Evaluation of Cryptographic Algorithms over an All Programmable SoC (AP SoC) Device

Ivan Gutierrez Agramont
Computer Engineering Lab, Universidad Católica Boliviana "San Pablo"
La Paz, Bolivia

and

Humberto Calderón
Computer Engineering Lab, Universidad Católica Boliviana "San Pablo"
La Paz, Bolivia

ABSTRACT

This paper runs an evaluation of cryptographic algorithms (block ciphers and public key ciphers) in an All Programmable SoC (AP SoC) device, comparing time, power consumption on two different test beds a PC and the AP SoC, with a taxonomy work done comparing all the previous papers about cryptographic algorithms implemented on FPGAs. The outcomes show that AES and RSA are the fastest in user time and both being well known cryptographic algorithms being AES the fastest with 39139 sec. and RSA with 58964 sec. which means that in 1 sec; 2.55 AES algorithm is run and 1.69 RSA is run in the AP SoC that consumes 0,279W for 100K iterations (Abstract).

Keywords: cryptography, SoC, security, IoT, fpga, zynq.

I. INTRODUCTION

We inhabit into the digital era, our life style, work style and even our human relations depend on the communication channel that we use, and the security that it can provide us [32]. Security in communication channels is an important research area, it seeks to create trusted systems or secure systems with continuously evolving mechanisms that prevents data theft or privacy concerns. The domain problem includes protocol improvements [7], development of customized hardware [8], with time demanding sophisticated algorithms which base their functionality in cryptographic implementations [43-46] that need to operate in a world with ubiquitous communication systems [60].

Security in communications is an ancient issue that requires to be solved, since the Babylonians, ancient Greek, and Rome, all these ancient empires have developed their own cryptographic technology, historically, modern cryptography started after World War II, it was considered more as an art than science itself [30] and Shannon has introduced the mathematical basis for modern cryptography.

Prior to that time, all useful modern encryption algorithms had been symmetric key algorithms, in which the same cryptographic key is used with the underlying algorithm by both the sender and the receiver, who must both keep it secret. All the electromechanical machines used in WWII were of this logical class, as were the Caesar and Atbash ciphers and essentially all cipher systems throughout ancient history.

In the 1970s there were two major advances, the first was the publication of the draft "Data Encryption Standard" in the U.S. Federal Register [3], the aim was the creation of secure electronic communications, especially for the banking business. The outcomes were adopted and published as the "Federal Information Processing Standard" in 1977. DES [31] was the first publicly accessible cipher to be impulse by a national agency such as NSA. The release of its specification by NBS stimulated an explosion of public and academic interest in cryptography. The second advance was the development, in 1976, that has fundamentally changed the way cryptosystems might work. This was the publication of the paper "New Directions in Cryptography" by Whitfield Diffie and Martin Hellman; it introduced a new method of distributing cryptographic keys, which went far towards solving one of the fundamental problems of cryptography, key distribution, and has become known as "Diffie-Hellman key exchange" [1]. The article also stimulated the almost immediate public development of a new class of ciphering algorithms, the asymmetric key algorithms.

The aged DES was officially replaced by the Advanced Encryption Standard (AES) in 2001, when NIST

announced FIPS 197 [11]. After the open competition [52], NIST selected Rijndael as the winner work submitted by two Belgian cryptographers, to be the AES. DES and more secure variants of it such as Triple DES, blowfish and serpent amongst others are still used today, having been incorporated into many organizational standards. However, a brute-force attack, undertaken by the cyber civil-rights group Electronic Frontier Foundation in 1997, succeeded to break DES in 56 hours [5]).

In this new century, Secure Systems (SS) are surpassing the local area networks (LAN) applications, the Internet and internetworking, Satellite Communications, Infrastructure wireless communications (microwave links and mobile links), RFID in the supply chain or retail [71], they are important for the emerging Internet of Things (IoT), especially when we address the smart cities at the consumer level (Wi-Fi, WiMAX, Bluetooth, etc.) and TV stations (in a ubiquitous way). The last implies that there is a renewed interest of the development of cryptographic machines for the embedded systems scenarios.

From the implementation point of view, basically, most solutions were software based, so the cryptographic algorithm has been implemented as a piece of software (encryption software) like Disk encryption software, file/folder encryption (e.g. TrueCrypt or any of its forks, LUKS [54]), Network traffic encryption (e.g. IPSec [53], OpenVPN [55]) amongst the most used applications/software. Hardware-based solutions have been used and are continuously reaching new states, since the 90s the industry has been using cryptographic devices such as CVAS III approved by the NSA with algorithms like DES and AES mostly for voice communications.

Recently advances in hardware solutions have reached their maturity. The 2000s depict a series of works that include the “Trusted Platform Module (TPM)”, which is used to implement the concept of “Trust Computing” [9]. So, TPM is an international standard based on hardware and software that enables trust in computing platforms in general [10]. The TPM technical specification was written by a computer industry consortium called the “Trusted Computing Group (TCG)”.

