

# Self-Reconfiguration on Spartan-III FPGAs with Compressed Partial Bitstreams via a Parallel Configuration Access Port (cPCAP) Core

Salih Bayar

Computer Engineering

Boğaziçi University

P.K. 2 TR-34342 Bebek, Istanbul, TURKEY

Phone: +90 212 359 7780

Fax: +90 212 287 2461

Email: salih.bayar@boun.edu.tr

Arda Yurdakul

Computer Engineering

Boğaziçi University

P.K. 2 TR-34342 Bebek, Istanbul, TURKEY

Phone: +90 212 359 7224

Fax: +90 212 287 2461

Email: yurdakul@boun.edu.tr

*Abstract—This paper presents an alternative approach for dynamic partial self-reconfiguration that enables a Field Programmable Gate Array (FPGA) to reconfigure itself at run-time partially through a parallel configuration access port (cPCAP) under the control of the stand alone cPCAP core within the FPGA instead of using an embedded processor. The cPCAP core with bitstream decompression module needs only 361 slices, which is approximately 18% of a Spartan-3S200 FPGA. The dynamic partial self-reconfiguration via cPCAP core works up to 50Mbyte/s. The compressed partial bitstream is stored in BlockRAM within the FPGA and decompressed via cPCAP core at the time of reconfiguration of the FPGA. This approach has been implemented on a pure Spartan-3 FPGA from Xilinx, but it can also be used for any other FPGA architectures, such as Virtex-II(Pro), Virtex-4, Virtex-5, etc.*

## I. INTRODUCTION

The dynamic partial self-reconfiguration (DPSR) concept is the ability to change the configuration of part of an FPGA device by itself while other processes continue in the rest of the device. A pure Spartan-3 FPGA, which doesn't have any multiboot capabilities and Internal Configuration Access Port (ICAP), cannot be reconfigured without any additional external hardware. However, some other FPGA series such as Spartan-3A(N), Virtex-II(Pro), Virtex-4, Virtex-5 FPGA series have this ICAP module on their predesigned hardware architecture[7][8][9][10][11]. In spite of the lack of an ICAP module on its architecture, dynamic reconfiguration is still supported in Spartan-3 via the external SelectMAP interface or JTAG. As a result, a component should be developed for pure Spartan-3 FPGAs, which acts as an ICAP and allows partial self-reconfiguration at run-time for Spartan-3 FPGA family.

In most cases a reconfigurable FPGA system consists of three main components: an external intelligent agent, some external (non-)volatile memory and a Complex Programmable Logic Device (CPLD). Such a reconfigurable FPGA system is described in detail in [5]. In some cases, systems may not require a CPLD if the used intelligent agent has a sufficient number of general purpose I/O (GPIO) pins. For these systems,

the FPGA can be (re)configured directly by the intelligent agent [5].

In this study, a custom soft-core is developed for self reconfiguration of Spartan-3 FPGAs. It is the extended version of our PCAP core [1] which controls the partial reconfiguration flow through SelectMAP port and supplies configuration clock for reconfiguration. cPCAP uses BlockRAMs to store partial configuration bitstreams. BlockRAMs provide on-chip fast memory in FPGAs. However, the number of BlockRAMs is limited. The new soft-core, cPCAP, maximizes the utilization of BlockRAMs by storing compressed partial bitstreams at initial configuration time and decompressing them during self reconfiguration. Due to compressed partial bitstream, more on-chip storage can be saved. Moreover, the reconfiguration clock speed of cPCAP is the same with that of PCAP in spite of the integrated decompression module. Because of being written entirely in VHDL, this cPCAP core is highly portable and can also be used for all other Xilinx FPGA architectures. Thus it is not necessary to use an external intelligent agent to control the partial reconfiguration flow.

This paper is organized as follows: In section 2, we briefly explain the main types of partial reconfiguration and give a few developed samples associated with DPSR concept. Section 3 presents the structure and functionality of our cPCAP core. Section 4 gives an example, where a run-time DCM reconfiguration via cPCAP is implemented. Finally, section 5 presents our conclusions.

