

Fig. 3. Threshold SNR for at least 90% success in correct estimation of scale and orientation as a function of number of illuminations. Fifty Monte-Carlo runs were performed. The stripe requires lowest threshold SNR and the square highest. The triangle lies in between the cases.

TABLE II

COMPARISON OF CRB'S AND MEAN SQUARE ERROR (MSE) FOR DIFFERENT ANGLES OF ORIENTATION (OTHER PARAMETERS ARE  $N_L = 16$ ,  $N_R = 32$ ,  $s = 2$ , AND SNR = 15.3 dB). THE WAVELENGTH OF THE RADIATION IS ASSUMED TO BE 1 m

Orientation angle ( $\beta$ )	CRB for Scale/ $s^2$	MSE in estimated scale/ $s^2$	CRB for orientation	MSE in estimated orientation
$0^\circ$	5.7089e-06	5.9037e-06	5.0642e-05	5.1746e-05
$10^\circ$	5.8043e-06	7.0073e-06	5.5717e-05	6.4249e-05
$20^\circ$	5.6221e-06	6.0444e-06	4.4946e-05	4.9411e-05
$40^\circ$	5.5518e-06	6.5848e-06	4.5063e-05	6.1276e-05

TABLE III

COMPARISON OF CRB'S AND MEAN SQUARE ERROR (MSE) FOR DIFFERENT SCALES (OTHER PARAMETERS ARE  $N_L = 16$ ,  $N_R = 32$ ,  $\beta = 20^\circ$ , AND SNR = 15.3 dB). THE WAVELENGTH OF THE RADIATION IS ASSUMED TO BE 1 m

Scale (s)	CRB for Scale/ $s^2$	MSE in estimated scale/ $s^2$	CRB for orientation	MSE in estimated orientation
0.25	8.7134e-06	9.2214e-06	8.2711e-04	9.5376e-04
0.5	8.3348e-06	8.8571e-06	1.3278e-04	1.4008e-04
1.0	7.4335e-06	7.9803e-06	7.3486e-05	1.0521e-04
2.0	5.6221e-06	6.0445e-06	4.4946e-05	4.9411e-05
3.0	4.7221e-06	5.0600e-06	3.4231e-05	3.6429e-05

VI. DISCUSSION

The CRB's for scale and orientation increase with wavelength. To explain this phenomenon, recall that as the wavelength increases, the size of the disc in the Fourier plane decreases, and hence, only a low-frequency part of the Fourier transform is available for shape estimation. The effect of orientation of the object has maximum effect in the high-frequency region, particularly for an equidimensional object such as a square. Therefore, for reliable estimation, the Fourier transform of the object over as large a disk as possible must be used. Conversely, the objects such as a strip or triangle having a significant angular variation of the spectrum are more likely to be correctly estimated. Thus, for a perfectly equidimensional object, such as a circle, it is not possible to estimate the orientation at all. For small objects (a fraction wavelength), the estimation error is higher because most of the high-frequency spectrum falls outside the disc.

REFERENCES

- [1] K. Iwata and R. Nagata, "Calculation of three dimensional refractive index distribution from interferograms," *J. Opt. Soc. Amer.*, vol. 60, pp. 133-135, 1970.
- [2] A. J. Devaney and G. A. Tsihirintzis, "Maximum likelihood estimation of object location in diffraction tomography," *IEEE Trans. Signal Processing*, vol. 39, pp. 672-682, 1991.
- [3] G. A. Tsihirintzis and A. J. Devaney, "Maximum likelihood estimation of object location in diffraction tomography, Part II; strongly scattering objects," *IEEE Trans. Signal Processing*, vol. 39, pp. 1466-1470, 1991.
- [4] —, "Application of a maximum likelihood estimator in an experimental study in ultrasonic diffraction tomography," *IEEE Trans. Med. Imag.*, vol. 12, pp. 545-554, 1993.
- [5] D. J. Rossi and A. S. Willsky, "Reconstruction from projections based on detection and estimation of objects—Parts I & II," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 886-906, 1984.
- [6] —, "ML estimation of object size and orientation from projection data," in *Proc. IEEE ICASSP*, 1984, vol. 3.
- [7] P. S. Naidu, A. Vasuki, P. Sathya Murthy, and L. Anand, "Diffraction tomographic imaging with a circular array," *IEEE Trans. Ultrason. Ferroelect. Freq. Contr.*, vol. 42, pp. 787-789, 1995.
- [8] A. J. Devaney and G. Beylkin, "Diffraction tomography using arbitrary transmitter and receiver surfaces," *Ultrason. Imag.*, vol. 6, pp. 181-193, 1984.
- [9] M. Abramowitz and L. Stegun, Eds., *Handbook of Mathematical Functions*. New York: NBS, 1964.
- [10] W. H. Press, "Numerical recipes in Fortran," *The Art of Scientific Computing*. Cambridge, U.K.: Cambridge Univ. Press, 1992.
- [11] H. L. Van Trees, *Detection, Estimation and Modulation Theory, Part I*. New York: Wiley, 1968.
- [12] A. Schatzberg, A. J. Devaney, and A. J. Wittens, "Estimating target location from scattered field data," *Signal Process.*, vol. 40, pp. 227-237, 1994.
- [13] D. J. Rossi, A. S. Willsky, and D. M. Spielman, "Object shape estimation from tomographic measurements—A performance analysis," *Signal Process.*, vol. 18, no. 1, pp. 63-87, 1989.

