Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı 26, S. 128-132, Temmuz 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue 26, pp. 128-132, July 2021

Research Article

FPGA Hardware Implementation of a SHA384 Accelerator for Internet of Things Applications

İhsan Çiçek[1*]

[1] İstinye Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Bölümü, İstanbul, Türkiye (ORCID: 0000-0002-7881-1263)

(3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications June 11-13, 2021)

(**DOI:** 10.31590/ejosat.951580)

ATIF/REFERENCE: Cicek, I. (2021). FPGA Hardware Implementation of a SHA384 Accelerator for Internet of Things Applications. *European Journal of Science and Technology*, (26), 128-132.

Abstract

The abundance of the IoT devices surrounding us brings new opportunities and challenges. IoT technology enables remote monitoring and control of cyber-physical systems on a global scale. One key aspect of IoT technology is the security which is usually neglected by manufacturers. Because of IoT based security breaches, IoT devices need cryptographic functions to provide confidentiality, integrity and authentication capabilities in modern applications. However, the limited computational power available in the processors used in IoT systems imposes the development and use of hardware peripherals dedicated for performing cryptographic operations. One of the most popular cryptographic functions used in the IoT applications is the secure hash algorithms. They are extensively used for data integrity and authentication applications. In this work, we have designed, verified, and implemented a hardware IP core of the SHA-384 algorithm. In addition, we have also integrated the SHA-384 hardware module with a synthesizable processor as an AXI4 peripheral to enable in-application testing using custom software. Our design can operate up to 170 MHz and occupies only 982 CLB slices and one BRAM on a Xilinx Artix-7 FPGA device. The estimated total power consumption is 223 mW when the module is integrated with a minimal configuration Microblaze processor system.

Keywords: SHA2, SHA-384, Secure Hash Algortihm, FPGA, Cryptography.

Nesnelerin İnterneti Uygulamaları İçin Bir SHA-384 Hızlandırıcısının FPGA Donanım Gerçeklemesi

Öz

Çevremizdeki IoT cihazlarının bolluğu yeni fırsatları ve zorlukları getiriyor. IoT teknolojisi, siber-fiziksel sistemlerin küresel ölçekte uzaktan izlenmesini ve kontrolünü sağlamaktadır. IoT teknolojisinin önemli bir yönü, genellikle üreticiler tarafından ihmal edilen güvenliktir. güvenlik ihlalleri nedeniyle IoT cihazları, modern uygulamalarda gizlilik, bütünlük ve kimlik doğrulama yetenekleri sağlamak için kriptografik işlevlere ihtiyaç duyar. Bununla birlikte, IoT sistemlerinde kullanılan işlemcilerin sınırlı hesaplama gücü, kriptografik işlemleri gerçekleştirmek için adanmış donanım çevre birimlerinin geliştirilmesini ve kullanılmasını zorunlu kılar. IoT uygulamalarında kullanılan en popüler kriptografik işlevlerden biri güvenli özet algoritmalarıdır. Veri bütünlüğü ve kimlik doğrulama uygulamaları için yaygın olarak kullanılırlar. Bu çalışmada, SHA-384 algoritmasını bir donanım çekirdeği şeklinde tasarladık, doğruladık ve gerçekledik. Ek olarak, özel yazılım kullanarak uygulama içi testler yapabilmek için SHA-384 donanım modülünü bir AXI4 çevre birimi olarak sentezlenebilir bir işlemci ile entegre ettik. Tasarımımız 170 MHz'e kadar çalışabilmektedir ve bir Xilinx Artix-7 FPGA tümleşik devresinde yalnızca 982 CLB dilimi ve bir BRAM kaplamaktadır. Modül minimum konfigürasyonlu Microblaze işlemci sistemiyle entegre edildiğinde tahmini olarak toplam 223 mW güç tüketmektedir.

Anahtar Kelimeler: SHA2, SHA-384, Güvenli özet algoritmaları, FPGA, Kriptografi.

[*] Corresponding Author: İstinye Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Bölümü, İstanbul, Türkiye, ORCID: 0000-0002-7881-1263, ihsan.cicek@istinye.edu.tr

1. Introduction

The rise of the Internet-of-Things (IoT) in the last decade has enabled many beneficial applications that yield new opportunities and challenges. IoT technology allows remote monitoring, control and micro-management of cyber-physical systems on a global scale. Future projections predict that the number of connected devices will be measured in billions and the economic impact of the IoT will be significant as presented in Figure 1 [1]. Accordingly, security became a critical aspect of the IoT technology. Unfortunately, it is usually overlooked in the product development cycle. As a result of the factors such as additional development costs, time-to-market push and lack of vision, IoT device manufacturers neglect the security requirements which creates the Achille's heel and paves the way for potential security breaches and hazards [2]. Considering security at the beginning of the design stage can save time and mitigate later corrective efforts. Additionally, secure-by-design approach can protect IoT companies against financial and reputation losses of the unpredictable future.

