Linearization and Message Modification Techniques for Hash Function Cryptanalysis

Jian Guo

Institute for Infocomm Research, Singapore.

ASK 2011, 30 August 2011

Overview

- Introduction
- Linearization and Message Modifications
- Application to ARIRANG
- Conclusions

Hash Functions - Definitions and Properties

Definition

A hash function h is a function to take a bit string of arbitrary length as input and produces a fixed-size output of n bits.

Hash Functions - Definitions and Properties

Definition

A hash function h is a function to take a bit string of arbitrary length as input and produces a fixed-size output of n bits.

Properties

- **Collision Resistance**: it is computationally difficult to find x and x', such that $h(x) = h(x')$ with expected complexity $2^{n/2}$.

Hash Functions - Definitions and Properties

Definition

A hash function h is a function to take a bit string of arbitrary length as input and produces a fixed-size output of n bits.

Properties

- **Collision Resistance**: it is computationally difficult to find x and x', such that $h(x) = h(x')$ with expected complexity $2^{n/2}$.
- **Preimage Resistance**: given a digest t, it is computationally difficult to find x, such that $h(x) = t$ with expected complexity 2^n.

Hash Functions - Definitions and Properties

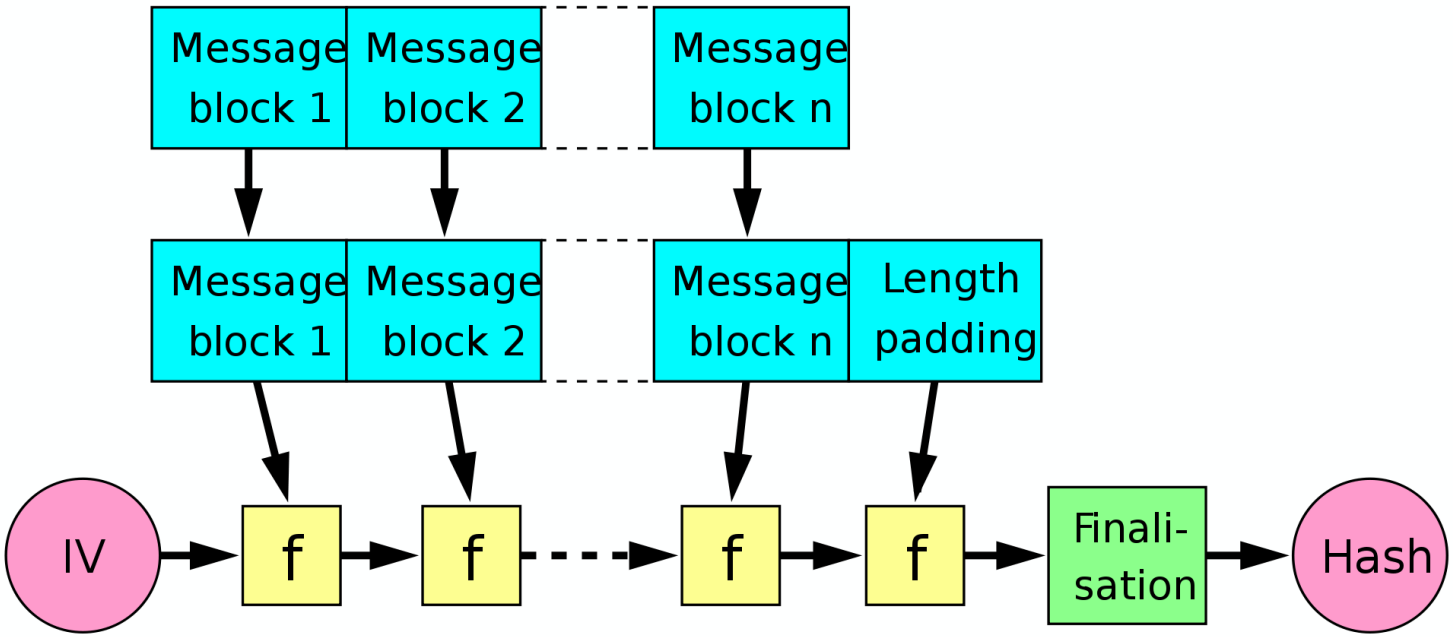
Definition

A hash function h is a function to take a bit string of arbitrary length as input and produces a fixed-size output of n bits.

Properties

- **Collision Resistance**: it is computationally difficult to find x and x', such that $h(x) = h(x')$ with expected complexity $2^{n/2}$.
- **Preimage Resistance**: given a digest t, it is computationally difficult to find x, such that $h(x) = t$ with expected complexity 2^n.
- **Second Preimage Resistance**: given a message x, it is computationally difficult to find $x' \neq x$, such that $h(x) = h(x')$ with expected complexity 2^{n-k}.

Merkle-Damgård Strengthening



by Merkle and Damgård in 1989, with proof for collision resistance reduction, i.e., if the compression function f is collision resistant, then the hash function.

Davies-Meyer Construction and Collisions

Davies-Meyer Construction

To construct a compression function f from block cipher E:

$$f(CV, m) = E_m(CV) \oplus CV$$

Davies-Meyer Construction and Collisions

Davies-Meyer Construction

To construct a compression function f from block cipher E:

$$f(CV, m) = E_m(CV) \oplus CV$$

Compression Function Collisions

- free-start collision: $f(CV, m) = f(CV', m')$.
- semi-free-start collision: $f(CV, m) = f(CV, m')$.

Davies-Meyer Construction and Collisions

Davies-Meyer Construction

To construct a compression function f from block cipher E:

$$f(CV, m) = E_m(CV) \oplus CV$$

Compression Function Collisions

- free-start collision: $f(CV, m) = f(CV', m')$.
- semi-free-start collision: $f(CV, m) = f(CV, m')$.

Note: collisions of compression function do not necessarily, and in most of the cases do not, lead to collisions of hash directly. However, it breaks the assumption of the collision proof, hence weakens the confidence on the hash securities.

Linearization

XOR Differences

In many designs, Addition-Rotation-Xor (ARX) are involved.

XOR Differences

Let $\Delta = x \oplus x'$, and denote $g(\Delta)$ as $g(x) \oplus g(x')$, then:

- when $g(x) = x \oplus C$ (C is a constant), then $g(\Delta) = g(x) \oplus g(x') = (x \oplus C) \oplus (x' \oplus C) = x \oplus x' = \Delta$.

XOR Differences

In many designs, Addition-Rotation-Xor (ARX) are involved.

XOR Differences

Let $\Delta = x \oplus x'$, and denote $g(\Delta)$ as $g(x) \oplus g(x')$, then:

- when $g(x) = x \oplus C$ (C is a constant), then $g(\Delta) = g(x) \oplus g(x') = (x \oplus C) \oplus (x' \oplus C) = x \oplus x' = \Delta$.
- when $g(x) = x \lll r$ (r is a rotation constant), then $g(\Delta) = (x \lll r) \oplus (x' \lll r) = (x \oplus x') \lll r = \Delta \lll r$.

XOR Differences

In many designs, Addition-Rotation-Xor (ARX) are involved.

XOR Differences

Let $\Delta = x \oplus x'$, and denote $g(\Delta)$ as $g(x) \oplus g(x')$, then:

- when $g(x) = x \oplus C$ (C is a constant), then $g(\Delta) = g(x) \oplus g(x') = (x \oplus C) \oplus (x' \oplus C) = x \oplus x' = \Delta$.
- when $g(x) = x \lll r$ (r is a rotation constant), then $g(\Delta) = (x \lll r) \oplus (x' \lll r) = (x \oplus x') \lll r = \Delta \lll r$.

However,

- when $g(x) = x + C$ (C is a constant), $g(\Delta) = (x + C) \oplus (x' + C)$, which is not Δ for some cases.

XOR Differences and Addition Modulo 2^8

Consider $g(x) = x + C$ with the simplest case, i.e., $x = 0, x' = 1$, hence $\Delta = 1$

C	$g(x)$	$g(x')$	$g(\Delta)$	Prob.
0	0	1	1	2^{-1}
01	01	10	11	2^{-2}
011	11	100	111	2^{-3}
...				
11111111	11111111	00000000	11111111	2^{-8}

XOR Differences and Addition Modulo 2^8

Consider $g(x) = x + C$ with the simplest case, i.e., $x = 0, x' = 1$, hence $\Delta = 1$

C	$g(x)$	$g(x')$	$g(\Delta)$	Prob.
0	0	1	1	2^{-1}
01	01	10	11	2^{-2}
011	11	100	111	2^{-3}
...				
11111111	11111111	00000000	11111111	2^{-8}

Linearization

Approximate the behaviour of addition, w.r.t. XOR differences, as XOR with probability $2^{-|\Delta|}$.

Special Notes

MSB is free

Difference in most significant bit (MSB) preserves with probability 1, i.e., with $x = 0, x' = 2^{k-1}, g(x) = (x + C) \mod 2^k$, $g(\Delta) = \Delta$ for any C.

Special Notes

MSB is free

Difference in most significant bit (MSB) preserves with probability 1, i.e., with $x = 0, x' = 2^{k-1}, g(x) = (x + C) \mod 2^k$, $g(\Delta) = \Delta$ for any C.
Linearization probability is $2^{-|\Delta^*|}$, where Δ^* is the difference Δ excluding MSB.

Special Notes

MSB is free

Difference in most significant bit (MSB) preserves with probability 1, i.e., with $x = 0, x' = 2^{k-1}, g(x) = (x + C) \mod 2^k$, $g(\Delta) = \Delta$ for any C.
Linearization probability is $2^{-|\Delta^*|}$, where Δ^* is the difference Δ excluding MSB.

Rotation Invariant Differences

Δ is called rotation invariant w.r.t r, if $\Delta \lll r = \Delta$. E.g., 10001000 is rotation invariant w.r.t. $r = 4$ with $k = 8$.

Special Notes

MSB is free

Difference in most significant bit (MSB) preserves with probability 1, i.e., with $x = 0, x' = 2^{k-1}, g(x) = (x + C) \mod 2^k$, $g(\Delta) = \Delta$ for any C.
Linearization probability is $2^{-|\Delta^*|}$, where Δ^* is the difference Δ excluding MSB.

Rotation Invariant Differences

Δ is called rotation invariant w.r.t r, if $\Delta \lll r = \Delta$. E.g., 10001000 is rotation invariant w.r.t. $r = 4$ with $k = 8$.
ALL-ONE difference $(111 \cdots 111)$ is rotation invariant w.r.t. any r, k.