Recently, the SS are being supported for the emerging Reconfigurable-Technologies, an important point of view is stated in “Secure Systems on Reconfigurable Hardware”, this work is implemented under the idea of different security levels, where one of the most important is to create buffers using AES [12] on reconfigurable hardware but doesn't get involved with embedded systems. Also, this technology was used to evaluate cipher algorithms like AES finalists [23], or more recent the creation of a coprocessor in reconfigurable technology [33, 34], [42] and [45] amongst the most important and was used for public key cryptography [10], [35 – 37].

Understanding the problem

Generally speaking, when you send sensitive data over a public network, there is always a chance that someone can eavesdrop or tamper the data. There are several proposals to solve the problem presented, however their compatibility with the requirements of the industry, e.g. "Trusted Computing" [9], [10] or "Hardware-Assisted Security" [13] is applicable on the Internet, LAN internetworking and essentially communication between two computers or more.

So, this means that there is work related to secure systems on a wider application, and it is this type of security application that can be implemented in other communications such as satellite, microwave, mobile or any current or future communication like "Internet of Things" or even "Smart Cities" [58 - 60].

The beggining of our roadmap: stepping the pace

In this paper, we study and categorize a group of potential cryptographic algorithms susceptible to be accelerated over an All Programmable System-on-Chip (AP SoC) device. The literature review has lead us to choose and evaluate the behaviour of AES and blowfish, 3DES as block ciphers and RSA as a public key cryptographic algorithm over an AP SoC device. Specifically, we have instantiated the aforementioned software algorithms over a personal computer, and we have also tested them over a Processing System (PS) in a Zynq device [72]. Our aim, looks for the development of a general method of hardware instantiation of secure systems (e.g. cryptographic machines) in different Embedded Systems applications, with limited budgets on both silicon use and power consumption and in this preliminary work we are evaluating the behaviour of the chosen algorithms in both scenarios, the high-end computing and embedded system.

The rest of this paper presents a taxonomy of current solutions categorizing them from the point of view of computational load vs. security, we present also the outcomes of the tested evaluations. In section 2 we outline the taxonomy of current solutions over Reconfigurable Technologies. Section 3 depicts the experimental results of the AES, RSA, and the rest of the algorithms all running on a PC and Zynq device. Finally, the article is concluded in Section 4.

II. BACKGROUND

Most of the solutions, use hardware but mainly software together in a way that all the data traffic is analysed, processed, and accepted, for instance, firewalls, routers, layer 3 switches, all of them work in that way.

TABLE I. SECURITY BASED ON TECHNOLOGY AND TYPE OF ATTACK [27]

Technology	Attacks	Abstraction Level	Countermeasures
Software	Software Attacks (Viruses, Trojan Horses, Worms)	Application	Microkernel
		Operating System	Memory Access Control
			Monitoring
Hardware	Board level attacks (bus probing, memory tampering)	System	Memory Access Control Monitoring
	Differential Fault Analysis (DFA) Timing Attack	Architecture	Encryption Integrity Cheking & Authentication Fault Tolerance, Twice encryption
	Differential Power Analysis (DPA) Electromagnetic Analysis (EMA)	Logic	Fixed time or unpredictable delay Random execution Dual rail - Triple track
		Techno	Asynchronous

When a secure channel is needed (based on the requirements of the security, see Table I), the same hardware is used, which that upper layers use for their specific routines like securing the channel. That piece of software (upper layers) use some protocols and those protocols use an encryption algorithm [12]; for all the above, new solutions try to isolate a piece of hardware and its only function will be security functions; in order to design and implement that hardware, first, we need to know and compare all the work related to cryptographic devices based on 2 premises, block ciphers (e.g. AES because is an standard) and public key cipher (e.g. RSA because is widely used); summarized as follows

Block ciphers on FPGAs

FPGA devices have become a promising alternative for implementing cryptographic algorithms due to their good performance and great flexibility as shown above, and that matches extremely well with block ciphers operations (e.g. bit-permutation, bit-substitution) and those operations can be executed more efficiently that in general purpose processors [33].

The following tables (Tables II - V) summarizes some important data about AES implementation based on the work of various authors where we must be aware that the structural differences between FPGA types can have an incident in the performance.

TABLE II. AES BASIC IMPLEMENTATION COMPARISON

Design	FPGA	Slices	Freq. MHz	Performance Gbps
Güneysu [34]	Virtex-5	93	550	1,76
Good et. al. [41]	Spartan-II	67	67	0,002
Chodowiec et. al. [43]	Spartan-II	222	60	0,166
Rouvoy et. al. [49]	Spartan-3	163	71	0,208
Algotronix [44]	Virtex-5	161	250	0,8

In Table II, we saw different implementations and we can realize that the solutions range from 60MHz. to 550MHz. with performances ranging from 0.002 Gbps to 1.76 Gbps being the fastest.

