FPGA Implementation of Modified Gram-Schmidt QR-Decomposition

P.N.Ganchosov[1,2], G.K.Kuzmanov[1], H.Kabakchiev[3], V.Behar[4], R.Romansky[2], G.N.Gaydadjiev[1]

CE EEMCS[1] Delft University of Technology,Makelweg 4, 2628 CD Delft
{P.N.Ganchosov, G.K.Kuzmanov, G.N.Gaydadjiev }@tudelft..nl
FKSU[2],Technical Univetsity Sofia,Kliment Ohridski Blvr 8,1000 Sofia Bulgaria, rrom@tu-sofia.bg
FMI[3],Sofia University, James Bouchier str 5,1164, Sofia, Bulgaria, ckakbakchiev@tu-sofia.bg
IPP[4],BAS Acad.G. Bonchev Str., 25-A, 1113 Sofia, Bulgaria, behar@bas.bg

Abstract. The QR-Decomposition is an essential stage in solving the least square minimization computational problem as part of the space and time adaptive filtering (STAP) and beam forming task. In order to meet the STAP filtering real-time demands for Global Positioning System (GPS) signals, precautions have been taken to reduce the computational complexity of the QR decomposition technique accordingly. The purpose of this work is to design an efficient reconfigurable architecture supporting a numerically stable Modified-Gram-Schmidt QR decomposition. By means of the AccelDSP tool of Xilinx, the floating point computational model is translated into a fixed point one, keeping the resolution of the dynamic input data at the task satisfactory level of accuracy, namely five digits after the decimal point. As a result, we obtain scalable and portable organization, which facilitates the implementation on a variety of Xilinx FPGA devices. Analytical resource estimates, along with preliminary suboptimal synthesis results illustrating the weight calculation delays are also presented. The achieved input sampling rate is 64.816 kilo samples per second @ 33,6 MHz as estimated design frequency. This is enough to meet the real time demands for decomposing a covariance matrix of seven receiving elements with 851 time stamps per channel sampled at 20 MHz.

Keywords: FPGA, QR-Decomposition, AccelDSP, STAP.

1 Introduction

The QR-Decomposition is a numerical method that is exploited in the least square minimization computational problem as part of many MIMO systems. It is a decomposition that factors a matrix into two other matrices, namely an orthogonal and an upper triangular. The QR triangularization of the input data matrix is a very powerful and numerically stable technique for the computation of the least squares estimates of an unknown finite impulse response system based on the minimization of the total squared error between the actual and the desired response signals over a given time interval. The purpose of this work is to introduce a hardware implementation of the Modified Gram Schmidt QR decomposition (MGS-QRD) as a computationally essential stage of maximizing signal to noise ratio in GPS front-end receivers. The high computational complexity gives us motivation for using a

dedicated FPGA based implementation of MGS-QRD in order to achieve real time requirements. In this paper we present our preliminary design and its evaluation. More specifically, we have achieved the following:

- Created a Matlab Model of the MGS-QRD arithmetic pipeline;
- Identified the computational kernels (referred as A, B and C), potential candidates for performance efficient hardware implementation supporting MGS-QRD;
- Obtained preliminary synthesis results on automatically generated VHDL. More specifically, at Virtex II Pro xc2vp30 we achieved performance of 64.816 KSPS @ 33.6 MHz for a design with 20 bits (4 bits integer and 16 bits fraction) fixed point arithmetic, which suits the real time and the accuracy requirements of the target application.

The remainder of the paper is organized as follows: Section 2 presents the theoretical background on the essentials of the MGS-QR algorithm and gives the motivation for using it. Section 3 introduces the proposed hardware implementation and our approach in utilizing MGS-QR through a matlab-code and an arithmetic operation flow diagram. Section 4 reports preliminary synthesis results. Section 5 is an exposure of ongoing work. Finally Section 6 concludes the paper and gives directions for future work.