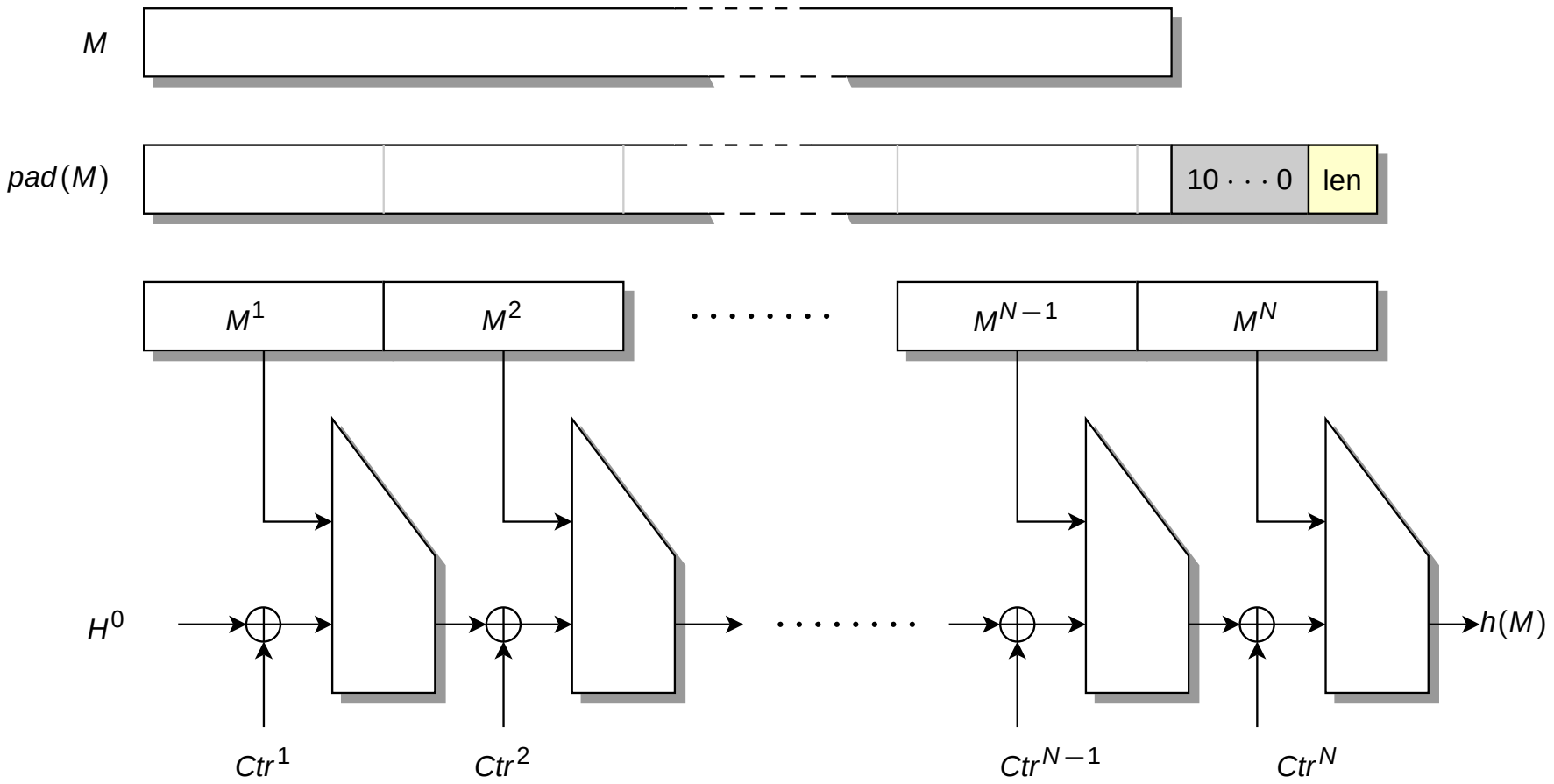
ARIRANG

SHA-3 candidate ARIRANG

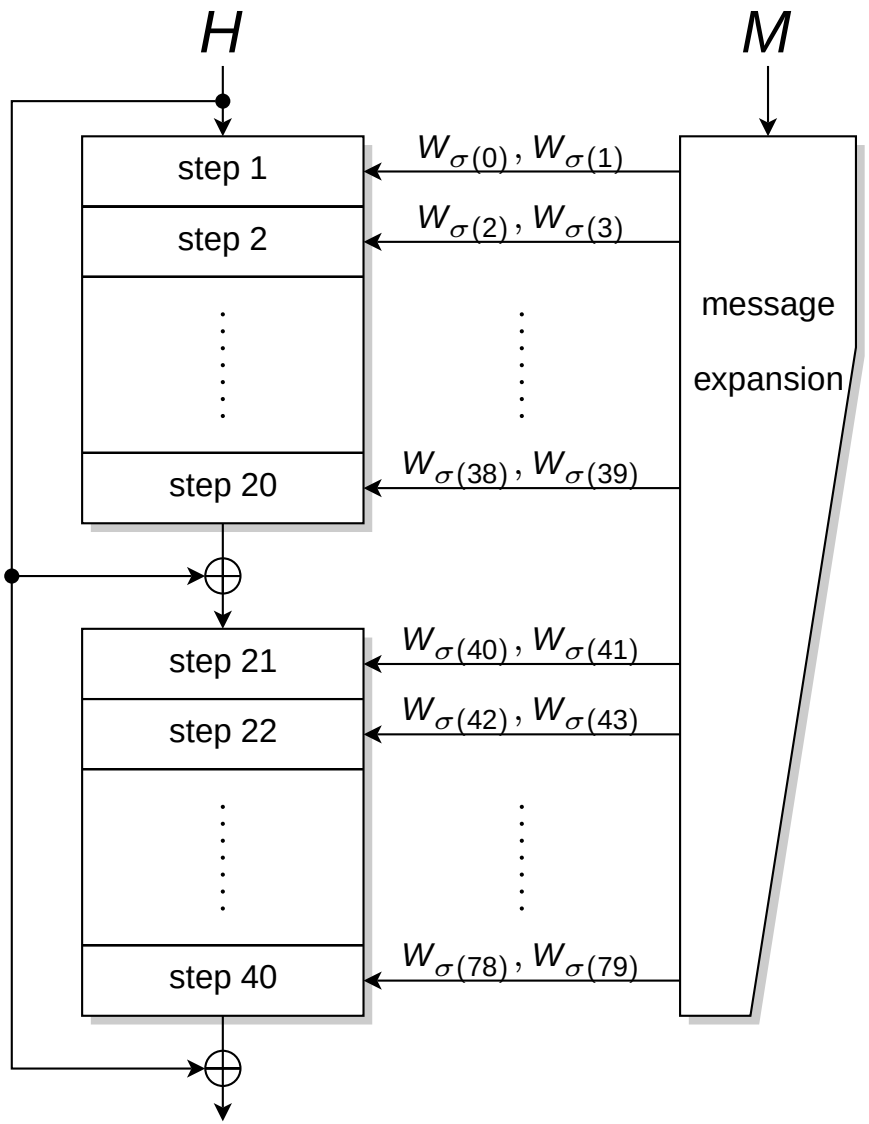
- One of the first round SHA-3 candidates
- Designed by a team from Center for Information Security Technologies (CIST), Korea University:

  Donghoon Chang, Seokhie Hong, Changheon Kang, Jinkeon Kang, Jongsung Kim, Changhoon Lee, Jesang Lee, Jongtae Lee, Sangjin Lee, Yuseop Lee, Jongin Lim, Jaechul Sung
- Design mixing parts from AES-based (S-box, MixColumn) and ARX designs (word addition, rotations, xor)
- Follows Merkle-Damgård strengthening

Hash function



Compression function



Message expansion

1. Generate 16 more words as linear combinations of $M_0, \ldots, M_{15}$
2. Pick (with repetitions) 80 words out of the 32 words obtained in the previous step

$$M_0, \ldots, M_{15} \longrightarrow$$

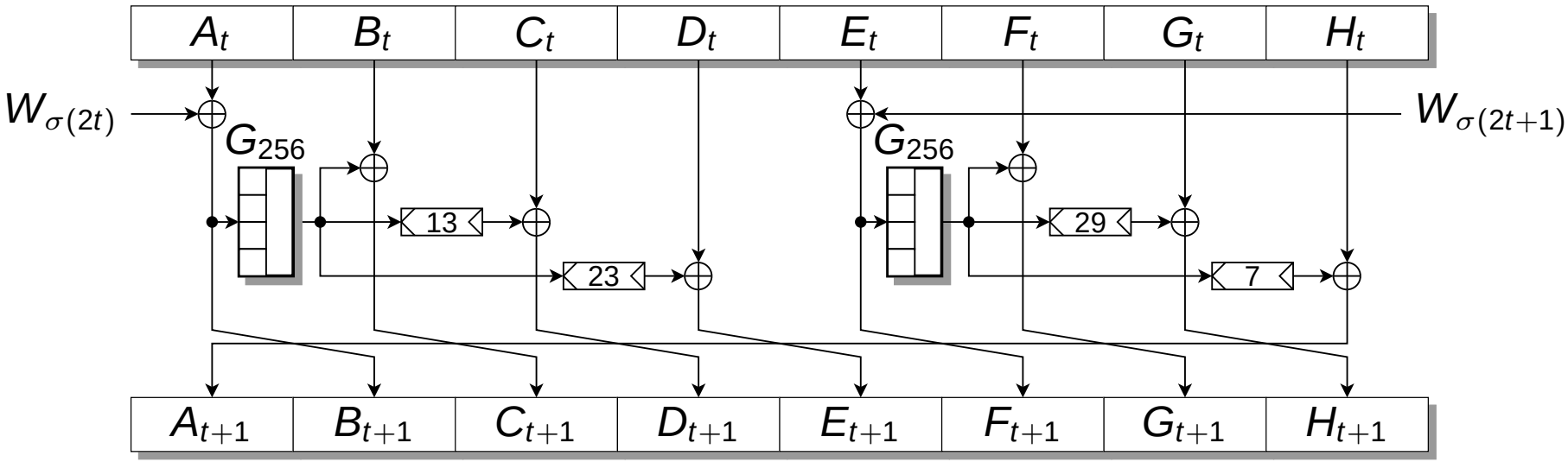
$$\begin{aligned}
W_{16} &\leftarrow (M_9 \oplus M_{11} \oplus M_{13} \oplus M_{15} \oplus K_0) \lll r_0 \\
W_{17} &\leftarrow (M_8 \oplus M_{10} \oplus M_{12} \oplus M_{14} \oplus K_1) \lll r_1 \\
W_{18} &\leftarrow (M_1 \oplus M_3 \oplus M_5 \oplus M_7 \oplus K_2) \lll r_2 \\
W_{19} &\leftarrow (M_0 \oplus M_2 \oplus M_4 \oplus M_6 \oplus K_3) \lll r_3 \\
W_{20} &\leftarrow (M_{14} \oplus M_4 \oplus M_{10} \oplus M_0 \oplus K_4) \lll r_0 \\
W_{21} &\leftarrow (M_{11} \oplus M_1 \oplus M_7 \oplus M_{13} \oplus K_5) \lll r_1 \\
W_{22} &\leftarrow (M_6 \oplus M_{12} \oplus M_2 \oplus M_8 \oplus K_6) \lll r_2 \\
W_{23} &\leftarrow (M_3 \oplus M_9 \oplus M_{15} \oplus M_5 \oplus K_7) \lll r_3 \\
W_{24} &\leftarrow (M_{13} \oplus M_{15} \oplus M_1 \oplus M_3 \oplus K_8) \lll r_0 \\
W_{25} &\leftarrow (M_4 \oplus M_6 \oplus M_8 \oplus M_{10} \oplus K_9) \lll r_1 \\
W_{26} &\leftarrow (M_5 \oplus M_7 \oplus M_9 \oplus M_{11} \oplus K_{10}) \lll r_2 \\
W_{27} &\leftarrow (M_{12} \oplus M_{14} \oplus M_0 \oplus M_2 \oplus K_{11}) \lll r_3 \\
W_{28} &\leftarrow (M_{10} \oplus M_0 \oplus M_6 \oplus M_{12} \oplus K_{12}) \lll r_0 \\
W_{29} &\leftarrow (M_{15} \oplus M_5 \oplus M_{11} \oplus M_1 \oplus K_{13}) \lll r_1 \\
W_{30} &\leftarrow (M_2 \oplus M_8 \oplus M_{14} \oplus M_4 \oplus K_{14}) \lll r_2 \\
W_{31} &\leftarrow (M_7 \oplus M_{13} \oplus M_3 \oplus M_9 \oplus K_{15}) \lll r_3
\end{aligned}$$

