Reconfigurable Architecture of Systolic Array Processors for Real Time Remote Sensing Image Enhancement/Reconstruction

[1,2]A. CASTILLO ATOCHE, [1]D. TORRES ROMAN, AND [1]Y. SHKVARKO
[1]Department of Telecommunications
CINVESTAV del IPN, Unidad Guadalajara
Avenida Científica # 1145, Colonia El Bajío, Zapopan, Jalisco, C.P. 45015.
MEXICO
{acastillo, dtorres, shkvarko}@gdl.cinvestav.mx
[2]Department of Mechatronics
Autonomous University of Yucatan
Av. Industrias no Contaminantes, Colonia Cordemex, Merida, Yucatan, Apdo. Postal 150.
MEXICO
acastill@uady.mx

Abstract: - In this paper, we propose a reconfigurable architecture of systolic array (SA) processors for near real time implementation of high-resolution reconstruction of remote sensing (RS) imagery. The proposed design is based on a Field Programmable Gate Array and performs the image enhancement/reconstruction tasks in an efficient reconfigurable processing architecture mode that involves the systolic array processors aimed to meet the (near) real time imaging systems requirements in spite of conventional computations. In particular, the reconfigurable architecture of SA processors is employed with the objective to decrease the computational load of the large-scale RS image enhancement/reconstruction tasks required to implement the RS enhancement/reconstruction algorithms based on the descriptive regularization techniques with the corresponding iterative fixed-point Projection Onto Convex Sets unified via the proposed Hardware/Software Co-Design paradigm.

Key-Words: - Remote sensing, Reconfigurable architecture, FPGA, Systolic array processors, Hardware/Software co-design.

1 Introduction

The newer techniques for high resolution remote sensing (RS) and radar image enhancement/reconstruction are computationally extremely expensive [1], [3]. Therefore, these techniques are not suitable for (near) real time implementation with existing digital signal processors (DSP) or personal computers (PC). The traditional architecture and model of computation of the majority of PCs and DSPs used today is still based on the *Von Neumann* architecture introduced in the late 1940s [10]. Any instruction is executed one by one in a sequential manner using one processor. Given such architecture, there exist physical limits beyond which it is very difficult to increase the computing power. The growing demands on speed and processing power of the recently developed image reconstruction RS techniques make them unacceptable to be implemented in a (near) real time. However, the use of specialized arrays of processors will become a real possibility for high speed RS applications in the nearest few years. In mid 80's, a lot of research work was undertaken to speed-up the execution of specific signal processing (SP) computationally intensive algorithms with the use of systolic arrays (SAs) architectures [2], [4], [10]. Such SA architecture consists of a mesh of regularly connected processing elements (PEs) with local memory and local interconnection topology. Each PE in a SA executes one instruction during one clock cycle. A SA belongs to the class of special-purpose parallel architectures also called *hardware-specific* or *dedicated* [10]. The SA may be used as a coprocessor with an embedded processor inside a FPGA where the data received from the embedded processor pass through the PEs and the final result is returned to the embedded processor. This interesting architecture design approach with the SA coprocessor and the embedded processor corresponds to the celebrated FPGA-based Hardware/Software (HW/SW) co-design. However, one crucial issue in the HW/SW co-design is the integration of the customized user cores (SA coprocessors) with the embedded processor (SW) due the intense data exchange of the large amount of data involved in such type of RS operations. Furthermore, the principal innovative proposition that distinguishes the proposed here approach from the previous studies [1], [3], [5]−[8], [17]−[19] consists of a Field Programmable System on Chip (FPSoC) implementation of the RS image enhancement/reconstruction tasks with the new reconfigurable architecture of SA processors. In this study, the RS tasks correspond to the descriptive experiment design regularization (DEDR) method that involves a convergence enforcing regularization based on the iterative fixed-point Projection Onto Convex Sets (POCS-regularization). The latter is performed in context of the (near) real time computing pursuing the Hardware/Software Co-Design paradigm.

Finally, we report and discuss the implementation and performance issues related to (near) real time enhancement of the large-scale real-world RS imagery indicative of the significantly increased processing efficiency gained with the developed reconfigurable architecture of SA processors.

2 Background

In this section, we present a brief summary of the DEDR-POCS-regularization method that was recently developed in [3], [8]. Hence, some crucial model descriptions are repeated for convenience to the reader. Consider a coherent RS experiment in a random medium and the narrowband SAR assumption [11] that enables us to model the extended object backscattered wavefield in the baseband format [12] by imposing its time invariant complex scattering (backscattering) function $e(\mathbf{x})$ in the object image domain (scattering surface) $X \ni \mathbf{x}$. The measurement radar/SAR data field $u(\mathbf{y}) = s(\mathbf{y}) + n(\mathbf{y})$ consists of the echo signals s and additive noise n and is available for observations and recordings within the prescribed time-space observation domain $Y = T\times\mathrm{P}$, where $\mathbf{y} = (t, \boldsymbol{\rho})^{\mathrm{T}}$ defines the time(t)-space($\boldsymbol{\rho}$) points in Y; $t\in T$, $\boldsymbol{\rho}\in\mathrm{P}$; $\mathbf{y}\in Y$.

The model of observation wavefield u is specified by the linear stochastic equation of observation (EO) of operator form [12]: $u = Se + n$; $e\in\mathrm{E}$; $u,n\in\mathrm{U}$; S:E→U. Next, we take into account the conventional finite-dimensional vector form approximation [12] of the continuous-form EO

$$\mathbf{u} = \mathbf{S}\mathbf{e} + \mathbf{n} \tag{1}$$

where **u**, **n** and **e** define the vectors composed of the coefficients of the finite-dimensional approximation of the fields u, n and e, respectively, and **S** is the matrix-form approximation of the signal formation operator (SFO) S specified by the particular modulation format employed in the RS system [12]. The average $\mathbf{b} = vect\{< e_k, e_k^* >; k = 1,...,K\}$ of the random scattering vector **e** has a statistical meaning of the average power scattering function traditionally referred as the spatial spectrum pattern (SSP), where the asterisk indicates the complex conjugate. The SSP is a second order statistics of the scattered field that represent the brightness reflectivity of the image scene $\mathbf{B}$=$L\{\mathbf{b}\}$, represented in a conventional pixel format over the rectangular scene frame [8]. The RS imaging problem is stated as follows: to find an estimate of the scene pixel-frame image $\hat{\mathbf{B}}$ via lexicographical reordering $\hat{\mathbf{B}} = L\{\hat{\mathbf{b}}\}$ of the spatial spectrum pattern (SSP) vector estimate $\hat{\mathbf{b}}$ reconstructed from whatever available measurements of independent realizations $\{\mathbf{u}_{(j)}; j = 1, \ldots, J\}$ of the recorded data vector**.**