A Two-Level Interleaving Architecture for Serial Convolver

Francescomaria Marino

**Abstract**—In this correspondence, we present a bit-serial architecture for convolving/correlating long numerical sequences by long filter functions. Because of its two-level interleaving structure, the proposed device does not require "wait cycles" between consecutive input samples. As a result, it achieves the highest possible throughput. Cascadability, fault tolerance, feasibility in VLSI technology, and computing performances are discussed and analyzed.

**Index Terms**—Bit serial ASICs, convolvers/correctors, pipelined architectures.

I. INTRODUCTION

In this correspondence, we present a bit-serial VLSI architecture for the direct convolution/correlation of long numerical sequences by

Manuscript received March 9, 1998; revised October 1, 1998. The associate editor coordinating the review of this paper and approving it for publication was Dr. Konstantinos Konstantinides.

The author is with the Dipartimento di Elettrotecnica ed Elettronica, Facoltà di Ingegneria, Politecnico di Bari, Bari, Italy (e-mail: marino@poliba.it).

Publisher Item Identifier S 1053-587X(99)03249-3.

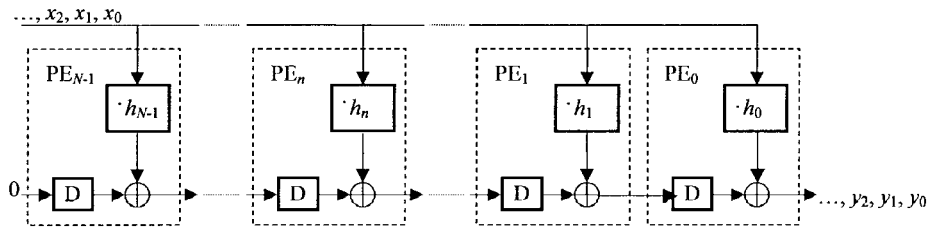


Fig. 1. Pipeline architecture. Word-serial input convolver.

long filter functions. Such an architecture is a highly modular pipeline structured in two levels of interleaving that avoid the need for “wait cycles” between consecutive input samples. As a result, the proposed device can work “on-the-fly,” i.e., its processing can proceed at the same frequency as data input access, achieving the highest possible throughput.

Even though, from the computational point of view, indirect approaches (such as the FFT [1]) are less complex than the direct approaches, they need the previous acquisition of the whole input sequence (from now on composed by  $M$  samples) when only a single point of the output sequence is required. As a consequence, indirect techniques are not practicable when an “on-the-fly” computation is desired. Moreover, although the direct convolution requires a computational complexity growing with  $M^2$ , it has greater simplicity and smaller sensitivity to error propagation than the indirect approach, especially when long filters are used [2]. Therefore, direct convolution notably contributes to hardware simplification and to overall performance.

Because of the real-time requirement of many applications needing convolution, a large number of convolver/correlator architectures have been proposed. Among these, the most suitable for VLSI implementation are those designed to exploit the bit-serial approach. In fact, this approach has many advantages with respect to the parallel one [3], such as a simpler communication strategy (single wires instead of data-buses), a reduced number of pins, and the possibility of achieving very high throughput by pipelining at the bit level. Such advantages increase when long filters have to be realized.

The removal (total or partial) of “wait cycles” between two consecutive input samples is a key for increasing the achievable throughput in bit-serial convolution and correlation. Such a task has been addressed in some architectures [4]–[7], which exploit particular processing elements (PE’s) schemes [6], double adder circuits [4], [6], or automatic word rounding [7]. All of these schemes are fully systolic and require a skewed parallel input of  $N$  words (where  $N$  is the number of filter taps). As a consequence, even though they are powerful for applications with small values of  $N$  (e.g., vector quantization for speech and image coding, where  $N$  is the dimension of the vectors [8]), because of the pin-limitation, they are absolutely not suitable for those applications where long filter functions are prescribed. Word-serial input schemes are the only practicable solutions for such applications since they need only one input line for any value of  $N$  and  $M$ .

To the best of our knowledge, the only bit-serial device that is able to convolve a word-serial input without the need for “wait cycles” is the Dadda’s polyphase convolver [9]. Polyphase convolvers require at least three phases, and therefore, they need three multipliers for each tap of the filter. Such multipliers allow the processing of input streams without “wait cycles,” but they are underutilized because one or more of them are idle during each clock cycle.

