Enhancing COPACOBANA for Advanced Applications in Cryptography and Cryptanalysis

Tim Güneysu*, Christof Paar*, Gerd Pfeiffer◇, Manfred Schimmler◇
* Horst Görtz Institute for IT Security, Ruhr University Bochum, Germany
◇ Institute of Computer Science, Christian-Albrechts-University of Kiel, Germany
{gueneysu,cpaar}@crypto.rub.de, {gp,masch}@informatik.uni-kiel.de

Abstract

Cryptanalysis of symmetric and asymmetric ciphers is a challenging task due to the enormous amount of involved computations. To tackle this computational complexity, usually the employment of special-purpose hardware is considered as best approach. We have built a massively parallel cluster system (COPACOBANA) based on low-cost FPGAs as a cost-efficient platform primarily targeting cryptanalytical operations with these high computational efforts but low communication and memory requirements. However, some parallel applications in the field of cryptography are too complex for low-cost FPGAs and also require the availability of at least moderate communication and memory facilities. Particularly, this holds true for arithmetic intensive application as well as ones with a highly complex data flow.
In this contribution, we describe a novel architecture for a more versatile and reliable COPACOBANA capable to host advanced cryptographic applications like high-performance digital signature generation according to the Elliptic Curve Digital Signature Algorithm (ECDSA) and integer factorization based on the Elliptic Curve Method (ECM). In addition to that, the new cluster design allows even to run more supercomputing applications beyond the field of cryptography.

1. Introduction

Cryptanalysis of modern cryptographic algorithms needs a significant amount of computational effort, often far beyond 2^{40} operations. Usually, this number of computations is considered to be addressed best with large computing clusters and/or special-purpose hardware. For cryptanalytical algorithms running in a highly parallel fashion and with very little interprocess communication, we have built an FPGA-based cluster with a strong focus on cost-efficiency, namely the COPACOBANA (Cost Optimized Parallel Code Breaker) [5].

The first version of COPACOBANA was equipped with 120 independent low-cost FPGAs (Xilinx XC3S1000), distributed over 20 modules which are plugged into a single backplane and connected via a parallel and shared data bus. The lack of additional memory or high-speed communication facilities supported the simple design approach and provided bare computational resources at low costs. However, the usability of COPACOBANA was yet limited to applications which do not have a high demand to one of these aspects like memory and high-speed communications. Moreover, although providing a high density of logic resources, low-cost FPGAs like the XC3S1000 devices only offer rather generic support for high-performance arithmetic on large integers. More precisely, wide multipliers with more than 160 bits – as typically used in public-key cryptosystems (and cryptanalysis) – consume large portions of the available logic when implemented with conventional structures, e.g., Wallace Trees[1]. Beside a high density of generic logical elements, more modern FPGAs offer integrated hardcores like *PowerPC* microprocessors or arithmetic function blocks (*DSP-blocks*) to accelerate complex Digital Signal Processing (DSP) operations. Recently, it has been shown how these DSP blocks can accelerate RSA encryptions [11] as well as attacks on RSA [2]. Based on the presented results, the use of DSP-block-based arithmetic in cryptographic functions let expect an increase in performance even by a few orders of magnitude.

In this contribution, we present a new cluster architecture capable to host more powerful Virtex-4 FP-

[1] In fact, there are a few 18×18 bit multiplier hardcores on XC3S1000 devices but not sufficiently many to support complex cryptographic operations.

978-1-4244-1961-6/08/$25.00 ©2008 IEEE.

GAs supporting microprocessor-based designs as well as accelerating arithmetic intense applications using the DSP blocks. Further fundamental modifications on the cluster include a Gigabit communication link between host and the FPGA cluster and also introduce a Hierarchical Communication Model (HCM) to effectively reduce the communication load between components by intermediate data aggregation. Based on this platform, we present parallel implementations for the generation of digital signatures over elliptic curves (ECDSA) as well as for factoring mid-size integers using the Elliptic Curve Method (ECM) [6]. With these applications we demonstrate that a massively parallel FPGA cluster can be used both to accelerate constructive cryptographic applications like high-performance message signing as well as destructive attacks, e.g., on the factorization problem of the well-known RSA encryption scheme. All in all, the increased versatility with more powerful computing nodes and the novel hierarchical communication system has advanced the COPACOBANA to a memoryless supercomputer even for use beyond the field of cryptography.

