A Survey on Lightweight Block Ciphers

Prabhat Kumar Kushwaha
Computer Science and Engineering
National Institute of Technology Patna
India

M. P. Singh
Computer Science and Engineering
National Institute of Technology Patna
India

Prabhat Kumar
Computer Science and Engineering
National Institute of Technology Patna
India

ABSTRACT

Ubiquitous and pervasive computing are new era of computing and it needs lightweight cryptographic algorithms for security. Lightweight cryptography is used for resource constrained devices (which have limited memory, limited power and less processing capability) such as radio frequency identification (RFID) tags, contactless smart cards, wireless sensor network, health care devices and internet of things (IoT). The design of lightweight block cipher has been active research topic over the years. The comparative evaluations of these block ciphers (which actually reach low cost goal) on any platform is hard. In this paper comparative evaluation of selected symmetric key lightweight block ciphers such as PRINT, PRESENT, EPCBC, DESL, TWINE, Puffin, KLEIN, KATAN, LED, LBLOCK and RECTANGLE is presented.

General Terms:

Cryptography, Symmetric key.

Keywords:

Lightweight block cipher, PRESENT, DESL, PRINT, EPCBC.

1. INTRODUCTION

Ubiquitous and pervasive computing are new era of computing and it needs lightweight cryptographic algorithms for security. Keeloq [1] is first lightweight block cipher which has 40 bits key size, 40 bits block size and used Feistel structure. Lightweight block ciphers rely more on basic operations such as bit wise XOR, bit wise AND, 4 * 4 bits S-box etc. leading to increase of required number of rounds; Lightweight block ciphers most of the times extremely simplify the key schedule due to memory requirements. Serialized architecture is used to implement lightweight block ciphers.

Resource constrained devices (devices with limited memory, limited power and lesser processing capability) such as radio frequency identification (RFID) tags [14], contactless smart cards, wireless sensor network, health care devices [16] and internet of things (IoT) [13] are now deployed in many applications. For security they need lightweight cryptographic algorithms.

Evaluations of cryptographic algorithms are based on the cost, performance and security. In lightweight block ciphers minimizing the cost is primary concern. That is why cost is used in combination with performance and security to achieve the goal of lightweight block cipher[8].

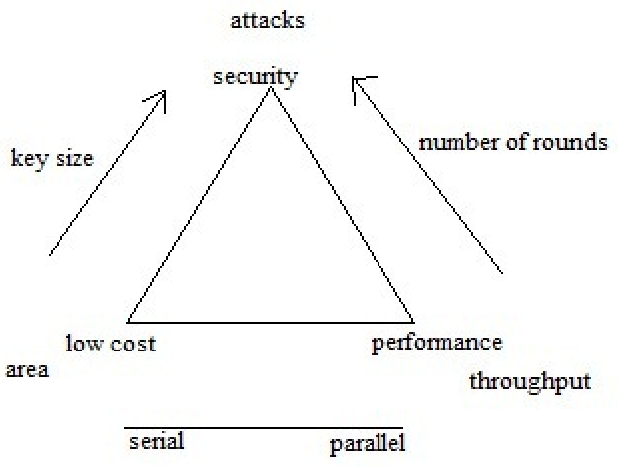

Fig. 1. The tradeoffs between costs, performance and security for cryptography[7]

Algorithms like PRINT, LED and KTANTAN are extremely low cost in terms of hardware (hardwired crossings). Algorithms such as PRESENT, EPCBC, DESL, KATAN, LED, LBLOCK and RECTANGLE use only the encryption functions on device while other algorithms like TWINE, Puffin and KLEIN use both encryption and decryption function. That is why comparative evaluations of these block ciphers which actually reach low cost goal on any platform is hard. As a result, to set tradeoff among cost, performance and security (tradeoffs) for these algorithms is tough.

Figure 1 shows the tradeoffs between cost, performance and security. For low cost the serialized structure and small key size is used. For high throughput, number of rounds is less and for high security number of rounds is more[7].

This paper is divided in four sections. Different symmetric key lightweight block cipher algorithms are discussed in section 2 and their comparative analysis are discussed in section 3. Section 4 concludes the paper.

2. LIGHTWEIGHT BLOCK CIPHERS

Different symmetric key lightweight block ciphers according to their key size, block size, round function, key scheduling and area (gate equivalents) of implementation are described as follows.