In the next section, we describe an architecture that uses only two multipliers per tap. These multipliers are fully utilized and allow

the highest possible throughput in each part of the architecture. As a result, the used VLSI area is optimized. The proposed device produces results without loss of precision and operates at the same frequency as the sampled bit-serial input, up to the maximum allowed by the implementation technology. The pipelining allows easy cascading, fault tolerance, and potential wafer scale integration. Theoretical comparative evaluation in terms of computing performance and hardware complexity is given in Section III, and some simulation results for a particular implementation technology are reported in Section IV. A conclusive remark is given in Section V.

## II. THE PROPOSED ARCHITECTURE

The aperiodic convolution of two numerical sequences (real or complex)  $\mathbf{x} = [x_0, \dots, x_{M-1}]$  and  $\mathbf{h} = [h_0, \dots, h_{N-1}]$  is the sequence  $\mathbf{y} = [y_0, \dots, y_{M+N-2}]$ , whose elements  $y_m$  are

$$y_m = \sum_{n=0}^{N-1} h_n \cdot x_{m-n}. \quad (1)$$

The reference processor structure that we used for the on-the-fly computation of (1) is the pipeline shown in Fig. 1 since, among many alternatives, this requires only one input line (for any  $M$  and  $N$ ) and provides the shortest latency. It requires a preloading phase of the filter function coefficients, but this does not restrict the computing performances since the filter function needs infrequent updating during processing. Moreover, because of the trivial relation between convolution and correlation,<sup>1</sup> the same device can be also used for computing a correlation (it is sufficient to preload  $\mathbf{h}$  in the inverted order).

According to this scheme, at the  $m$ th step, the  $n$ th processing element (PE) computes the product  $h_n x_m$  and sums this datum with the term  $(h_{n+1}x_{m-1} + (h_{n+2}x_{m-2} + (h_{n+3}x_{m-3} + (\dots))))$  that was computed during the  $(m-1)$ th step by the  $(n+1)$ th PE. The first output from the device ( $h_0 x_0$ ) is immediately available while  $x_0$  is input, and partial products are generated. No carry propagation occurs in this architecture since the sum computation is distributed.

In this scheme, we have introduced two levels of interleaving (IL’s) in order to allow a correct format expansion that occurs during the products and the sums required by (1).<sup>2</sup>

### A. First Interleaving Level (Product)

An analogy between multiplication and convolution algorithms suggests that the same architecture that is used for the convolver could be used at a lower level to implement the multipliers for on-the-fly product computation. This is possible when the bits are serially processed.

<sup>1</sup>The correlation of two numerical sequences  $\mathbf{x}$  and  $\mathbf{h}$  is the sequence  $\mathbf{y}$ , where  $y_m = \sum_{n=0}^{N-1} h_n \cdot x_{m+n}$ .

<sup>2</sup>If  $q_x$  and  $q_h$  are the number of bits used to quantize, respectively,  $\mathbf{x}$  and  $\mathbf{h}$ , each element of  $\mathbf{y}$  might require up to  $q_x + q_h + \lceil \log_2(N) \rceil$  bits.

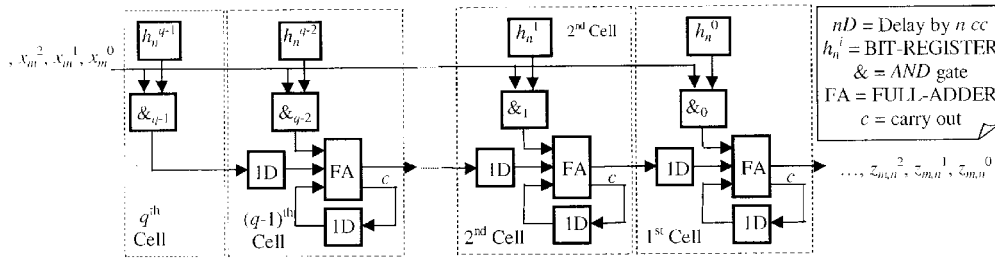


Fig. 2. Pipe-line architecture. Serial-parallel multiplier. Such a device computes  $z_{m,n} = x_m h_n$ , where  $h_n$  is available in parallel and  $x_m$  is bit-serially fed. Superscript index  $i$  denotes the  $i$ th bit, and 0 denotes the least significant bit.  $q$  has been used to denote  $q_h$ .

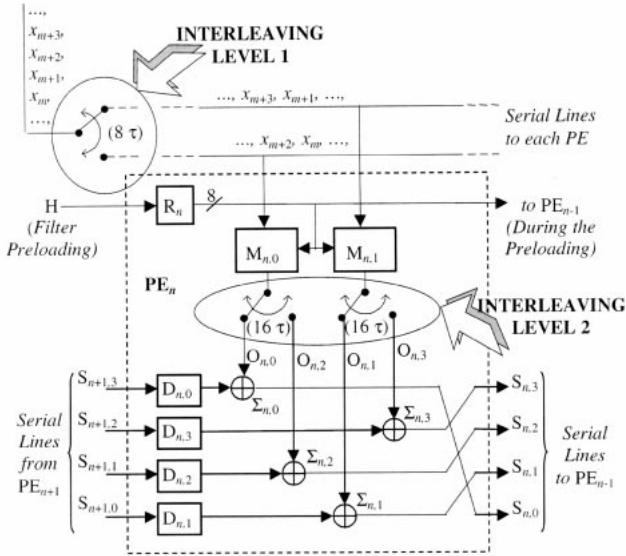


Fig. 3. IL 1 and IL 2 in  $PE_n$ .

