# PPE-ARX: Area- and Power-Efficient VLIW Programmable Processing Element for IoT Crypto-Systems

Mohamed El-Hadedy*‡, Xinfei Guo†, Mircea R. Stan†, Kevin Skadron‡
*Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801
†Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904
‡Department of Computer Science, University of Virginia, Charlottesville, VA 22904
Email: {mea4c, xg2dt, mircea, skadron}@virginia.edu

***Abstract*—In this paper, we propose a novel programmable processing element (PPE) for various cryptographic systems that can be used in IoT applications. The design enables the programmability, thus supporting a wide range of bit-widths (such as 16, 32, and 64). It employs a very long instruction word (VLIW) architecture with an instruction set and memory hierarchy specialized for crypto-processing. Both FPGA and ASIC implementations demonstrate that the design utilizes a very tiny area and consumes very low power. For example, it takes only 227 slices for FPGA implementations to include 512-byte instruction and coefficient memory along with the computational unit by achieving a maximum clock frequency of 250 MHz. For ASIC implementation (in 28/32nm technology), the design takes only 0.15$mm^2$ of silicon area and consumes only 34.5$\mu W$ of total power while achieving a maximum frequency of 952MHz. To evaluate the effectiveness of the design in a larger system, we implement Blue Midnight Wish (BMW) hash function with the PPE. Compared to the previous BMW-512 implementation which stores the intermediate coefficients of the BMW-512 in 2048 bytes, the proposed design just uses 512 bytes. Meanwhile, we reduce the instruction memory size from 4864 bytes to 1792 bytes.**

***Index Terms*—IoT, PPE, FPGA, ASIC, Crypto, BMW**

## I. Introduction

Over the next few years, smart "things" and wearable devices will be part of the same ecosystem. The Internet of Things (IoT) will become an important asset in the business intelligence; it will provide an integrated view of the production environments, as well as it will tighten monitoring and control domains typically managed as independent worlds [1]. Control systems and human operators will reach a significantly stronger level of interaction. However, new operational scenarios will require security and privacy, which will be no more an option on smart things. The attack to the Target company through the HVAC (Heating, Ventilating and Air Conditioning) system shows that smart and controlled devices are significant weak points that must be protected from hackers of any kind [2]. So far, security facilities for IoT have been inadequate for this aim. IoT architectures dedicate most of their limited energetic resources to guarantee operational continuity for elementary services, mostly sensor signal sampling, transmission and basic computation. Security is thus simple and ineffective against targeted attacks.

There are numerous applications of IoT applications which have strict security requirements, such as the pacemaker for medical implanting. The pacemaker is a tiny device generating electrical impulses, delivered by electrodes that contract the heart muscles to regulate the heartbeat [3]. The pacemaker comes without security primitives that secure the data transfer between the internal components and the device parts. Without a strong security mechanism in the pacemaker, the patient's life is simply in danger, the hacker can send a deadly signal to the device to increase the heart rate of the patient to unexpected level, and cause a serious damage to the heart muscle [4], [5], [6]. Furthermore, wearable gadgets collecting body parameters to track the human activities most notably smartwatch and fitbit need to be secure enough to prevent hackers to catch sensitive data and use it against persons for malicious aims [7], [8]. Securing these types of small gadgets can be achieved by employing cryptographic scientific theories and technical primitives. In general, these security solutions need to be adaptive to any IoT applications while consuming few resources (either silicon area or power consumptions).

In this paper, we present the design and analysis of a novel type of area- and power-efficient VLIW-based Programmable Processing Element (PPE) for use in IoT cryptographic systems. The proposed PPE can be used to implement all the crypto-systems that are based on ARX (Adding, Rotation and Xor'ing) operations and supports multiple data-path widths. In addition, the PPE is designed to be modular, for use in more complex systems. The presented PPE can be used as cryptographic coprocessor for secure communications on resource-constrained devices, and in particular, IoT applications. The key features of this VLIW PPE are 1) low power dissipation, 2) small area, 3) reconfigurability. The PPE architecture is prototyped in an FPGA implementation on a Xilinx Virtex-5 FPGA, and also implemented with a Top-down ASIC flow in a 28/32 nm technology node to demonstrate its potentials of being used as an IP macro for more complex designs.

As a proof of concept, we employ the PPE to implement the

A part of this work was done while Dr. Mohamed EL-Hadedy was a Research Associate with the Department of Computer Science at the University of Virginia.

