HIGHT: A New Block Cipher Suitable for Low-Resource Device *

Deukjo Hong[1], Jaechul Sung[2], Seokhie Hong[1], Jongin Lim[1], Sangjin Lee[1], Bon-Seok Koo[1], Changhoon Lee[1], Donghoon Chang[1], Jesang Lee[1], Kitae Jeong[1], Hyun Kim[4], Jongsung Kim[1], and Seongtaek Chee[3]

[1] Center for Information Security Technologies (CIST), Korea University, Seoul, Korea
{hongdj,hsh,jilim,sangjin,bskoo,crypto77, pointchang,jslee,kite,joshep}@cist.korea.ac.kr

[2] Department of Mathematics, University of Seoul, Seoul, Korea
jcsung@uos.ac.kr

[3] National Security Research Institute (NSRI), 161 Gajeong-dong, Yuseong-gu, Daejeon 305-350, Korea
chee@etri.re.kr

[4] Korea Information Security Agency (KISA), 78 Karak-dong, Songpa-gu, Seoul 138-160, Korea
hkim@kisa.or.kr

Abstract. In this paper, we propose a new block cipher HIGHT with 64-bit block length and 128-bit key length. It provides low-resource hardware implementation, which is proper to ubiquitous computing device such as a sensor in USN or a RFID tag. HIGHT does not only consist of simple operations to be ultra-light but also has enough security as a good encryption algorithm. Our hardware implementation of HIGHT requires 3048 gates on 0.25 μm technology.

Keywords: Block Cipher, Ubiquitous, Low-Resource Implementation

1 Introduction

Cryptographic applications providing various security services such as confidentiality, integrity, protection of privacy, and so on, are admitted as core technologies for advances in digital information society based on internet. Recently, ubiquitous computing system is in a matter of concern and interest, and designing cryptographic algorithms and applications suitable for such environment is an interesting research issue. For example, radio frequency identification (RFID) systems are useful for the automated electronic toll collection system, identifying and tracing pets, the administration of physical distribution, and so on, while the radio frequency communication between a reader and a tag causes

* This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment).

the problems about confidentiality and privacy. Such problems have been considered as obstacles to the advancement of RFID technology. However, since such ubiquitous computing technology has low-cost low-power light-weight platform, existing cryptographic algorithms can be hardly implemented under such resource constraint.

Recently, research on cryptographic protocols based on AES (Advanced Encryption Standard) [1] for resource-constraint environment is receiving a lot of attention. Further essentially, a few low-resource ASIC implementations of AES are presented [11, 12].

In this paper, we propose a new block cipher HIGHT (high security and light weight) with 64-bit block length and 128-bit key length, which is suitable for low-cost, low-power, and ultra-light implementation. HIGHT has a 32-round iterative structure which is a variant of generalized Feistel network. The prominent feature of HIGHT is that it consists of simple operations such as XOR, addition mod 2^8, and left bitwise rotation. So, it is hardware-oriented rather than software-oriented. We checked that HIGHT can be implemented with 3048 gates on 0.25 μm technology. Our circuit processes one round encryption per one clock cycle, thus its data throughput is about 150.6 Mbps at a 80 MHz clock rate. This performance is much faster than those of recently proposed low-resource hardware implementations of AES [11, 12].

The embedded CPU to sensor nodes in sensor networking system is 8-bit oriented. In case of 8-bit oriented software implementation, HIGHT is far faster than AES. The key schedule algorithm of HIGHT is designed to keep the original value of the master key after generating all whitening keys and all subkeys. Due to this property, the subkeys are generated on the fly in both encryption and decryption processes.

The paper is organized as follows. In Section 2, we present the specification and the design principle of HIGHT. Section 3 presents the design principles of HIGHT. In Section 4, we give the security analysis and statistical randomness tests of HIGHT against various existing attacks including differential and linear cryptanalysis. Section 5 treats the hardware implementation of HIGHT. In Section 6, we conclude this paper.

Table 1. Comparison the hardware implementation of HIGHT with AES's.