2. Theoretical Background

In linear algebra, the QR decomposition (also called the QR factorization) of a matrix is its decomposition into an orthogonal and a right triangular matrices. The QR decomposition is often used to solve the linear least squares problem. The QR decomposition is also the basis for a particular eigenvalue algorithm. The QR decomposition is widely applied nearly in every field in which a numerically stable and robust inversion of large dense matrixes is needed. Namely in the field of MIMO communications, software defined radio, adaptive and systolic recursive algorithms for noise suppression etc. Recently, there are three well-known algorithms for QR-Decomposition, namely: 1) Cholesky factorization or as it is also known as Givens rotations [4]; 2) Householder reflections [5]; and 3) addressed in this paper Modified-Gram-Schmidt decomposition [5], [6]. The modified Gram Schmidt is proven in [5] and [7] to be numerically identical to Givens Rotations while offering the same level of numerical stability and accuracy. In Table 1 we provide a comparison of the needed floating point (FP) operations as a function of the number of rows (m) and number of columns (n) in the matrix [8].All of the three methods have the same asymptotic complexity of $O(n^3)$.

Table 1. FP operation complexity of the three QR-decomposition algorithms

Methods	Num of Floating Point Operations
MGS	$8mn^3 - 2mn$
House-holder	$16\left(m^2 n - m^2 + \frac{1}{3}n^3\right)$
Givens Rotations	$14\,mn^2 - \frac{14}{3}n^3 - mn - \frac{13}{2}n^2 - \frac{11}{6}n$

For values of m such that $m \geq 2n/3$ the actual complexity of MGS is dominated by m. In this case MGS requires significantly fewer operations than GR and Householder. Such matrix sizes are the most common case when we are observing a GPS signal, which motivates us to use the latter algorithm. In the remainder of this paper, we will consider only MGS.

The Modified Gram-Schmidt Algorithm: Consider the Gram–Schmidt process, with the vectors to be processed as separate columns of the matrix $A = (a_1 \mid \ldots \mid a_n)$. We define $proj_e a = \frac{\langle e, a \rangle}{\langle e, e \rangle} e$ where $\langle e, a \rangle$ denotes the inner product of the vectors e and a. This operator projects the vector a orthogonally onto the vector e. Furthermore Gram-Schmidt process is mathematically described using the following recursive equations:

$$\begin{aligned}
u_1 &= a_1, & e_1 &= \frac{u_1}{\|u_1\|} \\
u_2 &= a_3 - proj_{e_1} a_2, & e_2 &= \frac{u_2}{\|u_2\|} \\
u_2 &= a_3 - proj_{e_1} a_2 - proj_{e_2} a_3, & e_3 &= \frac{u_3}{\|u_3\|} \qquad (1) \\
&\ldots \\
u_k &= a_k - \sum_{j=1}^{k-1} proj_{e_j} a_k, & e_k &= \frac{u_k}{\|u_k\|}
\end{aligned}$$

The sequence $u_1, \ldots, u_k$ is the required system of orthogonal vectors, and the normalized vectors $e_1, \ldots, e_k$ form an orthonormal set. Next we rearrange the equations above so that the a_i s are on the left side, producing the following equations.

$$\begin{aligned}
a_1 &= e_1 \|u_1\| \\
a_2 &= proj_{e_1} a_2 + e_2 \|u_2\| \\
a_3 &= proj_{e_1} a_3 + proj_{e_2} a_3 + e_3 \|u_3\| \qquad (2) \\
&\ldots \\
a_k &= \sum_{j=1}^{k-1} proj_{e_j} a_k + e_k \|u_k\|
\end{aligned}$$

Note that since the e_i are unit vectors, we have the following.

$$\begin{aligned}
a_1 &= e_1 \|u_1\| \\
a_2 &= \langle e_1, a_2 \rangle e_1 + e_2 \|u_2\| \\
a_3 &= \langle e_1, a_3 \rangle e_1 + \langle e_2, a_3 \rangle e_2 + e_3 \|u_3\| \qquad (3) \\
&\ldots \\
a_k &= \sum_{j=1}^{k-1} \langle e_j, a_k \rangle + e_k \|u_k\|
\end{aligned}$$