TABLE III. AES ROUND IMPLEMENTATION COMPARISON

Design	FPGA	Slices	Freq. MHz	Performance Gbps
Güneysu [34]	Virtex-5	277	485	6,21
Standaert et. al. [50]	Virtex-E	2257	169	2,008
Helion [46]	Virtex-5	349	350	4,07
Bulens et. al. [45]	Virtex-5	400	350	4,1
Chaves et. al. [42]	Virtex-II (PRO)	515	182	2,33

Table III shows another type of AES implementation (mode), the frequency ranges from 169 MHz to 485 MHz and performances starting at 2.008 Gbps to 6.21 Gbps.

TABLE IV. AES UNROLLED IMPLEMENTATION COMPARISON

Design	FPGA	Slices	Freq. MHz	Performance Gbps
Kotturi et. al. [51]	Virtex-II (PRO)	10816	126	16
Järvinen et. al. [48]	Virtex-II	10750	139	17,8
Hodjat et. al. [47]	Virtex-II (PRO)	5177	168	21,5
Chaves et. al. [42]	Virtex-II (PRO)	3513	272	34,7

Finally, we can realize from table IV that AES Unrolled implementation have frequencies ranging from 126 MHz up to 272 MHz reaching performances from 16 Gbps up to 34.7 Gbps.

We can also state that not all the chip is being used for the AES implementation (in the three tables above), and we can realize also that in general the best performance implementations tend to be the smallest ones too; this can leave an unused space for other kind of implementations, like RSA or other stuff that might be needed.

TABLE V. BLOCK CIPHER ALGORITHM BY NATURE IMPLEMENTED ON FPGA [23]

Design	Algorithm	Throughput (Mbps)	Area
Dandalis et. al.	MARS	101,88	6896
	RC6	112,87	2650
	Rijndael	353	5673
	Serpent	148,95	2550
	Twofish	173,06	9363

Literature shows some important comparison especially in throughput of different Block Ciphers Algorithms. Rijndael known as AES has a great performance and uses an area of 5673 [33].

Public key ciphers

Since Diffie and Hellman's breakthrough work about a new way of encrypting using a public key, lots of efforts

have been made so we can have that cryptographic technology on FPGAs due to their performance and flexibility even though these algorithms are time-consuming specially in decryption due the nature of the RSA algorithm (Eq. 1) [2], [16].

$n = p * q;$
$\varphi(n) = (p - 1) * (q - 1);$
Choose **e** such that $1 < e < \varphi(n)$ && **e** and **n** are coprime.
Public key is (e, n);
Private key is (d, n);
Encryption $C = M^e \bmod n$
Decryption $M = C^d \bmod n$

Eq. (1)

TABLE VI. RSA IMPLEMENTATION ON FPGA WITH EXPONENTIAL, CHINESE REMAINDER THEOREM AND ELLIPTIC CURVE CRYPTOGRAPHY

Design	FPGA	Slices	Freq. MHz	Exp. Time (ms) 1024 bits	CRT Time (ms) 1024 bits	ECC Time (ms)
Mentens 160 bits [10]	XC2VP30	2118	183			1.09 (ECC=160)
Mentens 256 bits [10]	XC2VP30	3171	182			1.10 (ECC=160)
Mentens 1024 bits [10]	XC2VP30	9090	152	2,33	0,64	1.32 (ECC=160)
G. Orlando and C. Parr [35]	XCV1000E	5735	40			3.00 (ECC=192)
K. Sakiyama et. al. [36]	XC2VP30	8954	100			1.04 (ECC=160)
	XC2VP30	10847	100			2.70 (ECC=256)
D. N. Amanor et. al. [37]	XVC2000E	4608	69	22,7	6,1	
C. McIvor et. al. [38]	XC2V6000	24767	101		2,63	
K Kelley and D. Harris [39]	XC2V2000	32MULTs + 2593LUTs	135	5		
	XC2V2000	8 MULTs + 695LUTs	135	17		
S. H. Tang et. al. [40]	XC2V4000	14334	90	2,33	0,66	

Table VI shows that RSA with Chinese Remainder Theorem is the fastest of all implementation ranging from 0.64 ms. to 6.1 ms.

Finally, we must state that all these cryptographic implementations were on FPGAs laboratory only; this means that they were not implemented with commercial or real live applications, like we intend to do.

Other Cryptographic Devices

Now there is a trade-off between security and computational process, because symmetric encryption uses less microprocessor power compared with asymmetric encryption, but the latter is more secure because is harder to break due to the nature of the algorithm [11], [16].

