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. This number of computations is usually 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) [10].

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 to accelerate complex DSP operations (*DSP-blocks*). Recently, it has been shown how these DSP blocks can accelerate RSA encryptions [20] as well as attacks on RSA [4]. 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.

To enable our machine with support for those new hardcores, we have built a new COPACOBANA sup-

---

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

porting Virtex-4 devices. For the series of Virtex-4 FPGAs, Xilinx offers distinct devices optimized for different applications: FPGAs either being dedicated to fixed-point number computations with DSP-blocks (SX), providing PowerPC microprocessors and extensive communication facilities (FX) or offering a large number of configurable logic elements (LX).

In this contribution, we present a new cluster architecture capable to host these more powerful Virtex-4 FPGAs 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 show novel and parallel implementations for the generation of digital signatures over elliptic curves (ECDSA) [1] as well as for factoring mid-size integers using the Elliptic Curve Method (ECM) [11]. 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 the our cluster can tackle cryptographic challenges like the generation of digital signatures and factorization of midsized numbers and will also give estimates on the respective performance of both applications.

2. Related Work and Systems

COPACOBANA was designed as a large array of low-cost FPGAs and has been employed for wide range of cryptographic applications. For example, legacy symmetric ciphers like the Data Encryption Standard have been successfully broken in less than a week [10] and also subsequent attacks on related security applications involving One-Time Password Tokens [6], hard disk encryption (*Norton Diskreet*) [9] and Machine

Readable Travel Documents (*ePassport*) [12] have been shown. Besides, even cryptanalysis on asymmetric ciphers can be tackled by COPACOBANA, e.g., computing the Elliptic Curve Discrete Logarithm Problem [7] which is known as the fundamental primitive for cryptosystems based on elliptic curves.

COPACOBANA has been designed for providing a significant amount of computing resources to applications with only a minor demand on memory and communications. The majority of other FPGA-based computing clusters or supercomputers, however, focus on data-oriented applications requiring large amounts of memory and widely-dimensioned bandwidth. Examples for such universal supercomputing systems are Cray's XD1 system [3] as well as the SGI RASC technology [18] also including reconfigurable devices in their design. Unfortunately, such platforms are inappropriate for most tasks in cryptanalysis due to their high costs and the related non-optimal cost-performance ratio. Here, to the best knowledge of the authors, COPACOBANA is the only low-cost alternative to commercial supercomputers offering no nameable amount of memory but a significant amount of computing resources for less than € 10,000.

3. 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. However, taking power distribution, and routing constraints, signal integrity and mechanical packing into account, we found 16 plug-in modules each hosting 8 FPGAs a good choice 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 $27 \times 27 \ mm$. Based on this package, we can choose from four different Virtex-4 LX FPGAs (LX 15, 25, 40, 60) to support varying demands on configurable logic, from two SX FPGAs (SX 25 and 35) or three different FX devices (FX 20, 40, 60) providing a distinct amount
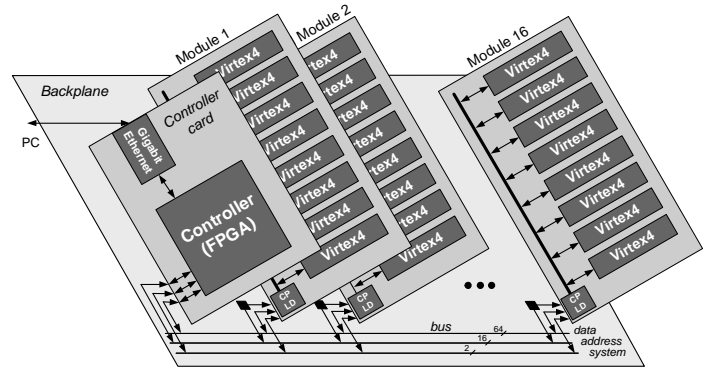
of DSP-blocks and PowerPCs, respectively. Note that although all devices share (nearly) the same footprint they are not all fully pin-compatible, i.e., switching to another device also requires extra routing efforts for some devices.

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 is able to do further, more sophisticated aggregation functions due to the availability of its embedded PowerPC. Finally, the condensed data is sent to the host computer via the Gigabit connection what is sufficient under the assumption that prior data reduction and aggregation has already taken place. The HCM and its corresponding tiers is shown in Fig. 1.
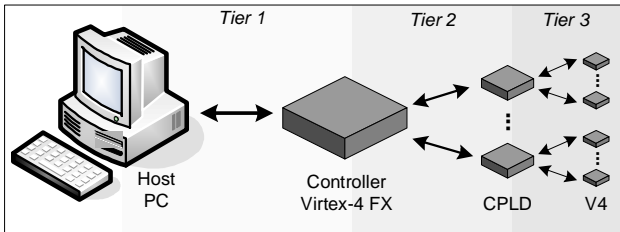
Figure 1. Hierarchical communication system on COPACOBANA v2 with 3 tiers

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. Conse-

quently, we chose a global DC power supply unit providing 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.

