The Erlangen Slot Machine – A Platform for Interdisciplinary Research in Dynamically Reconfigurable Computing

ESM – Eine Hardware-Plattform für interdisziplinäre Forschung im Bereich des dynamischen rekonfigurierbaren Rechnens

Josef Angermeier, Diana Göhringer, Mateusz Majer, Jürgen Teich, University of Erlangen-Nürnberg,
Sándor P. Fekete, Jan C. van der Veen, University of Braunschweig

Summary We introduce a hardware platform called Erlangen Slot Machine (ESM) that has been built in Erlangen within the project ReCoNodes for enabling interdisciplinary research on reconfigurable computing. For this dynamically reconfigurable computer, the cooperation partner in Braunschweig provides algorithmic solutions, in particular for the optimization of module placements and inter-module communication.

►►► **Zusammenfassung** Eine Hardware-Plattform mit Namen Erlangen Slot Machine (ESM) wird beschrieben, die im Projekt ReCoNodes in Erlangen entstanden ist. Sie soll dazu dienen, methodische Ansätze und Anwendungen anderer Projekte zu testen und interdisziplinär zugänglich zu machen. Der Kooperationspartner in Braunschweig liefert algorithmische Lösungen für diesen dynamisch rekonfigurierbaren Rechner, insbesondere zur Optimierung der Modulplatzierung und der Kommunikation zwischen Modulen.

KEYWORDS B.7 [Hardware: Integrated Circuits], C.1 [Computer Systems Organization: Processor Architectures], C.3 [Computer Systems Organization: Special-Purpose and Application-Based Systems], reconfigurable computing, System-on-a-Chip, dynamic hardware reconfiguration, adaptive computing

1 The Erlangen Slot Machine

1.1 Terminology: Static/Dynamic Versus Full/Partial Reconfiguration of Hardware

FPGAs (field-programmable gate arrays) denote a class of hardware devices that may be customized by means of (re)configuration. In case of SRAM-based devices, this is possible by loading a bitstream into the device. Such devices can be programmed either once (statically) before being deployed in the field. Alternatively they may be programmed as often as necessary (dynamically) at run-time. Orthogonal to this classification is the question whether only the whole device is programmed as one entity (full reconfiguration), or whether just parts of the device are reconfigured individually (partial reconfiguration).

Before a device is operational after reconfiguration, a certain time elapses, often called *reconfiguration time*. These different terms of reconfiguration are illustrated in Fig. 1.

1.2 Why a New Platform?

Very few FPGAs exist on the market that allow partial reconfiguration. For example, in Xilinx Virtex-II devices, the smallest possible entity

