The Architecture of a Reconfigurable ATM Switch (RECATS)

Edson L. Horta, Sergio T.Kofuji
Laboratory of Integrated Systems - Department of Electronic Engineering
Escola Politécnica da Universidade de São Paulo, São Paulo, Brazil
Av. Prof. Luciano Gualberto, trav. 3, 158. CEP 05508-900 - São Paulo, SP, Brazil
Tel: (011) 818-5589 fax: (011) 211-4574. horta@lsi.usp.br, kofuji@lsi.usp.br

Abstract :this work describes the architecture of a reconfigurable ATM switch that is capable of changing its hardware features "on the fly", according to special packets traversing the network. In order to accomplish this goal, it will be used a FPGA that supports partial reconfiguration.
Keywords: reconfigurable logic, partial reconfiguration, ATM switching.

1. Introduction

Reconfigurable logic can be used practically in every digital system to provide better performance. The main approaches of this technology are Run-Time Reconfiguration and Compile-Time Reconfiguration.

The ATM switch is a crucial component in the network technology. It can be implemented using standard or reconfigurable logic. When the switch is implemented with reconfigurable logic, it can change its features dynamically, providing to the network a great variety of services that would take so much longer to be available.

In the next sections, the architecture of this reconfigurable ATM switch will be presented and some applications for these elements will be discussed.

2. Reconfigurable Logic

Reconfigurable Logic is implemented with a component called FPGA (Field Programmable Gate Array), that is based on SRAM elements. These elements can be reprogrammed an unlimited number of times. According to [8], the implementation of a system that uses reconfigurability can be done in two ways: Compile-Time and Run-Time Reconfiguration.

In Compile-Time Reconfiguration (CTR) the FPGA does not change its configuration during the application lifetime. Each application has its specific functions that are loaded when you start it. Some examples of CTR systems are SPLASH I and II (a Reconfigurable Linear Logic Array) [2] [3] and PAM (Programmable Active Memories) [11] [12].

The Run-Time Reconfiguration (RTR) allows changing the functions of the system during the application execution. Some examples of RTR systems are Imputer (Imaging Computer) [13] and RRAN 2 (run-time reconfigurable artificial neural network) [4].

The implementation of RTR can be done in two ways: global or local.

The global RTR reconfigures all the FPGA when it changes the configuration. In order to have an efficient partition of configuration, one has to go through an iterative process that divides the system into segments using as much as reconfigurable resources as possible. This is an iterative process because no one knows the physical size of a partition before it has been placed and routed.

The local RTR, also named partial reconfiguration, consists of programming only a part of the FPGA, reducing the reconfiguration time. The partition process of this implementation is based on a functional division of the system.

3. The Reconfigurable Switch

A switch is a set of input and output ports that interconnects users, other switches or other network elements. Its architecture can be seen in the figure 1. According to [1] it can be divided in: SM (System Management), CAC (Connection Admission Control), IM (Input Modules), OM (Output Modules), and CSF (Cell Switch Fabric).

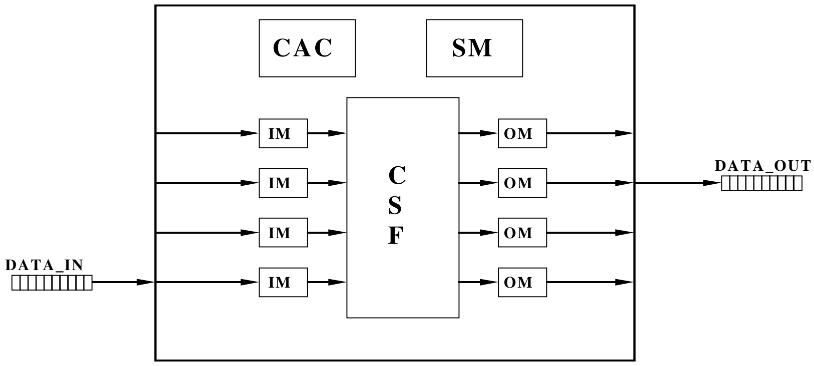

Figure 1 - A Switch

The System Management (SM) module is responsible for performing all the management and traffic control functions to ensure the correct and efficient operation of the switch.

