MATRIX INVERSE ON CONNEX PARALLEL ARCHITECTURE

Ana-Maria CALFA[1], Gheorghe ŞTEFAN[2]

Designed for embedded computation in system on chip design, the Connex Parallel Architecture can compete with general purpose devises in the domain of matrix computation, improving execution time and GIPS/Watt because of its specific architectural features. The actual performances of this architecture in the computation of matrix inversion are investigated using both analytical calculation and numerical simulation.

Keywords: parallel architecture, embadded systems, inverse of a matrix

1. Introduction

In this paper we will continue to investigate the Connex Parallel Architecture [1], started on [2] for the first computational motif from *Berkeley's view* [3] - linear algebra.

Two parallel algorithms to calculate the inverse of a matrix [4] are described in the second section, one optimized for energy use and the other one for area efficiency.

In the third section the peak performances of Connex Parallel Architecture are compared with a general purpose CPU and a general purpose microcontroller, in order to emphasize the advantages of Connex Architecture.

We conclude in the fourth section that, even if the Connex Architecture is a very simple architecture compared with its competitors, area and power gain are significant.

2. Matrix Inverse

In this paper, only dense matrices will be investigated and will be later mentioned as matrices.

The inverse of a matrix A is the matrix A^{-1} that satisfies the property:

$$A\,A^{-1} = A^{-1}A = I \tag{1}$$

[1] PhD student, Faculty de Electronics, Telecommunications and Information Tehchnology, University POLITEHNICA of Bucharest, Romania, e-mail: annie_calfa@yahoo.com

[2] Prof., Faculty de Electronics, Telecommunications and Information Tehchnology, University POLITEHNICA of Bucharest, Romania, e-mail: gstefan@arh.pub.ro

where I is the unity matrix.

Inverse of a matrix can be defined only for square matrices. Not all square matrices have inverses, but if they do, they are unique! A matrix with inverse is called invertible or nonsingular and one without inverse is called noninvertible or singular.

The inverse of a matrix can be calculated using two methods:

a. **Gauss-Jordan elimination** transforms [A | I] into [I | A^{-1}].

To exemplify, we consider matrices of 4x4:

$$\cdots\left(\begin{array}{cccc|cccc} a_{11} & a_{12} & a_{13} & a_{14} & 1 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & 0 & 1 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & 0 & 1 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & 0 & 0 & 0 & 1 \end{array}\right)\left(\begin{array}{cccc|cccc} b_{11} & b_{12} & b_{13} & b_{14} & 1 & 0 & 0 & 0 \\ b_{21} & b_{22} & b_{23} & b_{24} & 0 & 1 & 0 & 0 \\ b_{31} & b_{32} & b_{33} & b_{34} & 0 & 0 & 1 & 0 \\ b_{41} & b_{42} & b_{43} & b_{44} & 0 & 0 & 0 & 1 \end{array}\right)\cdots \quad (2)$$

The method of Gauss-Jordan elimination involves creating the index matrix instead of the initial matrix and its inverse instead of the unit matrix. This implies to form zeros, under and upper the main diagonal of the initial matrix. The matrix to be inverted and the index matrix attached are considered a unit and all the operations are going to be applied to it.

To use Connex specific architectural features, operation must be performed on column. On the first column, first step is to divide the first row by the element of the main diagonal of the matrix that needs to be inverted.

$$\cdots\left(\begin{array}{cccc|cccc} 1 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \frac{a_{14}}{a_{11}} & \frac{1}{a_{11}} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & 0 & 1 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & 0 & 1 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & 0 & 0 & 0 & 1 \end{array}\right)\left(\begin{array}{cccc|cccc} 1 & \frac{b_{12}}{b_{11}} & \frac{b_{13}}{b_{11}} & \frac{b_{14}}{b_{11}} & \frac{1}{b_{11}} & 0 & 0 & 0 \\ b_{21} & b_{22} & b_{23} & b_{24} & 0 & 1 & 0 & 0 \\ b_{31} & b_{32} & b_{33} & b_{34} & 0 & 0 & 1 & 0 \\ b_{41} & b_{42} & b_{43} & b_{44} & 0 & 0 & 0 & 1 \end{array}\right)\cdots \quad (3)$$

If this element is zero then we need to interchange this row with another one of grater index that doesn't have this element zero. If all the rows have the first element zero it means that the matrix is singular.