The DEDR-POCS-regularization method is described as follows [3], [8]:

First, to avoid the cumbersome operator inversions prescribed by the DEDR-related robust spatial filtering (RSF) and robust adaptive spatial filtering (RASF) methods [1], [3], [8] the processing techniques are modified to the iterative fixed-point procedures that define a sequence of estimates

$$\hat{\mathbf{b}}_{[i+1]} = \mathcal{T}_{[i]}\{\hat{\mathbf{b}}_{[i]}\} = \{\mathbf{K}_{[i]}\mathbf{S}^{+}\mathbf{R}_{\mathbf{n}}^{-1}\mathbf{Y}\mathbf{R}_{\mathbf{n}}^{-1}\mathbf{S}\mathbf{K}_{[i]}\}_{\mathrm{diag}}; \quad i = 0, 1, \ldots, \tag{2}$$

where $\mathbf{R}_{\mathbf{n}}^{-1} = (1/N_0)\mathbf{I}$, is the diagonal-form inverse noise correlation matrix specified by the noise power N_0, $\mathbf{Y} = \mathrm{aver}\{\mathbf{u}\mathbf{u}^{+}\}$ is the current RS correlation data matrix estimate and the nonlinear fixed-point iteration operator $\mathcal{T}_{[i]}$ is defined by the right-hand side of (2);

$$\mathbf{K}_{[i]} = \mathbf{K}(\hat{\mathbf{b}}_{[i]}) = (\boldsymbol{\Psi} + \alpha\mathbf{D}^{-1}(\hat{\mathbf{b}}_{[i]}))^{-1} \tag{3}$$

represents the adaptive reconstruction operator at the ith iteration step with the regularization term $\mathbf{D}^{-1}(\hat{\mathbf{b}}_{[i]}) = \mathrm{diag}^{-1}(\hat{\mathbf{b}}_{[i]})$ and the point spread matrix (PSM) [1]

$$\boldsymbol{\Psi} = \mathbf{S}^{+}\mathbf{R}_{\mathbf{n}}^{-1}\mathbf{S}\,. \tag{4}$$

Using (3)–(4), the (2) can be next transformed to the following iterative mapping

$$\hat{\mathbf{b}}_{[i+1]} = \mathcal{T}_{[i]}\{\hat{\mathbf{b}}_{[i]}\} = \mathbf{T}_{[i]}\hat{\mathbf{b}}_{[i]}\,;\; i = 0, 1, \dots \qquad (5)$$

with the output of the matched space filter (MSF) algorithm [3]

$$\hat{\mathbf{b}}_{[0]} = \hat{\mathbf{b}}_{(MSF)} = \{\,\mathbf{S}^{+}\mathbf{Y}\mathbf{S}\,\}_{\text{diag}} \qquad (6)$$

as the zero-step iteration and the matrix-form fixed-point iteration operator

$$\mathbf{T}_{[i]} = \mathbf{Q}_{[i]} - \mathbf{Q}_{[i]} \odot \mathbf{Q}^{*}_{[i]}\,;\; i = 0, 1, \dots \qquad (7)$$

where superscript * stands for complex conjugate, $\odot$ denotes the Shur-Hadamar (element-by-element) matrix product [11], and $\mathbf{Q}_{[i]}$ is adaptively updated at each iteration as

$$\mathbf{Q}_{[i]} = \mathbf{Q}(\hat{\mathbf{b}}_{[i]}) = \mathbf{I} - (\,\boldsymbol{\Psi} + \alpha\text{diag}^{-1}(\hat{\mathbf{b}}_{[i]}))\,;$$
$$i = 0, 1, \dots \qquad (8)$$

Note that the operator (8) does not involve operator inversions (in contrast to the initial one (2)), which are now performed in an iterative fashion (5).

Second, to modify the fixed-point algorithm (5)–(7) let us make the use of factorization of the PSM (4) over the azimuth (x) and range (y) coordinates valid for all SAR systems [1], [11], [12]. We formalize this stage by introducing the range-azimuth factorization operator $\mathcal{P}_{a\perp r}$. Next, we incorporate the sparseness properties of the PSM (4) via imposing the range-azimuth factorized projection operator $\mathcal{P}_{\kappa_a\perp\kappa_r}$ that acts as a composition of the orthogonal sliding windows with the window apertures adjusted to the PSM widths corresponding κ_a, κ_r along the range-azimuth coordinates.

The defined above orthogonal projecting window operator $\mathcal{P}_{\kappa_a\perp\kappa_r}$, the range-azimuth factorization operator $\mathcal{P}_{a\perp r}$ and the positivity operator $\mathcal{P}_{+}$ are projectors onto convex sets, i.e. POCS operators [3], [8] thus a composition

$$\mathcal{P}_{POCS} = \mathcal{P}_{+}\mathcal{P}_{\kappa_a\perp\kappa_r}\mathcal{P}_{a\perp r} \qquad (9)$$

is a POCS operator as well that constitute the necessary and sufficient conditions [8] of the convergence of the overall POCS-regularized fixed-point iterative RASF algorithm

$$\hat{\mathbf{B}}_{[i+1]} = \mathbf{T}_{POCS[i]}\,\{L\{\hat{\mathbf{b}}_{[i]}\}\};\; i = 0, 1, \dots, \qquad (10)$$

in which the zero-step iteration $\hat{\mathbf{B}}_{[0]} = L\{\hat{\mathbf{b}}_{[0]}\}$ is formed using the conventional (i.e. low-resolution) MSF imaging method (6), and the matrix-form POCS-regularized fixed-point iteration operator $\mathbf{T}_{POCS[i]}$ is specified now as [8]

$$\mathbf{T}_{POCS[i]} = \mathcal{P}_{+}\mathcal{P}_{\kappa_a\perp\kappa_r}\mathcal{P}_{a\perp r}\,\{\mathbf{Q}_{[i]} - \mathbf{Q}_{[i]} \odot \mathbf{Q}^{*}_{[i]}\};$$
$$i = 0, 1. \qquad (11)$$

The established POCS-regularized fixed-point iterative RASF algorithm (10), (11) does *not* involve the cumbersome operator inversions (in contrast to the initial one defined (2)) and, moreover, is performed separately along the range (y) and azimuth (x) directions making also an optimal use of the PSM sparseness properties.