This work is organized as follows: we start with a short review of previous work on cryptographic supercomputing. Next, we discuss our novel FPGA-based cluster architecture and, particularly, the applied modifications and changes with respect to the previous COPACOBANA system. Thereafter, we show how such a cluster can tackle cryptographic challenges like the generation of digital signatures and factorization of mid-sized numbers and will also give estimates on the respective performance of both applications.

2. The Virtex-4 Cluster Architecture

The original COPACOBANA cluster combined 120 Spartan-3 XC3S1000 FPGAs distributed along 20 plug-in modules in a single backplane. Since each plug-in card hosts only 6 FPGAs, this approach is not considered optimal, e.g., when using a binary address encoding. Instead, taking power distribution, and routing constraints, signal integrity and mechanical packing into account, we switched to a partitioning of 16 plug-in modules each hosting 8 FPGAs for the new cluster design, also supporting direct binary addressing. Similar to our original approach, all FPGAs on the plug-in cards are connected to a shared 64 bit single master bus and an additional 16 bit address bus on a backplane.

For the new design, we chose Virtex-4 FPGA devices for advanced functionality due to their integrated hardcores instead of the previously employed Spartan-3 devices. Here, we preferred the medium-sized Virtex-4 devices over very large ones due to their relatively lower costs and better availability. For these devices, we selected the smallest but most versatile footprint, precisely, the FF668/FF672 package with a size of $27 \times 27\ mm$.

On the same plug-in module, all eight FPGAs are connected to a single CPLD (CoolRunner-II) which simultaneously acts as bus driver and communication bridge between the shared bus on the backplane and the local bus on the module. The shared 64 bit bus on the backplane is driven and controlled by a further Virtex-4 FX FPGA at 20MHz placed on a separate controller module. This FPGA also integrates a full-blown TCP/IP stack running on the integrated PowerPC so that the FPGA cluster can establish a connection to a host computer via Gigabit Ethernet. Based on these intermediate communication elements, we designed the communication system as a three-tier architecture allowing a target application to implement a Hierarchical Communication Model (HCM). In this HCM, the bulk of data will be generated by the eight (locally interconnected) FPGAs on the same plug-in module which can be controlled and aggregated by the bus-mastering CPLD. The aggregated data stream is then transmitted via the shared bus to the central controller FPGA which can apply further, more sophisticated data compression functions due to the availability of its embedded PowerPC. Finally, the condensed data is sent to the host computer via the Gigabit connection what we assume sufficient with regard to that prior data reduction and aggregation has already taken place. The HCM and its corresponding tiers is shown in Fig. 1.

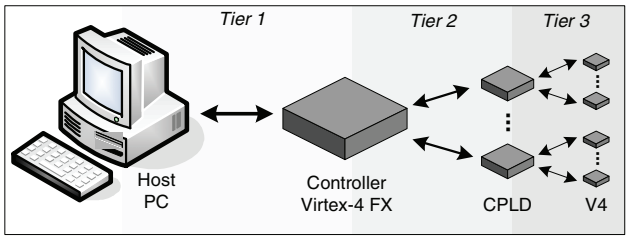

Figure 1. Hierarchical communication system on COPACOBANA v2 with 3 tiers

However, the improved performance of the Virtex-4 series FPGAs comes in line with an increased energy consumption per chip. Here, we estimated the required power per chip based on assumption that cryptanalytical applications are likely to utilize all available hardware resources. According to these requirements, the power distribution system was designed to supply each Virtex-4 FPGA with a maximum of 10W. Consequently, we chose a single DC power supply unit pro-

