GIFT: A Small Present

Towards Reaching the Limit of Lightweight Encryption

Subhadeep Banik[1,2] Sumit Kumar Pandey[1]
Thomas Peyrin[1] Yu Sasaki[3]
Siang Meng Sim[1] Yosuke Todo[3]

1. Nanyang Technological University, Singapore
2. École Polytechnique Fédérale de Lausanne, Switzerland
3. NTT Secure Platform Laboratories, Japan

CHES2017

Table of Contents

Table of Contents

10 Years Ago...

A decade ago, a lightweight block cipher, PRESENT, was presented at CHES2007.

31-round SPN block cipher with 64-bit block size.
Very simple design of Sbox layer and bit permutation (cost 0GE in hardware).

In 2012, selected as ISO standards, ISO/IEC 29192.

Block Cipher PRESENT

Its resistance against differential cryptanalysis (DC) comes from its Sbox which has differential branching number 3.

Differential branching number x (BNx): Total Hamming weight of any nonzero input and output differences is at least x.

Figure: Hamming wt2 Example.

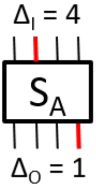
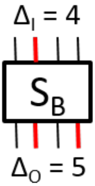


Figure: Hamming wt3 Example.



Block Cipher PRESENT

However, BN3 Sboxes are costly in general.
PRESENT Sbox (BN3) costs 21.33GE, while
SKINNY Sbox (BN2) costs 13.33GE.
This difference is multiplied in round based implementation.

Also, it is weaker against linear cryptanalysis (LC).

Now...

In CHES2017, we present a new lightweight block cipher, improving over PRESENT, we called it — GIFT.

By carefully crafting the bit permutation in conjunction with the Sbox properties, we can remove the constraint of BN3.

Advantages of GIFT compared to PRESENT:

- smaller area thanks to smaller Sbox and also lesser subkey additions,
- better resistance against LC thanks to good choice of Sbox and bit permutation,
- lesser rounds and higher throughput,
- simpler and faster key schedule.

Table of Contents

Block Cipher GIFT

There are 2 versions of GIFT:

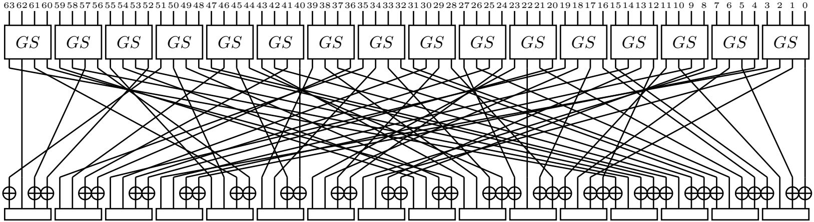
- GIFT-64, 28-round with 64-bit block size,
- GIFT-128, 40-round with 128-bit block size.

Both versions have 128-bit key size.

Round Function

Each round of `GIFT` consists of 3 steps:

SubCells, PermBits and AddRoundKey.



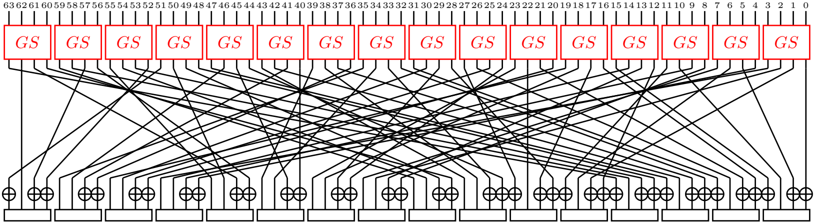
Denote rightmost bit as LSB b_0 and $\{b_{4i+j}\}$ as bit j.
E.g. $b_1, b_5, b_9, \ldots$ are bit 1.

SubCells

Apply 16 4-bit Sboxes, *GS*, in parallel to every nibble of the state.