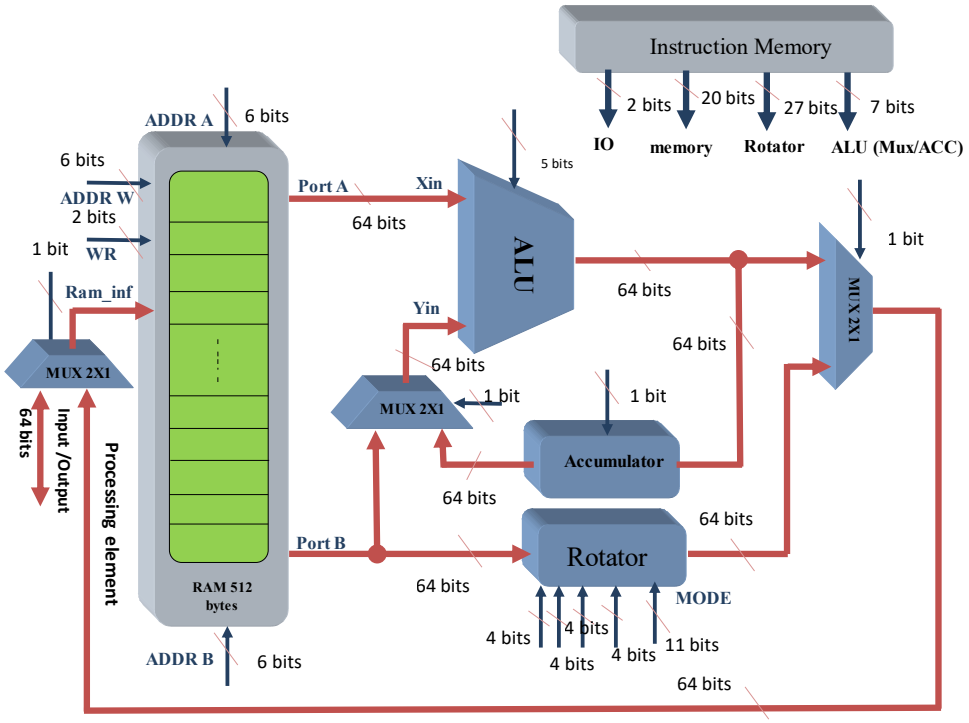



**Fig. 1:** Top level architecture of the proposed programmable processing element (PPE)

Blue Midnight Wish (BMW) hash function, which was one of the second-phase SHA-3 competition candidates [9]. It is also one of the selected hash algorithms for the crypto-currency QuarkCoin [10].

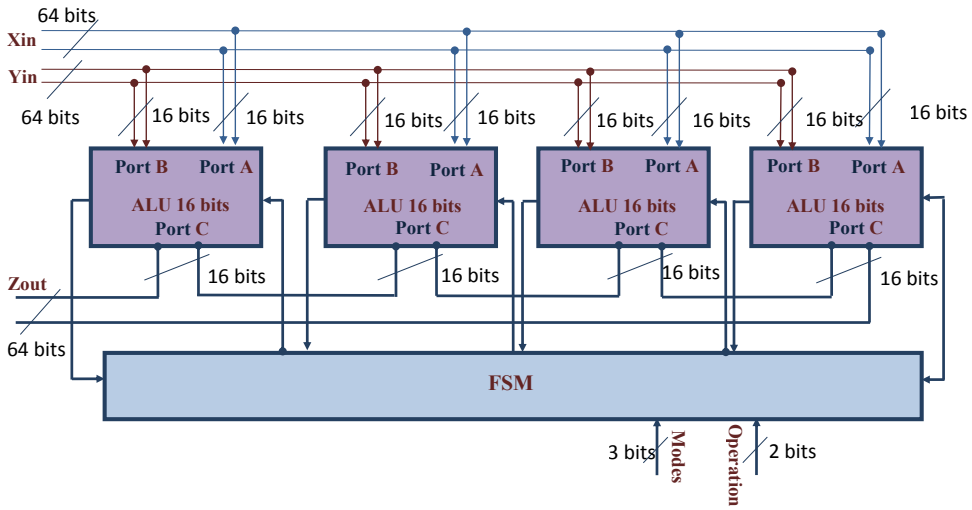
The rest of the paper is organized as follows. Section II discusses the previous work. The detailed architecture of the proposed PPE is introduced in Section III. Section IV presents the evaluation results on FPGA, followed by ASIC evaluation results in Section V. Section VI shows an example of implementing such PPE is bigger systems like BMW. The paper is concluded in Section VII.

## II. Related Work

There are various types of architectures for compact cryptosystems that were proposed in literature. [11] proposed a CGRA coprocessor CoARX which supports diverse ARX-based cryptographic primitives, the implementation was evaluated with 90nm, and achieved 700MHz of core performance. But in that work, they focused on performance-driven applications, and can't be easily adapted into resource-constrained applications, like IoT. Different from this work, our proposed architecture is targeting area and power-constrained IoT applications, and the idea is to reuse the limited on-chip resources to implement necessary security functions while keeping metrics optimal.

In [12], an architecture named "Blake" was presented as one of the SHA competition's candidates. It was implemented in just 56 slices with two block memories and can achieve 115 Mbps of throughput. [13] introduced a low area processor being used for implementing different versions of BMW (BMW-256, BMW-512). BMW-256 is implemented in just 51 slices, achieving a throughput of 68.71 Mbps, and BMW-512 in just 105 slices achieving a throughput of 112.18 Mbps. Both BMW implementations need two block memories to store the hash function internal values as well as the instruction sets. Eve though above two previous implementations offer a small area on the FPGA platform with reasonable throughput, the entire program must be loaded at once (overlapping of compute with streaming in of new instructions is not supported), thus requiring a much larger instruction memory. Different from their approaches, our approach avoids this by allowing only a much smaller instruction memory and significantly reduces the size of the processing element. Furthermore, their processors cannot process different data sizes with the same design. While our proposed architecture has the flexibility and programmability of implementing multiple data sizes.