$\longrightarrow$

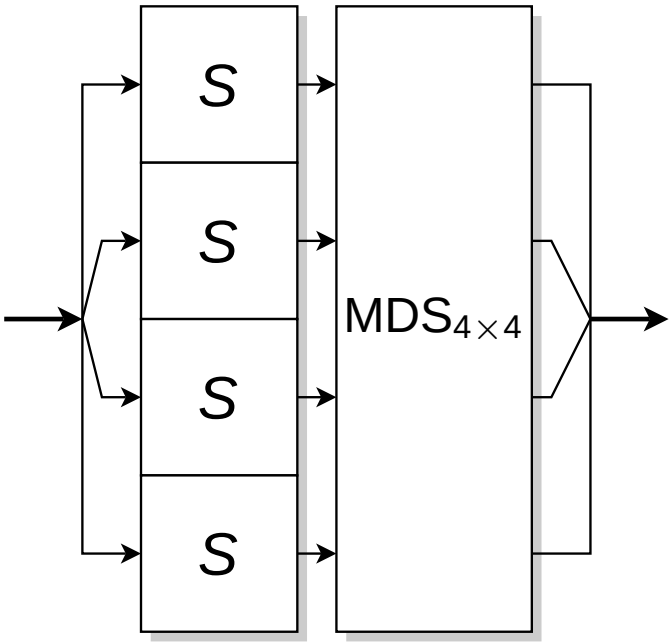
$\sigma(i)$	$\sigma(i)$
16, 17	24, 25
0, 1	12, 5
2, 3	14, 7
4, 5	0, 9
6, 7	2, 11
18, 19	26, 27
8, 9	4, 13
10, 11	6, 15
12, 13	8, 1
14, 15	10, 3
20,21	28, 29
3, 6	7, 2
9,12	13, 8
15, 2	3, 14
5, 8	9, 4
22,23	30, 31
11,14	15, 10
1, 4	5, 0
7,10	11, 6
13, 0	1, 12

Step transformation

- transforms 8 32-bit words of the state and 8 words of the expanded message to new state
- uses 32-bit rotations, XORs and a 32×32 bit function G_{256}
- only non-linear (over $\mathbb{F}_2$) part is G_{256}



Function G_{256}



32×32 composite "megabox":

- 4 bytewise AES S-boxes
- Followed by $MDS_{4\times4}$ transformation (AES MixColumn)

ARIRANG-512 uses a similar function G_{512} defined on 8 32-bit words and using $MDS_{8\times8}$.

Basic observations

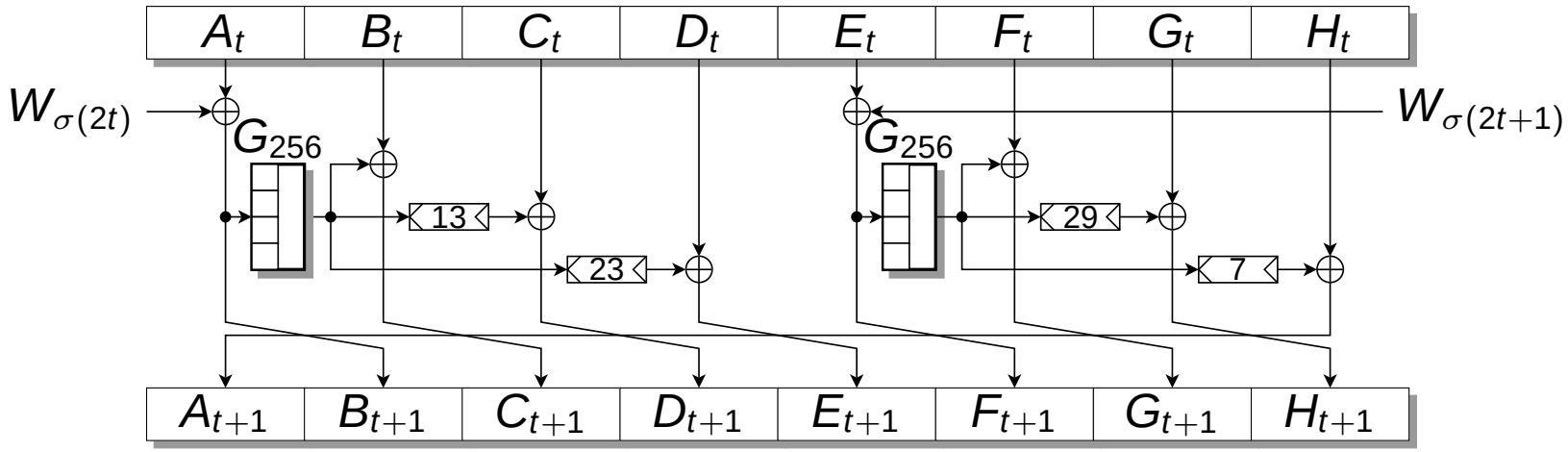
- $MDS_{4\times4}$ has fixed points of the form (a, a, a, a)

$$MDS_{4\times4} = \begin{bmatrix} z & z+1 & 1 & 1 \\ 1 & z & z+1 & 1 \\ 1 & 1 & z & z+1 \\ z+1 & 1 & 1 & z \end{bmatrix}$$

- S-box differential $\texttt{0xff} \rightarrow \texttt{0xff}$ is possible with prob. 2^{-7}.
- Differential $\texttt{0xffffffff} \rightarrow \texttt{0xffffffff}$ for G_{256} has probability 2^{-28}
- 512-bit variant: no fixed points for MDS, but still can get all-ones to all-ones differences

All-one differences

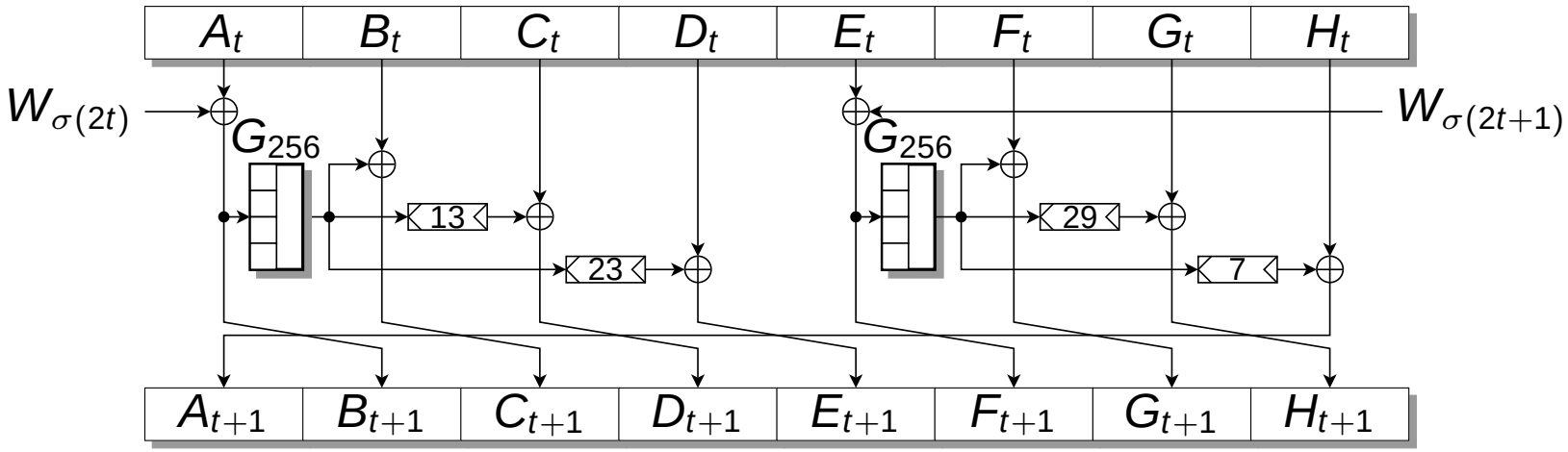
If we consider only all-one differences:



All-one differences

If we consider only all-one differences:

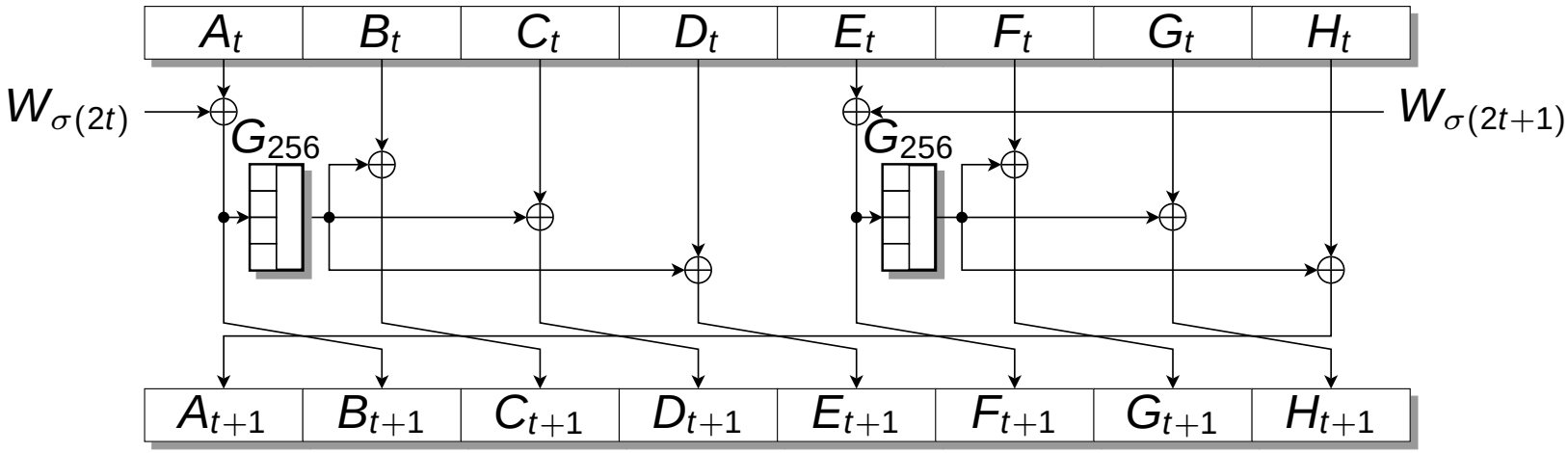
- rotations in step function do not play any role