In recent years, some new applications implemented both types of encryption at the same time (which is very likely), applications such as GnuPG [56] uses DSA / BLOWFISH, i.e., using an asymmetric key to authenticate communication and after that is performed satisfactorily, the symmetric key is used for data traffic, there is a fairly new implementation AES-NI[13], even though they have achieved good performance about two times faster in a specific processor (around 1.3 cycles/byte) they used a CPU with multiple cores with Hyper-threading; and this speed is achieved only in the encryption process and not in the decryption, this due to the algorithm nature and the instructions created for that processor. Although it's main focus is aiding software that will use this algorithm (AES) on this processor architecture [65], this work and set of instructions is all new.

Once again, both solutions (devices and software) are based for Internet and internetworking only, while GPG hybrid idea is interesting, this is only used in file encryption using computer software, which can be improved through the implementation of reconfigurable hardware to work with specific encryption algorithm like AES and RSA for quick work of encryption and decryption. The advantage of creating custom systems is that these are CPU independent [20] and can be used in different types of environments such as public networks, private, wireless, satellite networks, microwave, and so on [14].

III. EXPERIMENTAL RESULTS

The literature reviewed led us to choose AES and RSA because; RSA is computationally faster in decryption and equally safe in terms of algorithm compared with DSA and also it has been used for long time now [6] and AES is a standard adopted in several countries like the US [4] and the algorithm has proved strong against various types of attacks [11].

We studied AES first because it's important how this algorithm encrypts all data being sent. First, we implement an AES algorithm, later we also implemented Blowfish, 3DES and RSA, all developed in Python (without any security library) due its fast coding nature under a computer with a multipurpose processor (x86) running a Linux Operating System with these characteristics (see table VII):

TABLE VII. COMPUTER CHARACTERISTICS

RAM	15947 MB.
CPU Cores	8
CPU Type	Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz
CPU Cache	8192 KB

Then we ran some iterations of each algorithm 10, 100, 1k, 10k and 100k times, in a computer as well as in the test bed (AP SoC), these are the results as a time

consumed for the system and in this paper we are showing only the results for 10k and 100k.

TABLE VIII. TIME AND POWER CONSUMED FOR 10K ITERATIONS (COMPUTER)

Algorithm	User CPU time (sec)	System CPU time (sec)	Elapsed Clock Time (MM:SS)	Percent of CPU used	Physical Memory (KB)	power usage (Watt)
blowfish	1174,240	25,790	19:59.09	100%	6128,000	22,6884
aes	264,370	27,410	4:51.03	100%	6656,000	
3des	416,390	21,000	7:16.54	100%	5340,000	
rsa	383,200	49,590	7:12.62	100%	10256,000	

As shown in the above table (Table VIII), AES and RSA are the fastest algorithms (in that order) for 10K run under the same system and also for 100K (Table IX).

TABLE IX. TIME AND POWER CONSUMED FOR 100K ITERATIONS (COMPUTER)

Algorithm	User time (sec)	System time (sec)	Elapsed Clock Time (HH:MM:SS)	Percent of CPU used	Physical Memory (KB)	power usage (Watt)
blowfish	11919,460	315,700	03:23:46	100%	20148,000	21,5412
aes	2757,810	312,350	00:51:03.95	100%	20148,000	
3des	4331,190	259,410	01:16:22	100%	20152,000	
rsa	3976,370	564,490	01:15:40	100%	20148,000	

Now we show the result over an All Programmable System-on-Chip (AP SoC) device as a test bed, similar to the previous, we used the same python code and here we also used a Linux embedded system inside. The board has these characteristics:

TABLE X. AP SOC CHARACTERISTICS

RAM	499 MB
CPU Cores	2
CPU Type	ARMv7 Processor rev 0 (v7l)
CPU Cache	none

The same idea than in the computer was used here, 10, 100, 1k, 10k, 100k iterations for each algorithm was ran in the test bed, outcomes for 10K and 100K are presented on table XI and XII.

TABLE XI. TIME AND POWER CONSUMED FOR 10K ITERATIONS (AP SOC)

Algorithm	User time (sec)	System time (sec)	Elapsed Clock (HH:MM:SS)	Percent of CPU used	Physical Memory (KB)	power usage (Watt)
blowfish	19518,160	184,820	05:29:58	99%	17328,000	0,279
aes	3916,210	195,520	01:09:13	99%	18384,000	
3des	7013,170	193,210	02:00:27	99%	16016,000	
rsa	5895,020	1349,990	02:01:36	99%	25904,000	

TABLE XII. TIME AND POWER CONSUMED FOR 100K ITERATIONS (AP SOC)

Algorithm	User time (sec)	System time (sec)	Elapsed Clock (HH:MM:SS)	Percent of CPU used	Physical Memory (KB)	power usage (Watt)
blowfish	194709,500	2070,800	54:48:52	99%	78832,000	0,279
aes	39190,990	2100,520	11:37:10	99%	78832,000	
3des	70170,450	2123,630	20:09:33	99%	78832,000	
rsa	58964,200	13848,880	20:21:55	99%	78832,000	

The time showed in this paper (**User and System)** will tell us how much actual CPU time the process used. Note that this is across all CPUs.

