FPGA chips as a hardware simulator for multiplane $\log_2$N switching fabrics

Marek Michalski,
Poznan University of Technology, Faculty of Computing and Telecommunications, ul. Polanka 3, 60-965 Poznań, Poland

***Abstract*—In this paper there is presented a system for simulations realized in hardware. The subject are blocking states in optical switching fabrics. Model of such a fabric is presented, and the way of its analysis is described. FPGA Spartan-3 chips are used for fast calculations, Raspberry PI, small PC, is used as an interface between PC and electronic part of the system. System is dedicated for searching blocking states (which is realized in hardware) and their analysis (which is realized by GUI and software on PC). Main elements of system are: Web based GUI, scripts and database for storing results, subsystem for controlling FPGA chips (controller is realized on Raspberry PI and its GPIOs) and 18 (or more) FPGA modules as a calculating engines.**

I. Introduction

In the switching fabrics, connections are set up between inputs and outputs in order to transmit information. Nowadays, most connections are realized by packets or frames of different protocols. So, each connection is dedicated for one packet, it is very short, hence, the number of such connections can be huge, what is next, control algorithms has a lot of work. Control algorithms have to be efficient in time, energy consuming and many more dimensions. In this paper I will present a way to analyze switching fabric states in order to construct more efficient algorithms. This paper is focused on $\log_2$N switching fabrics, but presented rules and mechanisms can have more wider application.

II. Description of switching fabrics, algorithms

The $\log_2$N switching fabrics have been widely discussed in the literature [1]–[9]. Generally, they are composed of 2×2 basic switches organized in sections and planes. Such fabrics are self-routing, it means, path between each input and each output is determined by input and output number. So, there is no choice about path. The only, and very important choice, is to choose a proper plane. Paths between different input-output

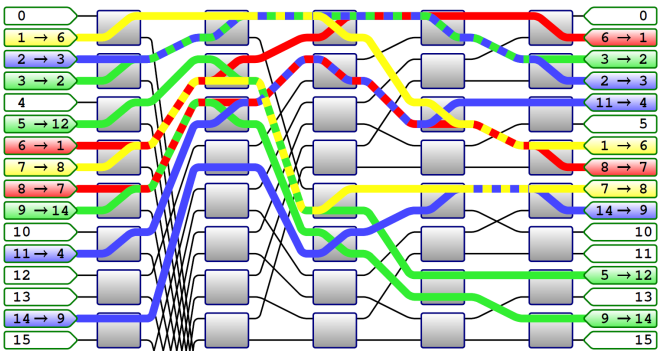

Fig. 1. Model of switching fabrics with connections realized in different planes (1st half of fabric with N=32 inputs)

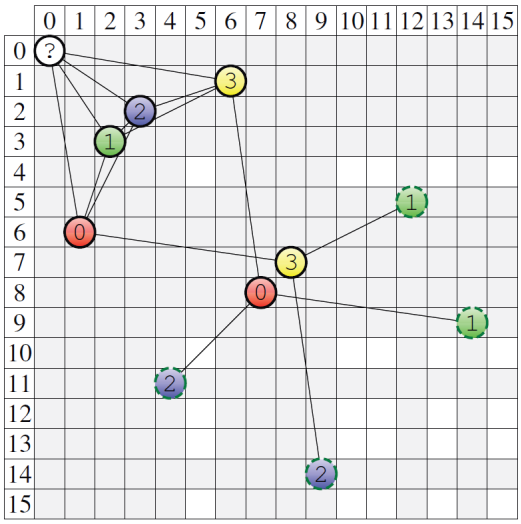

Fig. 2. Connections as nodes in graph in connection matrix G.

