

Hardware-Accelerated SSH on Self-Reconfigurable Systems

Ivan Gonzalez, Francisco J. Gomez-Arribas and Sergio Lopez-Buedo
Escuela Politecnica Superior, Universidad Autonoma de Madrid, Spain
{Ivan.Gonzalez, Francisco.Gomez, Sergio.Lopez-Buedo}@uam.es

Abstract

The performance of security applications can be greatly improved by accelerating the cryptographic algorithms in hardware. In this paper, an implementation of the Secure Shell (SSH) application is presented. A self-reconfigurable platform based on a Xilinx Spartan-3 FPGA has been employed to implement a MicroBlaze-based embedded system, which executes SSH under the uCLinux operative system. Run-time reconfiguration is used to change the cryptographic coprocessors utilized for data ciphering. The performance of SSH has been improved and, in comparison to other alternatives not using reconfiguration, a reduction of the area requirements was also achieved.

1. Introduction

Hardware acceleration of software applications is a commonly used methodology for high performance computing. By executing some computation-intensive tasks onto hardware, the throughput of a system can be significantly enhanced. Implementing this acceleration in reconfigurable HW improves its flexibility: Just like a processor switches between different executables, several bitstreams can be downloaded to the FPGA.

Configurable System-on-a-Chip (CSoC) devices are well-suited to implement cryptographic systems [1], because they combine near-hardware speed with the flexibility of software on general-purpose computers. They also provide better security than SW-based solutions. Besides, partial reconfiguration can be very useful for these applications. For example, if a system has to give support for many ciphering standards, and run-time reconfiguration is used, only the coprocessor for the chosen algorithm would have to be loaded in the FPGA. When it is no longer needed, it could be removed from the device to leave room for a new one.

In this paper, the possibilities of CSoCs to improve security applications are evaluated. The SSH protocol has been used as case-study, and it has

been accelerated using specific cryptographic cores as coprocessors.

2. MicroBlaze-based system

The developed system includes two modules (figure 1): The first one is composed by MicroBlaze [7] and its peripherals, and it is located in the left half of the FPGA. The second one is the coprocessor, placed to the right of the device. Different designs were constructed maintaining the same MicroBlaze module, but changing the cryptographic accelerators. Each coprocessor has been implemented with the same interface: one master FSL bus [8] for receiving the key, and two pairs of master/slave FSL busses to send/receive 64-bit data.

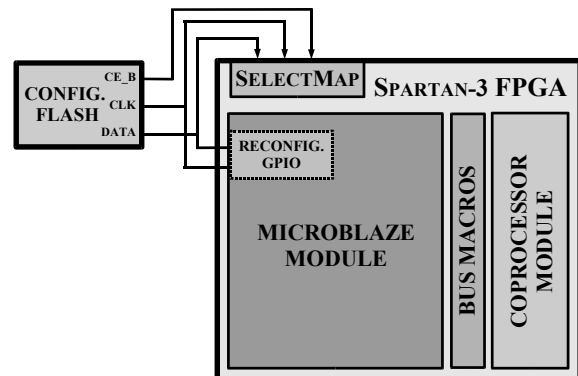


Figure 1. Self-reconfigurable platform based on Spartan-3

Traditionally, design automation tools have lacked support for reconfiguration in standard design flows. However, Xilinx offers an alternative to support partial reconfiguration via Modular Design together with the RECONFIG mode for the AREA_GROUP constraint [9]. Also, Xilinx provides the bus macro [9] to make external connections between modules. Unfortunately, this component needs tri-state buffers, so it cannot be used in Spartan-3 designs. But the concept behind bus macros can be extended to other FPGA components, following the idea suggested in [2]. In this way, custom ‘bus macros’ were generated for this project using buffers constructed with LUTs.

Modular design was used for all the steps but the final module assembly, which cannot be completed due to bugs in Xilinx's tools. Instead, a Perl script working over XDL files was developed as workaround for this problem [3]. Finally, self-reconfiguration was accomplished by means of a simple external circuit: You only need to add a general purpose I/O (GPIO) to the MicroBlaze embedded processor, and externally connect it to the pins of the parallel configuration interface. A simplified diagram of the circuit used to implement self-reconfiguration is shown in figure 1.