Algorithm	Technology (μm)	Area (GEs)	throughput (Mbps)	Max frequency (MHz)
AES [12]	0.35	3400	9.9	80
HIGHT	0.25	3048	150.6	80

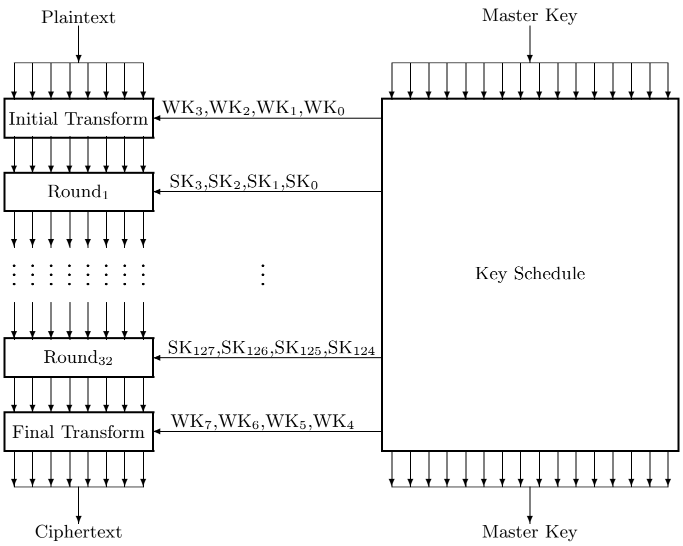

Fig. 1. Encryption process of HIGHT

2 Specification

2.1 Notations

We use the following notations for the description of HIGHT. The 64-bit plaintext and ciphertext are considered as concatenations of 8 bytes and denoted by $P = P_7||\cdots P_1||P_0$ and $C = C_7||\cdots C_1||C_0$, respectively. The 64-bit intermediate values are analogously represented, $X_i = X_{i,7}||\cdots X_{i,1}||X_{i,0}$ for $i = 0, \cdots, 32$. The 128-bit master key is considered as a concatenation of 16 bytes and denoted by $\mathrm{MK} = \mathrm{MK}_{15}||\cdots||\mathrm{MK}_0$. The followings are notations for mathematical operations:

$$\begin{aligned} \boxplus &: \text{addition mod } 2^8 \\ \boxminus &: \text{subtraction mod } 2^8 \\ \oplus &: \text{XOR (eXclusive OR)} \\ A^{\lll s} &: s\text{-bit left rotation of a 8-bit value } A \end{aligned}$$

We focus on the encryption process in the description of the specification of HIGHT because the decryption process is explained in the similar to the encryption process. The encryption process of HIGHT HightEncryption consists of key schedule, initial transformation, round function, and final transformation. Its description is as follows.

```
HightEncryption(P, MK) {
    KeySchedule(MK,WK,SK);
    HightEncryption(P, WK, SK) {
        InitialTransfomation(P, X_0, WK_3, WK_2, WK_1, WK_0);
        For i = 0 to 31 {
            RoundFunction(X_i, X_{i+1}, SK_{4i+3}, SK_{4i+2}, SK_{4i+1}, SK_{4i});
        }
        FinalTransfomation(X_32, C, WK_7, WK_6, , WK_5, WK_4);
    }
}
```

WK and SK mean whitening keys and subkeys, respectively.

2.2 Key Schedule

The key schedule KeySchedule for HightEncryption consists of two algorithms, WhiteningKeyGeneration which generates 8 whitening key bytes $\mathrm{WK}_0, \cdots, \mathrm{WK}_7$, and SubkeyGeneration which generates 128 subkey bytes $\mathrm{SK}_0, \cdots, \mathrm{SK}_{127}$.

```
KeySchedule(MK, WK, SK) {
    WhiteningKeyGeneration(MK, WK);
    SubkeyGeneration(MK, SK);
}
```

Whitening Key Generation HIGHT uses 8 whitening key bytes $\mathrm{WK}_0, \cdots, \mathrm{WK}_7$ for the initial and final transformations. The algorithm WhiteningKeyGeneration generates them as follows.

```
WhiteningKeyGeneration {
    For i = 0 to 7 {
        If 0 ≤ i ≤ 3, then WK_i ← MK_{i+12};
        Else, WK_i ← MK_{i−4};
    }
}
```

Subkey Generation 128 subkeys are used for 1 computation of HightEncryption, 4 subkeys per round. The algorithm SubkeyGeneration uses the subalgorithm ConstantGeneration to generate 128 7-bit constants $\delta_0, \cdots, \delta_{127}$, and then generates the subkeys $\mathrm{SK}_0, \cdots \mathrm{SK}_{127}$ with the constants.