Now the right sides of equations (3) can be written in matrix form as follows:

$$\underbrace{(e_1 \mid \ldots \mid e_n)}_{Q} \overbrace{\begin{pmatrix} \|u_1\| & \langle e_1, a_2 \rangle & \langle e_1, a_3 \rangle & \ldots \\ 0 & \|u_2\| & \langle e_2, a_3 \rangle & \ldots \\ 0 & 0 & \|u_3\| & \ldots \\ : & : & : & : \end{pmatrix}}^{R} \qquad (4)$$

However the product of each row and column of equation (4) matrices give us a respective column of A that we started with. Together, they give us the matrix A. So we have factorized A into an orthogonal matrix Q (the matrix of e_ks), via Gram Schmidt, and the obvious upper triangular matrix as a remainder R.

Employing the QR decomposition for GPS signal observations concern matrices of significant size ,e.g. *[1000x10]* elements that cannot be stored entirely into the internal memory of an FPGA chip, which is often an issue to achieve a performance and design efficiency. Data segmentation is desired and is a key factor for high performance [1]. There are two ways of segmentation: 1) manual, not so easy scalable, addressing number of samples, respectively size of vectors. 2) Compiler based techniques for automatic data segmentation [2]. In the most common case matrices are segmented into columns, which in case of extra large vector sizes is hard to achieve. Therefore for large vectors a square tiling is used instead [3].The price that should be paid is the overhead by additional update routines, but the positive effect is that we have an easily scalable architecture.

3. Proposed Implementation

Matlab Model: for validation purposes, we developed a matlab model of the MGS-QRD. The matlab model was intentionally split into two streams dividing the input data vector into real and imaginary part. No built-in functions were used, e.g. *norm*, *dot* and more. Only basic arithmetic operations like *sum*, *subtract*, *square root*, and *divide* are used. This is the pseudo code of MGS – QRD:

for $i = 1 : n$
$u_i = X_i$
$R_{ii} = \|u_i\|$
$Q_i = u_i / R_{ii}$
for $j = i + 1 : n$
$R_{ii} = Q_i u_j$
$u_j = u_j - R_{ij} Q_i$

The model was made without any error handling procedures, so valid data are expected at the input. For our matlab model we assume that the input matrix X can be expressed as follows:

$$X = \sum_{k=1}^{4} S_k A(\theta_k) + \sum_{j=1}^{J} Jam_j B(\phi_j) + N .$$

Where S is a matrix of GPS signals mixed with matrix of jamming signals Jam and matrix of Gaussian noise N. The validity of data is ensured by an external procedure which is not in the scope of the current work. For indicative evaluation of the potential positive impact of hardware MGS-QRD on the performance, the above matlab model is considered for automatic hardware generation. For this automatic hardware synthesis, we employ a synthesizable matlab script for the AccelDSP tool of Xilinx. We are aware that this automatically generated design will be not as performance efficient as a manually crafted one, but it can be used as a first indication of the potential efficiency of the MGS-QRD hardware support.

The following subsequent actions were performed to obtain the preliminary performance figures, needed for our evaluation:

1) Functional verification of the floating point model. This stage consists of compiling and running the matlab source of MGS-QRD;

2) Analysis of the dynamic range of the input data, checking for any incompatible and unsupported data structures and operations, such as dynamic fields, non-scalar structures of objects, arithmetic operations on entire structures, etc; They do exist as such in Matlab but precautions have been taken not to use it in the our synthesizable code in order to obey AccelDSP restrictions;

3) Generation of a fixed point model. Translations of the model into a fixed point one applying directives and quantification models. The quantification model is a result of analyzing the dynamic range of data. Based on the observation of the input stimuli, suitable bits for quantification are chosen. Minimum, maximum value and granularity are kept in mind in order to achieve desired resolution and accuracy in fewer possible bits;

4) Verification of the fixed-point model. The fixed point model of MGS-QRD is executed 140 times with pseudo random generated input data and every time its result is subtracted from the floating point model of the latter one. Thus, we have an idea what is the error introduced by the translation of the floating point design into a fixed point arithmetic one. (there are graphical result in section 4, Fig. 2).For better precision, more bits must be used in calculations which introduce the trade-off between the precision and area of the design. Fig. 1 illustrates the influence of the number of bits on the error.

