SHA2 and SHA-3 Accelerator Design in a 7nm Technology Within the European Processor Initiative

Pietro Nannipieri[a,*], Matteo Bertolucci,[a], Luca Baldanzi,[a], Luca Crocetti,[a], Stefano Di Matteo,[a], Francesco Falaschi,[a], Luca Fanucci,[a], Sergio Saponara,[a]

[a]*Department of Information Engineering, University of Pisa, Pisa, Italy*

Abstract

This paper proposes the architecture of the hash accelerator, developed in the framework of the European Processor Initiative. The proposed circuit supports all the SHA2 and SHA-3 operative modes and is to be one of the hardware cryptographic accelerators within the crypto-tile of the European Processor Initiative. The accelerator has been verified on a Stratix IV FPGA and then synthesised on the Artisan 7 nanometres TSMC silicon technology, obtaining throughputs higher than 50 Gbps for the SHA2 and 230 Gbps for the SHA-3, with complexity ranging from 15 to about 30 kGE and estimated power dissipation of about 13 (SHA2) to 26 (SHA-3) mW (supply voltage 0.75 V). The proposed design demonstrates absolute performances beyond the state-of-the-art and efficiency aligned with it. One of the main contributions is that this is the first SHA-2 SHA-3 accelerator synthesised on such advanced technology.

Keywords: SHA2, SHA-3, Hash, EPI (European Processor Initiative), Cryptography, ASIC, 7nm, FPGA verification, Keccak

1. Introduction

The European Processor Initiative (EPI) [1] is an H2020 project under development, whose aim is to design and implement a roadmap for a new family of low-power European processors for extreme-scale computing, high-performance Big-Data and a range of emerging applications, including automotive [2]. EPi is one of the cornerstones of the EuroHPC joint undertaking, a legal and funding entity which will enable to build and deploy in Europe world-competitive supercomputers. Entirely European processor Intellectual Proprieties (IPs) relays on an ARM64 processor unit array, beside heterogeneous tile for embedded FPGA, RISC-V 64 co-processors, Massively Parallel Processor Array (MPPA) accelerators, power controllers and secure elements.
In the framework of the targeted markets, automotive above all, robustness and safety issues play a significant role, exploiting the currently unaddressed need of having a proper architecture (notably at hardware level) to support the requirements of relevant standards and certifications. EPI consortium opted for a layered and itemized hardware architecture to address security and safety issues, involving different IPs to accelerate in critical hardware functions. Security is a significant aspect of the EPI chip, and the impact that security-related data processing has on a general-purpose elaborator suggested that state-of-the-art functions, as well as future Security algorithm, shall be hardware accelerated.
The proposed EPI hardware security architecture relies on isolated blocks of the circuit, with specific hardware and dedicated private resources for security responsibility. Figure 1 illustrates the proposed EPI hardware security architecture, which foresees multiple Security Domains (SD) each controlled by a Secure Core (SC). Each SC and SD is equipped with its cryptographic hardware accelerator, named cryptoprocessor tile, for supporting the cryptographic functions required by the high-level security subsystem. The high-level security subsystem also comprises a compact microcontroller called Secure Element (SE), that is in charge of providing the Root of Trust (RoT) of EPI chip and to act also as a main or secondary SE needed by automotive recommendation.

*Corresponding author
Email address: pietro.nannipieri@ing.unipi.it (Pietro Nannipieri)

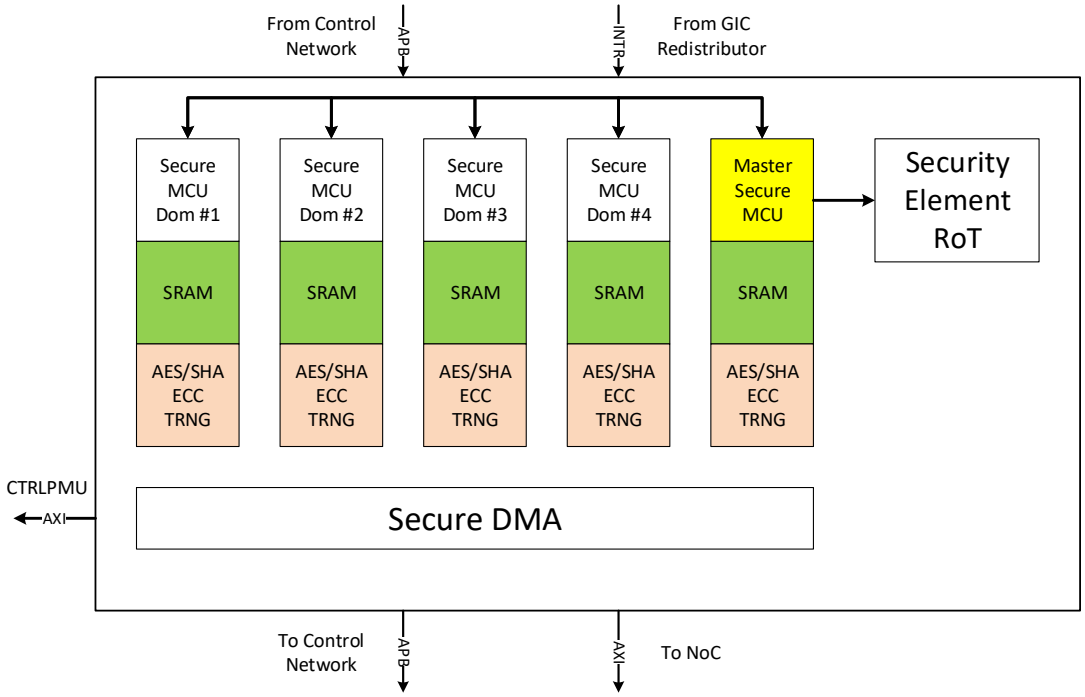

Figure 1: High-level EPI security subsystem architecture. Five secure domains are identified, each one with its Crypto-tile hardware accelerator (in orange)

