Jurnal Komputasi

Vol 14 No. 1, 2026

Implementasi Algoritma Kriptografi Ringan PRIDE dan PRESENT untuk Pengamanan Perangkat *Internet of Things*

[1,*)]Allwine, [2]Wahyu Aji Pulungan, [3]Indriyani Talitha Putri

[1,2,3] Ilmu Komputer, Universitas Lampung, Bandar Lampung, Lampung, Indonesia

Abstrak — Meningkatnya penggunaan perangkat yang terhubung ke internet dengan konsumsi daya rendah dan biaya yang efisien telah menimbulkan kebutuhan akan mekanisme kriptografi ringan. Salah satu algoritma enkripsi blok ringan yang dirancang untuk memenuhi kebutuhan tersebut adalah PRIDE, yang dikembangkan oleh Martin R. Albrecht. Algoritma ini merupakan salah satu *cipher* yang efisien untuk lingkungan *Internet of Things* (IoT) yang memiliki keterbatasan sumber daya. PRIDE sangat sesuai digunakan pada perangkat yang saling terhubung karena membutuhkan sumber daya implementasi yang relatif lebih sedikit serta mampu memberikan kinerja yang tinggi. PRIDE merupakan algoritma enkripsi ringan yang berorientasi pada perangkat lunak dan dioptimalkan untuk implementasi pada mikrokontroler. Penelitian ini berfokus pada implementasi algoritma enkripsi PRIDE pada *Field Programmable Gate Array* (FPGA), yaitu suatu perangkat keras digital berupa sirkuit terpadu yang dapat diprogram ulang oleh pengguna untuk mengimplementasikan fungsi logika tertentu secara fleksibel. Evaluasi dilakukan dengan mempertimbangkan metrik kinerja utama, yaitu throughput, konsumsi energi, dan konsumsi daya, penelitian ini juga menyajikan representasi diagramatik yang lebih sederhana dan baru dari implementasi *Matrix Layer* pada algoritma enkripsi PRIDE. Sebagai pembanding, algoritma enkripsi PRESENT juga diimplementasikan dengan menggunakan metrik evaluasi yang sama. Analisis dilakukan terhadap berbagai metrik desain pada FPGA serta membandingkan hasil implementasi PRIDE dengan algoritma PRESENT yang telah banyak dikenal. Hasil analisis tersebut memberikan wawasan mengenai tingkat efisiensi dan keandalan algoritma PRIDE dalam lingkungan IoT yang memiliki keterbatasan sumber daya. Secara spesifik, penelitian ini bertujuan untuk mengevaluasi kinerja implementasi PRIDE pada FPGA, mengidentifikasi efisiensi penggunaan sumber daya perangkat keras, serta membandingkannya dengan algoritma PRESENT. Hasil yang diperoleh menunjukkan bahwa PRIDE memiliki tingkat efisiensi yang kompetitif dalam hal *throughput* dan konsumsi daya, serta layak diimplementasikan pada perangkat IoT berbasis FPGA. Selain itu, penelitian ini juga mengusulkan beberapa arsitektur implementasi algoritma PRIDE untuk jalur data 16-bit dan 32-bit sebagai alternatif desain yang adaptif terhadap kebutuhan sistem.

Kata Kunci: Algoritma Enkripsi; Implementasi FPGA; Keamanan IoT; *PRESENT Cipher; PRIDE Cipher*

Abstract — *The increasing use of internet-connected devices with low power consumption and cost efficiency has created a demand for lightweight cryptographic mechanisms. One of the lightweight block cipher algorithms designed to address this need is PRIDE, developed by Martin R. Albrecht. This algorithm is an efficient cipher for Internet of Things (IoT) environments, which are characterized by resource constraints. PRIDE is well-suited for interconnected devices as it requires relatively low implementation resources while maintaining high performance. It is a software-oriented lightweight encryption algorithm optimized for implementation on microcontrollers. This study focuses on the implementation of the PRIDE encryption algorithm on a Field Programmable Gate Array (FPGA), which is a reconfigurable digital hardware device consisting of programmable logic blocks that can be customized by users to perform specific logical functions. The evaluation considers key performance metrics, including throughput, energy consumption, and power consumption. In addition, this study presents a simplified and novel diagrammatic representation of the Matrix Layer implementation in the PRIDE encryption algorithm. As a comparison, the PRESENT encryption algorithm is also implemented using the same evaluation metrics. The analysis is conducted on various FPGA design metrics and compares the implementation results of PRIDE with the widely recognized PRESENT algorithm. The findings provide insights into the efficiency and reliability of the PRIDE algorithm in resource-constrained IoT environments.Specifically, this study aims to evaluate the performance of PRIDE implementation on FPGA, identify the efficiency of hardware resource utilization, and compare it with the PRESENT algorithm. The results indicate that PRIDE demonstrates competitive efficiency in terms of throughput and power consumption, making it suitable for FPGA-based IoT devices. Furthermore, this study proposes several architectural implementations of the PRIDE algorithm for 16-bit and 32-bit datapaths as adaptive design alternatives to meet varying system requirements.*

Keywords:Encryption Algorithm; FPGA Implementation; IoT Security; PRESENT Cipher; PRIDE Cipher