2.1 PRINT

PRINT [11] cipher is designed for integrated circuit (IC) printing. IC-printing allows the production and personalisation of circuits at very low cost. PRINT is SPN structure, the substitution layer consists of 16 3 bits S-boxes, the permutation layer is linear diffusion. PRINT cipher has 48 or 96 bits block size, 80 or 160 bits key size with 48 or 96 rounds respectively. Both PRINT ciphers are fixed key (hardwired) algorithms and there is no need to update key in each round. Key is divided in two part, first XORed with cipher text and anther used for permutation before the S-box.

Each round consists of following 5 steps:
1- Key XORed with cipher.
2- Cipher shuffled using linear diffusion.
3- Cipher (rightmost 6 bits) XORed with round-constant.
4- Bits are permuted using key-dependent permutation.
5- Cipher is mixed using S-box.

S-box used in PRINT cipher is:

$s(x)$={0, 1, 3, 6, 7, 4, 5, 2}

2.2 PRESENT

PRESENT [3] cipher is a hardware-optimized ultra-lightweight block cipher that has been designed with area and power constraints. PRESENT is an example of SPN structure. It has 64 bits block size, 80 or 128 bits key size with 31 rounds. Here S-box is 4 bits and used 16 times in one round.

Each round consists of the following 3 steps:
1- AddRoundKey: Key XORed with cipher.
2- Substitution: Used 4 bits S-box.
3- Permutation: Used P-layer.

S-box used in PRESENT cipher is:

$s(x)$={C, 5, 6, B, 9, 0, A, D, 3, E, F, 8, 4, 7, 1, 2}

At any round, the 64 bits round subkey consists of the leftmost 64 bits of the current key and update as follows: Key bits rotates 61 times left and 4 bits k_{79}, k_{78}, k_{77} and k_{76} pass through S-box. Finally, 5 bits round-counter XORed with the key bits $k_{19}, k_{18}, k_{17}, k_{16}$ and k_{15}.

2.3 EPCBC

EPCBC[18] is designed for Electronic Product Code Encryption. Electronic Product Code Encryption needs 96 bits for unique identification but in PRESENT cipher, key size is 64 bits which needs two encryptions while in AES [6], key size is 128 bits so wastage of some bits occur. It has two types, EPCBC (48, 96) 48 bits block size, 96 bits key size with 32 rounds and EPCBC (96, 96) 96 bits block size, 96 bits key size with 32 rounds . Both algorithms uses the structure of PRESENT cipher (PR−48 and PR−96) for encryption and also used the PRESENT cipher's S-box. EPCBC cipher differ from PRESENT cipher only in key scheduling.

Table 1. S-box used in DESL and DESXL cipher is:

14	5	7	2	11	8	1	15	0	10	9	4	6	13	12	3
5	0	8	15	14	3	2	12	11	7	6	9	13	4	1	10
4	9	2	14	8	7	13	0	10	12	15	1	5	11	3	6
9	6	15	5	3	8	4	11	7	1	12	2	0	14	10	13

In key scheduling for EPCBC (48, 96), the left half of 96 bits key forms the first subkey and applies variant-Feistel structure for 8 rounds. Each variant-Feistel structure is 4 rounds of PR−96 cipher structure without key addition. For EPCBC (96, 96), all 96 bits of key is the first subkey and applies 32 rounds of PR−96 cipher structure without key addition for 32 sub-keys.

2.4 DES, DESL, DESX and DESXL

DESL[12] is the lightweight block cipher of DES [2] which has 64 bits block size, 56 bits key size and involves 16 rounds. Instead of using 8 different S-box of DES, DESL repeatedly uses a single (6 * 4 bits) S-box 8 times and discard the initial and final permutation which reduces the cost and the area of block cipher. For higher security, A variant of DES called DESX is used. DESX has key size of 184 (k = 56, k_1 = 64, k_2 = 64) bits and is define as:

$$DESX_{k.k_1.k_2}(x) = k_2 \oplus DES_k(k_1 \oplus x)$$

DESXL is the lightweight block cipher of DESX which has 64 bits block size, 184 bits key size with 16 rounds.