Next, we accomplish our algorithmic developments at the SW co-design stage with the analytical analysis of the convergence issues related to the presented unified DEDR-POCS method. Following the POCS regularization formalism [8], the convergence enforcing projectors in the iterated procedure are to be constructed formally as

$$\mathcal{P}_{\iota}^{\lambda} = \mathcal{I} - \lambda_{\iota}(\mathcal{P}_{\iota} - \mathcal{I})\,;\; \iota = 1, 2, \dots\,;$$
$$\mathcal{P}_{1} = \mathcal{P}_{a\perp r}\,;\; \mathcal{P}_{2} = \mathcal{P}_{\kappa_a\perp\kappa_r}\,;\; \mathcal{P}_{3} = \mathcal{P}_{+}, \qquad (12)$$

where λ_{ι}; ι = 1, 2, 3 represent the relaxation (speeding-up) regularization parameters and $\mathcal{I}$ is the identity operator. The iteration rule (10) for the composed regularizing projectors (12) becomes

$$\hat{\mathbf{B}}_{[n+1]} = \mathcal{P}_{3}^{\lambda}\mathcal{P}_{2}^{\lambda}\mathcal{P}_{1}^{\lambda}\,\hat{\mathbf{B}}_{[0]} + \mathcal{P}_{3}^{\lambda}\mathcal{P}_{2}^{\lambda}\mathcal{P}_{1}^{\lambda}\,\{\mathbf{T}_{[n]}\hat{\mathbf{B}}_{[n]}\}\,;$$
$$i = 0, 1, \dots \qquad (13)$$

and is *guaranteed* to converge to the point in the intersection of the convex sets specified by $\mathcal{P}_{\iota}^{\lambda}$ provided $0 < \lambda_{\iota} < 2$ for all ι = 1, 2, 3 *regardless* of the initialization $\hat{\mathbf{B}}_{[0]}$ that is a direct sequence of the fundamental theorem of POCS [8]. Note that the employed specifications of the projectors in (12), i.e., $\mathcal{P}_{1} = \mathcal{P}_{a\perp r}$; $\mathcal{P}_{2} = \mathcal{P}_{\kappa_a\perp\kappa_r}$; $\mathcal{P}_{3} = \mathcal{P}_{+}$; with $\lambda_{\iota} = 1$ for all ι = 1, 2, 3, and $\hat{\mathbf{B}}_{[0]} = L\{\hat{\mathbf{b}}_{MSF}\}$, satisfy these POCS convergence conditions, in which case the formal convergent POCS procedure (13) becomes the developed above fixed-point DEDR-POCS algorithm given by (10).

Note that the formal SW-level of such DEDR-POCS-regularization method (7), (8), (10), (11), can be considered as a properly ordered sequence of the vector-matrix and matrix-matrix multiplication procedures that one next can perform in an efficient computational fashion following the proposed

HW/SW co-design paradigm. Now we are ready to proceed with the design stage of the proposed reconfigurable architecture of SAs.

2.1 Matrix-Vector Multiplication

Let us consider the matrix **A** of order $n\times m$ and the vector **x** of order $m\times 1$, where **y** is an n element output vector. The *i-th* element of **y** is defined as:

$$y_i = \sum_{j=1}^{m} a_{ij} x_j, \quad \forall \ 1 \le i \le n \ \wedge \ 1 \le j \le m \qquad (14)$$

where a_{ij} represents the *ij-th* element of **A**.

To find their product $\mathbf{y} = \mathbf{Ax}$ the following piecewise regular locally recursive algorithm [2] can be used

input operations

$a[i,j] \leftarrow a_{ij} \qquad \forall \ 1 \le i \le n \ \wedge \ 1 \le j \le m$

$x[0,j] \leftarrow x_j \qquad \forall \ 1 \le j \le m$

$y[i,0] \leftarrow 0 \qquad \forall \ 1 \le i \le n$

computations

$y[i,j] \leftarrow y[i,j-1] + a[i,j] \cdot x[i,j]$

$x[i,j] \leftarrow x[i-1,j]$

output operations

$y[i] \leftarrow y[i,m] \qquad \forall \ 1 \le i \le n \ \wedge \ 1 \le j \le m$

where the index space is

$I = \{(i,j)^{\mathrm{T}} \in \mathbb{Z}^2 \mid 1 \le i \le n \ \wedge \ 1 \le j \le m\}$

2.2 Matrix-Matrix Multiplication

The product matrix **C = AB** of dimension $m\times p$ is the result of the multiplication of two matrices **A** of order $m\times n$ and **B** of order $n\times p$ that it is defined as

$$c_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}, \quad \forall \ 1 \le i \le n \ \wedge \ 1 \le j \le m \qquad (15)$$

The recurrent piecewise regular locally recursive algorithm that computes this matrix-matrix multiplication is specified as follows

input operations

$a[i,0,k] \leftarrow a_{ik} \qquad \forall \ 1 \le i \le n \ \wedge \ 1 \le k \le p$

$b[0,j,k] \leftarrow b_{jk} \qquad \forall \ 1 \le j \le m \ \wedge \ 1 \le k \le p$

$c[i,j,0] \leftarrow 0 \qquad \forall \ 1 \le i \le n \ \wedge \ 1 \le j \le m$

computations

$c[i,j,k] \leftarrow c[i,j,k-1] + a[i,j,k] \cdot b[i,j,k]$

$a[i,j,k] \leftarrow a[i,j-1,k]$

$b[i,j,k] \leftarrow b[i-1,j,k]$

output operations

$c[i,j] \leftarrow y[i,j,p] \qquad \forall \ 1 \le i \le n \ \wedge \ 1 \le j \le m$

where the index space is

$I = \{(i,j,k)^{\mathrm{T}} \in \mathbb{Z}^3 \mid 1 \le i \le n \ \wedge \ 1 \le j \le m$
$\wedge \ 1 \le k \le p\}$

Once the algorithms are represented into their locally recursive versions, the data dependencies in the computations are exposed in dependence graphs (DG) to represent the parallel characteristics of the algorithms. Now, the algorithms are ready to be transformed onto the SAs processors.