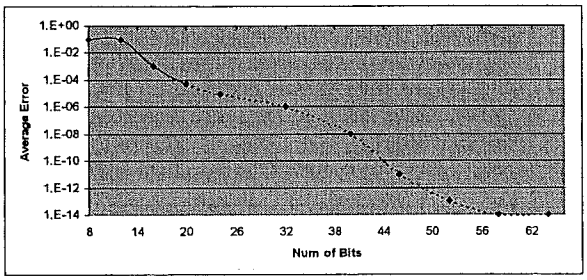

Fig. 1 Mean Error for fixed point arithmetic

Test points were taken through AccelDSP till the maximum allowed value from the environment, which is 20 bits fixed point representation, this maximum value of 20 bits was chosen as a constrain in automatically generated hardware. More specifically this is the exact representation of the current implementation.

Sign bit	3 Integer bits	16 Fractional bits

The data after 20^{th} bit are analytically estimated through matlab using the formula $\frac{\sum_{i=1}^{n} \left| a^{fp}{}_i - a^{int}{}_i \right|}{n}$ and shall be further confirmed by the manually crafted hardware.

Reconfigurable design implementation: Up to this stage, all the procedures and designs were kept hardware platform independent. In order to proceed with the actual implementation of the design, we considered real FPGA prototyping platform based on Xilinx Virtex II PRO (XC2VP30). To map the design on reconfigurable hardware we employed the following steps:

1) The obtained fixed point design is further processed for RTL translation into VHDL. In order to achieve a better performance, all of the *Array Multiplies*, *Array Subtracts* and *Array adds* are unrolled. Keeping in mind the need for low area consumption, none of the *for loops* and *Matrix Multiplies* were unrolled. For hardware and performance efficiency, the design exploits resource sharing and pipelining, which is set as an option in AccelDSP as it is shown in [9].

2) The VHDL code was synthesized using the XST 8.1i tool invoked from AccelDSP 10.1.

3) The synthesis report was generated and included in section 4, table 2.

In future work, our main aim is to employ custom designed fix-point division and square root implementations. This is expected to additionally increase the performance of our design and make it more efficient in terms of hardware resources utilization, compared to other traditional ones such as CORDIC architecture [10] or the logarithmic number system (LNS) based architecture [11].Observations how different sizes of the input matrix will affect resources utilization, the actual throughput of the architecture, the maximum sampling rates and the frequencies can be obtained from the synthesis results (section 4, table 2.). Up to this moment, only an IP core based on the Givens Rotation QR decomposition method is available in AccelWare library. Unfortunately that core has restrictions for the size of the input matrix, which is limited to [32x32] and cannot be scaled easily. That makes it inapplicable in the context of STAP GPS filtering algorithm based on minimum variance distortion response approach (MVDR).,where the size of the input matrix is orders of magnitude bigger i.e. [851x7]. Our goal is to provide a design that can meet these particular STAP GPS requirements, work that has not been proposed before to our best knowledge.

Implementation requirements: The real time demands for QR decomposing in the context of STAP GPS filtering are determined by three factors: 1) Sample Frequency 2) Number of channels 3) time stamps per channel. As it is suggested in [12], 20 MHz sampling frequency is widely used. In summary : For GPS the code length is 1023 chips, 1.023MHz chipping rate(1 ms period),50 Hz data rate (20 code periods per data bit), 90% of signal power is within 2 MHz

bandwidth/. $R = (851\ samples\ .\ 50\ ns).7\ channels\ = 29{,}785\ ms$ .The accuracy requirements: we estimate that minimum 4 digits after decimal point precision is needed, in order to achieve efficient noise cancellation (-100dB) and efficient gain (20 dB) in the direction of arrival of the useful GPS signal. The evaluations were based on our Matlab model of Minimum Variance Distortionless Response algorithm. The latter is where we employ MGS-QRD. These figures are our performance requirements which we target in our design.