For monitoring purposes, we have also added a bit-serial bus throughout the system which complies to the SMBus specification. On each plug-in card the CoolRunner-II CPLD operates as monitoring client and runs all system management operations exclusively. The temperature measurement diode of all Virtex-4 devices is used to initiate an automatic power down when the core temperature is close to exceed the maximum value of 85°C. Hence, the CPLD is connected to all monitoring diodes of each Virtex-4 FPGA and also to the power enabler of the DC/DC converters to control the shut-down of the plug-in module in case of overheating.

Including all modifications, the architecture of the new FPGA-cluster (COPACOBANA v2) is depicted in Fig. 3.

4. 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 to accelerate the fundamental integer computations.

4.1 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) [1]. 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 [2, 15] can be considered as high-performance implementations but even so both take more than $3\,ms$ per operation.

An example an application, where *real* high-performance cryptography is required, is the future eHealth initiative in Germany where medical doctors will access the case history of a patient from a central, country-wide database via digitally signed messages generated by the patient's insurance (smart)card. Since the central database server must be able to cope with hundreds or thousands of digitally signed messages in peak times, a powerful signature verification facility is required. For this reason, we will present a considerably more efficient implementation taking advantage of the integrated DSP-blocks in the Virtex-4 devices, which employs 256 bit security parameters standardized by NIST [14] 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.

The ECDSA signature scheme according to [1] relies on the intractability of point multiplications kP over elliptic curves where P is a base point on the elliptic curve and k a scalar. For verifying ECDSA signatures accordingly, we need to extend the simple computation of kP to determine a linear combination $kP+lQ$ of two point multiplications. Within these point multiplications, all involved elliptic curve operations rely on basic arithmetic over a finite field, e.g., prime fields $GF(p)$ or binary extension fields $GF(2^m)$. In our ECDSA implementation we will focus on prime fields only, precisely, the prime field P-256 specified by NIST [14]. For an optimal implementation of ECDSA, it is cru-

cial to implement the fundamental arithmetic operations in the prime field as efficient as possible. Hence, we realized the required multi-precision modular multiplication, addition and subtraction *solely* with cascades of DSP-blocks within the Virtex-4 FPGA leading to a high operation frequency and a low demand on configurable logic. While the addition and subtraction are straightforward operations, we implemented the modular multiplications in two steps: firstly, we performed a full-product multiplication $c = a \cdot b$ using a serial-to-parallel DSP-based multiplier, and secondly, we reduced $c \bmod p_{256}$ using the fast reduction scheme presented in [14]. Note that the arithmetic of additions and subtractions for the reduction process is also completely implemented using the DSP-blocks where all operands are 256 bit integers. With the units for finite field arithmetic at hand, we realized the group operation on the elliptic curve using projective Chudnovsky coordinates so that a further modular inverter unit is not necessary. The point multiplications kP and $kP + lQ$ were determined with a straightforward implementation of the binary method (*double-and-add* algorithm), also incorporating Shamir's trick [8].

The ECDSA core has been successfully simulated and tested but we could not yet run it on the new COPACOBANA cluster since the hardware is not yet available (but already in production). Hence, we will also provide estimates based on the existing results of a single core implementation (all estimated figures are denoted by asterisks). Note that due to the consequent use of DSP-blocks a single core implementation can be operated up to a frequency of 490 MHz (max. delay $2.040ns$). However, we assume a multi-core architecture with 6 cores not to exceed more than 245 MHz in frequency (50% of single core implementation) due to longer routing paths and necessary unrelated logic packing. Our estimates for 6 ECDSA cores on an XC4VSX35 device and COPACOBANA, respectively, 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 our best knowledge, this parallel implementation of an asymmetric signature scheme on a cluster system seems to provide the highest throughput reported in the open literature.

4.2 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 [17] is a prominent example

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 (XC4VSX35)	4,840 op/s*
Operations kP (COPA.)	620,000 op/s*
Operations $kP + lQ$ (XC4VSX35)	4,000 op/s*
Operations $kP + lQ$ (COPA.)	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

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 [11] which has been implemented in few hardware architectures on FPGAs [19, 5, 4]. 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 an 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. The first phase is basically a point multiplication kP over elliptic curves, similar to the one presented in Subsection 4.1 taking a large scalar product $k = \prod \mu_i$ of all primes μ_i smaller than a specific boundary $\mathcal{B}$ as input. The second phase is an extension to the first increasing its efficiency by utilizing the idea of the birthday paradox. Luckily, both implementation of the ECM can be implemented with only little memory and have been already shown to be efficiently implemented in FPGAs [19, 5, 4].

