

Securing IoT Devices with Fast and Energy Efficient Implementation of PRIDE and PRESENT Ciphers

Vijay Dahiphale^{a,*}, Hrishikesh Raut^{b,2}, Gaurav Bansod^{c,3} and Devendra Dahiphale^{d,4}

^aBinghamton University, Binghamton, USA

^bNorth Carolina State University, Raleigh, USA

^cPune Institute of Computer Technology, Pune, India

^dUniversity of Maryland Baltimore County, Maryland, USA

ARTICLE INFO

Keywords:

Lightweight Cipher
Internet of Things
IoT Security
Encryption Algorithm
FPGA Implementation
PRIDE Cipher
PRESENT Cipher

ABSTRACT

The rise of low-power, cost-efficient internet-connected devices has led to a need for lightweight cryptography. The lightweight block cipher PRIDE, designed by Martin R. Albrecht, is one of the most efficient ciphers designed for IoT-constrained environments. It is useful for connected devices, requires fewer resources to implement, and has high performance. PRIDE is a software-oriented lightweight cipher optimized for microcontrollers. This paper focuses on the FPGA implementation of the PRIDE cipher by keeping throughput, energy, and power consumption metrics focused. The paper also presents a novel and simpler diagrammatical view of a Matrix Layer implementation of the PRIDE cipher. We also implemented the PRESENT cipher using the same metrics. We analyzed different design metrics on Field Programmable Gate Arrays (FPGAs) and compared the metrics of the PRIDE implementation with the well-known cipher PRESENT. This gives us an insight into the efficiency and reliability of PRIDE in IoT-constrained environments. We also proposed different architectures of the PRIDE cipher for 16-bit and 32-bit datapaths.

1. Introduction


Technological advancements, innovations, and research in the semiconductor industry have brought many small-scale devices to the market. Thanks to their small size, mobility, and low power consumption, these devices can be deployed anywhere pervasively. Moreover, many IoT devices can access the internet and transmit significant amounts of data to the server. The majority of these devices, however, are located in hostile environments where attackers can access the data. Thus we need to ensure that these devices do not have any security and privacy concerns. For the same, the concept of lightweight cryptography is introduced.

In the 21st century, the Internet of Things has brought up many small-size and low-resource-constraint devices. These devices transfer a large amount of sensitive data in the network and have limited computational capabilities, which make them vulnerable to physical attacks. To overcome these issues, the lightweight cryptography field has grabbed the cryptographic community's interest. Since the last decade, many new and efficient lightweight ciphers have been developed, making this field very active. Advanced Encryption Standard (AES) [1] was a significant discovery in the field of lightweight cryptography, influencing many cipher designs. One of the merits of AES was demonstrating the effectiveness of a well-designed linear layer. There are mainly two design strategies in cipher construction: Substitution Box (S-Box) based and without S-Box. Furthermore, S-Box-based structures are divided into Feistel-ciphers and

substitution-permutation networks (SPN). While AES is an example of an SPN cipher, the former Data Encryption Standard (DES) [2] is a Feistel cipher. AES and DES implement secure environments for controllers with adequate processing power but do not fulfill the constraints of low power and low computational devices like Radio-Frequency Identification (RFID) tags or nodes in sensor networks.

Since low-end microcontrollers with small word sizes dominate the Internet of Things (IoT) industry, software-friendly lightweight ciphers having low cycle counts and high efficiency are desired. However, some devices like RFID tags and nodes in the wireless network do not have any software programmable processor. Hence for such devices, security is provided by implementing lightweight ciphers on the hardware layer. Therefore, Hardware-based security is considered to be very beneficial for IoT devices. The efficiency of implementation depends on different design metrics, such as chip space, power consumption, and throughput. These metrics highly depend on the architecture and datapath used for the hardware implementation. So datapath selection is a crucial part of the hardware implementation of a cipher.

FPGAs are becoming increasingly popular in the development of hardware systems. Their low cost and low power consumption make them ideal for small, battery-operated devices, such as sensor nodes and RFIDs. Additionally, FPGAs are reconfigurable, which means that they can be updated and modified without having to be replaced. This makes them well-suited for applications that require regular changes or upgradation in the configurations [3]. It is important to use as few resources as possible when implementing lightweight ciphers on an FPGA device. This will leave more resources available for other circuitry in

 vdahiphale@binghamton.edu (V. Dahiphale);

hrishikeshraut.hpr@gmail.com (H. Raut); gaurav249@gmail.com (G.

Bansod); devendradahiphale@gmail.com (D. Dahiphale)

ORCID(S): 0000-0002-7113-3666 (V. Dahiphale); 0000-0002-4089-9714 (G. Bansod)

the application. This can be accomplished by choosing the appropriate datapath size for the implementation. ANU [4] and RECTANGLE [5] implement various datapath designs of lightweight ciphers and also discuss methods to design those architectures. In addition, PRESENT [6], HIGHT [7], PRINCE [8], CLEFIA [9], SIMON and SPECK [10], LED [11], and Camellia [12] also present FPGA implementations of some well-known lightweight ciphers. Selecting the right architecture design for FPGA implementation is based on applications, and the trade-off between performance metrics plays a vital role in developing an efficient architecture. Implementing various datapath architectures of a cipher gives an additional advantage to the user; that is, considering all the performance metrics of implementation, one can choose the suitable architecture for his device. In this paper, we focused on applications that require high-speed operations and are battery-operated. Thus we implemented novel designs for PRIDE and PRESENT cipher with 64-bit datapaths architecture on FPGA.

In the last few years, several ciphers were proposed like PRESENT [13], SIMON [14], SPECK [14], TWINE [15], ANU [16], ANU-II [17], LBlock [18], RECTANGLE [19], LED [20] which require a large number of rounds to guarantee security. That's where the linear layer in PRIDE [21] plays the role of keeping the number of rounds to a minimum, making the cipher efficient. One of the most effective ciphers created for IoT-constrained environments is the lightweight block cipher PRIDE by Martin R. Albrecht. It offers good performance, benefits linked devices, and takes fewer resources to implement. Moreover, the low-cost linear and S-Box layers make it a hardware-friendly cipher. The same can be validated based on proposed architecture implementation results in which PRIDE outwits PRESENT mainly in terms of throughput, power consumption, and energy.

The performance of block cipher PRIDE is brought to light when compared with block cipher PRESENT. In this paper, we have also proposed some architectures which are further open to scrutiny and adaption.

1.1. Contributions

This paper proposes and implements a 64-bit datapath of the block cipher PRIDE on FPGA. Furthermore, we have evaluated the Gate Equivalents (GE) of the PRIDE cipher design for 64-bit datapath architecture. To the best of our knowledge, this paper presents the first FPGA implementation of the PRIDE cipher, along with a detailed metrics analysis.

We have proposed architectures for different datapaths of 16-bit and 32-bit. These novel architectures give the user insight and ease of implementation for different architectures based on applications in resource-constrained environments. The proposed and implemented 64-bit architecture of PRIDE is an efficient one which results in maximum throughput and less latency. All our results are compared with the PRESENT architecture, which shows the edge

of PRIDE cipher over PRESENT architecture in terms of throughput, latency, and power consumption.

We also implemented a datapath of PRESENT on the same platform so that the comparison could be justified. Our paper not only gives the hardware implementation of PRIDE cipher but also gives insight into different tradeoffs between lookup tables, GE, and throughput. For resource-constrained environments like IoT, efficient datapath implementation plays a crucial role which we have successfully addressed in this paper. This paper contributes to the full-fledged hardware implementation of lightweight cipher PRIDE and compares various hardware implementation's design metrics of PRIDE with a lightweight cipher PRESENT. It also gives the reader insight into different datapath designs and their different design metrics.