The SE requires hardware-assisted cryptographic functions, including Advanced Encryption Standard (AES), Secure Hash Algorithm (SHA), Elliptic Curve Cryptography (ECC), and Random Number Generation (RNG). The security crypto-processor tile can be employed, by exploiting the secure Main Control Unit (MCU) interface only, in case a secure Direct Memory Access (DMA) connection is not available. In literature, there are many examples of cryptoprocessor supporting single crypto accelerator, like [3].
The proposed work belongs to a wider cryptoprocessor design project. However, this paper proposes the architecture for the so-called Hash Engine (HE) of the breaking trough EPI processor. The main contributions include, but are not limited, to:

- Architectural design of the HE, a hardware accelerator compliant to both SHA2 and SHA-3 standards supporting multiple configurations at 224, 256, 384 and 512 bits for the hash function.
- FPGA verification on a STRATIX IV board, with preliminary analysis on complexity and throughput.
- Implementation on the cutting-edge 7nm TSMC silicon technology (first contribution available to the best of our knowledge), with a complete analysis of complexity, throughput and power dissipation.

The optimised architectures developed for EPI will be explained in Sections 2.1 and 2.2. Their FPGA verification approach is described in Section 2.3. In Section 3 synthesis results (complexity, frequency, throughput & power dissipation) will be presented on the Artisan 7nm TSMC technology. Up to the author knowledge, this is the most advanced contribution for SHA2 and SHA-3 hash engine ASICs. In Section 4 brief comparison will be carried out, with the fragmented information available in the literature, to point out the advantages and disadvantages of the proposed work.

2. Cryptoprocessor Development

Figure 2 shows the outline of the cryptoprocessor tile architecture. The AXI-4 connections with the secure DMA and the secure MCU are aligned with the scheme

already proposed in Figure 1. The cryptoprocessor tile includes:

- Four crypto engines, each one dedicated to a specific family of cryptographic functionalities (i.e. SHA), including also a management unit (Finite State Machine, FSM) and registers for handling the underlying cryptographic engine for accelerating the cryptographic operation;
- A set of global registers for configuration, control and status of the whole crypto-tile;
- A 32-bit AXI-4 slave interface (supporting Full AXI-4), on the secure MCU side, which has access to all the crypto-tile registers;
- Four 128-bit AXI-4 slave interfaces (supporting Full AXI-4), on the secure DMA side, one for each cryptoprocessor. They only have access to the data registers of the cryptoprocessor they are linked.

The crypto-tile offers security services and algorithms based on symmetric-key and public-key encryption and decryption, hash functions and random numbers generation, aiming to guarantee the data confidentiality, integrity, authenticity, authentication and signature services. The crypto-tile supports the following hash functions algorithms:

- SHA2-224, SHA2-256, SHA2-384, SHA2-512;
- SHA-3-224, SHA-3-256, SHA-3-384, SHA-3-512

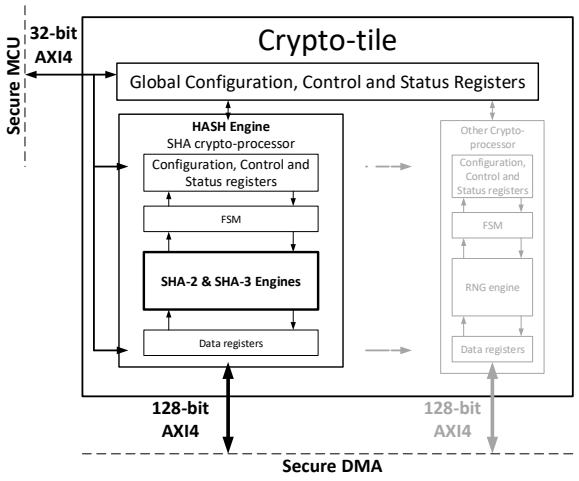

Figure 2: Cryptoprocessor tile architecture outline. It is composed by several Crypto-processor (we are interested in the Hash Engine), a global register space and AXI4 interfaces.

The SHA1 algorithm has not been considered secure since 2005 [6], and since 2010 many organizations have recommended its replacement by SHA2 or SHA-3 functions. Thus, the proposed cryptoprocessor does not support the SHA1 algorithm. However, since the cryptoprocessor is equipped with a RISC-V co-processor, it can be implemented in software for legacy support. Security services offered by the crypto-processor tile relies on the assumption that the minimum approved security strength for applying for long-term cryptographic protection on data is 128 bits while being 256 bits the maximum, as reported by Table 1.