2.5 TWINE

TWINE[15] fits in extremely-small hardware and provides a notable performance on embedded software. TWINE is Type-2 generalized Feistel [20] with a highly-diffusive block shuffle. TWINE has two types, TWINE−80 and TWINE−128. Both algorithms have 64 bits block size and 36 rounds but the key size in TWINE−80 is 80 bits and TWINE−128 is 128 bits. Each round of TWINE involves a nonlinear substitution layer using 4-bits S-boxes and a diffusion layer which permutes the blocks(4 bits). This round function is iterated 36 times for both key lengths.

S-box used in TWINE cipher is:

$s(x)$={C, 0, F, A, 2, B, 9, 5, 8, 3, D, 7, 1, E, 6, 4}

Encryption, 64 bits block size divided in sixteen 4 bits blocks and only odd blocks will modified as $S(X_{i-1} \oplus RK_i) \oplus X_i$ where (4 bits) RK_i is i^{th} block round key. Than all blocks will shuffled.

Key Scheduling, 80 bits key size divided in twenty 4 bits blocks. For first subkey, blocks 1, 3, 4, 6, 13, 14, 15 and 16 are selected and update as follows: block-1 $X_1 \oplus S(X_0)$, block-4 $X_4 \oplus S(X_{16})$, block 7 and 19 XORed with round-counter then all blocks cyclic shifted 4 blocks left.

2.6 Puffin

Puffin[5] is involutional substitution-permutation network (SPN) structure. Involutional operation requires same data path for both encryption and decryption. Puffin has simple key scheduling algorithm (on-the-fly sub key generation no need to store all sub keys). Puffin is novel compact block cipher targeted to be embedded on digital systems (digital hardware design). It has low hardware complexity and good throughput, and is suitable for Application specific integrated circuit (ASIC) and Field programmable gate array (FPGA) technology. Block size of Puffin is 64 bits, key size is 128 bits and requires 32 rounds for both encryption and decryption.

Each round of encryption / decryption consists of the following 3 steps:
1- Substitution S: Used 4 bits S-box.
2- Key addition K: Bit wise XOR.
3- Permutation P: Used P-layer.

$$Encryption = K_{k_0} P\{SK_{kr}P\}_{r=1}^{32}$$

$$Decryption = K_{Pk_{32}} P\{SK_{Pk_r}P\}_{r=31}^{0}$$

S-box used in Puffin cipher is:

$s(x)$={D, 7, 3, 2, 9, A, C, 1, F, 4, 5, E, 6, 0, B, 8}

This S-box is used in both the encryption and decryption at same time such that encryption S (D) = 0 and decryption S (0) = D.

Key Scheduling for Encryption is as follows: Permutation P in all rounds and then inversion of the $1^{st}, 2^{nd}, 3^{rd}$ and 5^{th} bits in all rounds except in $2^{nd}, 5^{th}, 6^{th}$ and 8^{th} rounds. Key Scheduling for Decryption is as follows: inverse Permutation P^{-1} in all rounds and then invert the $30^{th}, 62^{th}, 71^{th}$ and 120^{th} bits in all rounds except in $2^{nd}, 5^{th}, 6^{th}$ and 8^{th} rounds.

2.7 KLEIN

KLEIN [9] has advantage in the software performance on legacy sensor platforms, while its hardware implementation can be compact as well. KLEIN is SPN structure, the substitution layer (SubNibbles) consists of 16 4 bits S-boxes, the permutation layer consists of RotateNibbles and MixNibbles. KLEIN has 64 bits block size, 64 or 80 or 96 bits key size with 12 or 16 or 20 rounds. Key (n = 16 / 20 / 24), plaintext (n = 16) and ciphertext (n = 16) pictured as n * nibbles (4 bits).

Each round consists of the following 4 steps:
1- AddRoundKey: Bit wise XOR.
2- SubNibbles: Nibbles are passes through the S-box.
3- RotateNibbles: Nibbles rotated left two bytes.
4- MixNibbles: Same as MixColumns in AES.

S-box used in KLEIN cipher is:

$s(x)$={7, 4, A, 9, 1, F, B, 0, C, 3, 2, 6, 8, E, D, 5}

Key Scheduling takes place as follows: i^{th} sub key is divided in two part x, y (k_i=x.y). Then cycling left shift one bits in each part of key x, y ($k_i = x{<<}.y{<<}$) and swap (x, y) such as (x, y) = (y, x⊕y). Then XORed round-counter with 3^{rd} byte of x and substitute 2^{nd} and 3^{rd} byte of y with S-box.

