Computers and Electrical Engineering 98 (2022) 107707

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Embedded firewall for on-chip bus transactions☆,☆☆

Jesús Lázaro *, Unai Bidarte, Leire Muguira, Armando Astarloa, Jaime Jiménez

Department of Electronics Technology, University of the Basque Country, Bilbao, Spain

ARTICLE INFO

Keywords:
Communication system security
Data buses
Data security
Field programmable gate arrays

ABSTRACT

This article presents a novel approach towards System-on-Chip (SoC) security. Although communications security and operating system hardening have been studied, new application opportunities and menaces have appeared with the incorporation of Multiprocessor-System-on-Chip (MPSoC) into the Internet of Things (IoT). Reliable implementation environments have become necessary, so novel security architectures and solutions have been introduced to protect the vulnerable data, which could be used by plenty of these applications.

We propose an Advanced eXtensible Interface (AXI) transaction firewall, which, by checking the type of operation, the physical address, and the bandwidth according to a set of rules, rejects untrusted requests between cores. Results have been performed on a Zynq platform, and obtained results show that the proposed AXI-firewall can prevent unauthorized transactions consuming few hardware resources. Besides, the fully combinational nature of the firewall's AXI to AXI path entails that the firewall does not affect the overall performance of the system.

1. Introduction

MPSoCs and related on-chip networking architectures for communicating SoC elements have been extensively investigated in the past [1], while security is seldom mentioned. The accuracy and security of the hardware, especially the processors, need to be enhanced because they suffer from attacks, and hardware mending is rigid.

In embedded applications, failing to guarantee security involves economic consequences, raising attention for trusted computing solutions. Security criteria are user authentication, storage and communications security, and inputs/outputs security [2]. A Central Processing Unit (CPU) can access physical resources in MPSoC [3], allowing illegitimate processes executing in one or more CPUs to generate malicious requests. Sensitive information can be extracted, operations of MPSoC can be disabled, or system behavior modified due to attacks at MPSoCs [4]. It explains the constant increase of interest in security considerations in embedded systems [5]. Safety mechanisms are demanded to avoid the insertion of malicious data or orders into the system. The design of SoC maintaining expense levels with incorporated security features and cost limitations remains a challenge to overcome. Furthermore, this security must be maintained throughout all the life-cycle of the embedded system [6].

Due to the impact on the system security, the design of MPSoCs involves contemplating stringent constraints of real-time, which must always be guaranteed, and security requirements [7]. However, the implementation of a mechanism of tight real-time may affect the security of MPSoCs. Security must be considered a design parameter, and balancing performance with hard real-time

☆ This work has been supported by the Ministerio de Economía y Competitividad of Spain within the project TEC2017-84011-R and FEDER, Spain funds as well as by the Department of Education of the Basque Government within the fund for research groups of the Basque university system, Spain IT978-16.

☆☆ This paper was submitted for regular issues of CAEE, but is to be included in for special section VSI-fpga3. Reviews were processed by Area Editor Dr. E. Cabal-Yepez and recommended for publication.

* Corresponding author.
E-mail address: jesus.lazaro@ehu.eus (J. Lázaro).

https://doi.org/10.1016/j.compeleceng.2022.107707
Received 23 September 2021; Received in revised form 14 December 2021; Accepted 6 January 2022
Available online 1 February 2022

Acronyms

AXI	Advanced eXtensible Interface
BRAM	Block RAM
CAN	Controller Area Network
CPU	Central Processing Unit
DoS	Denial of Service
FF	Flip-Flop
FPGA	Field-Programmable Gate Array
GPIO	General-Purpose Input/Output
IoT	Internet of Things
IP	Intellectual Property
LED	Light Emitting Diode
LUT	Look-Up Table
MPSoC	Multiprocessor-System-on-Chip
NoC	Network on Chip
PS	Processing System
SoC	System-on-Chip

and security should be addressed [8]. The performance of MPSoCs is usually increased by dividing the applications into tasks and disseminating them among the computing Intellectual Property (IP) cores [4]. Nevertheless, it involves exchanging sensitive data, and IP cores can be exploited to attack the system.

The latest attacks use transient execution and microarchitectural weaknesses, demonstrating that the SoC, which includes the processor, cannot be considered the trustworthy computing pillar, in opposition to the theoretical model [9]. Processors are susceptible to complex attacks because the upper software layer ignores the management of the program's data at the microarchitectural side. These ingenious and effective attacks can be categorized into microarchitectural side-channel attacks and transient-execution attacks [10]. Side-channel attacks are not contemplated in the reliable execution environments presented by the industry [11], such as AMD SEV [12], Intel SGX [13], or ARM TrustZone [12]. So, although the enhancement of security, reliable implementation environments continue being vulnerable to threats.

