Design and Investigation of a Multi-Core System on Chip

Peter Georgiev Manoilov

Abstract - **The paper describes the design and investigation of a multi-core system with common data memory, implemented on a FPGA – chip. Our goal is, using a system on chip with parallel architecture, to achieve high speed and efficiency, solving the parallel computational problems. The paper presents the results of multi-core system design and functioning for a popular combinatorial problem - finding the shortest paths in weighted graphs.**

Keywords – **FPGA-chip, multi-core system, processor core, weighted graph, shortest path problem.**

I. INTRODUCTION

In graph theory, the shortest path problem [1] is the problem of finding a path between two vertices in a graph such that the sum of the weights of its constituent edges is minimized.

This is analogous to the problem of finding the shortest path between two intersections on a road map: the graph's vertices correspond to intersections and the edges correspond to road segments, each weighted by the length of its road segment. The Floyd–Warshall algorithm [2] is the well known graph analysis algorithm for finding shortest paths in a weighted graph.

A single execution of the algorithm will find the lengths (summed weights) of the shortest paths between all pairs of graph vertices.

To illustrate the shortest path problem and its decision, Fig.1 displays an exemplary weighted graph with five vertices and weighted edges. Fig.2 shows the adjacency matrix, which presents the graph - the value of any matrix element corresponds to the an edge weight and the number of row and column of this element show the numbers of adjacent vertices of this edge in the weighted graph.

The results of the problem decision we could see in the next Fig.3, presenting the shortest paths between every two graph edges, shown as values of the elements in resulting matrix:

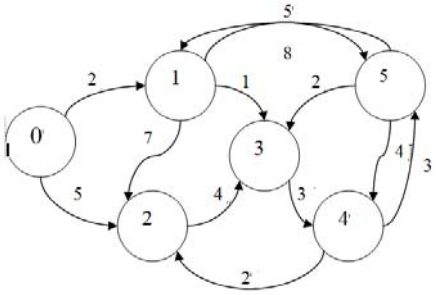

Fig.1. An exemplary weighted graph

P. Manoilov is with the Department of Computer Systems, Faculty of Computer Systems and Control, Technical University of Sofia, 8 Kliment Ohridski blvd., 1000 Sofia, Bulgaria, e-mail: pmanoilov@tu-sofia.bg

The Floyd–Warshall algorithm**,** could be easily parallelized. Fig. 4 presents the diagram of the algorithm with four branches, that run in parallel.

	0	1	2	3	4	5
0	0	2	5	–	–	–
1	–	0	7	1	–	8
2	–	–	0	4	–	–
3	–	–	–	0	3	–
4	–	–	2	–	0	3
5	–	5	–	2	4	0

Fig.2**.** The adjacency matrix presenting the exemplary graph

	0	*1*	*2*	*3*	*4*	*5*
0	0	2	5	3	6	9
1	–	0	6	1	4	7
2	–	15	0	4	7	10
3	–	11	5	0	3	6
4	–	8	2	5	0	3
5	–	5	6	2	4	0