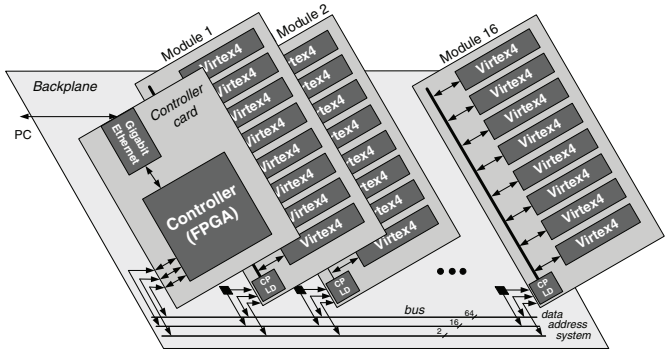

Figure 2. Architecture of COPACOBANA v2

viding 125A output at 12V. The corresponding 1500W of output power are distributed by the backplane to all plug-in cards and are locally transformed into the 1.2V core and 2.5V I/O voltage by individual DC/DC converters. The dissipation of 1500W electrical power requires a sophisticated thermal management in terms of the selection of fans, routing of air flow, and choice of effective heat sinks.

The architecture of the enhanced FPGA-cluster (COPACOBANA v2) is depicted in Fig. 2.

3. Applications

In this section we will outline two applications which significantly benefit from the advanced features of the new architecture. Note that both presented applications are arithmetic intense so that we populated the machine with Virtex-4 SX 35 devices providing 192 integrated DSP-blocks per FPGA for accelerating the fundamental integer computations.

3.1 High-Performance ECDSA Signature Generation and Verification

Our first application for the new cluster is the generation and verification of digital signatures according to the Elliptic Curve Digital Signature Algorithm (ECDSA). Digital signatures are employed in many cryptographic applications, like in the field of eCommerce, eHealth or automotive. Since all practical asymmetric signature schemes rely on hard problems, they are usually computationally challenging for the underlying processing platform. Particularly for back-end systems in companies and governments, this means that server systems might be faced with a large number of signatures to be concurrently verified where each verification usually takes a few milliseconds even with support of special hardware. For example, the FPGA-based designs for RSA and ECC-based systems presented in [1, 9] can be considered as high-performance implementations but even so both take more than 3 ms per operation.

For this reason, we will present a considerably more powerful implementation taking advantage of the integrated DSP-blocks in the Virtex-4 devices, which employs 256 bit security parameters standardized by NIST [8] providing a sufficient security margin even for the governmental applications of the next decades. Together with the computational power of the cluster at hand, we can present a FPGA-based system capable to tackle requirements like that of the German eHealth project. In this context, we adapted our implementation of the ECDSA architecture presented in [4] for the Virtex-4 SX 35 FPGAs which are available on our enhanced cluster system.

Since the cluster hardware is not completely built yet, we will provide performance estimates based on the known results of a single core implementation (all estimated figures are denoted by asterisks). We assume a multi-core architecture with 6 cores not to exceed more than 245 MHz in frequency what is 50% of single core implementation due to longer routing paths and effects from unrelated logic packing. Our estimates for performing ECDSA operations over the prime field P-256 on COPACOBANA are shown in Table 1. With the presented implementation, we are able to compute 256 bit ECDSA signatures for up to 158 MBit of data per second. To the best of our knowledge, this parallel implementation of an asymmetric signature scheme on a cluster system seems to provide the highest throughput reported in the open literature.

Aspect	**ECDSA** P-256
Number of Cores per FPGA	6
4-input LUTs per FPGA	10,326
Flip flops per FPGA	14,592
DSP blocks per FPGA	192
BRAMs per FPGA	66
Operations kP	620,000 op/s*
Operations $kP + lQ$	512,000 op/s*
Throughput kP	158 Mbit*
Throughput $kP + lQ$	131 Mbit*

Table 1. Results for ECDSA over the NIST prime field P-256 on a COPACOBANA populated with 128 Virtex-4 XC4VSX35 devices

3.2 Efficient Integer Factorization with ECM

The factorization of a large composite integer n where $n = \prod p_i$ with several prime factors p_i is a well-known mathematical problem which has attracted special attention since the invention of asymmetric cryptography. RSA [10] is a prominent example for an asymmetric cryptosystem what relies on the assumption of an attacker's inability to factor large numbers. Up to now, the best known method for factoring large integers is the General Number-Field Sieve (GNFS). An important step in this algorithm is the factorization of mid-sized numbers for the smoothness testing process. In this context, the Elliptic Curve Method (ECM) has been proposed by Lenstra [6] which has been implemented in few hardware architectures on FPGAs [3, 2]. In this work, we sketch a new multi-core ECM implementation for our COPACOBANA cluster which also makes heavy use of the arithmetic functions provided by the DSP-blocks in Virtex-4 devices.