Referring to Table 1, the *Process* column indicates whether cryptographic protection is being applied to data (e.g., encrypted), *Applying* case, or whether cryptographically protected data is being processed (e.g., decrypted), *Processing* case. The terms used in third and fourth columns have the following meaning: *Acceptable*, indicates that the algorithm or key length is not known to be insecure; *Legacy-use*, means that an algorithm or key length may be used because of its use in legacy applications; *Disallowed*, means that an algorithm or key length shall not be used for cryptographic protection.

Table 2 reports the comparison of the security strength level (in bits) offered by the typical cryptographic functions and algorithms [4], [5] (512+ means a number of bits greater or equal to 512).

In the case of hash functions HMAC, as shown in Table 2, the element which guarantees the security strength level is the digest generated by the algorithms indicated in those columns. The bit length of the digest is indicated by the last three digits of the algorithm name: for instance, the bit length of the digest generated by the SHA2-384 or SHA-3-384 algorithm is 384 bits. In the following, we provide the architectural description, highlining design choices, of the SHA2 and SHA-3 hardware accelerator for the EPI Processor.

2.1. SHA2 Architecture

Each SHA2 core (224, 256, 384 and 512) is based on the SHA2 cryptographic primitive. Of course, different cores will have different data paths (224, 256: 32 bits; 384, 512: 64 bits); however, they rely on the same architecture which is detailed and commented in this section. The implementation derived from the standard [7] has been improved to achieve higher throughput employing *Carry-Save Adder* (CSA) units for consecutive additions, together with retiming-pipelining operations to perform delay balancing. To better understand this architectural approach, it is necessary first briefly to describe the algorithm: we will take the SHA2-256 as

Table 1: Future projection of security strength protection, though 2030 and beyond [4].

Security strength	Process	Through 2030	2031 and beyond
< 112 bits	Applying	Disallowed	Disallowed
< 112 bits	Processing	Legacy-use	Legacy-use
112 bits	Applying	Acceptable	Disallowed
112 bits	Processing	Acceptable	Legacy-use
128 bits	Applying/Processing	Acceptable	Acceptable
192 bits	Applying/Processing	Acceptable	Acceptable
256 bits	Applying/Processing	Acceptable	Acceptable

Table 2: Cryptographic functions security strength [4], [5].

Security strength	Hash function	HMAC
112 bits	SHA2-224, SHA-3-224	-
128 bits	SHA2-256, SHA-3-256	SHA1
192 bits	SHA2-384, SHA-3-384	SHA2-224
256 bits	SHA2-512, SHA-3-512	SHA2-256, SHA2-384, SHA2-512, SHA-3-512

an example. It consists of two separate, ideally consecutive, procedures: the *message schedule* and the *compression function*. The *message schedule*, starting from the 512-bits *input message*, creates a *key* schedule, which is then provided as input to the *compression function*. The operation is performed through the σ_0 (Equation 1), σ_1 (Equation 2) and modulo 2^{32} adder (Equation 3) operations, whose operands (i.e. ROTL) are defined in [7]. The overall function is also know as *expansion*, as the input message (512 bits) is expanded to $32 \cdot 64 = 2048 bits$.

$$\sigma_0(x) = ROTR_7(x) \oplus ROTR_{18}(x) \oplus SHR_3(x) \quad (1)$$

$$\sigma_1(x) = ROTR_{17}(x) \oplus ROTR_{19}(x) \oplus SHR_{10}(x) \quad (2)$$

$$x \boxplus y = x + y \pmod{2^{32}} \quad (3)$$

$$W_t = \begin{cases} M_t & \\ & 0 < t < 15 \\ \sigma_1(W_{t-2}) \boxplus W_{t-7} \boxplus \sigma_0(W_{t-15}) \boxplus W_{t-16} & \\ & 16 < t < 63 \end{cases} \quad (4)$$

The optimization for the message schedule is performed on the adder chain through the use of CSA, which are essentially Full-Adder Arrays (FAA), producing Partial Sums (PS) and Shift-Carries (ShC).

$$\begin{aligned} ps_i &= a_i \oplus b_i \oplus b_i \\ shc_i &= (a_i \wedge b_i) \vee (a_i \wedge c_i) \vee (b_i \wedge c_i) \end{aligned} \quad (5)$$

Implementation on 45nm and 7nm ASIC standard-cell technologies demonstrated (see Section 3) that, when compared to *Carry-Lookahead Adder* (CLA) units, the delay relationship is $T_{CLA-32} = 1.78 \cdot T_{CSA-32}$ and $T_{CLA-32} = 1.87 \cdot T_{CSA-32}$ respectively for the two technologies. The optimized serial implementation is shown in the lower part of Figure 3, where the high-level timing block analysis shows that the critical path is reduced to $T_{\sigma_0} + 2 \cdot T_{CSA} + T_{\boxplus}$.