Several cryto-processors designed for IoT security applications were also presented recently. In [14], a multi-mode architecture which can support three different public-key cryptography (PKC) engines was proposed and demonstrated with silicon. Another multi-mode reconfigurable processor for AES applications was introduced in [15], the novelty of that work was to simplify the control logic and remove the pipelines to reduce the gate count. A novel SHA-3 implementation using instruction set extension based on a 32-bit LEON3 processor was proposed in [16], and the design is validated with KECCAK algorithm. [17] presented an efficient and flexible dual-field ECC processor using the hardware-software approach, while it focused on Elliptic curve cryptography only. [18] extended the work and applied on sensor nodes. [19] proposed a compact prime field (GF(p)) elliptic curve digital signature algorithm (ECDSA) engine suitable for use in IoT




ALU can process different word sizes as one 64-bit, two 32-bit, or four 16-bit operations per clock cycle

**Fig. 2:** Programmable architecture of the ALU

| Arithmetic and Logic Unit Truth Table | | | | | | | Operation |
|---|---|---|---|---|---|---|---|
| Acc | | Modes | | | Operations | | |
| Inst (6) | Inst (5) | Inst (4) | Inst (3) | Inst (2) | Inst (1) | Inst (0) | |
| Reset the accumulator | Accumulate | 1 | 1 | 1 | 0 | 0 | XOR (ALU 64 bits) |
| | | 1 | 1 | 1 | 0 | 1 | ADD (ALU 64 bits) |
| | | 1 | 1 | 1 | 1 | 0 | INC (ALU 64 bits) |
| | | 1 | 1 | 1 | 1 | 1 | SUB (ALU 64 bits) |
| | | 0 | 0 | 0 | 0 | 0 | XOR (ALU 32 bits) |
| | | 0 | 0 | 0 | 0 | 1 | ADD (ALU 32 bits) |
| | | 0 | 0 | 0 | 1 | 0 | INC (ALU 32 bits) |
| | | 0 | 0 | 0 | 1 | 1 | SUB (ALU 32 bits) |
| | | 1 | 0 | 1 | 0 | 0 | XOR (ALU 16 bits) |
| | | 1 | 0 | 1 | 0 | 1 | ADD (ALU 16 bits) |
| | | 1 | 0 | 1 | 1 | 0 | INC (ALU 16 bits) |
| | | 1 | 0 | 1 | 1 | 1 | SUB (ALU 16 bits) |
| | | 0 | 1 | 1 | 0 | 0 | XOR (ALU 48 bits) |
| | | 0 | 1 | 1 | 0 | 1 | ADD (ALU 48 bits) |
| | | 0 | 1 | 1 | 1 | 0 | INC (ALU 48bits) |
| | | 0 | 1 | 1 | 1 | 1 | SUB (ALU 48 bits) |

**TABLE I:** The truth table of the Arithmetic Logic Unit (ALU)

applications. In summary, all these work mentioned above provide various flavors of IoT security engine and clearly demonstrated the high demands of securing IoT devices with a low cost in terms of hardware. Our proposed PPE differs from previous work in several aspects. Firstly, the architecture is able to target various cryto-systems that are based on ARX operations. Secondly, the design supports a wide range of bit-width, thus a wide range of applications. Last but not least, most of the previous work were prototyped on FPGA only, our design has been implemented on both FPGA and ASIC platforms, and this provides a comprehensive evaluations. The results have demonstrated very low cost in terms of area and power consumption.

The proposed PPE is based on a very long instruction word (VLIW) architecture as shown in Fig. 1. It consists of an Arithmetic and Logic Unit (ALU), a Rotator block, an intermediate 512-byte coefficient memory, a 448-byte instruction memories. Multiplexers are used for organizing the data traffic from Input/Output ports and the feedback from the computational part to the coefficient memory. The VLIW architecture enables an efficient way to issue multiple operations in one clock cycle. Changes of control-flow (e.g., branches) are not supported since this is not needed in this design (we examined this for ciphers). The programs are created manually for now, but the limited set of operations and the regular structure of the VLIW architecture make it possible to include a compiler at the software potentially to support high level synthesize. The PPE proposed in this paper implements the ARX engine, and it assumes the input text has already been padded if necessary.

For instance, the mode control bits are 111 and the operation bits are 01 will enable addition of two 64-bit words.

## III. Architecture of The PPE

### A. ALU

Although ripple carry adders RCA is the most area-efficient adder, its worst-case propagation delay can be severe. For