3 Algorithms transformation onto SA

In this section, we briefly introduce the bases of how to synthesize a SA from a given functional specification of a given problem. The steps involved in this will be explained using the reconstructive RS operations for the DEDR-POCS-regularization method (i.e., the matrix-vector and matrix-matrix multiplication operations).

First, in order to derive a regular systolic array [2], a linear projection is often represented by a projection vector **d**. Next, we seek for an efficient linear matrix transformation **T** such that

$$\mathbf{T} : \mathbf{G}^N \rightarrow \hat{\mathbf{G}}^{N-1} \qquad (16)$$

where an N-dimensional DG ($\mathbf{G}^N$) is mapped onto a (N-1)-dimensional systolic array ($\hat{\mathbf{G}}^{N-1}$). The linear transformation matrix **T** can be partitioned in two functions as follows

$$\mathbf{T} = \begin{bmatrix} \mathbf{\Pi} \\ \mathbf{\Sigma} \end{bmatrix} \qquad (17)$$

where $\mathbf{\Pi}$ is ($1\times p$)-dimension vector of the first row of **T**, which determines the time scheduling. This vector indicates the normal direction of the equi-temporal hyper-planes in the corresponding DG. All the nodes on the same hyper-plane must be processed at the same time. The sub-matrix $\mathbf{\Sigma}$ of ($p-1$)$\times p$ dimension (the rest rows of **T**), determines the space processor.

Second, to achieve the maximal parallelism in an algorithm, the data dependencies in the computations must be analyzed [2], [4]. The systolic design maps the N-dimensional dependence graph (DG) into a lower dimensional DG ($N-1$ dimension).

3.1 Matrix-Vector SA Implementation

Let us consider a matrix-vector product with matrix **A** of size $m \times n$ and a vector **x** of size $n \times 1$, i.e. **y** = **Ax**. Next, select the projection vector **d** = $[1\ 0]^T$ with a vector schedule **s** = $[1\ 1]^T$. The corresponding SA for implementing this product and its dynamic behavior are illustrated in Fig. 1. The pipelining period for this SA is one. The number of PEs required by this structure is n. The computational time achieved is $2n-1$ clock periods.

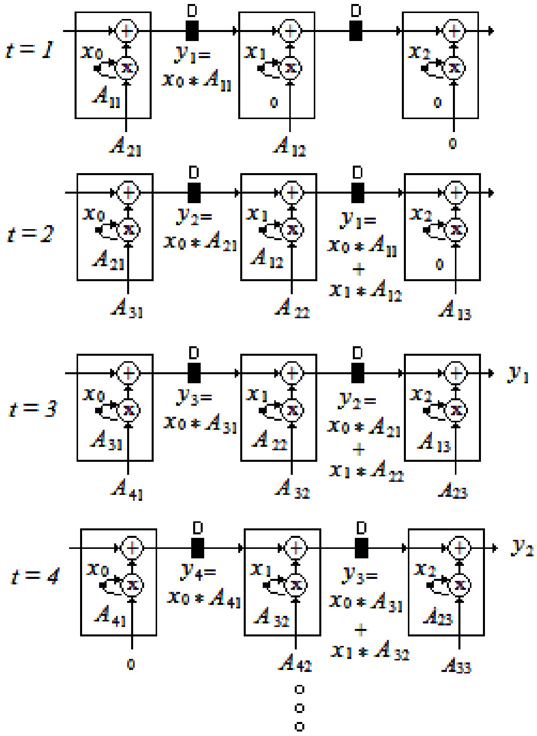


Fig. 1. Dynamic behavior of the systolic matrix-vector.

3.2 Matrix-Matrix SA Implementation

Let **A** be an $m \times n$ matrix and **B** be an $n \times k$ matrix. The product of the matrices is an $m \times k$ matrix **C=AB**. The DG of a standard matrix-matrix multiplication algorithm corresponds to a 3-D space representation. In Fig. 2, the dynamic behavior of the systolic structure for implementing such a product with the projection direction vector **d** = $[0\ 0\ 1]^T$ are presented. This architecture requires an array of mk PEs and $n+m+k-1$ clock periods.

4 Reconfigurable architecture of SA

One of the advantages of the FPGA-based embedded systems consists in their ability to integrate the customized user cores with a soft or hard embedded processor in system-on-a-chip (SoC) solutions for RS applications. Considerable improvements in the algorithm execution time are expected when such customized reconstructive SP cores are used as hardware accelerators to perform the computationally intensive RS techniques.

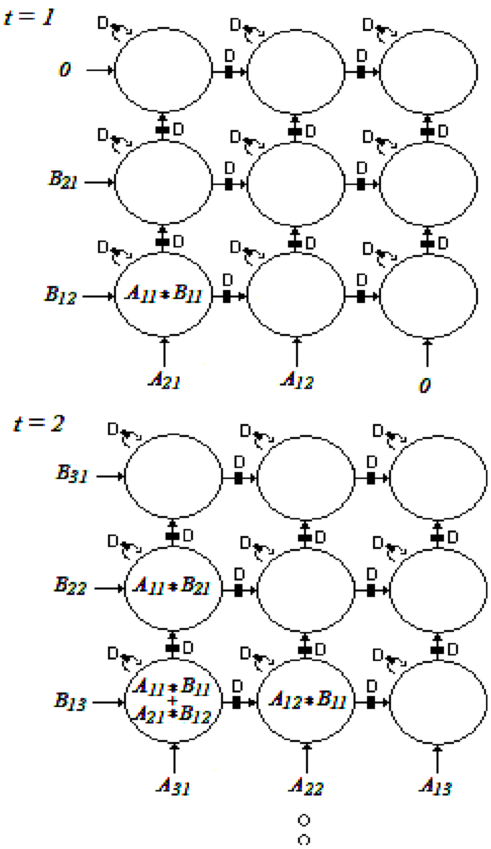


Fig. 2. Dynamic behavior of the matrix-matrix SA.