2.8 KATAN and KTANTAN

KATAN[4] is very efficient hardware oriented block ciphers accept 80-bits keys and different block sizes such as 32, 48 and 64 bits. KATAN block ciphers are divided into two sets. First set is the KATAN, KATAN−32, KATAN−48 and KATAN−64. Second set is the KTANTAN, KTANTAN−32, KTANTAN−48 and KTANTAN−64 respectively. KTANTAN is more compact than KATAN, but is suitable only where the key of the device is fixed. The two algorithms differ only in key scheduling.

Plaintext in KATAN and KTANTAN is divided into two parts. In each round, several bits are taken and enter into two nonlinear Boolean functions and output of the Boolean functions is shifted to the left one bit. Total 254 rounds are executed.

Two nonlinear boolean functions used in KATAN and KTANTAN:

$$f_a(A) = A[x_1] \oplus A[x_2] \oplus (A[x_3].A[x_4]) \oplus (A[x_5].IR) \oplus k_a$$

$$f_b(B) = B[y_1] \oplus B[y_2] \oplus (B[y_3].B[y_4]) \oplus (B[y_5].B[y_6]) \oplus k_b$$

Where x_i and y_j are selected bits of A and B portion respectively. IR is irregular update of round function for more detail [4].

Key schedule of the KATAN, loads the key into LFSR.
let the key K, then the subkey of round i is $k_a||k_b = k_{2.i}||k_{2.i+1}$ where

$$k_i = \begin{cases} K_i & \text{for } i = 0...79 \\ k_{i-80} \oplus k_{i-61} \oplus k_{i-50} \oplus k_{i-13} & \text{otherwise} \end{cases}$$

Finally, the LFSR is clocked twice.

2.9 LED

LED (Light Encryption Device) [10] is maintain a reasonable performance profile for software implementation, 64 bits block cipher with four key sizes 64 bits, 80 bits, 96 bits and 128 bits. LED cipher uses PRESENT cipher's s-box.

Encryption, 64-bits plaintext (sixteen four-bits nibbles) are arranged in a square array. Round function is as follows(s = 8 for 64 bits key and s = 12 for other keys):

for 0 to s-1
1- AddRoundKey- Key XORed with cipher.
2- STEP

STEP describe as follows:
for 0 to 3
1- AddConstants- Round constants are combined with cipher using bitwise XOR.
2- SubCells- Each nibbles is replaced by generated nibble using PRESENT s-box.
3- ShiftRows- i^{th} row rotated i cell position left.
4- MixColumnsSerial- Column replaced by the column vector.

In key scheduling for 64 bits key, all subkeys are same while for 128 bits key, subkeys are alternatively equaled to left part and right part.

2.10 LBLOCK

LBLOCK [17] implemented efficiently in hardware environments and also in software platforms. It has 64 bits block size, 80 bits key size and 32 rounds. LBLOCK used variant-feistal structure. Design of LBLOCK includes S-box layer and P permutation layer. In LBLOCK 10 different 4 * 4 bits S-boxes are used, 8 S-box in

Table 2. S-boxes used in LBLOCK cipher is:

S_0	14	9	15	0	13	4	10	11	1	2	8	3	7	6	12	5
S_1	4	11	14	9	15	13	0	10	7	12	5	6	2	8	1	3
S_2	1	14	7	12	15	13	0	6	11	5	9	3	2	4	8	10
S_3	7	6	8	11	0	15	3	14	9	10	12	13	5	2	4	1
S_4	14	5	15	0	7	2	12	13	1	8	4	9	11	10	6	3
S_5	2	13	11	12	15	14	0	9	7	10	6	3	1	8	4	5
S_6	11	9	4	14	0	15	10	13	6	12	5	7	3	8	1	2
S_7	13	10	15	0	14	4	9	11	2	1	8	3	7	5	12	6
S_8	8	7	14	5	15	13	0	6	11	12	9	10	2	4	1	3
S_9	11	5	15	0	7	2	9	13	4	8	1	12	14	10	3	6

encryption and other 2 in key scheduling.

