A Hardware Architecture for Integrated-Security Services

Fábio Dacêncio Pereira[1] and Edward David Moreno Ordonez[1,2]

[1] University of São Paulo-USP, Polytechnic School, São Paulo, SP, Brazil
[2] State University of Amazonas-UEA, Manaus, AM, Brazil
fabio.dacencio@poli.usp.br, edwdavid@gmail.com

Abstract. There are numerous techniques, methods and tools to promote the security of a digital system, however, each day the mechanisms of attack evolve and are integrated, creating separate spheres of combined attacks. In this context, this paper presents an embedded security system (into a SoC system) that has as main goal an integration of the security services. It reaches an improved performance and prevents malicious attacks on systems and networks. The SoC prioritizes the implementation of dedicated functions in hardware as cryptographic algorithms, communication interfaces, among others. In our prototype, initially, the flow control functions and settings are running in software. This article shows the architecture, functionality and performance of the system developed, and we discuss a real implementation in FPGA.

Keywords: Embedded System, Integrated Security Services, FPGA Performance and System-on-Chip.

1 Introduction

Researches on security systems have focused typically on creating new services or improving the performance and reliability of a single technique, algorithm or mechanism. The approaches that aim some integration among several security mechanisms have focused usually on a specific strategic application, since the systematic approach to integrate security systems requires an analysis of relations among the data.

In this context, we like to intend to describe not only an integration model for security services, but also provide mechanisms to implement it in hardware and software, and we will focus on requirements as transparency, performance and productivity.

The research on security systems integration (SSI) may be noted since 70'. The concern in become a security technical set in a single integrated system is of paramount importance, since the possible fragility of a security service may be offset by other.

Commercially, network devices and software of security have the challenge to create this integration. The greatest difficulty is for establishing a common strategy for innumerable devices, tools and techniques of information security.

The study of this integration is not new. The first research about this integration appeared in 1975, and it was proposed by David Elliott Bell and Len LaPadula and it is known as the Bell-LaPadula model [1]. This model was intended to create classes

M.L. Gavrilova et al. (Eds.): Trans. on Comput. Sci. IV, LNCS 5430, pp. 215–229, 2009.

of access control for the DoD's systems. After this pioneer model, many techniques to integrate security systems were developed [1-3].

The current models of SSI determine the relationship between a specific set of security mechanisms which exchange information for preventing or treat system's anomaly.

The current models propose solutions under a specific set of services disregarding the existence of other [4-8]. The most of those models has been used UML techniques to represent the features, functionality and methodology.

So, this paper proposes a special architecture and describes the functionalities of an Embedded System of SSI. It was implemented in FPGA of the Virtex kind. The different modules dedicated to security as AES, RSA, HASH, among others, were implemented. It is important note that the performance statistics (runtime related to circuit delays) and physical area of implementation in hardware are presented and discussed.

This paper is organized in 10 sections. The description, methodology and objectives of embedded system are in section 3, whereas section 2 is dedicated to related works. In section 4 has a better description of our proposed system, while section 5 emphasizes at libraries dedicated to security services. The section 6 shows the description of the services layer, and Soc Configuration Interface appears in section 7. Some results of implementation are provided in section 8 and section 9 ends with some conclusions.

2 Related Works

The integration of security services is a necessity since the attacks techniques are using the integration to increase the strength and efficiency of the attacks. Some works have solutions to integrate a specific set for security services set and they are bringing protection to the system.

The model proposed by Nimal Nissanke [8] focused on the protection of three pillars: secret information, user identification and access control mechanisms. The current models have mainly focused on services integration of access control and intrusion detection, they created a prevention system and dynamic protection.

Kim, in your work [25], affirms that Conventional security systems provide the functions like intrusion detection, intrusion prevention and VPN individually, leading to management inconvenience and high cost. To solve these problems, attention has been paid on the integrated security engine integrating and providing intrusion detection, intrusion prevention and VPN. This work introduces a security framework that allows secure networking by mounting integrated security engine to the network nodes like router or switch. The solution was tested and validated in Cisco Router 2620 and 3620. The results were not discussed in paper.