pairs in each plane can use different or the same interstage links. So, it is possible to block path for some connection in some plane, then we have to realize connection in different plane. But, what is bad case, it is possible to block paths for some connection in all available planes. Control algorithms should avoid such situations or minimize their probability. That is why different and complicated control algorithms have been developed for these fabrics. Good algorithm allows to reduce resources and obtain better results (i.e. loss probability). In my work, I will focuss on Log$_2$N switching fabrics. Example of such a fabric is presented in Fig. 1. There is shown upper half (inputs and outputs from 0 to 15) of fabric with 32 inputs (and also 32 outputs). We see connections marked with different colors. Each color corresponds to different plane. We see, that some interstage links are used by more then one connection, so, they have to be realized in different planes. If there is no available color (i.e. available plane) for some connection on a whole its path, such a connection is blocked. In this case we assumed that we have fabric with 32 inputs (and outputs) and it has 4 planes (It is known that 4 planes should be enough to obtain nonblocking with good algorithm [3]). Here we can see blocking state for connection 0-0, because there is a conflict on path with another connections in each possible plane. So, all of planes in this fabric, are not available for new connection. We can rearrange existing connections (move them to another planes) to unblock at least one plane for new connection or new connection will be lost. Another approaches assume different reactions, but in this paper we will focus on analysis of internal states of fabric and system for efficient analysis will be presented.