2. Lightweight Block Cipher - PRIDE

The lightweight block cipher PRIDE is one of the high-performing and most efficient ciphers proposed for IoT devices. It is an SPN-based structure with a specially designed linear layer for efficient use of resources. The round function of the PRIDE cipher consists of linear and non-linear functions layers with 4-bit S-Boxes, Matrix Layer, and a bit-permutation.

PRIDE has a 64-bit plaintext size and 128-bit key size. Similar to PRINCE [22], the cipher uses FX construction. Pre-whitening keys k_0 and k_2 are produced from 64 Least significant bits (LSB) of key k , while 64 Most Significant Bits (MSB) are used as a base for round key k_1 .

$$\begin{aligned} k &= k_0 || k_1 \\ k_2 &= k_0 \end{aligned}$$

The cipher begins and finishes with a bit-permutation which results in better efficiency of bit-sliced implementation. The cipher has 20 rounds consisting of 19 identical rounds and one different round. This round-based structure of PRIDE is shown in Figure 1.

2.1. Key Scheduling

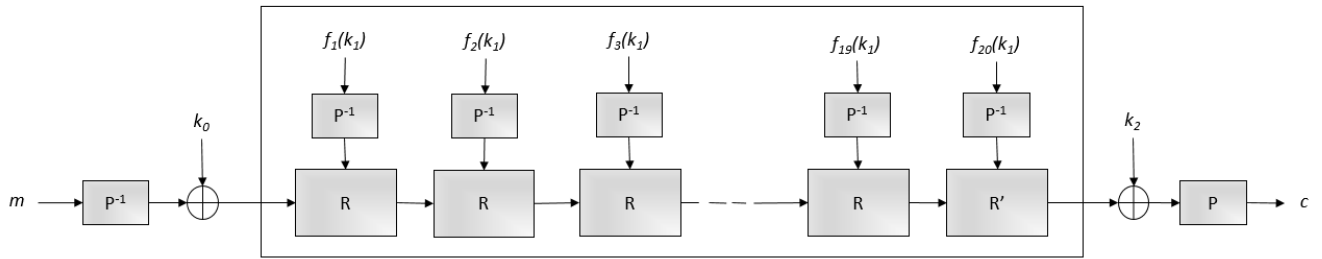
PRIDE has a total of 20 unique sub-key rounds. The unique sub-key for each round is produced by adding round constants to the parts of the key. The sub-key for i^{th} round can be obtained by $f_i(k_1)$ shown in Equation 1.

$$\begin{aligned} f_i(k_1) = & k_{10} || g_i^{(0)}(f_{i-1}(k_{11})) || k_{12} || g_i^{(1)}(f_{i-1}(k_{13})) \quad (1) \\ & || k_{14} || g_i^{(2)}(f_{i-1}(k_{15})) || k_{16} || g_i^{(3)}(f_{i-1}(k_{17})) \end{aligned}$$

Where,

$$\begin{aligned} g_i^{(0)}(x) &= (x + 193 * i) \bmod 256 \\ g_i^{(1)}(x) &= (x + 165 * i) \bmod 256 \\ g_i^{(2)}(x) &= (x + 81 * i) \bmod 256 \\ g_i^{(3)}(x) &= (x + 197 * i) \bmod 256 \end{aligned}$$

These functions are added to respective bytes of k_1 to perform a key scheduling algorithm [21].


Figure 1: Round Structure of PRIDE
Table 1
S-Box used for PRIDE

X	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[X]$	0	4	8	F	1	5	E	9	2	7	A	C	B	D	6	3

2.2. Substitution Box (S-Box)

S-Box is a critical component of the cipher's security analysis [13]. PRIDE uses a 4-bit Substitution Box (S-Box) to produce non-linearity in the encryption process. Since S-Box is the main non-linear component of the encryption design, it must be strong and hence must satisfy all standard S-Box criteria mentioned in [23]. Table 1 shows a S-Box used for PRIDE cipher. This is an involution S-Box which prevents the encryption/decryption overhead [21].

2.3. The Linear Layer (L)

PRIDE cipher has a Linear Layer 'L', which can be split into three sub-layers as:

1. Permutation Layer (P)
2. Inverse Permutation Layer (P^{-1})
3. Matrix Layer (M)

The relation between these layers is given in Equation 2.

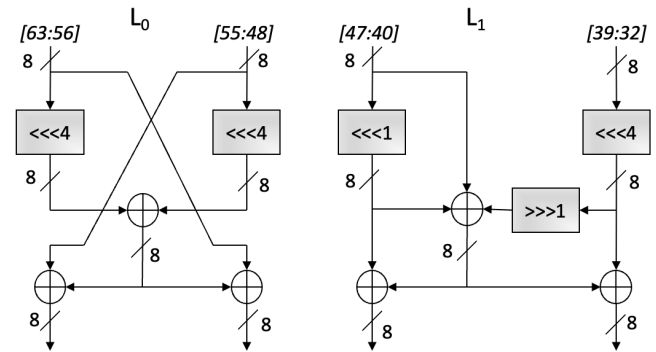
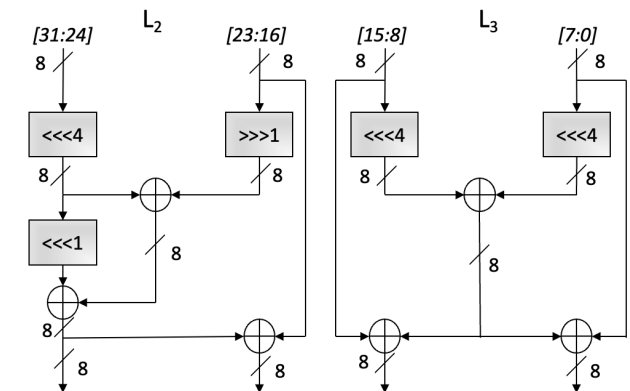
$$L = P(L_0 * L_1 * L_2 * L_3)P^{-1} \quad (2)$$

2.3.1. Permutation Layer (P -layer)

PRIDE uses two 64-bit permutation layers denoted as P and P^{-1} to shuffle bits during the encryption process. These layers act as complements of each other and are mainly used to generate a good avalanche effect. In turn, this effect helps the cipher to increase its complexity. 64-bit P -Layer and P_1 -Layer of PRIDE are given in [24].

2.3.2. Matrix Layer

The matrix layer serves as a key block in the PRIDE cipher, and this paper introduces an innovative approach for the FPGA implementation of matrix layer as shown in Figures 2 and 3. As the construction of PRIDE allows combining small matrices into bigger ones, the linear layer is implemented using four binary matrices of size 16x16 to construct a 64x64 matrix. This matrix layer is introduced to increase the overall efficiency and, mainly, the security of the


Figure 2: L_0 and L_1 Matrix Layer Implementation structure

Figure 3: L_2 and L_3 Matrix Layer Implementation structure

block cipher. Matrices L_0 , L_1 , L_2 , and L_3 are specifically invented for PRIDE cipher. These software-friendly matrices are given in the [5] Appendices section.

Serially implementing these matrices improves the hardware efficiency [25]. Serial implementation of matrices L_0 , L_1 , L_2 , and L_3 using equations in [25] is shown in Figures 2 and 3. Hence Matrix layer is implemented using Figures 2 and 3 without using matrices L_0 , L_1 , L_2 , and L_3 .