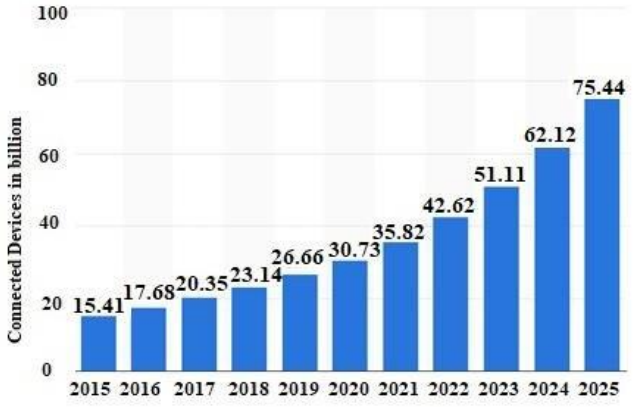

Figure 1. Global future projection for the deployed IoT devices toward 2025 [1].

Forward-looking security provisions impose the use of cryptography due to the raised risks associated with the ever-increasing connectivity of IoT devices. Cryptographic primitives and algorithms provide the methods for confidentiality, data integrity and authentication for communicating parties. However, the computational requirements of cryptographic algorithms still pose a problem for the cost-sensitive lightweight IoT systems. The low-cost processors used in such systems usually have confined computational power and performance of the cryptographic algorithms implemented in software is mediocre. Moreover, cryptographic software will consume the limited codespace that is mainly reserved for the IoT application of interest. Conventional approach is to use dedicated cryptographic hardware peripherals to aid the microprocessor in performing information security related tasks. This not only offloads the CPU but also provide additional security at the hardware level.

In IoT network communication, hash based message authentication codes (HMACs) are used for message authentication and/or integrity checking [3]. Although block ciphers can be used, hash functions are more preferred in practice, as they are easier to compute and flexible enough to be used globally without any export restrictions [3]. Thus, cryptographic strength of the HMAC depends heavily on the underlying hash function's cryptographic strength, the size of its hash output, and the size and randomness quality of the key used. The use of hash-based message authentication codes has been standardized and adopted to many popular communication protocols [4]. Today, hash-based message authentication codes are widely used within the SSH, TLS and IPSEC protocols. Consequently, it is always necessary to compute the hashes of the data to be transmitted or received in an IoT device and this can create a computational performance bottleneck for the local processor. Any cryptographic hash function, such as SHA-1, SHA-2 or SHA-3 can be used in the calculation of a hash-based message authentication code [5]. Secure hash functions are free of collisions and the generated outputs are unique and unparalleled for every single data input. Authentication and integrity are two of the essences needed to build secure network systems, and SHA algorithms provide both. SHA2 replaced SHA1 after the success of an attack and it has been further superceded by SHA3 as an upgrade [6,7]. However, SHA2 still remains to be the most widely deployed and used secure hash function. The SSL/TLS certificates today use 384-bit SHA2 (SHA-384) digests to ensure security for internet communication.

In this work, we present the hardware design of SHA-384 algorithm as an IP core that can be wrapped as a peripheral and integrated with a processor that targets lightweight and cost-sensitive IoT applications. Our key contribution is the development of a small-footprint SHA-384 hardware module that provides high performance in a small footprint. The paper is organized as follows: In Section 2, we briefly introduce the SHA-384 algorithm and its process flow. Section 3 provides the details of platform used for the development of the IP core along with functional verification and hardware in the loop validation of the proposed design. Section 4 discusses the evaluation results and marks future directions.

2. SHA-384 Secure Hash Algorithm Overview

Workflow of the SHA-384 algorithm is based on operations such as message padding, parsing and one way hash functions as presented in Figure 2. The listed steps below briefly explain how SHA-384 algorithm performs pre-processing, and how the hash value is calculated by using the padding, parsing and compressing functions according to [5]:

1. Padding the message (To make sure that the padded message is a multiple of 1024 bits).

2. Parsing the message (The massage and its padding are parsed into N 1024-bit blocks).

3. Setting the initial hash value (Consists of eight 64-bit words in hexadecimal format).

An overview of the algorithmic flow of the SHA-384 is shown in Figure 2. At the beginning, the message is divided into N-blocks, each of which is comprised of 1024 bits. The initial hash value (H_0) is assigned at first, before any operation. After the initialization process the successive hash values are computed sequentially, one block at a time, such that every calculated hash value is transferred to the next block. The final hash code of the message is determined after the computation of the N-th message block and set as the final output [5].