We adopted the serial/parallel multiplier shown in Fig. 2. Such a device is appropriate because in our scheme, the input  $x$  is serially fed, and the filter coefficients can be stored in parallel once and for all (in a preloading step) since they remain constant during the whole process. The bit serialization does not introduce processing delay, and the generation of the final result requires exactly the minimum number of computing steps. Moreover, in terms of hardware complexity, this multiplier is simpler than a fully parallel one (e.g., the one described in [10]) because the number of logic ports is reduced by a factor of approximately  $q_h$ .

In the following, we will refer to digital convolvers using 8-b quantization both for  $x$  and for  $h$  (i.e.,  $q_h = q_x = 8$ ). Extension of the model to the case of more precision is straightforward.

In this case, each term  $h_n x_{m-n}$  may require up to 16 bits. Therefore, eight additional processing steps (eight steps were performed during the bit-serial input of  $x_{m-n}$ ) are needed to output the complete result. This additional delay is a bottleneck to performing on-the-fly convolution. To solve this problem, a pair of multipliers ( $M_{i,0}$  and  $M_{i,1}$ ) has been used in each  $PE_i$ , where both are controlled by an interleaving unit (Fig. 3, IL 1).

A multiplier receives only one datum  $x_m$  (over two) of the bit-serial input stream  $x$ ; in Fig. 4, the axis A shows the data input to  $M_{i,0}$  (upper side) and to  $M_{i,1}$  (lower side) in each  $PE_i$ . As a result, exactly 16 clock cycles (from now on,  $cc$ 's) are given to each multiplier to correctly compute the product and to output the result. This time slice separates the input of the first bit of a datum (i.e., the LSB of  $x_m$ ) from the input of the first bit of the next datum to process

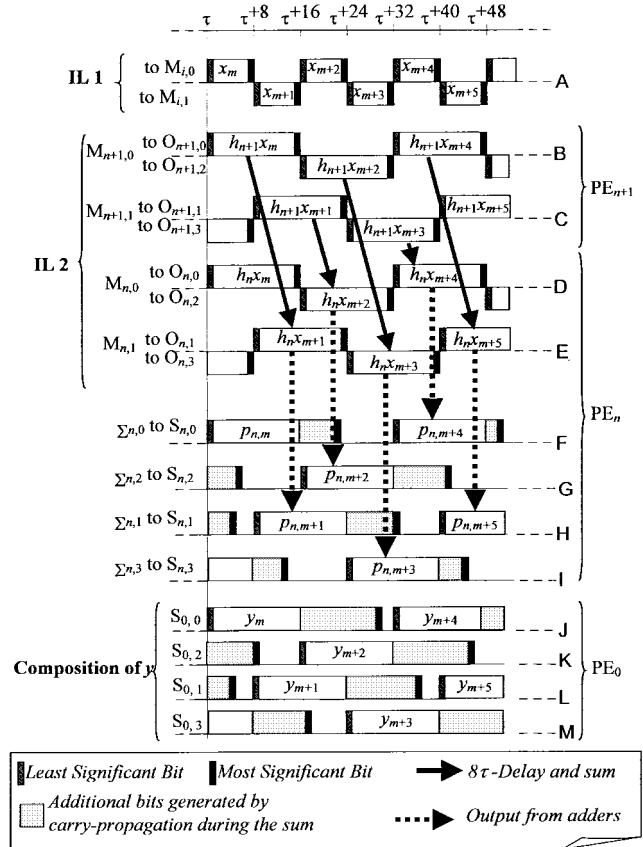


Fig. 4. Temporal diagram.

(i.e., the LSB of  $x_{m+2}$ ). This is represented on the axes B, C, D, and E, which are related, respectively, to  $M_{n+1,0}$ ,  $M_{n+1,1}$ ,  $M_{n,0}$ , and  $M_{n,1}$ .

We shall show that by introducing a second IL, which is constituted by two full adders in cascade for each multiplier, we also solve the problems caused by carry propagation in the sum chain, and we achieve the highest possible throughput in each part of the whole architecture.

### B. Second Interleaving Level (Sum)

To focus on the problems resulting from carry propagation, we consider one of the two multipliers  $M_{n,j}$ . If we assume only one adder circuit in cascade for such a multiplier at a given time, this adder should be used by the 16 bits of  $h_n x_{2k+j}$  and immediately after by the bits of  $h_n x_{2(k+1)+j}$  (these values have to be added with data coming from  $PE_{n+1}$ ).