All-one differences

If we consider only all-one differences:

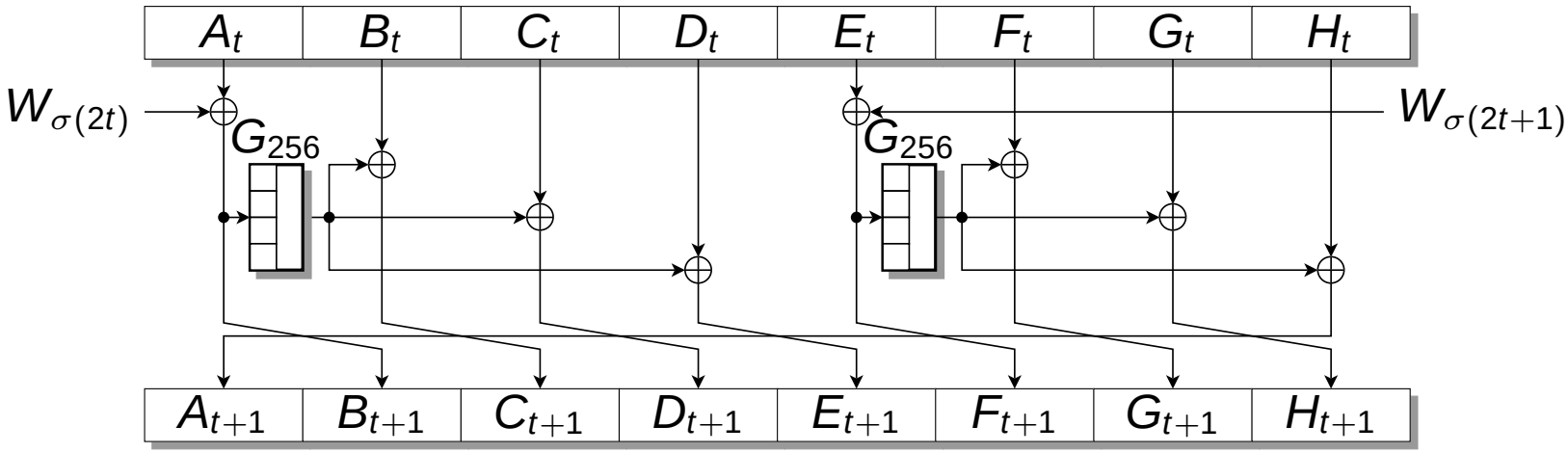
- rotations in step function do not play any role



All-one differences

If we consider only all-one differences:

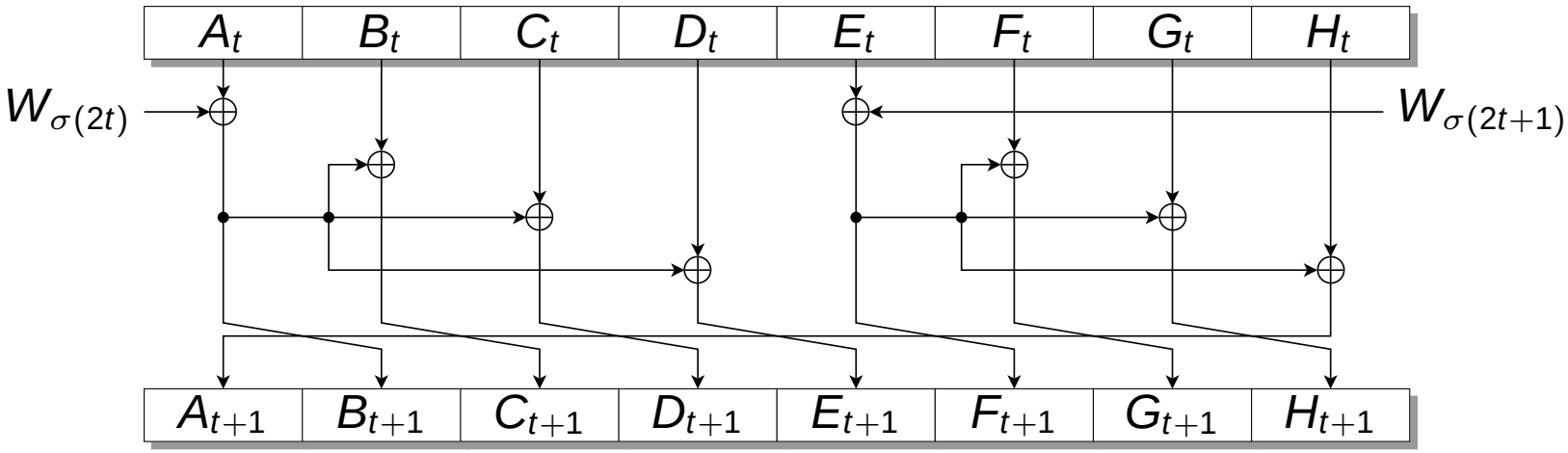
- rotations in step function do not play any role
- we can replace G_{256} with identity (with prob. 2^{-28}), i.e., 2^4 values.



All-one differences

If we consider only all-one differences:

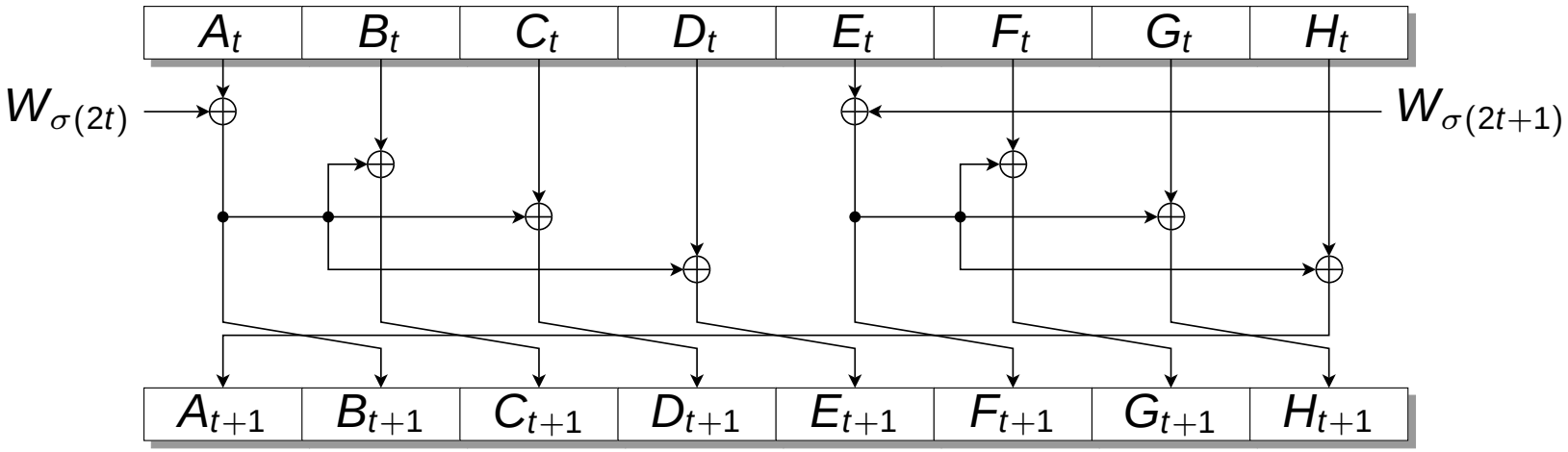
- rotations in step function do not play any role
- we can replace G_{256} with identity (with prob. 2^{-28}), i.e., 2^4 values.



All-one differences

If we consider only all-one differences:

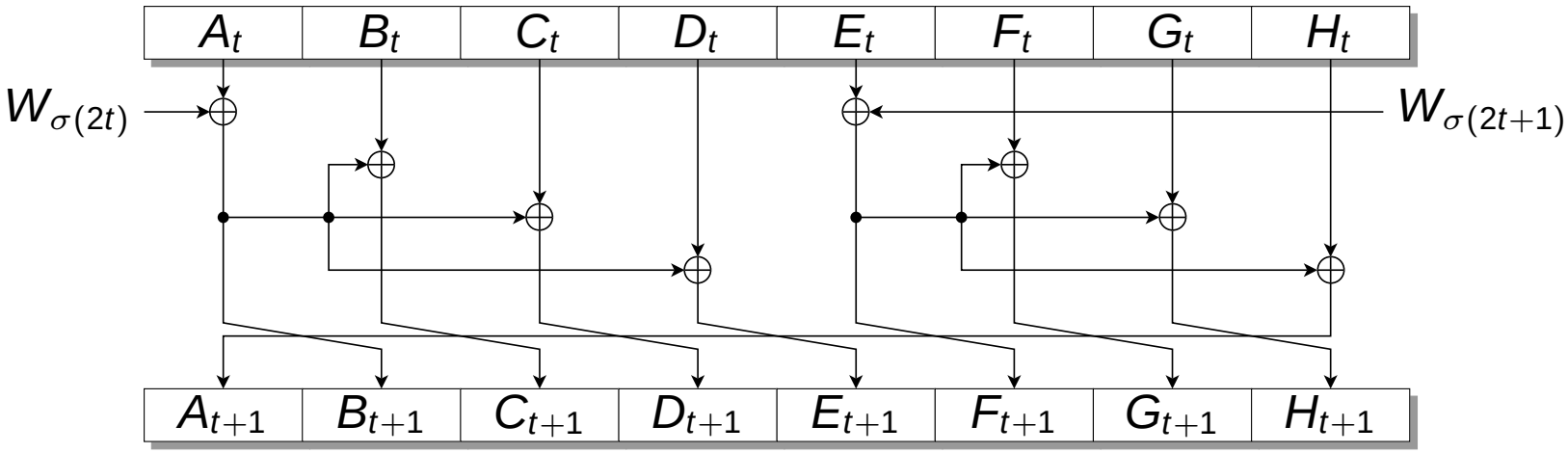
- rotations in step function do not play any role
- we can replace G_{256} with identity (with prob. 2^{-28}), i.e., 2^4 values.
- One register can be represented as a single bit (truncated differential)