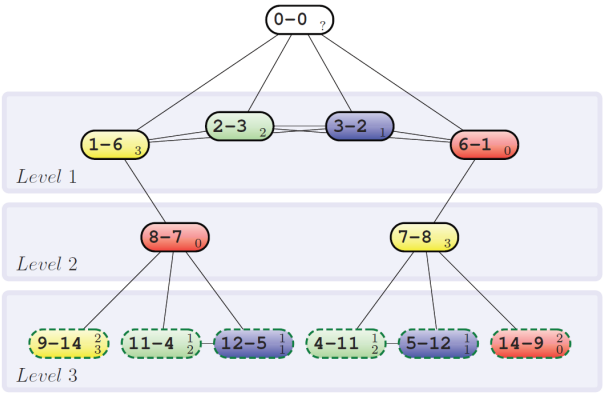

Fig. 3. State graph for blocked connection 0-0 in fabric with N=32 inputs

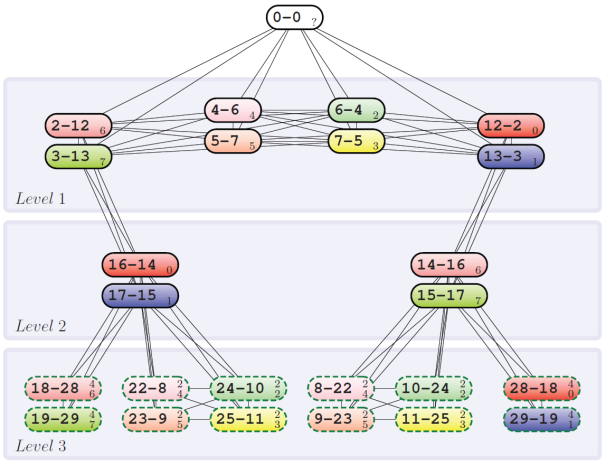

Fig. 4. State graph for blocked connection 0-0 in fabric with N=128 inputs

For better and easier analysis of situation of given connection, state graph has been developed. Example of such a graph is show in the Fig. 2. As a nodes in this graph, existing connections are treated. Edges represents blocking relations, i.e. connected nodes represents connections which are blocking each other, it means - they have to be realized in different planes. Now, we could mention graph coloring problems as a solution, but they are complicated and not efficient in this case. An efficient algorithm has been develop and presented for nonblocking fabric [3], [8], [9]. In blocking fabrics, which are cheaper, we have to accept possibility of losses, but with clever algorithms, we would like to minimize them. And this system allows us to analyze situation and construct more efficient solutions. In Fig. 3 and 4 we see graphs state for connection 0-0 which is blocked in fabrics with 32 and 128 inputs respectively. Internal state is represented by matrices. Each plane has its own set of 3 matrices, which are used for calculations. Matrix A stores information about existing connections in plane. Value 1 on a position $[x, y]$ represents connection form input $i/2$ to output $j/2$. Due to some mechanisms, it is possible to reduce size of matrices, because we can treat connection $< i, j >$ as a connection from input switch to output switch $< x, y >$, where $x = i/2$, and $y = j/2$. Each element of matrix B is calculated by a complicated formula and informs about availability of this plane for connection $< x, y >$. Matrix C represents costs of realization certain connection in this plane, by comparison costs of the same connection in different planes, algorithm can choose the cheapest one, i.e. the optimal solution. Fig. 6 shows content of matrices which represent state, when connections from Fig. 5 are realized. Pictures come from software which is a graphical user interface for this system, they contain additional information (hidden in colors, lines and other tricks).

III. Goals of this system

It is possible to create huge amount of examples of internal state. To analyze all of them, dedicated software can be used. But time for execution of such a software can be not acceptable. So, I found a way, to generate and analyze state in hardware chip, sort and filter them, and only interesting ones are sent to PC for further analysis.

IV. Software and hardware elements of system

System is dedicated for searching states with given features (i.e blocking states). It has GUI based on web application [10]. This applications has connection to database, where parameters of test and their results are stored and available for future usage. GUI sends parameters to FPGA chips, they are programmed to perform exactly these calculations which leads to results. They work very fast, in one clock cycle, they can generate and analyze one example of switching fabric state.

A. FPGA chip - Spartan-3

In this system Spartan-3 (XC3S200 [11]) is used. In our laboratory we have more than 20 developers modules with this chip (one of them is presented in Fig. 7). In each chip XC3S200 there are available more than 170 user IOs, but in