The remaining paper is organized as follows. In Section 2, we overview related work on hardware security. Section 3 describes the functionality and components of the proposed hardware firewall. Section 4 defines different use cases, single and multimaster environments, to examine the effect of the developed IP core. In Section 5, the obtained results regarding the area and time resources for a Zynq device are discussed. In Section 6, we compare the performance of our security approach against other hardware-based firewall solutions. Finally, Section 7 summarizes this paper and outlines future work.

2. Related work

The augmented complexity of the processors has been faced partitioning the design of a processor in two abstraction layers: the architecture and the microarchitecture [10]. The first one is a high-level abstraction, which reveals only the interfaces and attributes to the software, hiding the hardware specifics. The second one takes care of containing the hardware elements unnoticeable from the software.

Security solutions are commonly based on implementing diverse modules, such as a firewall, intrusion detection or prevention systems, or cryptographic function accelerators [14,15]. In [2], an overview of the main solutions regarding internal transaction protection is proposed. Memory protection approaches, bus, and Network on Chip (NoC) based security methods, and other methods are detailed. Naturally, performance and security – authentication, availability, and access control, among others – are significant design parameters. Nevertheless, as mentioned before, making the best use of security generally compromises performance and usability. The confidentiality of IP core communication was addressed in preceding works by cryptographic techniques [4]. Furthermore, firewalls can be implemented with three security levels, depending on the previously listed provided security features. Lower-level checking schemes are included in higher-level firewalls [16].

Hardware blocks or software functions can be deployed in embedded systems against attacks. Solutions implemented purely in hardware are typically faster concerning latency than a software solution [2]. Moreover, mechanisms implemented in hardware are more difficultly compromised than software ones. Microarchitectural side-channel attacks are predominantly implemented in software; however, hardware-based ones guarantee improved performance. On the other hand, transient execution attacks are mainly deployed in software or based on microcode updates. Nevertheless, hardware-based solutions also have been suggested [10].

Not only does security concern software design, but hardware must also take into account cyber-attacks. For instance, read/write AXI cycles identify the destination address but not the origin, so that a malicious IP could access sensitive memory blocks or peripherals. Authors of [2] exploit distributed hardware firewalls that support confidentiality, integrity, memory partitioning, traffic

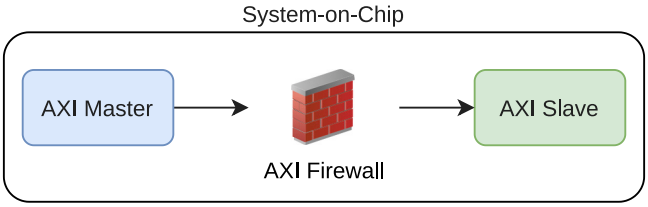

Fig. 1. The AXI Firewall stands between an AXI Master (processor, for example) and an AXI slave (AXI interconnect, for example). It blocks transactions according to several rules: address range, bandwidth limit,....

monitoring, and access control (R/W) through a set of parameters defining a security policy. However, they provide a latency of 2 clock cycles. On the other hand, [17] describes a hardware/software solution based on configurable access rights; its latency is three clock cycles. Hardware firewalls are more common in NoCs; for instance, [18] checks physical addresses in 4 clock cycles. In the NoC of [1], a firewall manages the access rights by means of a lookup table while processing the incoming packet — i.e., latency is sourced by frame computations. Tan et al. [19] create memory access mechanisms that allow safe use of shared IP with direct memory access, as well as shared libraries. They also present a prototype Isolation Unit that checks memory transactions and allows for dynamic configuration of permissions. In [20], Kornaros et al. proposed mechanisms consisting of hardware firewalling and on-chip network physical isolation, whose mechanisms are combined with system-wide cryptographic techniques in automotive Controller Area Network (CAN) communications to provide authentication and confidentiality.

3. Hardware description

The adaptability of the underlying hardware and upgradable security procedures can prepare electronic devices for forthcoming security vulnerabilities. An AXI transaction firewall has been proposed in this work (AXI Firewall). Our security approach is based on an IP core that must be placed next to the AXI Masters, between them and the slaves, to protect slaves from fraudulent Masters. Input/output cores in single or multimaster environments are protected with the presented firewall. Unreliable transactions are blocked according to several rues. The IP can refuse requests by comparing the type of operation – write or read –, the address range, and the required bandwidth with previously defined rules collections. A Field-Programmable Gate Array (FPGA) based Zynq device has been used to develop the hardware-based firewall demonstrating that demanded resources are pretty constrained. Several case studies have been implemented, proving the benefits of our security approach. A low-latency security solution with reduced resource consumption has been exposed.