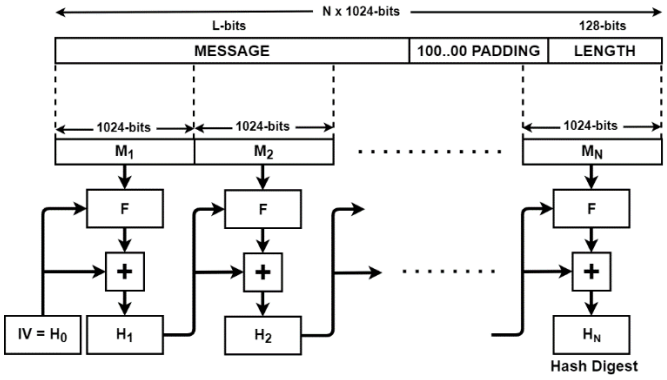

Figure 2. Algorithmic process flow of the SHA-384.

3. Hardware Implementation of the SHA-384 Secure Hash Algorithm

3.1. Design of the SHA-384 Secure Hash Algorithm IP Core

The reconfigurability and parallelization features of the field programmable gate arrays (FPGAs) render them as convenient platforms for hardware prototyping of algorithms. We used an FPGA development board that has a Xilinx Artix 7 family device (XC7A35T-1CPG236C) to design and implement the SHA-384 secure hash algorithm IP core. SHA-384 is primarily based on scarmbling the input with some predefined constants, then computing the hash values through corresponding special functions dedicated for each step, and finally, after the acquisition of all inputs, the yielding hash values are used to calculate the SHA-384 final output digest value. The top level interface of the SHA-384 module is shown in Figure 3. The design accepts message data in 64-bit blocks and also outputs the final hash value in six consecutive 64-bits of data, so six 64-bit or twelve 32-bit registers are needed to keep the 384-bit final hash value on the host.

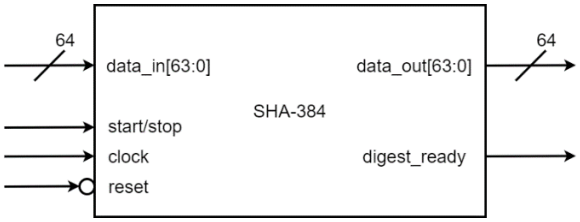

Figure 3. Hardware interface of the proposed SHA-384 module.

We developed and verified the hardware model of the SHA-384 using Verilog HDL in Xilinx Vivado integrated design environment. The module remains in idle mode by default after the power-on event and FPGA loads the configuration bitstream. The module starts operating after applying a logic high to the start/stop input. The 1024-bit message is loaded into the module in 16 clock cycles and hash computation starts right after. When the computation of the hash value is complete, and 384-bit computed hash value is output sequentially in six clock cycles by 64-bit words and the digest_ready flag output is held at logic high during this period. This signal can be used as an interrupt generator for the host system.

3.2. Functional Verification of the SHA-384 Secure Hash Algorithm IP Core

We performed functional verification of our design using the Vivado ISim HDL simulator. A testbench module is designed to drive the SHA-384 module and apply the test vectors. The functional verification simulation results are provided in Figure 4. The input presented in the simulation is “6162636461626364” corresponds to the hexadecimal value of the “abcdabcd”. The input message has been sent twice to the SHA-384 IP core, which means that the message passed to it is actually “abcdabcdabcdabcd”. After the message input has been received, the calculation of the hash is performed and the final digest value is presented by the DataOut signal. The final calculated digest output for the input message is equal to the expected value calculated using the software model of the SHA-384 algorithm.

3.3. Hardware Implementation of the SHA-384 Secure Hash Algorithm IP Core

The hardware module designed in Verilog HDL is synthesized and implemented for the target FPGA device (XC7A35T-1CPG236C) and the hardware resource utilization of the module is reported in Table 1 along with other studies in the literature. SHA-384 design can operate up to a maximum clock frequency of 170 MHz at the cost of 982 configurable logic block slices (CLBs) as shown in Table 1. Our design performs better in terms of both maximum clock frequency and hardware resource utilization when compared to the similar work in the literature.