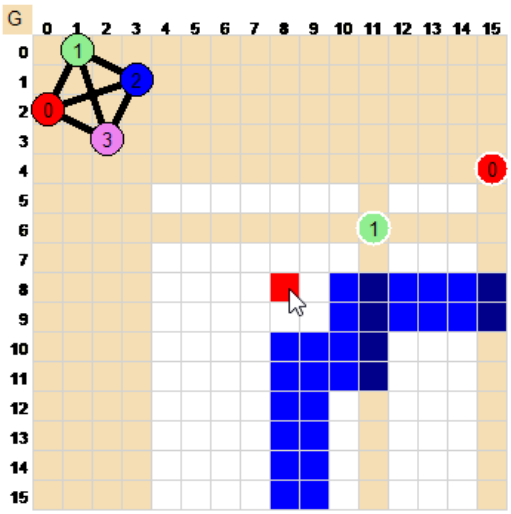

Fig. 5. Matrix G for fabric with N=16 inputs

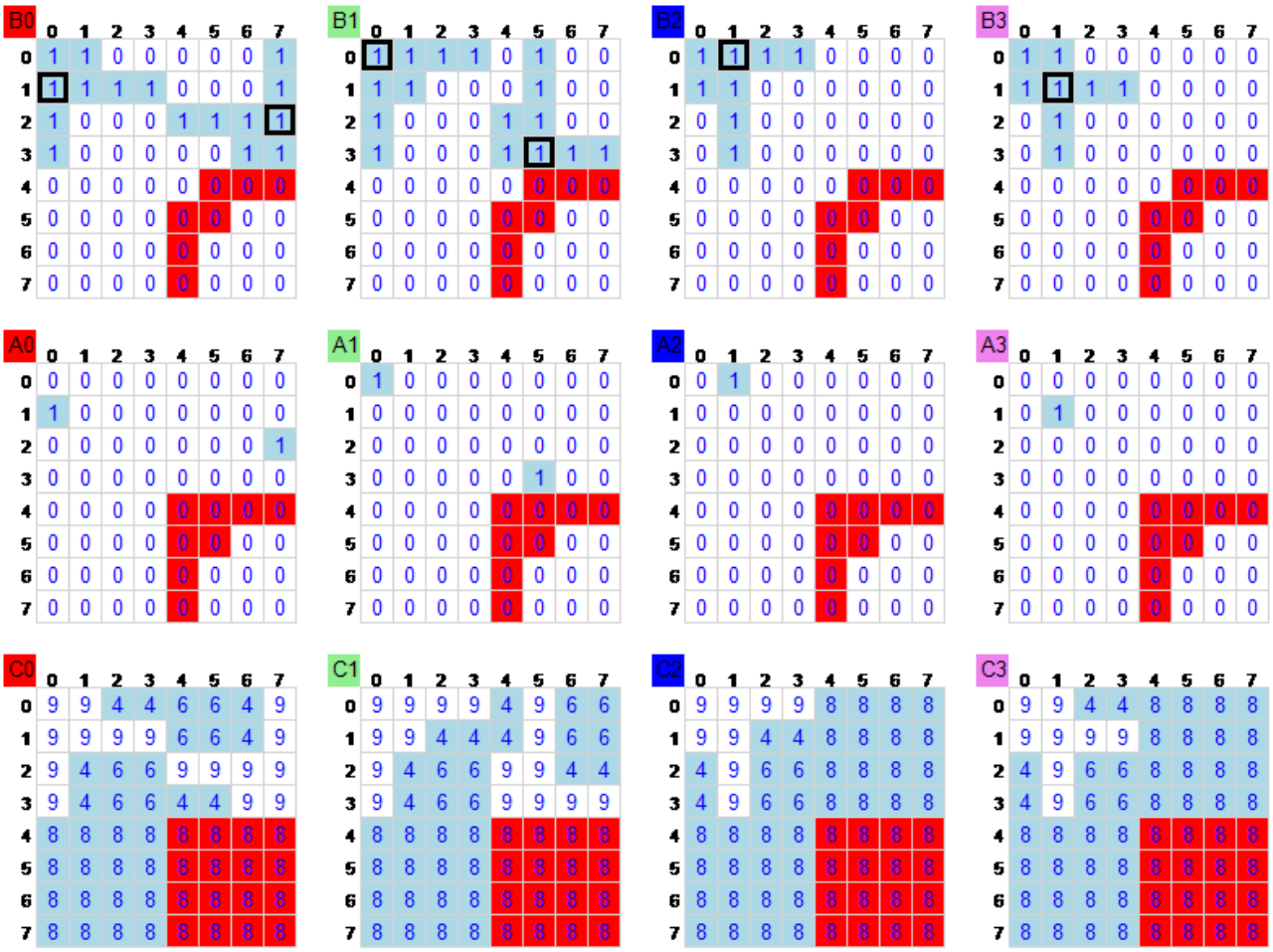

Fig. 6. Matrices A, B, and C for 4 planes with switching fabric state representation (fabric with 16 inputs and connections the same as from Fig 5).

our board [12], [13] only 62 of them are connected to pins. In chip there are also 200 000 programmable gates (4320 logical cels), all of them are available for our designs. They have clock with f=50Mhz. All variables represented in matrices A, B and C (see Fig. 6) necessary to realize calculations are implemented in FPGA and run as a working hardware design in Spartan-3 chips [14]–[16].

B. Raspberry PI

The very important element of this system is Raspberry PI - a small computer (see Fig. 7), but it allows to connect programmability with electronics. It works as a proxy between programmable software and programmable hardware. It has Ethernet connection with GUI (by RJ43 and cable or WiFi) and general purpose input output pins, which are connected to FPGA chips.