The main concept discussed by Zilys [5] in your paper involves representation through objects graphs that represent security services. From this concept may be a link among security events. The concept, which is formulated, enables strategic control of integrated security systems (ISS) considering from the influence-reaction parameter point. Reaction strategy algorithm selection allows minimizing reaction time to danger influence and maximizing efficiency of security system. The author does not explore the different possibilities of constructing security objects. The paper presents only a basic form that could be further explored.

Jonsson [7] proposed a model defines security and dependability characteristics in terms of a system's interaction with its environment via the system boundaries and attempts to clarify the relation between malicious environmental influence, e.g. attacks, and the service delivered by the system. The model is intended to help reasoning about security and dependability and to provide an overall means for finding and applying fundamental defense mechanisms. Since the model is high-level and conceptual it must be interpreted into each specific sub-area of security/dependability to be practically useful.

The model propose by Jonsson is a suggested an integrated conceptual model of security and dependability. The model is aimed at improving the understanding of the basic concepts and their interrelation.

The model categorizes system attributes into input, internal and output attributes, whether from the traditional security domain or the traditional dependability domain. It should be helpful for reasoning about security, so that effective defence methods can be developed and tangible results with respect to security/dependability performance can result. Furthermore, the model is intended for use for the development of security metrics. Did not find this model applied in a case study specific.

One more recent proposal of Hassan Rasheed [18] proved the integration of services as, intrusion detection, access control and the author specifies how to respond to intrusion detection. Along this same line other researchers have proposed similar works. Security devices, as the CISCO ASA family also demonstrated the integration of these services.

The differential of our proposed system with those related works is that our approach tries to explore a universe of greater security services and the union of features and functionality in an integration layer.

The main contribution of this work when compared with other listed earlier is that in addition to presenting a model different are also described mechanisms and an architecture capable of linking various security services into a single structure.

3 Embedded Security Services

This section shows descriptions and our methodology and main objectives related to our embedded system which should contain security services.

The embedded system of SSI proposed can be divided into cores in hardware and software libraries that collaborate among themselves for supporting the specifications of an integrated security system.

The proposed system use security services as asymmetric and symmetric encryption, hash functions, IDS, firewall and audit aspects. These services can do tasks for cooperating among them through a Security Services Integrated Layer (SSIL).

The main function of the SSIL is unifying the security services so that an application would have access to them through a common database and a specific set of methods. The advantages can be summarized mainly by transparency access to security data, productivity, and robustness and achieved performance.

The technology and methodology adopted in developing this project have direct impact on the viability, power consumption, area, complexity, flexibility and other factors related to the final application. In this context, we have studied three models: dedicated hardware, embedded software and a hybrid model:

- Dedicated Hardware: Initially was adopted that all modules and functionality would be described and implemented in hardware. The modules for processing and control should be able to run operations with a high level of abstraction and complexity. This proposal was discarded because it would lead to creation of hardware with a high power and area consumption. Besides it is being unwieldy and difficult to do updates.
- Embedded Software: This proposal adds flexibility to the system for allowing the creation of specific security libraries and implementation of robust operations. The use of embedded software allows the creation of interface functions as USB, RS232, Ethernet, SVGA, graphics displays, among others. The disadvantages of this model are in low performance in complex functions implementation as encryption and data compression when compared with hardware solutions.
- Hybrid Architecture: This model supports the integration into a single system, hardware and embedded software. So, by using an interface for communication could be made that hardware and software have interactions, extracting the benefits offered by both.

To classify the services that would be implemented in hardware and software we have adopted the following methodology. In first version, the system was described in embedded software and it is executed by the PowerPC which is embedded in a FPGA Virtex.

After, we have created some libraries with the function of time calculation and instructions counting (timer.h and InstC.h). These were used to detect code sentence and instructions sets that consume high processing time.

Based on these results optimizations were performed at the level of software, but the main contribution was the competence for detecting the system functions that could be implemented in hardware. The result of this technique can be seen in section 8 on the final performance of our system. In the sequence we present the organization in layers of our embedded system.

The final architecture is composed of dedicated cores and embedded software that implements the system's control. The software runs under the Linux operating system version 2.6.0, and this distribution was dedicated and compiled for the PowerPC processor 405.