The main objective of AXIFirewall is to be able to protect AXI slaves from bogus AXI Masters. In order to do so, the circuit must be introduced between both master and slave, but next to the master. The position is determined because AXI transactions do have destination addresses but not origin ones. Fig. 1 depicts a simple diagram with the proposed system architecture.

The IP can block requests due to the type of operation (read or write), address range, and bandwidth.

This AXI transaction firewall is fully compliant with the AXI Lite standard. The architecture is compatible with AXI Full, but we have decided to focus on small IP Cores that cannot enforce security. The main design guidelines that have been followed are:

- Fully customizable in the number of address ranges, bandwidth...
- Minimum latency
- Minimum area

In Fig. 2, a basic diagram is shown. The IP core is distributed in two equal parts, one for the read channel and one for the write channel. At first glance, they may seem different since the write channel has three parts (address, data, and response) while the read has only two (address and data). Thanks to the nature of the reading process, the read response channel is embedded in the read data channel.

Each one of the blocks (read and write) is subdivided into two sub-blocks, one for the address filtering and one for the bandwidth filtering. Fig. 3 depicts the basic structure of the address matching block for the write channel. It consists of a series of mask and address registers compared against the transaction's address. The use of a mask adds flexibility, allowing the safelisting of whole ranges of addresses. If there is a match, the input and output channels are connected. If there is no match, the transaction is blocked. No information goes towards the address channel, and, at the same time, the master receives an error. This error allows the AXI transaction to end, not blocking the connection. The transactions for the read channel are equivalent. If the address is not validated, an error response is sent through the data channel.

The other working mode is bandwidth control. The block diagram for the write channel can be seen in Fig. 4. In this case, there is a counter per each address/mask pair. If there is a match in the address block, the corresponding counter is checked. When it is greater than zero, there is credit for the transaction. Data is passed towards the address channel, and the counter is decremented. Periodically, the counter is updated, adding credit to the counter.

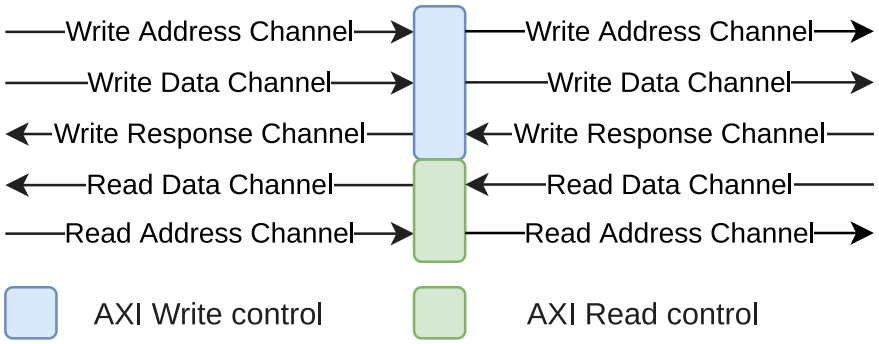

Fig. 2. AXI is composed of 5 channels. The firewall blocks 4 of them in two different manners. Valid blocking for address-related firewall rules. Ready blocking for bandwidth-related firewall rules.