## II. DYNAMIC PARTIAL SELF-RECONFIGURATION (DPSR) CONCEPT IN XILINX FPGAS

Partial reconfiguration is only possible through either serial JTAG interface or parallel slave SelectMAP mode. Since parallel slave SelectMAP interface has higher performance than the serial JTAG interface, the SelectMAP port is used in this study. Parallel SelectMAP port is used for either complete configuration or partial reconfiguration for applications, where the performance is the most important consideration.

To be able to perform dynamic partial self-reconfiguration on a FPGA-based system, there should be either an internal configuration access port or an equivalent port. Some FPGAs such as Spartan-3A(N), Virtex-II(Pro), Virtex-4, Virtex-5 from Xilinx, which have ICAP on their hardware, support self-reconfiguration without using an external intelligent agent. The fact that the pure Spartan-3 does not have such an internal configuration port, has rendered impossible the self-reconfiguration without using an external intelligent agent up to now apart from a few new studies, which are discussed below.

In [3], a soft ICAP, known as JCAP, has been developed in order to realize the self reconfiguration. As a reconfiguration interface they use serial JTAG interface which is very slow compared to parallel SelectMAP port. Though the ICAP on Virtex-II or Spartan3A devices have a reconfiguration speed 66MByte/s [7], JCAP only achieves a reconfiguration rate of 2Mbits/s. The reason of this huge performance difference between the ICAP and JCAP is the serial JTAG interface for JCAP.

In our study we have used parallel SelectMAP port instead of serial JTAG interface, thus we have developed a self-reconfigurable system on pure Spartan-3 series which should be at least 8 times faster than the developed system in [3]. Since a serial configuration method is used in [3], they achieved to send one bit per configuration clock cycle. However, in our study we use a parallel configuration method, hence we send 8 bits at each configuration clock cycle.

In [2], a self-reconfiguration system on pure Spartan-3 has been developed. They have solved the lack of ICAP on Spartan-3 FPGAs by adding an external loopback, therefore they have used a GPIO core on MicroBlaze and 11 external wires to accomplish the interface through SelectMAP port. In order to store initial configuration bitstream and generate configuration clock signal they have also used a XCF configuration flash PROM. Under the control of GPIO core of MicroBlaze they reconfigure the target FPGA through SelectMAP port. Though they have achieved a speed as in ICAP, they have used an external PROM to store initial configuration bitstream, MicroBlaze soft core to control the configuration flow and a TFTP server and onboard SDRAM to store the partial bitstreams. Although we have also used 11 external wires and the SelectMAP port as a reconfiguration interface, we have used BlockRAM to store partial bitstream, cPCAP core to control the reconfiguration flow. Since they designed a MicroBlaze-based system, they used 4198 slices of an Spartan-3S2000 FPGA. Applying such a system to small FPGAs (e.g. to a Spartan-3S200) is impossible. However, our cPCAP core is very small, which is 361 slices and only 18% of a Spartan-3S200. Thus, we have accomplished a processor-independent run-time reconfigurable system and presented a very new approach for storing compressed partial bitstreams on BlockRAM within the FPGA.

Both [3] and [2] process uncompressed partial bitstreams that are stored in external memory. However, cPCAP can process compressed partial bitstreams that are stored in Block-

RAMs. This has two main advantages: 1) A BlockRAM can hold more compressed partial bitstreams than regular uncompressed partial bitstreams 2) Access time to a partial bitstream in a BlockRAM is much shorter than the access time to a partial bitstream in an external memory. Decompression is realized in cPCAP during reconfiguration time without sacrificing from reconfiguration speed.

### III. CPCAP ARCHITECTURE

The Spartan-3 FPGA configures itself through its SelectMAP port under the control of cPCAP. Through the parallel SelectMAP interface, the partial reconfiguration information is accepted and the reconfiguration process is executed again by the same target FPGA, where this unique FPGA acts as not only a *slave* but also a *master* at the time of reconfiguration as shown in Figure 1.