δ_0 is fixed as 1011010_2. This is also the initial state $(s_6, \cdots, s_0)$ of 7-bit LFSR h. The connection polynomial of h is $x^7 + x^3 + 1 \in \mathbb{Z}_2[x]$. The algorithm ConstantGeneration uses the LFSR h to produce $\delta_1, \cdots, \delta_{127}$ from δ_0 as follows.

```
ConstantGeneration {
    s_0 ← 0; s_1 ← 1; s_2 ← 0;    s_3 ← 1;
    s_4 ← 1; s_5 ← 0; s_6 ← 1;
    δ_0 ← s_6||s_5||s_4||s_3||s_2||s_1||s_0;
    For i = 1 to 127 {
        s_{i+6} ← s_{i+2} ⊕ s_{i−1};
        δ_i ← s_6||s_5||s_4||s_3||s_2||s_1||s_0;
    }
}
```

Since x^7+x^3+1 is a primitive polynomial in $\mathbb{Z}_2[x]$, the period of h is $2^7-1=127$ and so $\delta_0=\delta_{127}$.

The algorithm SubkeyGeneration generates the subkeys as follows.

```
SubkeyGeneration(MK, SK) {
    Run ConstantGeneration
    For i = 0 to 7 {
        For j = 0 to 7 {
            SK_{16·i+j} ← MK_{j−i mod 8} ⊞ δ_{16·i+j};
        }
        For j = 0 to 7 {
            SK_{16·i+j+8} ← MK_{(j−i mod 8)+8} ⊞ δ_{16·i+j+8};
        }
    }
}
```

2.3 Initial Transformation

InitialTransformation transforms a plaintext P into the input of the first RoundFunction, $X_0=X_{0,7}||X_{0,6}||\cdots||X_{0,0}$ by using the four whitening-key bytes, WK_0, WK_1, WK_2, and WK_3.

```
InitialTransfomation(P, X_0, WK_3, WK_2, WK_1, WK_0) {
    X_{0,0} ← P_0 ⊞ WK_0; X_{0,1} ← P_1; X_{0,2} ← P_2 ⊕ WK_1; X_{0,3} ← P_3;
    X_{0,4} ← P_4 ⊞ WK_2; X_{0,5} ← P_5; X_{0,6} ← P_6 ⊕ WK_3; X_{0,7} ← P_7;
}
```

2.4 Round Function

RoundFunction uses two auxiliary functions F_0 and F_1:

$$F_0(x)=x^{\lll 1}\oplus x^{\lll 2}\oplus x^{\lll 7},$$
$$F_1(x)=x^{\lll 3}\oplus x^{\lll 4}\oplus x^{\lll 6}.$$

For $i=0,\cdots,31$, RoundFunction transforms $X_i=X_{i,7}||\cdots||X_{i,0}$ into $X_{i+1}=X_{i+1,7}||\cdots||X_{i+1,0}$ as follows.

$$
\begin{array}{|l|}
\hline
\mathsf{RoundFunction}(X_i, X_{i+1}, SK_{4i+3}, SK_{4i+2}, SK_{4i+1}, SK_{4i})\ \ \{ \\
\quad X_{i+1,1} \leftarrow X_{i,0};\ X_{i+1,3} \leftarrow X_{i,2};\ X_{i+1,5} \leftarrow X_{i,4};\ X_{i+1,7} \leftarrow X_{i,6}; \\
\quad X_{i+1,0} = X_{i,7} \oplus (F_0(X_{i,6}) \boxplus \mathrm{SK}_{4i+3}); \\
\quad X_{i+1,2} = X_{i,1} \boxplus (F_1(X_{i,0}) \oplus \mathrm{SK}_{4i+2}); \\
\quad X_{i+1,4} = X_{i,3} \oplus (F_0(X_{i,2}) \boxplus \mathrm{SK}_{4i+1}); \\
\quad X_{i+1,6} = X_{i,5} \boxplus (F_1(X_{i,4}) \oplus \mathrm{SK}_{4i}); \\
\} \\
\hline
\end{array}
$$

2.5 Final Transformation

FinalTransformation untwists the swap of the last round function and transforms $X_{32} = X_{32,7} || X_{32,6} || \cdots || X_{32,0}$ into the ciphertext C by using the four whitening-key bytes WK_4, WK_5, WK_6, and WK_7.

$$
\begin{array}{|l|}
\hline
\mathsf{FinalTransfomation}(X_{32}, C, \mathrm{WK}_7, \mathrm{WK}_6, \mathrm{WK}_5, \mathrm{WK}_4)\ \ \{ \\
\quad C_0 \leftarrow X_{32,1} \boxplus \mathrm{WK}_4;\ C_1 \leftarrow X_{32,2};\ C_2 \leftarrow X_{32,3} \oplus \mathrm{WK}_5;\ C_3 \leftarrow X_{32,4}; \\
\quad C_4 \leftarrow X_{32,5} \boxplus \mathrm{WK}_6;\ C_5 \leftarrow X_{32,6};\ C_6 \leftarrow X_{32,7} \oplus \mathrm{WK}_7;\ C_7 \leftarrow X_{32,0}; \\
\} \\
\hline
\end{array}
$$

2.6 Decryption Process

The decryption process HightDecryption is done in the canonical way to invert HightEncryption. Key schedule generates the subkeys in the reverse order. The round function in the decryption process has $\boxminus$ instead of $\boxplus$ and byte-swap with the opposite direction to that in the encryption process.

