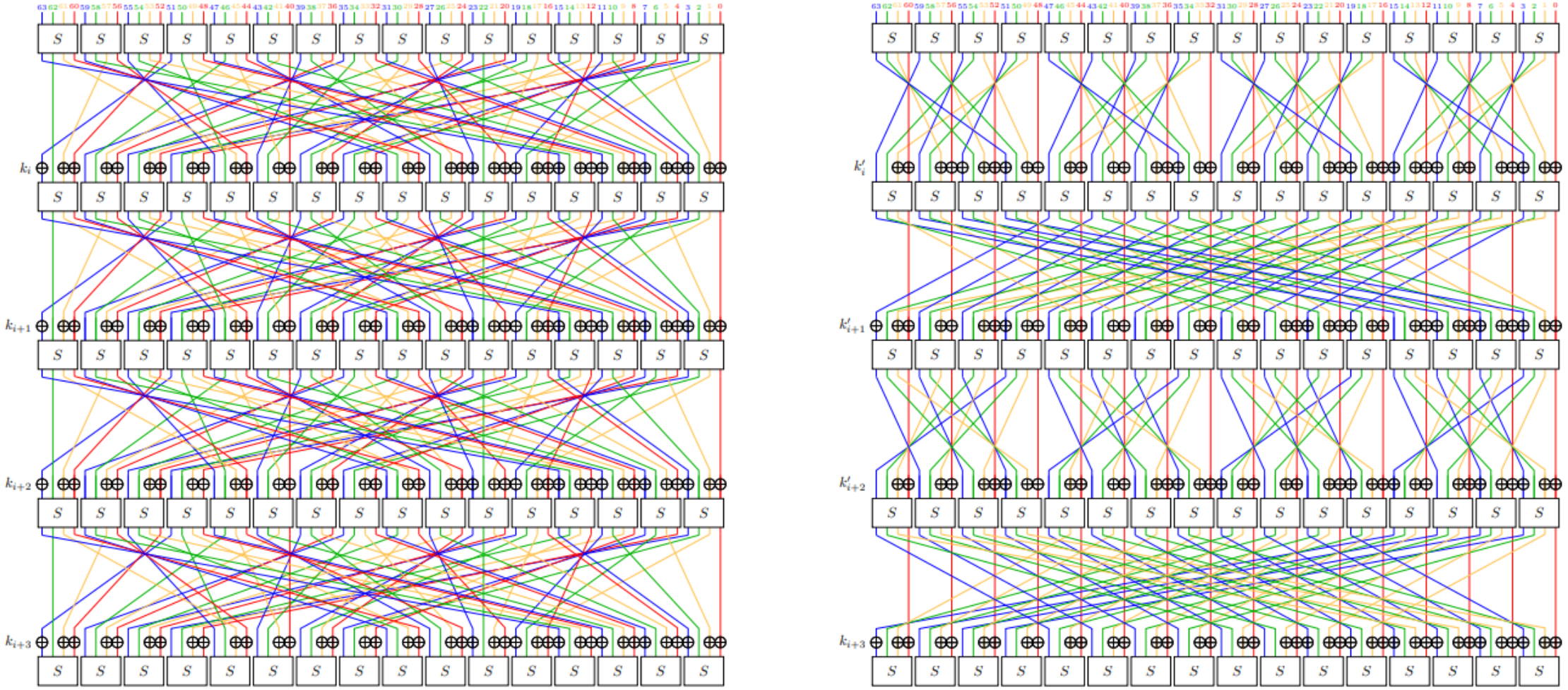
Fixslicing PIPO

https://youtu.be/whAoa22y640

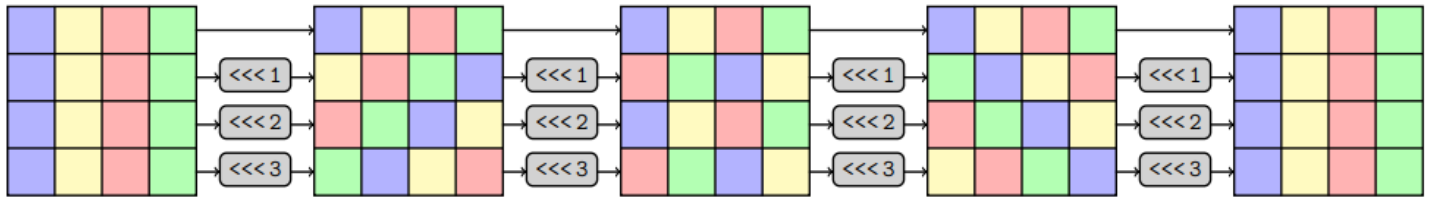
Fixslicing GFIT

- Adomnicai et. al “Fixslicing: A New GIFT Representation”(2020)
- 하드웨어 지향 GIFT 블록 암호에 대한 새로운 표현
- **몇 번의 회전**만을 사용하여 GIFT의 매우 효율적인 소프트웨어 비트슬라이싱 구현