** Corresponding author :*
Allwine
Universitas Lampung, Bandar Lampung, Indonesia
allwine@fmipa.unila.ac.id

1. PENDAHULUAN

Kemajuan teknologi semikonduktor mendorong berkembangnya perangkat IoT yang kecil, *mobile*, dan hemat daya, namun rentan terhadap ancaman keamanan data. Oleh karena itu, diperlukan mekanisme kriptografi yang mampu menjamin kerahasiaan dan integritas data pada perangkat dengan sumber daya terbatas. Algoritma konvensional seperti AES [1] dan DES [2] memiliki keamanan tinggi, tetapi kurang efisien untuk perangkat terbatas karena kompleksitasnya. Hal ini melatarbelakangi pengembangan kriptografi ringan yang lebih sesuai untuk IoT. Berbagai algoritma seperti PRESENT [6], HIGHT [7], PRINCE [8], SIMON dan SPECK [10], serta lainnya telah dikaji, terutama pada implementasi FPGA yang fleksibel dan efisien daya.Efisiensi implementasi perangkat keras sangat dipengaruhi oleh desain *datapath*, yang berdampak pada area, konsumsi daya, dan *throughput*. Namun, studi komparatif yang mengeksplorasi datapath besar (misalnya 64-bit) untuk mencapai keseimbangan performa dan efisiensi energi masih terbatas. Penelitian ini mengusulkan implementasi *datapath* 64-bit pada algoritma PRIDE [21] dan PRESENT [13] di FPGA. Kontribusi utama meliputi: (1) desain *datapath* 64-bit untuk performa tinggi, (2) analisis komparatif dua cipher dengan struktur berbeda, dan (3) evaluasi trade-off antara *throughput*, daya, dan efisiensi energi. PRESENT dikenal efisien namun membutuhkan banyak ronde, sedangkan PRIDE menawarkan struktur lebih ringan dengan ronde lebih sedikit dan efisiensi implementasi lebih baik. Dengan demikian, penelitian ini bertujuan menghasilkan referensi optimal dalam pemilihan kriptografi ringan untuk aplikasi IoT berbasis sumber daya terbatas.

2. METODOLOGI PENELITIAN

Penelitian ini mengusulkan dan mengimplementasikan *datapath* 64-bit dari algoritma *block cipher PRIDE* pada *platform Field Programmable Gate Array* (FPGA). Selain itu, penelitian ini juga mengevaluasi nilai *Gate Equivalents* (GE) dari desain cipher *PRIDE* pada arsitektur *datapath 64-bit*. Sepengetahuan kami, penelitian ini merupakan implementasi pertama algoritma *PRIDE* pada FPGA yang disertai dengan analisis metrik kinerja secara rinci. Dalam penelitian ini juga diusulkan beberapa arsitektur untuk berbagai konfigurasi *datapath*, yaitu 16-bit dan 32-bit. Arsitektur yang diusulkan tersebut memberikan wawasan bagi pengguna serta mempermudah proses implementasi berbagai desain arsitektur sesuai dengan kebutuhan aplikasi pada lingkungan yang memiliki keterbatasan sumber daya. Arsitektur *PRIDE* berbasis *datapath* 64-bit yang diusulkan dan diimplementasikan dalam penelitian ini menunjukkan efisiensi yang tinggi, yang ditandai dengan throughput yang maksimal serta latensi yang lebih rendah. Seluruh hasil implementasi kemudian dibandingkan dengan arsitektur *PRESENT*, yang menunjukkan bahwa cipher *PRIDE* memiliki keunggulan dibandingkan *PRESENT*, terutama dalam hal *throughput*, latensi, dan konsumsi daya. Selain itu, *datapath* dari algoritma *PRESENT* juga diimplementasikan pada platform yang sama untuk memastikan bahwa proses perbandingan yang dilakukan bersifat adil dan valid. Penelitian ini tidak hanya menyajikan implementasi perangkat keras dari algoritma cipher *PRIDE*, tetapi juga memberikan wawasan mengenai berbagai trade-off antara penggunaan *lookup tables*, *Gate Equivalents* (GE), dan *throughput*. Dalam lingkungan dengan keterbatasan sumber daya seperti *Internet of Things* (IoT), implementasi *datapath* yang efisien memegang peranan yang sangat penting, yang telah berhasil dibahas dan dianalisis dalam penelitian ini. Oleh karena itu, penelitian ini memberikan kontribusi terhadap implementasi perangkat keras secara komprehensif dari algoritma kriptografi ringan PRIDE, serta melakukan perbandingan berbagai metrik desain implementasi perangkat keras antara *PRIDE* dan algoritma kriptografi ringan *PRESENT*. Selain itu, penelitian ini juga memberikan wawasan kepada pembaca mengenai berbagai desain datapath serta metrik desain yang terkait dengan masing-masing arsitektur tersebut.

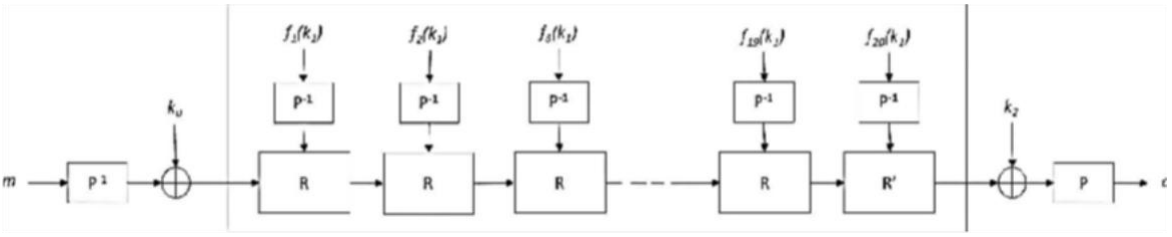
2.1. Algoritma *Block Cipher* Ringan *PRIDE*

Algoritma *lightweight block cipher PRIDE* merupakan salah satu cipher dengan performa tinggi dan efisiensi yang dirancang khusus untuk perangkat *Internet of Things* (IoT). Algoritma ini menggunakan struktur berbasis *Substitution-Permutation Network* (SPN) dengan lapisan linear yang dirancang secara khusus untuk meningkatkan efisiensi penggunaan sumber daya. Fungsi ronde (*round function*) pada cipher *PRIDE* terdiri atas kombinasi lapisan fungsi linear dan non-linear, yang meliputi *S-Box* berukuran 4-bit, *Matrix Layer*, serta *bit-permutation*. *PRIDE* menggunakan ukuran *plaintext* sebesar 64-bit dan ukuran kunci sebesar 128-bit. Serupa dengan algoritma *PRINCE* [22], cipher ini menggunakan konstruksi *FX construction*. Pada proses ini, kunci *pre-whitening* yaitu k_0 dan k_2 dihasilkan dari *64 Least Significant Bits* (LSB) dari kunci utama k, sedangkan 64 *Most Significant Bits* (MSB) dari kunci tersebut digunakan sebagai dasar pembentukan round key k_1.