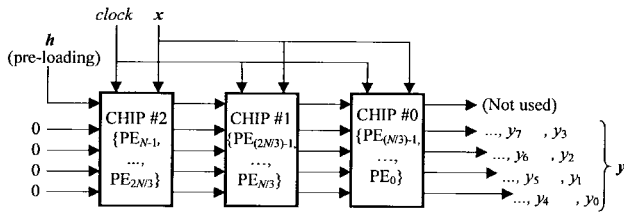


Fig. 5. Connectivity of  $C$  chips, each one having  $N/C$  PE's. In this example,  $C = 3$ .

Now, if we denote  $p_{k,L}$  as the output from  $PE_k$  after the input of  $L + 1$  elements of the sequence  $x$

$$p_{k,L} = \sum_{i=0}^L h_{k+i-L} x_{L-i} \quad (2)$$

after the addition of  $h_n x_{2k+j}$  with  $p_{n+1,2k+j-1}$ , the most significant part of such a sum will conflict with the least significant part of the following sum in the adder output. This is shown in Fig. 4, where the data on the axes F and G are partially overlapped concerning the additional bits generated by carry propagation.

To solve this problem, we have provided each multiplier  $M_{n,j}$  with a pair of output lines  $O_{n,j}$  and  $O_{n,j+2}$ , where each one is input to a single adder  $\sum_{n,j}$  and  $\sum_{n,j+2}$  (Fig. 3, IL 2). Such adders are realized simply by a FULL-ADDER with the carry suitably fed back. The axes B, C, D and E in Fig. 4 show how the results are interleaved on the output lines by each multiplier. In particular, each axis shows the data that are output by  $O_{n,j}$  (upper side) and by  $O_{n,j+2}$  (lower side).

In this way, there are 16 void  $cc$ 's between the MSB of  $h_n x_m$  and the LSB of  $h_n x_{m+2}$  on each channel  $O_{n,j}$ . Therefore, each adder may produce a result having up to 16 additional bits (one bit for each void  $cc$ ). This is shown on the axes F, G, H and I, where each one represents the output of a different adder.

Fig. 3 also shows how the output lines of each adder  $\sum_{n+1,j}$  must be delayed and connected to the adders in  $PE_n$  to correctly compute (1). These relationships are also noted in Fig. 4 by solid arrows.

Negative terms may be easily managed (e.g., as proposed in [11]) if they are presented in a two's complement form.

### C. Cascadability

As a consequence of the introduced two interleaving levels,  $2^{16}$  terms can be correctly added in the case of  $q_h = q_x = 8$ . This means that filters having  $2^{16}$  taps can be implemented by means of the proposed architecture. We would like to remark that this limit can be achieved independently by the integration, which means by cascading a suitable number of chips, since the architecture is fully modular. Such a possibility is shown in Fig. 5. An  $N$ -tap filter can be straightforwardly realized by means of  $C$  chips, where each one integrates  $N/C$  PE's.

### D. Fault Tolerance

The proposed architecture is a pipeline having identical stages (PE's). Therefore, fault tolerance capability can be added at the expenses of a little hardware overhead. A fault tolerant device is composed by  $N + \nu$  PE's, where  $\nu$  is the maximum number of tolerable faults. Such PE's are provided with one register storing one single bit  $w_i$ , which drives five 2:1 demultiplexers, as shown in Fig. 6. Such demultiplexers are used to configure the "logical" connectivity of the pipeline according to a status-word  $\mathbf{w} (= w_0 w_1 w_2 \dots w_{N+\nu-1})$ , which maps the "health" of the device. The status word  $\mathbf{w}$  is

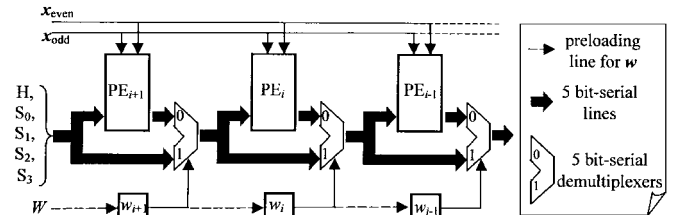


Fig. 6. Fault-tolerant connectivity.

determined by end-of-production testing.<sup>3</sup> Each bit  $w_i$  is set to 1 if  $PE_i$  is faulty. By this way, the data flow will bypass all the faulty PE's during processing. The preloading of  $\mathbf{w}$  must precede the preloading of  $\mathbf{h}$  in the initialization phase.

## III. COMPARATIVE EVALUATION

In this section, we evaluate the proposed two-level interleaving architecture (from now on,  $2L$ ) in terms of throughput  $T(2L)$  and hardware complexity  $C(2L)$ . In order to do comparative evaluations,  $T(\cdot)$  and  $C(\cdot)$  will be estimated in terms of  $[cc's]^{-1}$  and the number of transistors since such measures are independent from the implementation technology. In the next section, we shall present the "actual values" of silicon area and operative frequency that have been derived for a specific implementation of  $2L$ .