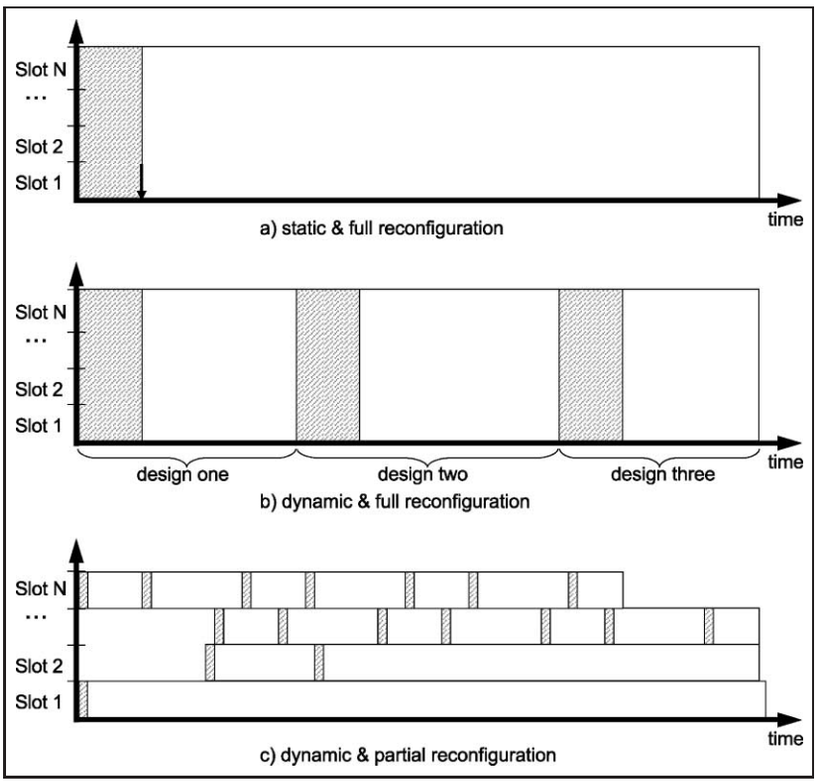


Figure 1 Terminology: a) static vs. b) dynamic full reconfiguration; c) dynamic partial reconfiguration supported by the ESM platform: individual slots may be reconfigured independently from each other. This gives a higher flexibility and reduces reconfiguration times (dashed time intervals).

of reconfiguration is a full column. Mainly due to this and the following factors, the practical realization and use of partial and dynamic reconfiguration on current existing FPGA-based computing platforms is highly restricted.

- **I/O-pin dilemma:** Most existing FPGA-based platforms have peripherals such as video, audio, A/D-converters and D/A-converter devices connected to fixed pins of the FPGA. This makes the free relocation of hardware modules or hardware tasks on the device tedious, if not impossible. The same holds for external memory devices such as RAMs, etc. This situation is present on Celoxica [1], XESS [2], Nallatech [3], as well as on Alpha boards [4]. On the XF-Board [5; 6] from ETH Zurich, the peripherals are connected to one side of the device: each module accesses its peripherals through an operating system (OS) layer implemented on the left and right part of the device. Other existing platforms like RAPTOR [7], Celoxica RC1000 and RC2000 [1] are PCI systems that require a workstation for operation. The use in stand-alone systems as needed in embedded systems is not possible.
- **Inter-module communication dilemma:** Modules placed at run-time on the device typically need to exchange data among each other. Such requests for communication are dynamic due to module placement at run-time. The dynamic routing of signal lines on an FPGA is a very cumbersome task. For efficiency reasons, new communication paradigms must be investigated to support such dynamic connection requests, such as packet-based *DyNoCs* [8], or self-circuit routing [9] mechanisms.
- **Memory dilemma:** Modules requiring larger amounts of local memory cannot be implemented, because a module can only occupy the memory inside its physical slot boundary. Storing data in off-chip memories is therefore the only solution.

With all the above limitations in mind, we designed a new FPGA-based reconfigurable computer called *Erlangen Slot Machine* (ESM). Its architecture circumvents many of the problems mentioned and is described next, see [10] for more details. We also developed a comprehensive model and method for the integrated problem of optimized scheduling, placement, and communication on ESM-like architectures; see [11].

2 Architecture of the ESM

The Erlangen Slot Machine consists of a BabyBoard mounted on a MotherBoard, see Fig. 2. The clustering of the system into two boards allows the BabyBoard containing the reconfigurable FPGA to be re-used for a wide variety of application domains, such as multimedia and automation [10]. For the integration of the ESM into a new domain such as, e. g., automotive, no redesign of the complete system is necessary. Only a new MotherBoard must be provided according to the computational and peripheral requirements in the new environment, e. g., A/D-converters, CAN- and FlexRay-controllers, PWM, etc.

In the following, we provide a detailed description of each of the board's features.

2.1 The BabyBoard

Computation and Reconfigurable Engine

The reconfigurable engine of the ESM platform is a BabyBoard that features a Xilinx Virtex-II 6000 FPGA, several SRAMs and a configuration circuit. The ESM owes its name from the organization of reconfigurable modules in a one-dimensional array of slots $S_1, \ldots, S_N$, see Fig. 3. The reason for the slot or-