All-one differences

If we consider only all-one differences:

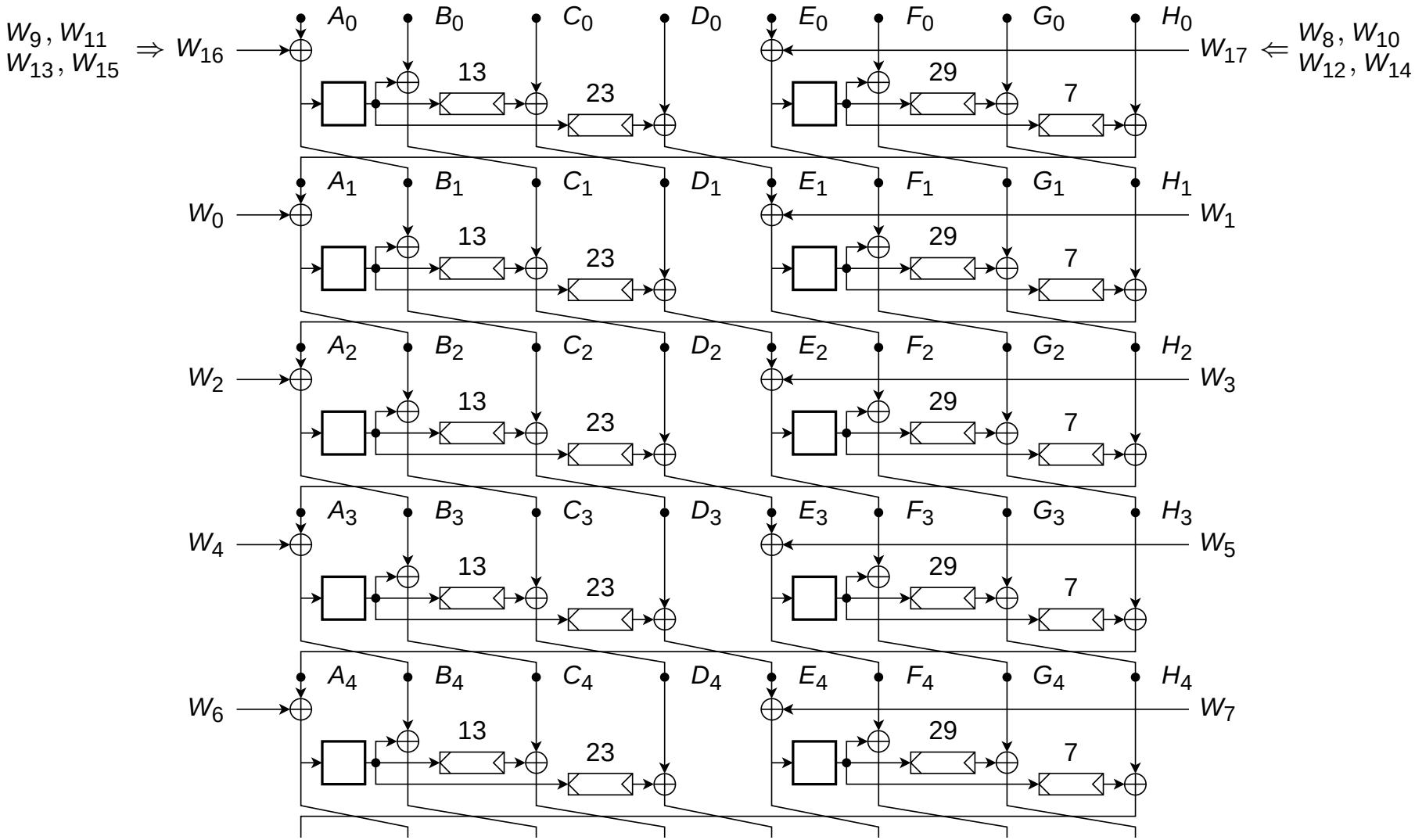
- rotations in step function do not play any role
- we can replace G_{256} with identity (with prob. 2^{-28}), i.e., 2^4 values.
- One register can be represented as a single bit (truncated differential)
- Linearized model has $8 + 16$ variables: we have 2^{24} paths