$k=k_0||k_1$

$k_2=k_0$

Cipher ini dimulai dan diakhiri dengan operasi *bit-permutation*, yang menghasilkan efisiensi yang lebih baik pada implementasi *bit-sliced*. Cipher ini memiliki 20 ronde, yang terdiri atas 19 ronde identik dan satu ronde yang berbeda. Struktur berbasis ronde dari PRIDE ditunjukkan pada Gambar 1.



Gambar 1.Struktur PRIDE.

2.2. Penjadwalan Kunci

PRIDE memiliki total 20 ronde sub-kunci yang unik. Sub-kunci unik untuk setiap ronde dihasilkan dengan menambahkan round constants ke bagian-bagian dari kunci utama. Sub-kunci untuk ronde ke-iii dapat diperoleh melalui fungsi $f_i(k)$ sebagaimana ditunjukkan pada Persamaan (1).

$$f_{i(k_1)} = k_{\{10\}} \| g_0^{\{(i)\}(k_{\{11\}})} \| k_{\{12\}} \| k_{\{13\}} \| g_1^{\{(i)\}(k_{\{13\}})} \| k_{\{14\}} \| g_2^{\{(i)\}(k_{\{15\}})} \| k_{\{16\}} \| g_3^{\{(i)\}(k_{\{17\}})} \quad (1)$$

Keterangan:

$g_0^{\{(i)\}(x)} = (x + 193 \times i) \backslash mod\ 256$

$g_1^{\{(i)\}(x)} = (x + 165 \times i) \backslash mod\ 256$

$g_2^{\{(i)\}(x)} = (x + 81 \times i) \backslash mod\ 256$

$g_3^{\{(i)\}(x)} = (x + 197 \times i) \backslash mod\ 256$

Konstanta-konstanta tersebut kemudian ditambahkan ke masing-masing byte dari kunci k_1 untuk melakukan proses algoritma penjadwalan kunci (*key scheduling*) [21].

2.3 *Substitution Box (S-Box)*

Substitution Box (S-Box) merupakan salah satu komponen penting dalam analisis keamanan suatu algoritma cipher [13]. Pada algoritma *PRIDE*, digunakan *S-Box* berukuran 4-bit untuk menghasilkan sifat non-linearitas dalam proses enkripsi. Karena *S-Box* merupakan komponen non-linear utama dalam desain enkripsi, maka *S-Box* harus memiliki tingkat keamanan yang kuat serta memenuhi berbagai kriteria standar *S-Box* yang dijelaskan dalam [23]. Tabel 1 menunjukkan S-Box yang digunakan dalam algoritma cipher *PRIDE*. *S-Box* ini merupakan *involution S-Box*, yang berarti fungsi substitusi yang digunakan pada proses enkripsi juga dapat digunakan pada proses dekripsi. Dengan demikian,

penggunaan *involution S-Box* dapat mengurangi *overhead* komputasi antara proses enkripsi dan dekripsi [21].

Tabel 1. S-Box digunakan untuk 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

2.4 Lapisan Linear

Algoritma *cipher PRIDE* memiliki sebuah *Linear Layer* (L) yang berfungsi untuk meningkatkan difusi pada proses enkripsi. Lapisan linear ini dapat dibagi menjadi tiga sub-lapisan utama, yaitu:

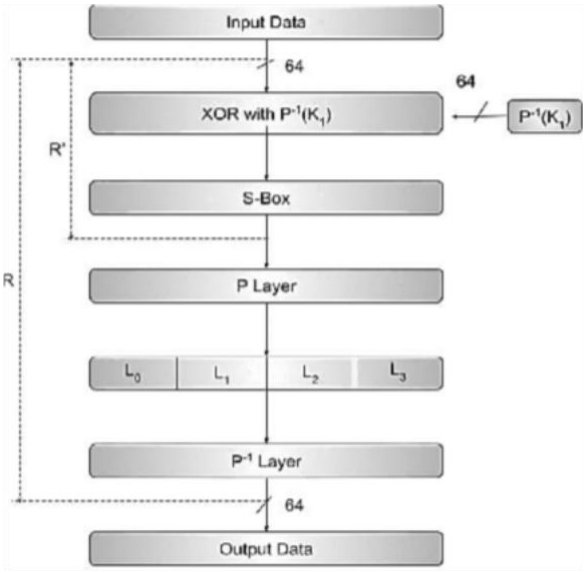
1. *Lapisan Permutasi (Lapisan-P)*
2. *Lapisan Permutasi Terbalik (P^{-1})*
3. *Lapisan Matriks (M)*

Hubungan antara ketiga lapisan tersebut dinyatakan dalam Persamaan (2) sebagai berikut.

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

2.5 Alur Kerja Cipher

Pada tahap awal, algoritma cipher melakukan operasi permutasi P^{-1} terhadap *plaintext*. Hasil dari proses tersebut kemudian dijumlahkan dengan kunci k_0. Setelah itu, cipher menjalankan 20 ronde proses enkripsi sebagaimana telah dijelaskan sebelumnya, kemudian menambahkan kunci k_2 pada hasil yang diperoleh. Selanjutnya, dilakukan operasi bit-permutation terhadap hasil dari 20 ronde tersebut untuk menghasilkan *ciphertext*. Struktur umum dari algoritma *PRIDE* cipher ditunjukkan pada Gambar 4 [26]. Proses difusi tidak diperlukan pada tahap terakhir dari cipher.