4. Preliminary Results

The normalized verification result of the of the fixed point model are shown on Fig. 2. We can see an equal distribution of the error in the 4th and 5th digit after the decimal point, that suits the accuracy needs for filtering a Jammed GPS signal. In case that, there is need of additional accuracy, there is always a possibility to increase the precision of quantification by increasing the number of bits of the fractional and non fractional parts of the integer representation. Of course this will affect the remaining design stages. The synthesis results in table 2. indicate that the architecture is suitable for implementation in Virtex II pro Xilinx device. More precisely, on the xc2vp30, we employ 60% of the slices, 54% of the LUTs and 35% of the 18x18 multipliers. The achieved performance of 64,816 KSPS@33,3MHz is enough to meet the demands of real-time QR decomposition in context of STAP filter for GPS signals.

Table 2 Synthesis results

a)

Part Information	
Vendor: Xilinx	
Technology	Virtex2p
Device	xc2vp30
Package	ff896
Speed Grade	-7

b)

Utilization		
Information	**Count**	**Percentage Use**
Slices	8258 of 13696	60%
Slice Flip Flops	3166 of 27392	11%
4 input LUTs	14870 of 27392	54%
MULT18X18s	48 of 136	35%

c)

Timing Path Information			
Clock Name	**Path Name**	**Estimated Frequency**	**Estimated Period**
Clock	Input to Register	81.9 MHz	12.2170 ns
Clock	Register to Register (worst case)	33.6 MHz	29.7270 ns
Clock	Register to Output	291.8 MHz	3.4270 ns

d)

Performance					
Clock Name	**Requested Frequency**	**Estimated Frequency**	**Estimated Period**	**Max Throughput**	**Input Sampling**
Clock	100.0 MHz	33.6 MHz	29.7270 ns	519	64.816 KSPS

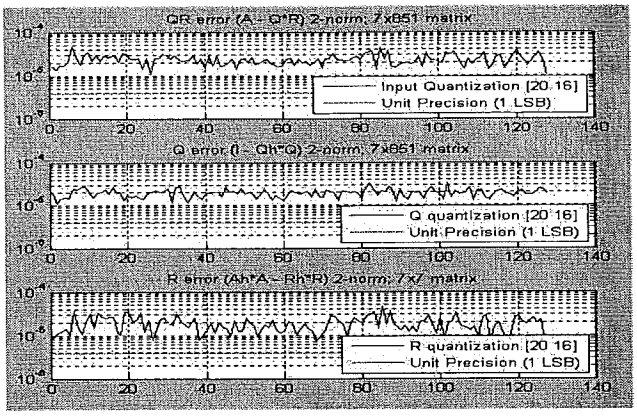

Fig. 2 Quantification error

Table 2a gives detailed information for the used chip, more specifically, which is its vendor (Xilinx), exact model(xc2vp30), type of pin layout(ball grid 896 pins) and its frequency (speed grade -7 denotes a 100 MHz front bus frequency). In 2b there is the information for resource utilization of the chip. In 2c timing path information presents the achieved frequency rates for the input, for the operational part and for the output of the design. In table 2d the performance results are shown. More specifically the requested speed is the maximum speed at which the chip can run. A 64.816 kilo samples per second input sampling rate is achieved at relatively lower frequency of 33.6 MHz. This is envisioned to be beneficial for the power consumption of the design and meanwhile it still meet the real-time requirements of MGS-QRD in the context of STAP for GPS. The max throughput denotes the actual throughput capacity of the design per cycle.

5. Ongoing work and proposal for implementation:

Once we have strong indications form the automatic synthesis that a hardware implementation of MGS-QRD is performance beneficial, we proceed to manual crafting of the design. The arithmetic Pipeline of the MGS-QRD is presented on Fig. 3. The dotted arrows are vector transactions with (851 elements) x (4 bytes each). The thick arrows denote the scalar transactions (1 element) x (4 bytes each). This arithmetic pipeline helps us to indentify three main candidates for computational kernels suitable for hardware implementation. Kernel A consists of vector by vector multiplier, vector by scalar multiplier and vector by vector subtraction module. Kernel B consists of vector involution of two, sum operation and square root operation. Kernel C consists of vector by scalar divider, vector transition, and vector by vector multiplication. In designing the Kernel C, we recognize that general division is an expensive operation in hardware, therefore a suitable approach exploiting divisions by the power of two allowing the use of shift registers will be applied. All these 3 kernels will be further considered for VHDL description and HW synthesis. An example of a

possible HW organization, supporting kernel A is depicted in Fig. 4. We perform this in two steps:

1) Indentify mathematical kernels that can be reused by the algorithm;

2) Implement these kernels in hardware, which will support the indentified math operations;

The designed units can be employed later either as subunits in a standalone design or as supporting hardware for an application specific instruction set architecture extension;

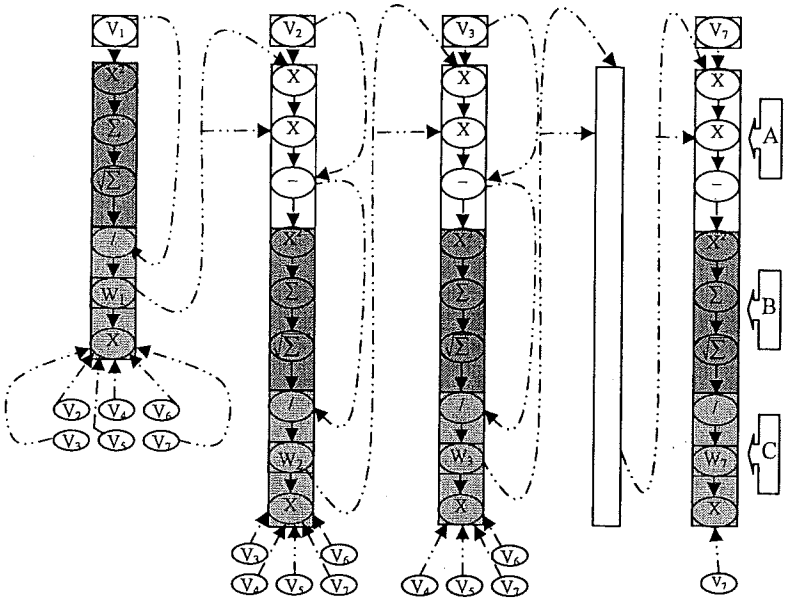

Fig. 3 Arithmetic pipeline of MGS-QRD

Here we propose a simplified design for the kernel A .The vector V_i and the vector V_{i+1} are concurrently fetched and directed into a vector multiplication pipe. The scalar result S is stored into a scalar register, which is further multiplied by the already fetched vector V_i .The intermediate result is stored into a vector buffer. Then, through the vector subtraction pipe, we subtract the content of vector buffer from fetched V_{i+1}.Thus the output of kernel A V_{res} is stored into the memory through the store pipe. All the depicted transactions are organized though FIFOs. The memory bandwidth that we assume is estimated to be 64 bit/cycle read mode and 32 bit/cycle write mode.