### A. Throughput

Because of its interleaving structure,  $2L$  does not require any "wait cycles" between two consecutive bit-serial input samples. As a consequence,  $2L$  can receive in input a new datum every  $q_x cc$ 's, and it can achieve a throughput  $T(2L)$ , which is the maximum achievable by a bit-serial architecture

$$T(2L) = 1/q_x (cc's)^{-1} = T_{\max}. \quad (3)$$

This result is reported in row #8 of Table I and represents an impressive speedup with respect to the conventional bit-serial architectures that achieve a throughput (row #0 in Table I)

$$T(\text{Conventional}) = 1/(q_x + q_h + \lceil \log_2(N) \rceil) \quad (4)$$

since they require  $q_h + \lceil \log_2(N) \rceil$  "wait cycles" between two consecutive input data in order to allow the format expansion in  $y_n$ .

In Table I, we also compare  $2L$  with other architectures which remove (totally or partially) the bottleneck of the "wait cycles." Such architectures are properly described in [4]–[7] and [9] and are briefly commented on by means of some remarks in Table I.

Architectures #1–4 require only  $\lceil \log_2(N) \rceil$  "wait cycles" between two consecutive data quantized by  $q_x$  bits, and therefore, they can achieve a throughput

$$T(\cdot) = 1/(q_x + \lceil \log_2(N) \rceil) \quad (5)$$

<sup>3</sup>Such a test is required only for the devices that were detected as faulty. Since it has to distinguish fault-free and faulty PE's, it has a complexity of  $O(N + \nu)$ . Briefly, it consists of a) verifying a fault-free connectivity (a faulty connectivity implies an unusable array). This is accomplished setting  $w_i = 1$  for each  $i$  (all the PE's are bypassed) and transmitting a set of test vectors  $\{\mathbf{h}, \mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3\}$  across the lines  $H, S_0, S_1, S_2,$  and  $S_3$ . We then have b) to characterize  $PE_j$  (for each  $j = 0, 1, \dots, N + \nu - 1$ ). This is accomplished setting  $w_j = 0$  and  $w_i = 1$  (for each  $i \neq j$ ). In this way, only  $PE_j$  is enabled. Afterwards,  $PE_j$  is tested by a set of test vectors  $\{\mathbf{x}, \mathbf{h}, \mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3\}$ , where  $\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3$  are addends that simulate the data that should be computed by  $PE_k$  ( $k > j$ ) during normal processing. The test does not need additional pins since in a), the correct transmission of the test vectors is verified through the pins used for connecting more chips. Even though run-time faults cannot be detected during the computation, periodical tests can make the device robust to these kind of faults by simply updating  $\mathbf{w}$ .

TABLE I

THROUGHPUT FOR DIFFERENT BIT-SERIAL CONVOLVERS:  $q_x$  AND  $q_h$  ARE THE QUANTIZATION LEVEL OF  $\mathbf{x}$  AND  $\mathbf{h}$ , RESPECTIVELY. REMARKS: *a*) WORD-PARALLEL INPUT MODE ( $N$  INPUT PINS). *b*) WORD-SERIAL INPUT MODE (ONE INPUT PIN). *c*) AUTOMATIC ROUNDING (LOSS OF PRECISION). *d*) FULL PRECISION

#	Name	Reference	Throughput [ $cc's^{-1}$ ]	Remarks
#0	Conventional Architectures	---	$1/(q_x + q_h + \lceil \log_2(N) \rceil)$	---
#1	Unidirectional ARIPA	[7]	$1/(q_x + \lceil \log_2(N) \rceil)$	<i>a, c</i>
#2	Orthogonal ARIPA	[7]	$1/(q_x + \lceil \log_2(N) \rceil)$	<i>a, c</i>
#3	Systolic Matrix-Vector Multiplier	[4]	$1/(q_x + \lceil \log_2(N) \rceil)$	<i>a, d</i>
#4	Systolic Inner Product Convolver	[6]	$1/(q_x + \lceil \log_2(N) \rceil)$	<i>a, d</i>
#5	Modified Unidirectional ARIPA	[7]	$1/q_x$	<i>a, c</i>
#6	Optimised Systolic Array for Convolution	[5]	$1/q_x$	<i>a, d</i>
#7	Polyphase Convolver	[9]	$1/q_x$	<i>b, d</i>
#8	Two-Level Interleaving Convolver	---	$1/q_x$	<i>b, d</i>

### Hardware Complexity

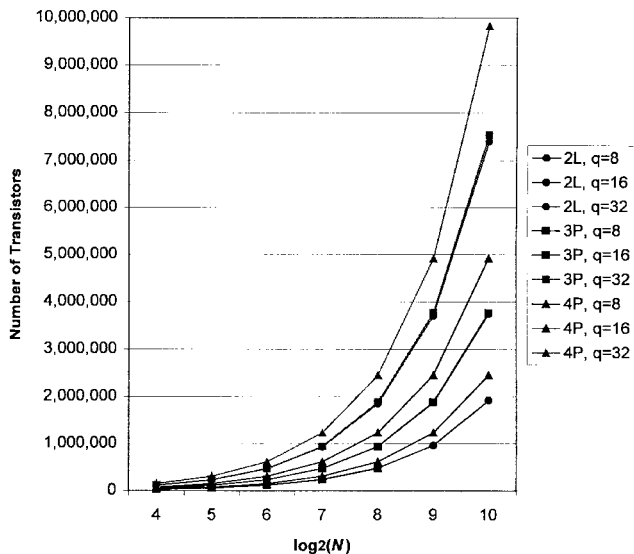


Fig. 7. Hardware complexity (evaluated in terms of number of transistors) for the proposed two-level interleaving architecture (2L), a three-phase convolver (3P) and a four-phase convolver (4P) [9] for different values of  $N$  and  $q = q_x = q_h$ . Note that  $\{3P, q = 8\}$  cannot implement filters having  $N > 2^8$  taps.