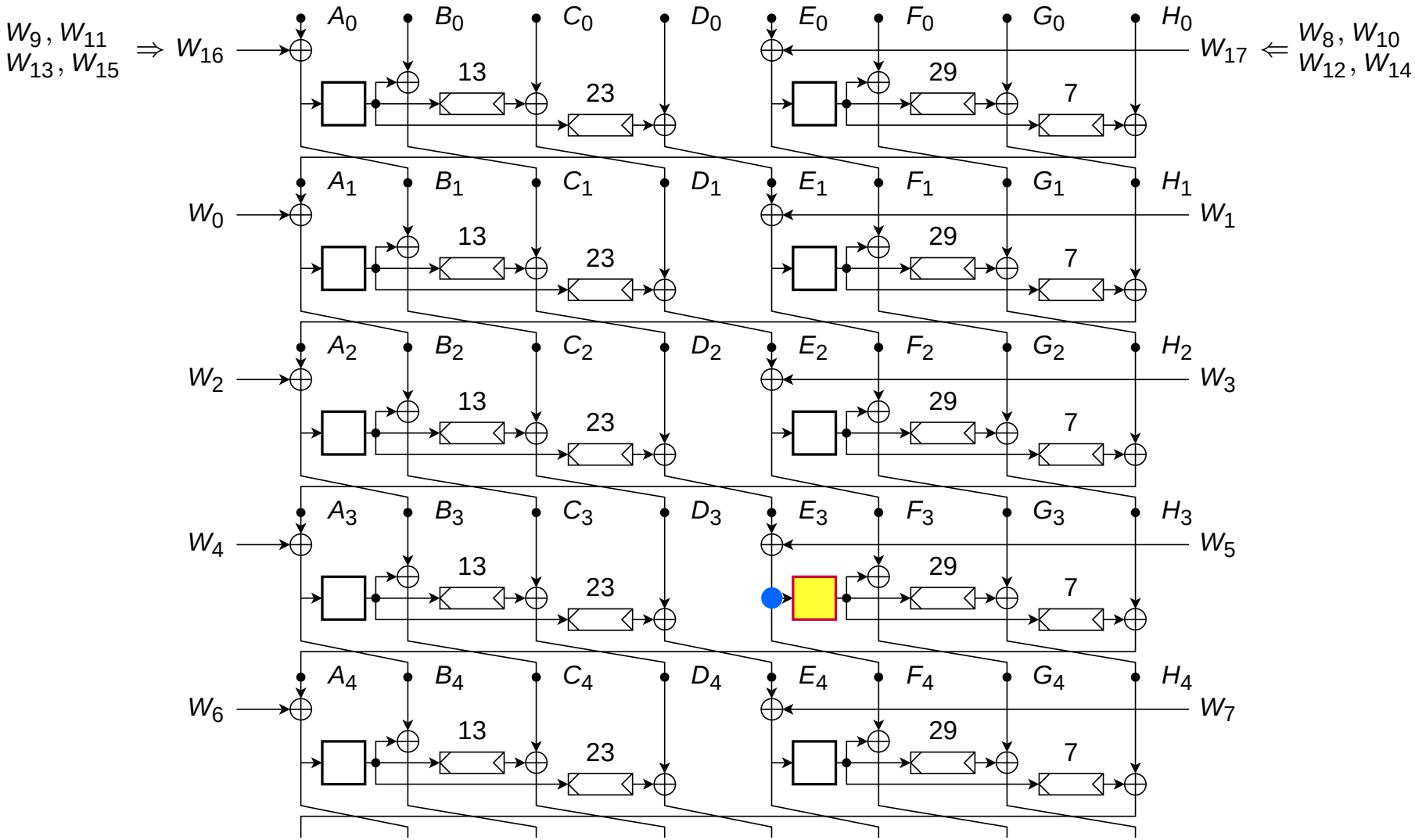


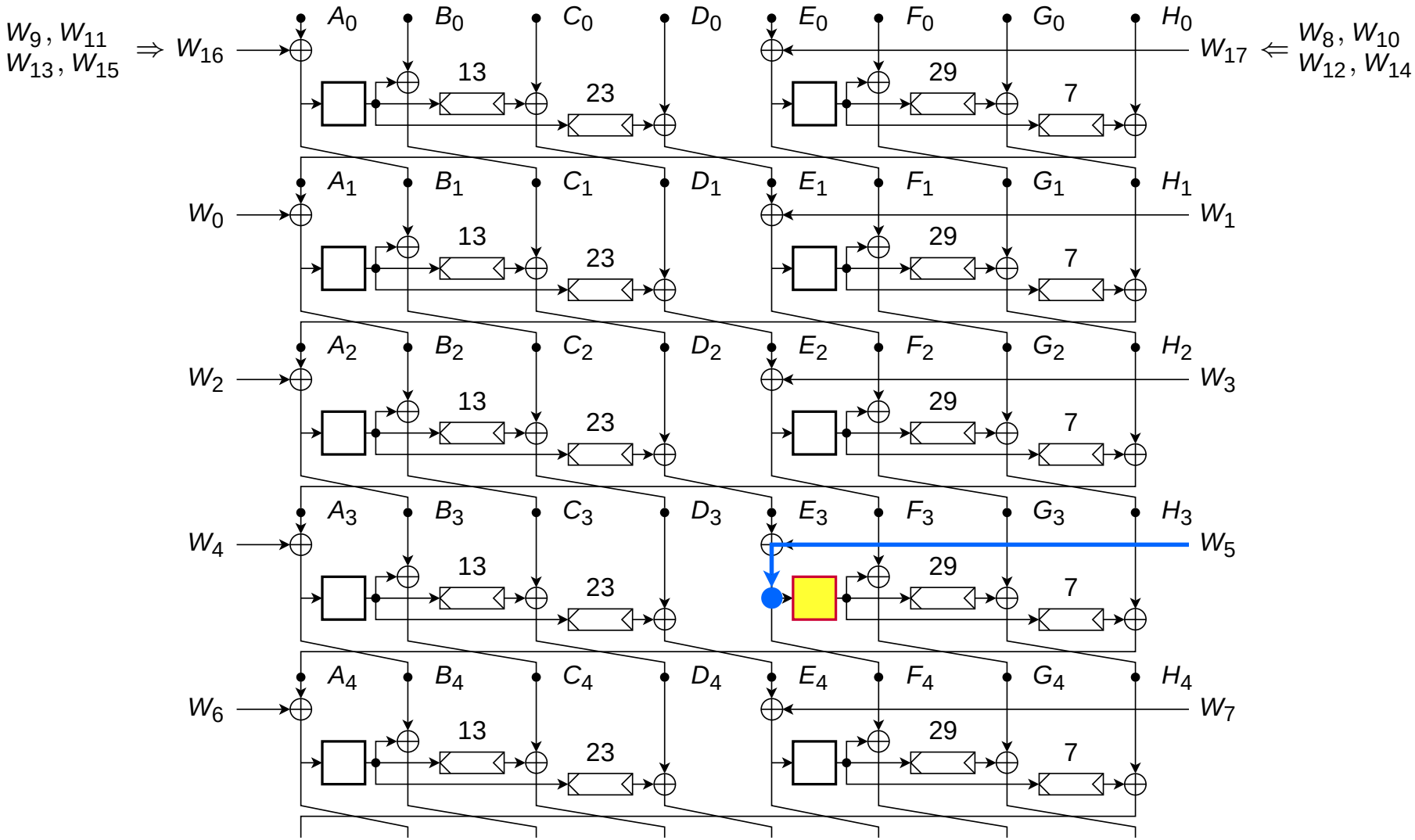
Satisfying conditions

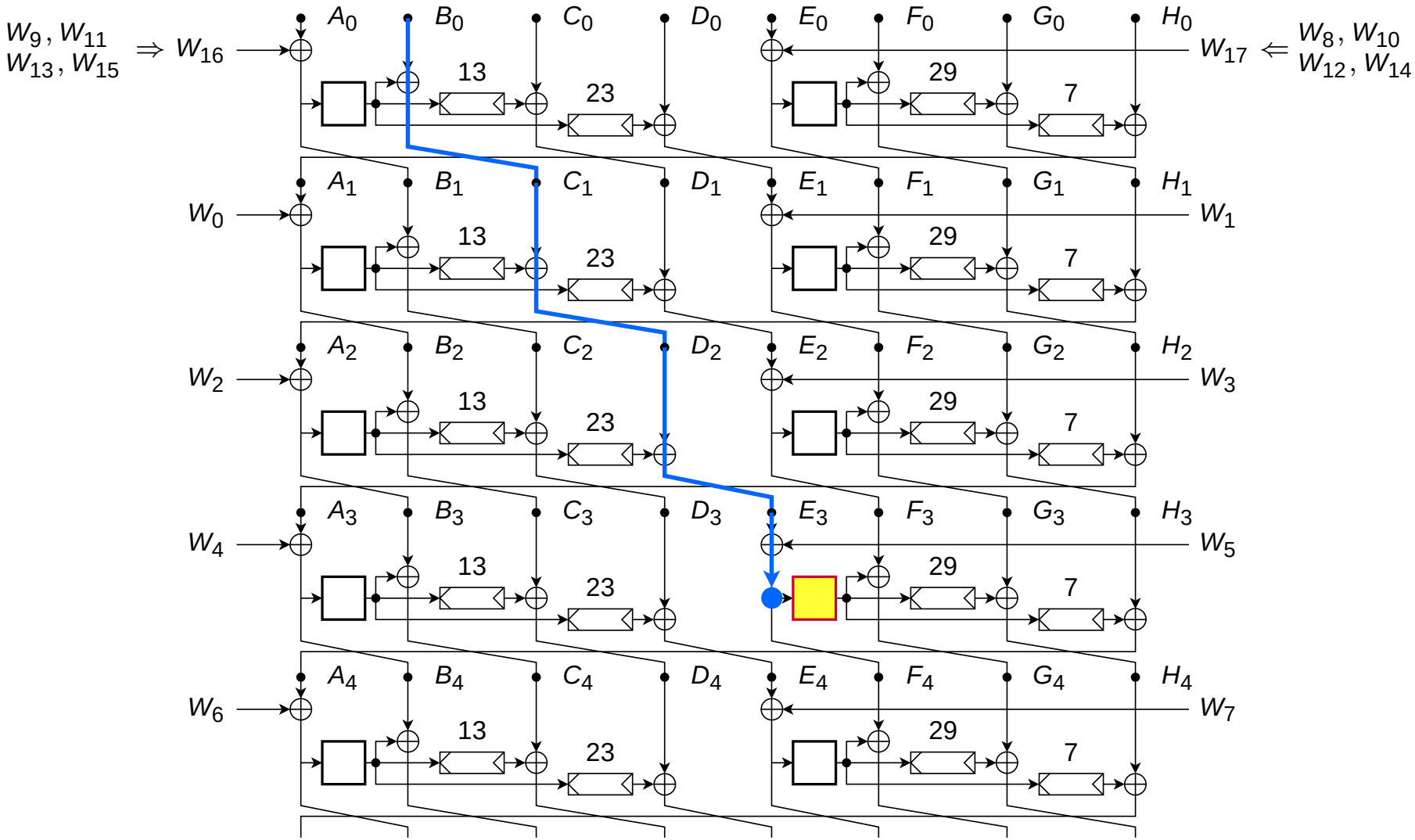
To eliminate probabilistic behaviour, we want to set inputs of active G_{256} to "good" values.

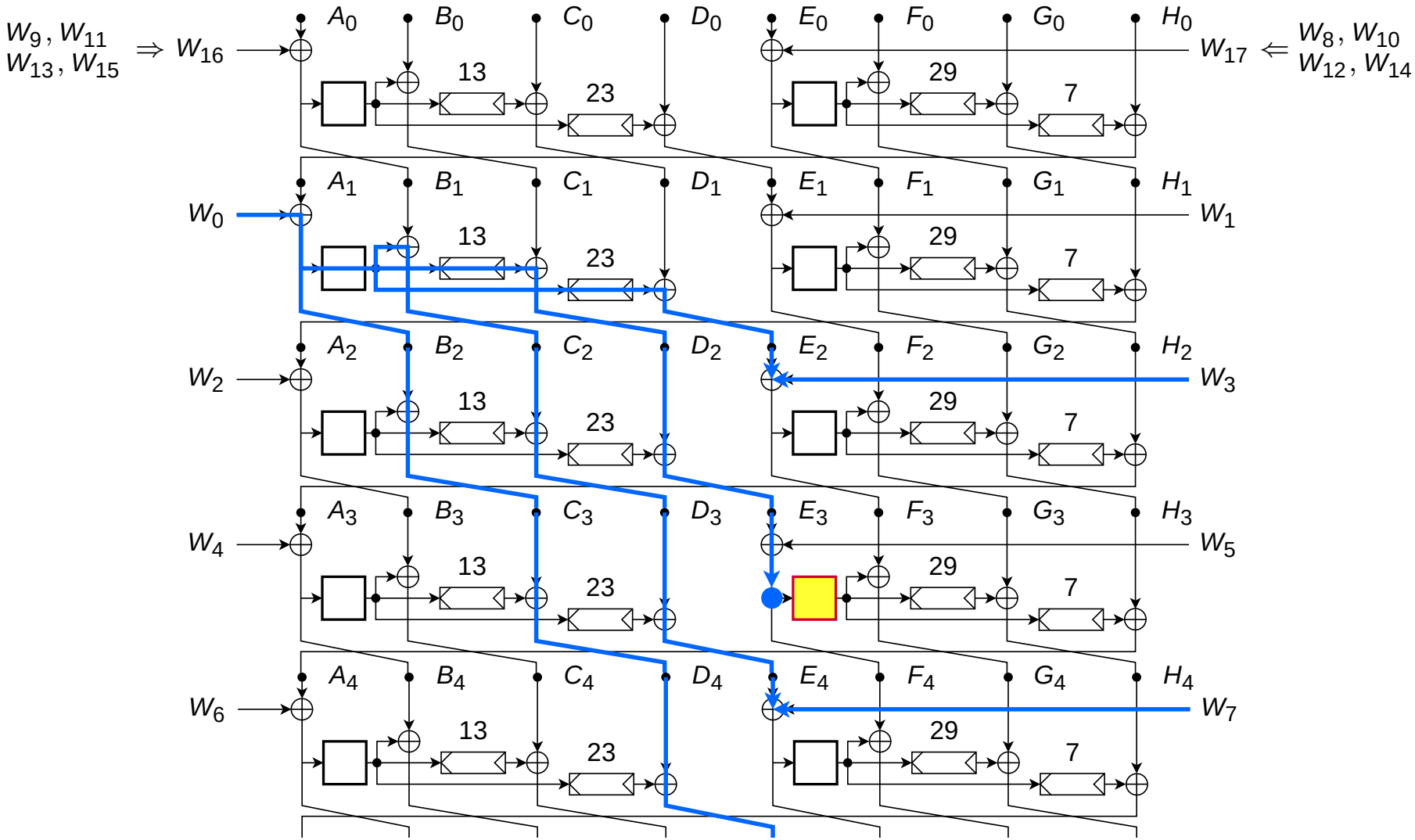
- We have full control over words $W_0, \ldots, W_{15}$
- Through linear combinations, we have some control over words $W_{16}, \ldots, W_{31}$
- For semi-free-start collisions and pseudo-collisions, we additionally have control over initial values $IV_0, \ldots, IV_7$


W_9, W_{11}
W_{13}, W_{15}
$\Rightarrow W_{16}$
A_0 B_0 C_0 D_0 E_0 F_0 G_0 H_0
$W_{17} \Leftarrow$
W_8, W_{10}
W_{12}, W_{14}
13 23 29 7
W_0
A_1 B_1 C_1 D_1 E_1 F_1 G_1 H_1
W_1
W_2
A_2 B_2 C_2 D_2 E_2 F_2 G_2 H_2
W_3
W_4
A_3 B_3 C_3 D_3 E_3 F_3 G_3 H_3
W_5
W_6
A_4 B_4 C_4 D_4 E_4 F_4 G_4 H_4
W_7


W_9, W_{11}
W_{13}, W_{15}
$\Rightarrow W_{16}$
A_0 B_0 C_0 D_0 E_0 F_0 G_0 H_0
$W_{17} \Leftarrow$
W_8, W_{10}
W_{12}, W_{14}
13 23 29 7
W_0
A_1 B_1 C_1 D_1 E_1 F_1 G_1 H_1
W_1
W_2
A_2 B_2 C_2 D_2 E_2 F_2 G_2 H_2
W_3
W_4
A_3 B_3 C_3 D_3 E_3 F_3 G_3 H_3
W_5
W_6
A_4 B_4 C_4 D_4 E_4 F_4 G_4 H_4
W_7