Let $M = X_1||X_0$ is 64 bits plaintext then the encryption algorithm is:

$X_i = F(X_{i-1}, K_{i-1}) \oplus (X_{i-2} <<< 8)$ for $i = 2...33$

Output $C = X_{32}||X_{33}$ is 64 bits ciphertext. Where the F (round function) is:

$F(X, K_i) = P(S(X \oplus K_i))$

Where S (confusion function) non-linear layer of F and consists of eight 4-bits s-boxes in parallel. And P (diffusion function) permutation of eight 4-bits words.

Let $C = X_{32}||X_{33}$ is 64 bits ciphertext then the decryption algorithm is:

$X_j = F(X_{j+1}, K_{j+1}) \oplus (X_{j+2} >>> 8)$ for $j = 31...0$

Output $M = X_1||X_0$ is 64 bits plaintext.

Key scheduling: Out of 80 bits key ($K = k_{79}k_{78}...k_1k_0$), the leftmost 32 bits are first subkey (K_1). Then key K is updated 31 times as follows: 29 times left shift then bits $[k_{79}k_{78}k_{77}k_{76}]$ and $[k_{75}k_{74}k_{73}k_{72}]$ passes through two different s-box then bits $[k_{50}k_{49}k_{48}k_{47}k_{46}]$ XORed with round-counter. Now the leftmost 32 bits are other subkey (K_2) and so on.

2.11 RECTANGLE

RECTANGLE [19] is a bit-slice ultra-lightweight block cipher suitable for multiple platforms which achieve very low area in hardware and also very competitive performance in software. RECTANGLE is SPN structure, the substitution layer (S-layer) consists of 16 4 bits S-boxes, the permutation layer (P-layer) is composed of 3 rotations. RECTANGLE has 64 bits block size, 80 or 128 bits key size with 25 rounds. Key (n = 20 / 32) and plaintext (n = 16) pictured as 4 * n rectangular array of bits.

Each round consists of the following 3 steps:
1- AddRoundkey: Bit wise XOR.
2- SubColumn: S-boxes used the 4 bits of the same column.
3- ShiftRow: Row 0 is not rotated, row 1 is left rotated 1 bit, row 2 is left rotated 12 bits, row 3 is left rotated 13 bits.

S-box used in RECTANGLE cipher is:

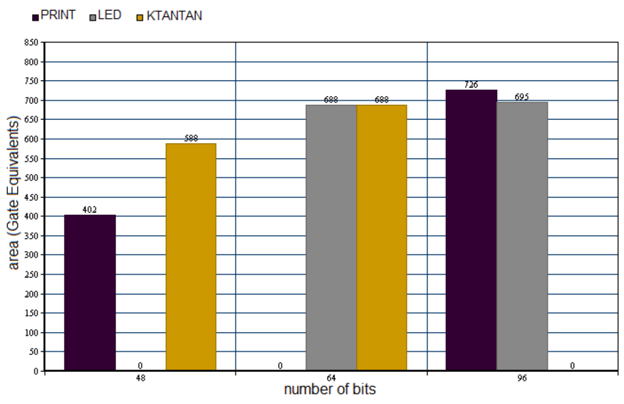

Fig. 2. Comparison between PRINT, LED and KTANTAN

$s(x)$={9, 4, F, A, E, 1, 0, 6, C, 7, 3, 8, 2, B, 5, D}

At any round, the 64 bits round subkey consists of the 16 rightmost columns of the current key and updated as follows: Applying the S-box to the 0^{th} column. Row 0 is left rotated 7 bits , row 1 is left rotated 9 bits, row 2 is left rotated 11 bits, row 3 is left rotated 13 bits and 5 bits round-constant (RC) is XORed with the rightmost 5 bits of row 0.

3. COMPARISONS

Different lightweight block ciphers shown in table 3 according to their key size, block size, function, architecture, structure, cycle per block, throughput and area (Gate Equivalents). Two types of architectures, serialized and round-based. Serialized architecture is used for low area (Gate Equivalents) in this data path of algorithm is equal to 4 bits and round-based architectures are used for high throughput in this data path of algorithm is equal to block size. Three types of structure, SPN (Substitution Permutation Network), Feistel Network and LFSR (linear-feedback shift register). SPN uses substitution by S-box and permutation by P-layer. Feistel network uses binary XOR and shifting of left-right portion of cipher. LFSR uses shift register whose input bit is a function of previous state.