which is lower than  $T_{max}$ . Note that architectures #1 and 2 achieve this target by means of automatic word rounding (remark *c*), and therefore, they are not suitable for applications requiring high precision, whereas architectures #3 and 4 do not have such a limitation (remark *d*).

Architectures #5–7, as well as the proposed one, are able to completely remove the need for “wait-cycles.” Therefore, they can achieve  $T_{max}$  as well. Note that architecture #5 uses automatic word rounding (remark *c*), whereas architectures #6 and 7, as well as 2L, compute results with full precision (remark *d*).

### B. Hardware Complexity

We would like to point out that all the architectures considered in Table I (except 2L and architecture #7) require a skewed parallel input of  $N$  words (remark *a*). As a consequence, they require  $N$

TABLE II  
NUMBER OF TRANSISTORS PER STANDARD CELL  
(0.7- $\mu$ m CMOS, ES2, TECHNOLOGY [12])

2-input And	D-Flip Flop	Full Adder	Bit Register
7	16	31	20

input pins, and therefore, they are absolutely not suitable for those applications that require long filter functions (e.g., high-precision radars) because of the pin limitation.

The only practicable solutions for those applications appear to be 2L and the polyphase convolvers (architecture #7) since such bit-serial schemes require only one word-serial input (remark *b*). As a consequence, such architectures need a global VLSI area that can be reasonably estimated by the number of transistors (say,  $N_\tau$ ) since they use only one input pin for any value of  $N$  and  $M$  and have only few bit-serial data lines, which are not local since they are semi-systolic. Conversely,  $N_\tau$  gives only a partial and rough estimation of the global VLSI area needed by architectures #1–6. In fact, in such architectures, the VLSI area needed by the necessary  $N$  input pins has the same order of magnitude of  $N_\tau$  (i.e.,  $O(N)$ ). At the current integration levels, such VLSI area becomes preponderant (or at least it cannot be neglected) with respect to the VLSI area needed by the transistors.<sup>4</sup>

For such a reason, in Fig. 7, we have estimated  $C(\cdot)$  by means of the number of transistors  $N_\tau$  only for 2L and for two different polyphase convolvers, namely, a three-phase convolver (3P) and a four-phase convolver (4P) [9]. In order to compute  $N_\tau$ , we have assumed an implementation based on standard cells having the same characteristics as those reported in Table II.

In the diagram shown in Fig. 7, we have considered different values of  $N$  and  $q = q_x = q_h$  (for the sake of simplicity, we assumed the same quantization level for  $\mathbf{x}$  and for  $\mathbf{h}$ ). Conversely,  $M$  has not been taken into account since  $C(\cdot)$  is independent from it for all the architectures. Note that even though  $C(2L)$  and  $C(3P)$  are almost equivalent, 2L can implement filters having up to  $2^{2q}$  taps, whereas 3P cannot implement filters having more than  $2^q$  taps [9]. For such filters, a polyphase convolver requires at least four phases (4P) and a complexity  $C(4P)$  that is approximately 32% higher than  $C(2L)$ .

<sup>4</sup>A detailed evaluation of  $N_\tau$  for architectures #1–6 in Table I can be found in [7].