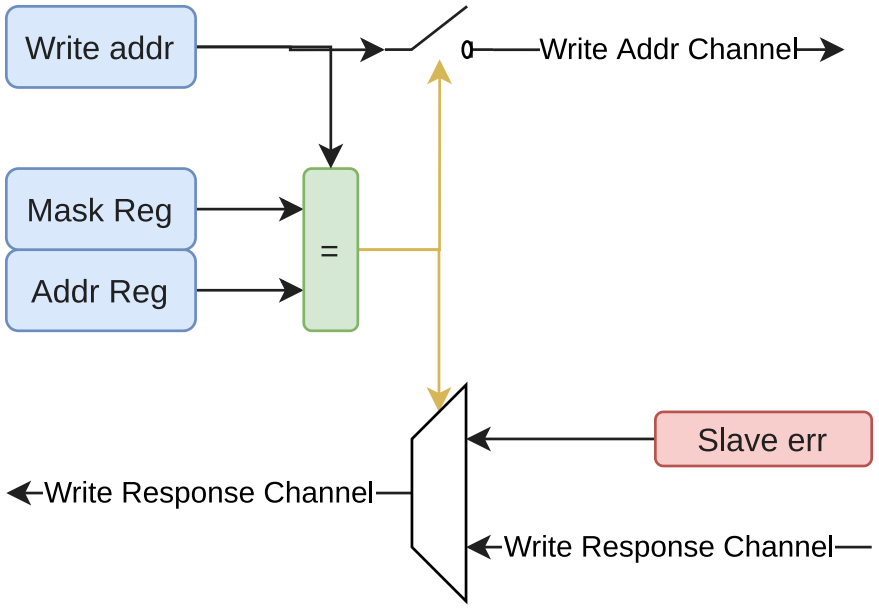

Fig. 3. Address blocking structure — write channel. Its main components are the safelist address registers and their corresponding mask registers.

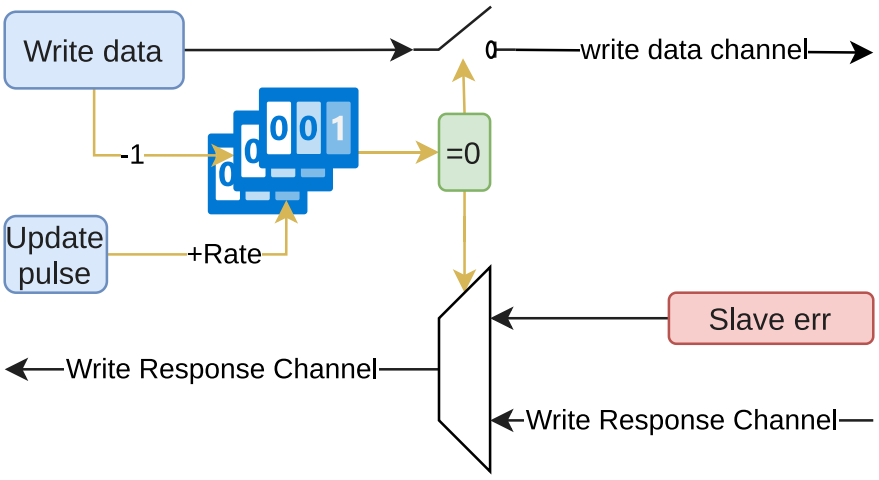

Fig. 4. Rate limiting structure — write channel. It uses a bucket structure. Every updating period credit is added to the bucket. Every data that passes the interface reduces the available credit. There is one counter per address range to rate limit.

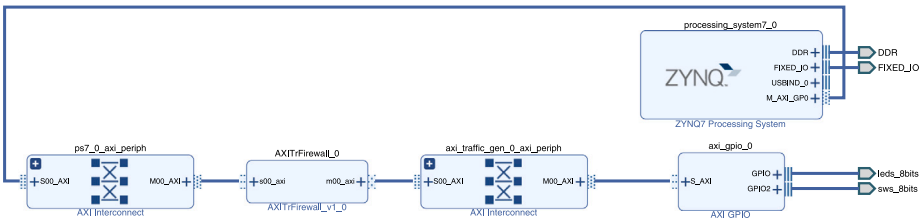

Fig. 5. Vivado block diagram of the final single master implemented design.

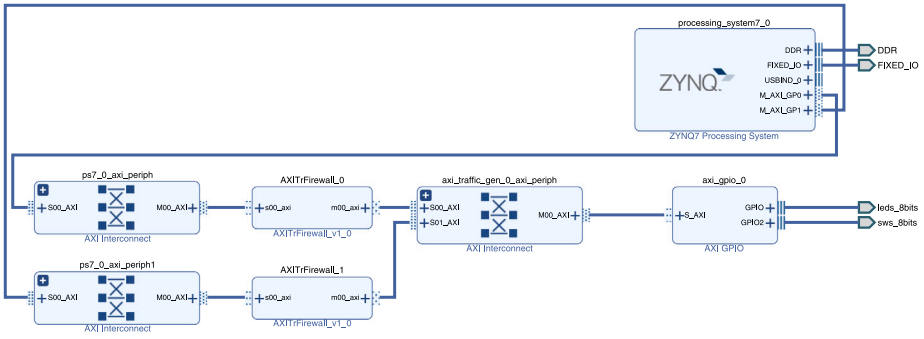

Fig. 6. Vivado block diagram of the final multimaster implemented design.

