AN ARCHITECTURE FOR EFFICIENT TIME DOMAIN SAR PROCESSING

G. Franceschetti [1],[2], A. Mazzeo [3], N. Mazzocca [3], V. Pascazio [4], G. Schirinzi [1]

[1] IRECE-CNR - via Diocleziano, 328 - 80124 Napoli (Italy)
[2] Department of Electronic Engineering - University of Naples - via Claudio, 21 - 80125 Napoli (Italy)
[3] Department of Computer Science - University of Naples - via Claudio, 21 - 80125 Napoli (Italy)
[4] I.T.T.O.E.M. - Istituto Universitario Navale - via Acton, 38 - 80133 Napoli (Italy)

ABSTRACT

We present a system for time-domain processing of Synthetic Aperture Radar (SAR) data coded at one bit. The architecture scheme is described and its time performance evaluation model is presented. The model allows design of a real time processing system taylored for a specific mission.

KEY WORDS: Synthetic Aperture Radar (SAR), Time Domain Convolution, Real Time SAR Processing.

1. INTRODUCTION

As well known Synthetic Aperture Radar (SAR) is a pulsed radar system carried over a spacecraft or an aircraft [1] emitting, and then receiving pulses every $T_C=1/PRF$ seconds, where PRF is the pulse repetition frequency .

A first problem to be solved is that SAR imaging requires heavy processing of received data. The processing consists essentially of a two dimensional (2D) space varying [2] convolution that can be performed either in frequency (via FFT techniques) or in time domain. Almost all SAR data processors operate in frequency domain [1-4] because in this way it is easier to take into account some effects, and then to correct them, without affecting processing time performances.

Today, great interest has assumed the design of real time SAR data processors. Recently, a new time domain processor [5] has been introduced; it is based on the fundamental hypotesis that the image obtained by considering only the signum of the two signals to be processed (data and filter) is equal, but for a scaling factor, to the image obtained starting from the conventionally quantized signals [6]. These two signals are referred to as signum coded (SC) sugnals and are coded at one bit , (assigning "1" to 1 and "0" to -1): all operations involved in the processing reduce to manipulations of binary signals and can be efficiently performed in time domain.

In this paper time domain 2D convolution between the binary matrixes obtained after SC operation is considered, as well as the pertinent algorithm, and its architectural implementation; in addition time performance evaluation, for the case study of SAR-X/SIR-C mission, is presented. Possibility to achieve real time processing is shown, by using castom made VLSI and ASIC chips.

2. SIGNUM CODED-SAR PROCESSING IMPLEMENTATION

The purpose of SAR processing is the precise extimation of the ground reflectivity pattern $\gamma(x,r)$ of the scene illuminated from the SAR antenna, where (x,r) are the ground coordinates. Unfortunately [2], it is necessary the use of a (x,r) ground coordinates dependent filter, so that SAR data processing is performed by means of a *space-varing* 2D convolution between the received signal data $U(x',r')$ and the system unit response $G(x',r';x,r)$, where (x',r') are the "on board" azimuth and range coordinates. The dependence on the range coordinate r is much stronger than the dependence on the azimuth one x, so that the last one can be omitted. Then, for precision SAR

processing the filter must be adjusted during the flight path and during the vertical mutual shift between the ground and the platform, to take into account this dependence. For the moment we neglect this dependence that does not affect the elaboration system design, and assume that processing reduces to a simple 2D convolution.

If we consider the signals obtained by taking the signum of the two signals involved in the convolution:

$$V(x',r')=\mathrm{sgn}\{Re[U(x',r')]\} + j\,\mathrm{sgn}\{Im[U(x',r')]\}, \quad (1.a)$$

$$H(x',r')=\mathrm{sgn}\{Re[G(x',r')]\} + j\,\mathrm{sgn}\{Im[G(x',r')]\}, \quad (1.b)$$

we obtain an image $I(x',r')$ that is essentially a replica of the conventional one, $I_C(x',r')$, but for a scaling factor α depending on the variance of the additive noise present in the received signal $U(x',r')$ [6]:

$$I(x',r')=\alpha I_C(x',r'). \quad (2)$$

After sampling and quantization of the two dimensional signals $H(\cdot)$ and $V(\cdot)$, the image matrix can be expressed by the two dimensional (2D) discrete convolution:

$$\mathbf{I}(m,n)=\sum_{p=1}^{P}\sum_{q=1}^{Q}\mathbf{H}(p,q)\,\mathbf{V}(m-p,n-q), \quad (3)$$

where the signal and filter matrixes are formed by NxM and PxQ binary samples respectively.