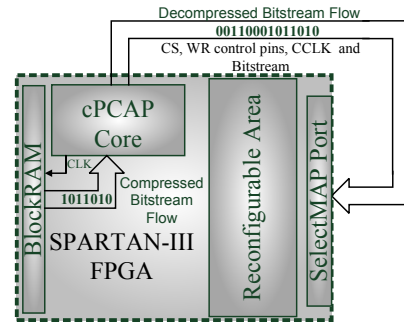


Fig. 1. Hardware architecture of whole system

As shown in Figure 2, in order to generate configuration clock frequency a Digital Clock Manager (DCM) component is used. Since we generate CCLK signal within FPGA, it acts as master, but at the same time we accept the CCLK signal through the SelectMAP interface into FPGA as if the signal comes from other intelligent agent, where the FPGA acts as slave. The source of CCLK is not important for the FPGA. The cPCAP core reads a byte from the BlockRAM at each clock cycle. Under the control of CS, WRITE, CCLK signals, this byte is sent to SelectMAP interface. The bitstream information, which is accepted from BlockRAM, is in compressed manner. Since we achieve the decompression of bitstream information at the time of reconfiguration, we do not need any additional time for decompression. Therefore when the CCLK speed is set to 50 MHz, the reconfiguration speed is 50MByte/s. Note that the reconfiguration speed is independent from the size of partial bitstream. The BUSY signal is only used if the configuration clock frequency exceeds 50 MHz. In our study, the cPCAP core can be configured to operate up to 50 MHz. We have not exceeded 50 MHz yet, that's why we have not used BUSY signal. However, it will be examined in the future work. The configuration flow of an FPGA for run-time reconfiguration via cPCAP is shown in Figure 3. The uncompressed partial bitstreams are generated with the help of bitgen -r flow[6].

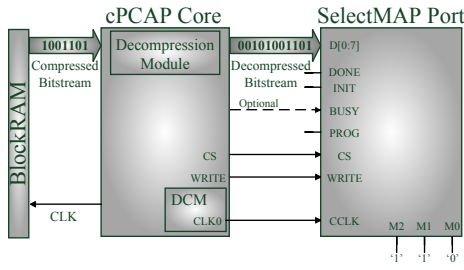


Fig. 2. cPCAP Core and SelectMAP interface

The storage of partial reconfiguration bitstream requires also an additional external hardware for reconfigurable systems. In the most of reconfigurable systems the partial reconfiguration file is stored in an external non-volatile device. It can be read from there under the control of either an external intelligent agent or the FPGA itself, where FPGA acts as a *slave* and *master* respectively. Contrary to the standard methods based on storing partial reconfiguration bitstream on external non-volatile devices, the partial reconfiguration bitstream in this study is stored in BlockRAM within the target FPGA. As a result of this new approach, there is no need to use an additional external device for storing partial reconfiguration bitstream for any system, which works continuously after the initial configuration.

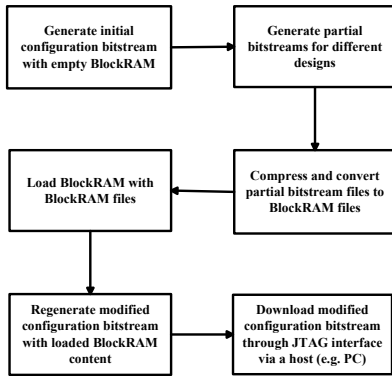


Fig. 3. Configuration flow

#### A. File Compression and Conversion

There are a lot of different methods to store information in a BlockRAM. Using generic template from language templates in ISE is one of them. According to the bit-width and -depth of BlockRAM, there are various prepared pieces of code for BlockRAM available, which can be easily inserted into the BlockRAM HDL source file. Since we need to store partial bitstream bytes in BlockRAM, we have used one of these BlockRAM template.

After generating partial bitstream files, we have compressed these files and converted them to a suitable form with ".vhd" extensions using a file compression and converter module, which is written in Java language, in Figure 4. Note that



Fig. 4. File conversion from partial bitstream file to BlockRAM coefficient file

the number of partial bitstreams needs not to be equal to the number of BlockRAMs.