4. Use cases

This section shows different use cases. The examples make use of several standard IP cores from Xilinx:

- AXI Interconnect: AXI switch that allows protocol translation and communication among different IP cores.
- AXI General-Purpose Input/Output (GPIO): AXI core that allows interfacing external generic pins, such as Light Emitting Diodes (LEDs) or switches, as in the examples.
- ZYNQ7 Processing System (PS): IP core that wraps the ARM processor subsystem present in the FPGA.
- AXI Traffic Generator: IP core that generates traffic on the AXI bus according to a configuration file.

Fig. 5 depicts a standard use case. In this case, we have a single master, although it is easily extensible to multiple masters. The proposed IP protects the path between master and slave. The IP can guarantee that special registers in the slaves cannot be written into or the write rate. Similarly, it can forbid the read of registers that should not be accessible to the master. There are multiple AXI Interconnects. These IP Cores are automatically introduced by the tool to translate any variations of the AXI standard between Cores. The first one is compulsory since the AXI flavor inside the PS is AXI 3 Full, while the IP Core is AXI 4 Lite. The second AXI Interconnect is optional and is meaningful when there are several masters.

Fig. 6 shows the same system in a multimaster environment. In this case, we can make that the first master only accesses the first port of the GPIO (LEDs) while the second master only accesses the second port (switches). The configuration for this example requires including the first register of the GPIO IP in the firewall connected to the first master and the second register for the second master. Any other configuration is possible; for example, one master writes, the other reads,...

In Fig. 7, we can see a particular case use of the proposed core. There are two masters in the system, one secured and one non-secured. The proposed IP protects the path from the non-secured master towards the slave. The main points of such an architecture are:

- The use of a hardware-only master (`axi_traffic_gen`) to securely configure the slave IP cores. This IP can be signed and encrypted as it is part of the bitstream.
- The use of a hardware/software master (in this case, the ARM core inside the FPGA) to perform reads on the IP core.
- The use of the proposed IP to enforce that even if the software becomes compromised, the slave IP core cannot be misconfigured.

5. Results

The verification process has been done using a modified version of the single master use case (see Fig. 5). Fig. 8 depicts the evaluation testbed. The circuit uses a `axi_traffic_gen` as a bus master for the transactions. This IP emulates a processor with

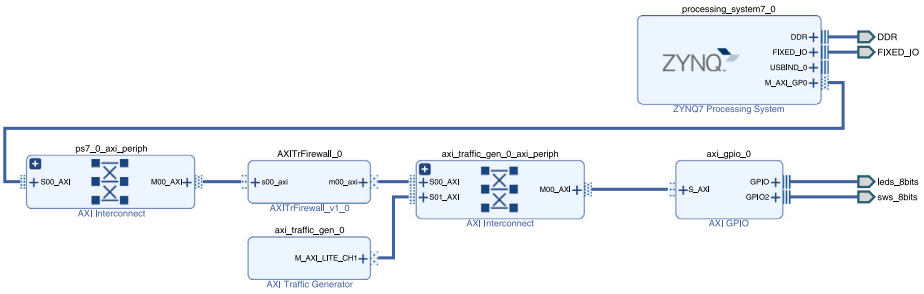

Fig. 7. Vivado block diagram of the implementation of two masters. Initialization secure master and run-time non-secured master.

Fig. 8. Vivado block diagram of the evaluation testbed. AXI master, AXI Firewall, and AXI Slave (GPIO) — with required AXI interconnect and other AXI infrastructure cores.

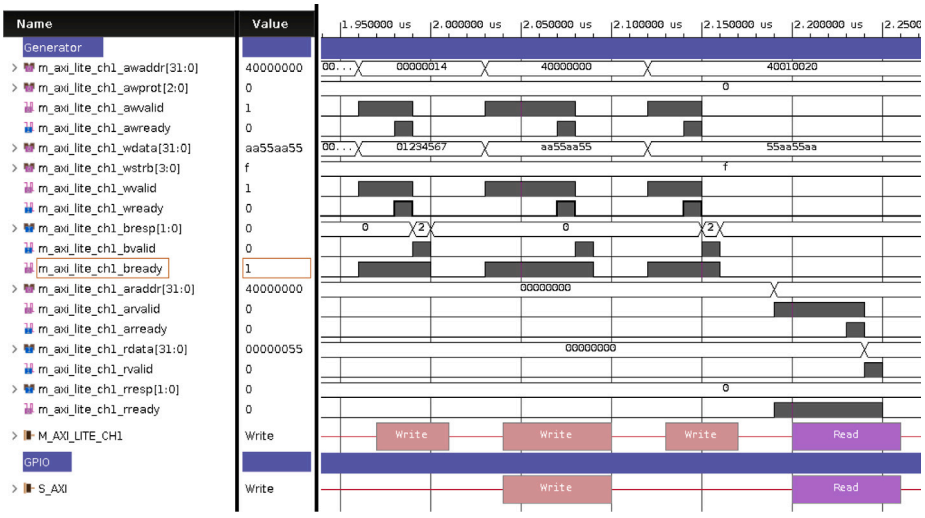

Fig. 9. Result for address filtering. Write to the selected slave (0x40000000) passes the firewall. The rest of the writes are responded with `bresp` signaling slave error.

