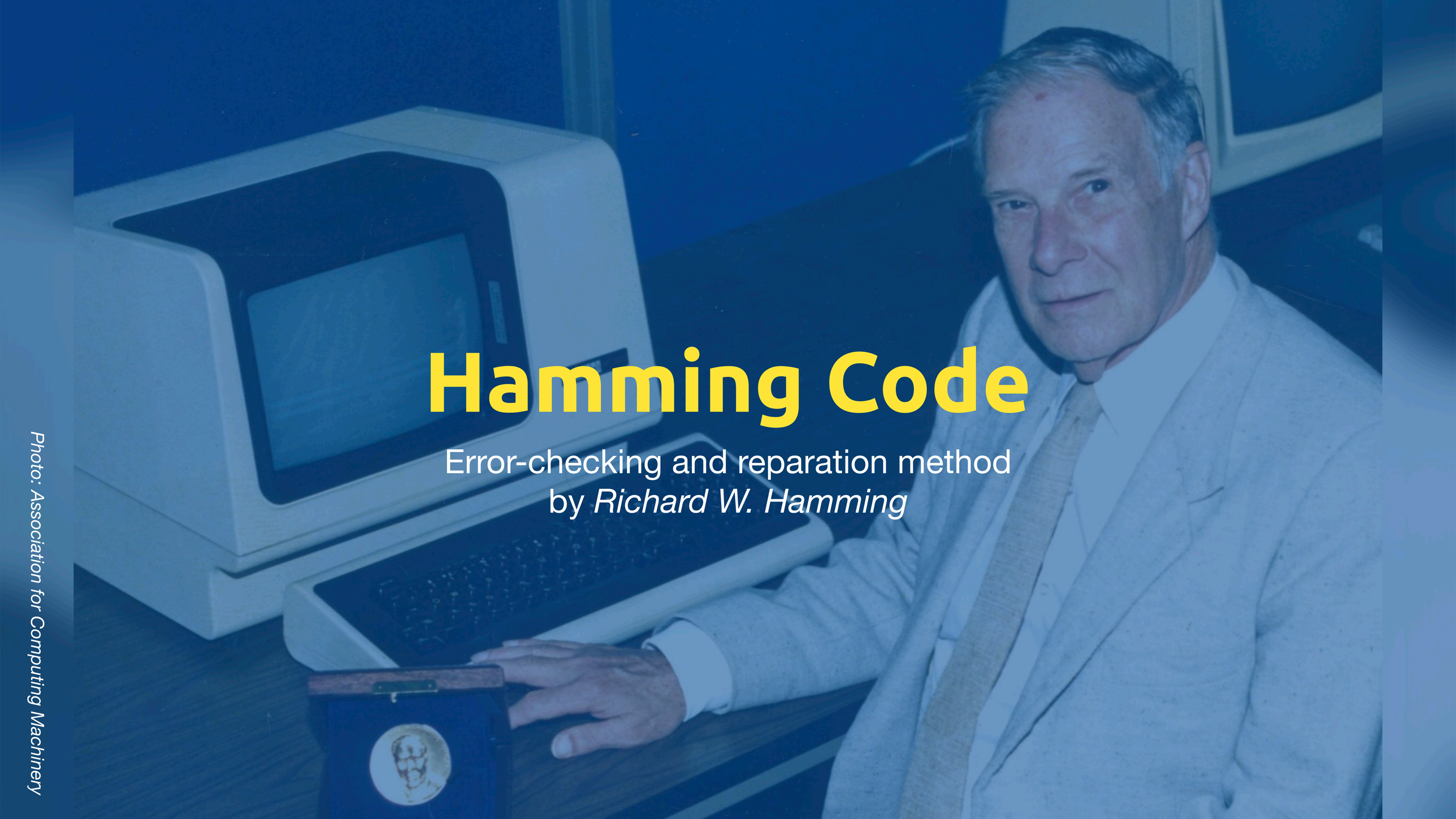
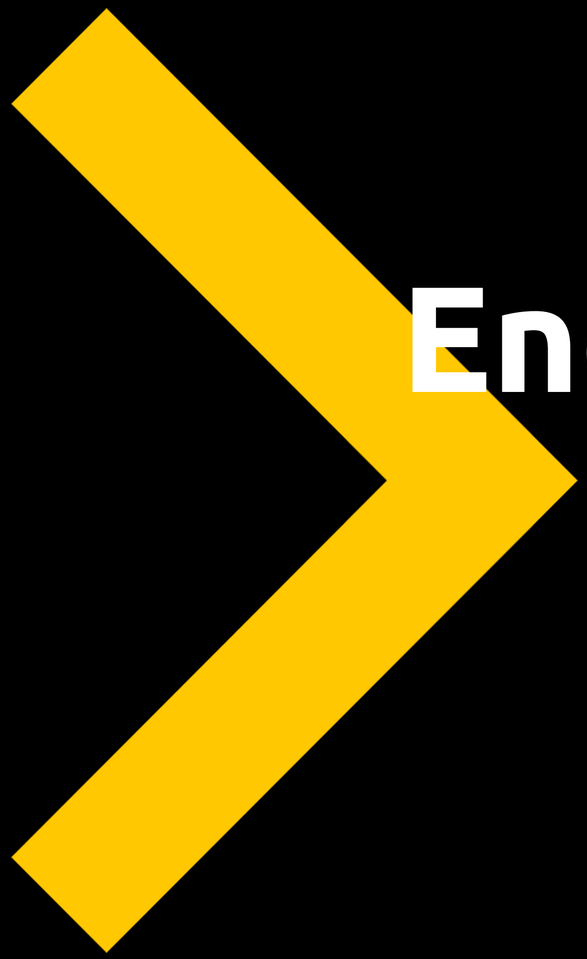


Hamming Code

Error-checking and reparation method
by *Richard W. Hamming*

Photo: Association for Computing Machinery



Encoding

Given that our data is

1 0 1 0 0 1 0 1

$2^3 = 8$ bits

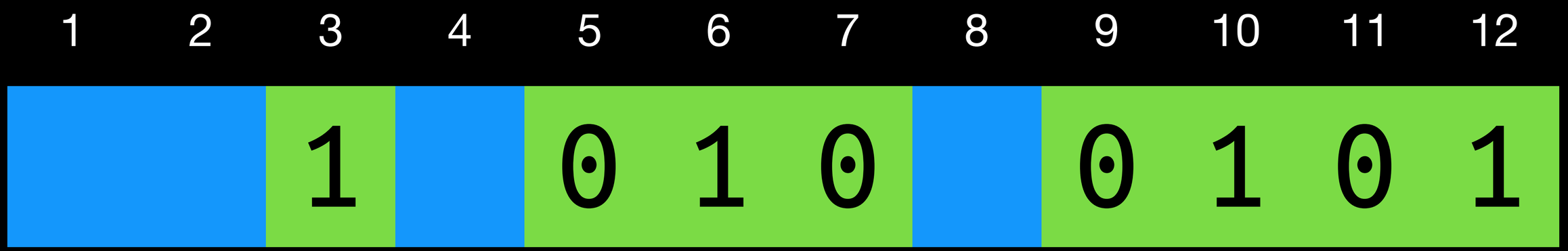
Number of parity bits:

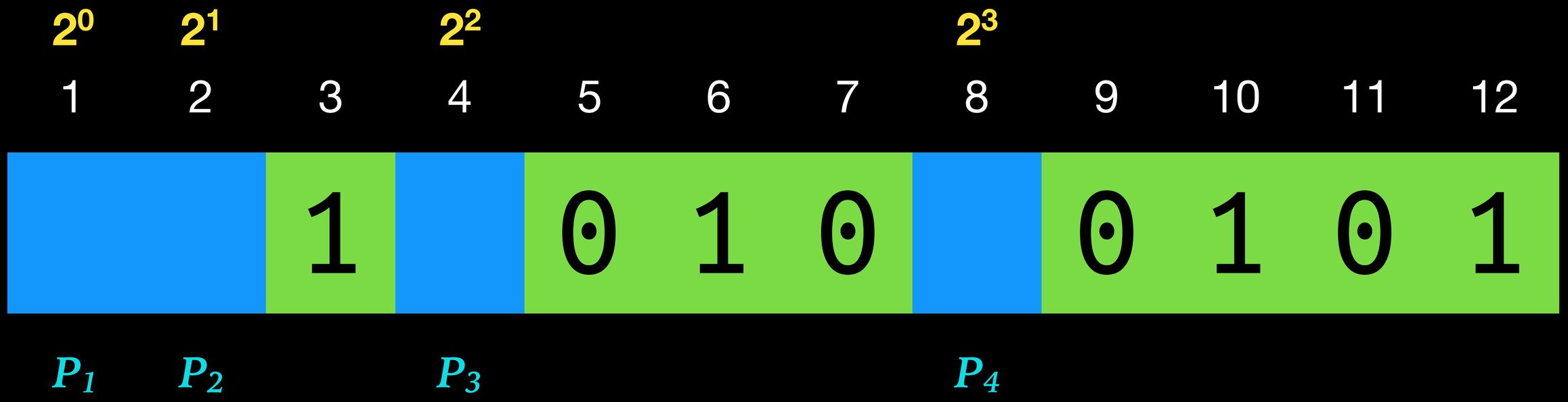
3 + 1

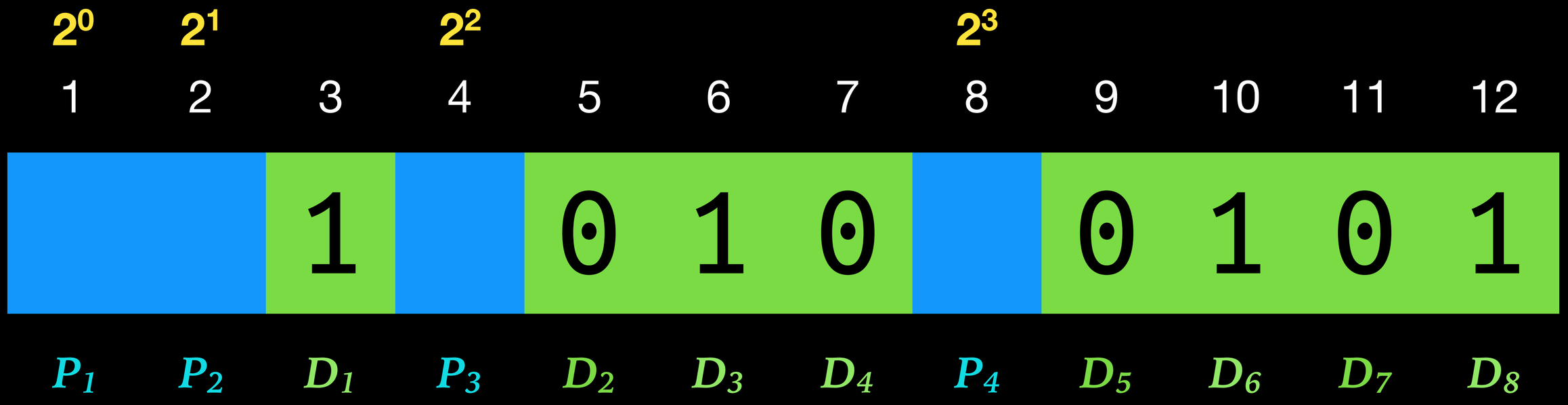
Number of parity bits:

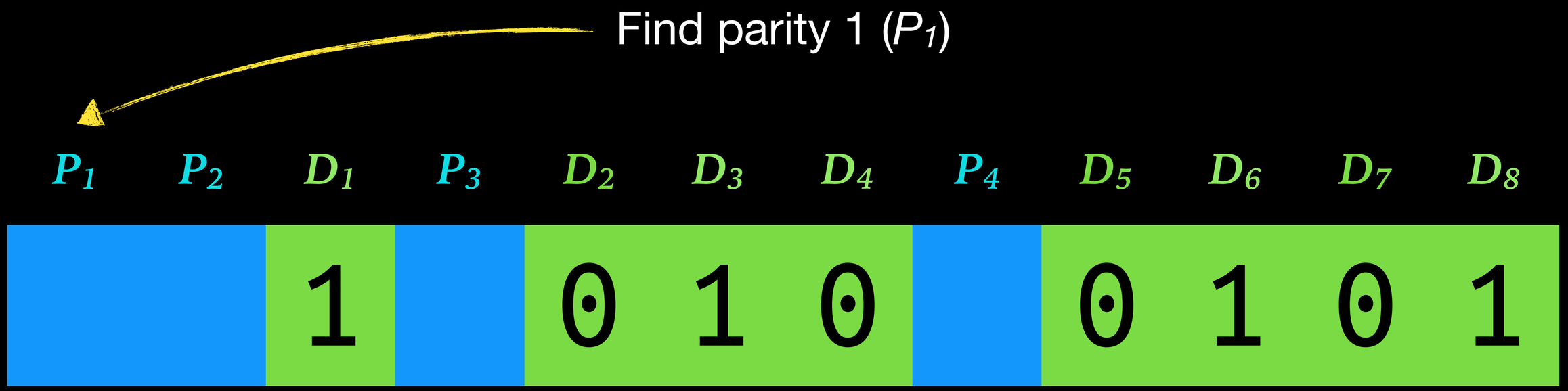
Arranging bits

1 0 1 0 0 1 0 1


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

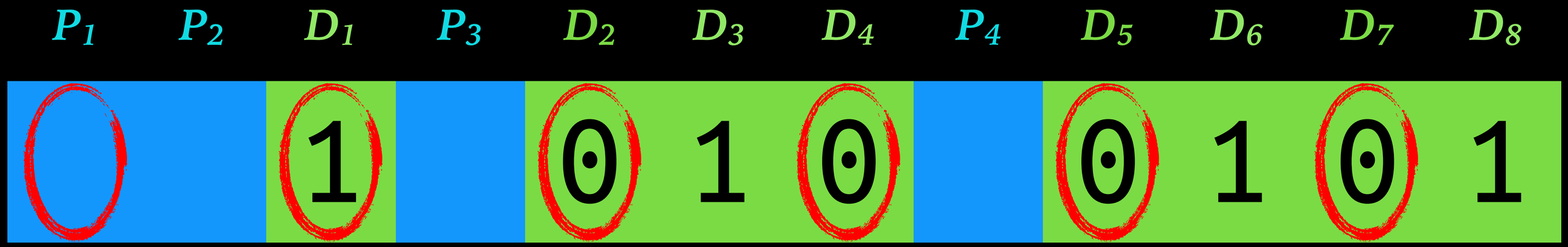

2^0
2^1
2^2
2^3
1 2 3 4 5 6 7 8 9 10 11 12
1 0 1 0 0 1 0 1
P_1
P_2
P_3
P_4


2^0
2^1
2^2
2^3
1 2 3 4 5 6 7 8 9 10 11 12
1 0 1 0 0 1 0 1
P_1 P_2 D_1 P_3 D_2 D_3 D_4 P_4 D_5 D_6 D_7 D_8



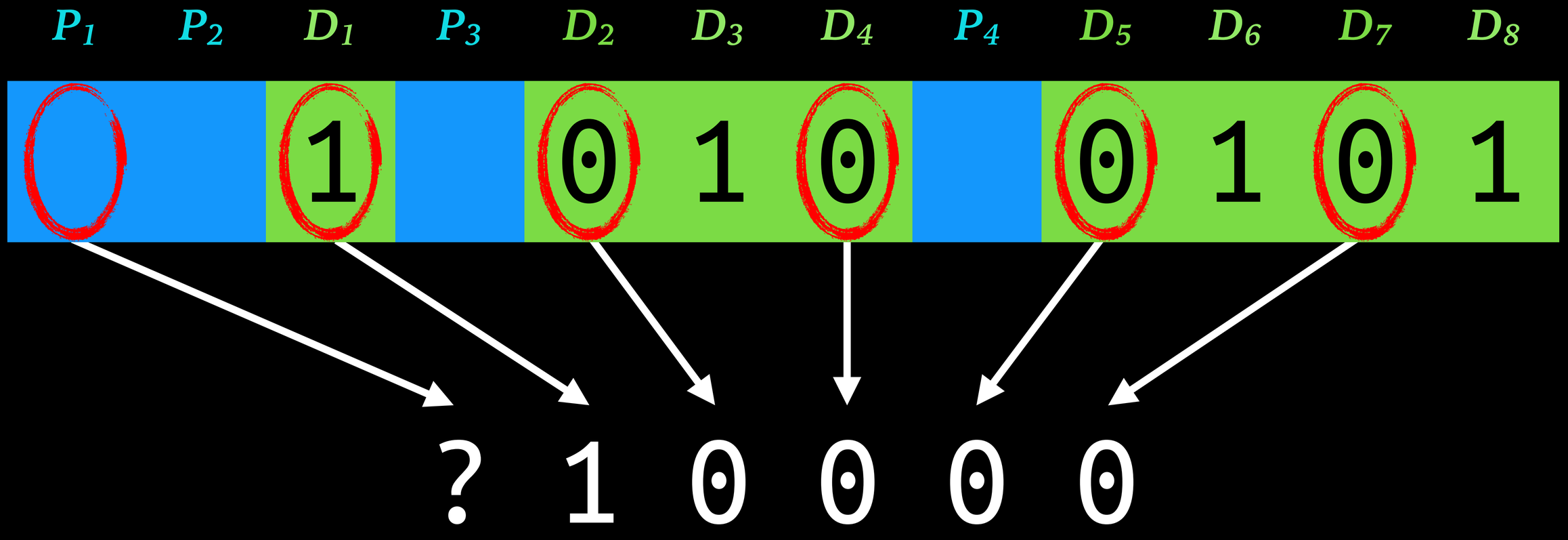
Pick 1, skip 1
(because the bit we are finding is at position 1)

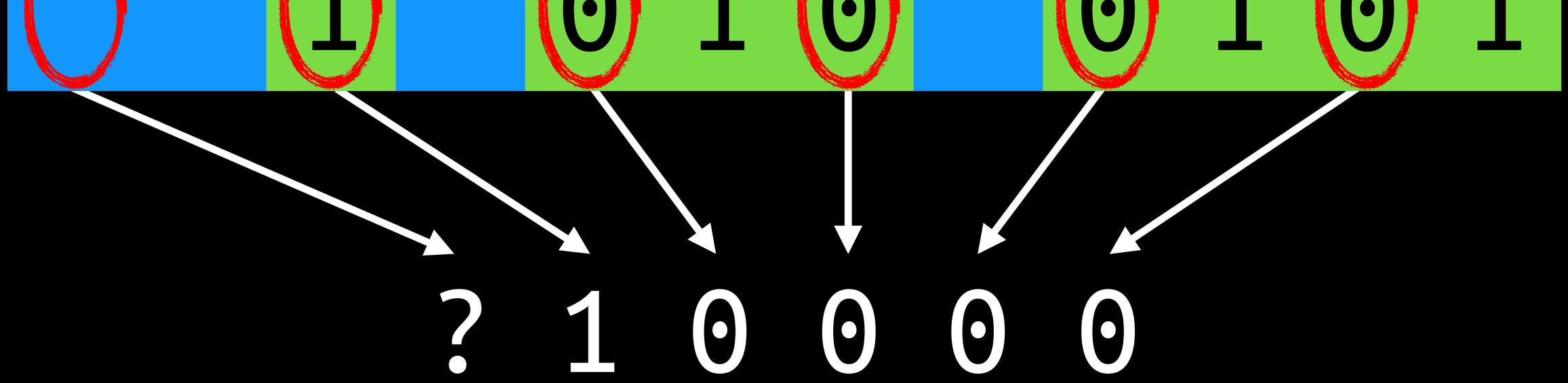
Find parity 1 (P_1)



Pick 1, skip 1
(because the bit we are finding is at position 1)

Find parity 1 (P_1)



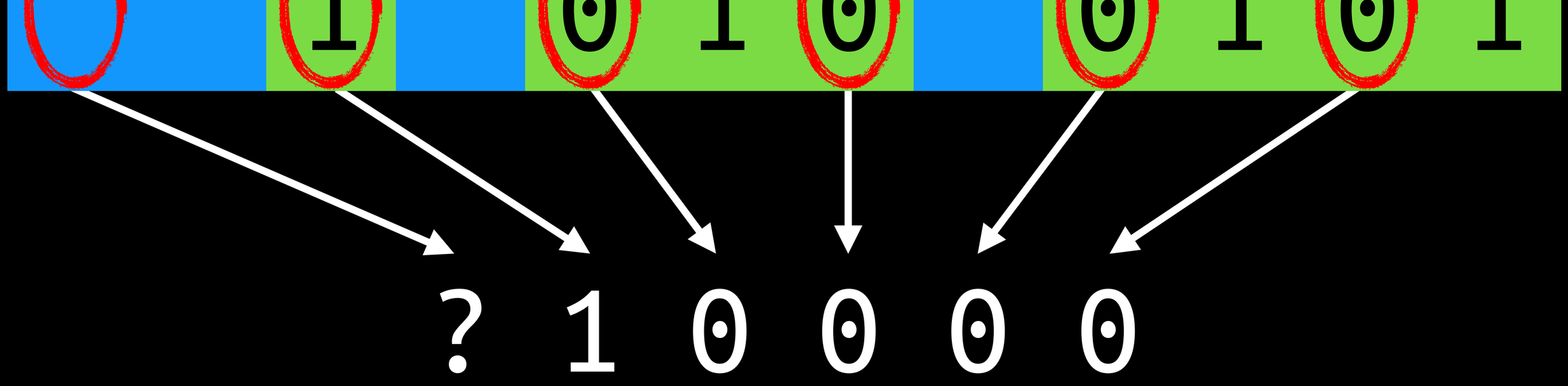


Count 1's

If the amount of 1's is odd, parity bit equals 1.

If the amount of 1's is even, parity bit equals 0.

In other word, make the amount of 1's even.

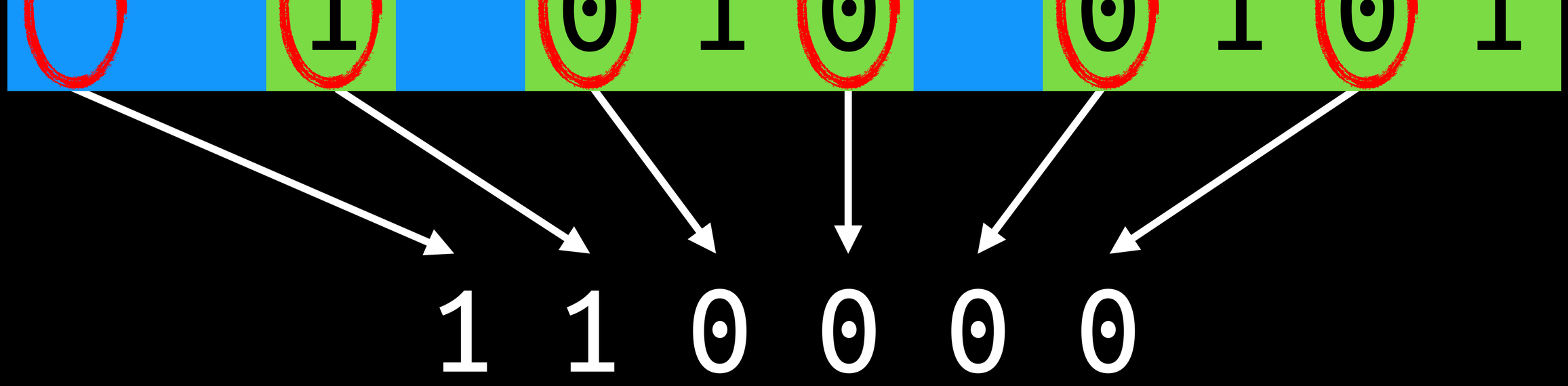


Count 1’s

If the amount of 1’s is odd, parity bit equals 1.