The HW/SW co-design is a hybrid method aimed at increasing the flexibility of the implementation and improvement of the overall design process. In this study, to perform the HW/SW co-design, we select the Microblaze embedded processor (for the restricted platform) and the On Chip Peripheral Bus (OPB) [9] for transferring the data from/to the embedded processor to/from the reconfigurable architecture of SA processors. Such OPB is a fully synchronous bus that connects other separate 32-bit reconfigurable architectures.

The proposed reconfigurable architecture of SAs manages the low data-bandwidth of the system input-output (I/O) transfer bus and the high bandwidth for the data exchange of the corresponding SA processors. To deal with such high data-bandwidth demanding requirements of the SA processors, we propose to incorporate memory buffers into the interface with the SAs.

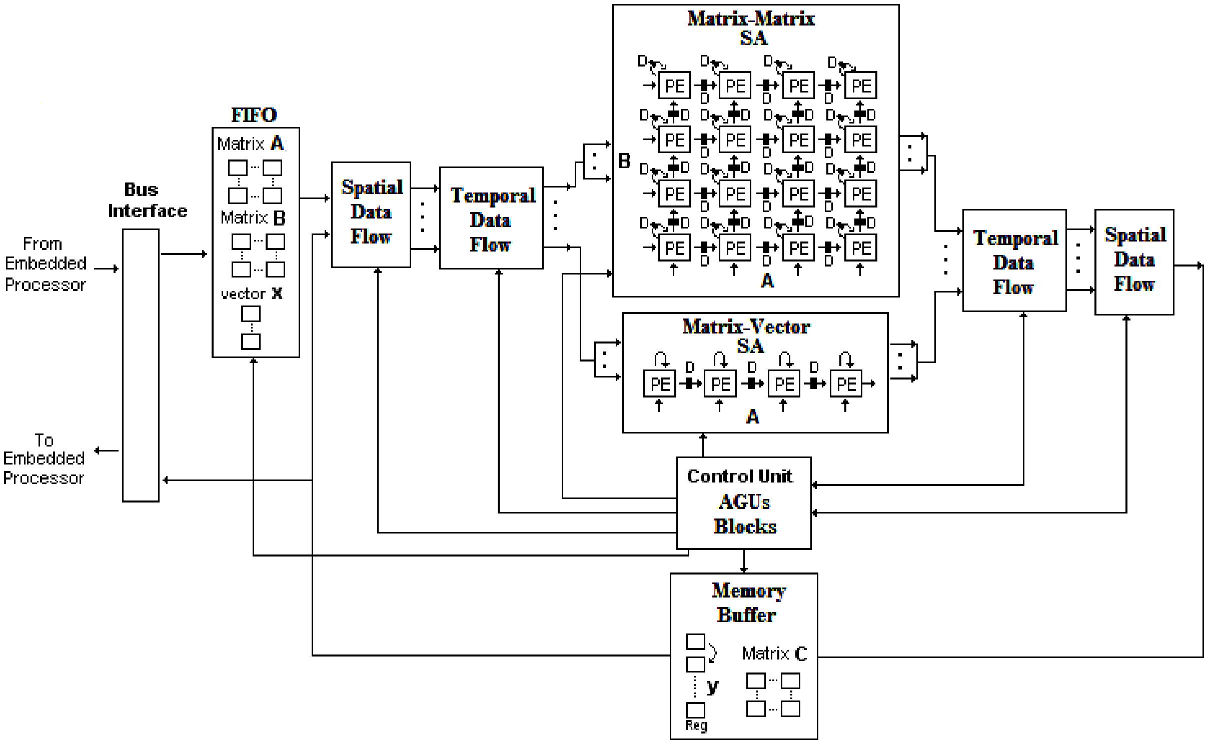


Fig. 3. Reconfigurable Architecture of SA.

Fig. 3 presents the reconfigurable architecture of the SAs for the selected matrix-matrix and matrix-vector operations. As one can deduce from the analysis of the structure presented in Fig. 3, the proposed architecture has the ability to reconfigurate the data sequence flow to the corresponding SAs in the specific time.

First, the first-input first-output (FIFO) memory receives the low bandwidth data.

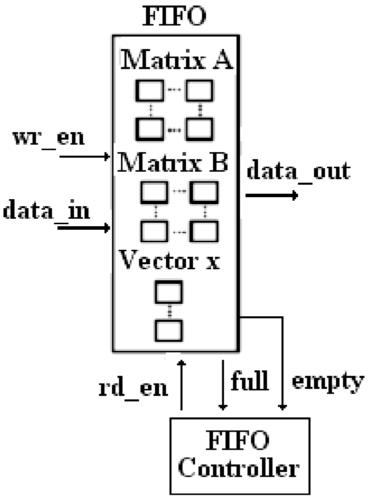


Fig. 4. Low data-bandwidth store block.

The FIFO memory stores the RS data acquisition data and the algorithm specifications, e.g., the noise correlation model, the POCS model parameters, the azimuth-range SFO, etc. that are properly ordered as a sequences of vector **x** and matrices **A** and **B** for the computations of the vector-matrix and matrix-matrix multiplication procedures as it is illustrated in Fig. 4.

Second, the spatial-temporal procedure rearranges the sequence of data into the specific order for the corresponding SAs. Two architectures blocks are defined: the spatial data flow block which is implemented by a multiplexer architecture and the temporal data flow employed by a set of Random Access memories (RAMs) as illustrated in Fig. 5.

As one can notice from Fig. 5, once the RAMs blocks are complete loaded the data are extracted in parallel (high data-bandwidth) to the corresponding SA architecture. For the purpose of the reconfigurability, the addressable generation units (AGUs) based on look up tables (LUTs) were incorporated into the architecture. With the AGUs block based on LUTs, the reconfigurable architecture is able to balance the low data-bandwidth from the system input-output (I/O) transfer bus in a high data-bandwidth for the corresponding SA processors.

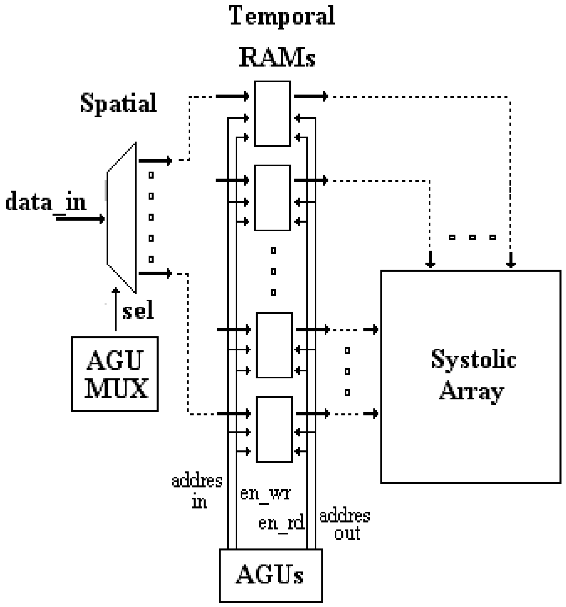


Fig. 5. Spatial-Temporal block.