#### B. Dynamic Partial Self-Reconfiguration Flow

Since we have accomplished a dynamic partial self-reconfiguration through the SelectMAP port, the developed cPCAP core behaves in our study as if it is a mirror of SelectMAP port, as same in ICAP. The following flow in Figure 5 is very similar to "SelectMAP configuration Flow Diagram" in [5] except that PROG, INIT, DONE, BUSY pins are not taken into account in our study. The first three control signals PROG, INIT, DONE are only used during complete (re)configuration. BUSY signal is used if the configuration clock (CCLK) frequency is greater than 50Mhz. Due to the fact that there is only a 50Mhz oscillator available on Spartan-3 Starter Board we have chosen the CCLK for our system exactly 50Mhz, thus we do not need to use BUSY indicator signal. Under some circumstances, such as using BUSY indicator signal where it is needed, the developed design can be clocked with any other frequency values, which are supported by Spartan-3 FPGA and its SelectMAP interface. We have experimented, that our cPCAP core can run safely at all frequencies up to 50Mhz. However, the performance of the cPCAP core at higher frequencies will be tested after implementing the rest of the handshake signals in future.

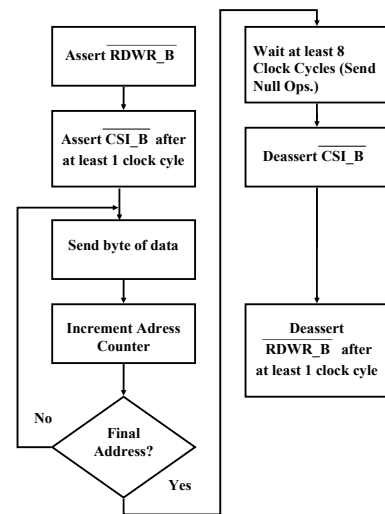


Fig. 5. cPCAP core Configuration Control Flow Diagram

#### IV. EXAMPLE: RUN-TIME DCM RECONFIGURATION

The soft cPCAP core, which is a pure VHDL code, has been synthesized on Spartan-3S200 Starter Kit Board. In this work we have reconfigured a clock output of Digital Clock Manager (DCM), which drives the complete system, at run-time. Such a DCM reconfiguration approach is described in detail in [4].

There are various applications, where clock frequency of a system should be change at run-time. Clock scaling method in [4] is one of them, which is mostly used to decrease FPGA power consumption by changing clock frequency of different components of system at run-time. Changing data transmission speed of a system, and other typical applications include speed drives, inverters, computers and computer controlled equipment, deep well pumps, industrial machinery, ships, aircraft.

In this work, a 4-bit up-down counter is taken as an example. This counter either works with 5 MHz or 50 MHz, which is accomplished by run-time DCM reconfiguration via cPCAP core. The size of each decompressed partial bitstream for DCM reconfiguration is 5 KByte. The reconfiguration speed is 50 MByte/s, which means that the DCM reconfiguration via cPCAP core takes approximately 0.1 ms. This counter is used solely to show that such a reconfiguration approach is possible for other reconfigurable systems where the frequency of a system or a component of system can be changed at run-time without affecting anything else in complete system.

To be able to do reconfiguration we have firstly generated complete bitstream files and then with the help of bitgen tool two fully routed NCD (Native Circuit Description) file for each different frequency values. Contrary to approach in [4] for generating partial bitstreams, the difference based approach is used in this work.

The implementation cost of our study is summarized in Table I. After adding the decompression module to the cPCAP core, we have needed to use only 1 BlockRAM to store two different compressed partial bitstreams within the BlockRAM. With this compression approach we have accomplished at least 76% space saving approximately, where the compression ratio is actually based on the structure and size of the partial bitstream.