To create zeros on the first column under the main diagonal we need to multiply, one by one, the first row with the first element of the row where we intend to create zero and then decrease it from the corresponding row.

Applying the procedure listed above for the second row:

$$..a_{21}\cdot\left(\begin{array}{cccc|cccc} 1 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \frac{a_{14}}{a_{11}} & \frac{1}{a_{11}} & 0 & 0 & 0 \end{array}\right)b_{21}\cdot\left(\begin{array}{cccc|cccc} 1 & \frac{b_{12}}{b_{11}} & \frac{b_{13}}{b_{11}} & \frac{b_{14}}{b_{11}} & \frac{1}{b_{11}} & 0 & 0 & 0 \end{array}\right)...=$$

$$..\left(\begin{array}{cccc|cccc} a_{21} & \frac{a_{21}\cdot a_{12}}{a_{11}} & \frac{a_{21}\cdot a_{13}}{a_{11}} & \frac{a_{21}\cdot a_{14}}{a_{11}} & \frac{a_{21}}{a_{11}} & 0 & 0 & 0 \end{array}\right)\left(\begin{array}{cccc|cccc} b_{21} & \frac{b_{21}\cdot b_{12}}{b_{11}} & \frac{b_{21}\cdot b_{13}}{b_{11}} & \frac{b_{21}\cdot b_{14}}{b_{11}} & \frac{b_{21}}{b_{11}} & 0 & 0 & 0 \end{array}\right)..\qquad(4)$$

and then, decreasing the result from the second row, results:

$$...\left(\begin{array}{cccc|cccc} 1 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \frac{a_{14}}{a_{11}} & \frac{1}{a_{11}} & 0 & 0 & 0 \\ 0 & a_{22}-\frac{a_{21}\cdot a_{12}}{a_{11}} & a_{23}-\frac{a_{21}\cdot a_{13}}{a_{11}} & a_{24}-\frac{a_{21}\cdot a_{14}}{a_{11}} & -\frac{a_{21}}{a_{11}} & 1 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & 0 & 1 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & 0 & 0 & 0 & 1 \end{array}\right)...\qquad(5)$$

Continuing this operation for all remaining rows, it will be obtained zeros on the first column:

$$...\left(\begin{array}{cccc|cccc} 1 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \frac{a_{14}}{a_{11}} & \frac{1}{a_{11}} & 0 & 0 & 0 \\ 0 & a_{22}-\frac{a_{21}\cdot a_{12}}{a_{11}} & a_{23}-\frac{a_{21}\cdot a_{13}}{a_{11}} & a_{24}-\frac{a_{21}\cdot a_{14}}{a_{11}} & -\frac{a_{21}}{a_{11}} & 1 & 0 & 0 \\ 0 & a_{32}-\frac{a_{31}\cdot a_{12}}{a_{11}} & a_{33}-\frac{a_{31}\cdot a_{13}}{a_{11}} & a_{34}-\frac{a_{31}\cdot a_{14}}{a_{11}} & -\frac{a_{31}}{a_{11}} & 0 & 1 & 0 \\ 0 & a_{42}-\frac{a_{41}\cdot a_{12}}{a_{11}} & a_{43}-\frac{a_{41}\cdot a_{13}}{a_{11}} & a_{44}-\frac{a_{41}\cdot a_{14}}{a_{11}} & -\frac{a_{41}}{a_{11}} & 0 & 0 & 1 \end{array}\right)...\qquad(6)$$

The zeros on the second column will be first created under the main diagonal, this means that we should start from the second row where we have to generate "1" on the main diagonal. This is done dividing the second row by $a_{22}-\frac{a_{21}\cdot a_{12}}{a_{11}}=a_{22\alpha}$.

To simplify the equations we make the following notations

$$a_{xy}-\frac{a_{x1}\cdot a_{1y}}{a_{11}}=a_{xy\alpha}\qquad(7)$$

Applying (7) to (6):

$$\left.\begin{pmatrix} 1 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \frac{a_{14}}{a_{11}} \\ 0 & 1 & \frac{a_{23\alpha}}{a_{22\alpha}} & \frac{a_{24\alpha}}{a_{22\alpha}} \\ 0 & a_{32\alpha} & a_{33\alpha} & a_{34\alpha} \\ 0 & a_{42\alpha} & a_{43\alpha} & a_{44\alpha} \end{pmatrix} \right| \begin{matrix} \frac{1}{a_{11}} & 0 & 0 & 0 \\ -\frac{a_{21}}{a_{22\alpha}\cdot a_{11}} & \frac{1}{a_{22\alpha}} & 0 & 0 \\ -\frac{a_{31}}{a_{11}} & 0 & 1 & 0 \\ -\frac{a_{41}}{a_{11}} & 0 & 0 & 1 \end{matrix} \Bigg) \ldots \tag{8}$$