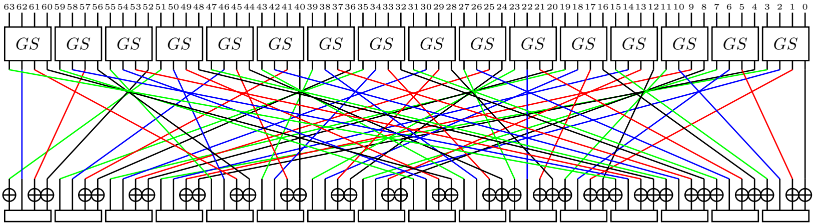
Table: GIFT Sbox *GS*

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$GS(x)$	1	a	4	c	6	f	3	9	2	d	b	7	5	0	8	e



PermBits

Pure bit permutation without any XOR gate.

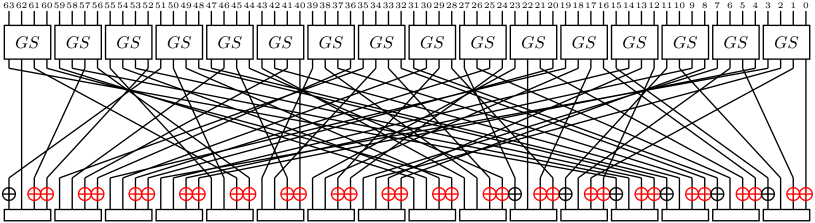


Map bit j to bit j.

AddRoundKey

Add 32-bit round key RK to the state,
$RK = U \| V = u_{15} \dots u_0 \| v_{15} \dots v_0$.

U and V are XORed to bit 1 and bit 0 respectively.



AddRoundKey

Add a single bit '1' is to the most significant bit, and a 6-bit round constant $C = c_5\ c_4\ c_3\ c_2\ c_1\ c_0$ is XORed to bit 3 of the first 6 nibbles.

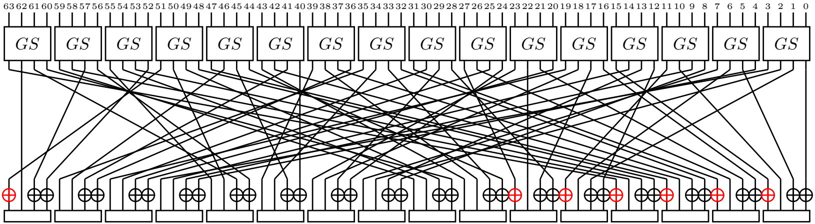


Table of Contents

Round Key

The 128-bit key is split into 8 16-bit words.

$$K = k_7 \| k_6 \| \ldots \| k_1 \| k_0, \text{ where } k_i \text{ is 16-bit words.}$$

k_1 and k_0 are extracted as the round key $RK = U \| V$.

						U	V
k_7	k_6	k_5	k_4	k_3	k_2	k_1	k_0

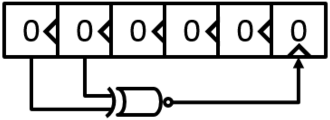
Key state is updated after key extraction.

$k_1 \ggg 2$	$k_0 \ggg 12$	k_7	k_6	k_5	k_4	k_3	k_2

where $\ggg i$ is an i bits right rotation within a 16-bit word.

Round Constants

Round constants are generated using a 6-bit affine LFSR with 1 XNOR gate (same as SKINNY's).



Initialised to zero, and updated before using as round constants.

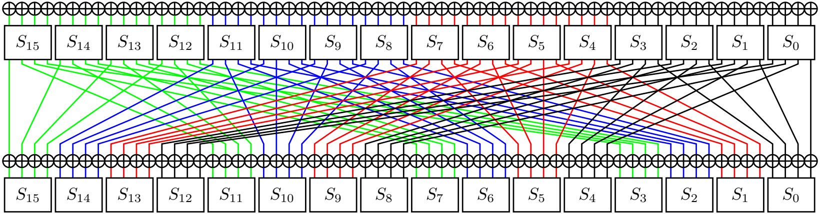
Rounds	**Constants**
1 - 16	01,03,07,0F,1F,3E,3D,3B,37,2F,1E,3C,39,33,27,0E
17 - 32	1D,3A,35,2B,16,2C,18,30,21,02,05,0B,17,2E,1C,38
33 - 48	31,23,06,0D,1B,36,2D,1A,34,29,12,24,08,11,22,04