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 [13] 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 the elliptic curve operations we used arithmetic over Montgomery curves what allows for simplified formulas and easier control flow.

In contrast to [4], we chose a multi-core design per FPGA similar to the one presented in [5]. 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 requires for two separate phases with separate control flow, we plan to implement an individual core each for computing phase 1 and phase 2. In the multi-core approach, this can be realized on the same FPGA with individual circuits for phase 1 and 2, or, in a larger scale, using the HCM and tier 3 communication among FPGAs on the same plug-in module. Although this application is still an ongoing research project, we can already provide performance figures as shown in Table 2 for the arithmetic units and compare our results to the implementation presented in [5]. It is evident that the DSP-block based arithmetic in our implementation allows for a higher performance compared to a conventional design.

Aspect	This work	[5]
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

We plan to provide more detailed and exact figures in the final version of this contribution (also including ECM phase 2).

5. Conclusion

The work at hand presents a novel cluster architecture populated with 128 Virtex-4 FPGAs (supporting LX, SX and FX devices) to run massively parallel applications with low memory requirements. Based on this cluster design we have implemented a crypto accelerator for ECDSA signatures providing the enormous throughput of more than 150 MBits/s for high security applications with standardized 256 bit parameters. Furthermore, we have outlined an implementation of a ECM factorization circuit which also benefits from the new features of our cluster architecture like the integrated DSP-blocks and the hierarchical communication system. However, it should be remarked that the modified hardware can be adopted to any suitable task with little requirements on memory and is thus not restricted only to code breaking. Hence, when taking a demand on memory and massive bandwidth out of consideration, COPACOBANA turns out as an alternative to supercomputers for this specific set of applications – still at much lower costs.

Future work includes further finalization and optimization of the parallel implementations of the presented cryptographic algorithms. Moreover, we are already looking for applications beyond cryptography. A potential opportunity for our modified FPGA cluster might arise in the field of biology: here, we plan to apply the Smith-Waterman algorithm [21, 16] for scanning sequences of DNA or RNA against databases.

References

[1] ANSI X9.62-2005. American National Standard X9.62: The Elliptic Curve Digital Signature Algorithm (ECDSA), 2005.

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

[3] Cray. Cray XD1 Supercomputer, 2008. `http://www.cray.com/downloads/FPGADatasheet.pdf`.

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

[5] 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.

[6] T. Güneysu and C. Paar. Breaking Legacy Banking Standards with Special-Purpose Hardware. In *Proceedings of Financial Cryptography*. LNCS, 2008.

[7] T. Güneysu, C. Paar, and J. Pelzl. Attacking elliptic curve cryptosystems with special-purpose hardware. In *Proceedings of FPGA*, pages 207–215. ACM Press, 2007.

[8] D. R. Hankerson, A. J. Menezes, and S. A. Vanstone. *Guide to Elliptic Curve Cryptography*. Springer, 2004.

[9] S. Kumar, C. Paar, J. Pelzl, G. Pfeiffer, A. Rupp, and M. Schimmler. How to Break DES for € 8,980. In *SHARCS Workshop*, 2006.

[10] 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.

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

[12] Y. Liu, T. Kasper, K. Lemke-Rust, and C. Paar. E-passport: Cracking basic access control keys. In *Proceedings of OTM Conferences*, volume 4804, pages 1531–1547. LNCS, 2007.

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

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

[15] 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.

[16] G. Pfeiffer, H. Kreft, and M. Schimmler. Hardware Enhanced Biosequence Alignment. In *International Conference on METMBS*, pages 11–17. CSREA Press, 2005.

[17] 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.

[18] SGI. SGI RASC Technology, 2008. `http://www.sgi.com/products/rasc/`.

[19] M. Šimka, J. Pelzl, T. Kleinjung, J. Franke, C. Priplata, C. Stahlke, M. Drutarovský, V. Fischer, and C. Paar. Hardware Factorization Based on Elliptic Curve Method. In *Proceedings of IEEE FCCM*, pages 107–116, 2005.

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

[21] C. Yu, K. Kwong, K. Lee, and P. Leong. A Smith-Waterman Systolic Cell. In *Proceedings of FPL 2003*, pages 375–384. Springer, 2003.