The SHA2-256 *compression function* performs 3 operations in sequence: initialization, one-way compression and termination. First, A-H variables are initialized with the intermediate hash value of $H^{(t-1)}$. The $H^{(0)}$ message block is a constant indicated in the standard). Then, the one-way compression operates 64 loops of elaboration(see [7] for details):

$$\begin{aligned} T_1 &\leftarrow H \boxplus \Sigma_1(E) \boxplus Ch(E,F,G) \boxplus K_j \boxplus W_j \\ T_2 &\leftarrow \Sigma_0(A) \boxplus Maj(A,B,C) \\ H &\leftarrow G \\ F &\leftarrow E \\ E &\leftarrow D \boxplus T_1 \end{aligned}$$

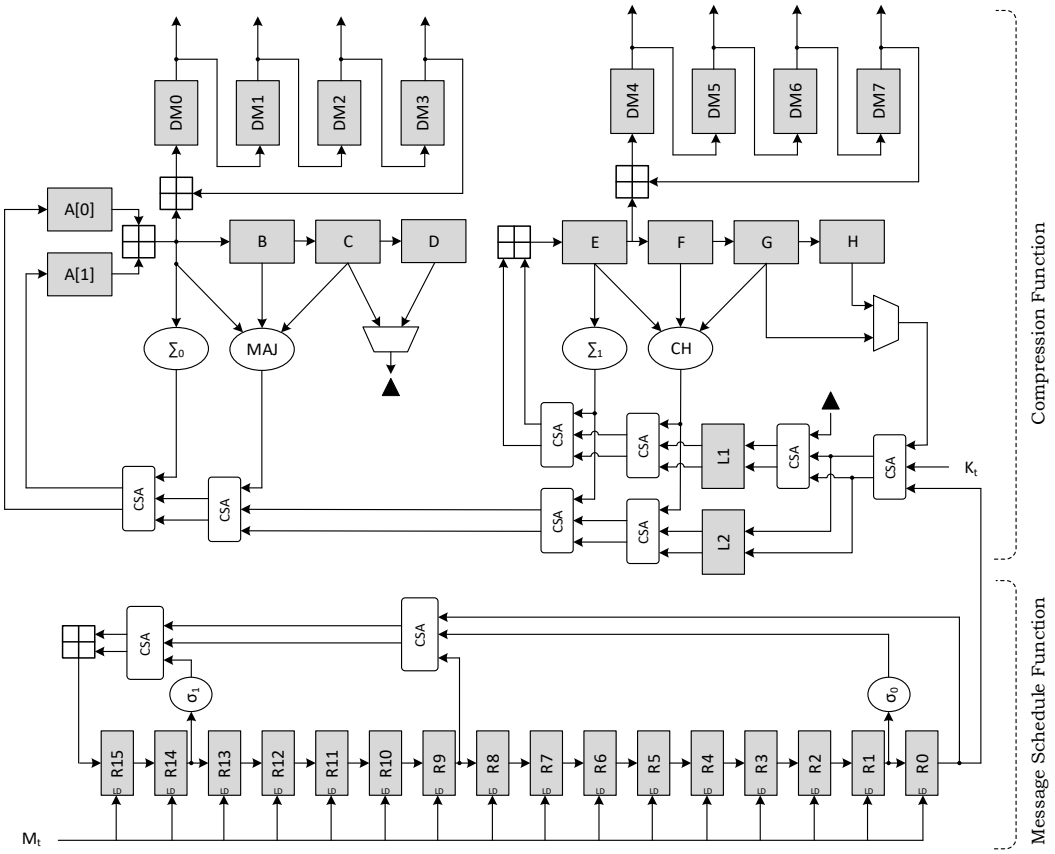

Figure 3: SHA2 accelerator architecture. In the top section, the compression function is carried out, in the bottom section the message-schedule function.

$$
\begin{aligned}
D &\leftarrow C \\
C &\leftarrow B \\
B &\leftarrow A \\
A &\leftarrow T_1 \boxplus T_2
\end{aligned}
$$

As last step, $H^{(t)}$ is calculated by adding modulo 2^{32} variables A-H after the one way compression and variables A-H at initialization time . The functions Maj, Ch, Σ_0 and Σ_1 are defined in [7].

$$
\begin{aligned}
&Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) \\
&Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z) \\
&\Sigma_0(x) = ROTR_2(x) \oplus ROTR_{13}(x) \oplus ROTR_{22}(x) \\
&\Sigma_1(x) = ROTR_6(x) \oplus ROTR_{11}(x) \oplus ROTR_{25}(x)
\end{aligned} \tag{6}
$$

The high-level block timing analysis showed that the critical path on the non-optimised architecture is located between register H and register E, involving 5 $\boxplus$ operations. The *compression function* has also been optimised employing CSA, retiming and delay balancing. In particular, all the adder chains were converted to CSA (except B and E register inputs). The path going from K_t-W_t was duplicated to allow the value of register D to be added immediately to $H + K_t + W_t$. Finally, a pipeline stage L_1, L_2 was added (with the associated C-D multiplexer to ensure the functionality), and the A register was split to move the CLA position. Its architectural scheme is shown in the upper part Figure 3.

2.2. SHA-3 Architecture

Continuing the exploration of the HE, the SHA-3 submodule focuses on high-performance computing. For this reason, our goal was to achieve the highest performance possible. We decided to consider the application scenario where only one user at a time have access to

Table 3: SHA-3 performances varying unrolling factor [8].

Unrolling	Circuit Area [kGE]	Frequency [MHz]	Throughput [Gbps]	Area Efficiency [Gbps/kGE]
n = 1	48	526	22.44	0.468
n = 2	67	333	28.44	0.424
n = 3	86	244	31.22	0.363
n = 4	105	192	32.82	0.313
n = 6	145	135	34.59	0.239

the function, meaning that multiple SHA-3 related requests will be executed one by one by the HE, and we optimized the core accordingly.