To create zeros on the second column under the main diagonal, we apply the same method as for the first column. Multiplying it, one by one, with the elements of grater index on the same column:

$$\ldots a_{32\alpha}\cdot\left(\begin{matrix} 0 & 1 & \frac{a_{23\alpha}}{a_{22\alpha}} & \frac{a_{24\alpha}}{a_{22\alpha}} \end{matrix}\right| \left.\begin{matrix} -\frac{a_{21}}{a_{22\alpha}\cdot a_{11}} & \frac{1}{a_{22\alpha}} & 0 & 0 \end{matrix}\right) = \ldots$$

$$\ldots\left(\begin{matrix} 0 & a_{32\alpha} & a_{32\alpha}\cdot\frac{a_{23\alpha}}{a_{22\alpha}} & a_{32\alpha}\cdot\frac{a_{24\alpha}}{a_{22\alpha}} \end{matrix}\right| \left.\begin{matrix} -\frac{a_{32\alpha}\cdot a_{21}}{a_{22\alpha}\cdot a_{11}} & \frac{a_{32\alpha}}{a_{22\alpha}} & 0 & 0 \end{matrix}\right) = \ldots \tag{9}$$

and then decreasing it from the corresponding row, results:

$$\left.\begin{pmatrix} 1 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \frac{a_{14}}{a_{11}} \\ 0 & 1 & \frac{a_{23\alpha}}{a_{22\alpha}} & \frac{a_{24\alpha}}{a_{22\alpha}} \\ 0 & 0 & a_{33\alpha}-a_{32\alpha}\cdot\frac{a_{23\alpha}}{a_{22\alpha}} & a_{34\alpha}-a_{32\alpha}\cdot\frac{a_{24\alpha}}{a_{22\alpha}} \\ 0 & 0 & a_{43\alpha}-a_{42\alpha}\cdot\frac{a_{23\alpha}}{a_{22\alpha}} & a_{44\alpha}-a_{42\alpha}\cdot\frac{a_{24\alpha}}{a_{22\alpha}} \end{pmatrix} \right| \begin{matrix} \frac{1}{a_{11}} & 0 & 0 & 0 \\ -\frac{a_{21}}{a_{22\alpha}\cdot a_{11}} & 1 & 0 & 0 \\ -(\frac{a_{31}}{a_{11}}+\frac{a_{32\alpha}\cdot a_{21}}{a_{22\alpha}\cdot a_{11}}) & \frac{a_{32\alpha}}{a_{22\alpha}} & 1 & 0 \\ -(\frac{a_{41}}{a_{11}}+\frac{a_{42\alpha}\cdot a_{21}}{a_{22\alpha}\cdot a_{11}}) & \frac{a_{42\alpha}}{a_{22\alpha}} & 0 & 1 \end{matrix} \Bigg) \ldots \tag{10}$$

Continuing these operations for all columns, instead of initial matrix will be obtained a matrix that has ones on the main diagonal and zeros under it. Applying the same method for the upper part of the matrix, starting with the last row, the unit and the inverse matrix will be obtained:

$$\ldots\left(\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{matrix}\right| \left.\begin{matrix} a_{11'} & a_{12'} & a_{13'} & a_{14'} \\ a_{21'} & a_{22'} & a_{23'} & a_{24'} \\ a_{31'} & a_{32'} & a_{33'} & a_{34'} \\ a_{41'} & a_{42'} & a_{43'} & a_{44'} \end{matrix}\right) \left(\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}\right| \left.\begin{matrix} b_{11'} & b_{12'} & b_{13'} & b_{14'} \\ b_{21'} & b_{22'} & b_{23'} & b_{24'} \\ b_{31'} & b_{32'} & b_{33'} & b_{34'} \\ b_{41'} & b_{42'} & b_{43'} & b_{44'} \end{matrix}\right) \ldots \tag{11}$$

b. **Cramer's rule** is defined by :

$$A^{-1} = \frac{adjoint(A)}{\det(A)} = \left(\frac{cofactor\_matrix(A)}{\det(A)}\right)^{T} \tag{12}$$

First step is to calculate the determinant for all defined matrices. It will be calculated using the Gaussian elimination method presented above, that will be applied until the original matrix becomes a triangular matrix, without creating "1"s on the main diagonal. Under this condition the determinant is defined as the product of all the elements on the main diagonal.