User is the amount of CPU time spent in user-mode code (outside the kernel) within the process. This is the only actual CPU time used in executing the process. Other processes and time the process spends blocked do not count towards this figure.

System is the amount of CPU time spent in the kernel within the process. This means executing CPU time spent in system calls within the kernel, as opposed to library code, which is still running in user-space. Like 'user', this is only CPU time used by the process.

Elapsed wall clock is the total time elapsed, not only system+user.

Physical Memory is the maximum amount of physical memory this process has used during the life of the process

Percent of CPU used is the actual percentage of CPU that was given to the process, the equation is (User + System) / Elapsed Wall Clock.

With these results, now we can compare each algorithm and we can determine the following table.

TABLE XIII. COMPARATIVE PERFORMANCE OF X86 PROCESSOR AND AP SOC IN TERMS OF TIME AND POWER CONSUMTION FOR 100K ITERATIONS

Algorithm	power usage (Watt)	User time (sec)	Performance (Speed)	power usage (Watt)	User time (sec)	Performance (Speed)
blowfish	21,5412	11919,460	1	0,279	194709,500	1
aes		2757,810	4		39190,990	5
3des		4331,190	2,5		70170,450	3
rsa		3976,370	3		58964,200	4

These results mean that the proposed algorithms, based on bibliographic research, were adequate and more on, AES is faster in both testbeds, the PC and the AP SoC and the result in terms of performance is similar (without looking at power consumption); and similar to the previous algorithm happens to the rest, being RSA the second fastest.

IV. CONCLUSIONS

With these results, we expect to show that the proposed solutions can be used in different embedded ubiquitous environments such as:

- Internetworking and Internet Communications.
- Satellite Communication.
- Mobile Communications (Mobile networks, Smart Phones, or Tablets).
- Air/Water mobile Communications (Drones, security or imagery hot air balloons, security airplanes and helicopters) amongst the most relevant [21].
- Security for RFID
- Security for devices involved in Blockchain
- Secure Communications in "not yet" implemented technologies like "Internet of Things" or "Smart Cities"

According to these results, the computer architecture is faster than the AP SoC, but this is not entirely a fair comparison because the Computer CPU consumes more power (22,11 W average) compared to the ARMv7 for the AP SoC; that consumes only 0,279W according to [57]. Even though the computer was fast, also the use of power usage was higher (almost 100x). In this low power consumption AP SoC, it has been demonstrated that in 1 second, an AES algorithm can run 2.55 times and an RSA algorithm can run 1.69 times.

So in terms of time, the system time (kernel time within the process) is good in the test bed, even though is about 100x slower (compared to the computer sys time), is still a good time for an embedded system over an ARMv7 processor without any special development in the hardware (co-processor), then, this shows a feasibility to use FPGA designed specifically for encryption/decryption even better for AES and RSA (the fastest algorithms in that order), so this architecture will help to off load the hard cryptographic work into this co-processor, do its job and then upload the data encrypted to the CPU, so the time is expected to be fast with a better throughput than previous works showed above.

It is anticipated that the proposed encryption module for SS will be done through a mixed logic, i.e. two different encryption algorithms, one is used to establish communication between any 2 points (RSA) and the other for sending data between these points (AES). Both algorithms were chosen, in an initial search and thanks to the taxonomy and the tests described above, also both are widely used in an Operating System level, and they are fast in encryption/decryption processes (like showed above), this last feature is very important specially in timing [23].

All the research and development of this paper were conducted with good practices in the industry and especially in the cryptographic area with a lot of simulation and tests with a premise like "*2n for n security key*"[22] in all encryption/decryption tests; also, the design and the implementation of the entire system (Secure Systems) will be executed in a way that all the system must be secure against known attacks like Side Channel Attacks [25,67-69], non-invasive Side Channel Analysis [66] or Electro Magnetic Attacks [70] , (Single Power Analysis SPA[26] and Differential Power Analysis DPA), Differential Electromagnetic Analysis [24], Timing Attacks [15] amongst the main types of attacks.

Even though, Reconfigurable Hardware like AP SoC can suffer different kinds of attacks [28], they offer some potential advantages [29] like algorithm agility, algorithm upload, architecture efficiency, resource efficiency, algorithm modification, throughput, and cost efficiency [27] and we must take advantage of the Table I showed before.