The SHA-3 standard [9] uses a sponge construction, in which data is absorbed into the sponge and then squeezed out to obtain the result. In the absorbing phase, message blocks are XORed into a subset of the state, which is then transformed as a whole using a permutation *f-function*. For each input block the *f-function* is repeated 24 times, each called round, before processing the next data block. The core of the SHA-3 architecture is then the *f-function*, which is made up of different sub-functions executed one following the other. All these sub-functions work on a 1600-bit state, which is shaped in a 5x5x64 cube visible in Figure 4.

1. θ, which computes the parity of each of the 320 columns, and XORs the result with two neighbouring columns chosen in a regular pattern;
2. ρ, which bitwise rotates each of the 25 64-bit words by a different number;
3. π, which permutes the 25 64-bit words using a fixed pattern;
4. χ, which bitwise combines along rows using bitwise XOR, NOT and AND operations;
5. ι which XORs a round constant into one 64-bit word of the state;

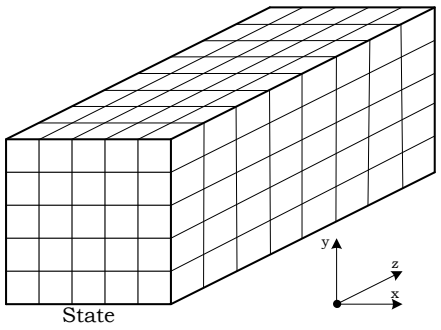

Figure 4: SHA-3 state rapresetnation: it is a 5x5x64 cube, represneting a 1600-bit state. The Keccak f-functon works on this state.

Over the years, many architectures have been developed for SHA-3 functions for high-performance computing [10, 8, 11, 12], where the main techniques adopted to increase throughput are generally unrolling, pipelining and sub-pipelining. Bertoni [8] showed that it is possible to increase the throughput by performing multiple rounds per clock cycles. With this approach, also exploiting the capabilities of synthesizers, it is possible to increase the throughput over the *one round per clock cycle* implementation. However, area efficiency drastically decreases with the unrolling factor, as demonstrated in Table 3. Please note that throughout the document, we will refer to GE as Gate Equivalent, which is used as a complexity measurement unit. In particular, a GE is equivalent to the area of a NAND2x1 (2 input NAND port with drive strength one) port.

The other approaches are pipelining [13] and sub-pipelining [11]: the first one consists in using multiple *f-function* separated by registers, while the second uses an intermediate register inside the *f-function* itself. Both approaches successfully increase the throughput when pipeline registers are kept full. In other words, considering the iterative loop processing of SHA-3, a single message cannot exploit the improvements of these solutions. Also, the combination of unrolling and sub-pipeline, which however leads to a high area increase, could not be exploited for the same reasons [12].

The final choice was to use the *one round per clock cycle* architecture, which has demonstrated to be the most performant and therefore the best candidate for the crypto-tile. The architecture of the implemented module (i.e. the one supporting SHA-3-224, SHA-3-256, SHA-3-384 and SHA-3-512) is depicted in Figure 5. The processing of a new message starts by driving the data multiplexer according to the selected SHA-3

Table 4: SHA2 & SHA-3 Stratix IV implementation results.

	Max. Frequency	LUT	Reg	ALM
SHA2	160 MHz	4200	2378	3057
SHA-3	110 MHz	5363	1610	4053