The evaluation of a single output value given in Eqn. (3) requires computation of $P \cdot Q$ products and $P \cdot Q$ summations. This number of elementary operation can assume a very high value. Then it can be convenient to decompose the the 2D-summation (3) in $K_1 \cdot K_2$ sets, by subdividing the overlapped part of the matrixes in the same number of blocks, as shown in Fig. 1. The lower indexes of each set are given by:

$$p_i=(i-1)\frac{P}{K_1}+1, \qquad q_j=(j-1)\frac{P}{K_2}+1. \quad (4)$$

The partial contribution of the ij^{th} block to the computation of the single image point is given by:

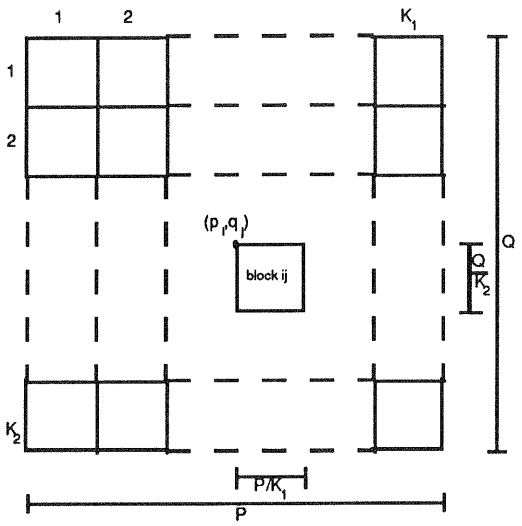

Fig. 1: *Matrix decomposition*

$$\mathbf{I}_{ij}(m,n)=\sum_{p=p_i}^{(p_{i+1}-1)}\sum_{q=q_j}^{(q_{j+1}-1)}\mathbf{H}(p,q)\,\mathbf{V}(m-p,n-q) \quad (5)$$

From Eqns. (3-5) we get:

$$\mathbf{I}(m,n)=\sum_{i=1}^{K_1}\sum_{j=1}^{K_2}\mathbf{I}_{ij}(m,n). \quad (6)$$

Hence, computation of a single image point first requires the computation of the terms $\mathbf{I}_{ij}(m,n)$ given by Eqn. (5), and then reduces to the evaluation of their summation given by Eqn. (6). Parallel elaboration is possible due to the totally independent evaluation of the terms given by Eqn. (5). The processing scheme is shown in Fig. 2.

Let us now consider Eqn. (5): it requires the computation of $P \cdot Q / K_1 \cdot K_2$ products, tsame number of summations and mutual shifting between the two binary sequences. The last operation can be easily performed if the two matrixes involved in the convolution are stored in two sets of P/K_1 Shift Registers (SR's) of dimension Q/K_2, as shown in Fig. 3.

In the case of binary elements, products reduce to EX-OR logic operations and summations reduce to evaluation of the difference between the number of bits "true" and the number of bit "false", resulting from of the EX-OR operations [7]. A functional scheme of the circuit for this evaluation is shown in Fig. 4.