Gambar 4. Struktur Putaran dari PRIDE Cipher [26].

2.6 Implementasi pada FPGA

Field Programmable Gate Array (FPGA) merupakan *Integrated Circuit* (IC) yang dapat dikonfigurasi ulang dan terdiri dari berbagai komponen seperti rangkaian *Flip-Flops* (F-F), rangkaian *Slices*, *Lookup Tables* (LUTs), serta rangkaian *routing switches*. Pengguna dapat mengonfigurasi IC ini sesuai dengan kebutuhan aplikasi yang akan dijalankan. Perangkat FPGA generasi terbaru menyediakan jumlah flip-flops, slices, dan LUTs yang lebih banyak [27,28], sehingga memungkinkan implementasi arsitektur yang lebih kompleks yang memerlukan area lebih besar. Kinerja implementasi pada FPGA bergantung

pada beberapa metrik, seperti *throughput*, *dynamic power*, *static power*, dan *energy consumption* [29,30]. Seluruh metrik tersebut sangat bergantung pada arsitektur yang digunakan, sehingga berbagai arsitektur perlu dirancang untuk mengatasi tantangan pada lingkungan dengan keterbatasan sumber daya seperti IoT. Namun, mencapai performa optimal pada seluruh metrik tersebut dalam satu arsitektur merupakan tugas yang cukup sulit karena terdapat *trade-off* antara parameter-parameter tersebut [31]. Dalam penelitian ini, algoritma *PRIDE* diimplementasikan pada empat *platform* FPGA menggunakan arsitektur datapath 64-bit berbasis *round*. Hasil implementasi *PRIDE* kemudian dibandingkan dengan algoritma *PRESENT* cipher.

2.6.1 *Platform*

Arsitektur yang diusulkan diimplementasikan pada *Xilinx* FPGA *board* menggunakan perangkat lunak ISE *Design Suite* 14.7. Empat *platform* FPGA yang digunakan dalam penelitian ini adalah: Arsitektur *Spartan-3 (xc3s700an-5fgg484)*, Arsitektur *Spartan-6 (xc6slx45t-3fgg484)*, Arsitektur *Virtex-4 (xc4vlx25-12ff668)*, Arsitektur *Virtex-5 (xc5vlx50t-3ff1136)*. Analisis hasil implementasi dilakukan dengan *speed grade* masing-masing: Arsitektur *Spartan-3 : -5;* Arsitektur *Spartan-6 : -3;* Arsitektur *Virtex-4 : -12;* Arsitektur *Virtex-5 : -3*. Frekuensi standar ISO untuk *smart cards* dan *RFID tags*, yaitu 13.56 MHz, digunakan dalam proses implementasi serta pengukuran metrik performa [32].

2.6.2 Konsumsi Daya

Konsumsi daya merupakan salah satu metrik desain yang sangat penting bagi perangkat IoT yang menggunakan sumber daya baterai. Pada setiap desain rangkaian, terdapat dua jenis konsumsi daya utama, yaitu: Static Power, Dynamic Power. Total konsumsi daya dari suatu implementasi dihitung sebagai penjumlahan dari kedua jenis daya tersebut, sebagaimana dinyatakan pada Persamaan (3).

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

Evaluasi konsumsi daya dilakukan pada frekuensi standar 13.56 MHz [32].

2.6.3 Kecepatan Operasi

Kecepatan operasi atau *throughput* sepenuhnya bergantung pada *latency*. *Latency* merupakan jumlah total siklus *clock* yang dibutuhkan untuk mengenkripsi satu blok data *64-bit*. Oleh karena itu, kecepatan desain yang diusulkan bergantung pada tiga parameter utama, yaitu: *Throughput pada frekuensi 13.56 MHz*, *Throughput maksimum*, *Throughput per slice*. Berbagai rumus digunakan untuk menghitung *throughput* dari implementasi cipher sebagaimana ditunjukkan pada Persamaan (4), (5), dan (6). Ukuran plaintext pada cipher PRIDE adalah 64-bit. *Throughput* dari arsitektur yang diusulkan diharapkan setinggi mungkin, sedangkan *latency* harus serendah mungkin agar desain dapat bekerja dengan cepat.

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

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

$$Maximum\ Throughput\,(Thr) = \frac{(Maximum\ Frequency * Plaintext\ Size)}{} Laten \quad (6)$$

2.6.4 Pemanfaatan Energi

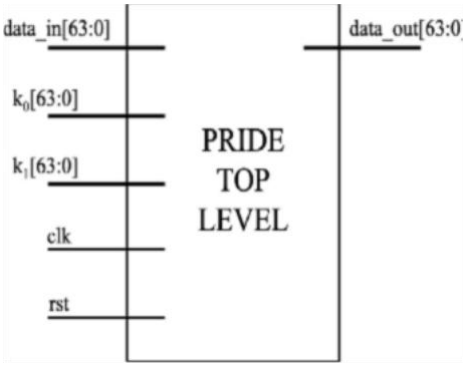
Arsitektur *cipher* yang dirancang seharusnya memiliki konsumsi energi yang rendah, yaitu energi total yang dibutuhkan untuk mengenkripsi satu blok *plaintext* berukuran 64-bit. Konsumsi energi dinyatakan menggunakan dua parameter utama yaitu: *Energy, Energy per bit*. Parameter tersebut dihitung pada frekuensi standar 13.56 MHz menggunakan Persamaan (7) dan (8).

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

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

2.7 Arsitektur yang Diusulkan

Model tingkat tinggi *(high-level model)* dari implementasi algoritma *PRIDE* ditunjukkan pada Gambar 5. Implementasi tersebut memerlukan **258** *Input-Output Blocks (IOBs)* untuk mendukung proses komunikasi antara rangkaian logika internal FPGA dengan perangkat eksternal.