If the amount of 1’s is even, parity bit equals 0.

In other word, make the amount of 1’s even.



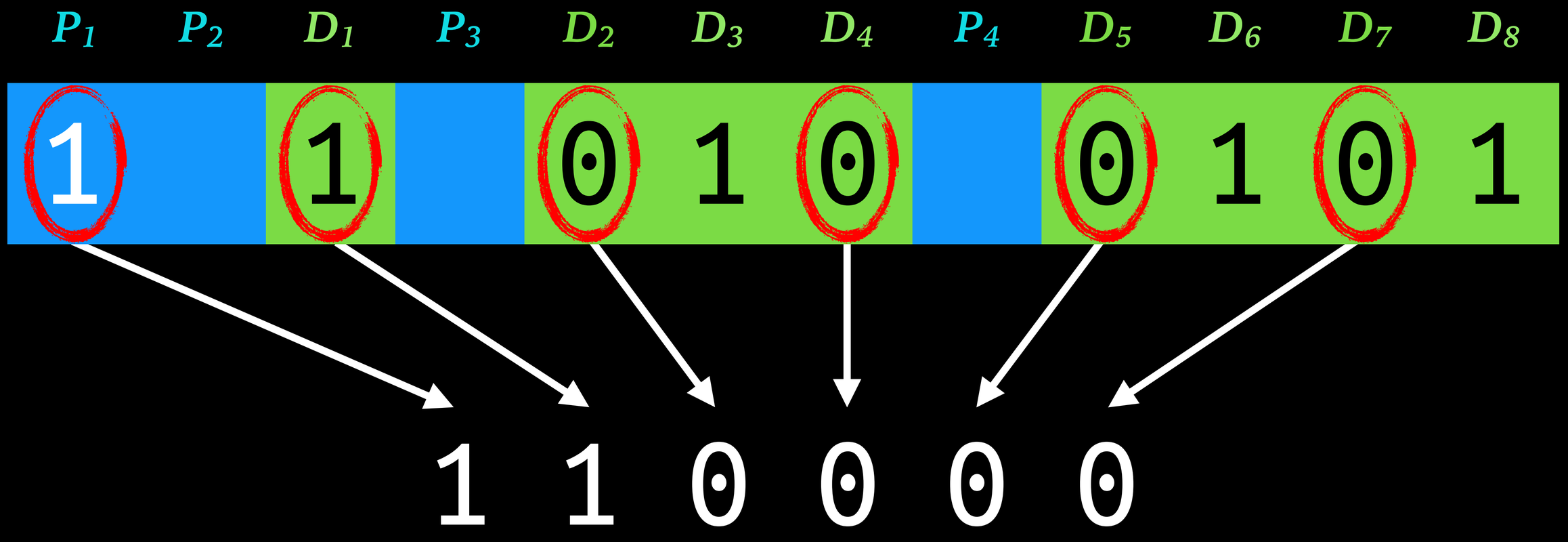
Count 1’s

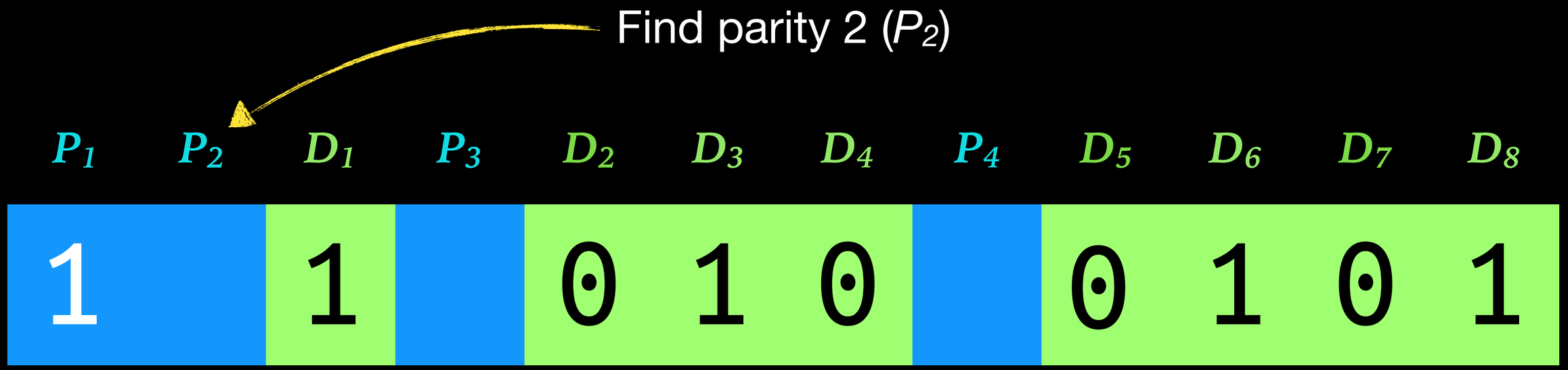
If the amount of 1’s is odd, parity bit equals 1.

If the amount of 1’s is even, parity bit equals 0.

In other word, make the amount of 1’s even.

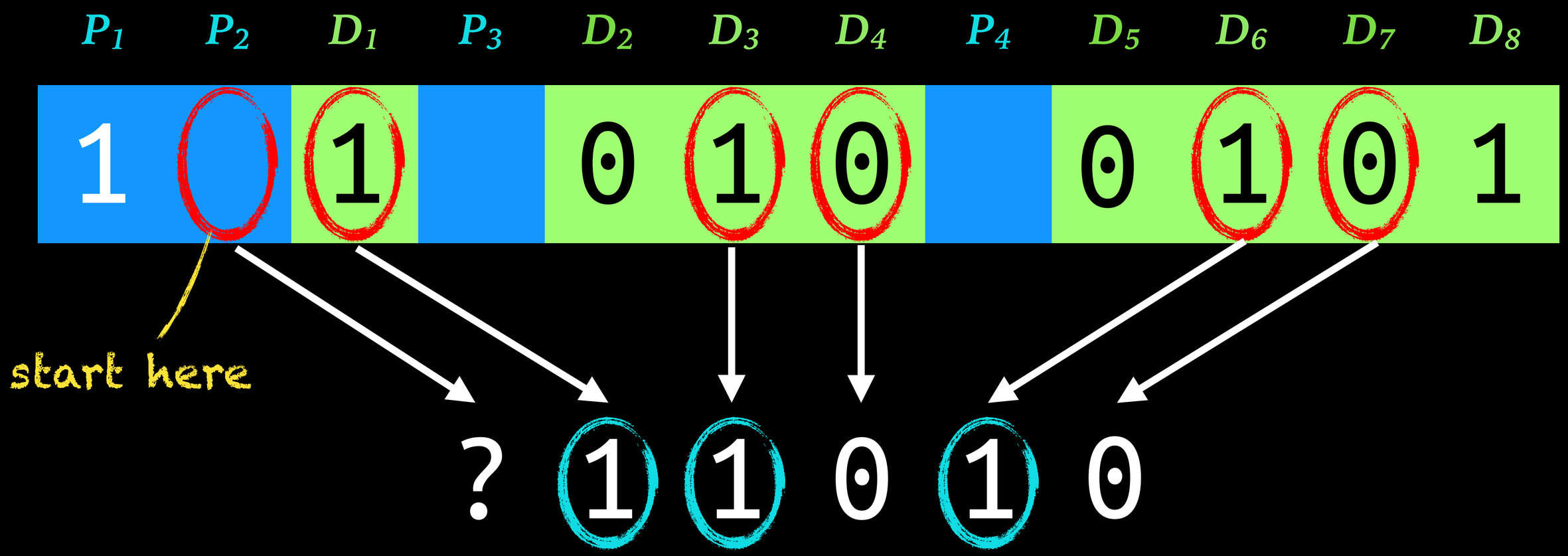
Find parity 1 (P_1)