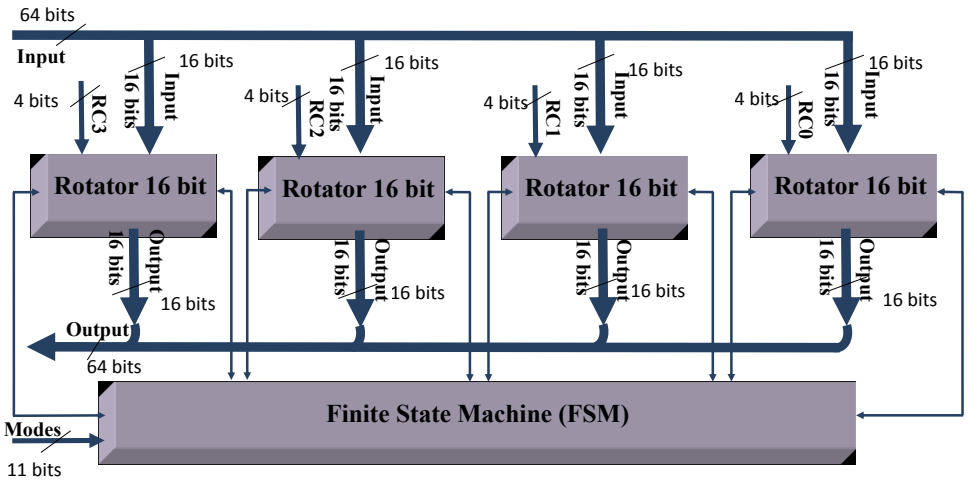



**Fig. 3:** The Rotator

| Operation | Mode | RC3 | RC2 | RC1 | RC0 | Data width |
|---|---|---|---|---|---|---|
| ROR (1 →15) | 00001010101 | 1→F | 1→F | 1→F | 1→F | 64 Bits Mode |
| ROR (16 →31) | 01101010101 | 1→F | 1→F | 1→F | 1→F | |
| ROR (16 →31) | 01101010101 | 1→F | 1→F | 1→F | 1→F | |
| ROR (32 →47) | 10001010101 | 1→F | 1→F | 1→F | 1→F | |
| ROR (48 →63) | 11101010101 | 1→F | 1→F | 1→F | 1→F | |
| ROR (1 →15)<br>ROR (1 →15) | 00011011101 | 1→F | 1→F | 1→F | 1→F | 32Bits Mode |
| ROR (16 →31)<br>ROR (16 →31) | 01111011101 | 1→F | 1→F | 1→F | 1→F | |

**TABLE II:** The truth table of the Rotator

instance, for a 32-bit adder, the delay would be about 63 ns if one assumes a gate delay of 1 ns [20].

We employ a carry lookahead adder (CLA), which calculates the carry signals in advance during adding based on the input bits to increase the computing performance [21].

The Virtex-5 FPGA has LUTs with six inputs and one output; a single LUT can output one bit of data from six-bit input data. If this resource is properly used for the circuit design, it is possible to obtain a higher performing circuit.

We use these capabilities to build the ALU based on the CLA as shown in Fig. 2. The ALU offers three different operations: word XOR, word addition, and subtraction (modulo $2^n$). The ALU can process data either from both coefficient memory ports (Port A and Port B) or the coefficient memory (Port A) and the accumulator register as shown in Fig. 1. In each cycle, the ALU can process 4 data words of 16 bits, 2 data words of 32 bit, or 1 data word of 64 bit based on the mode control bits In each cycle, the ALU can process 4 data words of 16 bits, 2 data words of 32 bit, or 1 data word of 64 bit, based on the value of the control mode bits as shown in Table. I.

### *B. Rotator*

The Rotator consists of four 16-bit sub rotations, controlled by the finite state machine (FSM). as shown in Fig. 3. Each 16 bit Rotator consists of a $64 \times 4$ array of $2 \times 1$ multiplexers, controlled by four 4-bit RC control bits. Based on the value of the mode control bits, the Rotator can rotate to the right four 16-bit words, two 32-bit words, or one 64-bit words. Table. II shows how the proposed Rotator operates. For instance, to rotate the 64 bit input data 12 times to the right, the Mode signal has to be 00001010101, and RC0, RC1, RC2, and RC3 have the same value of 1100 (0xC). On the other hand, if we have four data words of 16 bits, to rotate the first word 12 times to the right, the second word 5 times to the left, the third word 7 times to the right, and the fourth word 10 times to the left, the mode signal is 00000000000, and RC0 = 1100, RC1 = 1011 , RC2 = 0111, and RC3 = 0110.

### *C. Coefficient Memory*

As shown in Fig. 1, the coefficient memory has two output ports of 64-bits (Port A and Port B). There are two address ports, ADDRA and ADDRB, which are used to control the value of Port A and Port B. ADDRW is responsible for writing data into the memory through ram_inf. The memory size is 512 bytes.

### *D. Instruction Memory*

As shown in Fig. 4, The dual-port instruction memory of 448 bytes stores the program that implements the chosen cipher algorithm, expressed as a sequence of VLIW instructions. The instruction memory has two address ports; the first 6-bit port acts as an instruction fetch address. The other 6-bit port specifies a write address, to change the values of the instruction memory. There are two enable bits: one allows read for instruction fetch and the other port is used to change the values of the instruction memory.

This dual-ported design allows a new cipher program being streamed in while the current program progresses. Also it enables an early start on a new program while the rest of the program is still loading. In the case that a program cannot fit entirely into the on-chip memory, the dual-port can still guarantee the seamless processing. The instruction-fetch signal (*ADDR*) works as a pointer moving sequentially and wrapping

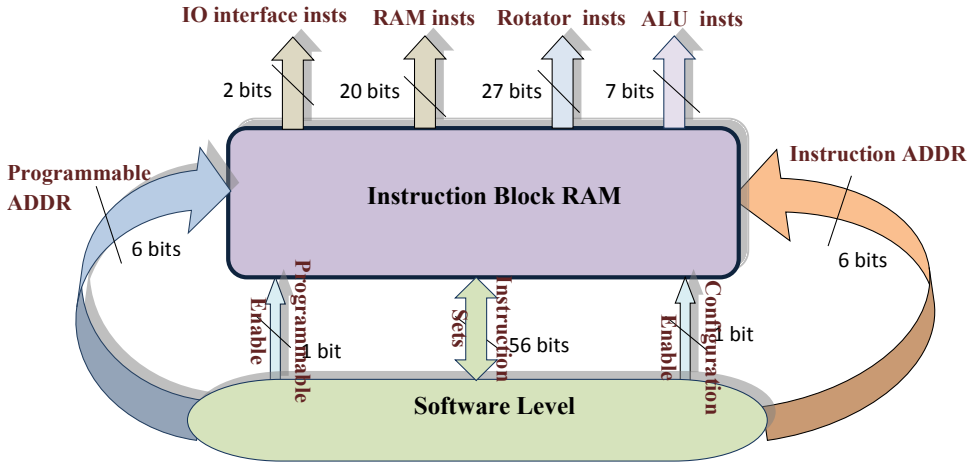
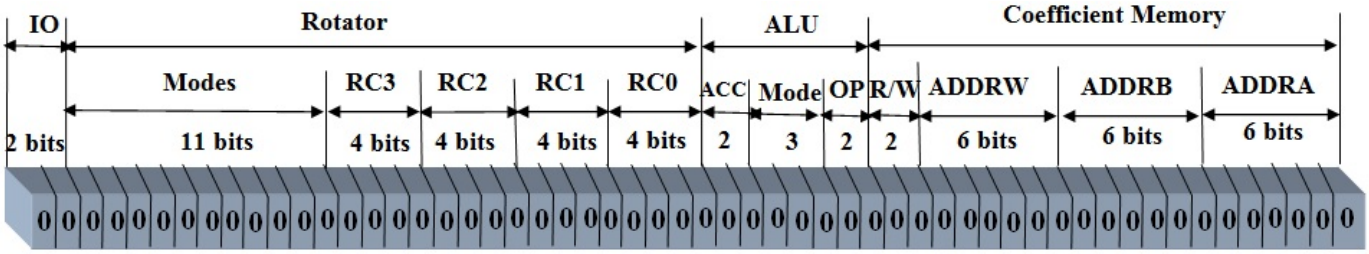