Table 1. Logic usage

	FPGA Slices	LUTs	FFs / Latches	18x18 Mult	BRAM
MicroBlaze	4198	4809	3618	3	20
AES Core	4326	4632	1802	0	0
3DES Core	5424	4493	5825	0	0

The MicroBlaze system has been implemented on Spartan-3 Development Board from Avnet (XC3S2000FG676-4C) running at 65 MHz. Version 6.3 of EDK was utilized. Table 1 presents the resources used for the different designs: The MicroBlaze system and the different cryptographic coprocessors.

3. SSH implementation

The operative system running on MicroBlaze is uCLinux [4], a port of regular Linux to microprocessors lacking a memory management unit. The OpenSSH [5] open source implementation of SSH [10] has been used as a basis for this project. It is available on the uCLinux distribution. To adapt OpenSSH to using cryptographic coprocessors, it was necessary to modify the OpenSSL library in order to change the cipher functions from software to hardware.

Table 2. File transfer speed [Kbytes/sec]

Protocol	Crypto. Algorithm	File 1 335 KB	File 2 2410 KB	File 3 8266 KB
SSH Software	AES	27.8	48.1	52.7
	3DES	16.7	26.6	30.3
SSH Hardware	AES	33.2	60.4	66.4
	3DES	32.0	60.3	67.7

The different speeds obtained while copying a set of files through a secure channel has been measured, in order to assess the enhancements due to the use of hardware acceleration. Two symmetric-key algorithms used in the SSH2 protocol were selected to be accelerated in hardware: the default AES128-CBC and the optional 3DES-CBC [6].

Table 2 shows the average results for the different SSH implementations (SW or HW) and algorithms employed. The small improvements are mainly due to the overhead from the hash algorithms used for data integrity, which are based on MD-5. However, the results illustrate the promising capabilities of the reconfigurable approach presented in this work.

4. Conclusions

Configurable System-on-Chip (CSoC) platforms that include processors and reconfigurable logic are good solutions to speed up embedded software, because critical parts of the code can be implemented as application-specific coprocessors. Good examples of this methodology are cryptography applications, because they are known to benefit significantly from hardware acceleration.

Although the speedup results are not excellent, this work allowed us to validate the feasibility of using self-reconfiguration to accelerate secure communication protocols. Further enhancements, like implementing the data integrity algorithms also in HW, will permit achieving a significant speedup. Moreover, the area savings due to this approach are remarkable, because only the coprocessor for the currently running algorithm has to be loaded in the FPGA.

5. References

- [1] Ray C.C. Cheung, Wayne Luk and Peter Y.K. Cheung, "Reconfigurable Elliptic Curve Cryptosystems on a Chip", *Proc. of the Design, Automation and Test in Europe Conference and Exhibition (DATE '05)*, pp.24-29, 2005.
- [2] M. Dyer, C. Plessl, and M. Platzner, "Partially Reconfigurable Cores for Xilinx Virtex", *Lecture Notes in Computer Science 2438*, pp. 292-301, Springer-Verlag, Berlin, 2002.
- [3] I. Gonzalez, E. Aguayo and S. Lopez-Buedo, "Sistemas Embebidos Auto-Reconfigurables sobre Spartan-3", *V Jornadas sobre Computacion Reconfigurable y Aplicaciones* (in press), Granada, Spain, 2005 (in Spanish).
- [4] Microblaze uCLinux Project. Home page: <http://www.itee.uq.edu.au/~jwilliams/mblaze-uclinux/>
- [5] OpenSSH Project. Home page: <http://www.openssh.com/>
- [6] B. Schneier, *Applied Cryptography 2nd Ed.*, John Wiley & Sons, 1996.
- [7] Xilinx Inc., "MicroBlaze Processor Reference Guide", 2004.
- [8] Xilinx Inc., "Connecting Customized IP to the MicroBlaze Soft Processor Using the Fast Simplex Link (FSL)", *Application Note 529*, 2004.
- [9] Xilinx Inc., "Two Flows for Partial Reconfiguration: Module Based or Difference Based", *Application Note 290*, 2004.
- [10] T. Ylonen, "SSH -- Secure Login Connections over the Internet", *Proc. 6th USENIX Security Symposium*, pp. 37-42, 1996.