TABLE I  
OCCUPIED RESOURCES FOR PCAP AND CPCAP CORES

Occupied Resources	<i>PCAP</i>	<i>cPCAP</i>
	(without compression)	(with compression)
BlockRAM	6 of 12, %50	1 of 12, %8
Slices	365 of 1920, %19	324 of 1920, %16
DCM	1 of 4, %25	1 of 4, %25

#### V. CONCLUSION

In this paper we have discussed dynamic partial self-reconfiguration of a pure Spartan-3 FPGA through the SelectMAP port and storing different partial bitstreams on BlockRAM within the target FPGA. The most important advantage

of this study is to achieve a very fast partial reconfiguration compared to other serial JTAG interfaces and using a new approach such as storing compressed partial bitstreams on on-chip memory and reading them from there under the control of an cPCAP core instead of an external intelligent agent, which reduces hardware cost and power consumption simultaneously. Furthermore, with the capability of storing compressed partial bitstreams, many different partial bitstreams can be stored on-chip memory at a glance.

This kind of implementation, using no other additional external devices apart from FPGA, is a perfect solution for almost all systems based on cost and power consumption. What's also very impressive about this implementation is that the developed cPCAP core can be applied not only on a pure Spartan-3 and also on other FPGA series such as Virtex-II, Virtex-4, Virtex-5, Spartan-3A(N) and so on.

In addition to this big advantage related to size, this cPCAP core can be used anywhere in the FPGA, whereas the location of the ICAP module on Virtex-II devices and two ICAP modules on Virtex-4 are fixed [8][10].

In our future work, we plan to implement other handshaking signals on cPCAP for providing self-reconfiguration at frequencies higher than 50MBytes/sec. Decoupling of the memory architecture from SelectMAP interface in cPCAP will also be done in order to support different memory types and reconfiguration interfaces.

#### ACKNOWLEDGMENT

This work is fully supported by The Scientific and Technological Research Council of Turkey, TÜBİTAK (Project Nr.: 104E038) and Boğaziçi University Scientific Research Projects (Project Nr.: 06M105).

#### REFERENCES

- [1] Salih Bayar and Arda Yurdakul. *Dynamic Partial Self-Reconfiguration on Spartan-III FPGAs via a Parallel Configuration Access Port (PCAP)*. HIPEAC2008, Gothenburg, Sweden, January 2008.
- [2] Ivan Gonzalez, Estanislao Aguayo, and Sergio Lopez-Buedo. Self-reconfigurable embedded systems on low-cost fpgas. *IEEE Micro*, pages 49 – 57, July-Aug. 2007.
- [3] K. Paulsson, M. Hübner, G. Auer, M. Dreschmann, L. Chen, and J. Becker. *Implementation of a Virtual Internal Configuration Access Port (JCAP) for enabling Partial Self-Reconfiguration on Xilinx Spartan-III FPGAs*. FPL, Amsterdam, Netherland, August 2007.
- [4] Katarina Paulsson, Michael Hübner, Salih Bayar, and Jürgen Becker. *Exploitation of Run-Time Partial Reconfiguration for Dynamic Power Management in Xilinx Spartan III-based Systems*. ReCoSoc2007, Montpellier, France, June 2007.
- [5] Xilinx. *Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode*. XAPP502, (v1.4) edition, November, 13 2002.
- [6] Xilinx. *Two Flows for Partial Reconfiguration: Module Based or Difference Based*. XAPP290, (v1.2) edition, September, 9 2004.
- [7] Xilinx. *Spartan-3 Generation Configuration User Guide*. UG332, (v1.2) edition, May, 23 2007.
- [8] Xilinx. *Virtex-4 Configuration Guide*. UG071, (v1.9) edition, October, 1 2007.
- [9] Xilinx. *Virtex-5 FPGA Configuration User Guide*. UG191, (v2.5) edition, October, 10 2007.
- [10] Xilinx. *Virtex-II Platform FPGA User Guide*. UG002, (v2.1) edition, 28 March 2007.
- [11] Xilinx. *Virtex-II Pro and Virtex-II Pro X FPGA User Guide*. UG012, (v4.1) edition, 28 March 2007.