V. System Architecture

Software part, which is run on a typical PC, realizes GUI, functionality for parameters and results storing and sending to FPGA modules via proxy. In order to do this, it communicates with Raspberry PI, which has physical connection with one

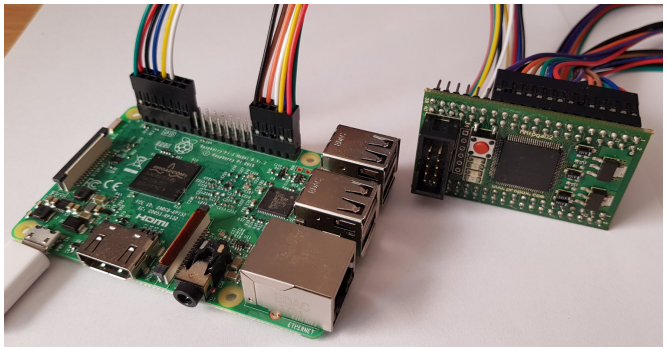

Fig. 7. Raspberry PI and first FPGA module.

FPGA chip. In whole system 18 FPGA chips are used, they are connected in a row (see Fig. 8). Each of them has its unique ID, whole control information are sent to each of them, but only chip with given ID (which is treated as an address) is sensitive on proper commands. This is the way to communicate with FPGA nodes. Logical architecture od this system is presented in Fig. 8.

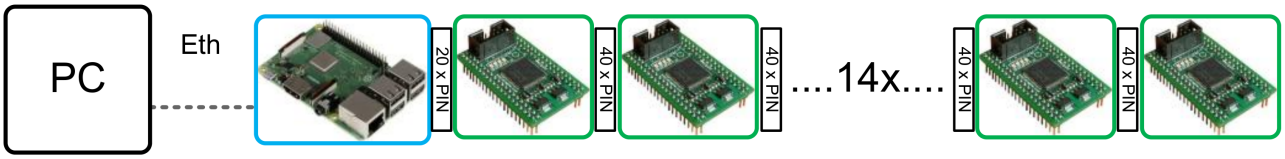

Fig. 8. Logical architecture of system

VI. Example of usage and results

The first purpose od this system was finding blocking states and prepare them for further analysis. Additional usage can assume calculation of relation between states with blocking and without blocking. In this case we obtain typical simulator. In Fig. 9 there is presented chart with several curves which represents blocking probability for different algorithms. Each point of this chart has been calculated by different node of this system. Here we see 8 curves for 4 different algorithms (the same color means the same algorithm). Results presented here are without statistical treatment, moreover, they are gathered in very early state of calculation, so, they are not accurate. It was intentional to show not overlapping points coming from different nodes.

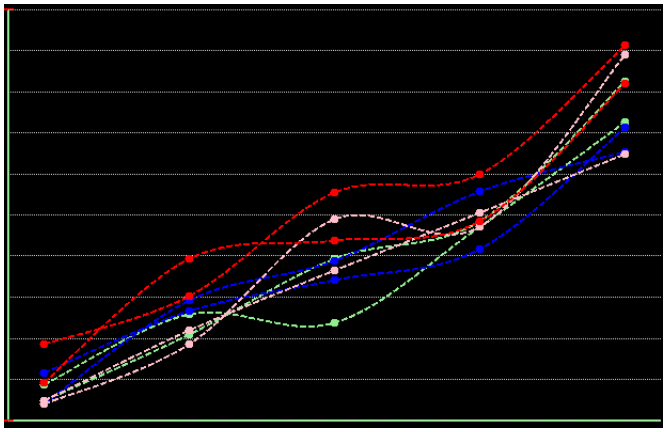

Fig. 9. Chart with relation of blocking and not-blocking states for different algorithms and loads.

VII. Conclusion and future works

This system allows to find huge amount of blocking states in a relatively short time. It is much faster than software run on a PC, it uses hardware speed of FPGA chips [17]. This system can be extended by additional hardware modules and new features and cooperation with other similar tools [18], [19]. I plane to move ideas presented here to bigger and newer chips (Virtex, ZYNQ) and also analyze different algorithms for different switching structures.