The Connection Admission Control (CAC) module processes and interprets signaling information and decides about the connection admission or rejection. This block is responsible for the interaction with other users and other switches through signaling, and resource allocation decisions for requested connection.

The Input Modules (IM) treat the incoming cells and prepare them for routing trough the CSF (explained later). It is possible, also, to have some functions of CAC and/or SM integrated on the IMs. This integration yields a better performance, but it is more expensive to implement.

The Output Modules (OM) receives cells from the CSF and prepares them for physical transmission on the output links. This block is simpler than the IM and may have some functions of CAC and/or SM integrated on it.

The Cell Switch Fabric (CSF) routes the user data cells from input to output ports and possibly route signaling and management cells between others functional blocks in the switch.

To transform the switch explained above in a reconfigurable one [5] it is necessary to include a Reconfigurable Control Block, a Configuration Memory block and make the CSF programmable (see figure 2).

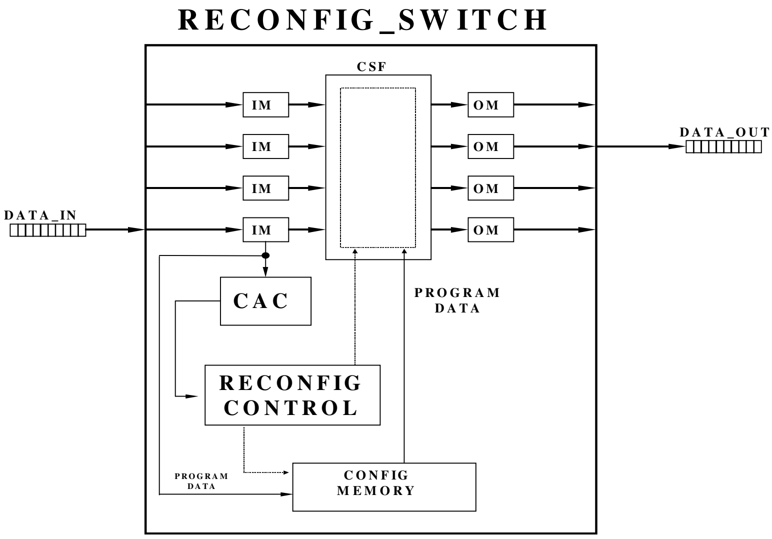

Figure 2 - Reconfigurable Switch

The CAC block will monitor the data entering the switch by each one of its input ports. When some special codes encapsulated in the data appear, the CAC sends them to the RECONFIG CONTROL.

In order to implement the reconfigurable switch (RECATS), the project ATLAS I [10] will be used as a starting point. It is a single-chip ATM switch with credit flow control, described in Verilog. The source code will be modified to introduce the reconfigurability.

The main blocks of ATLAS I [9] switch are:

- 16x622 Mbits/s I/O High Speed Serial Links: STRINGS (BULL macrocell), with IEEE 1355 HIC/HS;
- 4 parallel ports UTOPIA [14] like (8 bits);
- 1 microprocessor port used to control the chip;
- 1 serial ROM port, used to load the switch configuration;
- load monitoring block;
- 256-cell shared buffer, implemented in a 28-stage pipelined memory (SRAM's);
- 54 output queues (3x16 links + 3x1 multicast + 3x1 management) with credit-based flow control, implemented in seven memory blocks: two dual ported SRAM's, one four ported SRAM's (full-custom), two dual ported CAM's (full-custom), one dual ported SRAM (full-custom) and one triple ported CAM (full-custom);
- translation table, implemented with single ported memory;

In the first version of RECATS switch, the following changes will be made to the ATLAS switch:

- take out the 16 I/O Serial Links;
- take out 3 parallel ports;
- take out the microprocessor port;
- take out the serial ROM port;
- take out the load monitoring block;
- take out the credit-based flow control;