**Fig. 4:** The VLIW instruction memory




ADDRA, ADDRB, and ADDRW are the coefficient-memory address ports. R/W is used to control the coefficient memory input/output ports. OP is used to control the ALU's operations. Mode is used to control the ALU's word sizes. ACC is used to control the accumulator in the ALU. RC0, RC1, RC2, RC3, and Modes are used to control Rotator. IO is used to control the input/output ports

**Fig. 5:** The Instruction Set

around. While this processing is happening, the write ADDR is loading the new code. Because only one read and write are allowed per cycle, conflicts are thus avoided unless the read and write addresses are initialized to the same values, which we usually verify. Fig. 5 shows a snapshot of each row in the instruction memory. Each instruction consists of 2 bits to control the input/output interface, 20 bits for controlling the coefficient memory, 27 bits for controlling the different sections of the Rotator, and 7 bits for controlling the ALU.

## IV. FPGA Implementation of the PPE

The functionality of the proposed PPE was verified on the Xilinx Virtex-5 XC5VLX110 device. The PPE has been described in VHDL and was synthesized using ISE design suite 14.7. The results show that the PPE takes only 227 slices of the FPGA fabric, and this includes the 512-byte coefficient memory, a 448 byte instruction memory, and all the computational units (Rotator and ALU) and control logic. The design can be clocked at 250 MHz. Since the Virtex-5 FPGA has LUTs with six inputs and one output; a single LUT can output one bit of data from six-bit input data. This unique feature could be potentially used to further optimize the performance.

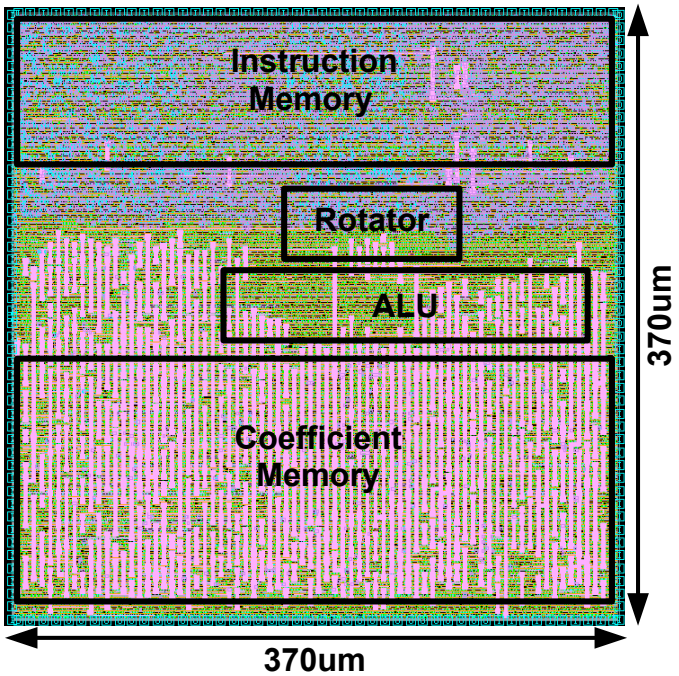



**Fig. 6:** The layout the 64-bit PPE design (core utilization = 80%)

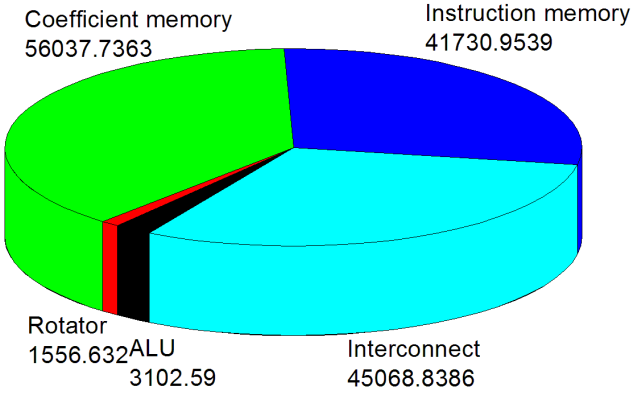



**Fig. 7:** The area breakdown of a 64-bit PPE ($um^2$)

## V. ASIC Implementation of the PPE

As silicon area for IoT devices becomes a huge constraint due to the considerations of size and weight, implementing a lightweight security blocks is crucial. Since the proposed PPE is digital in nature and doesn't need any special SRAM blocks, this makes it very adaptive to the standard ASIC flow. In this work, we implement the design in a Synopsys ASIC flow (from RTL to GDS II) with a 28/32nm design kit [22] from Synopsys. To estimate the extreme case, the design is synthesized on the worst case corner (Slow-Slow, high temperature and nominal voltage) of the technology. The maximum clock frequency is estimated by running timing analysis and timing closure to meet the positive slack requirement for both setup and hold time, and it is about 952MHz. The average power is evaluated with the Prime-time PX tool. The synthesized gate-level net-list is fed into the IC Compiler for place and route (P&R), after which the area information can be got.