If the determinant is zero then the matrix A is noninvertible. Because, with this architecture can be calculated up to 512 inverses in parallel (for matrices of 2x2), those matrices that have the determinant zero will be marked and the inverse will not be calculated for them. For the rest of the matrices the algorithm will continue until the inverses are found.

The cofactors of a matrix are those determinants that results after deleting the row and the column of an element from a matrix, with the sign given by the sum of the indexes of the given element. If the sum is an even number then the sign is "+", otherwise the sign is "-".

If we consider three matrices for which we want to compute the inverse in parallel:

$$\ldots \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{pmatrix} \begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{pmatrix} \ldots \tag{13}$$

first these matrices will be transposed with the method presented in [1], resulting:

$$\ldots \begin{pmatrix} a_{11} & a_{21} & a_{31} & a_{41} \\ a_{12} & a_{22} & a_{32} & a_{42} \\ a_{13} & a_{23} & a_{33} & a_{43} \\ a_{14} & a_{24} & a_{34} & a_{44} \end{pmatrix} \begin{pmatrix} b_{11} & b_{21} & b_{31} & b_{41} \\ b_{12} & b_{22} & b_{32} & b_{42} \\ b_{13} & b_{23} & b_{33} & b_{43} \\ b_{14} & b_{24} & b_{34} & b_{44} \end{pmatrix} \begin{pmatrix} c_{11} & c_{21} & c_{31} & c_{41} \\ c_{12} & c_{22} & c_{32} & c_{42} \\ c_{13} & c_{23} & c_{33} & c_{43} \\ c_{14} & c_{24} & c_{34} & c_{44} \end{pmatrix} \ldots \tag{14}$$

Then, in auxiliary matrices will be stored those matrices resulted after deleting the row and the column of a specific element, for which will be calculated the determinants (cofactors after adding the sign) using the method of Gauss-Jordan elimination.

```
for(i = 0; i < matr_size; i++){
        Index is 1 in position i from matr_size;
        WHERE(Index == 0)
                Tmp = Matr;
        Tmp is shifted one position to the left;
        for(j = 0; j < matr_size; j++){
                if(( j != 0)||(j != matr_size)){
                        for(k = j; k< matr_size; k++){
                                Tmp[k+1] = Tmp[k];
                        }
                        Calculate determinants for the Tmp matrices using
                        Gauss-Jordan elimination;
                        CofactorMatr[i][j] = (-1)^(i+j) ·det(Tmp);
                }
        }
}
```

If i = 0, the Index vector is:

$$...(1 \quad 0 \quad 0 \quad 0)(1 \quad 0 \quad 0 \quad 0)(1 \quad 0 \quad 0 \quad 0)... \tag{15}$$

applied to (14), it becomes:

$$...\begin{pmatrix} 0 & a_{21} & a_{31} & a_{41} \\ 0 & a_{22} & a_{32} & a_{42} \\ 0 & a_{23} & a_{33} & a_{43} \\ 0 & a_{24} & a_{34} & a_{44} \end{pmatrix}\begin{pmatrix} 0 & b_{21} & b_{31} & b_{41} \\ 0 & b_{22} & b_{32} & b_{42} \\ 0 & b_{23} & b_{33} & b_{43} \\ 0 & b_{24} & b_{34} & b_{44} \end{pmatrix}\begin{pmatrix} 0 & c_{21} & c_{31} & c_{41} \\ 0 & c_{22} & c_{32} & c_{42} \\ 0 & c_{23} & c_{33} & c_{43} \\ 0 & c_{24} & c_{34} & c_{44} \end{pmatrix}... \tag{16}$$