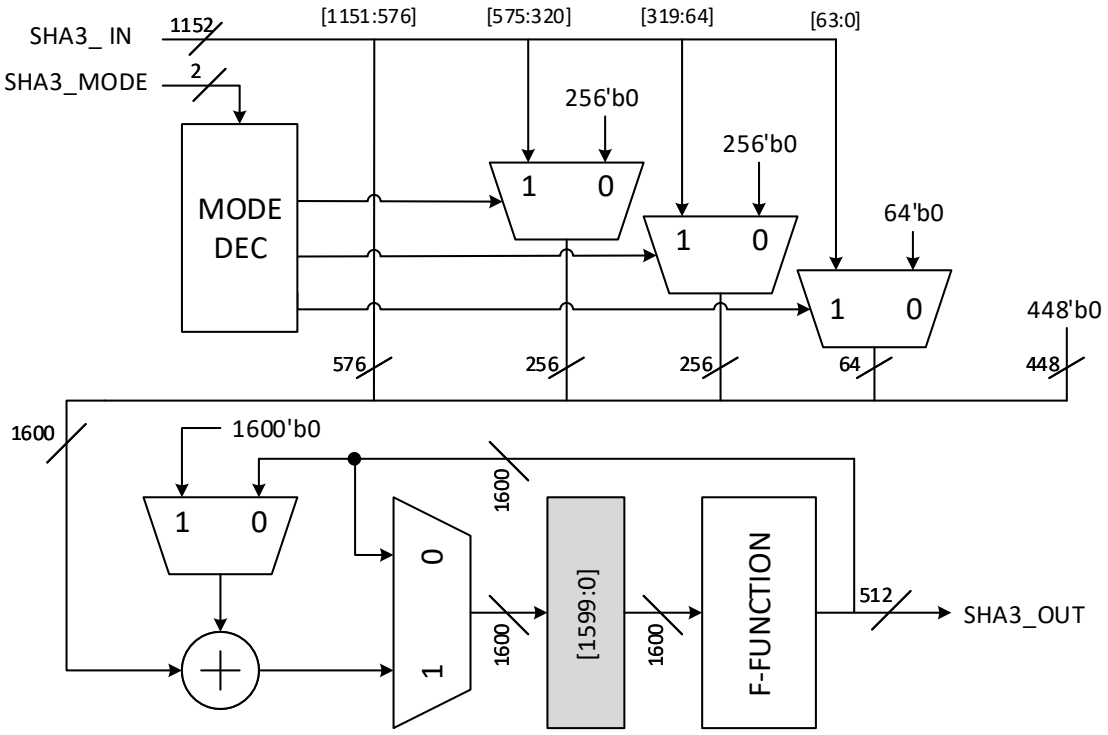

Figure 5: SHA-3 Architecture.

configuration (e.g. SHA-3-256 has 1088-bit wide input). The input data, where the not used bits are set to zero, are then initially XORed with a 1600'b0 signal and saved into the state register.

$$\begin{aligned} state(t = 0) &= message\_block[0] \oplus 1600'b0 \\ &= message\_block[0] \end{aligned} \quad (7)$$

For the 24 following clock cycles the *f-function* is executed on the whole state register enabling the loop controller multiplexers. After the 24 rounds, if the message is not ended, the content on the state register may be immediately XORed with the next new block to start a new absorbing phase. Otherwise, the final hash digest is available on the output port.

2.3. FPGA Verification

The cryptoprocessor has been verified on a Stratix IV (8EP4SGX230KF40C2) FPGA. Both the SHA2 and SHA-3 hash functions have been implemented, in their full hash configuration (224, 256, 384 and 512). The verification set-up consisted of a NIOS II processor, directly connected to the hash engine depicted in Figure 2. The processor executed the test vectors distributed by NIST in the Secure Hash Standard Validation System (SHAVS) [14] and the Secure Hash Algorithm-3 Validation System (SHA3VS) [15].

It compared the outputs with expected results, demonstrating full compliance to SHA2 and SHA-3 standards of the developed hash engines. The implementation results are summarised in Table 4, where the complexity for the SHA2 and SHA-3 accelerator supporting all the possible hash configuration is indicated; please note that no device-dependent optimisation was performed to push the performances to their limit, as FPGA implementation was meant to be an intermediate test. For the same reason, no accurate power dissipation simulation has been carried out. Both these aspects will be addressed during the synthesis on the target technologies, as shown in Section 3.

3. Implementation Results

SHA2 and SHA-3 modules have been synthesised with Design Compiler by Synopsys on the Artisan 7nm TSMC Standard-Cell technology, in the typical case (0.75V 85°C). All the synthesis performed were elaborated, targeting the maximum frequency.

SHA2 results are represented in Table 5 for optimized architecture.

For what concerns the SHA-3, results are presented in Table 6. The throughput is the value that helps to understand the amount of data these circuits can process. It is defined in Equation 8 as the ration between the amount of data processed dived by the amount of time needed for the processing. It is calculated and proposed for each SHA2 and SHA-3 algorithm in Table 7.

$$Throughput\,(Th) = \frac{freq \cdot dim\_in}{clock\_cycle} \tag{8}$$

Where *freq* is the circuit maximum operative frequency, *dim_in* is the input dimension and *clock_cycle* is the circuit latency. The efficiency of the system is defined both for area and power dissipation, as follows:

$$AE = \frac{Th}{Area}\,[Gbps/kGE] \tag{9}$$

$$FoM = \frac{Power\ dissipation}{Th}\,[\mu J/Gs] \tag{10}$$

The Area Efficiency (AE) indicates the throughput we are able to obtain for each kGE of logic used. The Energy Figure of Merit (FoM) indicates the amount of energy (micro joule, μJ) necessary to transmit 10^9 samples (Gs). The literature lacks recent surveys on ASIC implementation of the SHA2 algorithm. To the best of the author knowledge, this is the first contribution that fully details all the SHA2 and SHA-3 operative modes on the 7nm technology. Literature is populated with many FPGA implementations of SHA2 algorithm, like [16], but works on such advanced silicon technology are not currently available. Even though it is not possible to make a fair comparison, two recent works implemented the SHA2-256 mode on previous technological nodes. In [2], [17] we presented results, but limited to SHA2 core only, on a 45nm technology.

In [2], we presented results on the same SHA2 core on a 45nm technology. In [18] an implementation of the SHA2-256 algorithm optimised for high performances is presented on a 14nm technology, with a maximum operative frequency of 1.53 GHz. Thus the estimated throughput is 6,22 Gbps (assuming 256-bit inputs and 67 clock cycles latency). Finally, in [17], a customised version of our core is presented in a 7nm technology implementation, but just for the SHA2-256 core.