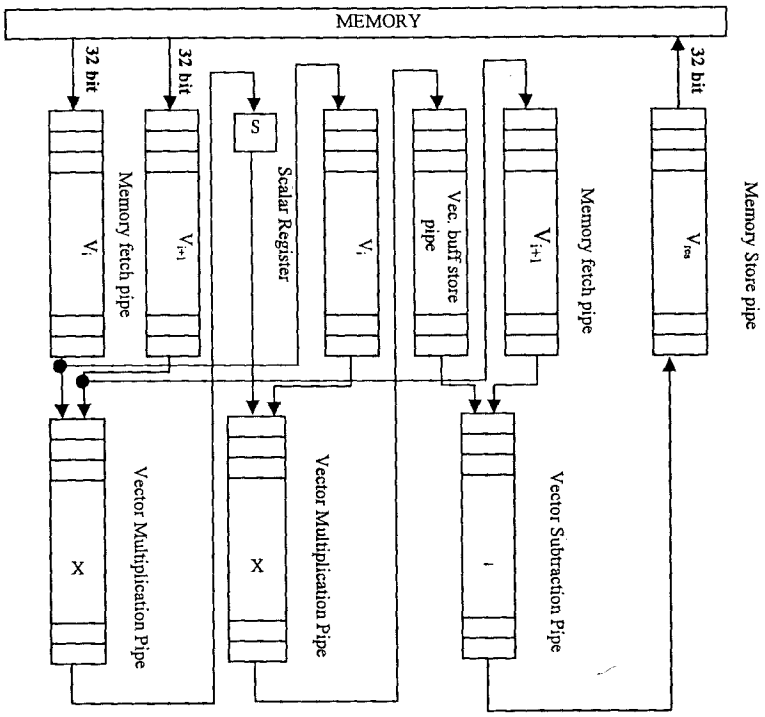

Fig. 4 simplified organization for kernel A

Identically, hardware supporting kernel B and kernel C will be designed. The expected result is three new functional units that can support the main kernels (A, B and C) for MGS-QRD that would contribute to its more efficient implementation.

6. Conclusion and Future work

This article presents our ongoing work on efficient implementation of MGS-QRD targeting Xilinx FPGAs. The main conclusion of the presented design effort is that a modified Gram-Schmidt QR-decomposition can be mapped on available reconfigurable devices in the context of STAP filter for GPS signals. Our preliminary experiments with automatically generated hardware, supporting this operations indicate that it can be accelerated. Currently we are designing handcrafted VHDL for the indentified kernels of interest. When all the kernels are ready one of the possible

directions is to make a heterogeneous model – mixing vector based computational engine, organized as systolic structure.

Acknowledgments: This work was financially supported by the HiPEAC Research cluster 1186 / 07/19/07 and Bulgarian National Science Fund BG MU - FS-05/ 07/3.3-02-7/08

References

[1] E. Anderson, Z.Bai, C. Bischof, S.Blackford, J.Demel, J. Dongara, J.D. Croz, A. Greenbaum, S.Hammerlen, A. McKenney, and D.Sorensen, "LAPACK User's Guide" 3rd edition, Society for industrial and Applied Mathematics,1999.

[2] Q.Yi, K.Kennedy, H. You, K, Seymor, and J. Dongarra, "Automatic blocking of QR and LU factorization for locality" in MSP'04: Proceedings of the 2004 workshop on memory system performance. New York, NY, USA: ACM Press, 2004, pp.12-22.

[3] B.C. Gunter and R.A.V.D.Geijn, " Parallel out-of-core computation and updating of the QR factorization", ACM Trans.Math.Softw.,vol.31,no.1,pp.60-75,2005.

[4] A.El-Amawy and K.R.Dharmarajan, "Parallel vlsi algorithm for stable inversion of dense matrices" IEEE proceedings on Computers and Digital Techniques, vol. 136, no. 6, pp. 575-580, Nov 1989.

[5] Golub, Gene H. & Van Loan, Charles F. (1996), "Matrix Computations" (3rd ed.), Johns Hopkins, ISBN 978-0-8018-5414-9.

[6] Stoer, Josef & Bulirsch, Roland (2002), "Introduction to Numerical Analysis" (3rd ed.), Springer, ISBN 0-387-95452-X.

[7] A, Bjork, "Numerical methods for Least Square Problems", Philadelphia PA-SIAM, 1996.

[8] Di Wu,Yi-Hsien Li,Johan Eilert and Dake Liu "Real-Time Space-Time Adaptive Processing on the STI CELL Multiprocessor", proceeding 4th EURAD,Munich 2007

[9] Xilinx "MATLAB forSynthesis" *Style Guide* Release 10.1 March, 2008

[10] Boppana.D,Dhanoa.K,Kempa,J "FPGA based Embedded Processing Architecture for QRD-RLS Algorithm" Field-Programmable Custom Computing Machines, 2004. FCCM 2004. 12th Annual IEEE Symposium

[11] Chitranjan K. Singh, Sushma H. Prasad and Poras T. Balsara "A Fixed-Point Implementation for QR Decomposition" 2006 IEEE Dallas/CAS workshop

[12] Kai Bore, Denis Akos, Nikolai Bertelsen, Peter Rinder, Soren Jensen"A Software-Defined GPS and Galileo Receiver: A Single-Frequency Approach" SBN: 0-8176-4390-7,2007