Shown in Fig. 6 is the layout after P&R with the core area utilization of 80%. Fig. 7 shows the corresponding area breakdown, the total area is only $0.15mm^2$, with cell area of about $0.1mm^2$ and the rest are interconnects. Although most of the area are from memory (instruction and coefficient), the overall area is very small. With the design being in a modular fashion, it is flexible to be expanded, the interconnect overhead between modules become minimal and the main area will be from the macro itself. The compact design makes it very suitable to be embedded as an IP macro into a larger IoT system which requires security and crypto features. Also the portability and programmability of the design provide huge potentials of being employed in various IoT applications.

Power is critical in IoT devices since these devices are usually powered by battery or through energy harvesting from the environment [28], [29]. The logic (or key function) blocks on these chips are already big "power hunters", thus the margin left for security block is really tiny. Also the leakage power in these devices need to be minimal due to that it directly determines the batter lifetime. Since the proposed PPE is compact and simple compared to other designs (shown in Section III), the leakage power will be reduced. Presented in Table III is the power breakdown of the 64-bit design at the maximum frequency. The total average power is only 34.5uW with a leakage of 11.126 uW. The expected power consumption will linearly increase as the bit width increases. As the first-order estimation, the total power consumption for a 512-bit width will be only about 260uW, which can be further reduced by lowering the voltage and relaxing the clock frequency, which are widely used in IoT systems to meet the SPEC.

**TABLE III:** Average Power Breakdown under Maximum Frequency of 952MHz and Nominal Vdd of 0.95V (uW)

| | Dynamic | Leakage | Total |
|---|---|---|---|
| Combinational Logic | 18.324 | 5.978 | 24.302 |
| Sequential Logic | 5.05242 | 5.148 | 10.2 |
| Total Power | 23.38 | 11.126 | **34.5** |

## VI. Blue Midnight Wish (BMW) Hash Function with The Proposed PPE

Many cryptographic algorithms are ARX based, but the algorithms vary widely in the number and sequence of ARX operations. In cryptography and information security, hash functions are considered the "Swiss army knife". They are used in countless protocols and algorithms. In this paper, we present the hardware implementation of the BMW hash function with the proposed PPE design. In the future, we intend to examine the proposed PPE by implementing wide-range of algorithms such as SHA-1 and SHA-2.

BMW is a wide-pipe Merkle-Damgård hash construction with an unconventional compression function [24]. The non-linearity in BMW is derived from the overlap of modular XOR and addition operations. The BMW-n hash function family contains four instances for n =224, 256, 384, and 512, where n is the size of the hash value. BMW performs four different operations in the hash computation stage: bit-wise logical word XOR, word addition and subtraction, shift operations (left or right), and rotate left operations. The size of a word is 32 bits for BMW-224/256 and 64 bits for BMW-384/512. As shown in Fig. 8, the computation engine of the BMW consists of three sub-functions called $f_0$, $f_1$, and $f_2$, in sequence to generate the chaining value. More details can be found in [30]. The i value is from 0 to 15. Inputs for the function $f_0$ are two arguments. The first argument consists of sixteen 32-bit words, which serve as initial double pipe values $H_0^{(i-1)}$, $H_1^{(i-1)}$,....,$H_{15}^{(i-1)}$. The second arguments consists of sixteen 64-bit words, which represent the input message block: $M_0^{(i)}$, $M_1^{(i)}$,....,$M_{15}^{(i)}$ for BMW-512. The second function $f_1$ takes $f_0$ output values $Q0^{(i)}$, $Q1^{(i)}$, ....., $Q15^{(i)}$ and the output operation of processing the message into the *AddElement* block to produce the second part of the quadruple pipe $Q16^{(i)}$, $Q1^{(i)}$, ....., $Q31^{(i)}$. The final function $f_2$ produces the $H_0^{(i)}$ , $H_1^{(i)}$ , $H_2^{(i)}$ ,..., $H_{15}^{(i)}$ by processing the output of the message, $f_0$, and $f_1$.

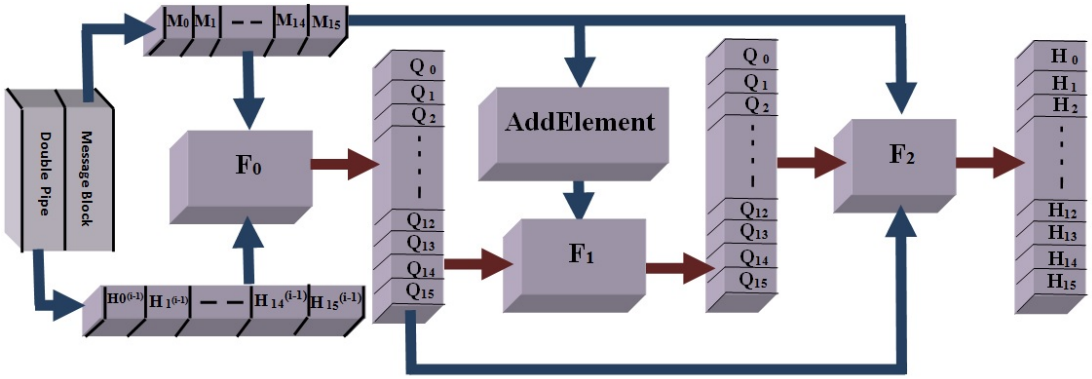
For using the PPE to implement BMW-512 on FPGA, $f_0$, $f_1$, and $f_2$ can be executed in just 1000 cycles. The throughput