3 Design Principles

In this section we list brief description of design principles of HIGHT.

- The structure of HIGHT is generalized Feistel-like. This kind of structure reduces restriction of designing inner auxiliary functions. Compared to SP-like structure, the round function is light. Since encryption process is simply converted into decryption process, implementation of the circuit supporting both encryption and decryption processes does not require much more cost than the encryption-only circuit.
- Every operation in HIGHT is 8-bit-processor-oriented. CPUs embedded into the sensors in USN (Ubiquitous Sensor Network) are based on 8-bit processor. So, HIGHT has efficient performance in such environment. We checked that in 8-bit-oriented software implementation HIGHT is faster than AES-128.
- We intend to combine XOR and addition mod 2^8 alternatively. The combination of these quite different operations spread out the whole round of the algorithm. It plays an important role for resistance against existing attacks.

- The inner functions F_0 and F_1 of the round function provide bitwise diffusion. These functions can be viewed as linear transformations from $GF(2)^8$ to $GF(2)^8$. We selected two among linear transformations which have best diffusion.
- The 128-bit register used in the key schedule algorithm contains the master key value both before and after running the algorithm. So, only one 128-bit register is required for both encryption and decryption processes.
- The whitening keys are used in the first and the last rounds of HIGHT. If the whitening keys are not used, then the inputs to F_0 and F_1 in the first and the last rounds are directly revealed from plaintexts and ciphertexts.
- The sequence $\delta_0, \cdots, \delta_{127}$ generated by the linear feedback shift register h enhances randomness of subkey bytes. It also provides the resistance against slide attack.

4 Security Analysis

We analyze the security of HIGHT against various attacks. As a result, we claim that HIGHT is secure enough for cryptographic applications. In this subsection, we present not only brief description of our analysis but also the result of the statistical tests on HIGHT.

4.1 Differential Cryptanalysis

The resistance of a block cipher against differential cryptanalysis [6] depends on the maximum probability of differential characteristics, which are paths from the plaintext difference to the ciphertext difference. First of all, we have implemented a simulation for finding the maximum differential characteristics of a small version of HIGHT, Mini-HIGHT, which consists of four 8-bit input registers when 2^{32} of all possible input values are given. As a result, we found two 8-round maximum differential characteristics $\alpha \to \beta$ with a probability of 2^{-28} in which there always exist a difference pattern such that hamming weight is one at a particular round, where $(\alpha, \beta) \in \{(d0\,00\,ed\,86_x, 00\,84\,82\,01_x), (04\,dc\,20\,e2_x, 00\,84\,82\,01_x)\}$.

Since it is impossible for us to find all of the corresponding differential characteristics of HIGHT for given 2^{64} possible input values, we considered the above difference pattern of Mini-HIGHT with a noticeable feature and then found several 11-round differential characteristics $\alpha \to \beta$ with probability 2^{-58} where $(\alpha, \beta) \in \{(11\,89\,25\,e2\,c8\,01\,00\,00_x, 45\,02\,01\,00\,00\,91\,29\,95_x), (c8\,01\,00\,00\,11\,89\,25\,e2_x, 00\,91\,29\,95\,45\,02\,01\,00_x)\}$. Each of them are constructed by setting a difference of a particular intermediate variable to the starting point, and by prepending and appending good one-round differential characteristics to it. We expect that they have the best probability over all the 11-round differential characteristics and that for $r > 11$, no r-round differential characteristic is useful for differential cryptanalysis of HIGHT because we checked that there is no any efficient iterative differential characteristic. Differential attack on 13-round HIGHT without the final transformation recovers the subkeys of the 12th and 13th rounds with 2^{62} plaintexts.