Using a Linux distribution allows the embedded software makes use of the advantages of an OS as described in [14]: scheduling process, semaphores, file system, memory access control, among others. These characteristics make this architecture a good choice for our project. Fig. 1 describes the organization of our embedded system.

As can be seen in Fig. 1, the system's features are divided into two categories: (i) high performance functions which are implemented in dedicated hardware (cores) and (ii) control functions, data structures and flow process, that are implemented in embedded software. In this case the C language was adopted as the best performance to native applications.

The PowerPC physically present into the FPGA Virtex [15] is responsible for running the embedded software including the security functions, flow control and operating system installed, which were also implemented in this project.

At the highest level of abstraction is the Security Services Integrated Layer (SSIL). The main function of this layer is ignore the existence of different security services, and spread the use of the system's features.

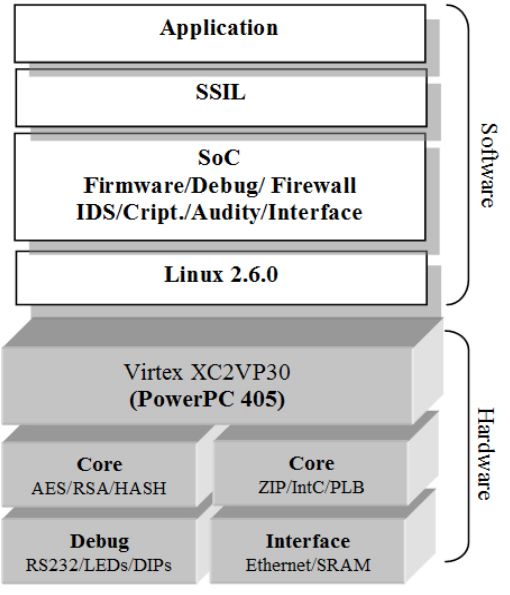

Fig. 1. Organization of embedded system in SoC

4 Top-Level Architecture

The project's architecture is composed of dedicated cores that communicate among them with the PowerPC processor through controlled buses (PLB). Fig. 2 shows how the embedded system proposed is organized, highlighting the main cores and software stored in memory of 512 Mbytes RAM.

We have projected a preliminary version of this system by using the XUPV2P platform which consists of a FPGA Virtex 2 Pro (XCV2P30) and peripherals of Input and Output, where there are the interfaces used as Ethernet, memory SRAM, RS232, Leds and Switches. In the sequence has the description of the cores use.

Xilinx PLB EMAC 10/100
This cores defines the Ethernet protocol layers, where can set a MAC address, allowing the communication through a conventional network. Thus, the embedded security system presents as a device to network.

Xilinx MCMP
To control the access was necessary to include the module MCMP (Multi-Controle Memory Port) which is the interface between the SRAM memory of 512 Mbytes and the PowerPC processor.

Xilinx IntC
IntC (Xilinx Interrupt Control) is responsible by interruptions control allowing different modules can interrupt the flow of the main program (PowerPC Processor), for

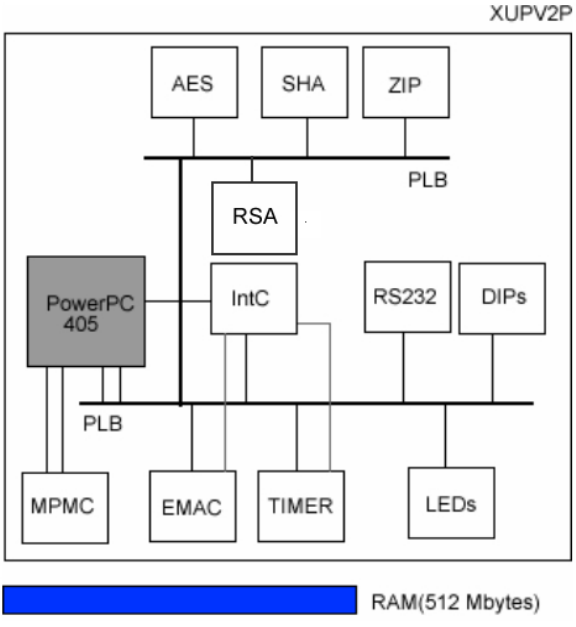

Fig. 2. Top-level architecture

treatment of an event. Currently the timer module and EMAC generate interruptions that are treated as dedicated routines for PowerPC.