Table of Contents

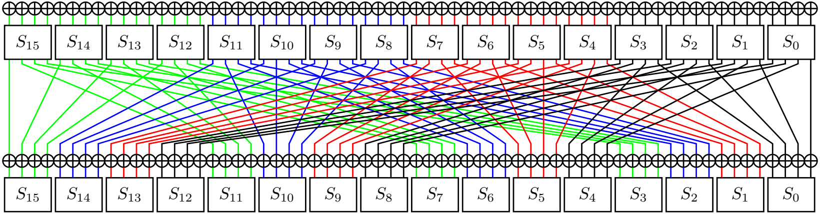
PRESENT Bit Permutation

To understand why BN2 Sboxes do not work for PRESENT, we have to look into the PRESENT bit permutation.

PRESENT bit permutation can be partitioned into 4 independent 16-bit permutations.



Group Mapping



A group mapping sends the 16 output bits of the Quotient group to the input of the Remainder group.

$Q0 = \{S_0, S_1, S_2, S_3\} \rightarrow R0 = \{S_0, S_4, S_8, S_{12}\}$.
$Q1 = \{S_4, S_5, S_6, S_7\} \rightarrow R1 = \{S_1, S_5, S_9, S_{13}\}$.
$Q2 = \{S_8, S_9, S_{10}, S_{11}\} \rightarrow R2 = \{S_2, S_6, S_{10}, S_{14}\}$.
$Q3 = \{S_{12}, S_{13}, S_{14}, S_{15}\} \rightarrow R3 = \{S_3, S_7, S_{11}, S_{15}\}$.

The group mappings are identical.

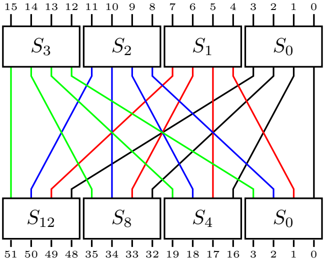
PRESENT Group Mapping

$$Q0 = \{S_0, S_1, S_2, S_3\} \rightarrow R0 = \{S_0, S_4, S_8, S_{12}\}.$$

Table: PRESENT group mapping.

$Q0$ \ $R0$	S_0	S_4	S_8	S_{12}
S_0	$(0,0)$	$(1,0)$	$(2,0)$	$(3,0)$
S_1	$(0,1)$	$(1,1)$	$(2,1)$	$(3,1)$
S_2	$(0,2)$	$(1,2)$	$(2,2)$	$(3,2)$
S_3	$(0,3)$	$(1,3)$	$(2,3)$	$(3,3)$

(i, j) means output bit i goes to input bit j



E.g. The b_1 is bit 1 of S_0, it is mapped to bit 0 of S_4, b_{16}. Hence $P(1) = 16$.

$1-1$ bit DDT

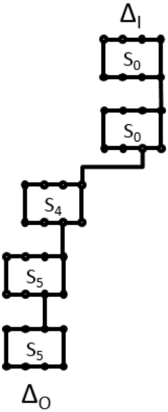
$1-1$ bit DDT as a sub-table of the DDT containing Hamming weight 1 differences.

Table: $1-1$ bit DDT Example

Δx \ Δy	1000	0100	0010	0001
bit 3 = 1000	0	2	4	0
bit 2 = 0100	0	0	0	0
bit 1 = 0010	0	0	0	0
bit 0 = 0001	0	2	2	0

An Sbox has BN3 if and only if its $1-1$ bit DDT is all zeroes.

