A VLSI Implementation of the PRESENT Cipher for System-on-Chip Applications

J. G. Pandey and Abhijit Karmakar

Abstract—**Fundamental essence of internet-of-things (IoT) infrastructure is based on the security of communicated data. Here, lightweight cryptography plays a vital role in resource-constrained environments and applications. In this paper, we proposed a resource-efficient VLSI architecture for the PRESENT lightweight block cipher algorithm. The architecture is based on 8-bit datapath. An ASIC implementation of the proposed architecture is done by in SCL 0.18 µm technology. Here, the chip operates on 3.3 V and the core works at 1.8V. Gate equivalent (GE) of the proposed architecture is 1608 GEs. At 100 MHz operating frequency, total power consumption of the chip is 2.45 mW, where, the dynamic component is 2.452 mW and the static one is 3.26 µW. A throughput of 16.326 Mbps, energy 1.202 µJ, energy/bit 0.150 µJ /bit, and 0.010 efficiency is obtained. Area of chip is 1.55 mm2.**

***Keywords*— Lightweight cryptography; PRESENT; block cipher; VLSI architectures; ASIC.**

I. Introduction

The recent proliferation of the internet-of-things (IoT) enables resource-constrained devices to be interconnected through Internet. These applications heavily rely on the communicated data that can be between human-to-machines or machines-to-machines and any their combinations. Accordingly, application and system developers are more focused towards design of a new class of applications and associated intellectual-properties (IPs) and system-on-chips (SoCs) that are optimized for resource, latency, power and bandwidth design metrics. To secure electronic data, cryptography plays a vital role. In the context of lightweight cryptography, ISO/IEC 29192-2 has standardized symmetric block cipher algorithm PRESENT in the year 2012 [1], [2].

In this paper we propose a VLSI architecture for the PRESENT lightweight block cipher algorithm. The architecture works on the 64-bit input and of 80-bit user key and provides 64-bit ciphertext. The architecture processes 8-bit data at a time and provides ciphertext in 47 clock cycles. An ASIC implementation of the proposed architecture is done in SCL 0.18 µm technology. Here, the chip operates on 3.3V and the core work at 1.8V. Gate equivalent (GE) of the proposed architecture is 1608 GEs. The total power consumption of the chip is 2.45 mW, where, the dynamic component is 2.452 mW and the static one is 3.26 µW. With 100 MHz operating frequency the obtained a throughput 16.326 Mbps, energy 1.202 µJ, energy/bit 0.150 µJ /bit is obtained. The efficiency of the design is 0.010. Area of chip is 1.55 mm^2.

II. The PRESENT Algorithm

The PRESENT algorithm accepts a block size of 64-bit with 80/128-bit user keys. The algorithm is based on the SP-network and consists of 31 rounds [2]. Each of the 31 rounds consist of an XOR operation, which is required to introduce a round key K_i for $0 \le i \le 31$ in which K_{31} is used for post-whitening operation. Further, there is a linear bitwise permutation layer and a non-linear substitution layer. The non-linear layer uses a single 4-bit S-box that is applied 16-times in each round.

III. An Architecture for PRESENT Block Cipher

A proposed architecture for the PRESENT block cipher is shown in Fig. 1. Here, 8-bit datapath has been chosen that provides optimal trade-off in terms of area and power. The proposed 64-bit datapath based architecture is capable of performing the encryption operation with a key register. The architecture consists of a 64-bit *Encryption Register* as in above figure, which is used to store the internal states of encryption operation. A 64-bit register is used to get the ciphertext at the output. By this the output gets synchronized with the last round. The main advantage of the proposed architecture is that there is a reduction in the latency along with efficient utilization of the hardware resources. Here, round keys are computed on-the-fly.

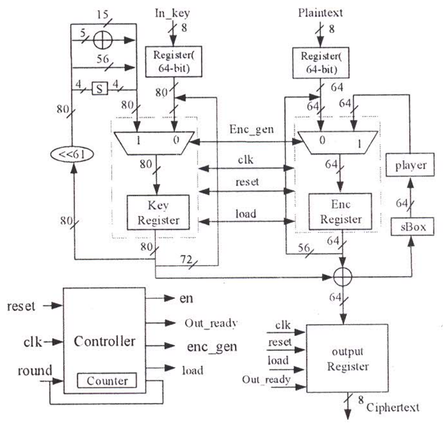

Fig. 1. Proposed architecture for the PRESENT cipher.

Plaintext is loaded in the first clock cycle. In the next clock cycle mux switches data and then for next 31 cycles all intermediate states are computed. Data is available at the *Encryption Register* and is mixed with intermediate round key i.e., XORed with first 64-bit of round key. Further, the mixed state is passed to *sboxlayer* for state processing, which provides 64-bit data concurrently to *P-layer* and subsequently, it is passed to the *Encryption Register* through the multiplexer. In the last clock cycle, ciphertext is available at output register. Thus, here, a total of 10+31+8=49 clock cycles are required to encrypt a single block of 64-bit plaintext.