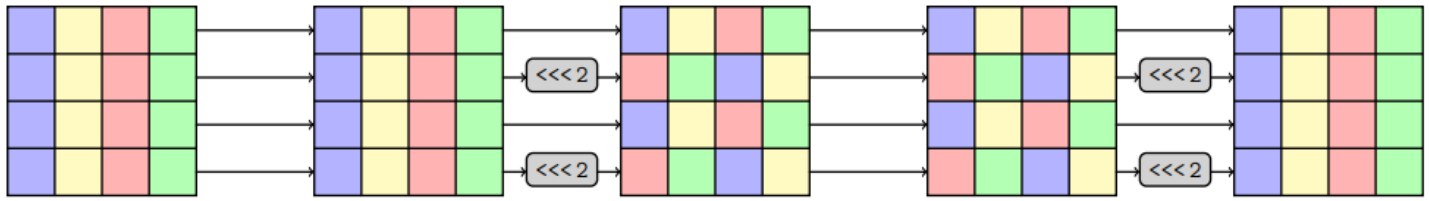


Fixslicing AES

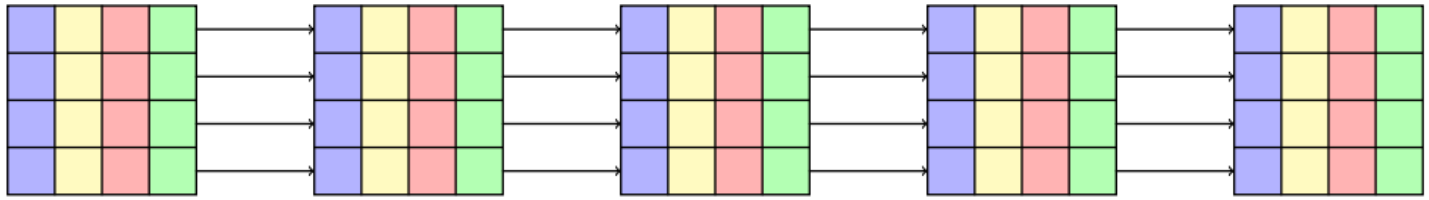
- Adomnicai et. al “Fixslicing AES-like Ciphers”(2021)
- 다른 암호에도 적용될 수 있음을 보임
  - Fixslicing 기법을 적용하여 PIPO의 빠른 구현 가능성
  - 32bit 임베디드 Cortex-m에서는 구현 결과 없음