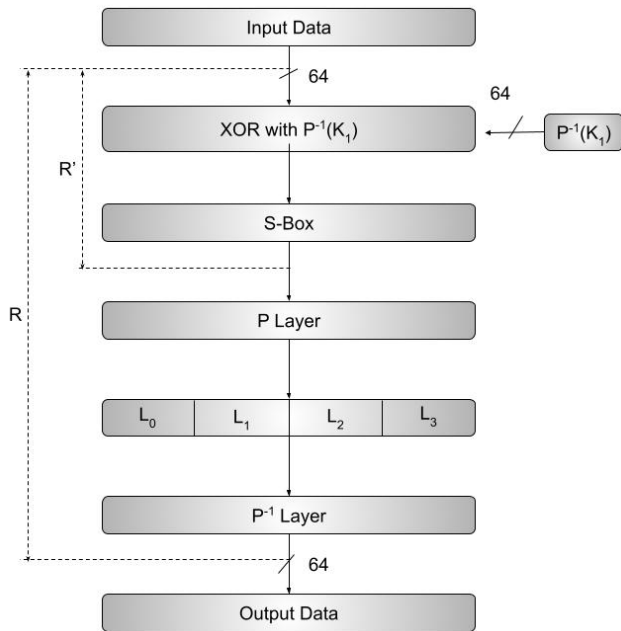


Figure 4: Round Structure of PRIDE Block Cipher [26]

2.4. Flow of the Cipher

At the beginning, the cipher performs P^{-1} on plaintext and then it adds the result and k_0 . Then it performs the 20 rounds as described previously and adds the key k_2 to the result. Finally, the bit-permutation is performed on the result from the previous 20 rounds to generate the Ciphertext. The general structure of PRIDE is shown in Figure 4 [26].

The diffusion is not necessary at the last step of the cipher. As a result, the last round finishes after the substitution layer. This data is used for post-whitening, where key k_2 is applied, and its permutation gives us the Ciphertext.

3. Hardware Implementation

FPGA is a configurable Integrated Circuit (IC) that consists of components like Flip-Flops (F-F), Slices, Lookup Tables (LUTs), routing switches, etc. The user can configure these configurable ICs differently based on the application. New generation FPGA devices provide more flip-flops, slices, and LUTs [27] [28], enabling the implementation of more complex architectures that require larger areas. The performance of an FPGA implementation depends on metrics such as throughput, dynamic power, static power, and energy [29] [30]. Newer FPGA devices are often LUT-6-based, which allows for more compact architecture implementations than LUT-4-based devices.

Area (i.e., Flip-Flops, LUTs, Slices) is the critical design metric in implementing the lightweight cipher in a highly resource-constrained environment, and the other design metrics like Throughput, Latency will play a critical role where the fast operation is required. Most of the devices in IoT and Embedded System environments are battery-operated. Hence Power consumption will be the critical parameter for

these devices. All these metrics are architecture dependents, and different architectures should be proposed to address the challenges posed by resource-constrained environments like IoT. Getting all these metrics right in single architecture is an arduous task for the researcher, as a trade-off exists between these parameters [31].

In this paper, PRIDE is implemented on four FPGA platforms with its 64-bit datapath (round-based) architecture. The implementation results of the PRIDE cipher are compared with a PRESENT cipher. Also, we have proposed serialized architectures for PRIDE in APPENDIX A.

3.1. Performance Metrics

The performance of the lightweight cipher depends on a few parameters like area, the speed of operation, and energy utilization and power consumption. The right platform is the most critical aspect while implementing a cipher on FPGA, as the parameters of the cipher depend on the platform. Although all the parameters are important, we have primarily focused on the speed of execution for implementing the PRIDE lightweight cipher.

3.1.1. Platform

Proposed architectures are implemented on the Xilinx FPGA board using the ISE Design Suite 14.7. Four different FPGA platforms - Spartan-3 (xc3s700an-5fgg484), Spartan-6 (xc6slx45t-3fgg484), Virtex-4 (xc4vlx25-12ff668), Virtex-5 (xc5vlx50t-3ff1136) are used to get a clear idea about implementation and performance of the proposed designs.

Platforms are selected based on their ability to support LUT-4 and LUT-6-based technology. Spartan-3 and Virtex-4 device is LUT-4 based, and Spartan-6 and Virtex-5 is LUT-6 based technology. Implementation results are analyzed at speed grades of -5, -3, -12, and -3 for Spartan-3, Spartan-6, Virtex-4, and Virtex-5, respectively. ISO standard frequency for RFID tags and smart cards, i.e., 13.56 MHz, is used for the implementation and the performance metrics [32].

3.1.2. Area

The area metric is a measure of the space required to implement a design. It consists of components like flip-flops, LUTs, and slices used. A new generation FPGA device has many of these components compared to the old FPGA devices. Also, LUT-6-based devices require fewer components than LUT-4-based devices. So, new generation LUT-6-based devices can be used to implement more complex designs and compact hardware implementation. Therefore, the area needed for the implementation should be as low as possible to implement the design in a constrained environment.

3.1.3. Power Consumption

Power consumption is also a crucial design metric for IoT devices that are battery-powered. Any circuit design has two types of power: static and dynamic.

The total power consumption of the implementation is calculated by the addition of these two powers, as given in Equation 3. The power consumption is evaluated at the standard frequency of 13.56 MHz [32]. Therefore, the circuit's

power consumption or architecture should be low to have a more extended battery backup.

$$Total\ Power = Dynamic\ Power + Static\ Power \quad (3)$$

3.1.4. Speed of Operation

The speed of operation or throughput purely depends on the latency. Latency is the total number of clock cycles needed to encrypt one 64-bit block of data. Therefore, we have proposed the speed of the design based on three different parameters, i.e., throughput at 13.56 MHz, maximum throughput at maximum frequency, and throughput per slice.

Various Formulas used to calculate the speed of the implementation are shown using Equations 4, 5, and 6. The plaintext size for the PRIDE cipher is 64-bit. The throughput of the architecture should be as high as possible, and latency should be as low as possible for fast operation of the design.

$$Throughput_{@13.56MHz}(Thr^*) = \frac{13.56 \times Plaintext\ Size}{MHz \times Latency} \quad (4)$$

$$Throughput\ per\ slice = \frac{Throughput_{@13.56MHz}(Thr^*)}{Slices} \quad (5)$$

$$Maximum\ Throughput(Thr) = \frac{Maximum\ Frequency \times Plaintext\ Size}{Latency} \quad (6)$$

3.1.5. Energy Utilization

The circuit or architecture should consume less energy, which is the total energy needed to encrypt a 64-bit plaintext. The energy consumption is expressed in two parameters: Energy and Energy per bit. Both parameters are calculated at a standard frequency of 13.56 MHz using Equations 7 and 8.

$$Energy\ (E^*) = \frac{Total\ Power * Latency}{13.56MHz} \quad (7)$$

$$Energy\ (E^*)\ per\ bit = \frac{Energy}{Plaintext\ Size} \quad (8)$$

4. Proposed Architecture

The high-level model of the PRIDE implementation is shown in Figure 5. It requires 258 input-output blocks (IOBs).

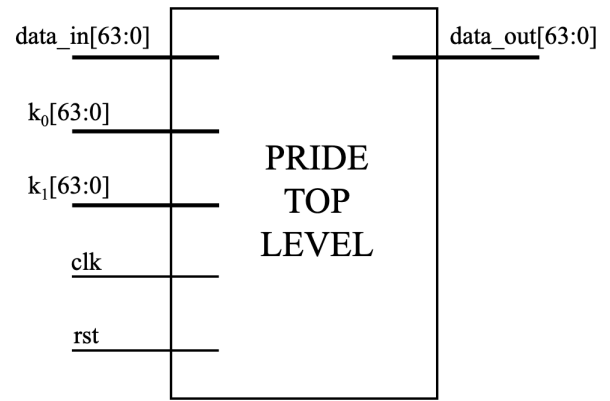


Figure 5: Top Level Model of PRIDE Implementation

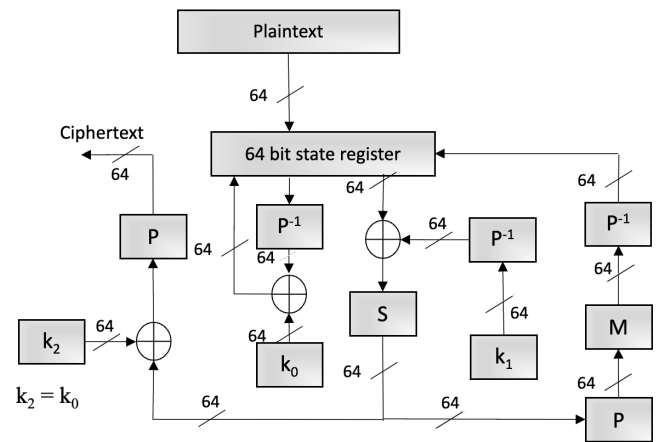


Figure 6: Data-Layer of round-based PRIDE Implementation (A1).