PRINT−48, PRINT−96, LED−64, LED−80, LED−96, LED−128, KTANTAN−32, KTANTAN−48 and KTANTAN−64 (shown as * in table) are fixed key (hardwired) algorithms and used only for specific applications. For 80 bits key size and 48 bits block size PRINT has lesser area (Gate Equivalents) than KTANTAN but the throughput of the KTANTAN is more then PRINT. For 80 bits key size and 64 bits block size KTANTAN and LED has same area but the throughput of the KTANTAN is more then LED and shown in fig 2. For 160 bits key size PRINT is used, for 64 or 96 or 128 bits key size LED is used and for 32 bits block size KTANTAN is used.

PRESENT, EPCBC, DESL, KATAN, LED, LBLOCK and RECTANGLE are only the encryption algorithms on device (decryption on the server side) with various key sizes. For 80 bits key size LBLOCK has lesser area (Gate Equivalents) than LED, PRESENT, KATAN and RECTANGLE but the throughput of the KATAN is more then LBLOCK, LED, PRESENT and RECTANGLE. For

Table 3. Comparison of lightweight block ciphers

Ciphers	Function	Architecture	Structure	Key size	Block size	Rounds	Cycles / Block	Throughput	Area (GEs)
PRINT-48*	Enc.	Serialized	SPN	80	48	48	768	6.25	402
PRINT-48*	Enc.	Round-based	SPN	80	48	48	48	100	503
PRINT-96*	Enc.	Serialized	SPN	160	96	96	3072	3.13	726
PRINT-96*	Enc.	Round-based	SPN	160	96	96	96	100	967
LED-64*	Enc.	Serialized	SPN	64	64	32	1248	5.1	688
LED-80*	Enc.	Serialized	SPN	80	64	48	1872	3.4	690
LED-96*	Enc.	Serialized	SPN	96	64	48	1872	3.4	695
LED-128*	Enc.	Serialized	SPN	128	64	48	1872	3.4	700
KTANTAN-32*	Enc.	Serialized	LFSR	80	32	254	255	12.5	462
KTANTAN-48*	Enc.	Serialized	LFSR	80	48	254	255	18.8	588
KTANTAN-64*	Enc.	Serialized	LFSR	80	64	254	255	25.1	688
PRESENT-80	Enc.	Serialized	SPN	80	64	32	516	12.4	1030
PRESENT-80	Enc.	Round-based	SPN	80	64	32	32	200	1570
PRESENT-128	Enc.	Serialized	SPN	128	64	32	528	12.12	1339
PRESENT-128	Enc.	Round-based	SPN	128	64	32	32	200	1886
EPCBC-48	Enc.	Serialized	SPN	96	48	32	396	12.12	1008
EPCBC-96	Enc.	Serialized	SPN	96	96	32	792	12.12	1333
DES	Enc.	Serialized	Feistel	56	64	16	144	5.55	2309
DESL	Enc.	Serialized	Feistel	56	64	16	144	5.55	1848
DESX	Enc.	Serialized	Feistel	184	64	16	144	5.55	2629
DESXL	Enc.	Serialized	Feistel	184	64	16	144	5.55	2168
TWINE-80	Enc.	Round-based	Feistel	80	64	36	36	178	1503
TWINE-80	Enc.	Serialized	Feistel	80	64	36	540	11.8	1116
TWINE-80	Enc+Dec	Round-based	Feistel	80	64	36	36	178	1799
TWINE-128	Enc.	Round-based	Feistel	128	64	36	36	178	1866
TWINE-128	Enc+Dec	Round-based	Feistel	128	64	36	36	178	2285
Puffin	Enc+Dec	Round-based	SPN	128	64	32	32	194	2577
KLEIN-64	Enc+Dec	Round-based	SPN	64	64	12	13	492.3	2475
KLEIN-80	Enc+Dec	Round-based	SPN	80	64	16	17	376.5	2629
KLEIN-96	Enc+Dec	Round-based	SPN	96	64	20	21	304.8	2769
KLEIN-64	Enc+Dec	Serialized	SPN	64	64	12	207	30.9	1220
KLEIN-80	Enc+Dec	Serialized	SPN	80	64	16	271	23.6	1478
KLEIN-96	Enc+Dec	Serialized	SPN	96	64	20	335	19.1	1528
KATAN-32	Enc.	Serialized	LFSR	80	32	254	255	12.5	802
KATAN-48	Enc.	Serialized	LFSR	80	48	254	255	18.8	927
KATAN-64	Enc.	Serialized	LFSR	80	64	254	255	25.1	1054
LED-64	Enc.	Serialized	SPN	64	64	32	1248	5.1	966
LED-80	Enc.	Serialized	SPN	80	64	48	1872	3.4	1040
LED-96	Enc.	Serialized	SPN	96	64	48	1872	3.4	1116
LED-128	Enc.	Serialized	SPN	128	64	48	1872	3.4	1265
LBLOCK	Enc.	Round-based	Feistel	80	64	32	32	200	1320
LBLOCK	Enc.	Serialized	Feistel	80	64	32	576	11.1	866
RECTANGLE-80	Enc.	Round-based	SPN	80	64	25	26	246	1467
RECTANGLE-128	Enc.	Round-based	SPN	128	64	25	26	246	1787
RECTANGLE-80	Enc.	Serialized	SPN	80	64	25	461	13.9	1066