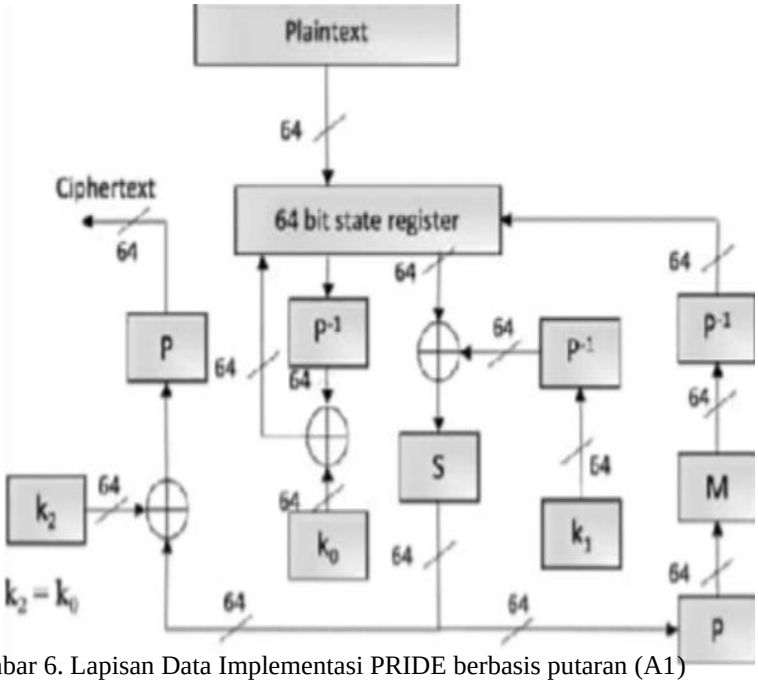
Gambar 5. *high-level model* Implementasi PRIDE

2.7.1 Arsitektur *PRIDE* Berbasis *Round* (A1)

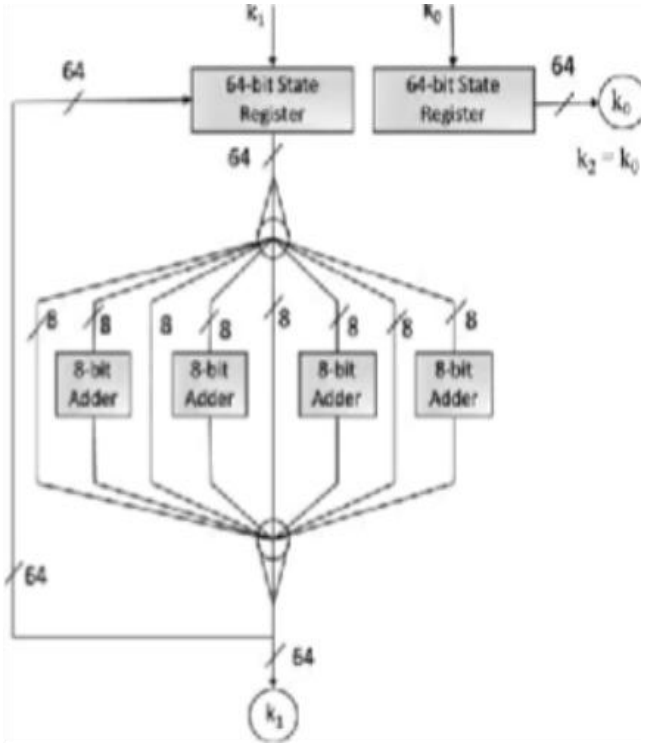
Gambar 6 menunjukkan arsitektur berbasis *round* (A1) untuk cipher *PRIDE* dengan ukuran *datapath* 64-bit. Arsitektur ini dirancang dan dioptimalkan untuk menghasilkan *throughput* setinggi mungkin selama proses enkripsi data. Arsitektur ini merupakan desain berbasis *round* yang menggunakan beberapa *S-Box, adder 8-bit,* serta komponen lainnya secara paralel untuk mendukung proses enkripsi yang cepat.

2.7.1.1 Operasi

Pada tahap awal, data disimpan dalam *register state 64-bit*. Operasi desain dibagi menjadi tiga tahap utama, yaitu: *Pre-whitening, Identical round, Post-whitening***.** Perlu dicatat bahwa implementasi berbasis *round* ini tidak memerlukan *multiplexer* untuk menyimpan data [13].



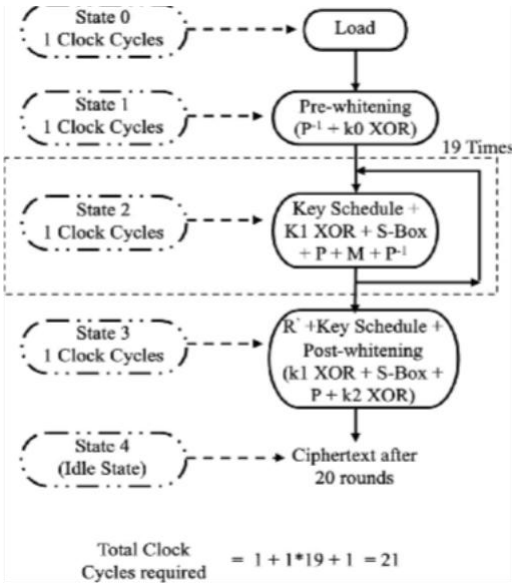
Gambar 6. Lapisan Data Implementasi PRIDE berbasis putaran (A1)



Gambar 7. Lapisan Kunci Implementasi PRIDE berbasis putaran.