4.1. Round-Based PRIDE Architecture (A1)

Figure 6 shows the round-based architecture (A1) for the PRIDE Cipher with 64-bit datapath size. It is built and optimized to produce the highest possible throughput while encrypting the data. This architecture is a round-based design employing multiple S-Boxes, 8-bit adders, and other components in parallel to facilitate rapid encryption. This architectural decision is not contingent upon FPGA or ASIC implementation; rather, it is aimed at enhancing operational speed. Specifically tailored for applications with high-speed requirements, this architecture prioritizes efficiency in encryption processes.

4.1.1. Operation

Initially, data is stored in the 64-bit state register. Design operation is divided into three primary operations: pre-whitening, identical round, and post-whitening. Note that the round-based implementation does not require multiplexers to store the data [13].

- **Pre-Whitening Operation:**

The pre-whitening operation consists of an inverse Permutation layer and XOR with key k_0 , shown in

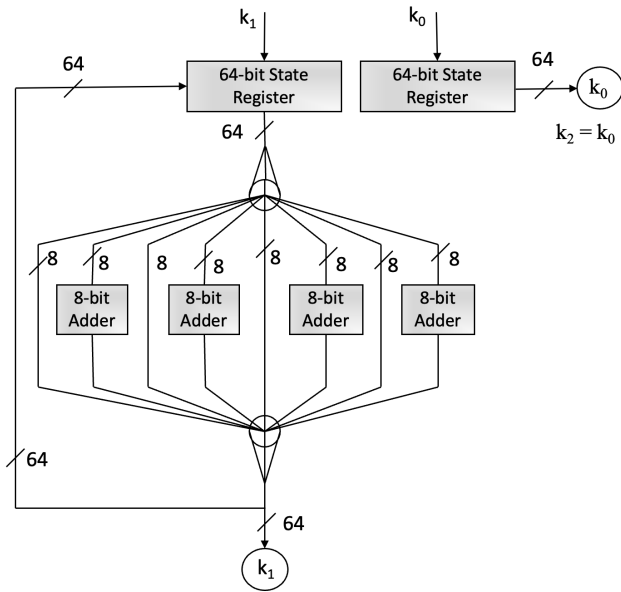


Figure 7: Key-Layer of round-based PRIDE Implementation.

Figure 6. The entire pre-whitening operation can be executed in one clock cycle and needs only one state.

- **Identical Rounds:**

Identical rounds are shown in Figure 6. PRIDE cipher has a total of 19 identical rounds. The identical round performs key k_1 XOR, S-Box substitution, permutation layer, Matrix layer, and inverse permutation in only one clock cycle. All these operations need only one state to perform.

- **Round R' and Post-Whitening Operation:**

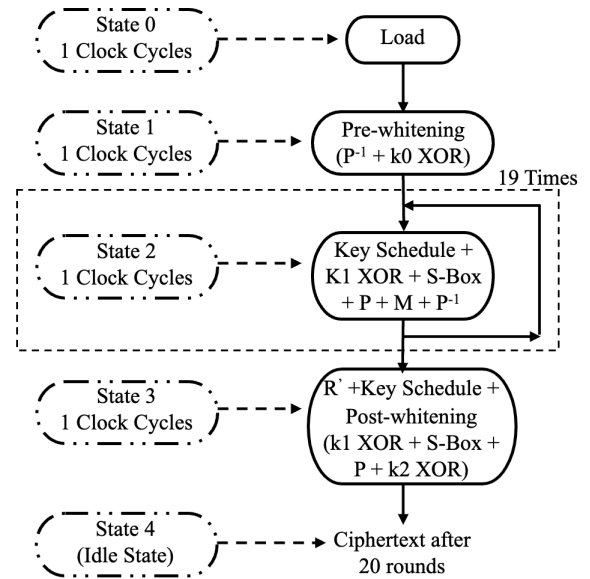
Round R' shown in Figure 4 consists of two operations, i.e., key k_1 XOR and S-Box substitution. In contrast, the post-whitening operation consists of key k_2 XOR and permutation layer as shown in Figure 6. All these operations in R' and post-whitening key are performed in a single clock cycle.

- **Key Scheduling:**

Key Scheduling datapath architecture is shown in Figure 7, which requires four 8-bit full adders. However, since the operation is on just 8 bits, there is no need to divide it by 256 (mod 256).

4.1.2. Clock Cycles and States

Figure 8 shows the clock cycles and states required for the round-based implementation of the PRIDE cipher. The implementation involves five states and one clock cycle for executing each stage. Among these five states, state one is used for pre-whitening, state 2 is an identical round state and runs 19 times, and state 3 is for post-whitening along with the last round operation; thus, the complete PRIDE implementation requires 21 clock cycles which results in higher throughput of the proposed PRIDE architecture.



$$\text{Total Clock Cycles required} = 1 + 1 \cdot 19 + 1 = 21$$

Figure 8: Clock Cycle Analysis Diagram

Table 2

Standard Library Values.

Gate	D-FF	XOR	1-bit Full Adder	MUX	AND	OR
GE	4.25	2	5.75	2.25	1.25	1.25

4.1.3. Gate Equivalents (GE)

The proposed architecture's gate equivalents are estimated using the ARM-7 standard ASIC library IBM 8RF with 0.130-micron technology. As indicated in Table 2 [33], this library specifies GE for logic gates, flip-flops, multiplexers, etc.

Tables 3, 4, and 5 show the GE required for S-Box, Data-Layer, and Key-Layer, respectively, of the proposed architecture. PRIDE cipher uses 4-bit input ($[I_3, I_2, I_1, I_0]$) and 4-bit output ($[O_3, O_2, O_1, O_0]$) S-Boxes and Equation 9 is used for calculating GEs required for a single S-Box.

$$\begin{aligned} O_3 &= I_1 \oplus (I_3 \& I_2) \\ O_2 &= I_0 \oplus (I_2 \& I_1) \\ O_1 &= I_3 \oplus (O_3 \& O_2) \\ O_0 &= I_2 \oplus (O_2 \& O_1) \end{aligned} \quad (9)$$

The round-based implementation of the PRIDE cipher requires a total of $1056 + 728 = 1784$ GE. Note that GE for the matrix layer is calculated using Figures 2 and 3.

Table 3
GE Calculations for single S-Box

S-Box Components	Quantity	GE
XOR	4	4*2=8
AND	4	4*1.25=5
Total		13

Table 4
GE Calculations for Data-Layer of Round-based implementation of PRIDE

Data-Layer Components	Quantity	GE
64-bit register	1	64*4.25 = 272
4-bit S-Box	16	16*13 = 208
64-bit XOR	3	3*64*2 = 384
8-bit XOR (M layer)	12	12*8*2 = 192
Total		1056

Table 5
GE Calculations for Key-Layer of Round-based implementation of PRIDE.