W_9, W_{11}
W_{13}, W_{15}
$\Rightarrow W_{16}$
W_8, W_{10}
W_{12}, W_{14}
$\Leftarrow W_{17}$
A_0 B_0 C_0 D_0 E_0 F_0 G_0 H_0
A_1 B_1 C_1 D_1 E_1 F_1 G_1 H_1
A_2 B_2 C_2 D_2 E_2 F_2 G_2 H_2
A_3 B_3 C_3 D_3 E_3 F_3 G_3 H_3
A_4 B_4 C_4 D_4 E_4 F_4 G_4 H_4
13
23
29
7
W_0
W_1
W_2
W_3
W_4
W_5
W_6
W_7

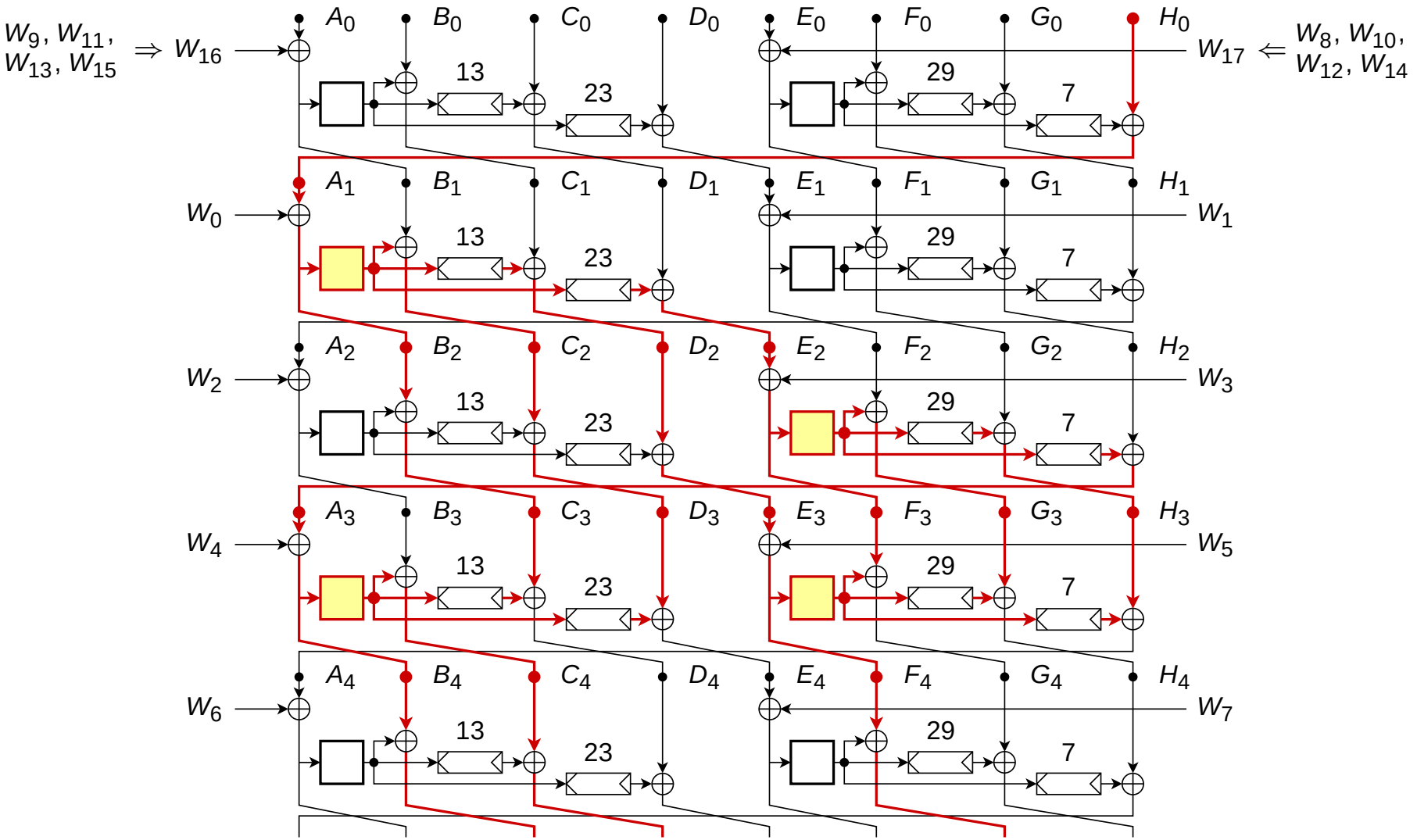

W_9, W_{11}
W_{13}, W_{15}
$\Rightarrow W_{16}$
W_8, W_{10}
W_{12}, W_{14}
$\Leftarrow W_{17}$
A_0 B_0 C_0 D_0 E_0 F_0 G_0 H_0
A_1 B_1 C_1 D_1 E_1 F_1 G_1 H_1
A_2 B_2 C_2 D_2 E_2 F_2 G_2 H_2
A_3 B_3 C_3 D_3 E_3 F_3 G_3 H_3
A_4 B_4 C_4 D_4 E_4 F_4 G_4 H_4
13
23
29
7
W_0
W_1
W_2
W_3
W_4
W_5
W_6
W_7


W_9, W_{11}
W_{13}, W_{15}
$\Rightarrow W_{16}$
W_8, W_{10}
W_{12}, W_{14}
$\Leftarrow W_{17}$
A_0 B_0 C_0 D_0 E_0 F_0 G_0 H_0
A_1 B_1 C_1 D_1 E_1 F_1 G_1 H_1
A_2 B_2 C_2 D_2 E_2 F_2 G_2 H_2
A_3 B_3 C_3 D_3 E_3 F_3 G_3 H_3
A_4 B_4 C_4 D_4 E_4 F_4 G_4 H_4
13
23
29
7
W_0
W_1
W_2
W_3
W_4
W_5
W_6
W_7

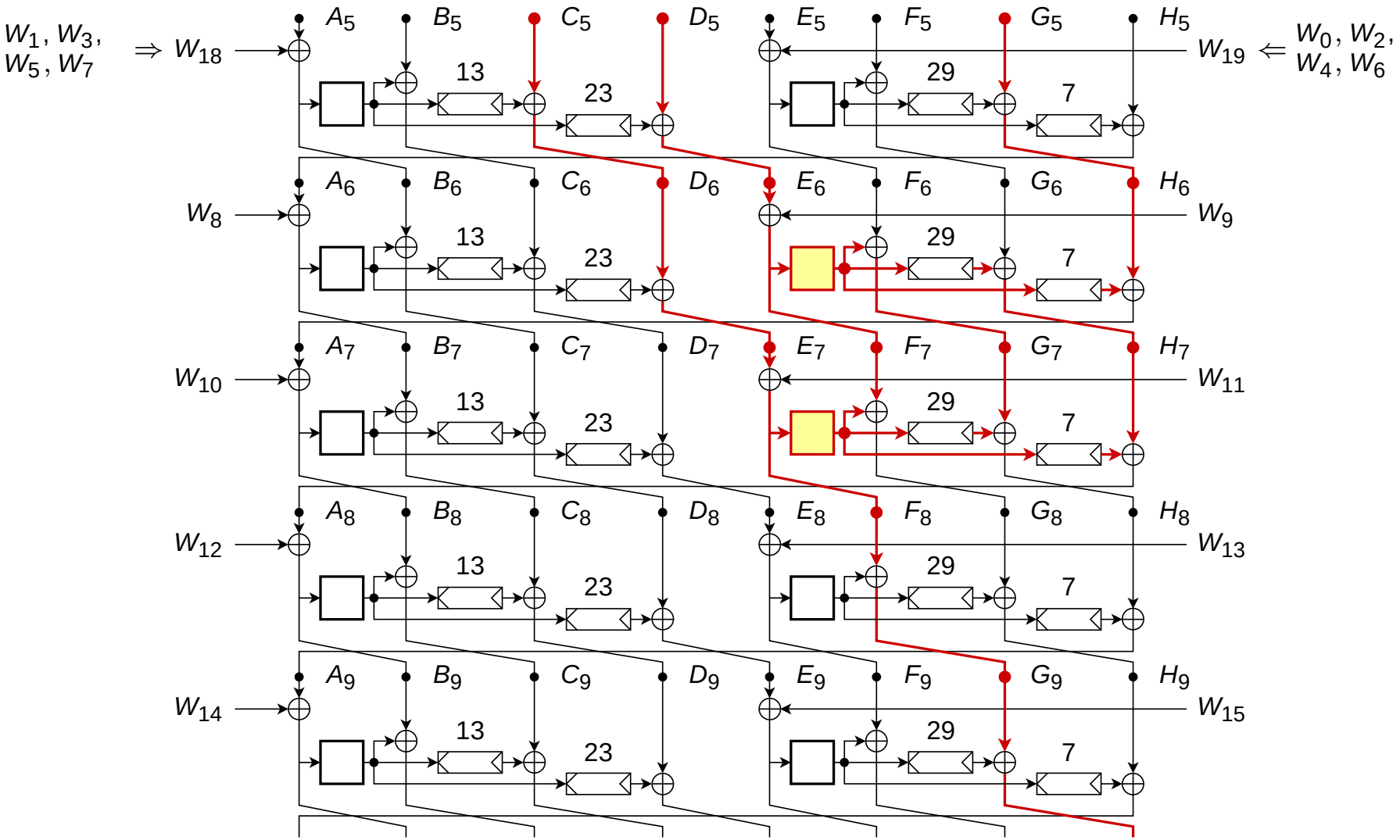
Satisfying conditions

- If we can use initial values, conditions in steps 1–4 are always possible
- Depending on the number of active G, usually we can correct around 16–18 steps
- Might be possible to correct 20 steps in some cases