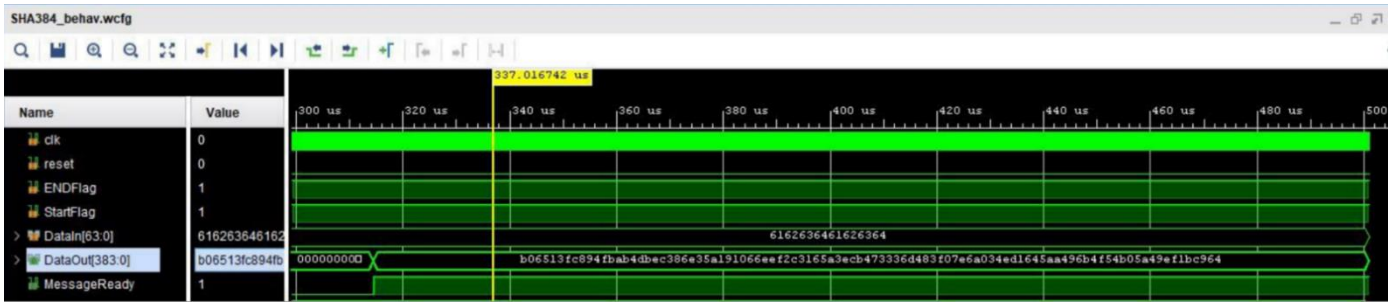

Figure 4. Functional verification of the designed SHA-384 module.

Table 1. FPGA Hardware Resource Utilization for SHA-384 module

Metric	Our Design	[7]	[8]	[9]
Max clock frequency (MHz)	170	74	120.83	128.584
BRAM	1	N/A	N/A	N/A
CLB Slices	982	1966	4240	4289

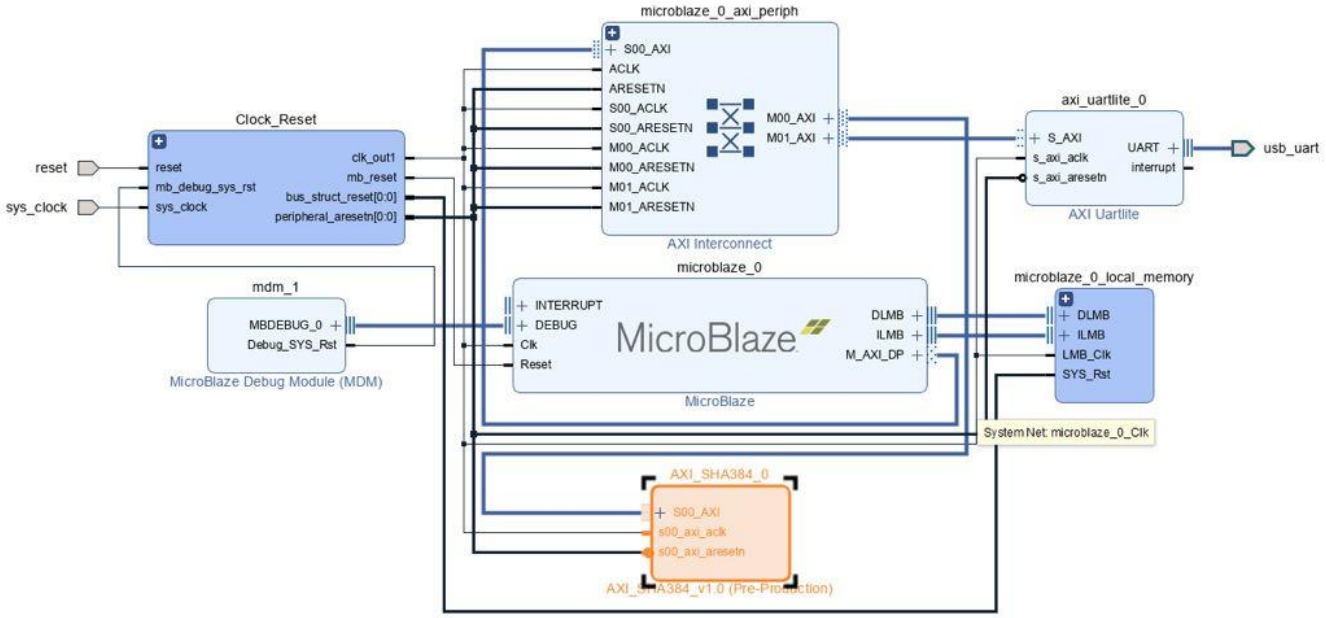

Figure 5. SHA-384 module integrated with the Microblaze RISC processor.