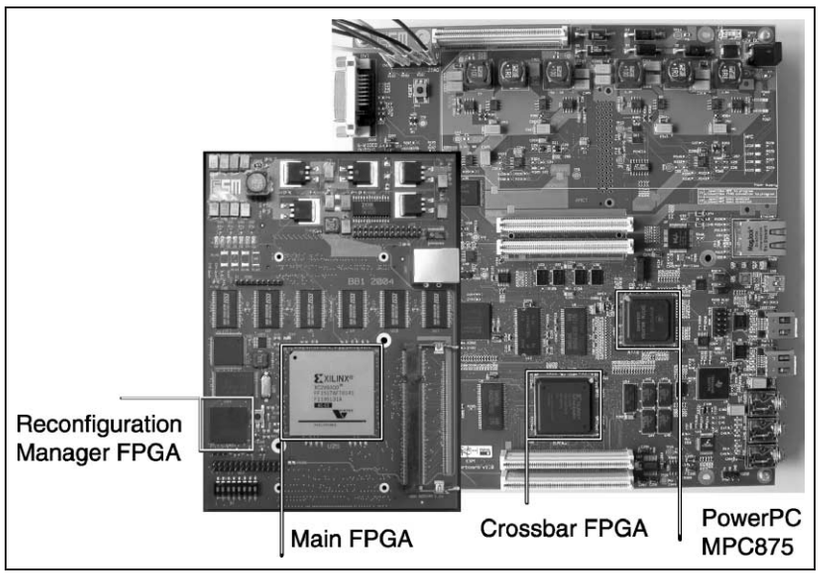


Figure 2 ESM computer: View on BabyBoard and MotherBoard.

ganization is that Virtex-II FPGAs only offer partial reconfiguration support for full columns. In the following, we describe how many deficiencies of existing platforms were avoided in the ESM computer architecture.

Solving the I/O-pin dilemma: On-line placement of modules on the FPGA is achieved by downloading a partial bitstream that implements a given hardware task in the FPGA. This requires a relocation that places a module in a location different from the one for which it was compiled. Such a *module relocation* can be accomplished only if all the resources in the target destination (e. g., slot S3) are free and structured in the same way as for the synthesized (source) destination (e. g., slot S1). This includes also the device pins used by the module. We solved the I/O-pin dilemma on the ESM by avoiding fixed connections of peripherals to pins of the FPGA. As shown in Fig. 3, all the bottom pins of the FPGA are connected to an external *crossbar*, and all the peripherals located on the MotherBoard are connected to the crossbar.

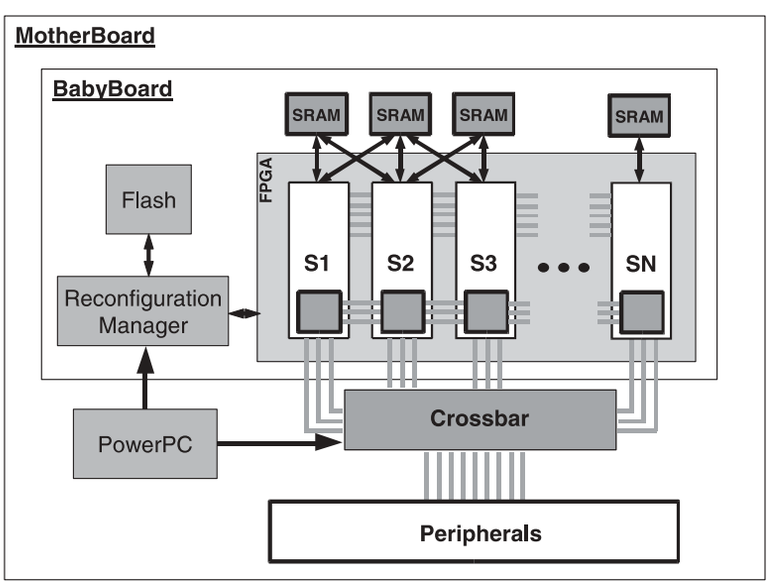


Figure 3 Architecture of the ESM computer.

Solving the inter-module communication dilemma: In Section 3, we present different solutions to the inter-module communication dilemma, once the architecture of the ESM computer has been fully described.