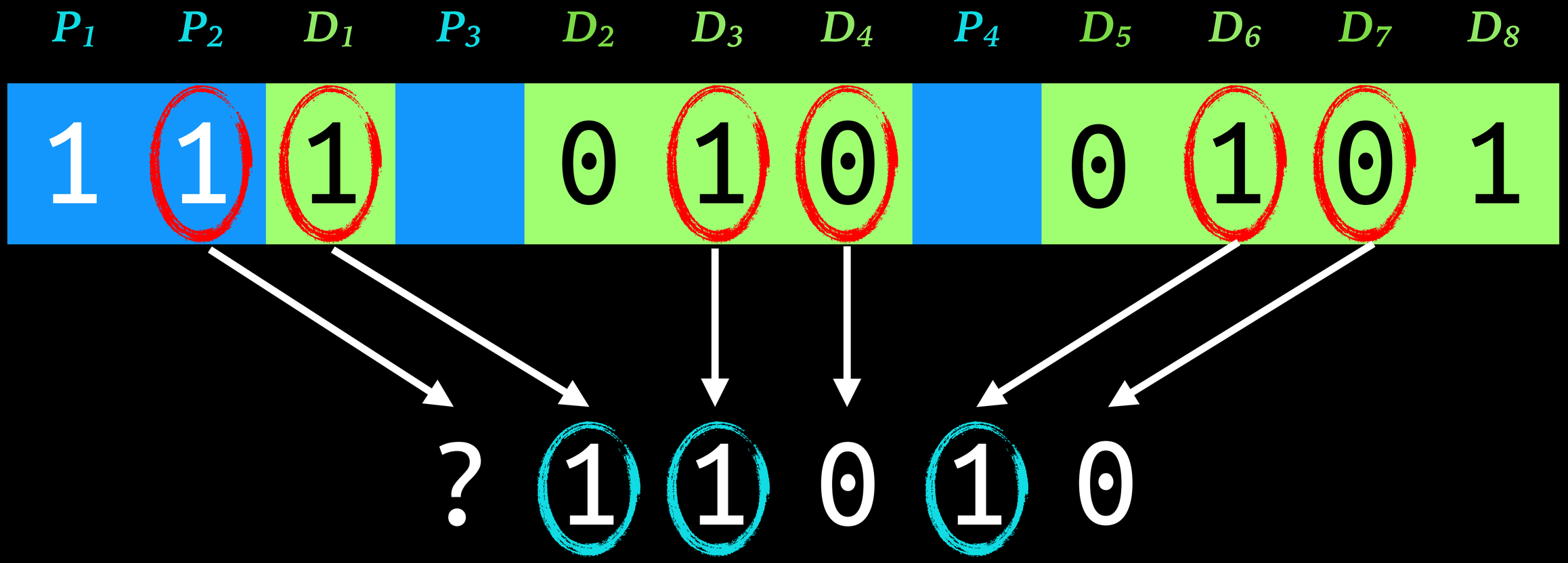
Pick 2, skip 2
(because the bit we are finding is at position 2)

Find parity 2 (P_2)

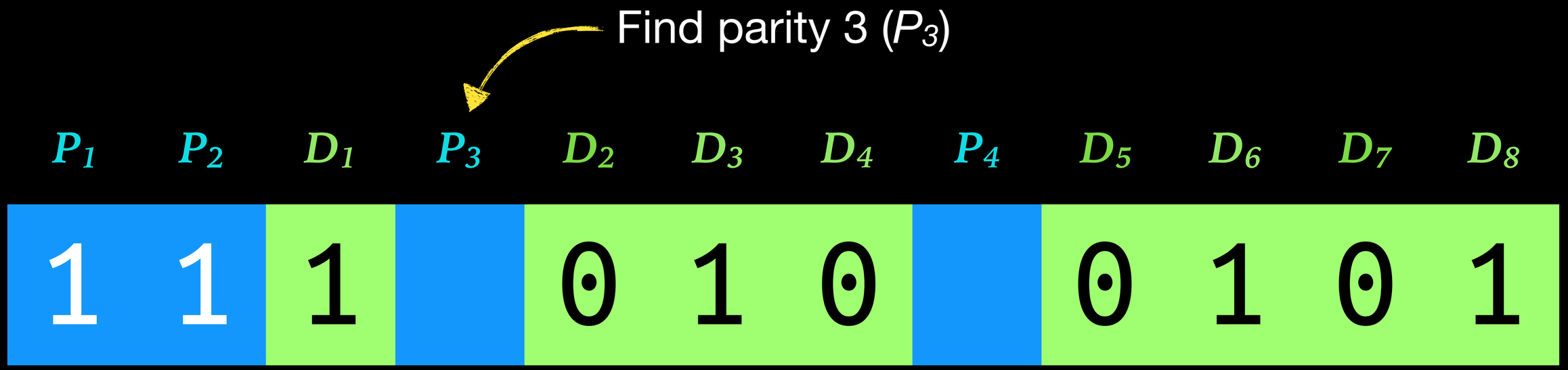


Pick 2, skip 2
(because the bit we are finding is at position 2)

Find parity 2 (P_2)

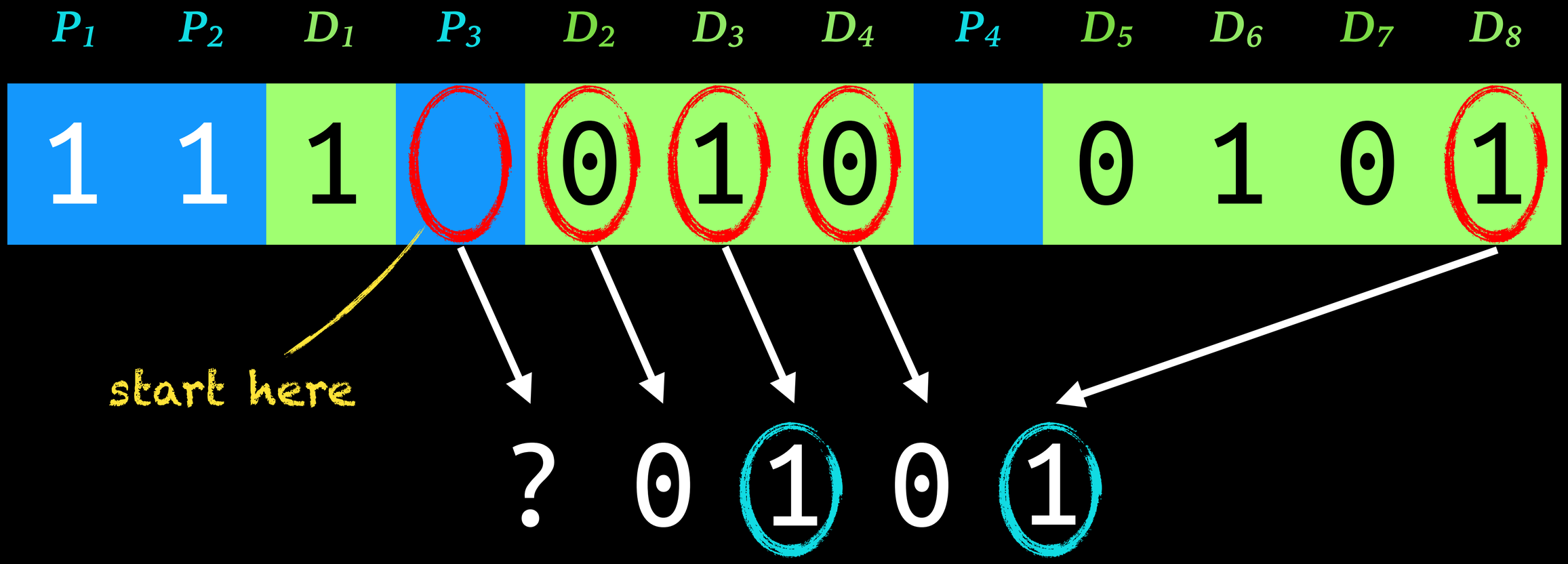


Pick 2, skip 2
(because the bit we are finding is at position 2)