Key-Layer Components	Quantity	GE
64-bit register	2	2*64*4.25 = 544
8-bit full adders	4	4*8*5.75 = 184
Total		728

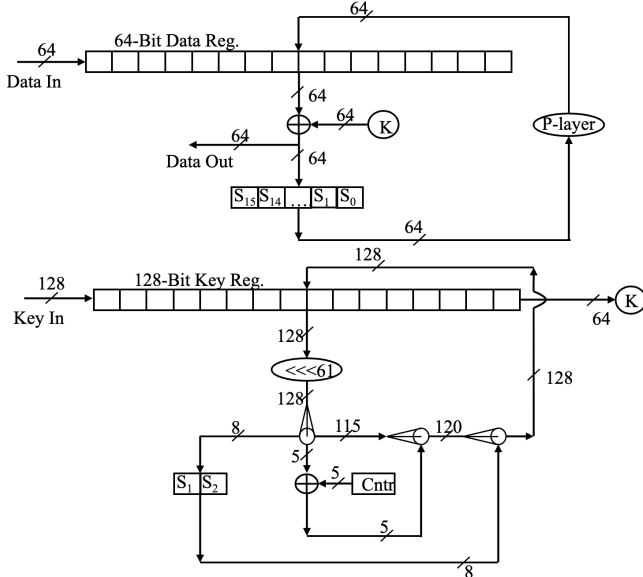


Figure 9: Proposed architecture of PRESENT for Comparison

4.2. Proposed PRESENT Architecture (A2)

Figure 9 shows the proposed architecture of PRESENT [13]. This architecture closely resembles A3, as proposed in [34], and is implemented similarly to A1 to ensure a fair comparison of the ciphers across the selected platforms.

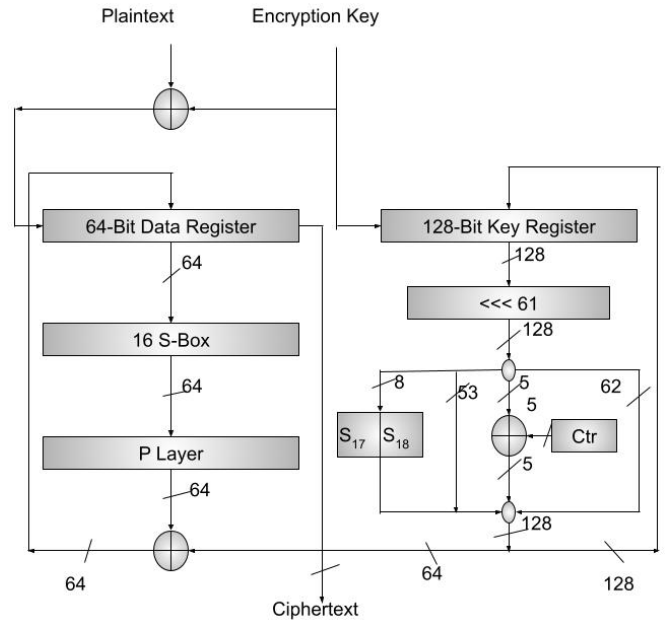


Figure 10: Iterative Architecture of PRESENT proposed in [34]

Table 6
Different Architectures Description

Architecture	Cipher	Data path size	Data Size	Key Size	Reference	Description
A1	PRIDE	64	64	128	This work	Round based
A2	PRESENT	64	64	128	This work	Round based
A3	PRESENT	64	64	128	[34]	Iterative

4.3. Iterative Architecture - Neil Hanley et al (2012) [34] (A3)

This architecture is proposed in [34]. PRIDE is implemented using round-based (Iterative architecture), and similar PRESENT architecture is chosen for comparing these two ciphers. The architecture is shown in Figure 10 [29].

Table 6 shows the description of the different architectures used for the comparison.

5. Results and Evaluation

All the implementation results of the proposed architecture are shown in Tables 7 and 8. Table 7 compares the area and throughput results, and Table 8 compares the energy and power consumption results of different architectures A1, A2, and A3. The architectures A1 and A2 are evaluated on our platform, and architecture A3 is implemented in [34]. Since the proposed architecture of PRIDE is implemented using round-based architecture, An Iterative (Round based) architecture of PRESENT in [34] (A3) is chosen for the comparison.

Table 7
Area and Throughput Comparison of Different Architectures

Architecture	Device	Data Size (bit)	Key Size (bit)	F-F	LUT's	Slices	Clock Cycles	Max. Freq. (MHz)	Thr @ Fmax (Mbps)	Thr* @ 13.56 (Mbps)	Thr* per slice (Kbps/slice)
A1	Spartan-3	64	128	201	654	372	21	179.329	546.52	41.32	111.09
A2	xc3s700an-	64	128	201	614	330	31	247.687	511.35	27.99	84.83
A3	5fgg484	64	128	-	-	-	55	-	-	15.77	-
A1	Virtex-4	64	128	204	744	405	21	314.323	957.93	41.32	102.02
A2	xc4vlx25-	64	128	237	609	344	31	472.891	976.29	27.99	81.36
A3	12ff668	64	128	200	382	192	55	284.330	330.85	15.77	82.13
A1	Spartan-6	64	128	200	372	113	21	222.844	679.14	41.32	365.66
A2	xc6slx45t-	64	128	203	246	87	31	277.027	571.92	27.99	321.72
A3	3fgg484	64	128	-	-	-	55	-	-	15.77	-
A1	Virtex-5	64	128	201	517	176	21	382.424	1165.48	41.32	234.77
A2	xc5vlx50t-	64	128	201	377	158	31	475.692	982.07	27.99	177.15
A3	3ff1136	64	128	200	283	88	55	271.670	316.12	15.77	179.20
PRESENT Iterative [6]	Spartan-3 XC3S200	64	128	200	313	165	55	234.300	272.64	15.778	95.624
PRESENT Serial [6]	-5FT256	64	128	203	260	132	303	176.734	37.33	2.864	21.696
Piccolo [35] (section 2.A)	Spartan-3 XC3S50-5	64	128	206	545	286	62	69.56	71.8	13.99	46.478
Piccolo [35] (section 2.B)		64	128	248	575	301	62	48.23	49.78	13.99	46.478
LED (X)4 [5]		64	128	76	456	233	48	98.700	131.20	18.08	77.596
RECTANGLE D1 [5]	Spartan-3 xc3s700an	64	128	199	593	317	26	278.750	686.153	33.378	105.293
ANU D4 [4]	-5fgg484	64	128	199	513	272	25	262.123	671.03	34.71	127.610
HIGHT [7] (SK, UF = 4)	Spartan-3 XC3S200	64	128	-	-	908	-	-	366	-	-
PRINCE [8] (two-cycle)	Virtex-6	64	128	-	-	831	-	172.265	5512.489	-	-
RECTANGLE A2 [36]	Spartan-3 xc3s4000l-4fg900	64	128	203	377	212	48	108.51	144.643	18.08	85.283

5.1. Graphs

We have implemented PRIDE on four different platforms. The implementation results depend on the platform chosen for the evaluation [29]. Hence choosing the right platform is also the most critical task for hardware implementation. To get a clear idea about the implementation, the proposed architecture is implemented on LUT-4 and LUT-6-based technology.

Operating frequency is also an essential parameter while designing the circuit as power consumption, energy, throughput, etc., depend on it. Therefore, all implementations use the standard frequency of 13.56 MHz, which is the standard frequency for smart cards and RFID tags [32]. Graphs in Figures 11 to 17 compare all the implementation metrics.

Figures 11 to 13 compares Flip-Flops, LUTs, and Slices of PRIDE implementation with the PRESENT cipher. Based on the implementation results, it can be observed that PRIDE is a little expensive in terms of Area metric. Hence it may not be the best choice to implement PRIDE in area constrained environment.