Solving the memory dilemma: Six SRAM banks, each of size 2 MBytes, are vertically attached to the board on the top side of the device, thus providing enough memory to six different slots for temporal data storage. The SRAMs can be also used for shared memory communication between adjacent modules, e. g., for streaming applications. They are connected to the FPGA in such a way that the reconfiguration of a given module will not affect the access to other modules.

The Reconfiguration Manager

Apart from the main FPGA, the BabyBoard also contains the configuration circuitry. It consists of a CPLD, a configuration FPGA (a small Spartan-II FPGA) and a Flash.

- The CPLD is used to download the Spartan-II configuration from the Flash upon power-up. It also contains board initialization routines for the on-board PLL and the Flash.
- The reconfiguration management is implemented on the Spartan-II FPGA. This device contains a circuit to perform module relocation while loading a new partial module bitstream.
- The flash memory provides a capacity of 64 MBytes, thus enabling the storage of up to 32 full configurations and a few hundred partial module bitstreams.

Debugging

Debugging capabilities are offered through general purpose I/Os provided at regular distances between the slots. A JTAG port provides de-

bug capabilities for the main FPGA, the CPLD, and the Spartan-II.

2.2 The MotherBoard

The MotherBoard provides programmable links from the FPGA to a lot of peripherals for multimedia and external communication, such as IEEE1394, USB, Ethernet, PCMCIA, video respectively audio-I/Os. The physical connections are established at run-time through a programmable crossbar implemented statically on a Spartan-II chip on the MotherBoard. Video capture and rendering interfaces, as well as high speed communication links also exist on the MotherBoard, on which the BabyBoard is mounted using four connectors (see Fig. 2). A PowerPC processor (MPC875) running Linux is the core of the MotherBoard. It is used to control the complete ESM. In particular, it manages the flow of data on the MotherBoard as well as the interfaces to the external world, e. g., Ethernet and USB. Upon start-up, the user can log into the ESM like a conventional computer system. The PowerPC of the ESM is used for a) application development, and b) for the development of new placement and scheduling strategies, e. g., operating system functions for module management.

3 Inter-module Communication

One of the central limiting factors for the wide use of partial dynamic reconfiguration that has yet to be addressed is the problem of intermodule communication. Each module that is placed in one or more slots on the device must be able to communicate with other modules.

3.1 Paradigms for Communication

For the ESM, we provide four main paradigms for communication among different modules (see Fig. 4): The first one is a direct communication using bus macros between adjacent modules (see Fig. 4a). Secondly, shared memory communication using SRAMs or BlockRAMs is possible (see Fig. 4b). However, only adjacent modules can use these two communication modes. For modules placed in non-adjacent slots, we provide a dynamic signal-switching communication architecture called reconfigurable multiple bus (RMB) [9] (see Fig. 4c). Finally, the communication between two different modules can also be realized through the external crossbar (see Fig. 4d).

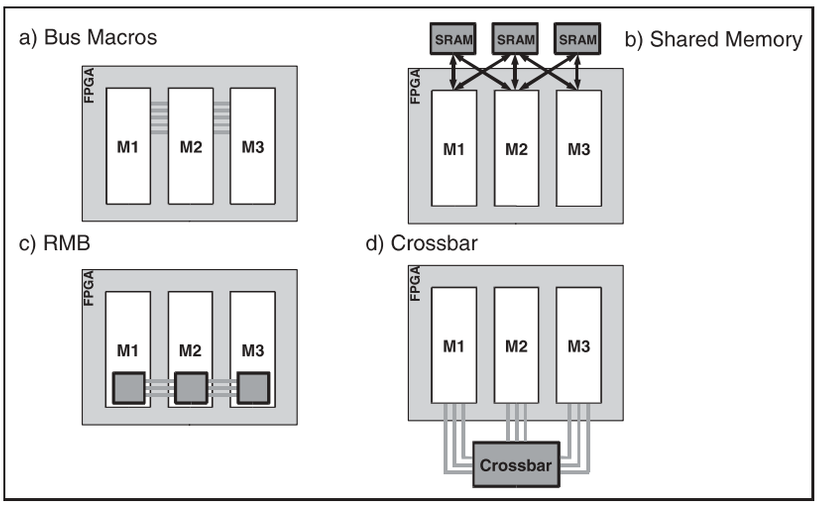


Figure 4 Intermodule communication possibilities on the ESM: a) bus macro, b) shared memory, c) reconfigurable multiple bus (RMB), d) external crossbar.