It should be noted that all the work studied in the literature were FPGA only solutions, which are not valid for ecosystems like IoT or Smart Cities due to power consumption, but in the other hand all the IoT applications that are emerging right now needs to be secure, but the level of security or type of security will be determined by the data and usage given to the solution, non the less, it looks like AP SoCs can give us that customized security for the IoT communications which no FPGA can provide.

Thus, is posed to obtain a **product** that can be used in the areas mentioned before, where the demand for security technologies with **lower silicon consumption, smaller size and higher energy efficiency** will be the result of this research as these features are the future for all the ubiquitous environments that are being developed right now.

References

[1] Whitfield Diffie and Martin Hellman, "New Directions in Cryptography", IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976.

[2] Nele Mentens (2007) PhD Thesis. Secure and Efficient Coprocessor Design for Cryptographic Applications on FPGAs

[3] FIPS PUB 46-3 Data Encryption Standard (DES). Third edition, 1999.

[4] Daemen, Joan; Rijmen, Vincent. "AES Proposal: Rijndael". National Institute of Standards and Technology. February 2013.

[5] Whitfield Diffie and Martin Hellman, "Multi-user cryptographic techniques" Diffie and Hellman, AFIPS Proceedings 45, June, 1976.

[6] D. Boneh. Twenty years of attacks on the RSA cryptosystem. Notices of the American Mathematical Society, 46(2), 1999

[7] Mark Dermot Ryan, The University of Birmingham, "Secure protocols", Computer Security lecture notes http://www.cs.bham.ac.uk/~mdr/teaching/modules04/security/lectures/protocols.html, 2004

[8] Trusted Computing Group, "Main Specification Version 1.1b," February 2002.

[9] Trusted Computing Group, "TCG Specification Architecture Overview," April 2004.

[10] "ISO/IEC 11889-1:2009". ISO.org. International Organization for Standardization. Retrieved 29 November 2013.

[11] Dan Boneh, "The AES Block Cipher", Stanford University.

[12] Huffmire, T., Brotherton, B., Callegari, N., Valamehr, J., White, J., Castner, R., y Sherwood, T., "Designing secure systems on reconfigurable hardware", ACM Trans. Des. Autom. Electron. Syst. 13, 3, Article 44, ACM 2008