In contrast, the proposed implementation of PRIDE performs excellently in terms of throughput, latency, power, and energy consumption. The same can be observed from the implementation results shown in Figures 14 to 17 and Table 8 where PRIDE outwits PRESENT almost in every implemented platform. Therefore, this architecture would be a better choice for IoT applications where high-speed

Table 8
Power and Energy Consumption Comparison of Different Architectures

Architecture	Device	Data Size (bit)	Key Size (bit)	Latency	Static Pow. (mW)	Dynamic Pow. (mW)	Total Pow. (mW)	E* (uJ)	E* per bit (nJ/bit)
A1	Spartan-3	64	128	21	36.51	40.19	76.69	0.118	1.843
A2	xc3s700an-	64	128	31	36.35	6.48	42.83	0.097	1.515
A3	5fgg484	64	128	55	-	-	-	-	-
A1	Virtex-4	64	128	21	233.19	24.43	257.61	0.398	6.218
A2	xc4vlx25-	64	128	31	233.03	16.15	249.18	0.569	8.890
A3	12ff668	64	128	55	333.44	15.44	348.88	1.415	22.109
A1	Spartan-6	64	128	21	36.21	11.19	47.40	0.073	1.140
A2	xc6slx45t-	64	128	31	49.49	13.25	36.24	0.082	1.281
A3	3fgg484	64	128	55	-	-	-	-	-
A1	Virtex-5	64	128	21	560.20	19.26	579.46	0.897	14.015
A2	xc5vlx50t-	64	128	31	560.22	20.85	581.06	1.328	20.750
A3	3ff1136	64	128	55	560.04	3.47	563.51	2.285	35.703
RECTANGLE A2 [36]	Spartan-3 xc3s4000l-4fg900	64	128	48	268.27	5.68	273.94	0.967	0.085
RECTANGLE D1 [5]	Spartan-3 xc3s700an	64	128	26	36.38	13.16	49.54	0.094	1.468
ANU D4 [4]	-5fgg484	64	128	25	36.35	6.59	42.94	0.079	1.234

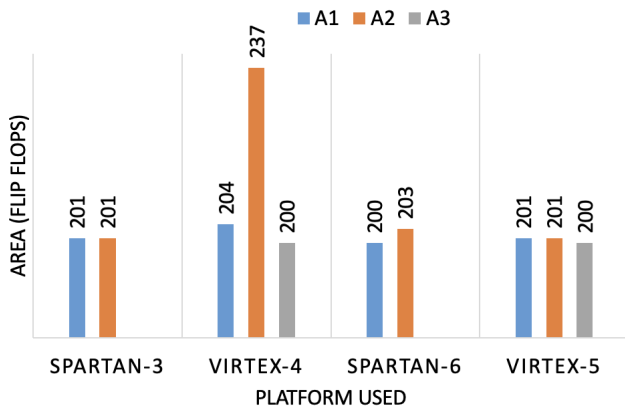


Figure 11: Flip-Flops Comparison of A1, A2, and A3

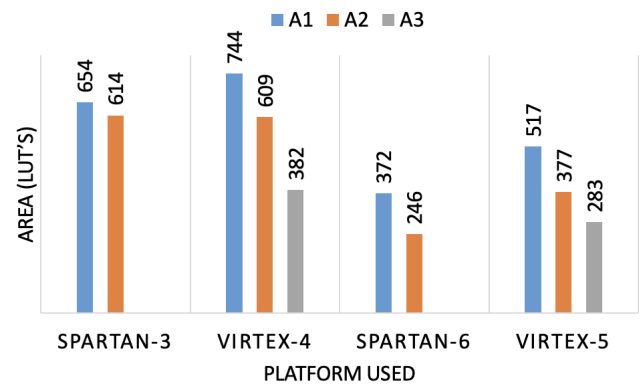


Figure 12: LUTs Comparison of A1, A2, and A3

data communication is essential and beneficial for battery-operated devices.

6. Conclusion and Future Scope

This paper compares hardware architectures for the block cipher PRIDE and PRESENT. Both the ciphers were implemented on multiple platforms with 64-bit data and 128-bit keys for encryption. The experimental findings for

the proposed 64-bit architecture of PRIDE and PRESENT were achieved by running the Verilog code on four different FPGAs. The shown results are the outcome of fair experimentation. It is possible to replicate the results for all the architectures mentioned in this paper using the source files.

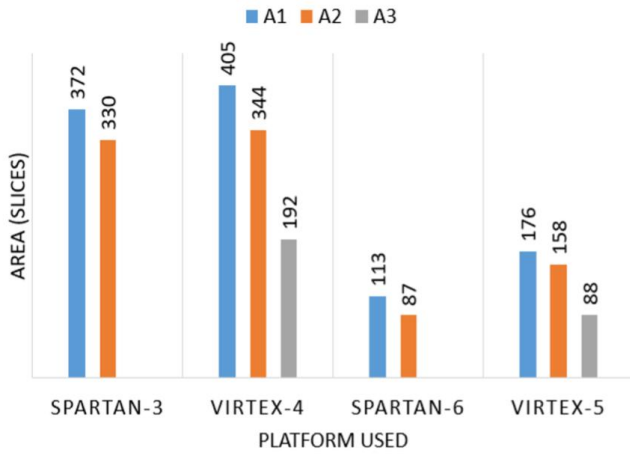


Figure 13: Slices Comparison of A1, A2, and A3

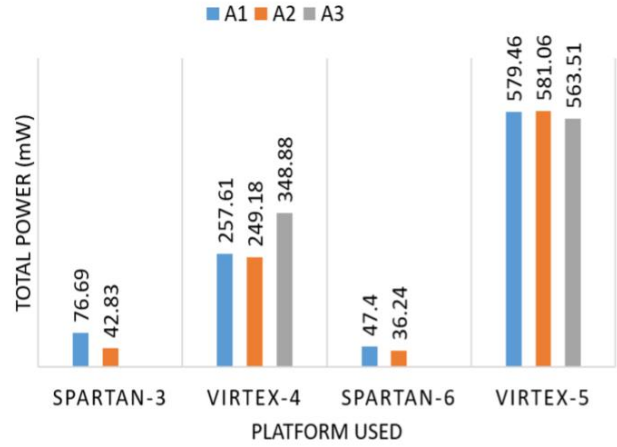


Figure 16: Total Power Comparison of A1, A2, and A3

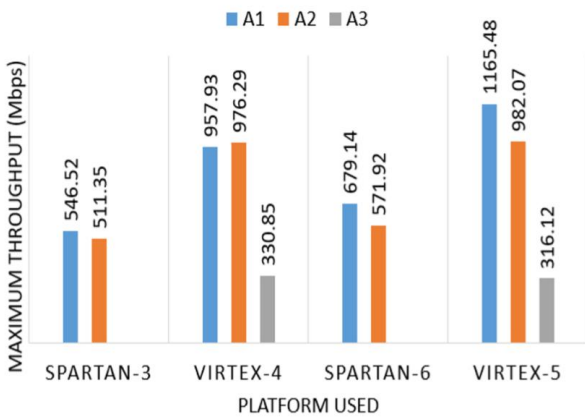


Figure 14: Maximum Throughput Comparison of A1, A2, and A3

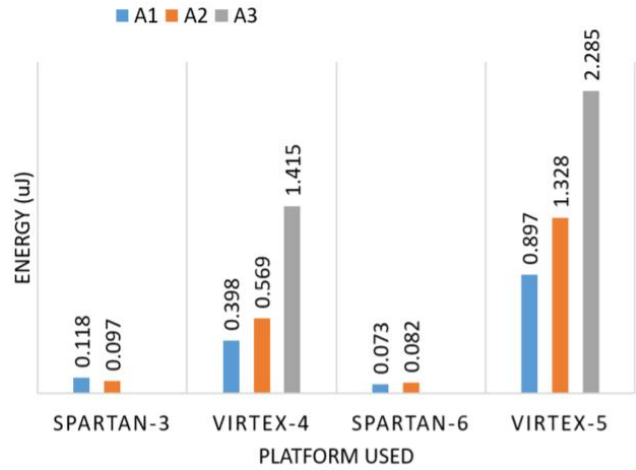


Figure 17: Energy Consumption Comparison of A1, A2, and A3

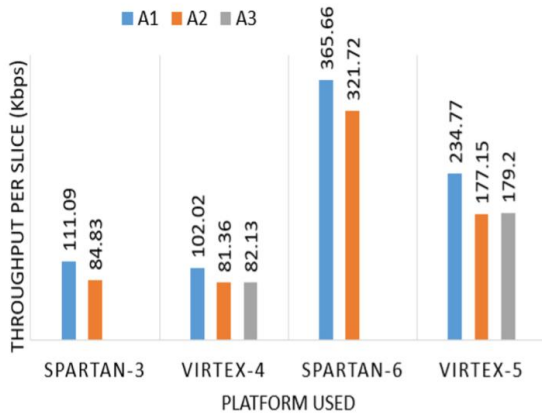


Figure 15: Throughput per Slice Comparison of A1, A2, and A3

From the graphs and tables, we can compare the two ciphers. On the Spartan-6 platform, we get a compact implementation. We get the highest throughput on the Virtex-5 platform. Dynamic power consumption is less on both, Spartan-6 and Virtex-5. The latency of block cipher PRIDE