Adjacent Communication

On the ESM, bus macros are used to realize a direct communication between adjacent modules, providing fixed communication channels that help to keep the signal integrity upon reconfiguration. Because only eight signals can be passed for each bus macro, the number of bus macros needed for connecting a set of n signals between two placed modules is $\lceil n/8 \rceil$.

Communication via Shared Memory

Communication between two neighboring modules can be done in two different ways using shared memory: First, dual-ported BlockRAMs can be used for implementing communication among two neighbor modules working in two different clock domains. The sender writes on one side, while the receiver reads the data on the other side. The second possibility uses external RAM. This is particularly useful in applications in which each module must process a large amount of data and then sends the processed data to the next module, as it is the case in video streaming. On the ESM, each SRAM bank can be accessed by the module placed below as well as those neighbors placed right and left. A controller is used to manage the SRAM access. Depending on the application, the user may set the priority of accessing the SRAM for the three modules.

Communication via RMB

The Reconfigurable Multiple Bus (RMB) architecture [9; 12; 13] consists of a set of processing elements or modules, each possessing an access to a set of switched bus connections to other processing elements. The switches are controlled by connection requests between individual modules. The RMB is a one-dimensional arrangement of switches between N slots (see Fig. 5). In our FPGA implementation, the horizontal arrangement of parallel switched bus line segments allows for the communication among modules placed

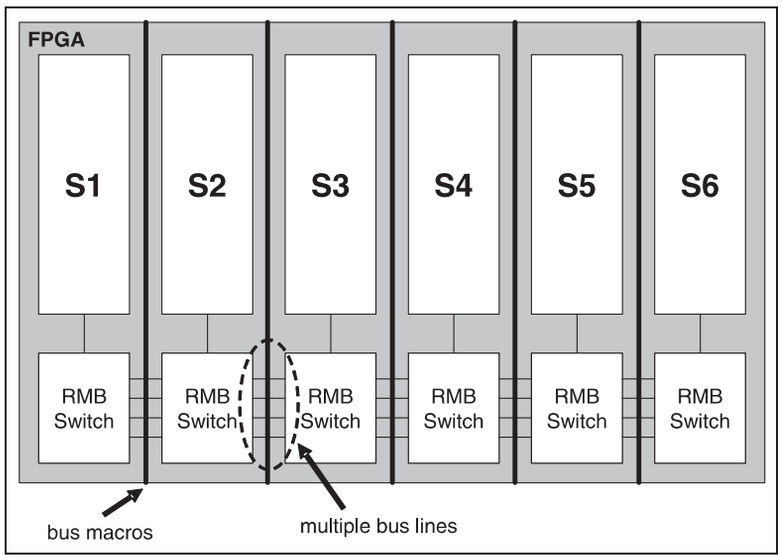


Figure 5 FPGA implementation of the RMB for partial reconfiguration.

in the individual slots. In [14], we describe a method for finding a module placement on the ESM that minimizes the number of bus lines, i. e., the width of the ESM for a given set of modules with communication requirements.

In our implementation of the RMB, we separated the switching controller from the modules. In this way, we provide a uniform interface to designers for connecting their modules to the bus. The implementation of the RMB structure on an FPGA Virtex-II 6000 with four processors and four parallel bus segments of width 16 bit results in an area overhead of 4% with a frequency of 120 MHz on the controller [9]. As shown in Fig. 5, bus macros are used at the boundary of modules and controllers to ensure a correct operation upon reconfiguration.

Communication via Crossbar

Another possibility of establishing a communication among modules is to use the crossbar. Because all the modules are connected to the crossbar via the pins at the south of the FPGA, the connection of signals between two modules can also be set in the crossbar.