(a) Classical



(b) Semi-fixsliced

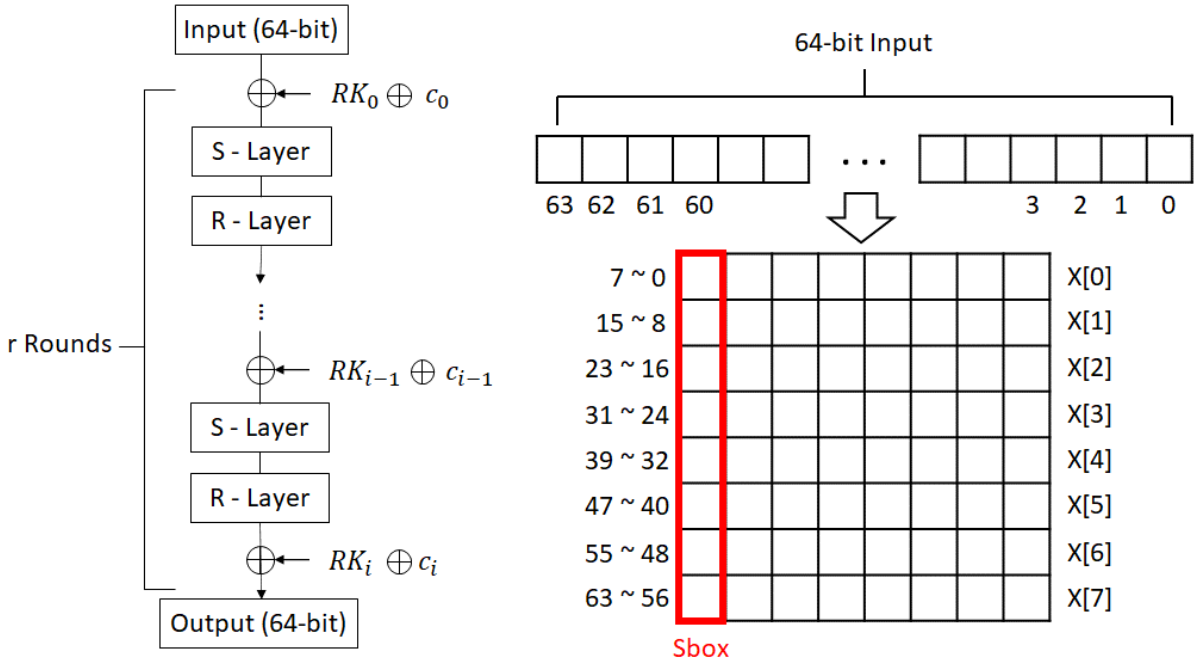


(c) Fully-fixsliced

PIPO Revisit

- 8bit 임베디드 환경 최적화
- 효율적인 비트슬라이싱 구현이 가능하도록 설계

- 기존 Fixslicing 기법 구현은 Permuation Layer을 변형하여 구현
- PIPO에는 Substitution Layer을 변형 하여 적용함