4.2 Linear Cryptanalysis

Linear cryptanalysis [17, 18] uses linear relations of the plaintext, ciphertext, and key which hold with a probability. We call them, linear approximations. Let $p = 1/2 + \varepsilon$ be the probability of a linear approximation. ε is called, bias. If ϵ^2 is relatively high, the linear approximation is very useful for linear cryptanalysis. We found several 10-round linear approximations with $\varepsilon^2 = 2^{-54}$. Similarly to differential cryptanalysis of HIGHT, they were constructed by putting a 1-bit position of an intermediate variable to the starting point, and by prepending and appending good one-round linear approximations to it. We expect that they have the best bias over all the 10-round approximations and that for $r > 10$, no r-round linear approximation has good bias because we checked that there is no any iterative linear approximation in HIGHT. Linear attack on 13-round HIGHT without the final transformation recovers 36 bits of the subkeys of the 1st, 12th, and 13th rounds. It requires 2^{57} plaintexts with the success rate 96.7%.

4.3 Truncated Differential Cryptanalysis

Truncated differential characteristic [15] is a path from a partial difference of the input to a partial difference of the output. In order to find good truncated differential characteristics, we computed the probabilities of all differential characteristics with the following form:

$$\mathtt{00}\ \alpha_1\ \mathtt{00}\ \alpha_2\ \mathtt{00}\ \alpha_3\ \mathtt{00}\ \alpha_4 \rightarrow \mathtt{00}\ \beta_1\ \mathtt{00}\ \beta_2\ \mathtt{00}\ \beta_3\ \mathtt{00}\ \beta_4 \tag{1}$$

where all α_i, β_j are 1-byte values. The truncated differential characteristics with such form can be iterated, but their probabilities are terribly low. Even the sum of them is too low to be applied to the attack.

As the second approach, we considered several 10-round truncated differential characteristics with probability 1. For example, one among them has the following form: the input difference is $\mathtt{80\ e9\ 00\ 00\ 00\ 00\ 00\ 00}_x$ and the output difference is $\gamma\ \delta_1\ \delta_2\ \delta_3\ \delta_4\ \delta_5\ \delta_6\ \delta_7$ where γ is a nonzero 1-byte value and δ_i's are arbitrary 1-byte values. This truncated differential characteristic provides us with only one information about the output difference that the left-most byte of the output difference is nonzero. Since the probability of the characteristic is 1, we have information enough for the attack on HIGHT. We can use the truncated differential characteristic to recover 96 bits of the subkeys used from the 11th round to the 16th round in 16-round HIGHT. The attack requires $2^{14.1}$ plaintexts and $2^{108.69}$ encryptions of 16-round HIGHT.

4.4 Impossible Differential Cryptanalysis

We can construct a differential characteristic, which never occurs, by composing two short truncated differential characteristics with the probability 1 which do not meet in the middle. We call it an impossible differential characteristic [2]. Such differential characteristic can be used for attacks on block ciphers. Roughly speaking, since a key candidate satisfies an impossible differential characteristic is a wrong key, we can reduce the number of the key candidates by repeating such tests. We investigated all of the possible characteristics for all of the possible input differences and then found a 14-round impossible differential characteristic $\alpha{\rightarrow}\beta{\neq}\gamma{\leftarrow}\delta$ where $\alpha = (80\ e9\ 00\ 00\ 00\ 00\ 00\ 00)_x$, $\beta = (\triangle,\ ?,\ ?, ...,\ ?)_x$ ($\triangle$: a nonzero), $\gamma = (00,\ 00,\ ?,\ ?,\ ?,\ ?,\ ?,\ ?)_x$, and $\delta = (00\ ?\ ?\ ?\ 00\ 00\ 00\ 00)_x$. We can use this 14-round impossible differential characteristic to attack 18-round HIGHT. This attack requires $2^{46.8}$ chosen-plaintexts and $2^{109.2}$ encryptions of 18-round HIGHT.

4.5 Saturation Attack

The saturation attack [10, 16] uses a saturated multiset of plaintexts. The attacker needs the property that XOR sum of particular parts of the corresponding ciphertexts is zero. We call it a saturation characteristic. Saturation characteristics useful for the attack are often found in block ciphers in which small portions of the bits are interleaved by a strong nonlinear function while the main interleaving stage is linear. There exist 12-round saturation characteristics with the probability 1 in HIGHT, e.g., $\alpha = (S, C, C, C, C, C, C, C) {\rightarrow} \beta = (?, ?, ?, ?, B_0, ?, ?, ?)$ where S: a saturation set, C: a fixed constant, and B_0: a balanced set for the least significant bit. We can apply them to the attack on 16-round HIGHT. It requires 2^{42} plaintexts and 2^{51} encryptions of 16-round HIGHT.