[13] Leslie Xu “Securing the Enterprise with Intel® AES-NI”, Intel Corporation 2010
[14] Huffmire, T., Levin T., Nguyen T., Irvine C., Brotherton B., Wang G., Sherwood T., y Kastner R., “Security Primitives for Reconfigurable Hardware-Based Systems”, ACM Trans. Des. Autom. Electron. Syst. 3, 2, Article 10, ACM 2010
[15] P. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems. In N. Koblitz, editor, Advances in Cryptology – Proceedings of CRYPTO, number 1109 in Lecture Notes in Computer Science. Springer-Verlag, 1996
[16] RSA Cryptography Standard v 2.2, RSA Laboratories October 2012
[17] Kocher, Paul; Jaffe, Joshua; Jun, Benjamin. - Cryptography Research, Inc "Cryptography Research Q&A on Differential Power Analysis" 1998, 1999.
[18] Ross Anderson, Markus Kuhn "Tamper Resistance - a Cautionary Note" The Second USENIX Workshop on Electronic Commerce Proceedings, Oakland, California, November 18-21, 1996.
[19] Ross Anderson, Markus Kuhn "Low Cost Attacks on Tamper Resistant Devices" Security Protocol Workshop. April 1997. M Lomas et al. (ed.), Security Protocols, 5th International Workshop, Paris, France, April 7-9, 1997, Proceedings, Springer.
[20] M Maclean and J Moore, “Securing FPGAs for red/black systems: FPGA-Based single chip cryptographic solution”, Military Embedded Systems Mag. 2007
[21] Departament of Defense, United States of America, “Unmanned Systems Integrated Roadmap FY2011-2036”, 2011
[22] Justin Troutman, “Ideal-to-Realized Security Assurance In Cryptographic Keys”, Parts 1 and 2, Article, WindowSecurity.com 2005
[23] Andreas Dandalis, Viktor K. Prasanna and Jose D.P. Rolim, “A Comparative Study of Performance of AES Final Candidates Using FPGAs”, Cryptographic Hardware and Embedded Systems - CHES 2000 Second International Workshop Worcester, MA, USA, August 17–18, 2000 Proceedings
[24] Agrawal, D., Rohatgi, P., Rao, J.R.: Multi-channel Attacks. Lecture Notes in Computer Science 2003
[25] Guilley, S., Pacalet, R.: SoC security: a war against side-channels. Annals of the Telecommunications. Système sur puce Électronique pour les Télécommunications 2004
[26] Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. Lecture Notes in Computer Science 1999
[27] Benoit Badrignans, Jean Luc Danger ,Viktor Fischer, Guy Gogniat and Lionel Torres.: Security Trends for FPGAS From Secured to Secure Reconfigurable Systems, Springer 2011
[28] Wollinger, T., Paar, C.: How secure are FPGAs in cryptographic applications. In: Proceedings of the 13th International Conference on Field-Programmable Logic and Applications 2003
[29] Wollinger, T., Paar, C.: Security aspects of FPGAs in cryptographic applications. In: Lysaght, P., Rosenstiel, W. (eds.) New Algorithms, Architectures and Applications for Reconfigurable Computing, Springer, 2005
[30] Berlekamp, Elwyn; Solomon W. Golomb, Thomas M. Cover, Robert G. Gallager, James L. Massey, and Andrew J. Viterbi (January 2002)."Claude Elwood Shannon (1916–2001)".Notices of the AMS 49 (1): 8–16. Retrieved 18 September 2013.
[31] Schneier. Applied Cryptography (2nd ed.). p. 280.
[32] Humberto Calderón, Christian Elena and Stamatis Vassiliadis. Soft Core Processors and Embedded Processing: a survey and analysis
[33] Andreas Dandalis, Viktor Prasanna and Jose Rolim. A Comparative Study of Performance of AES Final Candidates Using FPGAs, CHES 2000, Spinger.
[34] Tim Erhan Güneysu Cryptography and Cryptanalysis on Reconfigurable Devices, Security implementations for hardware and Reprogrammable devices, 2009.
[35] G. Orlando and C. Paar. A scalable GF(p) elliptic curve processor architecture for programmable hardware. In C¸ .K. Ko¸c, D. Naccache, and C. Paar, editors, Proceedings of the 3rd International Workshop on Cryptograpic Hardware and Embedded Systems (CHES), number 2162 in Lecture Notes in Computer Science. Springer-Verlag, 2001.
[36] K. Sakiyama, E. De Mulder, B. Preneel, and I. Verbauwhede. A parallel processing hardware architecture for elliptic curve cryptosystems. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2006
[37] D. N. Amanor, V. Bunimov, C. Paar, J. Pelzl, and M. Schimmler. Efficient hardware architectures for modular multiplication on FPGAs. In Proceedings of the 15th International Conference on Field Programmable Logic and Applications (FPL), IEEE, 2005
[38] C. McIvor, M. McLoone, J. V. McCanny, A. Daly, and W. Marnane. Fast Montgomery modular multiplication and RSA cryptographic processor architectures. In Proceedings of the 37th Annual Asilomar Conference on Signals, Systems and Computers, 2003
[39] K. Kelley and D. Harris. Parallelized very high radix scalable Montgomery multipliers. In Conference Record of the Thirty-Ninth Asilomar Conference on Signals, Systems and Computers, 2005
[40] S. H. Tang, K. S. Tsui, and P. H. W. Leong. Modular exponentiation using parallel multipliers. In Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT), 2003
[41] T. Good and M. Benaissa. AES on FPGA from the fastest to the smallest. In J. R. Rao and B. Sunar, editors, Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES 2005), volume 3659 of LNCS. Springer-Verlag, 2005
[42] R. Chaves, G. Kuzmanov, S. Vassiliadis, and L. Sousa. Reconfigurable memory based AES co-processor. In Proceedings of the Workshop on Reconfigurable Architectures (RAW 2006), 2006.
[43] P. Chodowiec and K. Gaj. Very compact FPGA implementation of the AES algorithm. In C. D. Walter, C¸ . K. Ko¸c, and C. Paar, editors, Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES 2003),volume 2779 of LNCS. Springer-Verlag, 2003.
[44] Algotronix Ltd. AES G3 data sheet: Xilinx edition, October 2007. http://www.algotronix-store.com/kb_results.asp?ID=7
[45] P. Bulens, F.X. Standaert, J.-J. Quisquater, P. Pellegrin, and G. Rouvroy. Implementation of the AES-128 on Virtex-5 FPGAs. In S. Vaudenay, editor, Proceedings of First International Conference on Cryptology in Africa AFRICACRYPT 2008, volume 5023 of LNCS Series. Springer-Verlag, 2008.
[46] Helion Technology. High performance AES (Rijndael) cores for Xilinx FPGAs, 2007. http://www.heliontech.com/downloads/aes_xilinx_helioncore.pdf.
[47] A. Hodjat and I. Verbauwhede. A 21.54 Gbits/s fully pipelined AES processor on FPGA. In Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM 2004). IEEE Computer Society, 2004.
[48] K. Järvinen. Studies on High-Speed Hardware Implementations of Cryptographic Algorithms. PhD thesis, Helsinki University of Technology, 2008.
[49] G. Rouvroy, F.-X. Standaert, J.-J. Quisquater, and J.-D. Legat. Compact and Efficient Encryption/Decryption Module for FPGA Implementation of the AES Rijndael Very Well Suited for Small Embedded Applications. International Conference on Information Technology: Coding and Computing, 2004
[50] F.-X. Standaert, S. B. Örs, and B. Preneel. Power analysis of an FPGA implementationof Rijndael: Is pipelining a DPA countermeasure? In M. Joye and J.-J. Quisquater, editors, Proceedings of the Workshop on Cryptograpic Hardware and Embedded Systems (CHES 2004), volume 3156 of LNCS. Springer-Verlag, 2004.
[51] D. Kotturi, Seong-Moo Yoo and J. Blizzard. AES crypto chip utilizing high-speed parallel pipelined architecture.