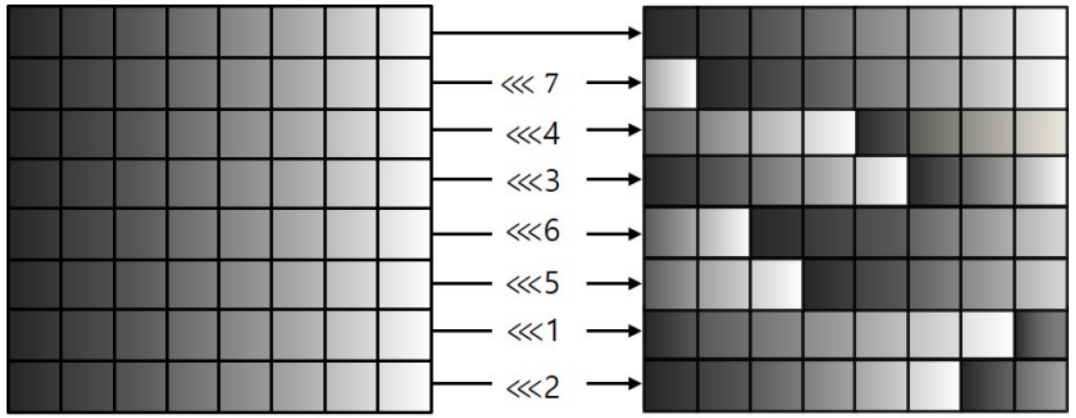


Fig. 3. R-layer

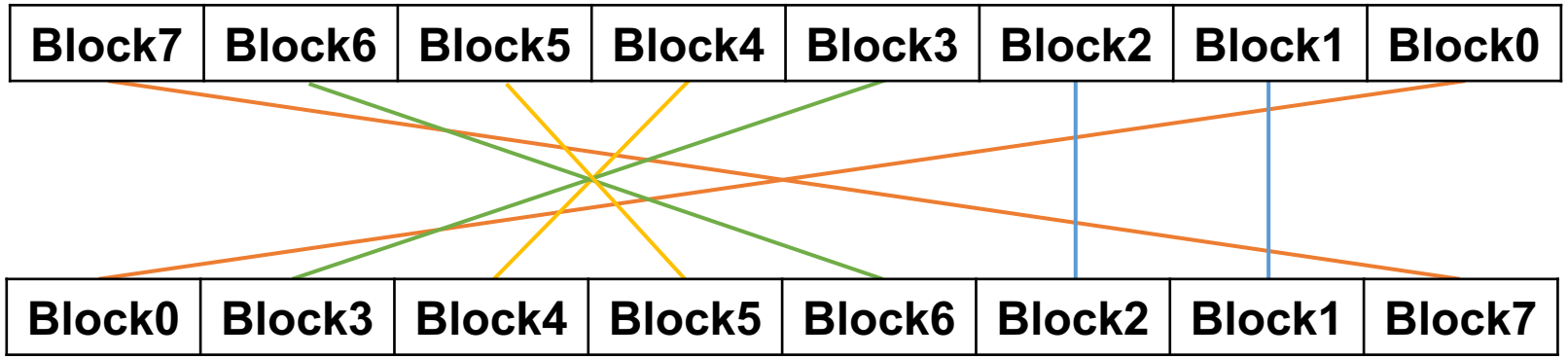
S-layer

```
#define PIPO_SBOX(X0, X1, X2, X3, X4, X5, X6, X7)                \
        X5 ^= (X7 & X6); X4 ^= (X3 & X5);                        \
        X7 ^= X4; X6 ^= X3;                                      \
        X3 ^= (X4 | X5); X5 ^= X7;                               \
        X4 ^= (X5 & X6); X2 ^= X1 & X0;                          \
        X0 ^= X2 | X1; X1 ^= X2 | X0;                            \
        X2 = ~X2; X7 ^= X1;                                      \
        X3 ^= X2; X4 ^= X0;                                      \
        X6 ^= (X7 & X5); T0 = X7^X6;                             \
        X6 ^= (X4 | X3); T1 = X3^X5                              \
        X5 ^= (X6 | X4); X2 ^= T0;                               \
        X[2] ^= T[0];   T[0] = X[1] ^ T[2];                      \
        X[1] = X[0]^T[1];   X[0] = X[7];                         \
        X[7] = T[0];  T[1] = X[3];                               \
         X[3] = X[6];   X[6] = T[1];                             \
        T[2] = X[4];   X[4] = X[5];   X[5] = T[2];               \
```

```
#define PIPO_SBOX(X0, X1, X2, X3, X4, X5, X6, X7)                \
        X5 ^= (X7 & X6); X4 ^= (X3 & X5);                        \
        X7 ^= X4; X6 ^= X3;                                      \
        X3 ^= (X4 | X5); X5 ^= X7;                               \
        X4 ^= (X5 & X6); X2 ^= X1 & X0;                          \
        X0 ^= X2 | X1; X1 ^= X2 | X0;                            \
        X2 = ~X2; X7 ^= X1;                                      \
        X3 ^= X2; X4 ^= X0;                                      \
        X6 ^= (X7 & X5); T0 = X7^X6;                             \
        X6 ^= (X4 | X3); T1 = X3^X5                              \
        X5 ^= (X6 | X4); X2 ^= T0;                               \
        T2 = X7; X1 ^= X4 ^ (T1 & T0);                           \
        X0 = X0^T1;                                              \
```

S-layer

(0, 3, 4, 5, 6, 2, 1, 7)의 순서에서 (7, 6, 5, 4, 3, 2, 1, 0)로 정렬



Classic

Register 0	7	6	5	4	3	2	1	0
Register 1	15	14	13	12	11	10	9	8
Register 2	23	22	21	20	19	18	17	16
Register 3	31	30	29	28	27	26	25	24
Register 4	39	38	37	36	35	34	33	32
Register 5	47	46	45	44	43	42	41	40
Register 6	55	54	53	52	51	50	49	48
Register 7	63	62	61	60	59	58	57	56

Fixslicing

63	62	61	60	59	58	57	56
15	14	13	12	11	10	9	8
23	22	21	20	19	18	17	16
55	54	53	52	51	50	49	48
47	46	45	44	43	42	41	40
39	38	37	36	35	34	33	32
31	30	29	28	27	26	25	24
7	6	5	4	3	2	1	0