Xilinx Timer
Timer used by cores and PowerPC to create and define time scales.

Xilinx PLB
The integration of the PowerPC with dedicated cores is achieved through a common bus, called the PLB (Processor Local Bus). To control the access to bus there is an access arbiter, avoiding conflicts and information collisions.

Debug Interface
Primitive interfaces of the debugging and configuration are used only at the development stage, validation and prototyping processes.

- 4 DIP Swtiches
- 4 Leds
- 1 RS232 serial interface (integration with the PC): transfer rate 9,600 bps, without parity control.

The system debugging is important, especially at the development stage and validation to reduce the design time and minimize the occurrence of errors not foreseen during the project specification.

Symmetric Encryption
Dedicated component that has exclusive function of cipher and decipher the information by using symmetric encryption algorithm, AES (Advanced Encryption Standard). In our project, the AES module can be configured to blocks from 128, 192 and 256 bits. It is important to note that the operation modes, the generation of cryptographic keys and padding control are implemented by software.

Asymmetric Encryption
The module of asymmetric cryptography uses the RSA algorithm that is prepared to encrypt information using keys of 512 and 1024 bits. Later will be implemented a version of 2048 bits. Again the control of vector of clear text and encrypted, keys generation and padding control are implemented in embedded software.

Hash Functions
The hash function initially implemented in system uses the MD5 and SHA-2 algorithms. Even knowing the fragility of the MD5 algorithm compared to the number of collisions detected [19], many applications have adopted. So was implemented and inserted in this project. The flow control is done by software.

Binary Compression
The compression algorithms are used in system as a complement of the embedded application. We adopted the algorithms Shannon-Fano and Huffman [12]. They have a good compression rate to generic applications. There are some algorithms for specific applications as audio and video that can achieve higher compression rates for these specific cases.

In the proposed architecture are implemented in hardware the specific modules for security, compression and the communication interface. The control flow of the data capture to the hard core drive was implemented in software and runned by the PowerPC embedded, this feature adds greater flexibility for security system. The PLB Bus facilitates the addition of new security services. Thus the system can be adapted and updated for different needs.

5 Dedicated Security Libraries

The libraries in software are of critical importance to the viability of the project, since they make use of cores of interface and security for running the functions that require performance. Next, we present a brief feature's description of the main libraries created:

- crypto.h: invokes the functions and parameters for the encryption algorithms based initially in AES and RSA, where can generate key, Subkey, set the mode of operation, encrypt, decipher, among others.
- firewall.h: functions and parameters to insert, delete, alter, and select information in rules table.
- ids.h: allows operations to insert, delete, alter, select anomaly patterns in specific table.
- hash.h: implements operations related to hash functions MD5, SHA-1 and SHA-2.

- audit.h: functions of creation of logs and failures debugging.
- zip.h: data compression (huffman and shannon fano)
- hashtable.h: allocates structure in memory and implements manipulation functions and selection. This is used by other libraries to create customized tables.
- xml.h: contains a parser to that the information generated by the security system can be accessed by other systems through the XML format.
- ciss.h: describes functions for creating of the SSIL, described in the next section.
- http.h: implements the HTTP protocol for access to Soc Configuration Interface.
- server.h: primitive of a web server for access to Soc Configuration Interface.

These libraries are used by embedded application and SSIL to create an environment of personal security.

6 Security Services Integration Layer (SSIL)

The Security Services Integration Layer (SSIL) describes a mechanism to store information on security system, in pursuit of different devices, techniques and tools share relevant information.

The SSIL will attend a universe of devices, tools and algorithms that exist and that still will be created. This is possible only if have a dynamic structure that allows customize their characteristics, maintaining the information integrity.