is better as it requires only 20 rounds to encrypt the data. From the data obtained, we can conclude that 64-bit PRIDE gives much higher performance and throughput compared to 64-bit PRESENT at the cost of slightly more chip area.

In this paper, we proposed novel state-of-the-art architectures for different datapath sizes. We also have proposed 16-bit and 32-bit datapaths architectures of block cipher PRIDE with detailed latency calculations in APPENDIX A. These architectures require more machine cycles but can be implemented on lower-end devices. These proposed architectures have scope for further research and modifications.

References

- [1] M. Dworkin, E. Barker, J. Nechvatal, J. Foti, L. Bassham, E. Roback, J. Dray, Advanced encryption standard (aes) (2001-11-26 2001). doi: <https://doi.org/10.6028/NIST.FIPS.197>.
- [2] N. I. of Standards, Technology, Data Encryption Standard (DES), <https://csrc.nist.gov/CSRC/media/Publications/fips/46/3/archive/1999-10-25/documents/fips46-3.pdf> (1999).
- [3] E. B. Kavun, RESOURCE-EFFICIENT CRYPTOGRAPHY FOR UBIQUITOUS COMPUTING, <https://d-nb.info/1079843078/34>

- (2014).
- [4] V. Dahiphale, G. Bansod, A. Zambare, N. Pisharoty, Design and implementation of various datapath architectures for the anu lightweight cipher on an fpga, *Frontiers of Information Technology & Electronic Engineering* 21 (2020) 615–628.
 - [5] V. Dahiphale, H. Raut, G. Bansod, Design and implementation of novel datapath designs of lightweight cipher rectangle for resource constrained environment, *Multimedia Tools and Applications* 78 (2019) 23659–23688.
 - [6] S. Ashaq, M. Nazish, M. Ali, I. Sultan, M. Tariq Banday, Fpga implementation of present block cypher with optimised substitution box, in: *2022 Smart Technologies, Communication and Robotics (STCR)*, 2022, pp. 1–6. doi:10.1109/STCR55312.2022.10009366.
 - [7] B. Rashidi, High-throughput and lightweight hardware structures of hight and present block ciphers (Aug. 2019). doi:10.1016/j.mejo.2019.06.012.
URL <http://dx.doi.org/10.1016/j.mejo.2019.06.012>
 - [8] B. Rashidi, Low-cost and two-cycle hardware structures of prince lightweight block cipher (Jul. 2020). doi:10.1002/cta.2832.
URL <http://dx.doi.org/10.1002/cta.2832>
 - [9] B. Rashidi, Efficient and flexible hardware structures of the 128 bit clefia block cipher (Jan. 2020). doi:10.1049/iet-cdt.2019.0157.
URL <http://dx.doi.org/10.1049/iet-cdt.2019.0157>
 - [10] B. Rashidi, High-throughput and flexible asic implementations of simon and speck lightweight block ciphers (May 2019). doi:10.1002/cta.2645.
URL <http://dx.doi.org/10.1002/cta.2645>
 - [11] B. Rashidi, Flexible structures of lightweight block ciphers present, simon and led (Feb. 2020). doi:10.1049/iet-cds.2019.0363.
URL <http://dx.doi.org/10.1049/iet-cds.2019.0363>
 - [12] B. Rashidi, Flexible and high-throughput structures of camellia block cipher for security of the internet of things (Mar. 2021). doi:10.1049/cdt2.12025.
URL <http://dx.doi.org/10.1049/cdt2.12025>
 - [13] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, C. Vikkelsoe, Present: An ultra-lightweight block cipher, in: P. Paillier, I. Verbauwhede (Eds.), *Cryptographic Hardware and Embedded Systems - CHES 2007*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 450–466.
 - [14] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, L. Wingers, The simon and speck families of lightweight block ciphers, *Cryptology ePrint Archive, Paper 2013/404*, <https://eprint.iacr.org/2013/404> (2013).
URL <https://eprint.iacr.org/2013/404>
 - [15] T. Suzaki, K. Minematsu, S. Morioka, E. Kobayashi, Twine: A lightweight block cipher for multiple platforms, in: *Selected Areas in Cryptography: 19th International Conference, SAC 2012*, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers 19, Springer, 2013, pp. 339–354.
 - [16] G. Bansod, N. Raval, N. Pisharoty, Implementation of a new lightweight encryption design for embedded security, *IEEE Transactions on Information Forensics and Security* 10 (1) (2014) 142–151.
 - [17] V. Dahiphale, G. Bansod, J. Patil, Anu-ii: A fast and efficient lightweight encryption design for security in iot, in: *2017 International Conference on Big Data, IoT and Data Science (BIG)*, IEEE, 2017, pp. 130–137.
 - [18] W. Wu, L. Zhang, Lblock: A lightweight block cipher, in: J. Lopez, G. Tsudik (Eds.), *Applied Cryptography and Network Security*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 327–344.
 - [19] W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang, I. Verbauwhede, Rectangle: a bit-slice lightweight block cipher suitable for multiple platforms, *Cryptology ePrint Archive* (2014).
 - [20] J. Guo, T. Peyrin, A. Poschmann, M. Robshaw, The led block cipher, in: B. Preneel, T. Takagi (Eds.), *Cryptographic Hardware and Embedded Systems - CHES 2011*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 326–341.
 - [21] M. R. Albrecht, B. Driessen, E. B. Kavun, G. Leander, C. Paar, T. Yalcın, Block ciphers – focus on the linear layer (feat. pride), in: J. A. Garay, R. Gennaro (Eds.), *Advances in Cryptology – CRYPTO 2014*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 57–76. doi:https://doi.org/10.1007/978-3-662-44371-2_4.
 - [22] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, T. Yalcın, Prince – a low-latency block cipher for pervasive computing applications, in: X. Wang, K. Sako (Eds.), *Advances in Cryptology – ASIACRYPT 2012*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 208–225.
 - [23] A. Poschmann, Lightweight cryptography - cryptographic engineering for a pervasive world, *Cryptology ePrint Archive, Paper 2009/516*, <https://eprint.iacr.org/2009/516> (2009).
URL <https://eprint.iacr.org/2009/516>
 - [24] Y. Dai, S. Chen, Cryptanalysis of full PRIDE block cipher, <https://eprint.iacr.org/2014/987.pdf> (2016).
 - [25] D. Augot, M. Finiasz, Direct construction of recursive mds diffusion layers using shortened bch codes (2014). doi:10.48550/ARXIV.1412.4626.
URL <https://arxiv.org/abs/1412.4626>
 - [26] B. Lac, M. Beunardeau, A. Canteaut, J. J. A. Fournier, R. Sirdey, A first dfa on pride: From theory to practice, in: F. Cuppens, N. Cuppens, J.-L. Lanet, A. Legay (Eds.), *Risks and Security of Internet and Systems*, Springer International Publishing, Cham, 2017, pp. 214–238.
 - [27] Xilinx, Spartan3A FPGA Family, Data Sheet, <https://docs.xilinx.com/v/u/en-US/ds529>.
 - [28] xilinx, Spartan6 FPGA Configuration, <https://docs.xilinx.com/v/u/en-US/ug380>.
 - [29] C. A. Lara-Nino, A. Diaz-Perez, M. Morales-Sandoval, Lightweight hardware architectures for the present cipher in fpga, *IEEE Transactions on Circuits and Systems I: Regular Papers* 64 (9) (2017) 2544–2555. doi:10.1109/TCSI.2017.2686783.
 - [30] nbspTadashi Okabe, Fpga implementation and evaluation of lightweight block cipher - boron, *International Journal of Engineering Development and Research* 5 (2017) 207–216.
 - [31] T. Xu, J. B. Wendt, M. Potkonjak, Security of iot systems: Design challenges and opportunities, in: *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 417–423. doi:10.1109/ICCAD.2014.7001385.
 - [32] I. 14443-2:2010, Identification cards — Contactless integrated circuit cards — Proximity cards — Part 2: Radio frequency power and signal interface (2010).
 - [33] G. Bansod, A. Patil, S. Sutar, N. Pisharoty, Anu: an ultra lightweight cipher design for security in iot, *Security and Communication Networks* 9 (18) (2016) 5238–5251. arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1002/sec.1692>, doi:<https://doi.org/10.1002/sec.1692>.
URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/sec.1692>
 - [34] N. Hanley, M. O'Neill, Hardware comparison of the iso/iec 29192-2 block ciphers, in: *2012 IEEE Computer Society Annual Symposium on VLSI*, 2012, pp. 57–62. doi:10.1109/ISVLSI.2012.25.
 - [35] A. Mhaouch, Elhamzi, W., Abdelali, A.B., M. Atri, Optimized piccolo lightweight block cipher: Area efficient implementation, in: *Traitement du Signal*, Vol. 39, No. 3, pp. 805-814, 2022, pp. 1–6. doi:10.18280/ts.390305.
 - [36] N. Shrivastava, B. Acharya, Fpga implementation of rectangle block cipher architectures (Aug. 2019). doi:10.35940/ijitee.g5481.0881019.
URL <http://dx.doi.org/10.35940/ijitee.g5481.0881019>