its associated software. The master is in charge of performing several reads and writes to test the different required configurations. This procedure can also be used to test the rules included in the firewall in a controlled environment. In this way, the set of rules can be tested in a continuous integration flow without actual hardware.

The results can be seen in Figs. 9 and 10. The main control signals, as per the AXI standard, are `VALID` and `READY`. No transaction is performed until both signals are true during a rising edge of the bus clock.

In Fig. 9, address filtering is shown. First and third write accesses are blocked while second write and first read are performed. The mechanism to block includes informing the master of an error in the slave so that the AXI bus is not locked.

Fig. 10 depicts a rate limitation. The IP core is configured only to allow four transactions per cycle (256 clock cycles). Both parameters – allowed transactions and cycle size – are fully parameterized. The capture clearly shows that the write rate has been controlled.

With these proofs of concepts, we can demonstrate that AXIFirewall is capable of solving the following security risks:

- A rogue master attempting to access a non-allowed slave register.
- A malicious master trying a Denial of Service (DoS) attack on a slave.

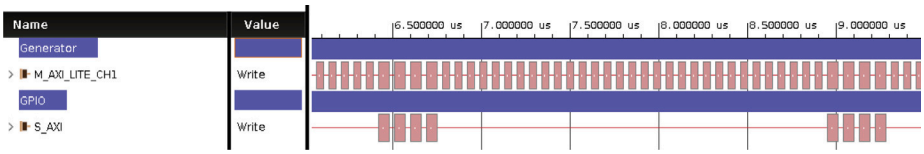

Fig. 10. Result for rate filtering. The allowed bandwidth is 4 bytes per cycle (256 clock cycles).

Table 1
Comparison of area results. The results for this paper are for 4 R/W areas.

Author	LUT	FF	Block RAM BRAM
This paper	188	90	0
Cotret et al. [2]	293	123	1
Gundabolu et al. [17]	228	228	0
Tan et al. [19]	237	–	0
Kornaros et al. [20]	195	107	0

Table 2
Comparison of timing results.

Author	Latency (clock cycles)	Clock frequency (MHz)
This paper	0	166
Cotret et al. [2]	2	100
Gundabolu et al. [17]	3	–
Grammatikakis et al. [18]	4	100
Kornaros et al. [20]	5	–

5.1. Area

The IP core uses a straightforward approach to minimize the required resources. For four read and four write areas with their corresponding bandwidth limitation, the area resources for a Zynq device are:

- 128 Look-Up Tables (LUTs)
- 90 Flip-Flops (FFs)

5.2. Time

The AXI to AXI path in the firewall is fully combinational. Only a switch connects the input to the output, as seen in Fig. 3, allowing a zero clock cycle delay. In other words, the presence of the firewall does not affect the overall performance of the system.

The IP has its sequential parts and adds a combinational delay to the AXI to AXI path. This can have an impact on the maximum AXI clock frequency. In our experiments, the AXI clock runs at 166 MHz. The same system, without the firewall, was capable of running at 187.5 MHz. The result is a 12 % decrease in maximum clock frequency.

6. Comparison

The proposed IP core is compared with those present in the literature to understand the contributions better. Table 1 shows the area results for several proposals presented in Section 2.

As can be seen, the proposed IP core performs slightly better than others. It is to be noted that this approach does not require Block RAMs (BRAMs), making only use of general-purpose resources.

Table 2 shows the results for timing comparisons. Nevertheless, the papers present in the literature do not always provide every timing parameter required for a complete comparison.

The approach presented in this paper has the best possible latency, zero. At the same time, it allows fast bus speeds. This proposal performs favorably in every aspect compared to the papers previously studied.

The proposed simulation testbed also provides advantages over those implementations discussed in this section from the simulation point of view. Instead of using analytical methods, AXIFirewall is tested in an environment that is useful for demonstration purposes and the implementation phase. As mentioned before, the testbed can be included in the development phase to test that the implemented rules can protect the system against the threats that want to be covered.