All information relating to system security or network that can be considered vital can be stored locally for each security service, and this can generate an event and register in SSIL frame. The storage structure (frame) is shown in Fig. 3 and is described in the sequence:

- Package ID: identifies a single package; field with auto-increment.
- Application ID (2 bytes): This field stores the application ID that generated the notification.
- Service Type (1 byte): classifies the security service type that detected the anomaly.
- Anomaly Level: quantifies and specifies the depth of anomaly.
- Reference ID: reporting reference number (more details on anomaly identified).
- Register Number (1 byte): sequential number indicating the number of notifications made by a particular application on an anomaly identified.
- IP / MAC (10 bytes): register the IP number and MAC address of the creator of notification
- Permission (4 bits): Set the access level
- Remote access (4 bits): allows or not remote access to notification.
- Data (100 bytes): information on notification.

All and any anomaly of system identified by some of security services must be formatted according as structure shown in Fig. 3. This structure contains sufficient information to a security policy adopted a decision consistent in regarding indications of possibles anomalys.

The Decision-making after the identification of an anomaly is the responsibility of the application that uses the SSIL. For the initial tests was adopted a simplified model of integrated services based on the security structure published in 2007 by NIS-SANKE [8].

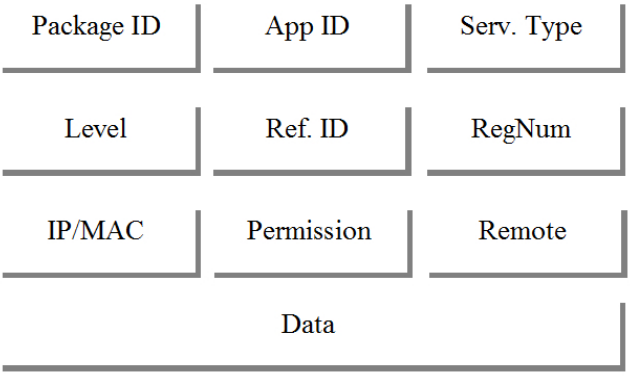

Fig. 3. SSIL Frame

This structure aims to prioritize the network access control. That is, the sum of deficiencies identified by the system can lead to blockage of access. Thus, devices of a network that make anomalies will be blocked.
The system will be able to deny access to network without necessarily knowing where or what security services identified the anomaly.

Specific information about any anomaly may be necessary for the decision. This information can be accessed through the Reference ID that has the function to point the index or code to locate the anomaly generated by a specific security service.

Our security model is simplified and implements the decision based on IP field and anomaly level. The level is defined with the values of 1 to 5 for gravity anomaly identified. In this case, we decided that an accumulation of 12 points on the same network device leads a blockage of service. The appropriate decision about these values deserves more attention and research, but it is not the focus on this phase of our project. Thus, regardless of who generated the notifications, the device network will be blocked.

It is important to note that more sophisticated and specific rules can be created easily if supported the structure (frame) of SSIL, as shown in Fig. 3.

7 SoC Configuration Interface

The SoC Configuration Interface is important to system, because through a friendly interface the user can configure the main modules of the system as the dedicated cores and routines of the overall functioning of the system.

The Interface, implemented in HTML + Javascript, allows the access by a conventional network computers, only the IP mapping pre-defined (198.162.0.l: 8008) in a browser. Fig. 4 shows the Soc Configuration Interface version 1.3. The Interface is not the focus of this article and will not be explored in detail.