2.7.1.2 Siklus *Clock* dan *State*

Gambar 8 menunjukkan jumlah siklus *clock* dan *state* yang diperlukan untuk implementasi berbasis round dari cipher PRIDE. Implementasi ini melibatkan lima *state*, di mana setiap tahap dieksekusi dalam satu siklus *clock*. Dari lima state tersebut: *State* 1 digunakan untuk proses *pre-whitening***,** *State* 2 digunakan untuk *identical rounds* dan dijalankan sebanyak 19 kali**,** *State* 3 digunakan untuk *post-whitening* sekaligus operasi *round* terakhir**.** Dengan demikian, keseluruhan implementasi cipher PRIDE memerlukan 21 siklus *clock*, yang menghasilkan *throughput* yang lebih tinggi pada arsitektur PRIDE yang diusulkan.



Gambar 8.Diagram Analisis Siklus

2.7.1.3 Persamaan Gerbang Logika (GE)

Persamaan Gerbang Logika dari arsitektur yang diusulkan diperkirakan menggunakan *standard ASIC ARM-7 library IBM 8RF* dengan teknologi 0.130-micron. Seperti ditunjukkan pada Tabel 2 [33].

Tabel 2. Nilai Standar Library

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

pustaka ini menentukan nilai GE untuk berbagai komponen logika seperti: *logic gates, flip-flops, multiplexers*, serta komponen logika lainnya. Tabel 3,

Tabel 3. Perhitungan GE S-Box Tunggal

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

Tabel 4. Perhitungan GE *Data Layer* untuk penerapan 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

dan Tabel 5,

Tabel 5. Perhitungan GE Key Layer untuk penerapan 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

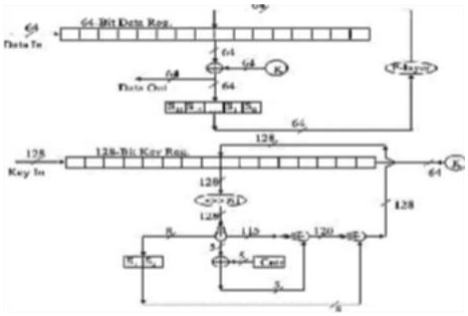
menunjukkan jumlah GE yang dibutuhkan untuk: *S-Box*, *Data Layer*, *Key Layer* dari arsitektur yang diusulkan. Cipher PRIDE menggunakan *S-Box* dengan *input* 4-*bit* [I_3, I_2, I_1, I_0dan *output* 4-bit [O_3, O_2, O_1, O_0]. Persamaan (9) digunakan untuk menghitung GE yang diperlukan untuk satu S-Box.

$$O_3 = I_1 \oplus (I_3 \& I_2)$$
$$O_2 = I_0 \oplus (I_2 \& I_1)$$
$$O_1 = I_3 \oplus (O_3 \& O_2)$$
$$O_0 = I_2 \oplus (O_2 \& O_1) \quad (9)$$

Implementasi round-based dari cipher PRIDE memerlukan total 1056 + 728 = 1784 GE.Perlu dicatat bahwa GE pada Matrix Layer dihitung menggunakan Gambar 2 dan Gambar 3.

2.8 Arsitekur PRESENT

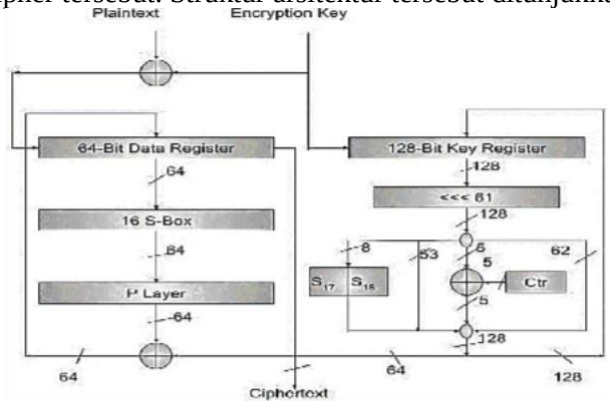
Gambar 9 menunjukkan arsitektur yang diusulkan dari cipher PRESENT [13]. Arsitektur ini sangat mirip dengan arsitektur A3 yang diusulkan pada penelitian [34], dan diimplementasikan dengan cara yang sama seperti arsitektur A1. Pendekatan ini digunakan untuk memastikan perbandingan yang adil antara kedua cipher pada platform yang dipilih.



Gambar 9.Arsitektur PRESENT yang diusulkan untuk perbandingan

2.9 Arsitektur Iterative

Arsitektur ini diusulkan dalam penelitian [34]. Cipher PRIDE diimplementasikan menggunakan arsitektur berbasis round (*iterative architecture*), dan arsitektur PRESENT yang serupa dipilih untuk membandingkan kedua cipher tersebut. Struktur arsitektur tersebut ditunjukkan pada Gambar 10 [29].



Gambar 10. Arsitektur Iteratif dari PRESENT yang diusulkan [34]

Tabel 6 menyajikan deskripsi berbagai arsitektur yang digunakan dalam proses perbandingan.

Tabel 6. Deskripsi Arsitektur yang Berbeda

Architecture	Cipher	Data path size	Data Size	Key Size	Refer-ence	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

3. HASIL DAN PEMBAHASAN

Seluruh hasil implementasi dari arsitektur yang diusulkan ditunjukkan pada Tabel 7 dan Tabel 8. Tabel 7 membandingkan hasil area dan *throughput*,