Table 2. FPGA Hardware Resource Utilization for the Microblaze Processor System

Clock Frequency (MHz)	Power (mW)	BRAM	CLB Slices	LUT	FlipFlop	DSP
100	223 mW	9	1538	5855	6152	1

The SHA-384 module has been integrated with a Microblaze synthesizable softcore processor. The use of a processor enables software control, and provides a convenient way to transfer data between the FPGA and PC in test scenarios. A customized AXI IP core that wraps the SHA-384 module and enables a communication interface with the Microblaze's AXI4 bus has been created. As MicroBlaze has many options for configuration, it had to be optimized by choosing the right specifications to generate a minimum footprint processor. Thus, a cacheless Microblaze core with a local memory of 32-KBytes has been chosen as the base hardware platform. The system clock at which the processor operates is configured as 100 MHz to reduce clocking hardware overhead. In addition, other IP cores such as the clocking and reset module and AXI4 bridge required for the proper operation of Microblaze processor have been added as shown in Figure 5. We used a UART-Lite IP core configured at 115200 bps to allow serial communication with a personal computer using the RS-232 protocol. Table 2 reports the hardware implementation results of the synthesized Microblaze based system with the SHA-384 module, and the total estimated power dissipation is 223 mW.

We have used the software development kit (SDK) provided by Xilinx to develop a test program using C. Developed software used SDK generated drivers for communicating with the synthesized Microblaze hardware system. We have successfully verified the correct operation of the integrated SHA-384 hardware peripheral in application.

4. Conclusion

In this study, we have designed a hardware IP core of SHA-384 secure hash algorithm in Verilog HDL for use in IoT applications such as HMAC generation. We functionally verified the correct operation of the design using HDL simulations. We have synthesized and implemeted the design on a Xilinx Artix-7 FPGA device. Our design is better in terms of area and speed when compared to the other designs in the literature. Designed SHA-384 hardware module can operate up to 170 MHz clock speed and consumes only 982 CLB slices on a Xilinx Artix-7 FPGA device. In addition, we have integrated the designed IP core with a Microblaze synthesizable processor for use in-application testing by custom software. Estimated total power consumption was reported to be 223mW. The compact footprint of the proposed design makes it very suitable for integration with processors used in IoT applications.

References

[1] Statista. (2016, November 27). Internet of Things - number of connected devices worldwide 2015-2025. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ (Accessed Feb 2, 2021).

[2] CISO Magazine. (2020, January 10). 10 IoT Security Incidents That Make You Feel Less Secure. https://cisomag.eccouncil.org/10-iot-security-incidents-that-make-you-feel-less-secure/ (Accessed Feb 2, 2021).

[3] Bellare M., Canetti R., Krawczyk H. (1996) Keying Hash Functions for Message Authentication. In: Koblitz N. (eds) Advances in Cryptology — CRYPTO '96. CRYPTO 1996. Lecture Notes in Computer Science, vol 1109. Springer, Berlin, Heidelberg.

[4] Dang Q. (2008, July). The Keyed-Hash Message Authentication Code (HMAC), Federal Inf. Process. Stds. (NIST FIPS 198-1), National Institute of Standards and Technology, Gaithersburg, MD, https://doi.org/10.6028/NIST.FIPS.198-1.

[5] National Institute of Standards and Technology (2015, August) Secure Hash Standard (SHS), Federal Inf. Process. Stds. (NIST FIPS 180-4), National Institute of Standards and Technology, Gaithersburg, MD, https://doi.org/10.6028/NIST.FIPS.180-4.

[6] Lenstra, A. (2005, February 26). Further progress in hashing cryptanalysis. Lucent Bell Laboratories, http://bell-labs.co/who/akl/hash.pdf (Accessed Feb 4, 2021).

[7] Sklavos N., Koufopavlou O. (2003). On the hardware implementation of the SHA-2 (256, 384, 512) Hash functions. In 2003 International Symposium on Circuits and Systems (ISCAS), (pp.153-156). IEEE. https://doi.org/10.1109/ISCAS.2003.1206214.

[8] McLoone M., McCanny J. V. (2003, March 26). Efficient single-chip implementation of SHA-384 and SHA-512, In 2002 International Conference on Field-Programmable Technology (FPT), (pp. 311-314,). IEEE. https://doi.org/10.1109/FPT.2002.1188699.

[9] Li M., Xu J., Yang X., and Yang Z. (2009). Design and Implementation of Reconfigurable Security Hash Algorithms Based on FPGA. In *2009 WASE International Conference on Information Engineering*, (pp. 381-384). IEEE. https://doi.org/10.1109/ICIE.2009.278.