After shifting it one position to the left, the auxiliary matrices are:

$$...\begin{pmatrix} a_{21} & a_{31} & a_{41} & 0 \\ a_{22} & a_{32} & a_{42} & 0 \\ a_{23} & a_{33} & a_{43} & 0 \\ a_{24} & a_{34} & a_{44} & 0 \end{pmatrix}\begin{pmatrix} b_{21} & b_{31} & b_{41} & 0 \\ b_{22} & b_{32} & b_{42} & 0 \\ b_{23} & b_{33} & b_{43} & 0 \\ b_{24} & b_{34} & b_{44} & 0 \end{pmatrix}\begin{pmatrix} c_{21} & c_{31} & c_{41} & 0 \\ c_{22} & c_{32} & c_{42} & 0 \\ c_{23} & c_{33} & c_{43} & 0 \\ c_{24} & c_{34} & c_{44} & 0 \end{pmatrix}... \tag{17}$$

$$\ldots\begin{pmatrix} a_{22} & a_{32} & a_{42} & 0 \\ a_{23} & a_{33} & a_{43} & 0 \\ a_{24} & a_{34} & a_{44} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} b_{22} & b_{32} & b_{42} & 0 \\ b_{23} & b_{33} & b_{43} & 0 \\ b_{24} & b_{34} & b_{44} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} c_{22} & c_{32} & c_{42} & 0 \\ c_{23} & c_{33} & c_{43} & 0 \\ c_{24} & c_{34} & c_{44} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}\ldots \tag{18}$$

For these matrices of size matr_size-1 will be calculated the determinants. It represents the cofactors values for (0,0).

Repeating these actions for all columns and rows the adjoint matrices will be obtained.

$$\ldots\begin{pmatrix} & \begin{vmatrix} a_{22} & a_{32} & a_{42} \\ a_{23} & a_{33} & a_{43} \\ a_{24} & a_{34} & a_{44} \end{vmatrix} & & \\ (-1)^{1+1} & & (-1)^{2+1}\begin{vmatrix} a_{12} & a_{32} & a_{42} \\ a_{13} & a_{33} & a_{43} \\ a_{14} & a_{34} & a_{44} \end{vmatrix} & (-1)^{3+1}|..| & (-1)^{4+1}|..| \\ & \begin{vmatrix} a_{21} & a_{31} & a_{41} \\ a_{23} & a_{33} & a_{43} \\ a_{24} & a_{34} & a_{44} \end{vmatrix} & & & \\ (-1)^{1+2} & & (-1)^{2+2}|..| & (-1)^{3+2}|..| & (-1)^{4+2}|..| \\ & \begin{vmatrix} a_{21} & a_{31} & a_{41} \\ a_{22} & a_{32} & a_{42} \\ a_{24} & a_{34} & a_{44} \end{vmatrix} & & & \\ (-1)^{1+3} & & (-1)^{2+3}|..| & (-1)^{3+3}|..| & (-1)^{4+3}|..| \\ & \begin{vmatrix} a_{21} & a_{31} & a_{41} \\ a_{22} & a_{32} & a_{42} \\ a_{23} & a_{33} & a_{43} \end{vmatrix} & & & \\ (-1)^{1+4} & & (-1)^{2+4}|..| & (-1)^{3+4}|..| & (-1)^{4+4}|..| \\ & & & & \end{pmatrix}\ldots \tag{19}$$

The inverse of matrices will be obtained dividing adjoint matrices with the determinant of initial matrices.

3. Performance of Connex Architecture

The figures in *Table I* represents the clock cycles per operation after running the presented algorithms on the emulator of Connex Architecture for scalar vectors of floating points, without considering the input-output transfer of data.

To emphasize the performances of Connex, the results will be compared with released figures of a general purpose CPU and a microcontroller used in DSP application:

- Processor : 450 MHz, 34 W
- Microcontroller : 30 MHz

The Gauss-Jordan elimination has the advantage that it is quicker than the Cramer's rule, for big matrices, but the number of inverted matrices is halved. Depending on the applications requests one of the two methods can be used.

Table I

Results of matrix invert on Connex Parallel architecture

	Gauss-Jordan elimination		**Cramer's rule**	
	No. of matrices	**Clock cycles**	**No. of matrices**	**Clock cycles**
2x2	256	760	512	300
4x4	128	4472	256	16530
8x8	64	33782	128	492459
16x16	32	312330	64	17098225

In Fig.1 it can be observed that the clock cycles are function of $O(n^3)$ for small n, and function of $O(n^4)$ for n bigger than 8. An improvement of at least *n* is obtained compared with sequential method.

Comparing both inversion methods on Connex Architecture with a sequential processor [5] in *Table II*, for matrices of 4x4, it can be observed that an acceleration of at least 13 times is obtained if is used Cramer's rule method, and efficiency of power (array_inverse/*Watt)* is increased around 150 times.