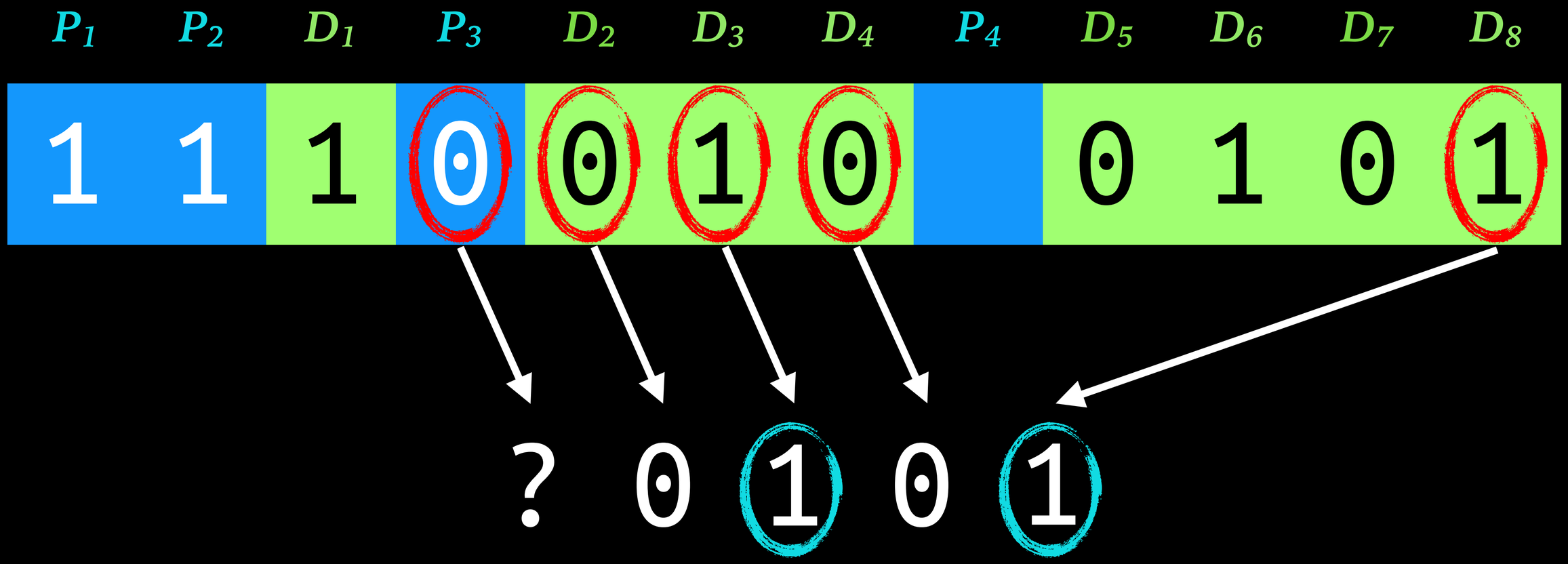
Pick 4, skip 4
(because the bit we are finding is at position 4)

Find parity 3 (P_3)



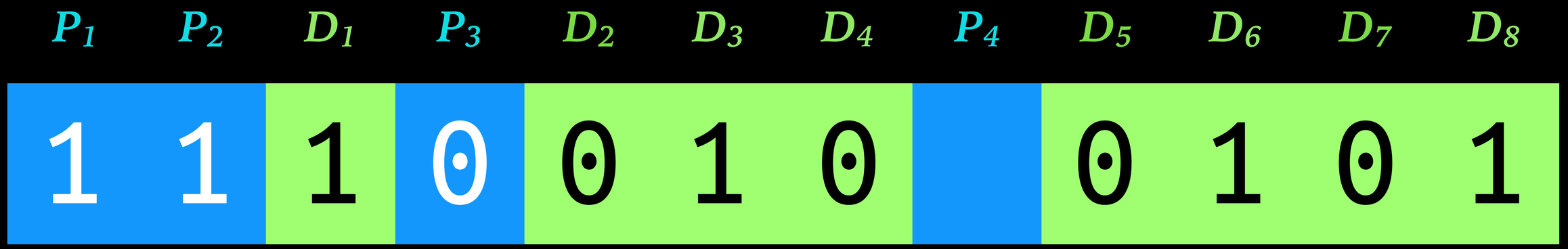
Pick 4, skip 4
(because the bit we are finding is at position 4)

Find parity 3 (P_3)



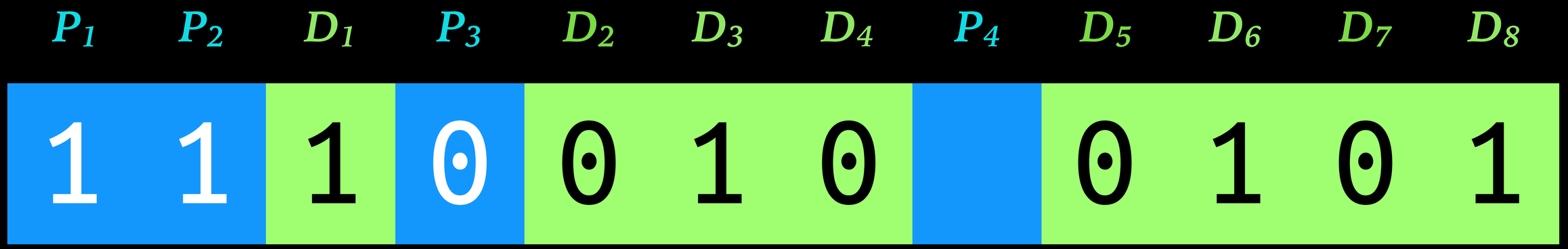
Pick 4, skip 4
(because the bit we are finding is at position 4)

Find parity 4 (P_4)