4.6 Boomerang Attack

The main idea behind the boomerang attack [20] is to use two short differential characteristics with relatively high probabilities instead of one long differential with low probability. The boomerang attack has been improved to the amplified boomerang [14] and the rectangle [4, 5] attacks. This kind of attacks treat the block cipher E as $E = E_1 \circ E_0$ a cascade of E_0 and E_1. We assume that for E_0 there exists a differential characteristic $\alpha \rightarrow \beta$ with probability p and that for E_1 there exists a differential characteristic $\gamma \rightarrow \delta$ with probability q. Then the boomerang characteristic which is constructed from two differential characteristics $\alpha \rightarrow \beta$ and $\gamma \rightarrow \delta$ has probability p^2q^2. We applied the amplified boomerang attack to 13-round HIGHT without final transformation. We build a 11-round boomerang characteristic of HIGHT with probability 2^{-58} from two differential characteristics — one with probability 2^{-12} decipted in Table 2 and the other one with probability 2^{-17} decipted in Table 3. We use the 11-round boomerang characteristic to recover the subkeys of the 13th round with 2^{62} plaintexts.

4.7 Interpolation and Higher Order Differential Attack

Interpolation [13] and higher order differential [15] attacks are aimed against block ciphers which have low algebraic degree. Since the degree of a round function of HIGHT is 8, the full-round HIGHT has a high degree as a vector Boolean

Table 2. The 5 rounds differential characteristics (the 1st round $\sim$ the 5th round) with probability 2^{-12}.

$\alpha \longrightarrow \beta$
82 01 00 00 00 00 00 00_x $\longrightarrow$ 00 90 95 ca 01 00 00 00_x
00 00 00 00 82 01 00 00_x $\longrightarrow$ 01 00 00 00 00 90 95 ca_x

Table 3. The 6 rounds differential characteristics (the 6th round $\sim$ the 11th round) with probability 2^{-17}

$\gamma \longrightarrow \delta$
42 82 01 00 00 00 00 00_x $\longrightarrow$ 00 90 95 ca 01 00 00 00_x
00 00 00 00 42 82 01 00_x $\longrightarrow$ 01 00 00 00 00 90 95 ca_x

function. Furthermore, we believe that the result of higher order differential attack on HIGHT is less than the result of saturation attack on HIGHT because saturation attack can be viewed as a special and more effective case of higher order differential attack.

4.8 Algebraic Attack

In order to apply the algebraic attack [9] to block ciphers, we should derive an over-defined system of algebraic equations. Since a round function of HIGHT is the degree 8 as a vector Boolean function, it may be impossible to convert any equation system in HIGHT into an over-defined system.

4.9 Slide and Related-Key Attacks

Slide [7, 8] and related-key [3] attacks use some weakness of key schedule. The subkey generation algorithm of HIGHT has a simplicity and a linearity but resistance enough to frustrate those attacks due to the use of the round function with strong non-linearity and avalanche effect. It is known that the iterated ciphers with identical round functions, that is, equal structures and equal subkeys in the round functions, are vulnerable to slide attacks. However, since HIGHT uses the different constant for each round, it is secure against slide attack.

We are also convinced that the key schedule and round function of HIGHT makes related-key attacks difficult although the relation between two master keys is known and the corresponding relations between the subkeys can be predetermined due to linearity of the key schedule. To find long related-key differential characteristics with high probability and mount a successful distinguishing attack, we must keep the number of additions small. This can be done by trying to cancel out differences in XORs and additions but this work is not easy. So, by trial and error, we constructed 18-round related-key boomerang distinguisher which is composed of two short related-key differential characteristics with relatively high probability; one is the first 8 rounds, $(2c\ 00\ 80\ 00\ 00\ 00\ 00\ 00)_x \rightarrow$

$(00\ 00\ 00\ 00\ 43\ 80\ 00\ 00)_x$ under the related-key difference $(00\ 00\ 80\ 2c\ 00, ..., 00)$ with probability 2^{-6} and the other one is 10 rounds, $(08\ 9e\ 6f\ 80\ 2c\ 00\ 80\ 00)_x$ $\rightarrow (2c\ 00\ 80\ 00\ 00\ 00\ 00\ 00)_x$ under the related-key difference $(80\ 2c\ 00\ 00, ..., 00)$ with probability 2^{-23}. This is useful to attack on 19 rounds HIGHT but can be used to attack on full-round HIGHT.

4.10 Weak Keys

Originally, a weak key is defined as a key under which the encryption function is involution [19]. We checked that there does not exists any equivalent or weak key in HIGHT. In a broad sense, a weak key can be defined as a key under which the resistance of the block cipher against any attacks falls off. We suppose that it is very difficult to find such kind of weak keys in HIGHT.