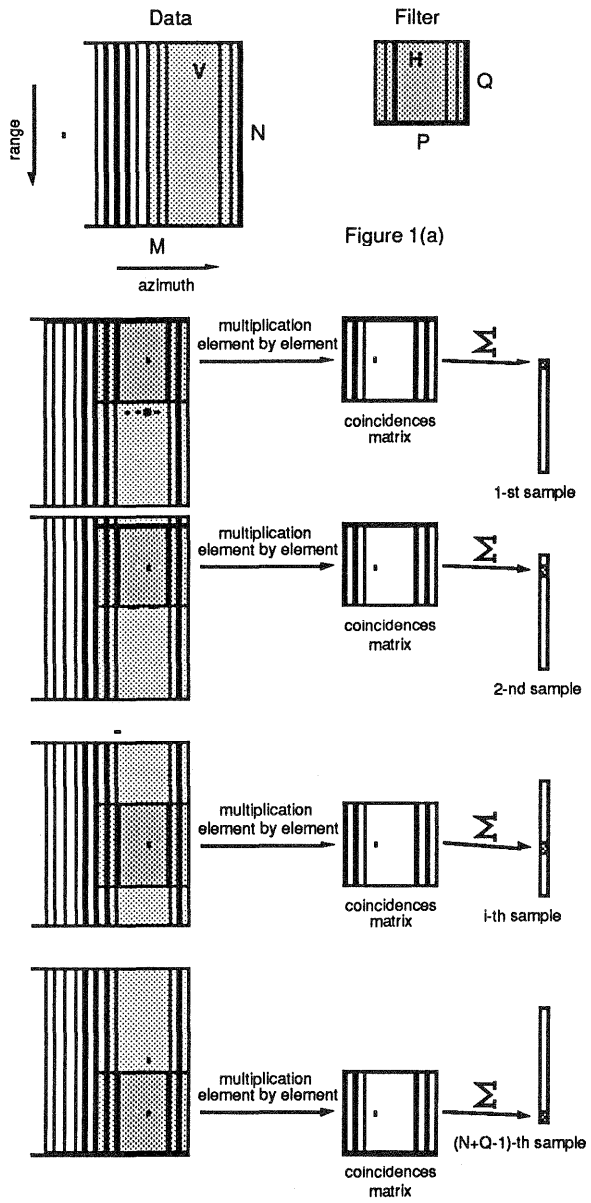

Fig. 2: *Processing scheme*

In the left part of the figure the two (ideal) vectors **A** and **B** of dimension P/K_1 are shown; their binary elements are inputs of P/K_1 EX-OR's. Results of this operation generate another (ideal) vector **C** of dimension equal to P/K_1. The purpose of the block positioned in the right part of the figure is the evaluation of the number of these coincidences. We will refer in the following to the number of coincidences evaluation network as CN.

The evaluation of the number of coincidences is equivalent to the summation of the input values. This can be performed by means of a cascade of adders. But, due to the large amount of data to be processed, the adders cascade would result too complicated and not efficient. Then, a transcoding network using counters and memory chips could be much more efficient and with very limited dimension.

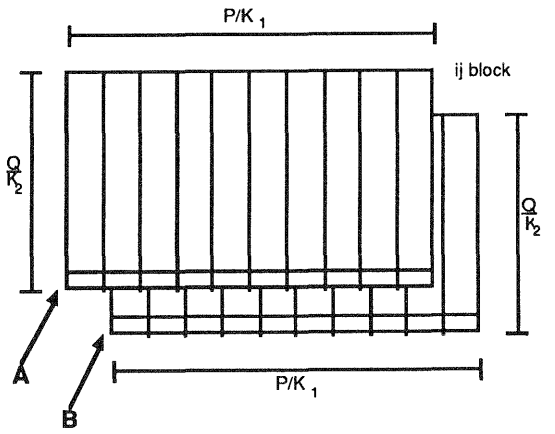

Fig. 3: *Shift register structure*

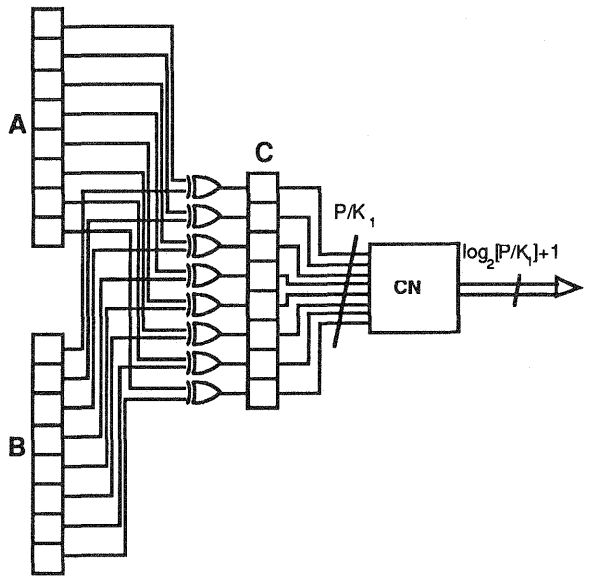

Fig.4: *Coincidence evaluation: functional scheme*

In the last case the P/K_1 input lines of the CN can be viewed as the address of memory locations containing a numerical value equal to the number of "true" bits minus the number of "false" bits present in the address. The dimension of the address bus estabilishes two parameters of the memory. The first one is the memory dimension, equal to $2^{P/K_1}$; the second one is the maximum numerical value that can be present in a memory location, i.e. the maximum number of coincidences equal to P/K_1. This estabilishes the dimension of the output bus, that is formed by $(\log_2[P/K_1]+1)$ lines.

A comment is now in order. The number of input lines could cause an unacceptable growth of the used memory dimension. For instance, if P/K_1=32 the dimension of the memory is equal to 2^{32}=4 Gb! A solution to this problem is the division of the input bus in L groups, so that each group, constituted by $P/(K_1L)$ lines, is the input of a memory device. In this way, we need L memory devices instead of a single one, but their dimension is $2^{(P/K_1-L)}$ times smaller than the dimension of the single one. This subdivision implies use of a two stage network (Fig. 5), since the outputs of L memory groups need to be summed up. This procedure can be repeated up to the point that the number of output lines of the memory stage allows to use an adders cascade in an efficient way.