7. Conclusion

The growth of MPSoC technology results in an interest increase in security considerations and memory protection applications. MPSoCs can be protected against data modification, data extraction, and denial of service attacks. The embedded system could be in jeopardy because of modules that can be programmed and of unknown IPs. Hence, requirements of security ought to be suitably deployed. Packet inspection through firewall insertion is a widespread method. It is based on improving the security of the hardware performing as firewalls. Besides, it is also possible to modify security tables dynamically.

An approach to enhance security in SoC is presented in this work. Embedded system memories and communications have been protected with the approach of distributed firewalls. AXI transaction firewalls have been implemented to provide security, and updates can be executed through additional software.

The presented IP core can block requests between AXI Master and Slaves based on the type of operation, address range, and bandwidth. So, our AXI transaction firewall filters malevolent data intrusion. It blocks or allows requests depending on the correspondence between the content of the packet and the firewall security rules.

The complex challenge of secure architectural solutions guarantees a correct and adequate separation between program code and data among reliable and unreliable applications without compromising performance. Several use cases have been implemented, and a verification process has been performed to test the firewall's included rules and prove the proposed IP's effectiveness. The presented AXI firewall uses a straightforward approach to reduce the required resources, and its presence does not affect the system's overall time performance. The impact on the area resources and the maximum AXI clock frequency for a Zynq device has been measured to characterize and validate the proposed security solution.

CRediT authorship contribution statement

Jesús Lázaro: Conceptualization, Methodology, Validation, Resources, Writing – original draft, Writing – review & editing, Supervision, Project administration. **Unai Bidarte:** Investigation, Writing – review & editing. **Leire Muguira:** Investigation, Data curation, Writing – review & editing. **Armando Astarloa:** Investigation, Writing – review & editing. **Jaime Jiménez:** Methodology, Formal analysis, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Fiorin L, Lukovic S, Palermo G, di Milano P. Implementation of a reconfigurable data protection module for NoC-based MPSoCs. In: 2008 IEEE international symposium on parallel and distributed processing. IEEE; 2008, http://dx.doi.org/10.1109/ipdps.2008.4536514.

[2] Cotret P, Gogniat G, Sepúlveda Flórez MJ. Protection of heterogeneous architectures on FPGAs: an approach based on hardware firewalls. Microprocess Microsy 2016;42:127–41. http://dx.doi.org/10.1016/j.micpro.2016.01.013.

[3] Wolf W, Jerraya A, Martin G. Multiprocessor system-on-chip (MPSoC) technology. IEEE Trans Comput-Aided Des Integr Circuits Syst 2008;27(10):1701–13. http://dx.doi.org/10.1109/tcad.2008.923415.

[4] Sepulveda J, Flórez D, Immler V, Gogniat G, Sigl G. Efficient security zones implementation through hierarchical group key management at NoC-based MPSoCs. Microprocess Microsy 2017;50:164–74. http://dx.doi.org/10.1016/j.micpro.2017.03.002.

[5] Fiorin L, Silvano C, Sami M. Security aspects in networks-on-chips: Overview and proposals for secure implementations. In: 10th Euromicro conference on digital system design architectures, methods and tools. IEEE; 2007, http://dx.doi.org/10.1109/dsd.2007.4341520.

[6] Ray S, Peeters E, Tehranipoor MM, Bhunia S. System-on-chip platform security assurance: Architecture and validation. Proc IEEE 2018;106(1):21–37. http://dx.doi.org/10.1109/jproc.2017.2714641.

[7] El Salloum C, Elshuber M, Höftberger O, Isakovic H, Wasicek A. The ACROSS MPSoC – A new generation of multi-core processors designed for safety–critical embedded systems. Microprocess Microsy 2013;37(8):1020–32. http://dx.doi.org/10.1016/j.micpro.2013.08.002.

[8] Hagan M, Siddiqui F, Sezer S, Kang B, McLaughlin K. Enforcing policy-based security models for embedded SoCs within the internet of things. In: 2018 IEEE Conference on dependable and secure computing. IEEE; 2018, http://dx.doi.org/10.1109/desec.2018.8625140.

[9] Kocher P, Horn J, Fogh A, Genkin D, Gruss D, Haas W, et al. Spectre attacks: Exploiting speculative execution. In: 2019 IEEE symposium on security and privacy. IEEE; 2019, http://dx.doi.org/10.1109/sp.2019.00002.

[10] Dessouky G, Frassetto T, Jauernig P, Sadeghi A-R, Stapf E. With great complexity comes great vulnerability: From stand-alone fixes to reconfigurable security. IEEE Secur Privacy 2020;18(5):57–66. http://dx.doi.org/10.1109/msec.2020.2994978.