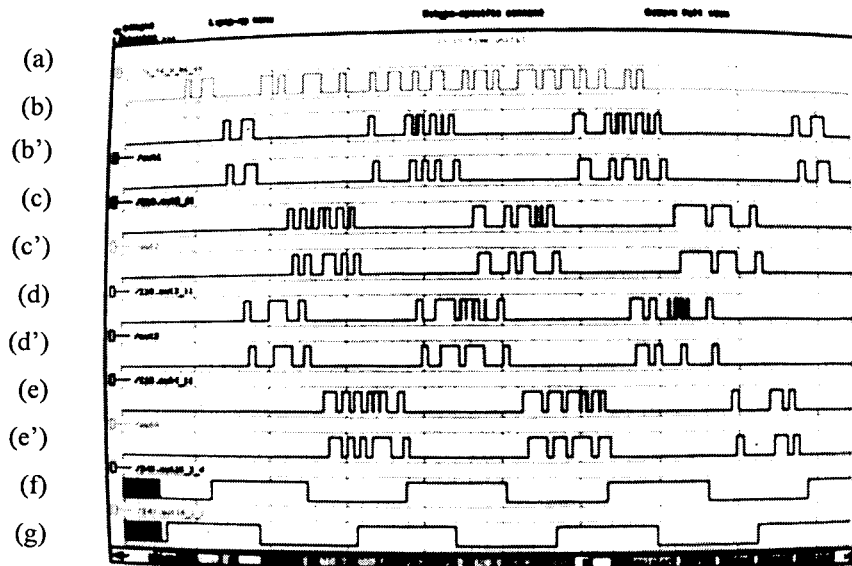


Fig. 8. Logical simulation.  $N = 300$ ; tested frequency = 50 MHz. In order to avoid aliases due to the capacitive effects and to reduce the length of critical paths (e.g., for the line feeding  $x$ ) latches have been inserted in the pipe every 50 PE's. (a) Bit serial input data stream ( $q = 8$ ); it does not require "wait cycles." (b)–(e) Bit serial output data streams probed before the latches. (b')–(e') bit serial output data streams probed after the latches. (f) and (g) control signals for IL 2. They have a semiperiod of 16  $cc$ 's.

#### IV. SIMULATION RESULTS

In order to compute the "abstract values" of silicon area and operative frequency for a specific technology, the above-derived  $C(\cdot)$  and  $T(\cdot)$  have to be multiplied by the average area needed by a transistor and by the inverse of one period of clock, respectively. Note that due to some physical phenomena (e.g., load capacitances, fan-in, and fan-out, etc.), "actual values" may be different from the derived "abstract values." Such "actual values," as well as the silicon area needed by the interconnecting wires, can be known only by means of a detailed analysis of the implementation. For such a purpose, an implementation of  $2L$  in standard cells ( $0.7\text{-}\mu\text{m}$  CMOS, ES2 technology [12]) was evaluated.

The complete placement and routing for 300 PE's ( $q_h = 8$ ) in an  $11.6 \times 11.6\text{ mm}^2$  chip was automatically obtained using the software tool SOLO2030 developed by CADENCE. An operating frequency of 50 MHz was verified through a logical simulation performed by SILOS, in which standard cell characteristic data were used (Fig. 8). Such a result means that an 8-b quantized sequence could be bit-serially convolved by a long filter function at 6.25 Msamples/s without loss of precision.

#### V. CONCLUSION

We have described a bit-serial VLSI architecture for the on-the-fly convolution of numerical sequences with long filter functions. Such a device operates without loss of precision, achieving complete hardware exploitation; in addition, it can be fully pipelined. The obtained throughput is the highest possible one since no "wait cycles" are required between consecutive input samples. Its feasibility in VLSI has been verified. Moreover, a comparative evaluation with other existing architectures in terms of throughput and hardware complexity has been provided.

#### ACKNOWLEDGMENT

The author wishes to thank V. Piuri, Polytechnic of Milan, for suggestions that have improved this correspondence. He would also like to acknowledge the anonymous reviewers for their helpful comments.

#### REFERENCES

- [1] A. V. Oppenheim and R. W. Schaffer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975.
- [2] A. V. Oppenheim and C. J. Weinstein, "Effects of finite register length in digital filtering and the fast Fourier transform," *Proc. IEEE*, vol. 60, pp. 957–976, 1972.
- [3] P. Denyer and D. Renshaw, *VLSI Signal Processing: A Bit Serial Approach*. Reading, MA: Addison-Wesley, 1985.
- [4] R. B. Urquhart and K. Wood, "Systolic matrix and vector multiplication methods for signal processing," *Proc. Inst. Elect. Eng. F*, vol. 131, no. 6, pp. 623–631, Oct. 1984.
- [5] J. V. McCanny, J. C. McWhirter, and K. Wood, "Optimized bit level systolic array for convolution," *Proc. Inst. Elect. Eng. F*, vol. 131, no. 6, pp. 632–637, Oct. 1984.
- [6] R. A. Evans and R. Eames, "Modified bit-level systolic inner product/convolver architecture with increased throughput," *Electron. Lett.*, vol. 23, no. 9, pp. 460–461, 1987.
- [7] M. Yan and J. V. McCanny, "Systolic inner product arrays with automatic word rounding," *J. VLSI Signal Process.*, vol. 4, pp. 227–242, 1992.
- [8] M. Yan, J. V. McCanny, and Y. Hu, "VLSI architectures for vector quantization," *J. VLSI Signal Process.*, vol. 10, pp. 5–23, 1995.
- [9] L. Dadda, "A polyphase architecture for serial-input convolvers," *J. VLSI Signal Process.*, vol. 2, pp. 17–27, 1990.
- [10] M. Hatamian and G. L. Cash, "A 70-MHz 8-bit  $\times$  8-bit parallel pipelined multiplier in 2.5 mm CMOS," *IEEE J. Solid-State Circuits*, vol. SC-21, pp. 505–513, 1986.
- [11] L. Dadda, "Fast multipliers for two's complement numbers in serial form," in *Proc. 7th IEEE Symp. Comput. Arith.*, 1985.
- [12] *ES2 ECPD07 Library Databook*, ES2 Cadence Design Kit, ver. 2.1, Euro. Silicon Structures, 1993.