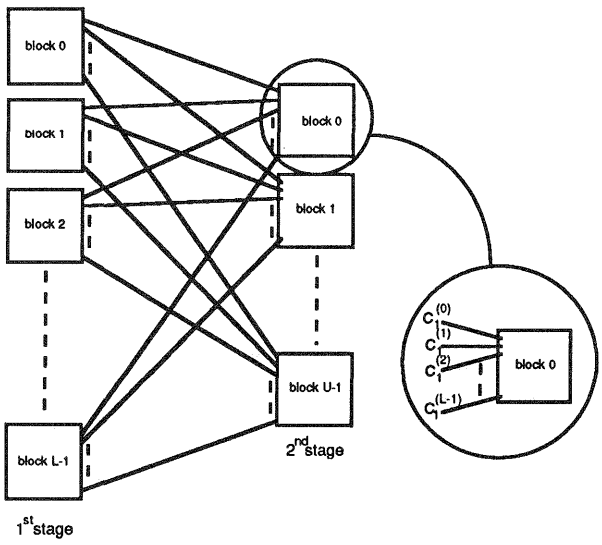

Fig.5: *two stages coincidence network*

Regarding the first stage of memory, use of counters in place of memory devices could be advisable: if the first input bus is divided in L groups, the maximum numerical value stored in the memory locations is decreased of 1/L. Use of counters instead of memory chips provides a less redundant use of devices, but renders the single computation sequential.

In conclusion, the total CN can be realized by means of a multi-stages network. The first stage is constituted by counters, the stages from the second to the last but one by memory devices, and the last by adders.

Design and performances of this multi-stages network are discussed in the next section.

3. ARCHITECTURE PERFORMANCE EVALUATION

In this section we develope a simple model for the time performance evaluation of the architecture introduced in previous sections. We recall that the SC-data to be processed and the SC-values of the filter are stored in SR's. The time domain convolution consists of evaluation of the coincidences between the omologous element of the two SC-matrixes for each mutual position.

First of all, we suppose that the working frequency of the SR's is high enough, so that the time required for one shift does not affect the total time architecture performance. Furthermore, the architecture is constituted by a pipe of stages. It is well known that when in this case the required time is given by the single stage largest time. Hence, our analysis consists of evaluation of these elementary times.

We begin with the time of the 1^{st} stage of EX-OR's shown in Fig. 4. We name t_X the elementary time required to pass through one EX-OR device. The output values of this first stage are the input to the CN. They first pass through the counters stage, then through the memories stages, and finally through the adders stage. The time needed to pass through the counter stage is:

$$t_C = \frac{P}{P/K_1L} \cdot \frac{1}{f_c}, \qquad (7)$$

where $P/(K_1L)$ is the number of input values to be counted, f_C is the working frequency of the counter and is the input frequency of the $P/(K_1L)$values.

Suppose now to have N_M stages, each one characterized by a time $t_{M,i}$ equal to the cycle time of the memory; the greatest of these values is t_M.

Finally, the time t_A necessary to pass through the adders stage, has to be considered. It depends on the number of output bits of the last memory stage and, obviously, by the kind of used adders.

The time required to process a single row of the submatrixes (block ij), is given by:

$$t_P = \max\{t_X, t_C, t_M, t_A\}. \qquad (8)$$

For the evaluation of a single output point, we remember now that the $K_1 \cdot K_2$ blocks are constituted by Q/K_2 rows and P/K_1

coloumns: for this reason each one is stored in P/K_1 SR's of lenght Q/K_2. The coincidences evaluation, as above mentioned, is row by row and is performed by means of P/K_1 EX-OR's. For the entire submatrix, the total time needed to parallel processing the partial sums given by Eqn. (5) is given by:

$$T_P = t_P \cdot \frac{Q}{K_2} \qquad (9)$$

The last operation to be performed is the sum given by Eqn. (6); this operation can be executed by adders. The total time T required for a single output point is clearly given by summing these two partial times.