Tabel 7. Perbandingan Luas Area dan Kapasitas

Architecture	Device	Data Size (bit)	Key Size (bit)	F-F	LUTs	Slices	Clock Cycles	Max. Freq (MHz)	Thr @ Max (Mbps)	Thr @ 13.56 (Mbps)	Thr per Slice (Kbps)
A1	Spartan-3 xc3s400	64	128	198	620	360	22	165.45	480.62	39.46	109.61
A2	Spartan-3 xc3s400	64	128	205	598	348	31	230.12	474.21	27.44	78.88
A3	Virtex-4 xc4vlx25	64	128	228	702	395	21	305.78	931.14	39.46	99.98
A1	Virtex-5 xc5vlx50	64	128	236	610	332	21	458.90	964.02	28.75	86.58
A2	Spartan-6 xc6slx45	64	128	210	382	120	55	270.34	315.10	16.12	94.65
A3	Spartan-6 xc6slx45	64	128	199	250	92	21	215.67	643.18	39.46	356.02
A1	Virtex-4 xc4vlx25	64	128	208	365	185	55	272.15	317.46	16.12	88.90
A2	Virtex-5 xc5vlx50	64	128	214	380	170	31	468.73	942.64	27.44	161.44
A3	Virtex-5 xc5vlx50	64	128	206	285	90	55	268.91	312.88	16.12	173.20

PRESENT	Spartan-3 xc3s400	64	128	200	310	168	55	226.42	268.73	15.12	89.93
ITERATIVE	XC5T256	64	128	198	255	130	303	160.72	33.45	2.84	21.70
Serial (5)	Spartan-3 xc3s200	64	128	206	540	282	62	70.55	72.80	14.02	46.95
Serial (3)	Spartan-3 xc3s200	64	128	248	560	305	62	49.10	50.22	14.02	44.05
LED	Virtex-4 xc4vlx25	64	128	80	440	220	48	96.75	128.05	18.15	74.78
RECTANGLE	Spartan-3 xc3s200	64	128	188	580	315	27	265.30	655.78	33.15	104.99
HIGHT	Spartan-3 xc3s200	64	128	194	505	268	25	248.44	646.55	34.25	126.01
PRINCE	Virtex-6 xc6vlx75	64	128	-	-	820	-	168.33	508.21	-	-
RECTANGLE A2	Spartan-3 xc3s200	64	128	200	365	210	48	105.40	140.02	18.02	83.40

sedangkan Tabel 8 membandingkan konsumsi energi dan konsumsi daya dari berbagai arsitektur, yaitu A1, A2, dan A3.Arsitektur A1 dan A2 dievaluasi pada platform yang digunakan dalam penelitian ini, sedangkan arsitektur A3 diimplementasikan pada penelitian sebelumnya [34].

Tabel 8. Perbandingan Konsumsi Daya dan Energi dari Berbagai Arsitektur

Architecture	Device	Data Size (bit)	Key Size (bit)	Latency	Static Pow. (mW)	Dynamic Pow. (mW)	Total Pow. (mW)	E* (µJ)	E* per bit (nJ/bit)
A1	Spartan-3 xc3s500	64	128	22	34.80	38.10	72.90	0.121	1.892
A2	Spartan-3 xc3s500	64	128	32	33.75	7.20	40.95	0.096	1.503
A3	Virtex-4 xc4vlx30	64	128	54	220.45	17.30	237.75	1.415	22.109
A1	Virtex-5 xc5vlx50	64	128	22	229.10	26.10	255.20	0.405	6.328
A2	Virtex-5 xc5vlx50	64	128	32	230.55	14.95	245.50	0.581	9.072
A3	Virtex-5 xc5vlx50	64	128	54	341.80	16.05	357.85	1.460	22.812
A1	Spartan-6 xc6slx45	64	128	22	35.10	12.25	47.35	0.076	1.188
A2	Spartan-6 xc6slx45	64	128	32	5.10	14.10	19.20	0.083	1.297
A3	Spartan-6 xc6slx45	64	128	54	558.90	21.40	580.30	0.915	14.298
A1	Virtex-4 xc4vlx30	64	128	32	561.40	22.15	583.55	1.350	21.094
A2	Virtex-4 xc4vlx30	64	128	54	560.75	4.20	564.95	2.301	35.954
RECTANGLE	Spartan-3 xc3s200	64	128	48	260.50	6.15	266.65	0.982	0.091
RECTANGLE	Spartan-3 xc3s200	64	128	26	35.25	14.20	49.45	0.101	1.575
ANU	Virtex-4 xc4vlx30	64	128	25	34.90	7.10	42.00	0.081	1.266

Karena arsitektur PRIDE yang diusulkan menggunakan arsitektur berbasis round, maka arsitektur iterative (round-based) dari PRESENT pada penelitian [34] (A3) dipilih sebagai pembanding dalam penelitian ini.

4. KESIMPULAN

Penelitian ini menunjukkan bahwa implementasi *block cipher* PRIDE 64-bit pada *platform* FPGA memiliki kinerja yang lebih unggul dibandingkan dengan PRESENT 64-bit, terutama ditinjau dari aspek *throughput* dan *latency*. Peningkatan kinerja tersebut dipengaruhi oleh jumlah ronde enkripsi PRIDE yang lebih sedikit, sehingga menghasilkan proses komputasi yang lebih efisien. Meskipun demikian, implementasi PRIDE memerlukan sumber daya area yang sedikit lebih besar. Dari sisi *platform*, Spartan-6 memberikan efisiensi penggunaan area yang lebih baik, sedangkan Virtex-5 mampu menghasilkan *throughput* maksimum dengan konsumsi daya dinamis yang relatif rendah. Implikasi praktis dari temuan ini menunjukkan bahwa PRIDE lebih sesuai untuk aplikasi berbasis perangkat keras yang menuntut kinerja tinggi, sementara arsitektur dengan *datapath* lebih kecil (16-bit dan 32-bit) dapat menjadi alternatif implementasi pada perangkat dengan keterbatasan sumber daya. Selain itu, arsitektur yang diusulkan dalam penelitian ini masih memiliki potensi untuk dikembangkan lebih lanjut guna mencapai optimasi yang lebih baik antara kinerja, konsumsi daya, dan penggunaan area.

DAFTAR PUSTAKA

[1] Dworkin, M., Barker, E., Nechvatal, J., Foti, J., Bassham, L., Roback, E., & Dray, J. Advanced Encryption Standard (AES), NIST FIPS PUB 197. (2001).