ISCAS 2005. IEEE International Symposium on Circuits and Systems, 2005.
[52] "Announcing the ADVANCED ENCRYPTION STANDARD (AES)". Federal Information Processing Standards Publication 197. United States National Institute of Standards and Technology (NIST). November 26, 2001.
[53] Hoffman, P. "Cryptographic Suites for IPsec". IETF. RFC 4308. December 2005.
[54] Fruhwirth, Clemens "LUKS On-Disk Format Specification Version 1.1.1". December 8, 2008.
[55] "Security Overview OpenVPN cryptographic layer" OpenVPN, retrieved 19 May 2014. http://openvpn.net/index.php/open-source/documentation/security-overview.html
[56] "The Libgcrypt Library" g10 Code GmbH and the Free Software Foundation version 1.6.0, 13 December 2013 (accessed 19 may 2014): https://www.gnupg.org/documentation/manuals/gcrypt/
[57] Xilinx Power Estimator (XPE) 7 Series and Zynq®-7000 2014.1 http://www.xilinx.com/products/design_tools/logic_design/xpe.htm
[58] M. Giannikos et.al, "Towards Secure and Context-Aware Information Lookup for the Internet of Things", Proceedings of the International Conference on Computing, Networking and Communications, 2013
[59] Hans Schaffers, et.al, Smart Cities as Innovation Ecosystems Sustained by the Future Internet, www.fireball4smartcities.eu, 2012.
[60] IDC EMEA, "The European Network and Information Security Market Scenario, Trends and Challenges", A study for the European Commission, DG Information Society and Media, 2009.
[61] Helena Rifa-Pous and Jordi Herrera-Joancomartí. Computational and Energy Costs of Cryptographic Algorithms on Handheld Devices, Future Internet 2011, 3, 31-48; doi: 10.3390/fi3010031.
[62] Manifavas, C. et. al. Lightweight Cryptography for Embedded Systems – A Comparative Analysis.
[63] Kocher, P. et. al. Security as a New Dimension in Embedded System Design.
[64] Ellsbury, Graham (1988), "2. Description of the Bombe", *The Turing Bombe: What it was and how it worked*. http://www.ellsbury.com/bombe2.htm. Retrieved 14 Jan 2016.
[65] Akdemir, Kahraman, et. al. "Breakthrough AES Performance with Intel AES New Instructions" (Intel White Paper), https://software.intel.com/sites/default/files/m/d/4/1/d/8/10TB24_Breakthrough_AES_Performance_with_Intel_AES_New_Instructions.final.secure.pdf
[66] Christian Kison, Jurgen Frinken and Christof Paar. "Finding the AES Bits in the Haystack: Reverse Engineering and SCA Using Voltage Contrast". In Tim Güneysu and Helena Handschuh, editors, Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES), volume 9293 of LNCS. Springer, 2015
[67] J. Longo, E. De Mulder, D. Page and M. Tunstall. "SoC It to EM: ElectroMagnetic Side-Channel Attacks on a Complex System-on-Chip". In Tim Güneysu and Helena Handschuh, editors, Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES), volume 9293 of LNCS. Springer, 2015
[68] Amir Moradi and Alexander Wild, "Assessment of Hiding the Higher-Order Leakages in Hardware What Are the Achievements Versus Overheads?." In Tim Güneysu and Helena Handschuh, editors, Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES), volume 9293 of LNCS. Springer, 2015
[69] Tanja Lange, Christine van Vredendaal, and Marnix Wakker, "Kangaroos in Side-Channel Attacks". Netherlands Organisation for Scientific Research (NWO) January 29, 2015
[70] N. Homma, Y. Hayashi, N. Miura, D. Fujimoto, M. Nagata, and T. Aoki, "Design Methodology and Validity Verification for a Reactive Countermeasure Against EM Attacks". Journal of Cryptology. Volume 30 Number 2, April 2017
[71] Qinghan Xiao, Thomas Gibbons and Hervé Lebrun, "RFID Technology, Security Vulnerabilities, and Countermeasures". Supply Chain, The Way to Flat Organization, 2008.
[72] Crockett, L. Elliot, R. Enderwitz, M. Stewart, R. The Zynq Book, University of Strathclyde, Glasgow, Scotland, 2014.