Such AGUs are implemented based on the Random Access memories (RAM), so the user can load each AGU via the corresponding embedded processor as specified in the block diagram of Fig. 6.

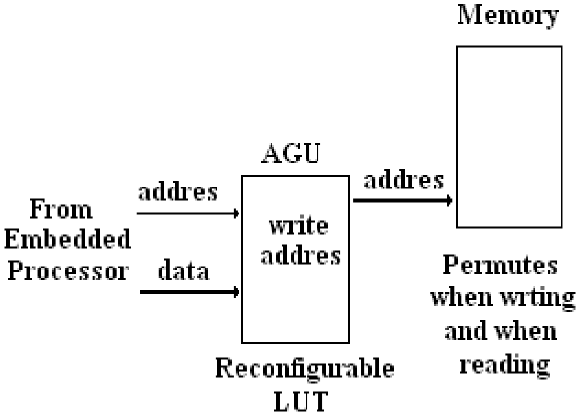


Fig. 6. AGU block based on LUT.

Next, in order to balance the high bandwidth requirements of the SA processors, memory buffers were incorporated into the reconfigurable architecture of SAs.

In summary, the developed reconfigurable architecture of SAs provides the necessary reconfigurability for the HW-level implementation of the SW-optimized complex multi-purpose RS imaging algorithms.

5 Simulations and Performance Analysis

5.1 Performance Metrics

In the simulation experiments, it is considered a conventional side-looking SAR with the fractionally synthesized aperture as an RS imaging system [11]. The regular signal formation operator (SFO) of such SAR is factored along two axes in the image plane [3]: the azimuth or cross-range coordinate (horizontal axis, x) and the slant range (vertical axis, y), respectively. The conventional triangular, $\Psi_r(y)$, and Gaussian approximation, $\Psi_a(x) = \exp(-(x)^2/a^2)$ with the adjustable fractional parameter a, are considered for the SAR range and azimuth ambiguity function (AF) [1], [3]. Note that in the imaging radar applications [1], [12], an AF is referred to as the continuous-form approximation of the PSM $\mathbf{\Psi}$ defined by (4) and serves as an equivalent to the point spread function in the conventional image processing terminology [13]. The image degradation and noising effects were incorporated to simulate the process of formation of the degraded speckle-corrupted MSF images. Following [1] the degradation in the spatial resolution due to the fractional aperture synthesis mode were simulated via blurring the original image with the range AF $\Psi_r(\Delta y)$ along the y axis and with the azimuth AF $\Psi_a(\Delta x)$ along the x axis, respectively. Next, the degradations at the image-formation level due to the propagation uncertainties were simulated using the statistical model of SAR image defocusing [3], [14].

In analogy to the image reconstruction [13], first, it is employed the quality metric defined as an improvement in the output signal-to-noise ratio (*IOSNR*)

$$IOSNR = 10\log_{10} \frac{\sum_{k=1}^{K}\left(\hat{b}_k^{(MSF)} - b_k\right)^2}{\sum_{k=1}^{K}\left(\hat{b}_k^{(p)} - b_k\right)^2}\ ; p = 1, 2 \quad (18)$$

where b_k represents the value of the kth element (pixel) of the original image $\mathbf{b}$, $\hat{b}_k^{(MSF)}$ represents the value of the kth element (pixel) of the degraded image formed applying the Matched Space Filter (MSF) technique (6), and $\hat{b}_k^{(p)}$ represents a value of the kth pixel of the image reconstructed with two developed methods, p = 1, 2, where p = 1 corresponds to the POCS-RSF algorithm and p = 2 corresponds to the POCS-RASF algorithm, respectively.

According to these quality metrics, the higher are the *IOSNR*, the better is the improvement of the image enhancement/reconstructed with the particular employed algorithm.

5.2 Simulations

In this study, the simulations were performed with a large scale (1K-by-1K) pixel format image borrowed from the real-world high-resolution terrain SAR imagery (south-west Guadalajara region, Mexico [15]). The quantitative measures of the image enhancement/reconstruction performance gains achieved with the particular employed POCS-RSF and POCS-RASF techniques, evaluated via *IOSNR* metric (18), are next reported in Table 1 and Fig. 7.

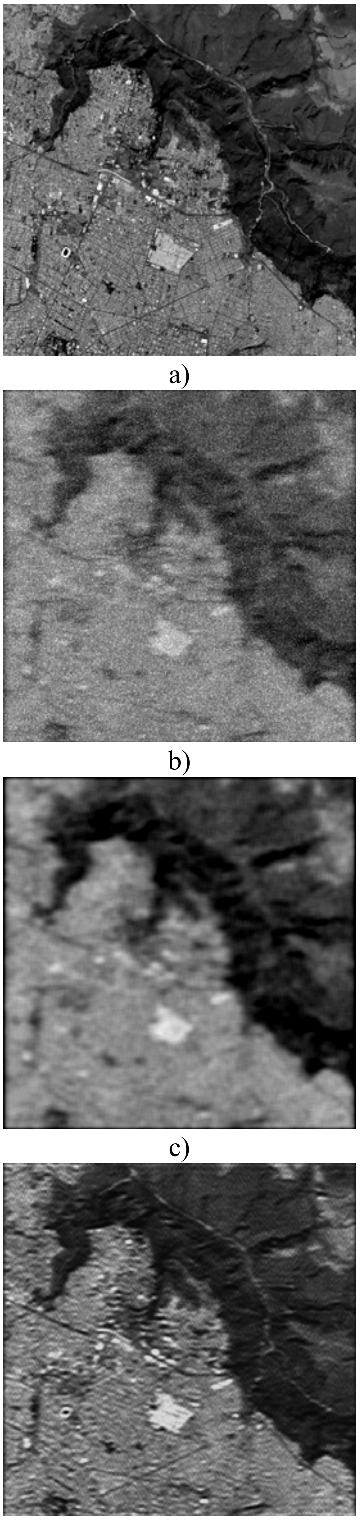
Table 1. Image enhancement of the DEDR-POCS RSF/RASF algorithms.