Table II

Performance of Connex Architecture compared with Intel processor

Method	**Connex (clk cyc/matr)**	**CPU (clk cyc/matr)**	**Performances**	
			Acceleration	**Power eff.**
Gauss-Jordan Elimination	34,94	1074	30,74x	348,4x
Cramer's rule	64,57	846	13,10x	148,5x

If the results are compared with a general purpose microcontroller [6] when the dimension of the matrices is squared, for Cramer's rule, the acceleration is still tens [7] (see *Table III*).

Table III

Performance of Connex Architecture compared with PIC microcontroller for Cramer's rule

Method	**Connex**	**uC**	**Performance**
			Acceleration
Cramer's rule	267160	2521742	9,44x

Connex Architecture, besides lowering the execution time also power is more efficient used compared with current solutions. In the domain of big matrices, performances are even higher due to architectural features.

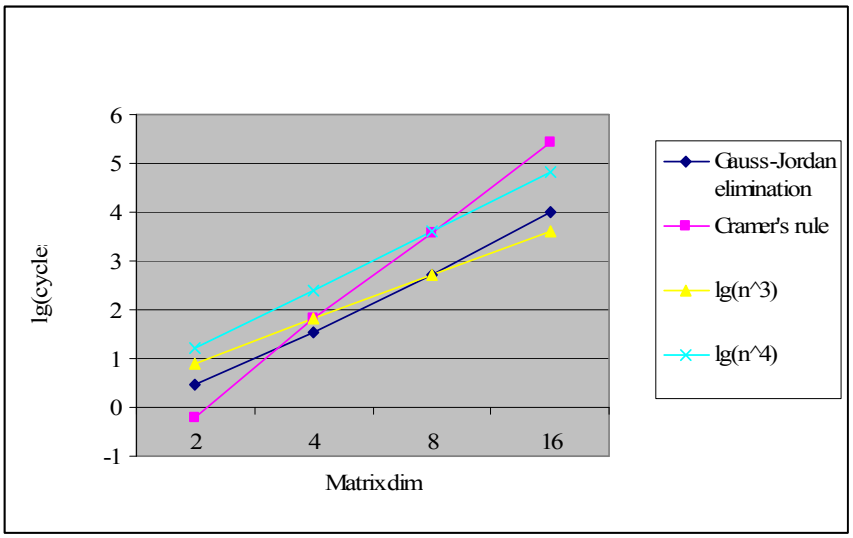

Fig.1 Evolution of clock cycles function of matrix range

Nowadays, there are many security and commercial application that uses matrix inversion: mathematical research systems, cryptography, search engines, automatic pilot s.a. These application processes large amounts of data, and due to its performance, Connex Architecture can be integrated.

The methods and results presented above apply only for dense matrices. Due to high proportion of non useful data, spare matrices have dedicated methods to store the data in the memory and to compute it. Spare matrices are part of future investigation.

4. Conclusion

Due to large amount of low power application that uses matrix inversion, improvements of time execution and power efficiency (costs lowering) in the domain of linear algebra are mandatory.

Even if parallel processors are usually used as accelerators where they are integrated, and most of the times are dedicated to application that processes a large amount of data, due to its performances translated in lower costs, Connex Parallel Architecture can be used in a variety of other applications outside of its design domain.

REFERENCES

[1] *Gheorghe Ştefan*, The CA1024: SoC with Integral Parallel Architecture for HDTV Processing", 4th International System-on-Chip (SoC) Conference & Exhibit, November 1 & 2, 2006, Radisson Hotel Newport Beach, CA

[2] *Ana-Maria Calfa, Gheorghe Stefan*, Matrix Computation on Connex Parallel Architecture, in ICSES 2010 Proceedings, Gliwice – Poland, September 2010

[3] *K. Asanovic*, *et. al.*, The Landscape of Parallel Computing Research: A View from Berkeley, Technical Report No. UCB/EECS-2006-183, December 18, 2006

[4] *H. Anton*, Elementary Linear Algebra, 7th Edition, John Wiley & Sons Inc., 1994

[5] *Intel Corporation*, Streaming SIMD Extensions - Inverse of 4x4 Matrix, AP-928, Order Number 245043-001, March 1999

[6] *Microchip*, dsPIC Language Tools Libraries, DS51456B, 2004

[7] *Artit C. Jarapatnakul,* A Multi-Sensor Embedded Microcontroller System for Condition Monitoring of RC Helicopters, The Pennsylvania State University, 2005