A. Appendix: Serialized Proposed Architectures for 16-bit and 32-bit

Figures 18 to 21 depict the PRIDE cipher’s serialized architecture for 16-bit and 32-bit datapaths. It’s an open-ended research problem to optimize datapath, resulting in fewer gate counts.

Table 9
Latency Comparison of the Proposed Architectures

Figure	Data Size	Key Size	Datapath Size	Latency (Clock Cycles)					
				Load	Pre-Whitening	Round R	Round R'	Post-Whitening	Total
18, 19	64	128	32	2	2	19*4	2	2	84
20, 21	64	128	16	4	4	19*8	4	4	168

The operations of the proposed architectures are divided into three major steps. Step1 consists of Pre-Whitening operations, Step2 consists of Round Structure operations, and Step3 performs Post-Whitening. Table 9 compares these proposed architectures based on the clock cycle. We hope the cryptography and the hardware community will collaborate on these architectures to increase hardware efficiency.

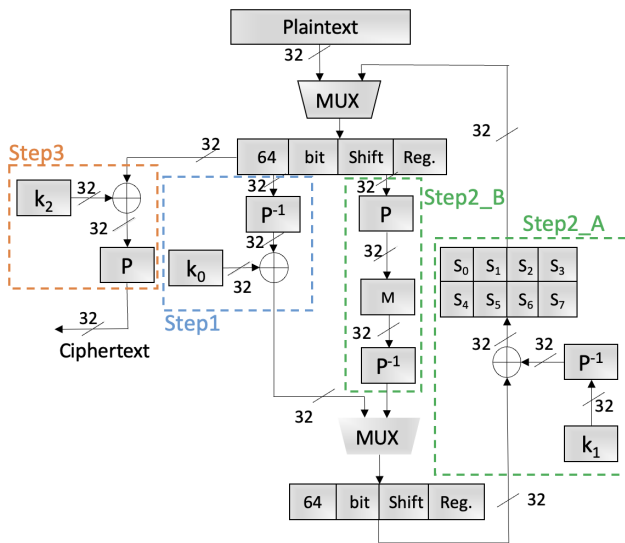


Figure 18: 32-bit Datapath Architecture of PRIDE (Data-Layer)

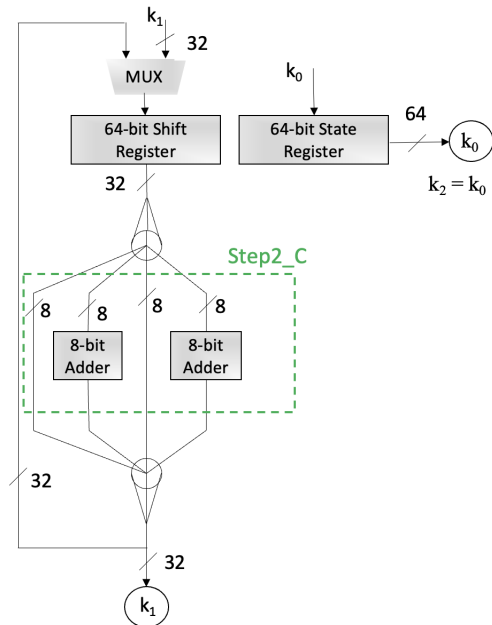


Figure 19: 32-bit Datapath Architecture of PRIDE (Key-Layer)

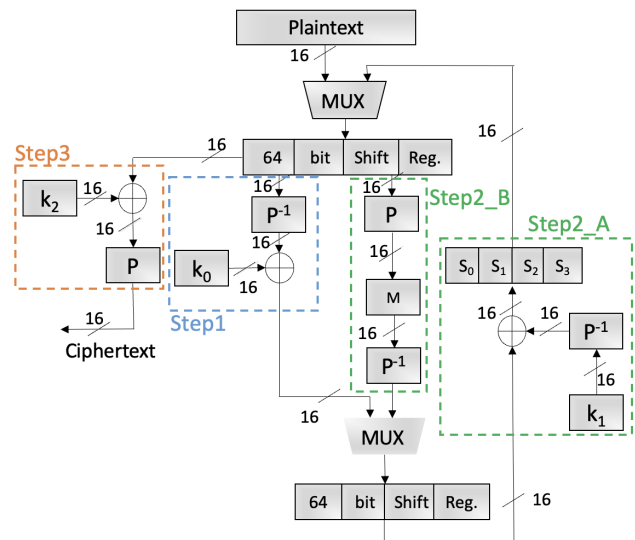


Figure 20: 16-bit Datapath Architecture of PRIDE (Data-Layer)

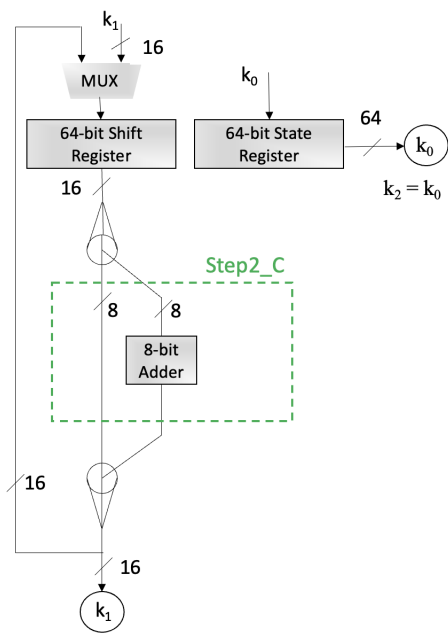


Figure 21: 16-bit Datapath Architecture of PRIDE (Key-Layer)