Fig.3**.** The matrix with shortest paths between exemplary graph edges

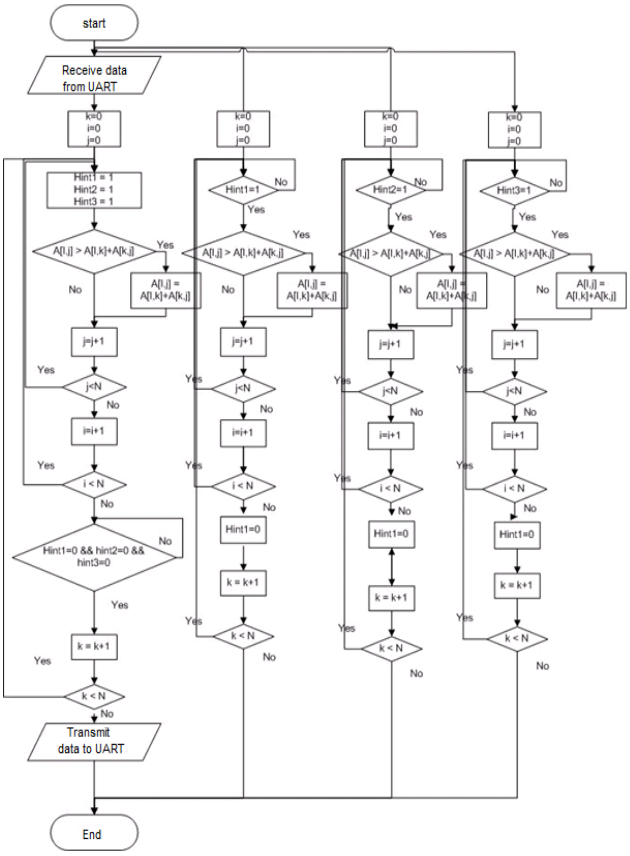

Fig.4**.** The diagram of the parallelized Floyd–Warshall algorithm

II. THE ARCHITECTURE OF THE MULTI-CORE SYSTEM ON CHIP

The multi-core system was designed and implemented on chip by means of an automated design system (Integrated Software Environment – ISE WebPACK of Xilinx Corp.)[3]. We have created therefor an input language description of system using specialized hardware description language – VHDL [4]. The multi-core system is prototyped and implemented on a FPGA chip – xc3s500e of Xilinx Spartan 3e family[5]. We are using the development board: Spartan 3e Starter Kit. The block scheme of the designed and implemented multi-core system is presented on Fig.5.

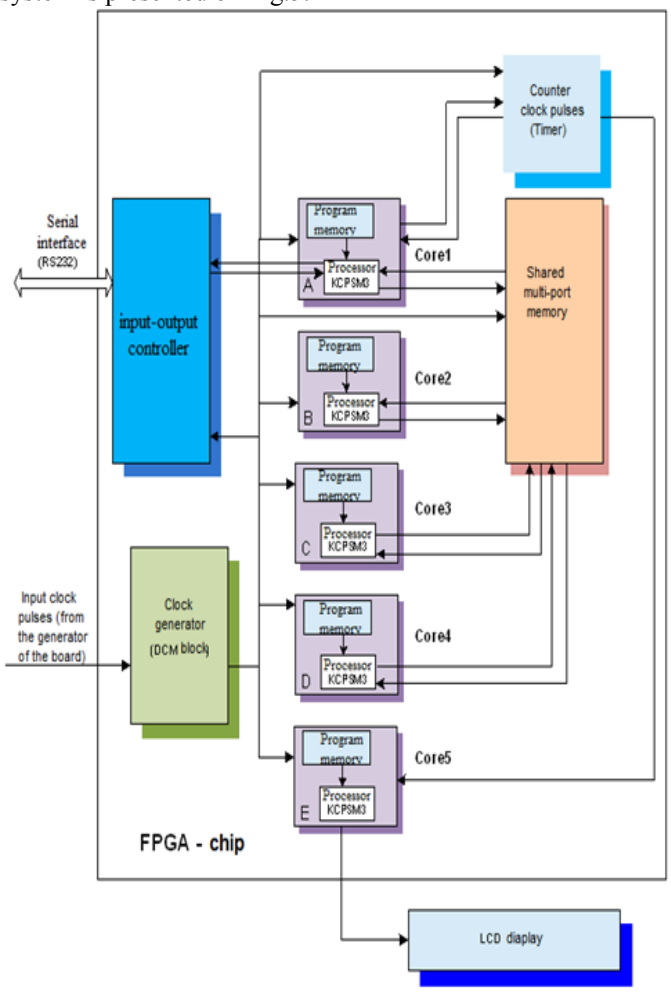

Fig.5. The block-scheme of the designed and implemented on FPGA multi-core system

The multi-core system includes five processor cores. The "operational" processor cores in the system (which execute the Floyd–Warshall algorithm calculations, for example) are four – with names Core1 to Core4 on Fig.5. The multicore system includes another (fifth) processor Core5, intended to control the LCD display of the development board.