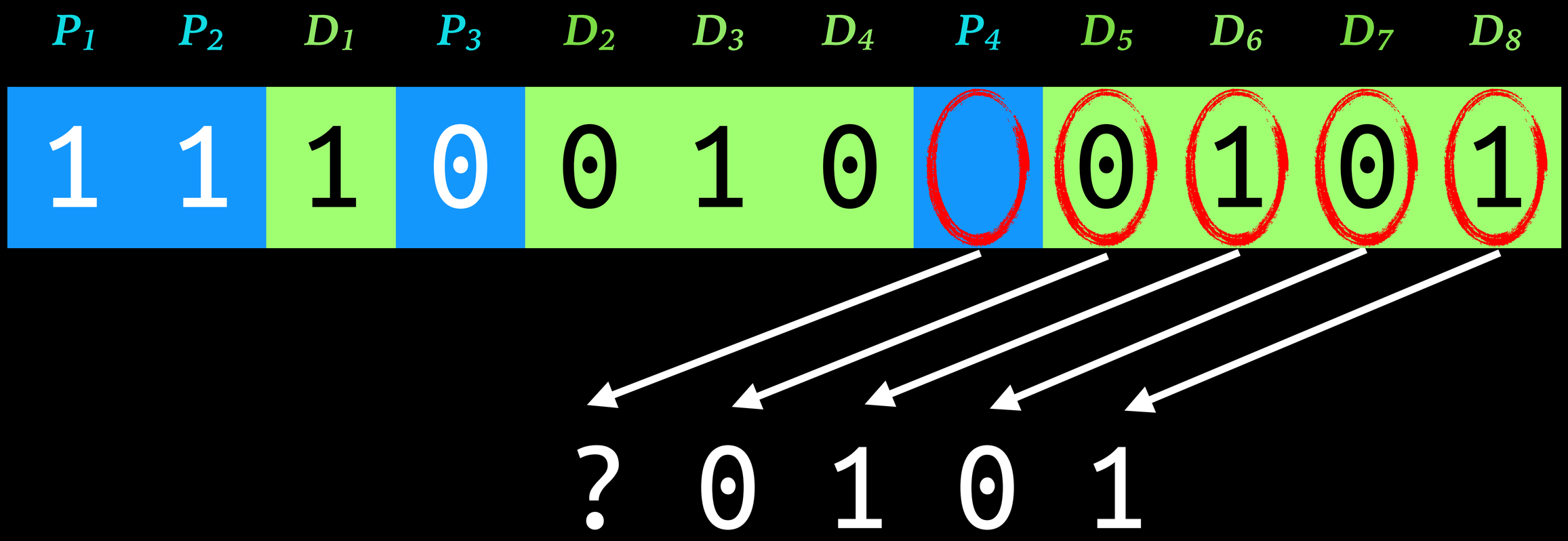
Pick 8, skip 8
(because the bit we are finding is at position 8)

Find parity 4 (P_4)



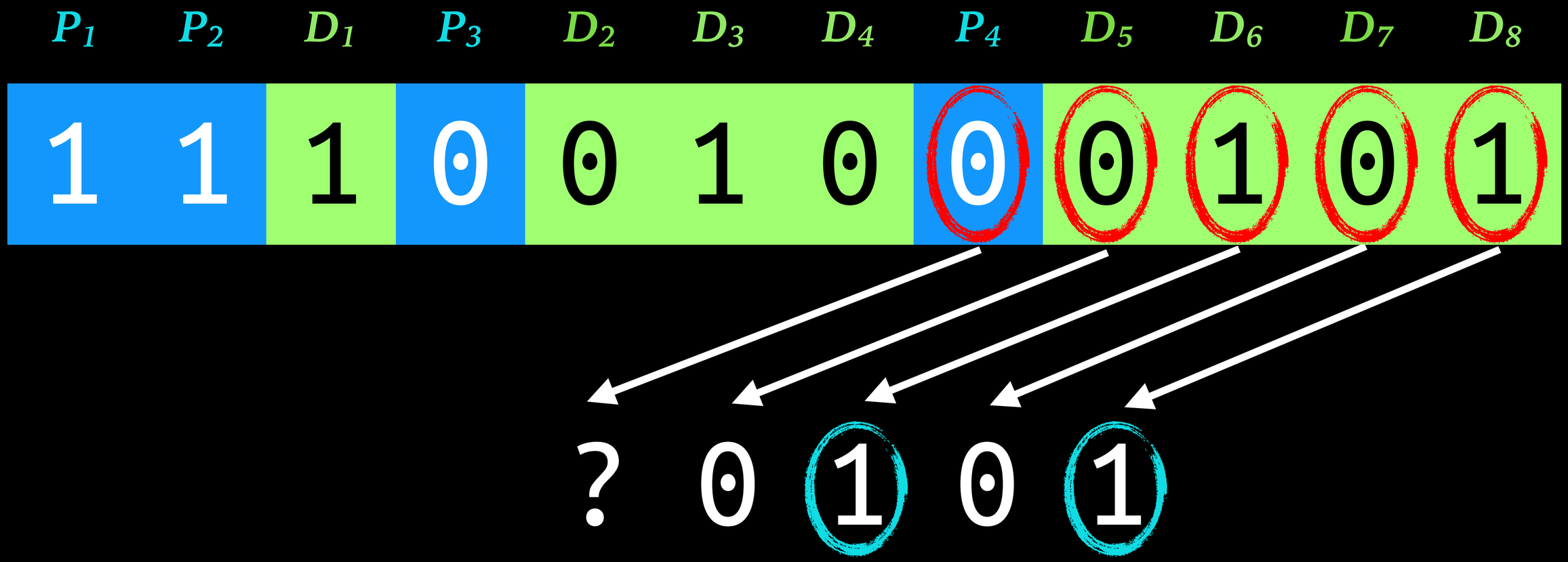
Pick 8, skip 8
(because the bit we are finding is at position 8)

Find parity 4 (P_4)



Pick 8, skip 8
(because the bit we are finding is at position 8)

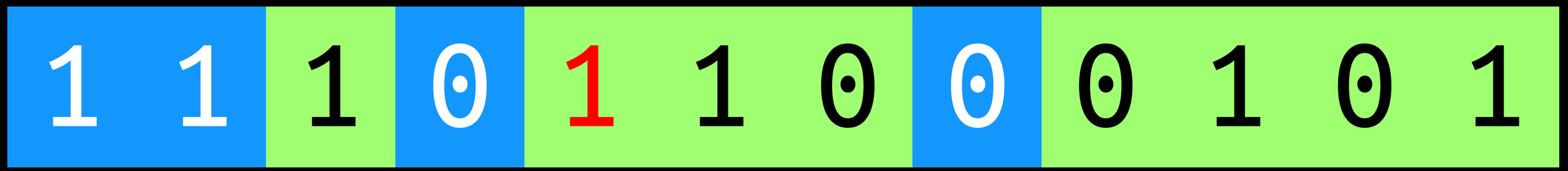
Find parity 4 (P_4)



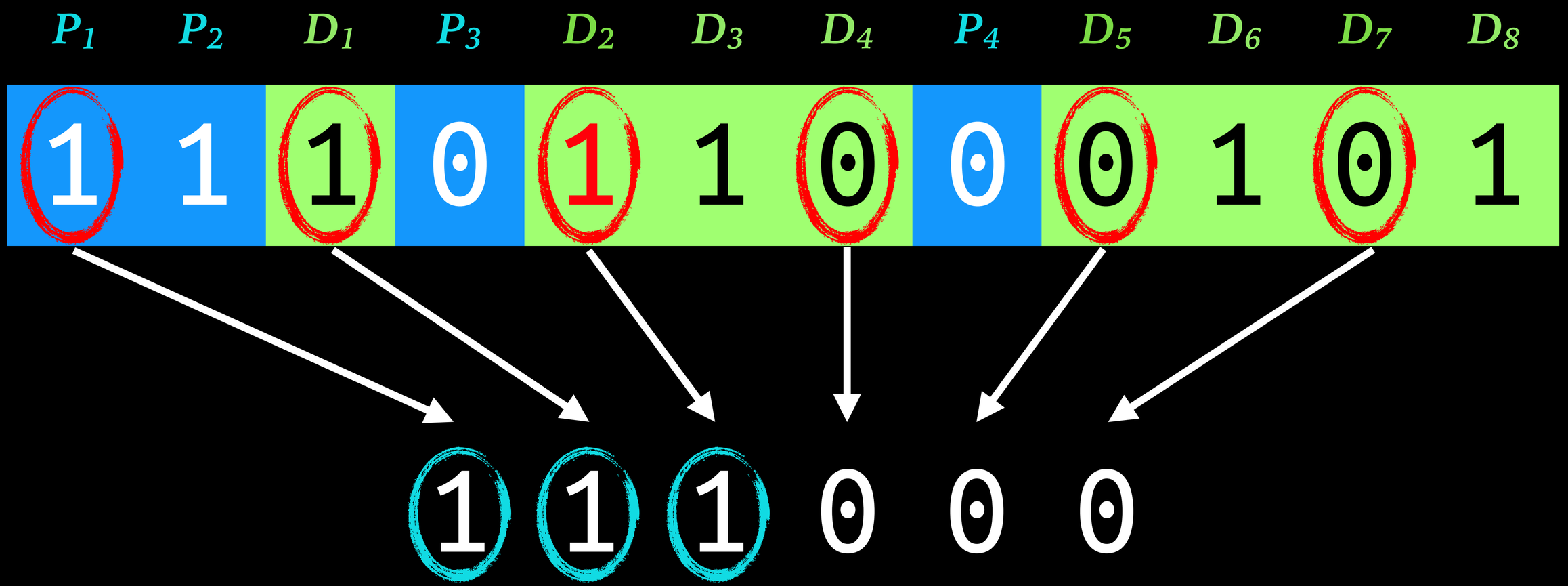
Pick 8, skip 8
(because the bit we are finding is at position 8)

