmillan Press Ltd., 1979), pp. 125–127
2 HWANG, K.: 'Computer arithmetic. Principles, architectures and design' (John Wiley & Sons, Inc., 1979), pp. 161–209
3 SINHA, B.P., and SRIMANI, P.K.: 'Fast parallel algorithms for binary multiplication and their implementation on systolic architectures', *IEEE Trans.*, March 1989, **C-38**, (3), pp. 424–431
4 NAKAMURA, S.: 'Algorithms for iterative array multiplication', *IEEE Trans.*, Aug. 1986, **C-35**, (8), pp. 713–719
5 SHARMA, R., LOPEZ, A.D., MICHEJDA, J.A., HILLENIUS, S.J., ANDREWS, J.M., and STUDWELL, A.J.: 'A 6.75-ns 16 × 16-bit multiplier in single-level-metal CMOS technology', *IEEE J. Solid-state Circuits*, Aug. 1988, **24**, (4), pp. 922–927