[11] Zhang N, Sun K, Shands D, Lou W, Hou YT. TruSense: Information leakage from TrustZone. In: IEEE INFOCOM 2018 - IEEE conference on computer communications. IEEE; 2018, http://dx.doi.org/10.1109/infocom.2018.8486293.

[12] Wu Y, Liu Y, Liu R, Chen H, Zang B, Guan H. Comprehensive VM protection against untrusted hypervisor through retrofitted AMD memory encryption. In: 2018 IEEE international symposium on high performance computer architecture. HPCA, IEEE; 2018, http://dx.doi.org/10.1109/hpca.2018.00045.

[13] Xu J, Zhang Y, Fu K, Peng S. SGX-based secure indexing system. IEEE Access 2019;7:77923–31. http://dx.doi.org/10.1109/access.2019.2921223.

[14] Papagrigoriou A, Petrakis P, Grammatikakis M. A firewall module resolving rules consistency. In: 2017 13th Workshop on intelligent solutions in embedded systems. IEEE; 2017, p. 73–8. http://dx.doi.org/10.1109/wises.2017.7986931.

[15] Sharma G, Bousdras G, Ellinidou S, Markowitch O, Dricot J-M, Milojevic D. Exploring the security landscape: Noc-based MPSoC to cloud-of-chips. Microprocess Microsy 2021;84:103963. http://dx.doi.org/10.1016/j.micpro.2021.103963.

[16] Hu Y, Muller-Gritschneder D, Sepulveda MJ, Gogniat G, Schlichtmann U. Automatic ILP-based firewall insertion for secure application-specific networks-on-chip. In: 2015 Ninth international workshop on interconnection network architectures: On-chip, multi-chip. IEEE; 2015, http://dx.doi.org/10.1109/ina-ocmc.2015.9.

[17] Gundabolu S, Wang X. On-chip data security against untrustworthy software and hardware IPs in embedded systems. In: 2018 IEEE computer society annual symposium on VLSI. IEEE; 2018, http://dx.doi.org/10.1109/isvlsi.2018.00122.
[18] Grammatikakis MD, Papadimitriou K, Petrakis P, Papagrigoriou A, Kornaros G, Christoforakis I, et al. Security in MPSoCs: a noc firewall and an evaluation framework. IEEE Trans Comput-Aided Des Integr Circuits Syst 2015;34(8):1344–57. http://dx.doi.org/10.1109/tcad.2015.2448684.
[19] Tan B, Biglari-Abhari M, Salcic Z. A system-level security approach for heterogeneous MPSoCs. In: 2016 Conference on design and architectures for signal and image processing. IEEE; 2016, http://dx.doi.org/10.1109/dasip.2016.7853800.
[20] Kornaros G, Tomoutzoglou O, Coppola M. Hardware-assisted security in electronic control units: Secure automotive communications by utilizing one-time-programmable network on chip and firewalls. IEEE Micro 2018;38(5):63–74. http://dx.doi.org/10.1109/mm.2018.053631143.

Jesús Lázaro is a Full Professor at the Department of Electronics Technology of the University of the Basque Country. He is the author or co-author of 4 patents, 35 articles in international scientific. His main research areas are high-speed circuits based on reconfigurable devices and communications devices.

Unai Bidarte received M.S. and Ph.D. degrees in Telecommunication Engineering from the University of the Basque Country (UPV/EHU), Spain, in 1996 and 2004. He is Associate Professor at UPV/EHU and researcher of the Applied Electronics Research Team. He is co-author of 3 patents, more than 10 papers indexed in JCR, and more than 60 other contributions to magazines and conferences.

Leire Muguira received a Ph.D. degree in telecommunications engineering from the University of the Basque Country (UPV), Spain, in 2015. In 2018, she started at the UPV. She has participated in 10 research projects and 2 research contracts. She is the author or co-author of a patent, 6 JCR articles, a book chapter and 23 papers in scientific conferences.

Armando Astarloa is a Full Professor at the Department of Electronics Technology of the University of the Basque Country. He is the author or co-author of 30 articles in international scientific magazines. His main research areas are high-speed circuits based on reconfigurable devices, digital control architectures, and communications devices.

Jaime Jiménez received M.S. and Ph.D. degrees from the University of the Basque Country, in 1991 and 2005. He has participated in 45 competitive research projects supported by public institutions and 39 private research contracts. He is author or co-author of 26 articles in scientific international journals. His areas of research are high-speed circuits on reconfigurable devices and communications devices.