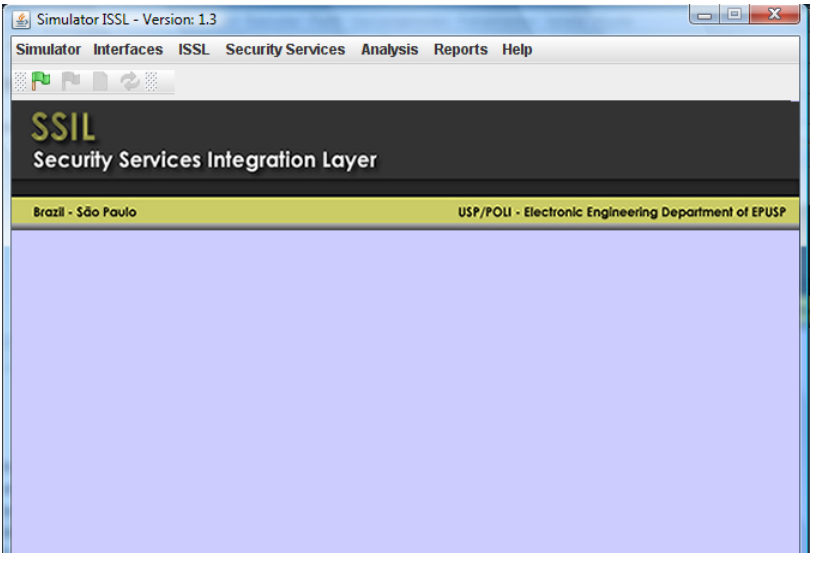

Fig. 4. SSIL Configuration Interface

8 Statistics of Occupation and Performance

The performance and occupation are important factors to evaluate the proposed system. The performance of the embedded system varies depending of the application implemented. In this section, we describe the fully implemented version in embedded software and the hybrid version.

Initially, we present results of the fully software implementation for highlighting the critical performance points (see table 1).

The times recorded were made after the execution of each service on a load of 1MB storing information in memory.

For a detailed review of performance results, in software, described in table 3, it is evident the need to dedicated cores in hardware in pursuit of optimizing the overall processing system.

In this context, the table 2 shows the modules implemented in hardware (VHDL description). The impact of this implementation can be viewed in the following sections.

Table 1. Runtime of some security services in software

Services	Runtime(ms)
AES	753
RSA	8582
SHA-1	638
SHA-2	787
MD5	523
Huffman	1014
Shannon-Fano	938

In this analysis was considered the total time of propagation of the system including all modules described. Information on occupation were evaluated by the number of slices and the total percentage in relation to selected FPGA (see table 2).

8.1 Analysis of Spatial Resources

The table 2 presents the results generated after synthesis and routing of the cores of the system, We show the numbers of SLICES consumed by each module and of the all system.

As can be seen between the security cores, the RSA algorithm consumed the largest area (already expected) given its complexity and difficulty of implementation in hardware [17].

Table 2. Occupancy Statistics in FPGA

Core	Slices	Occupation
AES	810	3%
RSA1024	3929	13%
SHA-2	1974	6%
MD5	1225	4%
Shanon	2407	8%
Debug Interface	10765	35%
Total	25391	84%

The total area consumed can not be considered the sum of areas of each core since the statistical data presented represent the occupation of each module specific (as data generated by the synthesis process) and does not consider the logic consumed, for example the integration / interconnection of them.

The total occupancy was 84% of the FPGA. Unable to enter simultaneously some modules as RSA 512 and 1024 bits, well as the data compression algorithms, Shannon-Fano and Huffman.

Thus, at this stage of the project, the statistical data of total occupation FPGA are only made in 1024-bit RSA algorithms and Shannon-Fano.

8.2 Analysis of Propagation Time

The performance analysis supports mainly in propoagation time of each module and the system propagation time. Table 3 presents statistical data performance. The time unit is the nanoseconds (ns).

When looking table 3 note that the propagation time of the complete system is approximately 8.4 ns, this indicates that the external clock can be a maximum value of 119 MHz.

Considering the device and system complexity the maximum frequency is satisfactory. The propagation time of each module dedicated shows again that the RSA has the largest delay. In this case, the RSA algorithm 1024 and the Shannon-Fano was implemented in this full version of the system.

Table 3. Performance Statistics

IP Core	Propagation Time	Frequency (Max.)
AES	8,8495 ns	113 MHz
RSA1024	9,3458 ns	47 MHz
SHA-2	6,3694 ns	157 MHz
MD5	7,4627 ns	134 MHz
Shanon	8,2645 ns	121 MHz
Debug Interface	8,6206 ns	116 MHz
Total	8,4033 ns	119 MHz