4. TIME DOMAIN SAR PROCESSING

The purpose of the described architecture is to achieve real time processing of the SC data. Real time elaboration of SAR data is feasible if the time needed for processing each output image columns does not exceed the time interval T_d between two successive data acquisitions. Hence, an output column must be processed, and then stored, or displayed, in T_d seconds, before a new radar echo return is received. If M and N are the data azimuth and range dimensions, respectively, and P and Q are the azimuth and range filter dimensions, respectively, the processing of an output image coloumn is obtained by means of the convolution between the whole filter matrix and a piece of the data matrix of dimension P (along the azimuth) and N (along the range). If we consider, the values of the SIR-C mission, i.e. N=5400, P=355, Q=900 samples, and PRF=2000 Hz, the processing time of a complete output coloumn must be lower than $T_d=1/PRF=0.5\cdot10^{-3}$ seconds, and the time for the processing of a single output sample must be lower than $T_s=T_d/(N+Q-1)= 1\ 0^{-7}$ seconds; $P\cdot Q$ multiplications and the same number of summations must be performed for the computation of each output sample.

The generation of a single output coloumn is schematically shown in Figs. 2. Different output samples are obtained for different mutual position of the two matrixes involved in the convolution, and the number of samples for each output coloumn is equal to $(N+Q-1)$. When a new input coloumn is received, its position in the new data frame to be processed is ideally at the right side of the previous data frame. The processing of the new output coloumn can now be performed after an azimuthal mutual shift of the two matrixes. Such a shift makes unnecessary the presence of the more left data column used for the computation of the previous output one: the dimension, along the azimuth coordinate, of the data to be processed is always the same, and is equal to P.

We enphasize that, for time domain processing of SAR data, a mutual two-dimensional shift between the two matrixes involved in the convolution is required. For real time processing the horizontal shifts must be performed each $T_h<T_C$ seconds while the vertical ones each $T_v<T_s=T_C/(N-Q+1)<<T_C$ seconds. The rate of the vertical shifts seem to be more critical to approach than the rate of the horizontal ones. This is partially true; but the problem is "naturally" solved because the vertical shifts are performed along the SR's in which the single SC radar echos are stored, and the working frequency of the today available SR'r easily allow shifts with this rate. More critical is a real time performance of the horizontal shifts, because these shifts are in a direction orthogonal to the SR'r. This problem can be solved as follows. When a new echo is received it should be stored in the P column of data matrix, the other P-1 column have to shift of one place in the left direction, while the column occupying the first position can be eliminated. This operation implies the shift of P-1 column of Q-samples. Alternatively, the new echo can be stored in the first column and the shifting can be performed on the filter matrix of smaller dimensions with respect to the data one. In this way the number of samples to be shifted in much smaller. Once the elaboration of an image column is completed, a new echo is received and stored in the second column of the data matrix. The procedure can be iterated for the following received echos. By taking into account the reduced amount of the data to be shifted and the maximum admitted horizontal shifting time T_h (much more greater than T_v), the two dimensional mutual shift is performed within the required times.

5. CONCLUSIONS

The described architecture is attractive inusmuch it attains real time performance with very simple devices. The large amount of data requires a prohibitive high number of elementary gates. For this reason the system hardware implementation needs an integrated logic developement. The model flexibility allows

the use of different technologies. Preliminary simulation results for SAR-X mission show that real time processing requires a system of 180 chips, each integrating 200,000 transistors.

REFERENCES

[1] C. Elachi, T. Bicknell, R.L. Jordan, C. Wu, "Spaceborne Synthetic Aperture Imaging Radars: Applications, Techniques and Technology", Proc. IEEE, **70**, 1174-1209, 1982.

[2] G. Franceschetti, G. Schirinzi, "A SAR Processor Based on Two Dimensional FFT Codes", *IEEE Trans. Aerosp. Electr. Syst.*, **AES-26**, 356-366, 1990.

[3] C. Wu, K.Y. Liu and M. Jim, "Modelling of Spaceborne Synthetic Aperture Radar Response and New Digital Processing Algorithm for Producing Imagery", *IEEE Trans. Aerosp. Electr. Syst.*, **AES-18**, 563-575, 1982.

[4] W.J. van de Lint , "Digital Technique for Generating Synthetic Aperture Images", IBM Research Develop., **21**, 415-432, 1977.

[5] G. Alberti, G. Franceschetti, V. Pascazio, G. Schirinzi, "Time Domain Convolution of One Bit Coded Radar Signals", *Proc. IEE, Pt. F,* **138**, 438-444, 1991.

[6] G. Franceschetti, V. Pascazio, G. Schirinzi, "Processing Of Signum Coded SAR Signals: Theory and Experiments", *Proc. IEE, Pt. F,* **138**, 192-198, 1991.

[7] G. Franceschetti, A Mazzeo, N. Mazzocca, V. Pascazio, G. Schirinzi, "Time domain Processing of SAR Data in Real Time", IGARSS'91, 283-286, 1991