Being all the reported results related to different technological process, from different silicon foundries and with process corners (voltage, temperature, ...) not always specified, it is not possible to carry out a fair comparison. However, we can compare the reported values considering a $\sqrt{2}$ scaling factor for the frequency (thus for the throughput) for each technological step. Considering that, we can observe throughput values on the 7nm technology vastly exceeds the one available in the literature. For the SHA2 256, we obtain 33.24 Gbps against 6,22 Gbps [18] in the literature (high performance targeted synthesis), with just one technological node difference.

For what concerns SHA-3, although its specification has been released recently, there are already contributions documenting implementation on various technological nodes. However, up to the author knowledge, no literature contribution is available on the full suite of SHA-3 operational modes. Nevertheless, no contribution to the 7nm technology implementation is available. Following the consideration done for the SHA2, we report the most recent contributions. In [19] and [20] a report on a lightweight SHA-3 implementation is given for Xilinx FPGAs, the most advanced is carried out on the Xilinx Kintex 7 device (28 nm technology) at 200MHz frequency, reaching a throughput of 5.2 Gbps. There are various contributions based on previous node technologies, like [21], which reports on SHA3 candidates (included Keccak) on 130nm. The most advanced contribution we considered in this direction is included in [12], where an area optimised 512 version of the SHA-3 is documented, reporting synthesis results on a 65 nm technology. They obtained a circuit with a complexity of 105 kGE, running at 1GHz, with a throughput of 48 Gbps and an area efficiency of 0.45 $\frac{Gbps}{kGE}$.

Due to the lack of 7nm implementation, also here a fair comparison can not be conducted. For the sack of providing an estimation, we can compare the 512 operational modes, where our proposed work achieves 115.20 Gbps, against the 48 Gbps of [12]; for what concerns complexity the circuit proposed in [12] embeds two *f-function* in the iteration loop, each with two pipeline stage; on the contrary, our proposed solution employs only one *f-function* with a single pipeline stage. Of course, thanks to the high performance of the 7nm technology, we did not consider it necessary to push even more the performances. This results obviously in an advantage in terms of circuit complexity (expressed in kGE): our proposed solution is less than one third

Table 5: SHA2 Implementation - 7nm ASIC Technology.

Operation	Circuit Area [kGE]	Max. Frequency [MHz]	Estimated Power [mW]
224	15.43	5150	13.43
256	15.45	5150	13.45
384	28.28	4600	22.56
512	29.93	4850	24.66
256-224	15.47	5150	13.47
384-256	31.33	4350	21.47
384-224	31.14	4350	21.67
512-384	30.32	4350	24.97
512-256	31.26	4350	21.47
512-224	31.35	4350	21.44
384-256-224	31.19	4350	21.46
512-256-224	31.42	4350	21.51
512-384-224	31.62	4350	21.68
512-384-256	31.92	4350	21.69
512-384-256-224	31.79	4350	21.70

Table 6: SHA-3 Implementation - 7nm ASIC Technology.

Operation	Circuit Area [kGE]	Max. Frequency [MHz]	Estimated Power [mW]
224	31.27	5000	24.96
256	31.55	5000	25.29
384	31.36	5000	25.07
512	30.74	5100	25.67
256-224	31.65	5000	25.19
384-256	31.93	5000	24.03
384-224	32.47	5000	24.80
512-384	32.17	5100	27.54
512-256	31.85	5000	26.18
512-224	32.11	5000	25.73
384-256-224	32.21	5000	25.41
512-256-224	32.33	5000	26.33
512-384-224	32.21	5000	25.58
512-384-256	33.07	5000	23.18
512-384-256-224	33.43	5000	25.29

smaller respect to the one proposed in [12].
Therefore, if we compare the area efficiency of the two circuits, we observe that in our work, the area efficiency is almost ten times higher (3.75 against 0.45 $\frac{Gbps}{kGE}$). The same considerations apply to the circuit proposed and documented in [22], demonstrating again that the obtained performances are aligned with top-performant results available in the literature.

4. Conclusions

In this work, we proposed a Hash Engine, fully supporting SHA2 and SHA-3 algorithms. It has been verified on a Stratix IV FPGA with official NIST test vectors and synthesised on the 7nm TSMC technology. It will be employed in the Crypto-tile of the European Processor Initiative, hardware accelerating all SHA-based cryptographic functions.
Our work is the first contribution available in 7nm technology. Thus it has not been possible to carry out a fair comparison with state-of-the-art. Looking at the

Table 7: SHA2 & SHA-3 Cores Throughput.

Operation	Latency [Clk cycles]	Throughput [Gbps]	Area Efficiency [Gbps/kGE]	Energy FoM [uJ/Gs]
SHA2 224	67	33.24	2.15	404.03
SHA2 256	67	33.24	2.15	404.63
SHA2 384	83	53.67	1.90	420.34
SHA2 512	83	53.67	1.79	459.47
SHA-3 224	25	230.40	7.36	108.33
SHA-3 256	25	217.60	6.90	116.22
SHA-3 384	25	166.40	5.31	150.66
SHA-3 512	25	115.20	3.75	222.83

area efficiency (Gbps/kGE) we observe values aligned with state of the art, while for what concerns raw performances, it offers unprecedented throughputs to users: more than 50 Gbps for the SHA2 and more than 230 Gbps for the SHA-3, with power dissipation in the order of 13-26 mW. The performance reached overcome, in the absolute value, state-of-the-art works, and can be optimised even more with performance optimised architecture.