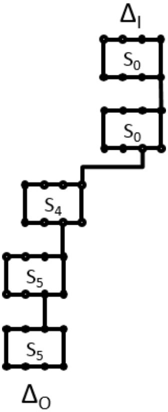
BN2 Sbox in PRESENT



$Q0$ \ $R0$	S_0	S_4	S_8	S_{12}
S_0	$(0,0)$	$(1,0)$	$(2,0)$	$(3,0)$
S_1	$(0,1)$	$(1,1)$	$(2,1)$	$(3,1)$
S_2	$(0,2)$	$(1,2)$	$(2,2)$	$(3,2)$
S_3	$(0,3)$	$(1,3)$	$(2,3)$	$(3,3)$

Δx \ Δy	bit 3	bit 2	bit 1	bit 0
bit 3	0	2	4	0
bit 2	0	0	0	0
bit 1	0	0	0	0
bit 0	0	2	2	0

BN2 Sbox in PRESENT



5 active Sboxes in 5 rounds (BN2 Sbox) vs
10 active Sboxes in 5 rounds (original).

PRESENT bit permutation is not compatible with Sboxes with BN2.

Table of Contents

Bad Output must go to Good Input (BOGI)

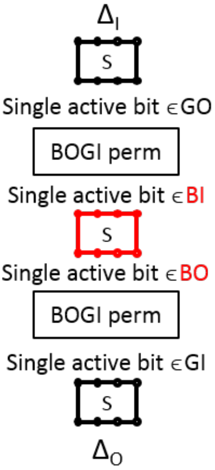
Table: $1-1$ bit DDT Example

Δx \ Δy	bit 3	bit 2	bit 1	bit 0
bit 3	0	2	4	0
bit 2	0	0	0	0
bit 1	0	0	0	0
bit 0	0	2	2	0

Let GI, GO, BI, BO denote the set of good inputs, good outputs, bad inputs and bad outputs respectively.

$$GI = \{\text{bit } 2, \text{bit } 1\},\ GO = \{\text{bit } 3, \text{bit } 0\},$$
$$BI = \{\text{bit } 3, \text{bit } 0\},\ BO = \{\text{bit } 2, \text{bit } 1\}.$$

Core Idea



Observation:
If a single active bit transition occurs, the input and output active bit must be in *BI* and *BO*.

Core idea:
We send the bit from *BO* to *GI* so that single bit transition does not happen continuously.
Same for backward direction.

Both Δ_I and Δ_O have at least 2 active bits.

$\geq$ 7 active Sboxes in 5 rounds!

BOGI Permutation

Let $\pi_1 : BO \to GI$ and $\pi_2 : GO \to (\pi_1(BO))^c$.
BOGI permutation π is the union of π_1 and π_2.

$$GI = \{\text{bit } 2, \text{bit } 1\},\ GO = \{\text{bit } 3, \text{bit } 0\},$$
$$BI = \{\text{bit } 3, \text{bit } 0\},\ BO = \{\text{bit } 2, \text{bit } 1\}.$$

For this example, π can be an identity mapping.
I.e. $\pi : \text{bit } j \mapsto \text{bit } j$.

Necessary and sufficient condition:

$$|BO| \leq |GI| \implies |GI| + |GO| \geq 4$$

Denote $|GI| + |GO|$ the score of an Sbox.

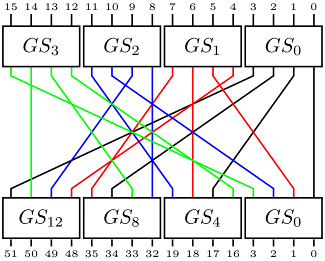
This can be extended to the $1-1$ bit LAT and linear cryptanalysis, which is the Achilles' heel of PRESENT.

GIFT-64 Group Mapping

New bit permutation based on BOGI group mapping.

Table: GIFT-64 group mapping

$Q0$ \ $R0$	GS_0	GS_4	GS_8	GS_{12}
GS_0	$(0,0)$	$(1,1)$	$(2,2)$	$(3,3)$
GS_1	$(1,1)$	$(2,2)$	$(3,3)$	$(0,0)$
GS_2	$(2,2)$	$(3,3)$	$(0,0)$	$(1,1)$
GS_3	$(3,3)$	$(0,0)$	$(1,1)$	$(2,2)$



Select an Sbox with score 4 and has BOGI identity permutation.