128 bits key size LED has lesser area than PRESENT but the throughput of the PRESENT is more then LED and shown in fig 3. For 96 bits key size EPCBC and LED are used, for 32 or 48 bits block size KATAN is used and for 56 or 184 bit key size DESL or DESXL is used.

TWINE, Puffin and KLEIN have both encryption and decryption function with various key size. For 80 bits key size KLEIN has lesser area (Gate Equivalents) than TWINE but the throughput of the TWINE is more then KLEIN. For 128 bits key size TWINE has lesser area than Puffin but the throughput of the Puffin is more then TWINE and shown in fig 4. For 64 and 96 bits key size KLEIN is used.

4. CONCLUSION

In this paper, a comparative analysis of different symmetric key lightweight block ciphers such as PRINT, PRESENT, EPCBC, DESL, TWINE, Puffin, KLEIN, KATAN, LED, LBLOCK and RECTANGLE is presented. It highlight the tradeoffs between cycle per block, throughput and area of different algorithms. Although the cost is low, the symmetric key lightweight block ciphers are needs to be improved in many directions like number of cycle per

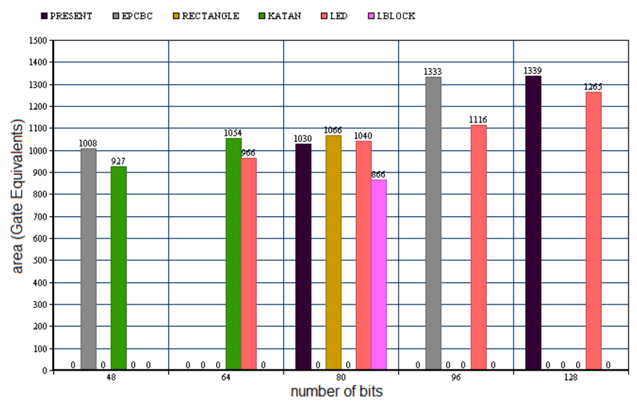

Fig. 3. Comparison between PRESENT, EPCBC, DES, KATAN, LED, LBLOCK and RECTANGLE

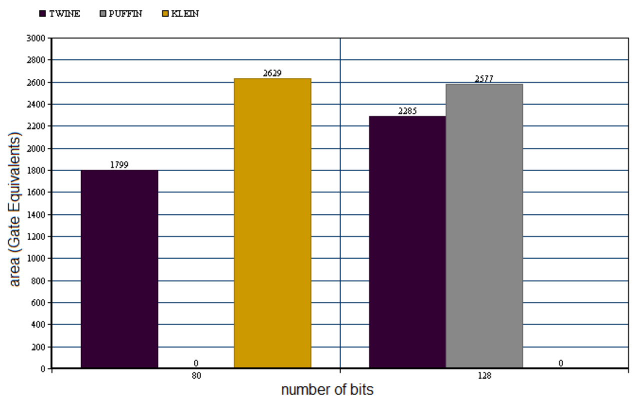

Fig. 4. Comparison between TWINE, PUFFIN and KLEIN

block, throughput and area (Gate Equivalents). Scopes for further research include the extension of this paper towards more algorithms like public-key lightweight block ciphers.

5. REFERENCES

[1] `http://en.wikipedia.org/wiki/KeeLoq`.

[2] `http://en.wikipedia.org/wiki/Data_Encryption_Standard`.