**TABLE IV:** BMW-512 performance using the PPE

| Algorithm | FPGA Type | Area(slice) | Frequency (MHz) | Throughput (Mbps) | Memory Blocks | Throughput/Area(slice) |
|---|---|---|---|---|---|---|
| PPE (BMW-512) | XC5VLX110 | 227 | 250 | 256 | — | 1.3 |
| BMW-512 [13] | XC5VLX110 | 105 | 115 | 112.18 | 3 | 1.09 |
| BMW-512 [23] | XC5VLX330T | 9810 | 10.004 | 287 | — | 0.028 |
| BMW-256 [24] | Xilinx Virtex 5 | 1980 | 56 | 5 | — | 0.025 |

**TABLE V:** Comparisons with Published ASIC Results (Both ARX-based and other IoT cryto-architectures)

| Design Name | Technology | Maximum Frequency (MHz) | Total Power | Area | Programmable |
|---|---|---|---|---|---|
| CoARX[11] | 90nm | 700 | - | 95k | Yes |
| BLAKE[12] | 90nm | 532 | 2.93mW | 79k | No |
| Skein-512[25] | 90nm | 251 | - | 43.13k | No |
| Salsa20 4xS-QR[26] | 90nm | 365 | - | 22.81k | No |
| ChaCha 4xS-QR[26] | 90nm | 366 | - | 22.44k | No |
| PKC-based IoT Security Protocols[14] | 90nm | 50 | 15.8mW | 116.3k | No |
| Dual-Field Elliptic Curve Cryptographic Processor[17] | 55nm | 316 | - | 189k | Yes |
| BMW[27] | 65nm | 800 | 1.11mW | 50K | No |
| This Work (BWM with PPE) | 28/32nm | 952 | 120.75uW | 30.2k | Yes |




Input message M is divided to $M_0^{(i)}$, $M_1^{(i)}$,....,$M_{15}^{(i)}$. $H_0^{(i-1)}$, $H_1^{(i-1)}$,....,$H_{15}^{(i-1)}$ are the initial values. $Q0^{(i)}$, $Q1^{(i)}$, ....., $Q15^{(i)}$ are the output of the Bijective function $f_0$. $Q15^{(i)}$, $Q16^{(i)}$, ....., $Q31^{(i)}$ are the output of the expansion function $f_1$. $H_0^{(i)}$, $H_1^{(i)}$,....,$H_{15}^{(i)}$ are the final hash value

**Fig. 8:** BMW Hash Compression Function

of the design is given in Equation 1. As summarized in Table IV, the throughput/area ratio (area efficiency) for the proposed PPE in BMW-512 is much higher than three of the closest previous implementations. Compared to [13] which used 3 block memories for storing the intermediate coefficients and the instruction sets, we don't use any block memories at all. This contributed directly to the huge area saving and throughput improvement. The BMW design in [23] needs to store the intermediate coefficients of the BMW-512 in 2048 bytes, while we only uses 512 bytes. Meanwhile, they dedicate 4864 bytes for storing the instructions, while we use in total just 1792 bytes. In [23] and [24], they implemented BMW-512 on the FPGA without using the BRAMs but with much larger area (slices) than the proposed design. Overall, the proposed implementation of BMW using PPE achieved the best throughput/area.

$$Throughput = \frac{Number\ of\ input\ bits \times Max frequency}{Number\ of\ clock\ cycles\ per\ block} \quad (1)$$

Shown in Table V is the comparison between the BWM algorithm implemented with the proposed PPE against other ASIC implementations of cryto-processors for ARX-based functions and for IoT security in general. Since most of previous ARX-based implementations focus on performance-driven applications, power consumptions were missing in their work. Overall, our proposed design outperforms most of the designs in terms of the area (we pick # of gate to make fair comparisons across different technology nodes), only two designs presented in [26] employ lower area than us, but they don't support programmbility and variable bit-widths. In terms of performance, our design outperforms all the rest, one reason is that we implement with the more advanced technology node than others, but even we do a simple first-order scaling, the performance is at least comparable to other designs. The proposed design consumes lowest total power among all the designs as well. In summary, the proposed processing element is small and ultra low power (8.4nW/bit) while keeping the performance optimal. Thus it has huge potentials of being

embedded in IoT applications where security is becoming crucial.

## VII. Conclusions and Future Work

In this paper, we presented both the FPGA and ASIC implementations of a novel area- and power-efficient, programmable VLIW processing element based on ARX (add, rotate, xor) operations to be used in IoT cryptographic systems. We have implemented the ARX engine and demonstrated its effectiveness for the BMW-512 hash function. The results show great improvement compared to previous implementations in terms of throughput, area (number of slices or silicon area), frequency, and the throughput/area.

As the future work, the effectiveness of this PPE will be evaluated with more IoT cryptographic systems. Also, a compiler for the proposed architecture is expected to be proposed.