Acknowledgements

The work described in this paper was financed from the funds of the Ministry of Science and Higher Education for year 2020.

References

[1] W. Kabaciński and M. Michalski, "Wide-sense nonblocking $\mathrm{Log}_2(N,0,p)$ switching networks wiht even number of stages," in *Proc. IEEE ICC 2005*, Seoul, South Korea, May 2005.
[2] W. Kabaciński and M. Michalski, "Lower Bounds for WSNB Multi-$\mathrm{Log}_2 N$ Switching Networks". *Conference on Telecommunications A-ICT 2005*, (Lisbon, Portugal), pp. 202–206, July 2005.
[3] W. Kabaciński and M. Michalski, "The routing algorithm and wide-sense nonblocking conditions for multiplane baseline switching networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 35–44, December 2006.
[4] G. Danilewicz, W. Kabaciński, M. Michalski, M. Zal, "A new control algorithm for wide-sense nonblocking multiplane photonic banyan-type switching fabrics with zero crosstalk", *IEEE Journal on Selected Areas in Communications* vol 26, no. 3, part 2, pp. 54–64, April 2008.
[5] W. Kabaciński, J. Kleban, M. Michalski, M. Zal, A. Pattavina, G. Maier: Rearranging Algorithms for $\mathrm{Log}_2(N,0,p)$ Switching Networks with Even Number of Stages. *International Workshop on High Performance Switching and Routing*, Paris, France June 22 - 24, 2009.
[6] W. Kabaciński, M. Michalski: The FPGA Implementation of the $\mathrm{Log}_2(N,0,p)$ Switching Fabric Control Algorithm, *IEEE International Conference on High Performance Switching and Routing 2010*, Dallas, TX, USA 13-16 June 2010.
[7] W. Kabaciński, M. Michalski: The FPGA Controller for the Rearrangeable $\mathrm{Log}_2(N,0,p)$ Fabrics with an Even Number of Stages, *IEEE International Conference on High Performance Switching and Routing*, HPSR 2011, Cartagena, Spain 4-6 July 2011.
[8] W. Kabaciński, M. Michalski: The Algorithm for Rearrangements in the $\mathrm{Log}_2(N,0,p)$ Fabrics with Odd Number of Stages, *IEEE International Conference on Communications ICC2011*, Kyoto, Japan 5-9 June 2011.
[9] W. Kabaciński, M. Michalski: The Control Algorithm and the FPGA Controller for Non-interruptive Rearrangeable $\mathrm{Log}_2(N,0,p)$ Switching Networks. *IEEE International Conference on Communications*, ICC2013, Budapest, Hungary 9-13 June 2013.
[10] M. Michalski: *A software and hardware system for a fully functional remote access to laboratory networks*, The Fifth International Conference on Networking and Services, Valencia, Spain, April 20-25, 2009.
[11] https://www.xilinx.com/support/documentation/data_sheets/ds099.pdf
[12] http://www.propox.com/download/docs/MMfpga02_pl.pdf
[13] propox.com/download/docs/MMfpga12_Instrukcja_Programowania.pdf
[14] "Xilinx ISE 10.1 Quick Start Tutorial",
Available at: *http://www.xilinx.com/itp/xilinx10/books/docs/qst/qst.pdf*.
[15] ISE In-Depth: *http://www.xilinx.com/direct/ise10_tutorials/ise10tut.pdf*.
[16] Xilinx Platform Studio; http://www.xilinx.com/tools/xps.htm.
[17] M. Michalski: The System for Delay Measurement in Ethernet Networks on NetFPGA Cards, IEEE International Conference on High Performance Switching and Routing 2014, Vancouver, Canada 1-4 July 2014.
[18] M. Michalski, The Configurations for Experimental Study of the Network Performance, *8th IEEE, IET International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP 2012)* 18-20 July 2012.
[19] IP Pool manager, *8th IEEE, IET International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP 2012)* 18-20 July 2012.