The work done for SHA-3, the most advanced algorithm in this contribution, will be the baseline for the future development of Shake algorithm support, to be able to run post-quantum cryptography algorithms like Newhope.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826647 (European Processor Initiative).

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] E. Consortium, EPI website.
URL `https://www.european-processor-initiative.eu/`

[2] L. Baldanzi, L. Crocetti, S. DI MATTEO, L. Fanucci, S. Saponara, H. Patrice, Crypto accelerators for power-efficient and realtime on-chip implementation of secure algorithms, in: IEEE ICECS 2019, IEEE, 2019, pp. 1–4.

[3] C. Deng, B. Wang, L. Liu, M. Zhu, Y. Wu, H. Li, S. Yin, S. Wei, A 60 Gbps-level coarse-grained reconfigurable cryptographic processor with less than 1 W power, IEEE Transactions on Circuits and Systems II: Express Briefs (2019).

[4] E. Barker, W. Barker, W. Burr, W. Polk, M. Smid, et al., Recommendation for key management: Part 1: General, National Institute of Standards and Technology, Technology Administration, 2006.

[5] M. Abdalla, T. E. Bjørstad, C. Cid, B. Gierlichs, A. Hülsing, A. Luykx, K. G. Paterson, B. Preneel, A.-R. Sadeghi, T. Spies, et al., Algorithms, key size and protocols report, ECRYPT–Coordination & Support Action (2016).

[6] M. Stevens, P. Karpman, T. Peyrin, Freestart collision for full SHA-1, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2016, pp. 459–483.

[7] Q. H. Dang, Secure hash standard, Tech. rep. (2015).

[8] G. Bertoni, SHA-3 implementation v3.1.
URL `https://keccak.team/`

[9] M. J. Dworkin, SHA-3 standard: Permutation-based hash and extendable-output functions, Tech. rep. (2015).

[10] B. Baldwin, A. Byrne, L. Lu, M. Hamilton, N. Hanley, M. O'Neill, W. P. Marnane, FPGA implementations of the round two SHA-3 candidates, in: 2010 International Conference on Field Programmable Logic and Applications, IEEE, 2010, pp. 400–407.

[11] M. Sundal, R. Chaves, Efficient FPGA implementation of the SHA-3 hash function, in: 2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), IEEE, 2017, pp. 86–91.

[12] M. M. Wong, J. Haj-Yahya, S. Sau, A. Chattopadhyay, A new high throughput and area efficient SHA-3 implementation, in: 2018 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, 2018, pp. 1–5.

[13] H. E. Michail, L. Ioannou, A. G. Voyiatzis, Pipelined SHA-3 implementations on FPGA: Architecture and performance analysis, in: Proceedings of the Second Workshop on Cryptography and Security in Computing Systems, 2015, pp. 13–18.

[14] L. E. Bassham III, T. A. Hall, The secure hash algorithm validation system (SHAVS), NIST Information Technology Laboratory (2004).

[15] S. S. Keller, L. E. Bassham III, The secure hash algorithm 3 validation system (SHA3VS) (2016).

[16] I. Algredo-Badillo, C. Feregrino-Uribe, R. Cumplido, M. Morales-Sandoval, FPGA-based implementation alternatives for the inner loop of the secure hash algorithm SHA-256, Microprocessors and Microsystems 37 (6-7) (2013) 750–757.

URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84886894612&doi=10.1016%2fj.micpro.2012.06.007&partnerID=40&md5=63b3760fe8886ef09201f1763d8db60e

[17] L. Baldanzi, L. Crocetti, F. Falaschi, M. Bertolucci, J. Belli, L. Fanucci, S. Saponara, Cryptographically secure pseudo-random number generator ip-core based on SHA2 algorithm, Sensors 20 (7) (2020) 1869.

[18] X. Zhang, W. Ruizhen, M. Wang, L. Wang, A high-performance parallel computation hardware architecture in ASIC of SHA-256 hash, in: 2019 21st International Conference on Advanced Communication Technology (ICACT), IEEE, 2019, pp. 52–55.

[19] B. Jungk, M. Stöttinger, Serialized lightweight SHA-3 FPGA implementations, Microprocessors and Microsystems 71 (2019).
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85070611757&doi=10.1016%2fj.micpro.2019.102857&partnerID=40&md5=22f774a919f9c788edb0f5d238658f4e

[20] A. Alzahrani, F. Gebali, Multi-core dataflow design and implementation of Secure Hash Algorithm-3, IEEE Access 6 (2018) 6092–6102.

[21] M. Srivastav, X. Guo, S. Huang, D. Ganta, M. Henry, L. Nazhandali, P. Schaumont, Design and benchmarking of an ASIC with five SHA-3 finalist candidates, Microprocessors and Microsystems 37 (2) (2013) 246–257.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84875928648&doi=10.1016%2fj.micpro.2012.09.001&partnerID=40&md5=56adf0de7f3a76ee5a5fa99579cfc008

[22] F. K. Gürkaynak, K. Gaj, B. Muheim, E. Homsirikamol, C. Keller, M. Rogawski, H. Kaeslin, J.-P. Kaps, Lessons learned from designing a 65nm ASIC for evaluating third round SHA-3 candidates, in: Third SHA-3 Candidate Conference, 2012, pp. 1–22.