R-layer

Classic

7	6	5	4	3	2	1	0
15	14	13	12	11	10	9	8
23	22	21	20	19	18	17	16
31	30	29	28	27	26	25	24
39	38	37	36	35	34	33	32
47	46	45	44	43	42	41	40
55	54	53	52	51	50	49	48
63	62	61	60	59	58	57	56

<<< 7							
<<< 4							
<<< 3							
<<< 6							
<<< 5							
<<< 1							
<<< 2							

7	6	5	4	3	2	1	0
8	15	14	13	12	11	10	9
19	18	17	16	23	22	21	20
28	27	26	25	24	31	30	29
34	33	32	39	38	37	36	35
42	41	40	47	46	45	44	43
54	53	52	51	50	49	48	55
61	60	59	58	57	56	63	62

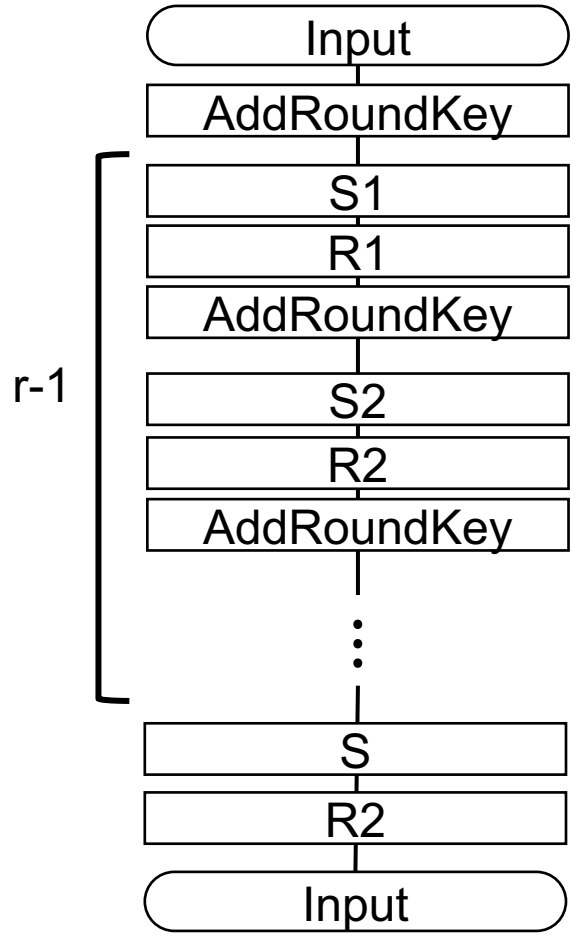
Fixslicing

63	62	61	60	59	58	57	56
15	14	13	12	11	10	9	8
23	22	21	20	19	18	17	16
55	54	53	52	51	50	49	48
47	46	45	44	43	42	41	40
39	38	37	36	35	34	33	32
31	30	29	28	27	26	25	24
7	6	5	4	3	2	1	0

<<< 2							
<<< 7							
<<< 4							
<<< 1							
<<< 5							
<<< 6							
<<< 3							

61	60	59	58	57	56	63	62
8	15	14	13	12	11	10	9
19	18	17	16	23	22	21	20
54	53	52	51	50	49	48	55
42	41	40	47	46	45	44	43
34	33	32	39	38	37	36	35
28	27	26	25	24	31	30	29
7	6	5	4	3	2	1	0

Fixslicing PIPO



S : Classic S-Lyaer - input {0, 1, 2, 3, 4, 5, 6, 7}

S1 : Fixslicing S-Layer - input {0, 1, 2, 3, 4, 5, 6, 7}
S2 : Fixslicing S-Layer - input {7, 1, 2, 6, 5, 4, 3, 0}

R1 : Fixslicing R-Layer - left rotation {0, 7, 4, 3, 6, 5, 1, 2}
R2 : Fixslicing R-Layer - left rotation {2, 7, 4, 1, 5, 6, 3, 0}

성능 비교

- 어셈블리 구현
  Microchip Studio 시뮬레이터 8bit ATmega128 보드

	Classic	Fixslicing	속도 향상
8bit ATmega128	1,574	1,501	13%

- C 구현
  32bit cortex-m3 보드

	Classic	Fixslicing	속도 향상
32bit cortex-m3	3,404	2,216	53%

추후 연구

- 32bit Cortex-m3
- GPU

Q & A