## Acknowledgements

This work was supported in part by NSF Grant No. CDI-1124931 and by the Center for Future Architectures Research (C-FAR), one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA. Furthermore, this work was supported by Azure Research Hardware Program, Grant Award No. CRM0518510.

## References

[1] Gi-chur Bae and Kyung-wook Shin. An Efficient Hardware Implementation of Lightweight Block Cipher Algorithm CLEFIA for IoT Security Applications. *Journal of the Korea Institute of Information and Communication Engineering*, 20(2):351–358, 2016.

[2] Kaveh Paridari et al. Cyber-Physical-Security Framework for Building Energy Management System. In *2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems (ICCPS)*, pages 1–9. IEEE, 2016.

[3] Daniel Halperin et al. Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses. In *2008 IEEE Symposium on Security and Privacy*, pages 129–142. IEEE, 2008.

[4] C. Osborne. Samartwatch Security Fails to Impress: Top Devices Vulnerable to Cyberattach, 2015.

[5] B. Cole. How Secure are Smartwatches? Not Very, 2015.

[6] Steven J Johnston et al. Recommendations for securing Internet of Things devices using commodity hardware. *WF-IoT*, 2016.

[7] Kat Austen. The trouble with wearables. *Nature*, 525(7567):22, 2015.

[8] Charles Walter et al. Toward Predicting Secure Environments for Wearable Devices. In *Proceedings of the 50th Hawaii International Conference on System Sciences*, 2017.

[9] Meltem Sönmez Turan et al. Status report on the second round of the SHA-3 cryptographic hash algorithm competition. *NIST Interagency Report*, 7764, 2011.

[10] J. Stevenson. *Bitcoins, litecoins, what coins?: A global phenomenon*. Stevenson, J., 2013.

[11] Khawar Shahzad et al. CoARX: a coprocessor for ARX-based cryptographic algorithms. In *Proceedings of the 50th Annual Design Automation Conference*, page 133. ACM, 2013.

[12] Nuray At et al. Compact Hardware Implementations of ChaCha, BLAKE, Threefish, and Skein on FPGA. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 61(2):485–498, 2014.

[13] Mohamed El-Hadedy et al. Area efficient processing element architecture for compact hash functions systems on VIRTEX5 FPGA platform. In *AHS, 2011 NASA/ESA Conference on*, pages 240–247. IEEE, 2011.

[14] Cheng-Rung Tsai et al. A 1.96 $mm^2$ low-latency multi-mode crypto-coprocessor for PKC-based IoT security protocols. In *Circuits and Systems (ISCAS), 2015 IEEE International Symposium on*, pages 834–837. IEEE, 2015.

[15] Ning Ma et al. A hierarchical reconfigurable micro-coded multi-core processor for iot applications. In *Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC), 2014 9th International Symposium on*, pages 1–4. IEEE, 2014.

[16] Yi Wang et al. Fpga-based sha-3 acceleration on a 32-bit processor via instruction set extension. In *Electron Devices and Solid-State Circuits (EDSSC), 2015 IEEE International Conference on*, pages 305–308. IEEE, 2015.

[17] Zilong Liu et al. An Efficient and Flexible Hardware Implementation of the Dual-Field Elliptic Curve Cryptographic Processor. *IEEE Transactions on Industrial Electronics*, 2016.

[18] Zhe Liu et al. On emerging family of elliptic curves to secure internet of things: Ecc comes of age. *IEEE Transactions on Dependable and Secure Computing*, 2016.

[19] T. YalÃğin. Compact ECDSA engine for IoT applications. *Electronics Letters*, 52(15):1310–1312, 2016.

[20] B. Gadamsetti and A.D. Singh. Current sensing completion detection for high speed and area efficient arithmetic. In *APCCAS*, pages 240–243, Dec 2010.

[21] Sean Kao et al. A 240ps 64b carry-lookahead adder in 90nm CMOS. In *ISSCC*, pages 1735–1744. IEEE, 2006.

[22] Synopsys. 32/28nm Generic Library for Teaching IC Design.

[23] B.Baldwin, N. Hanley, M. Hamilton, L. Lu, A. Byrne, M. ONeill, and W. P. Marnane. FPGA Implementations of the Round Two SHA-3 Candidates. In *Proceedings of the NIST SHA-3 Conference*.

[24] M. El-Hadedy et al. Low area FPGA and ASIC implementations of the hash function "Blue Midnight Wish-256";. In *2009 International Conference on Computer Engineering Systems*, pages 10–14, Dec 2009.

[25] M. Knezevic et al. Fair and Consistent Hardware Evaluation of Fourteen Round Two SHA-3 Candidates. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 20(5):827–840, May 2012.

[26] L. Henzen et al. VLSI hardware evaluation of the stream ciphers Salsa20 and ChaCha, and the compression function Rumba. In *2008 2nd International Conference on Signals, Circuits and Systems*, pages 1–5, Nov 2008.

[27] Xu Guo et al. On the impact of target technology in sha-3 hardware benchmark rankings. *IACR Cryptology ePrint Archive*, 2010:536, 2010.

[28] Uming Ko. Ultra-low power SoC for wearable & IoT. In *VLSI-TSA*, pages 1–1. IEEE, 2016.

[29] Mihai Sanduleanu and Ibrahim Abe M Elfadel. Ultra low power integrated transceivers for near-field IoT. In *DAC*, page 143. ACM, 2016.

[30] D. Gligoroski et al. Blue Midnight Wish. In *Proceedings of The First SHA-3 Candidate Conference*, Feb 2009.