Table of Contents

GIFT Sbox Criteria

GIFT Sbox criteria:

1. Significantly lighter than PRESENT Sbox.
2. At least score 4 for both differential and linear cases.
3. There exists BOGI identity permutation for both differential and linear cases.
4. For Δ_I, Δ_O s.t. $p(\Delta_I \rightarrow \Delta_O) > 2^{-2}$, $wt(\Delta_I) + wt(\Delta_O) \geq 4$.

The last criterion ensures that when sub-optimal differential transition occurs, there is at least a total of 4 active Sboxes in the previous and next round.

GIFT Sbox

Our GIFT Sbox *GS* has:

- cost of 16GE, lighter than PRESENT Sbox (21.33GE),
- maximal differential probability of $2^{-1.415}$,
    - only 2 transitions with probability $2^{-1.415}$,
    - sum of Hamming weight of input and output differences is 4.
- maximal absolute linear bias of 2^{-2},
- algebraic degree 3,
- no fixed point.

Table of Contents

Differential and Linear Bounds

Table: Lower bounds for number of active Sboxes.

Cipher	DC/LC	Rounds								
		1	2	3	4	5	6	7	8	9
GIFT-64	DC	1	2	3	5	7	10	13	16	18
	LC	1	2	3	5	7	9	12	15	18
PRESENT	DC	1	2	4	6	10	12	14	16	18
	LC	1	2	3	4	5	6	7	8	9
GIFT-128	DC	1	2	3	5	7	10	13	17	19
	LC	1	2	3	5	7	9	12	14	18

GIFT matches the differential bound of PRESENT— an average of 2 active Sboxes per round.
In addition, GIFT achieved the same ratio for linear bound at 9-round where PRESENT could not.

Differential and Linear Probabilities

Table: 9-round Differential/Linear Probabilities

Cipher	No. of Rounds	Differential Probability	Linear Hull Effect	Est. Rounds Needed
GIFT-64	28	$2^{44.415}$	$2^{49.997}$	14
PRESENT	31	$2^{40.702}$	$2^{27.186}$	22
GIFT-128	40	$2^{46.99}$	$2^{45.99}$	27

Table of Contents

Round-based Implementation

Comparison of performance metrics for round based implementations synthesized with STM 90nm Standard cell library.

Cipher	Area (GE)	Delay (ns)	Cycles	TP_{MAX} (MBit/s)	Power (μW) (@10MHz)	Energy (pJ)
GIFT-64-128	1345	1.83	29	1249.0	74.8	216.9
SKINNY-64-128	1477	1.84	37	966.2	80.3	297.0
PRESENT 64/128	1560	1.63	33	1227.0	71.1	234.6
SIMON 64/128	1458	1.83	45	794.8	72.7	327.3
GIFT-128-128	1997	1.85	41	1729.7	116.6	478.1
SKINNY-128-128	2104	1.85	41	1729.7	132.5	543.3
SIMON 128/128	2064	1.87	69	1006.6	105.6	728.6
AES 128	7215	3.83	11	3038.2	730.3	803.3

Bit-slice Implementation

Bitslice software implementations of GIFT and other lightweight block ciphers. Performances are given in cycles per byte, with messages composed of 2000 64-bit blocks to obtain the results.

Cipher	**Speed** (c/B)	**Cipher**	**Speed** (c/B)
GIFT-64-128	2.10	GIFT-128-128	2.57
SKINNY-64-128	2.88	SKINNY-128-128	4.70
SIMON-64-128	1.74	SIMON-128-128	2.55

Table of Contents

Conclusion

- Propose new lightweight block cipher with 2 block sizes, GIFT-64 and GIFT-128.
- Improvement of PRESENT:
  - remove Sbox constraint of BN3,
  - use lighter Sbox than PRESENT Sbox,
  - prevent the LC weakness in PRESENT,
  - improve performances,
  - extend to 128-bit block size.
- Strong against classical DC/LC and other cryptanalysis.
- Better performances than existing lightweight block ciphers: area, throughput, energy.

Thank you. :)