The system architecture represents a symmetric multiprocessor, using common shared data memory [6]. Each processor Core1 to Core4 is connected to the common shared memory by own input and output memory data port. To meet the simultaneous processor requests, the data memory must have 4 input and 4 output ports – generally it possesses 8 data ports (Fig.6):

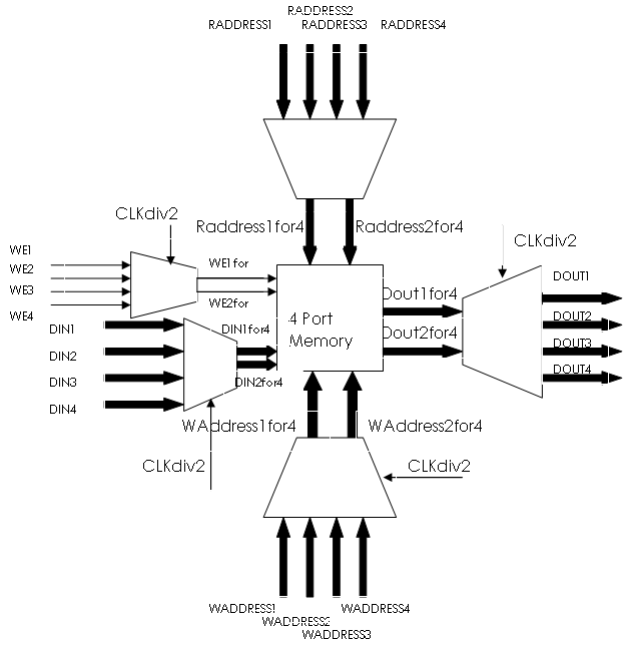

Fig.6. The block scheme of the multiport common data memory

To get a fast decision of shortest path problem we need a parallel processor system with fast execution of simple calculation operations – for adding, accumulating and comparing operations mainly. Therefore we have decided to use in the system architecture a simple, but fast soft processor – 8 bit KCPSM3 . Fig.6 presents the block – scheme of processor architecture.

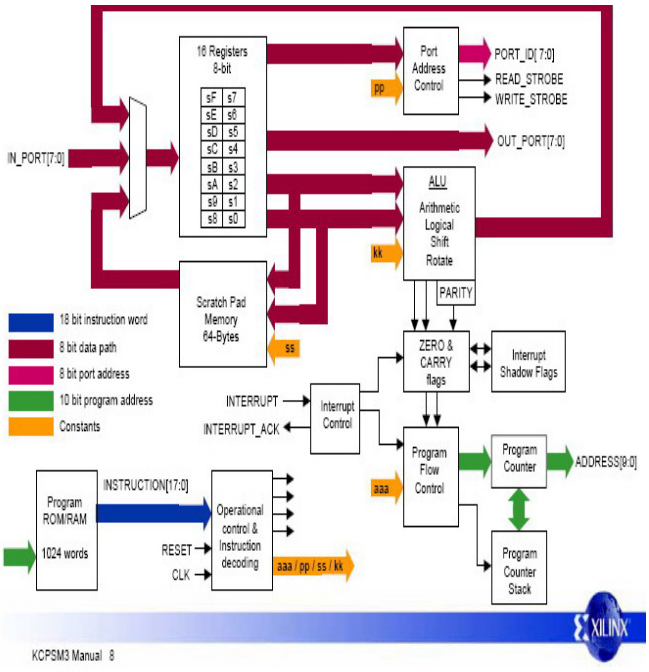

Fig. 6. Blok-scheme, prezenting the architecture of the used soft processor KCPSM3

The processor cores possess speed data registers and any its instruction is executed quickly - in only two periods of synchronization pulse.

The Input-Output (UART) Controller of multicore system (Fig.5**)** performs under the control of Core1 the receiving the arguments of executed task from Host computer (the adjacency graph matrix of shortest path task, for example and the transmitting the results (the matrix with shortest paths for same task) – back to the Host computer.