***SNR* [dB]**	**DEDR-POCS RSF Method**	**DEDR-POCS RASF Method**
	IOSNR [dB]	*IOSNR* [dB]
5	4.21	6.74
10	5.56	8.38
15	7.78	9.86
20	9.26	11.47

From the analysis of the qualitative and quantitative simulation results reported in Figure 6 and Table 1, one may deduce that, the RASF method over-performed the RSF method in all simulated scenarios. Moreover, the relationship between the resulting *IOSNR* quality metric and the visually reconstructed image represents the high degree of correlation between the two images (original and reconstructed image). In this study, with the POCS regularization, the appearance of the POCS-RSF/RASF reconstructed images demonstrated substantial improvement up to 10 iterations from the initial MSF image.

5.3 HW/SW Co-Design Performance Analysis

The synthesis metrics related to the implementation of the reconfigurable architecture of SAs are summarized in Table 2. These metrics specify the area and time behaviors of the corresponding interface and the hardware systolic arrays, i.e. the Matrix-Vector and the Matrix-Matrix cores. The system clock was adjusted to 100 MHz and we assumed a precision of 32 bits running in a Virtex4 XC4VS X35-10ff668.



a)

b)

c)

d)

Fig. 7. Operational scenario, scene (SNR = 15 dB): (a) original scene; (b) degraded uncertain scene image formed applying the MSF method; (c) image reconstructed applying the POCS-RSF algorithm; (d) image reconstructed employing the RASF algorithm.

The components blocks of the reconfigurable architecture of SAs are exemplified for the simple illustrative test case of the data matrix of size 4×4 and the matrix-vector of size 4×1.

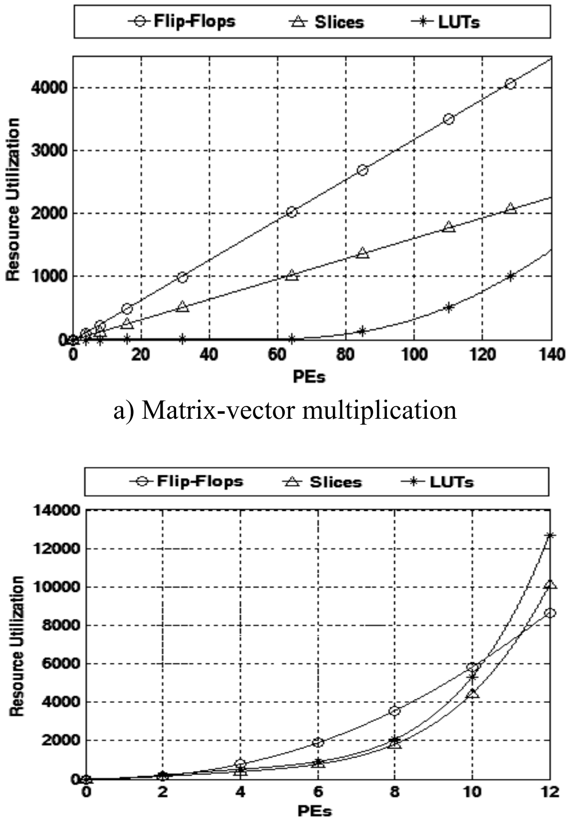
Table 2. Synthesis results of the components blocks of the proposed reconfigurable architecture of SAs.

Synthesis metrics	**Systolic matrix-matrix**	**Systolic matrix-vector**	**FIFO Block**
Number of Slices	386	48	62
Number of DSP'48	16	4	NA
Number of LUTs	513	NA	54
Number of Flip-Flops	768	96	16
Equivalent Gate Count	13,222	1024	4,676
Fmax (MHz)	115.3	210.6	215.4
Synthesis metrics	**Spatial-Temporal blocks**	**AGUs blocks**	**Buffer Memory Block**
Number of Slices	763	351	82
Number of DSP'48	NA	NA	NA
Number of LUTs	626	483	71
Number of Flip-Flops	896	326	23
Equivalent Gate Count	15,148	5,771	4,836
Fmax (MHz)	410.7	200.2	224.3

In Fig. 8, we report the resource utilization of the proposed here SAs hardware architectures (i.e., the matrix-vector and the matrix-matrix multiplication blocks) for different numbers of employed processors elements (PEs).

The proposed reconfigurable architecture represents a parallel and pipelined solution which exploits the hardware efficiency of the SAs. This proposed architecture is required to improve the time implementation performance of the complex RS algorithms. For example, in [5] it was presented the parallel computing Cluster-based algorithm for hyperspectral image processing where the achieved processing time using 64 processors was 48 seconds. To further speed-up the processing time, it is necessary to implement the corresponding high performance reconfigurable architecture based on specialized hardware modules such as proposed here reconfigurable FPGA-SAs.

In order to compare the SA matrix operations with another stand alone modules or FPGA-based modules, the relevant examples of efficient circuits for matrix operations are presented.



a) Matrix-vector multiplication

b) Matrix-matrix multiplication

Fig. 8. Resource utilization with different PEs.

In the case of the matrix-vector multiplication SA architecture, the total time for performing the multiplication of a square $n \times n$ matrix and an n-vector requires only $2n$-1 clock periods and occupies an area of 48 slices (for the test example of $n=4$) with a data precision of 32-b. An interesting alternative design of a unidirectional linear systolic array (ULSA) for computing a matrix-vector multiplication was performed in [16], however, the total time required was $3n-2$ clock cycle, i.e. superior to the proposed here solution.

With the $n \times n$ matrix-matrix multiplication systolic architecture developed in this study, the most time consuming operations required only $3n$-2 clock cycles and the area occupied 386 slices for the data precision of 32-b (e.g., considering the same $n=4$ test case). Another alternative implementation for systolic matrix-matrix multiplication was presented in [20] where the systolic matrix-matrix multiplication design occupied an area of 110 slices (i.e., data precision of 8-b) with the corresponding processing time of n^2+3n+2 clock cycles. Mencer et al. in [21] presents the matrix-matrix multiplication architecture with an area performance of 954 slices for a data precision of 8-*b*. Thus, it is evident that the proposed SAs architectures manifest the best area-time tradeoff.