Table 4. Results of HIGHT

Statistical Test	Proportion	
	High Density	Low Density
Frequency	0.994(Pass)	0.986(Pass)
Block Frequency ($m = 100$)	0.993(Pass)	0.991(Pass)
Runs	0.990(Pass)	0.982(Pass)
Long Runs of Ones	0.990(Pass)	0.994(Pass)
Rank	0.988(Pass)	0.992(Pass)
Spectral DFT	1.00(Pass)	0.990(Pass)
Non-overlapping Templates ($m = 9$)	0.990(Pass)	0.990(Pass)
Overlapping Templates ($m = 9$)	0.978(Pass)	0.984(Pass)
Universal	0.992(Pass)	0.980(Pass)
Lempel-Ziv Complexity	0.986(Pass)	0.980(Pass)
Linear Complexity ($M = 500$)	0.984(Pass)	0.994(Pass)
Serial ($m = 5$)	0.992(Pass)	0.985(Pass)
Approximate Entropy ($m = 5$)	0.986(Pass)	0.990(Pass)
Cusum	0.992(Pass)	0.988(Pass)
Random Excursions	0.986(Pass)	0.990(Pass)
Random Excursions Variant	0.989(Pass)	0.987(Pass)

4.11 Random Test

We show the results of the NIST statistical test on HIGHT. We use 500 samples of about 10^6 bit sequences for each test. Consequently, 500 (sample) $\times$ 10^6 (sequence) bits are used for each test. The Table 4 shows results of HIGHT. Here input parameters used in these tests has been included in parenthesis beside the name of the statistical test. From the Table 4, it is clear that the statistical test results for HIGHT don't indicate a deviation from random behaviour.

5 Hardware Implementation

We designed a simple circuit of HIGHT in order to check the hardware complexity on $0.25\mu m$ CMOS technology. The circuit consists of three parts: RoundFunction, KeySchedule, and Control Logic. RoundFunction processes whitening-key addition or round function with 64-bit input data and 4-byte round key, and KeySchdule generates 4-byte round key (four byte whiteningkeys or subkeys). Control Logic controls RoundFunction and KeySchedule to process HIGHT algorithm. The total size corresponds to 3048 NAND gates as you see in Table 5. Our circuit processes one round encryption per one clock cycle, thus its data throughput is about 150.6 Mbps at a 80 MHz clock rate. Note that our circuit is not area-optimized, and in order to reduce the gate count, we can simply modify it to process 1/2 or 1/4 of one round operation per a clock cycle. In the case of 1/4 round design, we estimate the minimized circuit would require much less than 3000 gates on $0.25\mu m$ technology and its data throughput would be about 37.6 Mbps at a 80 MHz clock rate. Meanwhile the last hardware implementation result of AES-128 [12] requires about 3400 gates and its data throughput is about 9.9 Mbps under the same clock rate.

Table 5. Gate count for hardware implementation of HIGHT

Component	Gate Count
RoundFunction	838
KeySchedule	1648
Control Logic	562
Total	3048

6 Conclusion

We proposed a block cipher HIGHT with 64-bit block length and 128-bit key length. HIGHT was designed to be proper to the implementation in the low-resource environment such as RFID tag or tiny ubiquitous devices. From security analysis, we are sure that HIGHT has enough security. Our implementation circuit processes one HIGHT encryption with 34 clock and requires 3048 gates. The data throughput of the circuit is about 150.6 Mbps under the operating frequency 80 MHz.