The system device "Counter of clock pulses (Timer)" registers the duration of "pure" time interval, spent for task execution, excluding the time, intended for data transfer between the System on Chip and Host – computer. The fifth processor core (Core 5) displays this value on the LCD display of the development board. Fig.7 shows the placement of the main system devices on the FPGA – chip blocks. The placement is displayed by means of specialized program in integrated design environment – Floorplanner.

III. The Results of Multi-core System Investigation

Table 1 shows the percent of FPGA chip resources, occupied for various processor core numbers (1,2,4) of the system on chip. The results were obtained from used design environment.

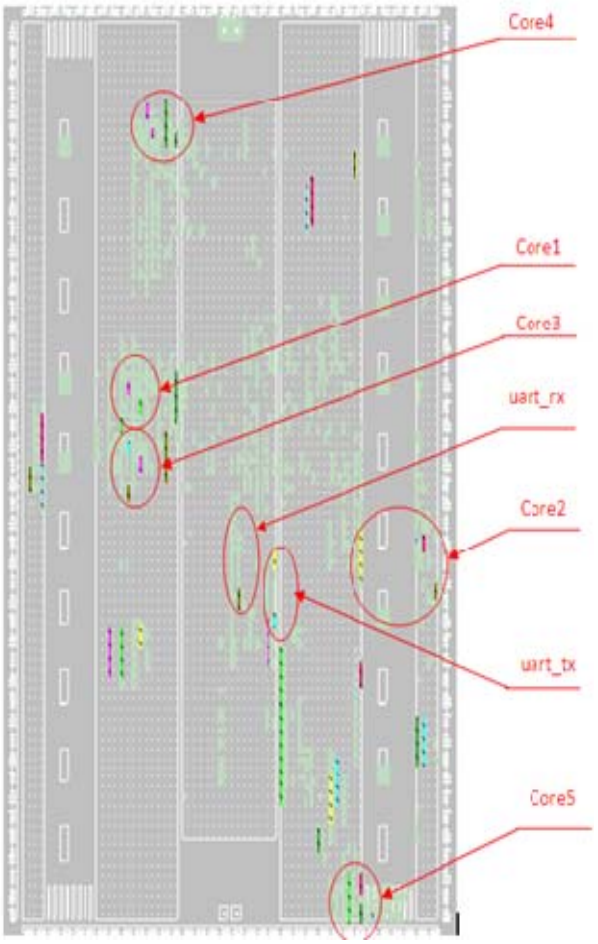

Fig.7. The placement of the main system devices on the used FPGA – chip area

TABLE 1**.** RESOURCES, USED ON THE FPGA CHIP ACCORDING THE INCREASING NUMBER OF CONNECTED PROCESSOR CORES:

	Number of Slices	Number of Slice Flip Flops	Number of 4-input LUTs	Number of bonded IOBs	Number of BRAMs	Number of GCLKs	Number of DCMs
1 processor	8%	3%	8%	31%	20%	20%	50%
2 processors	10%	4%	10%	43%	25%	20%	50%
4 processors	16%	7%	16%	68%	45%	20%	50%

The table shows, that we could implement twice as many processor cores on the used FPGA chip. A problem would, however, to design a multiport memory, which provides data ports, dedicated to each processor – 8 input and 8 output ports for 8 processors. Realized by multiplexing the data and address signals in time, this architecture of multiport memory significantly increases the time period and decreases the frequency of the processor cores clock signal. To realize the multi – core system with more than 4 processor cores it is more efficient to use a standard two –port data memory (with one input port and one output port) and a specialized device – memory controller, which is intended to resolve the memory request processor conflicts.

The next, Fig.8 and Fig.9 present the timing results of the investigated multi-core system functioning. These results were obtained by means of the system device - clock pulses counter (Timer – Fig.5**)**. The results show, that the time interval, spent for solving the shortest paths problem by multi-core system on chip, decreases in proportion to the number of processor cores used.