3.2 Performance Numbers

Table 1 presents performance numbers for inter-module communication characteristics based on the first prototype of the ESM platform. Maximum achievable bandwidth, communication delay and setup times are shown for the four currently supported types of inter-module communication. The setup times denote the number of clock cycles needed to establish a communication link in the worst case before being able to send or communicate data over this link. In case of an RMB, this value depends linearly on the number of existing slots.

Table 1 Maximum achievable bandwidth, communication delay, and setup times of the four inter-module communication types currently supported on the ESM platform.

Type	Bandwidth	Delay	Setup time
Bus macros	19.2 Gbits/s	2 ns	none
RMB	6.4 Gbits/s	3 ns * #Slots	4 cycles * #Slots
Crossbar	1.8 Gbits/s	15 ns	18 cycles
SRAM	0.4 Gbits/s	20 ns	2 cycles
SDRAM	84.4 Mbits/s	55 ns	20 cycles

4 Tool Support

We currently developed a tool called *SlotComposer* for automated communication and synthesis flow generation for partially reconfigurable hardware modules. Using the existing Xilinx PR Tool Flow, *SlotComposer* inserts slice-based *bus macros* between adjacent modules, or, if specified, connects partially reconfigurable modules to the *Reconfigurable Multiple Bus* (*RMB*) communication infrastructure. At the same time, it automatically generates all necessary constraint files and optimizes the usage and placement of bus macros. Finally, it instantiates all intermediate communication signals.

5 Further Information

Further information on the platform, including applications, is available at *http://www.r-space.de*. Detailed information on the reconfiguration management on the ESM computer can also be found in [10]. The ESM platform was first presented to open public at a tutorial organized at the conference ARCS'2007 held in Zurich in March 2007.

First experiments with the platform, i. e., the external crossbar architecture, have shown that a substantially higher flexibility in placement and communication between modules can be gained by separating the placement problem from the routing (interconnect) problem. These experiments also suggest that future FPGA devices having separate independent programming layers for interconnect and logic resources would greatly improve flexibility and potentials for applications benefiting from dynamic partial reconfiguration.

At this point, we would like to thank the DFG for funding the priority program and Xilinx Research Labs, San Jose, for providing us with support on design tools and FPGA

samples for building the very first ESM prototypes.

References

[1] *RC2000 Development Board*, Celoxica Ltd. [Online]. Available: http://www.celoxica.com/.

[2] *Xess Corp*. [Online]. Available: http://www.xess.com.

[3] *Nallatech, Inc.* [Online]. Available: http://www.nallatech.com.

[4] *ADM-XRC-II Xilinx Virtex-II PMC*, Alpha Data Ltd., 2002, http://www.alpha-data.com/adm-xrc-ii.html.

[5] H. Walder, S. Nobs, and M. Platzner. "Xf-board: A prototyping platform for reconfigurable hardware operating systems". In: *ERSA*, 2004, p. 306.

[6] C. Steiger, H. Walder, M. Platzner, and L. Thiele. "Online scheduling and placement of real-time tasks to partially reconfigurable devices". In: *Proc. of the 24th Int'l Real-Time Systems Symp. (RTSS'03)*, Dec. 2003.

[7] U. R. H. Kalte and M. Porrmann. "A prototyping platform for dynamically reconfigurable system on chip designs". In: *Proc. of the IEEE Workshop Heterogeneous reconfigurable Systems on Chip (SoC)*, Hamburg, Germany, Sept. 2002.

[8] A. Ahmadinia, C. Bobda, M. Majer, J. Teich, S. Fekete, and J. van der Veen. "DyNoC: A dynamic infrastructure for communication in dynamically reconfigurable devices". In: *Proc. of the Int'l Conf. on Field-Programmable Logic and Applications*, Tampere, Finland, Aug. 2005, pp. 153–158.

[9] A. Ahmadinia, C. Bobda, J. Ding, M. Majer, J. Teich, S. Fekete, and J. van der Veen. "A practical approach for circuit routing on dynamic reconfigurable devices". In: *Proc. of the 16th IEEE Int'l Workshop on Rapid System Prototyping*, Montreal, Canada, June 2005, pp. 84–90.