The ECM is a factorization algorithm derived from J. Pollard's $p-1$ method and adapted by H. Lenstra on elliptic curves. Despite of Lenstra's original proposal, late implementations prefer to split the computation process in two phases. Both phases of the ECM can be efficiently implemented with only little memory in FPGAs what has been shown, e.g., in [3].

However, although relying on elliptic curves either, we *cannot* reuse the presented ECDSA core from the previous section, since ECM requires computation over an *arbitrary* modulus instead of a fixed one (cf. to the NIST prime P-256). Hence, to support arbitrary moduli, we decided to implement a high-radix Montgomery multiplication algorithm [7] and took again all efforts to shift as much of the arithmetic complexity into DSP-blocks as possible. Using the different opmodes of the DSP-blocks we realized multiply-and-accumulate functions in the hardcores for a fixed radix $h = 17$ which is determined by the maximum width of the DSP-block for unsigned multiplication.

For our implementation, we chose a multi-core design per FPGA similar to the one presented in [3]. For one core, we need, beside the arithmetic unit for modular multiplication and additions, control logic for computing a point multiplication required in phase 1 and additional ROM tables for phase 2. Since the ECM has two separate phases with separate control flow, we plan to implement an individual core each for computing phase 1 and phase 2. Note that this application is still an ongoing research project, but we can already provide performance figures as shown in Table 2 for the arithmetic units and compare our (preliminary) results to the implementation presented in [3].

Aspect	This work	[3]
Point Doubling	287 clk	n/a
Point Doubling & Addition	377 clk	947 clk
Clock Frequency	171 MHz	135 MHz
ECM Phase 1 (kP)	460 op/s*	140 op/s*

Table 2. Clock cycles and frequency for an ECM core (phase 1) to factor a 151 bit integer using a 980 bit scalar k on a Virtex-4 device

References

[1] T. Blum and C. Paar. High Radix Montgomery Modular Exponentiation on Reconfigurable Hardware. *IEEE Transactions on Computers*, 50(7):759–764, 2001.

[2] G. de Meulenaer, F. Gosset, M. M. de Dormale, and J.-J. Quisqater. Integer factorization based on elliptic curve method: Towards better exploitation of reconfigurable hardware. In *Proceedings of IEEE FCCM*, 2007.

[3] K. Gaj, S. Kwon, P. Baier, P. Kohlbrenner, H. Le, M. Khaleeluddin, and R. Bachimanchi. Implementing the Elliptic Curve Method of Factoring in Reconfigurable Hardware. In *CHES*, volume 4249, pages 119–133. LNCS, 2006.

[4] T. Güneysu and C. Paar. Ultra High Performance ECC over NIST Primes on Commercial FPGAs. In *Proceedings of CHES*, volume 5154, pages 62–78. LNCS, 2008.

[5] S. Kumar, C. Paar, J. Pelzl, G. Pfeiffer, and M. Schimmler. Breaking Ciphers with COPACOBANA - A Cost-Optimized Parallel Code Breaker. In *Proceedings of CHES*, pages 101–118. LNCS, 2006.

[6] H. Lenstra. Factoring integers with elliptic curves. *Annals Math.*, 126:649–673, 1987.

[7] P. L. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, 44(170):519–521, April 1985.

[8] National Institute of Standards and Technology (NIST). Recommended Elliptic Curves for Federal Government Use, July 1999.

[9] G. Orlando and C. Paar. A Scalable $GF(p)$ Elliptic Curve Processor Architecture for Programmable Hardware. In *Proceedings of CHES*, volume 2162, pages 348–363. LNCS, 2001.

[10] R. L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[11] D. Suzuki. How to Maximize the Potential of FPGA Resources for Modular Exponentiation. In *CHES Workshop*, volume 4727, pages 272–288. LNCS, 2007.