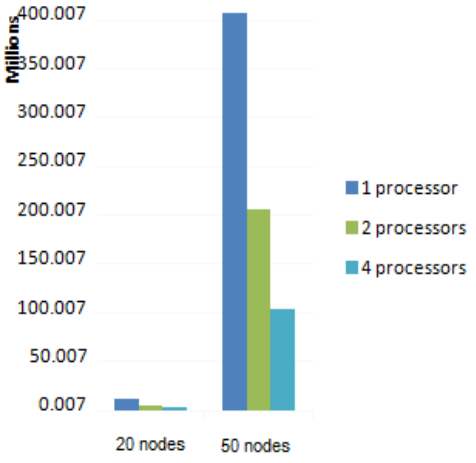

Fig.8**.** Time interval spent for task solving as a function of used processor number (for weighted graphs with 20 and 50 nodes)

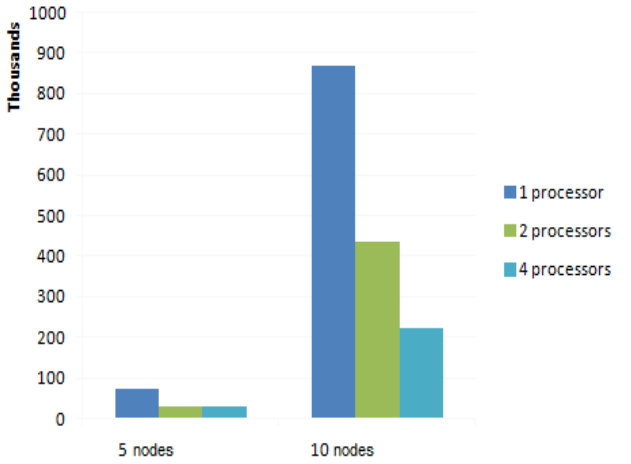

Fig.9. Time interval spent for task solving as a function of used processor number (for weighted graphs with 20 and 50 nodes)

TABLE 2. THE SPEEDUP OF MULTICORE SYSTEM

Number of graph nodes	2 processors	4 processors
5 nodes	1.452445285	2.575624291
10 nodes	1.983925916	3.881741402
20 nodes	1.987841362	3.864485711
50 nodes	1.979896084	3.915530908

The speed-up of the system in solving the shortest path problem was calculated and is shown on the TABLE 2**.** The speed-up increases in proportion to the number of processors used (with the exception of the tasks with a smaller volume of the input data, for example, less than 5 graph nodes).

IV. CONCLUSION

- The results of multi-core system investigation: the time interval for problem solving and speed-up, according the number of used processor cores of and the size of the weighted graph are close to optimal – its values vary in proportion to the number of used processor cores.

- These results are due to the chosen parallel system architecture, representing parallel multiprocessor. All the processor cores execute in parallel their programs to achieve the desired performance. The access to the shared data memory is performed simultaneously and independently of each other. Conflicts do not exist, which is the reason the executed of multi-core system tasks to achieve the desired performance.

REFERENCES:

[1] https://www.cs.princeton.edu/~rs/AlgsDS07/15S shortestPaths.pdf.
[2] Kapyris G. – *Introduction to Parallel Computing, Graph Algorithms,*- www-users. cs.umn.edu/ ~karypis/ parbook /Lectures/GK-CS5451/Chapter%2010%20-%20Graph%20 Algorithms.pdf.
[3] http://www.xilinx.com/products/design-tools/ise-design-suite/ise-webpack.htm.
[4] Маноилов П. – *Проектиране на цифрови устройства върху свръхголеми интегрални схеми с помощта на VHDL,* Изд. На ТУ – София, 2010.
[5] http://www.xilinx.com/support/documentation/ data_sheets/ ds312.pdf
[6] http://www.princeton.edu/~achaney/ tmve/ wiki100k/ docs/Symmetric_multiprocessing.html
[7] http: //lniv.fe.uni-lj.si/ courses/div/ KCSM3_Manual.pdf