• transform the parallel port in a UTOPIA interface, with 4 MPHY links;
• reduce the output queues to 15 (3x4 links + 3x1 multicast);
• introduce a reconfigurable module in the output queues;
• implement a special link that will be multiplexed with the UTOPIA link, in order to provide the path to the reconfigured portion of the switch;

The main blocks of RECATS are:

• Input Module (IM)
• Output Module (OM)
• Shared Data Buffer (SDB)
• Queues Management (QM)
• VPI/VCI Translation Table (VPI/VCI TT)
• Control Unit (CU)
• Partial Reconfigurable (PR)

The IM will receive the cells from a UTOPIA receive interface with four links (MPHY). The cell bodies will go to the shared buffer and the headers will go to the VPI/VCI Translation Table. The IM will also send the OAM cells to the Control Unit.

The Translation Table will perform some operations in the cell header and, according to them, it will assign to the cell a specific priority level in the Queue Management block.

The Control Unit will monitor the OAM cells and it will program the translation table in the signaling phase. It will also be responsible for the partial reconfiguration of the switch and the internal registers programming.

In the output of the Shared Data Buffer it will be placed the Partial Reconfigurable block. This module will be connected to the Output Module via a multiplexer, that will be used to decide if the cell body will pass through the partial reconfigured part or not.

The Output Module will reassemble the cell, with the header from the Queue Management block and it will send them through the UTOPIA transmit interface.

The Queue Management Block will have three priority levels: high, middle and low. In the high level, the cells will go directly to the output link. In the middle and low priority level, the cells will be scheduled to go out of the switch as soon as the high level queue of the desired link is empty.

The RECATS block diagram is shown figure 3.

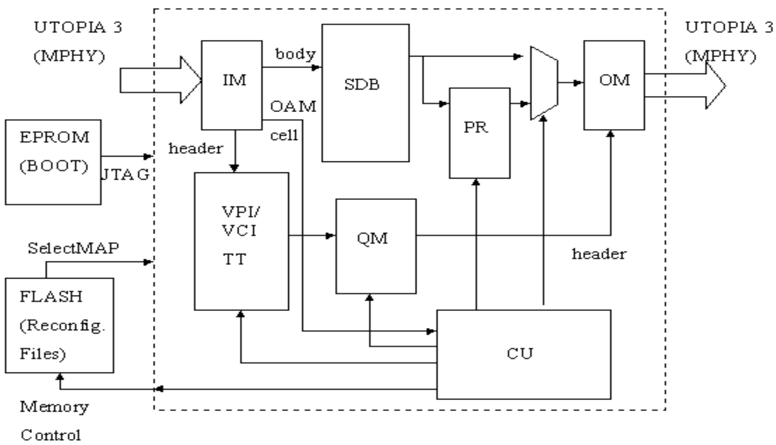

Figure 3 - RECATS

The initial and default configuration file for the FPGA will be loaded from a BOOT EPROM. After this, the partial reconfiguration files will be loaded through special cells (with unused VPI/VCI fields) and they will be stored in nonvolatile memories. When the switch receives a reconfiguration command encapsulated in a special cell, the Control Unit will access these memories and the files will be transferred to the FPGA.

The VIRTEX [15] FPGA will be used to implement the RECATS, due to its partial reconfiguration properties. The floorplanning will be designed to allocate a specific physical space in the component for the partial reconfiguration portion.

The switch will operate in normal mode until it receives a configuration command and an activating command. First, it will reconfigure the switch. Then, after the reconfiguration is done and after it receives the activating command, it will control the multiplexer for the output, in order to the cell bodies be modified by the reconfigured part.

The BOOT EPROM will be connected to the JTAG pins of the FPGA and the nonvolatile memories will be connected to the SelectMAP interface of the FPGA. The Control Unit via an external signal to a controlled oscillator will enable the CCLK of SelectMAP interface. The Control Unit will generate the non-volatile memories control signals also.