A. Controller for Encryption Operation

A controller shown in Fig. 2 is use to generate five control signals which are: *en*, *key_load*, *inout_load*, *en_gen* and *out_ready*. as shown in Fig. 2 is designed to generate various required signals for controlling the key generation and encryption engine. There are six states in the FSM. In state S0 and S1, 8-bit plaintext and cipher key are loaded. In state S_2 the counter is enabled through *en*='1'. In fourth state (S3) multiplexers are switched and the *enc_gen* signal is used to start the intermediate operations by enabling the encryption and key registers. The state remains in S_3 until counter value reaches 31. Then, the state is switched to state S_4 where counter is disabled through *en*='0' and *out_ready* signal is switched to logic '1'. Further, in S5, it remains in the same state for next eight clock cycles and ciphertext is available through the output register.

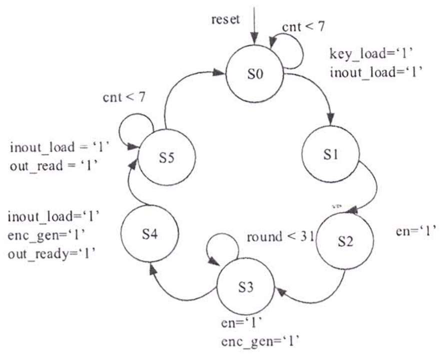

Fig. 2. FSM for the PRESENT cipher.

B. *An Inteface of the Crypto Core in SoC Environment*

Fig. 3 shows an interface of the PRESENT crypto core in the SOC environment. To integrate the proposed IP core in an SoC environment Cobham Gaisler open source IP suite is used.

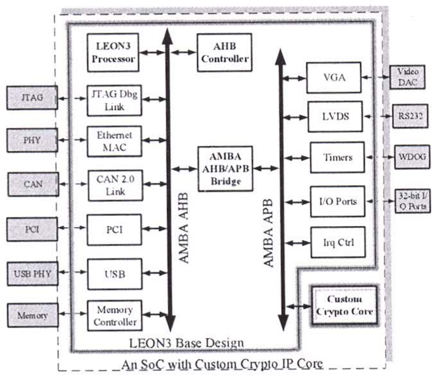

Fig. 3. PRESENT Crypto core in an SoC environment.

It provides a range of IP cores and supports tool for custom SoC design [3]. These cores communicate with a common bus interface known as advanced microcontroller bus architecture (AMBA). We can interface custom cores by requisite wrapping with AMBA advanced high-performance bus (AHB) or advanced peripheral bus (APB) buses. The core works as a slave with LEON3 processor as a master. The master requests data from the slave by enabling the APB bus and appropriate read/write signals as per the AHB/APB protocols [3].

IV. Experimental Results and Discussions

The core of requires, clock (*clk*) and *reset*, 8-bit input data, 8-bit user key and 8-bit output. Where, pc3d01 is used as an input pad; pc3o01for output pad and pc3c01 for the clock pad, where *pfrelr* is used as corner pads which is shown in Fig. 4. For supplying 3.3 V power *pvda* pad is used with *pv0a* as ground. Similarly, for 1.8 V power *pvdi* pad is used with *pv0i* as ground. Table I show the experimental result in terms performance of the implemented core after complete back-end design. The FPGA synthesis results of are shown in Table III.

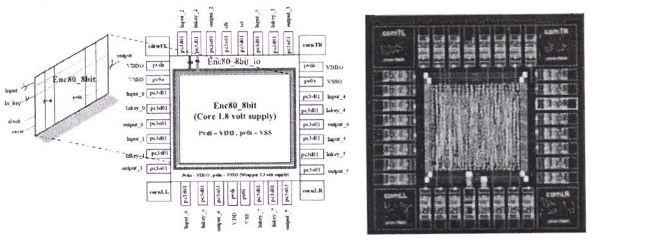

Fig. 4. PRESENT crypto chip (a) pin arrangement (b) layout.

TABLE I. Performance of the Core in SCL 0.18 μm Technology.

Elements	**Utilization**
Dynamic Power (mW)	2.452
Static Power (μW)	0.326
Total Power (mW)	2.453
Throughput (Mbps)	16.326
Energy (μJ)	1.202
Energy/bit (μJ/bit)	0.150
Efficiency	0.010

TABLE II. Comparison of Resource Utilization of [4] and Proposed Architecture on Xilinx Virtex-5 XC5VFX50 FF324-1 FPGA.

Elements	**Available Resources**	**Architecture [4]**	***Proposed Architecture***
Slice LUTs	28,800	285	267
Slice Registers	28,800	200	211
Total Slices	7,200	87	69
Latency	-	47	49
Max. freq. (MHz)	-	250.89	381.042
Throughput (Mbps)	-	42.70	62.21

It can be observed that in comparison to the implementation reported in [4], the proposed architecture requires 20.3% lower FPGA slices and able to work on an increased maximum frequency by 25.4%, and gain in the throughput by 25.4%.

V. Conclusion

In this paper, we have presented ASIC and FPGA implementations of the PRESENT cipher for SoCs application.

References

[1] "Information technology – Security techniques – Part 2: Block ciphers," Jan. 2012.

[2] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin and C. Vikkelsoe, "PRESENT: An ultra-lightweight block cipher," in *Int'l Workshop on Cryptographic Hardware and Embedded Systems*, Vienna, Austria, Springer, 2007, pp. 450-466.

[3] C. G. AB, "GRLIB IP Library," [Online]. Available: https://www.gaisler.com/index.php/products/ipcores/soclibrary. [Accessed 26 March 2018].

[4] N. Hanley and M. O'Neill, "Hardware Comparison of the ISO/IEC 29192-2 Block Ciphers," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Amherst, MA, USA, 19-21 Aug. 2012.