Last, the required processing times for two different implementation techniques were compared as reported in Table 3. In the first case, the iterative fixed-point DEDR-POCS-regularization RSF/RASF algorithms [3], [8] were implemented in the conventional MATLAB software on a personal computer (PC) running at 1.73GHz with a Pentium (M) processor and 1GB of RAM memory, while in the second case, the same iterative fixed-point POCS-regularized algorithms were implemented using the proposed reconfigurable architecture FPGA-based via the HW/SW co-design paradigm.

Table 3.Comparative time processing analysis

Method	**Processing Time [secs]**	
	RSF	**RASF**
Iterative fixed-point DEDR-POCS-Regularized (PC implementation)	19.70	20.05
Proposed Reconfigurable Architecture of SAs for the DEDR-POCS-Regularization algorithm via HW/SW Co-design	2.34	2.41

Analyzing the reported results one may deduce that the proposed reconfigurable architecture of SAs via the HW/SW co-design for implementing the iterative fixed-point DEDR-POCS-regularized RSF/RASF image reconstruction algorithms manifests the finest (near) real time computational requirements.

6. Acknowledgment

The authors would like to thank to Dr. Manuel Guzman of Intel Corporation for his technical support in this study.

7 Conclusion

The principal result of the undertaken study relates to the design of a reconfigurable architecture of SAs. With the proposed SA-based architecture, the corresponding iterative fixed-point DEDR-POCS-regularized RSF/RASF algorithms were executed in a (near) real time computational mode (the 'near-real' being understood in a context of conventional RS users). The latter was achieved pursuing the proposed hardware/software co-design paradigm. Doing this, we performed the reconstructive post-processing of the large-scale real-world RS imagery attaining the desirable enhancement performance in a real-time mode. We do believe that by pursuing the addressed HW/SW co-design paradigm one could approach definitely the real time image processing requirements.

References:

[1] Y. V. Shkvarko, Unifying regularization and Bayesian estimation methods for enhanced imaging with remotely sensed data. Part I and II, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, 2004, pp. 932-940.

[2] S.Y. Kung, *VLSI Array Processors*, Prentice Hall, 1988.

[3] Y.V. Shkvarko, H.M. Perez and A. Castillo, Enhanced radar imaging in uncertain environment: A descriptive experiment design regularization paradigm, *Intern. Journal of Navigation and Observation*, vol. 2008, Article ID 810816, 2008, pp. 1-11.

[4] S.C. Lo and S. N. Jean, Mapping algorithms to VLSI array processors, *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, 1988, pp. 2033-2036.

[5] C. Gerardino, Y. Riviera, J. Goodman and W. Rivera, Parallel implementation of an inversion model for hyperspectral remote sensing, *49th IEEE MWSCAS*, vol. 2, 2006, pp. 606-609.

[6] P. Zicari, P. Corsonello, S. Perri and G. Cocorullo, A matrix product accelerator for field programmable systems on chip, *Journal of Mycroprocessors and Mycrosistems*, vol. 33, 2008, pp. 53-67.

[7] M. Karra, M. Bekakos, I. Milovanovic, E, Milovanovic, FPGA implementation of a unidirectional systolic array generator for matrix-vector multiplication, *IEEE ICSPC*, 2007, pp.153-156.

[8] A. Castillo Atoche, Yuriy V. Shkvarko, D. Torres, H. Pérez Meana, Convex Regularization-Based Hardware/Software Co-Design for Real-Time Enhancement of Remote Sensing Imagery, *Int. Journal of Real-Time Image Processing (IJRTIP)*, Edit. SPRINGER, Special Issue, vol. 4, num. 3, 2009, pp. 261-272. ISSN: 1861-8200.

[9] IBM Microelectronics division, On Chip Peripheral Bus, IBM, version 2.1, 2001.

[10] K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*, Edit. Wiley-Interscience, 2000.

[11] D.R., Wehner, *High-Resolution Radar*. 2nd edn. Artech House, Boston, 1994.

[12] F.M., Henderson, A.V., Lewis; *Principles and Applications of Imaging Radar. In: Manual of Remote Sensing*. 3rd edn. Wiley, New York, 1998.

[13] H.H. Barrett and K.J. Myers, *Foundations of Image Science*, New York: Willey, 2004.

[14] J. S. Lee, "Speckle suppression and analysis for synthetic aperture radar images", Optical Engineering, vol. 25, no. 5, pp. 636-643, 1986.

[15] Space Imaging. In: http://www.spaceimaging.com/quicklook. GeoEye Inc., 2008.

[16] I.Z. Milovanovic, E.I. Milovanovic, M.P. Bekakos, Synthesis of a unidirectional systolic array for matrix-vector multiplication, Mathematical and Computer Modeling, Edit. Elsevier, Volume 43, Issues 5-6, March 2006, Pages 612-619, 2006.

[17] Cheng-Hsiung Hsieh, Po-Chin Huang, Sheng-Yung Hung, Noisy Image Restoration Based on Boundary Resetting BDND and Median Filtering with Smallest Window, WSEAS TRANSACTIONS on SIGNAL PROCESSING vol. 5, Issue 1, pp.178-187, January 2009.

[18] Slami Saadi, Hamza Mekki, Abderrezak Guessoum, Object Detection and Segmentation Algorithm Implemented on a Reconfigurable Embedded Platform Based FPGA, WSEAS TRANSACTIONS on SIGNAL PROCESSING vol. 4, Issue 9, pp.552-561, September 2008.

[19] Jeng-Kuang Hwang, Yuan-Ping Li, Modular Design and Implementation of FPGA-based Tap-selective Maximum-likelihood Channel Estimator, WSEAS TRANSACTIONS on SIGNAL PROCESSING vol. 4, Issue 9, pp.552-561, September 2008.

[20] Zicari, P.; Corsonello, P.; Perri, S; Cocorullo, G.: A matrix product accelerator for field programmable systems on chip, Microprocessors and Microsystems, Volume 32, Issue 2, pp. 53-67, 2008.

[21] O. Mencer, M. Morf, M. Flynn, PAM-Blox: "High performance FPGA design for adaptive computing", in: IEEE Symposium on Field-Programmable Custom Computing Machines, Napa Valley, California, 1998.