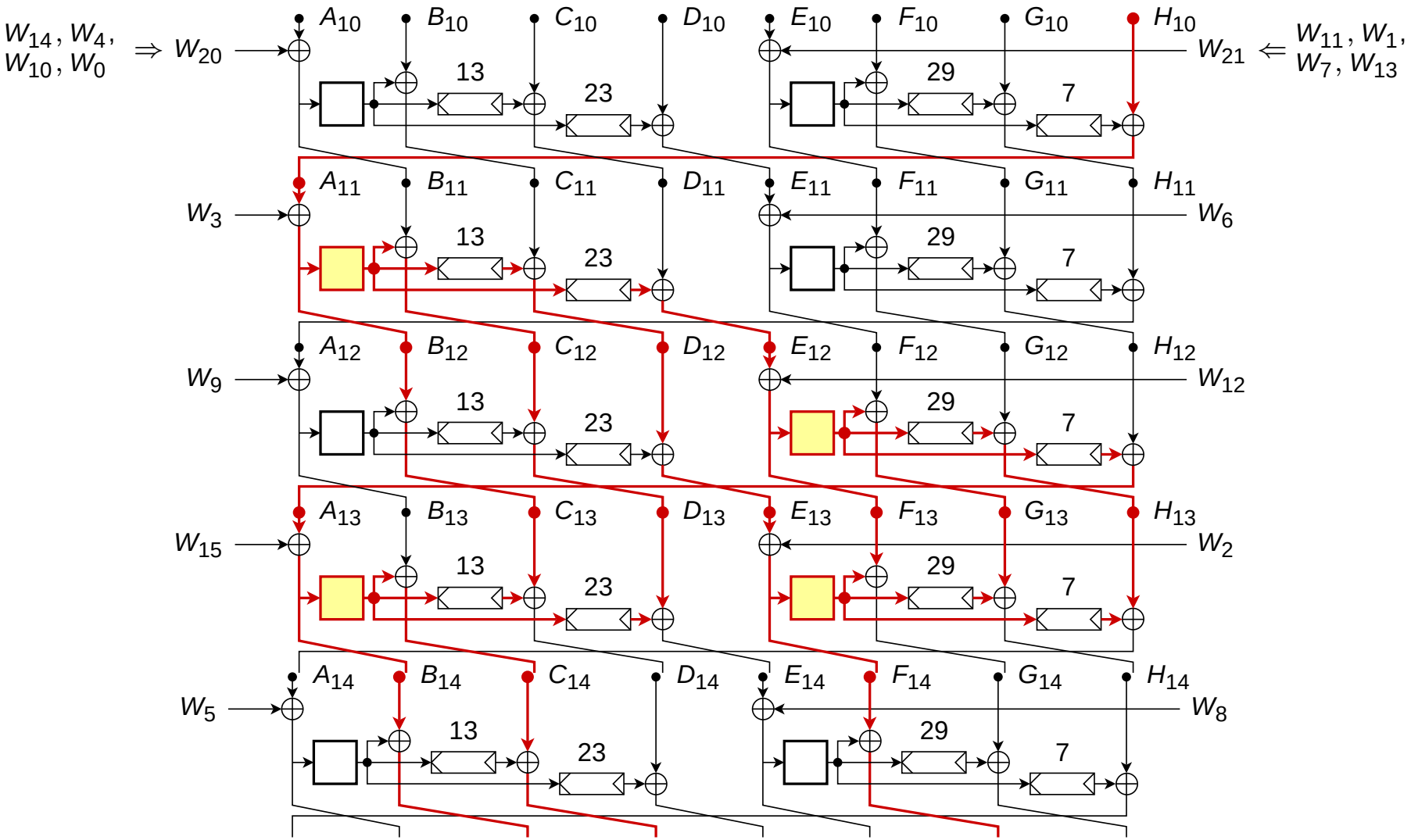
Pseudo-collision path: steps 1 – 5



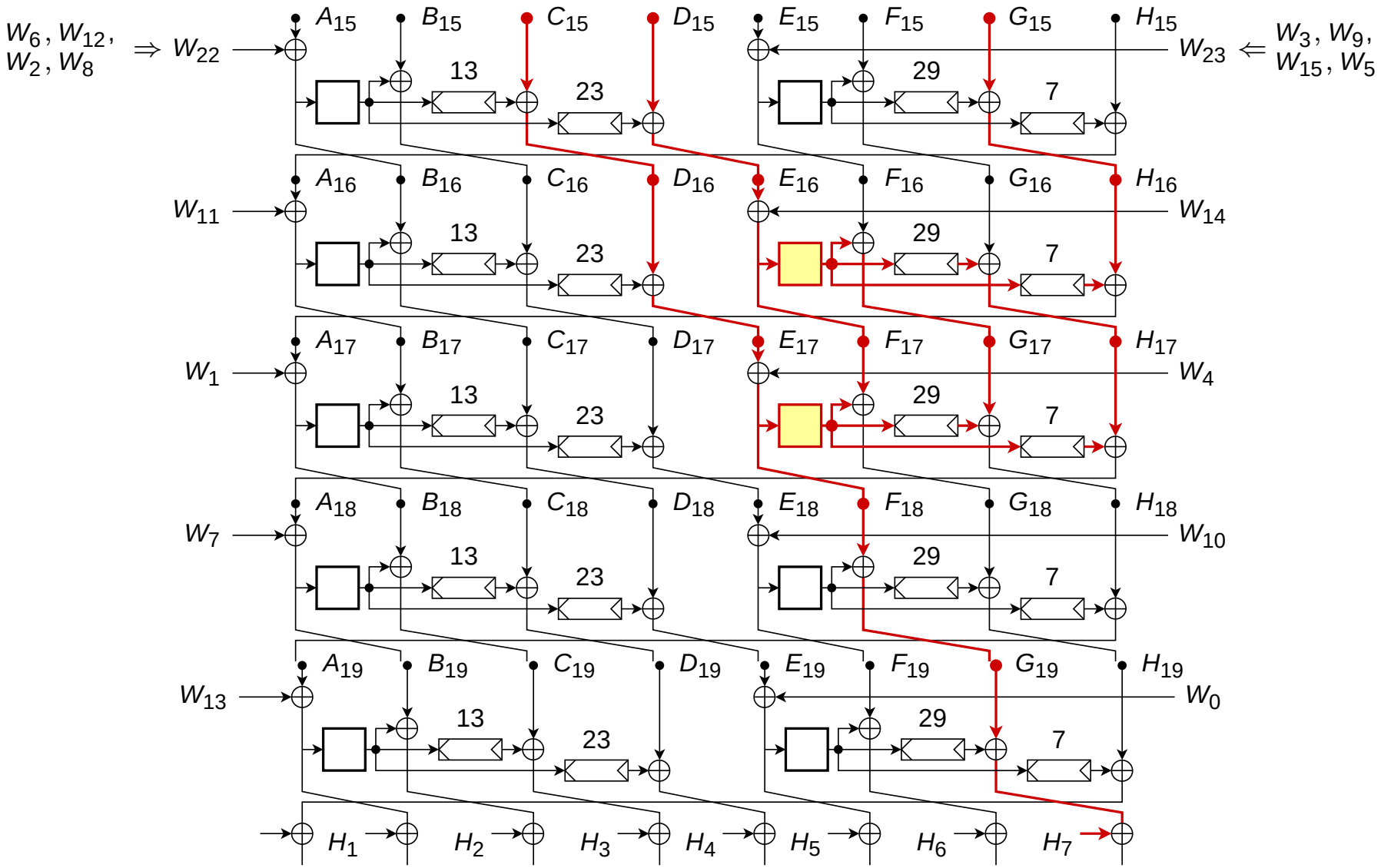
Pseudo-collision path: steps 6 – 10



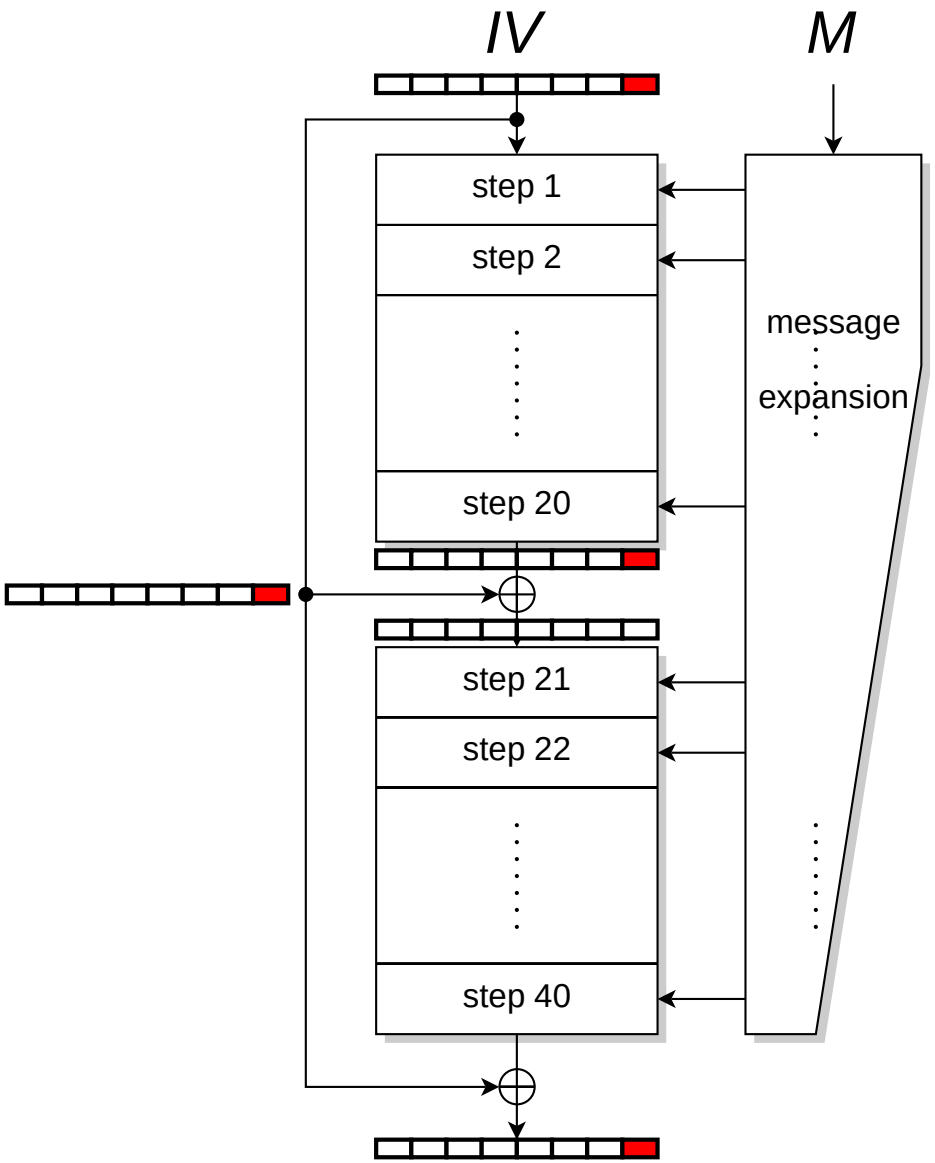
Pseudo-collision path: steps 11 – 15



Pseudo-collision path: steps 16 – 20



Pseudo-collisions for ARIRANG-224/384



- single message block
- can use 14 message words, last two for padding
- message corrections: 12 active G_{256} in steps 2–18, complexity $\approx 2^{23}$
- register H discarded for ARIRANG-224/384
- pseudo-collision for the complete hash function

Summary of results

Compression function		
Result	Complexity	Example
32-bit near-collision for full ARIRANG-256 compress	1	Y
64-bit near-collision for full ARIRANG-512 compress	1	Y
26-step (out of 40) collision for ARIRANG-256/512	1	Y
Hash function		
Result	Complexity	Example
pseudo-collision for full ARIRANG-224/384 hash	2^{23} / 1	Y

Conclusions

A brief introduction on linearization and message modification techniques have been introduced, with example of applications to ARIRANG.

Conclusions

A brief introduction on linearization and message modification techniques have been introduced, with example of applications to ARIRANG.

Thanks for your attention!