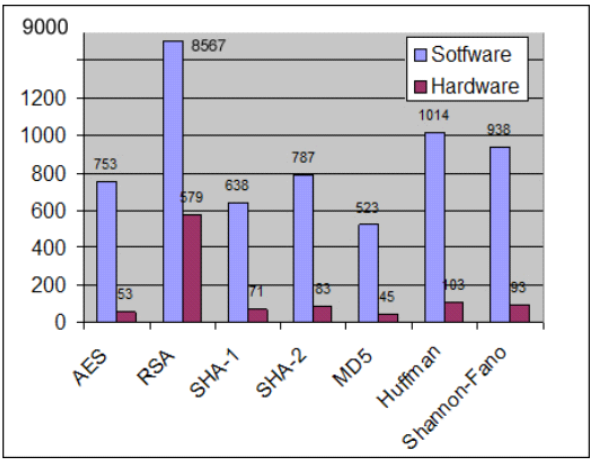

Fig. 5. Impact of the dedicates cores

The frequency for the PowerPC is 300MHz, so the performance implementation in software is associated with this frequency. It is important to note that the Soc Configuration Interface is described in software consuming part of DDR memory SRAM (external) and processing time of the PowerPC.

8.3 The Impact of Adding Dedicated Cores

The performance difference between the version fully in software and the hybrid version can be measured in Fig. 5.

As can be noticed the hybrid version promotes a more effective and efficient processing. The average optimization was of 12.3 times in performance of the proposed system.

8.4 Comparison with Other Dedicated Cores

There are many cryptographic cores and compression of data. The table 4 presents data from implementations works related. In a comparison would be ideal important that all cores were implemented the same device and synthesized by the same software. Even with these differences is possible compare the proposed project, based on the number of Slices and Frequency Acquired.

Table 4. Comparison with related works

Algorithm	Version	FPGA Device	Occupation	Frequency
AES	SSIL	VC2VP30	810 Slices	113 MHz
	[22]	XC2V4000	753 Slices	118 MHz
RSA	SSIL	VC2VP30	3929 Slices	47 MHz
	[20]	XC40250XV	2902 Slices	31 MHz
SHA-2	SSIL	VC2VP30	1974 Slices	157 MHz
	[23]	VC2VP30	1667 Slices	141 Mhz
MD5	SSIL	VC2VP30	1225 Slices	134 MHz
	[21]	XC2V4000	1057 Slices	78 MHz
Huffman	SSIL	VC2VP30	2407 Slices	121 MHz
	[24]	EP20k100E	4160 LEs	173 MHz

In reviewing the information you can see that the system developed a good track record regarding occupation and performance. The focus of this project is not conceive of color to compete with the best implementations, but providing an environment for testing of integrated services and security.

The biggest difference was found on the RSA algorithm. This difference may be explained by different techniques adopted in implementation. The version implemented by Mazzeo [20] is based on the Montgomery algorithm, the version already implemented this project uses conventional multiplier. Most of the slices are consumed for the implementation of large multiplier justifying the difference more pronounced.

9 Conclusions

This work presented a specialized embedded System of Integrated Security Services. We propose and discuss the architecture, features and performance when it is projected in a FPGA Virtex.

The main objectives of the project have been achieved. The security services were created and we have a prototype of the embedded platform which offers integration among different security services. Our embedded system offers transparent and user-friendly provided to the user, productivity and greater involvement of the security mechanisms of the present system.

The robustness of the system generated a concern about the performance that it could achieve. As the performance data and area may be noted that the system has reached a good performance level.

The performance of dedicated security cores can be compared to the best published implementations of algorithms AES [9], SHA [11], MD5 [10]. A new systematic search is being conducted to find better solutions to implementation of algorithms compression and RSA in hardware [13].

In the near future, it would be interesting work on the following issues: The proposed system can be expanded with the addition of new modules for security, as new encryption algorithms.

It is evident that the insertion of new services directly to performance and involve mainly in the area occupied. As future work aims to study the possibility reconfigure

the device to facilitate the customization for specific applications and verify the impact of reconfiguration in security applications.

The system was developed on the platform XUPV2P by the availability, but it would be very important test in the most robust platforms such as a FPGA Virtex 5.

Finally, it would be very important to create a Soc Configuration Interface with greater resources and assessment.