[10] M. Majer, J. Teich, A. Ahmadinia, and C. Bobda. "The Erlangen Slot Machine: A Dynamically Reconfigurable FPGA-Based Computer". In: *Journal of VLSI Signal Processing Systems*, vol. 46(2), March 2007.

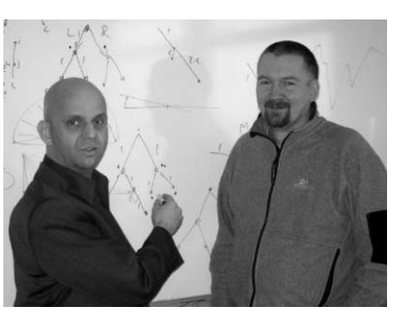
[11] J. Angermaier, S. Fekete, D. Göhringer, M. Majer, J. Teich, and J. van der Veen. "Scheduling and communication-aware mapping of HW-SW modules for dynamically and partially reconfigurable SoC architectures". In: *Proc. Dynamically Reconfigurable Systems Workshop (DRS 2007)*, Zurich, Switzerland, March 2007.

[12] H. A. ElGindy, A. K. Somani, H. Schröder, H. Schmeck, and A. Spray. "RMB – a reconfigurable multiple bus network". In: *Proc. of the Second Int'l Symp. on High-Performance Computer Architecture (HPCA-2)*, San Jose, California, Feb. 1996, pp. 108–117.

[13] R. Vaidyanathan and J. L. Trahan. *Dynamic Reconfiguration: Architectures and Algorithms*. IEEE Computer Society, 2003.

[14] S. P. Fekete, J. C. van der Veen, M. Majer, and J. Teich. "Minimizing communication cost for reconfigurable slot modules". In: *Proc. 2006 Int'l Conf. on Field Logic and Applications (FPL 2006)*, IEEE, 2006, pp. 535–540.

Mateusz Majer, Josef Angermeier, Diana Göhringer, and Jürgen Teich (left to right)



Sándor P. Fekete and Jan C. van der Veen (left to right)

1 Dipl.-Ing. Josef Angermeier is a research assistant at the chair of Hardware/Software Co-Design at the Department of Computer Science of the University of Erlangen-Nuremberg.
Address: Institut für Informatik 12, Am Weichselgarten 3, 91058 Erlangen, Germany, Tel.: +49-9131-8525158, Fax: +49-9131-8525149, E-Mail: angermeier@informatik.uni-erlangen.de

2 Dipl.-Ing. Diana Göhringer is a research assistant at the chair of Hardware/Software Co-Design at the Department of Computer Science of the University of Erlangen-Nuremberg.
Address: s. o. Tel.: +49-9131-8525147, Fax: +49-9131-8525149, E-Mail: goehringer@informatik.uni-erlangen.de

3 Dipl.-Ing. Mateusz Majer is a research assistant at the chair of Hardware/Software Co-Design at the Department of Computer Science of the University of Erlangen-Nuremberg.
Address: s. o. Tel.: +49-9131-8525159, Fax: +49-9131-8525149, E-Mail: majer@informatik.uni-erlangen.de

4 Prof. Dr.-Ing. Jürgen Teich heads the chair of Hardware/Software Co-Design at the Department of Computer Science of the University of Erlangen-Nuremberg.
Address: s. o. Tel.: +49-9131-8525150, Fax: +49-9131-8525149, E-Mail: teich@informatik.uni-erlangen.de

5 Prof. Dr. Sándor P. Fekete is professor for mathematics at the Department of Mathematical Optimization at Braunschweig University of Technology.
Address: Institut für Mathematische Optimierung, Pockelsstr. 14, 38106 Braunschweig, Germany, Tel.: +49-531-391-7551, Fax: +49-531-391-7559, E-Mail: s.fekete@tu-bs.de

6 Dipl.-Math. Jan C. van der Veen is a research assistant at the Department of Mathematical Optimization at Braunschweig University of Technology.
Address: s. o. Tel.: +49-531-391-7555, Fax: +49-531-391-7559, E-Mail: j.van-der-veen@tu-bs.de