[3] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. Present: An ultra-lightweight block cipher. In *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.

[4] Christophe Cannire, Orr Dunkelman, and Miroslav Kneevi. Katan and ktantan a family of small and efficient hardware-oriented block ciphers. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer Berlin Heidelberg, 2009.

[5] Huiju Cheng, Howard M. Heys, and Cheng Wang. Puffin: A novel compact block cipher targeted to embedded digital systems. In *Proceedings of the 2008 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools*, DSD '08, pages 383–390. IEEE Computer Society, Washington, DC, USA, 2008.

[6] Joan Daemen and Vincent Rijmen. *The Design of Rijndael*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.

[7] T. Eisenbarth and S. Kumar. A survey of lightweight-cryptography implementations. *Design Test of Computers, IEEE*, 24(6):522–533, 2007.

[8] Thomas Eisenbarth, Zheng Gong, Tim Güneysu, Stefan Heyse, Sebastiaan Indesteege, Stéphanie Kerckhof, François Koeune, Tomislav Nad, Thomas Plos, Francesco Regazzoni, François-Xavier Standaert, and Loic van Oldeneel tot Oldenzeel. Compact implementation and performance evaluation of block ciphers in attiny devices. In *Proceedings of the 5th International Conference on Cryptology in Africa*, AFRICACRYPT'12, pages 172–187, Berlin, Heidelberg, 2012. Springer-Verlag.

[9] Zheng Gong, Svetla Nikova, and YeeWei Law. Klein: A new family of lightweight block ciphers. In Ari Juels and Christof Paar, editors, *RFID. Security and Privacy*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer Berlin Heidelberg, 2012.

[10] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matt Robshaw. The led block cipher. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 326–341. Springer Berlin Heidelberg, 2011.

[11] Lars R. Knudsen, Gregor Leander, Axel Poschmann, and Matthew J. B. Robshaw. Printcipher: A block cipher for ic-printing. In *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop*, volume 6225 of *Lecture Notes in Computer Science*, pages 16–32. Springer, 2010.

[12] Gregor Leander, Christof Paar, Axel Poschmann, and Kai Schramm. New lightweight des variants. In *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers*, volume 4593 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 2007.

[13] Fagen Li and Pan Xiong. Practical secure communication for integrating wireless sensor networks into the internet of things. *Sensors Journal, IEEE*, 13(10):3677–3684, Oct 2013.

[14] D. Maimut and K. Ouafi. Lightweight cryptography for rfid tags. *Security Privacy, IEEE*, 10(2):76–79, 2012.

[15] Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. TWINE: A lightweight block cipher for multiple platforms. In LarsR. Knudsen and Huapeng Wu, editors, *Selected Areas in Cryptography*, volume 7707 of *Lecture Notes in Computer Science*, pages 339–354. Springer Berlin Heidelberg, 2013.

[16] C.C. Tan, Haodong Wang, Sheng Zhong, and Qun Li. Ibe-lite: A lightweight identity-based cryptography for body sensor networks. *Information Technology in Biomedicine, IEEE Transactions on*, 13(6):926–932, 2009.

[17] Wenling Wu and Lei Zhang. Lblock: A lightweight block cipher. In Javier Lopez and Gene Tsudik, editors, *Applied Cryptography and Network Security*, volume 6715 of *Lecture*

Notes in Computer Science, pages 327–344. Springer Berlin Heidelberg, 2011.

[18] Huihui Yap, Khoongming Khoo, Axel Poschmann, and Matt Henricksen. Epcbc - a block cipher suitable for electronic product code encryption. In Dongdai Lin, Gene Tsudik, and Xiaoyun Wang, editors, *Cryptology and Network Security*, volume 7092 of *Lecture Notes in Computer Science*, pages 76–97. Springer Berlin Heidelberg, 2011.

[19] Wentao Zhang, Zhenzhen Bao, Dongdai Lin, Vincent Rijmen, Bohan Yang, and Ingrid Verbauwhede. Rectangle: A bit-slice ultra-lightweight block cipher suitable for multiple platforms. Cryptology ePrint Archive, Report 2014/084, 2014. `http://eprint.iacr.org/`.

[20] Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. On the construction of block ciphers provably secure and not relying on any unproved hypotheses. In Gilles Brassard, editor, *Advances in Cryptology CRYPTO 89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 461–480. Springer New York, 1990.