References

1. Bell, D.E., Lapadula L.: Secure Computer System: Unified Exposition and Multics Interpretation. Technical Report MTR-2997 Rev. 1, MITRE Corporation, Bedford, MA (1975)
2. Baker, M.P.: Integrated security system. In: Proceedings International Carnahan Conference on Security Technology (1989)
3. Okamoto, E.: Proposal for integrated security systems. In: Proceedings of the Second International Conference on Systems Integration ICSI 1992 (1992)
4. Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Transactions on Information and System Security 4(3), 224–274 (2001)
5. Zilys, M., Valinevicius, A., Eidukas, D.: Optimizing strategic control of integrated security systems. In: 26th International Conference on Information Technology Interfaces (2004)
6. Ghindici, D., Grimaud, G., Simplot-Ryl, I., Liu, Y., Traore, I.: Integrated Security Verification and Validation: Case Study. In: IEEE Conference on Local Computer Networks (2006)
7. Jonsson, E.: Towards an integrated conceptual model of security and dependability, Availability, Reliability and Security. In: ARES 2006 (2006)
8. Nissanke, N.: An Integrated Security Model for Component–Based Systems. In: IEEE Conference Emerging Technologies & Factory Automation, ETFA (2007)
9. Zambreno, J., Nguyen, D., Choudhary, A.: Exploring Area/Delay Tradeoffs in an AES FPGA Implementation, Department of Electrical and Computer Engineering Northwestern University (2004)
10. Deepakumara, J., Heys, H.M., Venkatesan, R.: FPGA implementation of MD5 hash algorithm. In: Emerging VLSI Technologies and Architectures. IEEE Computer Society, Los Alamitos (2006)
11. McEvoy, R.P., Crowe, F.M., Murphy, C.C., Marnane, W.P.: Optimisation of the SHA-2 family of hash functions on FPGAs. In: Emerging VLSI Technologies and Architectures. IEEE Computer Society, Los Alamitos (2006)
12. Brian Connell, J.: A huffman-shannon-fano code. In: Proceedings of the IEEE, pp. 1046–1047 (July 1973)
13. Michalski, A., Buell, D.: A Scalable Architecture for RSA Cryptography on Large FPGAs, Field-Programmable Custom Computing Machines. In: FCCM 14th Annual IEEE Symposium (2006)
14. Monta Vista Embedded Linux Software, `http://www.mvista.com/`
15. Xilinx Document, PowerPC 405 Processor Block Reference Guide, Embedded Development Kit, document: ug018 (2008)
16. den Boer, B., Bosselaers, A.: Collisions for the compression function of MD-5. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 293–304. Springer, Heidelberg (1994)

17. Fry, J., Langhammer, M.: RSA & Public Key Cryptography in FPGAs, Altera document (2005)
18. Rasheed, H., Randy, Y.C., Chow: An Information Model for Security Integration. In: 11th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS 2007) (2007)
19. Sasaki, Y., Wang, L., Ohta, K., Kunihiro, N.: Security of MD5 challenge and response: Extension of APOP password recovery attack. In: Malkin, T.G. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 1–18. Springer, Heidelberg (2008)
20. Mazzeo, A., Romano, L., Saggese, G.P.: FPGA-based Implementation of a serial RSA processor. In: Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE) (2003)
21. Järvinen, K., Tommiska, M., Skyttä, J.: Hardware Implementation Analysis of the MD5 Hash Algorithm. In: Proceedings of the 38th Hawaii International Conference on System Sciences (2005)
22. Zambreno, J., Nguyen, D., Choudhary, A.: Exploring area/Delay tradeoffs in an AES FPGA implementation. In: Becker, J., Platzner, M., Vernalde, S. (eds.) FPL 2004. LNCS, vol. 3203, pp. 575–585. Springer, Heidelberg (2004)
23. Chaves, R., Kuzmanov, G., Sousa, L., Vassiliadis, S.: Improving SHA-2 Hardware Implementations. LNCS. Springer, Heidelberg (2006)
24. Zeng, G., Ito, H.: Efficient Test Data Decompression for System-on-a-Chip Using an Embedded FPGA Core. In: Proceedings of the 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT) (2003)
25. Kim, J.: Design and implementation of integrated security engine for secure networking. In: IEEE Advanced Communication Technology (2004)