References

1. National Institute of Standards and Technology (NIST), FIPS-197: Advanced Encryption Standard, November 2001. http://www.itl.nist.gov/fipspubs/
2. E. Biham, A. Biryukov and A. Shamir, "Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials," *Advances in Cryptology - EUROCRYPT'99*, J. Stern, Ed., LNCS 1592, Springer-Verlag, pp. 12-23, 1999.
3. E. Biham, "New Types of Cryptanalytic Attack Using Related Keys," *Journal of Cryptology*, Volume 7, Number 4, pp. 156–171, 1994.
4. E. Biham, O. Dunkelman, N. Keller, "The Rectangle Attack – Rectangling the Serpent," *Advances in Cryptology – EUROCRYPT 2001*, LNCS 2045, Springer-Verlag, pp. 340–357, 2001.
5. E. Biham, O. Dunkelman, N. Keller, "New Results on Boomerang and Rectangle Attacks," *FSE 2002*, LNCS 2365, Springer-Verlag, pp. 1–16, 2002.
6. E. Biham, A. Shamir, "Differential Cryptanalysis of the Data Encryption Standard," Springer-Verlag, 1993.
7. A. Biryukov, D. Wagner, "Slide Attacks," *Advances in Cryptology – FSE'99*, LNCS 1687, Springer-Verlag, pp. 244-257, 1999.
8. A. Biryukov, D. Wagner, "Advanced Slide Attacks," *Advances in Cryptology – EUROCRYPT 2000*, LNCS 1807, Springer-Verlag, pp. 589–606, 2000.
9. N. Courtois, J. Pieprzyk, "Cryptanalysis of Block Ciphers with Overdefined Systems of Equations," *Advances in Cryptology – ASIACRYPT 2002*, LNCS 2501, Springer-Verlag, pp. 267–287, 2002.
10. J. Daemen, L. Knudsen and V. Rijmen, "The Block Cipher SQUARE," *FSE'97*, LNCS 1267, Springer-Verlag, pp. 137–151, 1997.
11. M. Feldhofer, S. Dominikus, and J. Wolkerstorfer, "Strong Authentication for RFID Systems Using the AES Algorithm," *CHES'04*, LNCS 3156, pp. 357–370, Springer-Verlag, 2004.
12. M. Feldhofer, J. Wolkerstorfer, and V. Rijmen, "AES Implementation on a Grain of Sand," *IEE Proceedings on Information Security*, Volume 152, Issue 1, pp. 13–20, 2005.
13. T. Jakoben and L. R. Knudsen, "The Interpolation Attack against Block Ciphers," *FSE'97*, LNCS 1267, Springer-Verlag, pp. 28–40, 1997.
14. J. Kelsey, T. Kohno, B. Schneier, "Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent," *FSE 2000*, LNCS 1978, Springer-Verlag, pp. 75–93, 2001.
15. L. R. Knudsen, "Truncated and Higher Order Differential," *FSE 94*, LNCS 1008, Springer-Verlag, pp. 229–236, 1995.
16. S. Lucks, "The Saturation Attack – a Bait for Twofish," *FSE 2001*, LNCS 1039, Springer-Verlag, pp. 189-203, 2001.
17. M. Matsui, "Linear Cryptanalysis Method for DES Cipher," *Advances in Cryptology – EUROCRYPT'93*, T. Helleseth, Ed., LNCS 765, Springer-Verlag, pp. 386–397, 1994.
18. M. Matsui, "The First Experimental Cryptanalysis of DES," *Advances in Cryptology – CRYPTO'94*, LNCS 839, Springer-Verlag, pp. 1–11, 1994.
19. A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.
20. D. Wagner, "The Boomerang Attack," *FSE'99*, LNCS 1636, Springer-Verlag, pp. 156–170, 1999.

A Figure of Functions in HIGHT

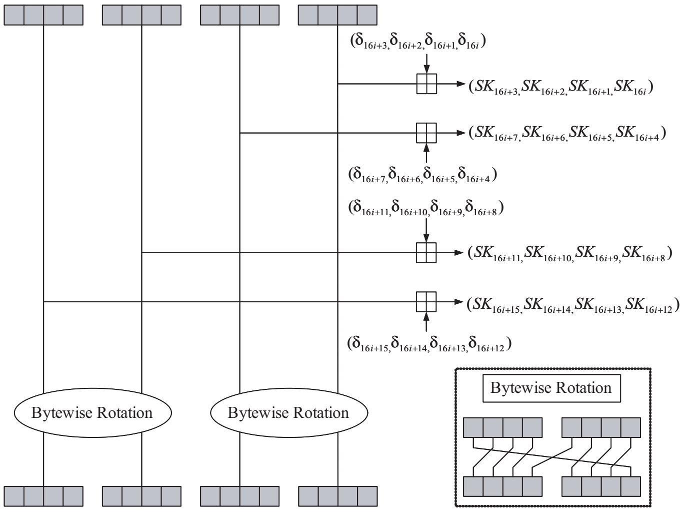

Fig. 2. Subkey generation of HIGHT key schedule

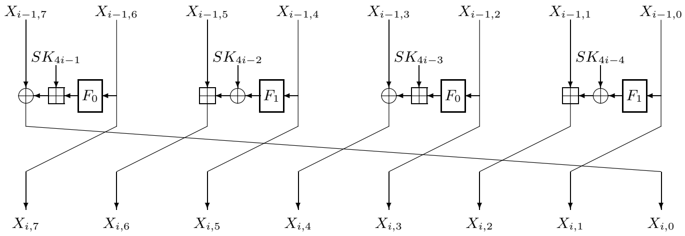

Fig. 3. The i-th RoundFunction of HIGHT for $i = 1, \cdots, 32$