4. Conclusion

The architecture of a Reconfigurable ATM Switch (RECATS) was presented. Its main features are:

- one UTOPIA interface, with four links (MPHY);
- three output queues per link, with priority levels;
- partial reconfiguration of the CSF outputs;
- internal VPI/VCI translation table.

Some applications for this switch are:

- the implementation of some MPI Collective Operations [6];
- the implementation of hardware-based cryptography;
- the implementation of Active Networks [7].

5. References

[1] T. M. Chen, and S. S. Liu, ATM Switching Systems, 1995, Norwood, MA: Artech House, Inc, 1995.

[2] M. Gokhale, Bill Holmes. Andrew Kopser, Dick Kunze, Dan Lopresti, Sara Lucas, Ron Minnich, Peter Olsen, SPLASH: A Reconfigurable Linear Logic Array, January 1994

[3] D. T. Hoang. Searching genetic databases on Splash 2. In D. A. Buell and K. L. Pocek, editors, Proceedings of IEEE Workshop on FPGAs for Custom Computing Machines, pages 185-191, Napa, CA, April 1993.

[4] J.D. Hadley, B. L. Hutchings, Designing a partially reconfigured system, in Field Programmable Gate Arrays (FPGAs) for Fast Board Development and Reconfigurable Computing, John Schewel, Editor, Proc. SPIE 2607, pp. 210-220 (1995).

[5] Edson L. Horta, Carlos A. P. S. Martins, Sérgio T. Kofuji, RECATS: A Reconfigurable ATM Switch Proposal, em Workshop em Arquiteturas Reconfiguráveis-IX Simpósio Brasileiro de Arquiteturas de Computadores-Processamento de Alto Desempenho, Campos do Jordão - Brasil, 1997

[6] Edson L. Horta, Sérgio Takeo Kofuji, Using a reconfigurable switch to improve MPI performance, 4th International Conference on Computer Science and Informatics, Research Triangle Park, NC, USA, October 1998

[7] Edson L. Horta, Sérgio Takeo Kofuji, Using reconfigurable logic to implement na active network, 15th International Conference on Computer and Their Applications, New Orleans, LO, USA, March 2000

[8] B. L. Hutchings, M.J. Wirthlin. Implementation Approaches for Reconfigurable Logic Applications, In 5th International Workshop on Field Programmable Logic and Applications, pp 419-428, August 1995, Oxford, England.

[9] G. Kornaros, D. Pnevmatikatos, P. Vatsolaki, G. Kalokerinos, C. Xanthaki, D. Mavroidis, D. Serpanos, M. Katevenis Implementation of ATLAS I: a Single-Chip ATM Switch with Backpressure, in Proc. IEEE Hot Interconnects VI Symposium, Stanford, California, USA, Aug. 1998.

[10] M. Katevenis, D. Serpanos, P. Vatsolaki, ATLAS I: A General-Purpose, Single-Chip ATM Switch with Credit-Based Flow Control, in Proc. Hot Interconnects IV Symposium, Stanford Univ., CA, USA, Aug. 1996, pp. 63-73.

[11] Bertin Patrice, Roncin Didier, and Vuillemin Jean, Programmable Active Memories: a Performance Assessment, Research on Integrated Systems: Proceedings of the 1993 Symposium, G. Borriello and C. Ebeling, editors, pp. 88-102, MIT Press, 1993.

[12] Bertin Patrice and Touati Hervé. PAM Programming Environments: Practice and Experience, IEEE Workshop on FPGAs for Custom Computing Machines, D. A. Buell and K. L. Pocek, editors, Napa, CA, USA, 1994.

[13] D. Ross, O. Vellacott, and Turner. An FPGA-based hardware accelerator for image processing. In W. Moore and W. Luk, editors, More FPGAs: Proceedings of the 1993 International workshop on field-programmable logic and applications, pages 299-306, Oxford, England

[14] UTOPIA Level 2, V.1.0, The ATM Forum Technical Committee, June 1995.

[15] XILINX Inc. : Virtex 2.5 Field Programmable Gate Arrays, Xilinx, Inc., 1999.