[2] National Institute of Standards and Technology. (1999). *Data Encryption Standard (DES).*

[3] Kavun, E. B. (2014). *Resource-efficient cryptography for ubiquitous computing.*

[4] Dahiphale, V., Bansod, G., Zambare, A., & Pisharoty, N. (2020). *Design and implementation of various datapath architectures for the ANU lightweight cipher on an FPGA.* Frontiers of Information Technology & Electronic Engineering, 21, 615–628.

[5] Dahiphale, V., Raut, H., & Bansod, G. (2019). *Novel datapath designs of lightweight cipher RECTANGLE.* Multimedia Tools and Applications, 78, 23659–23688.

[6] Ashaq, S., Nazish, M., Ali, M., Sultan, I., & Banday, M. T. (2022). *FPGA implementation of PRESENT block cipher with optimized substitution box.* Smart Technologies, Communication and Robotics (STCR).

[7] Rashidi, B. (2019). *High-throughput and lightweight hardware structures of HIGHT and PRESENT block ciphers.*

[8] Rashidi, B. (2020). *Low-cost hardware structures of PRINCE lightweight block cipher.*

[9] Rashidi, B. (2020). *Efficient and flexible hardware structures of the CLEFIA block cipher.*

[10] Rashidi, B. (2019). *High-throughput and flexible ASIC implementations of SIMON and SPECK lightweight block ciphers.*

[11] Rashidi, B. (2020). *Flexible structures of lightweight block ciphers PRESENT, SIMON, and LED.*

[12] Rashidi, B. (2021). *Flexible and high-throughput structures of Camellia block cipher for IoT security.*

[13] Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J. B., Seurin, Y., & Vikkelsoe, C. (2007). *PRESENT: An ultra-lightweight block cipher.* In Cryptographic Hardware and Embedded Systems (CHES) (pp. 450–466).

[14] Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., & Wingers, L. (2013). *The SIMON and SPECK families of lightweight block ciphers.*

[15] Suzaki, T., Minematsu, K., Morioka, S., & Kobayashi, E. (2012). *TWINE: A lightweight block cipher.* In Selected Areas in Cryptography (SAC) (pp. 339–354).

[16] Bansod, G., Raval, N., & Pisharoty, N. (2014). *Lightweight encryption design for embedded security.* IEEE Transactions on Information Forensics and Security, 10(1), 142–151.

[17] Dahiphale, V., Bansod, G., & Patil, J. (2017). *ANU-II: A fast and efficient lightweight encryption design.* In International Conference on Big Data, IoT and Data Science (BID).

[18] Wu, W., & Zhang, L. (2011). *LBLOCK: A lightweight block cipher.* In Applied Cryptography and Network Security (ACNS) (pp. 327–344).

[19] Zhang, W., Bao, Z., Lin, D., Rijmen, V., Yang, B., & Verbauwhede, I. (2014). *RECTANGLE: A bit-slice lightweight block cipher.*

[20] Guo, J., Peyrin, T., Poschmann, A., & Robshaw, M. (2011). *The LED block cipher.* In Cryptographic Hardware and Embedded Systems (CHES) (pp. 326–341).

[21] Albrecht, M. R., Driessen, B., Kavun, E. B., Leander, G., Paar, C., & Yalçın, T. (2014). *Block ciphers—Focus on the linear layer (PRIDE).* In Advances in Cryptology (CRYPTO).

[22] Borghoff, J., Canteaut, A., Güneysu, T., Kavun, E. B., Knezevic, M., Knudsen, L. R., Leander, G., Nikov, V., Paar, C., Rechberger, C., Rombouts, P., Thomsen, S. S., & Yalçın, T. (2012). *PRINCE: A low-latency block cipher.* In Advances in Cryptology (ASIACRYPT).

[23] Poschmann, A. (2009). *Lightweight cryptography.*

[24] Dai, Y., & Chen, S. (2016). *Cryptanalysis of full PRIDE block cipher.*

[25] Augot, D., & Finiasz, M. (2014). *Direct construction of recursive MDS diffusion layers.*

[26] Lac, B., Beunardeau, M., Canteaut, A., Fournier, J. J. A., & Sirdey, R. (2017). *A first DFA on PRIDE: From theory to practice.*

[27] Xilinx. (n.d.). *Spartan-3A FPGA family datasheet.*

[28] Xilinx. (n.d.). *Spartan-6 FPGA configuration guide.*

[29] Lara-Niño, C. A., Diaz-Perez, A., & Morales-Sandoval, M. (2017). *Lightweight hardware architectures for the PRESENT cipher in FPGA.* IEEE Transactions on Circuits and Systems I, 64(9), 2544–2555.

[30] Okabe, T. (2017). *FPGA implementation and evaluation of lightweight block cipher BORON.* International Journal of Engineering Development and Research, 5.

[31] Xu, T., Wendt, J. B., & Potkonjak, M. (2014). *Security of IoT systems: Design challenges and opportunities.* In IEEE/ACM International Conference on Computer-Aided Design (ICCAD).

[32] ISO/IEC. (2010). *ISO/IEC 14443-2: Identification cards.*

[33] Bansod, G., Patil, A., Sutar, S., & Pisharoty, N. (2016). *ANU: An ultra-lightweight cipher design for IoT.* Security and Communication Networks, 9(18), 5238–5251.

[34] Hanley, N., & O'Neill, M. (2012). *Hardware comparison of ISO/IEC block ciphers.* In IEEE Computer Society Annual Symposium on VLSI (ISVLSI).

[35] Mhaouch, A., Elhamzi, W., Abdelali, A. B., & Atri, M. (2022). *Optimized Piccolo lightweight block cipher: Area-efficient implementation.* Traitement du Signal, 39(3).

[36] Shrivastava, N., & Acharya, B. (2019). *FPGA implementation of RECTANGLE block cipher architectures.*