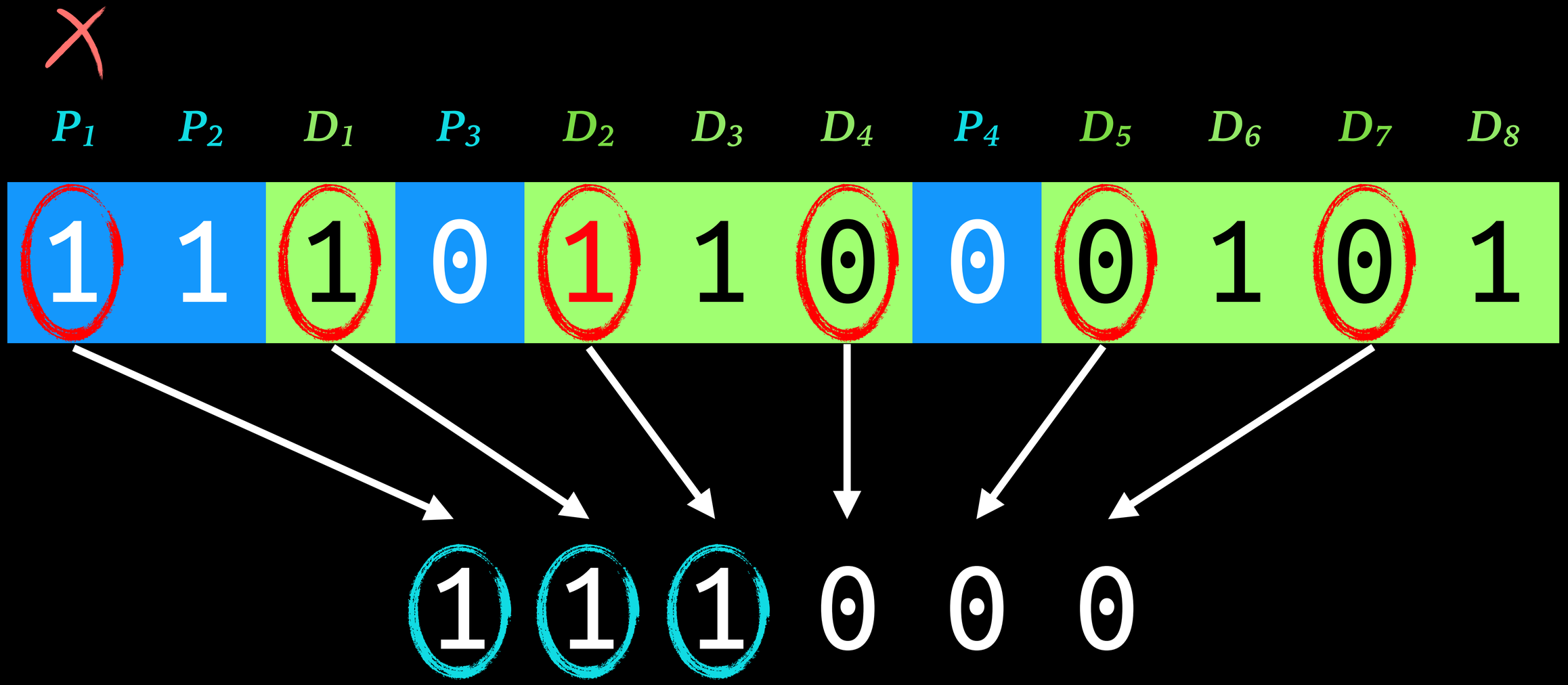
Checking

Given that D_2 is a faulty bit



Let’s check with P_1

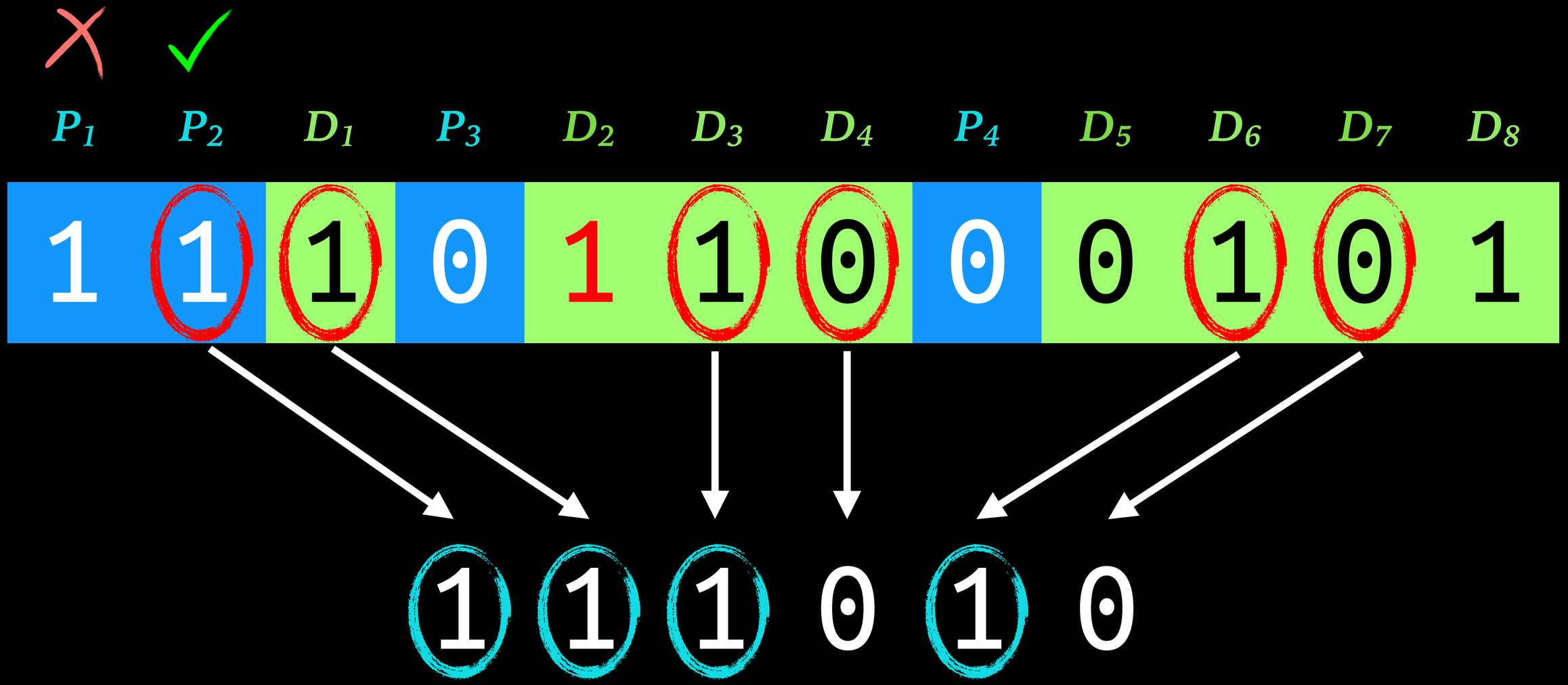




There is an odd number of 1’s, so something related to P_1 is incorrect.

Let’s check with P_2

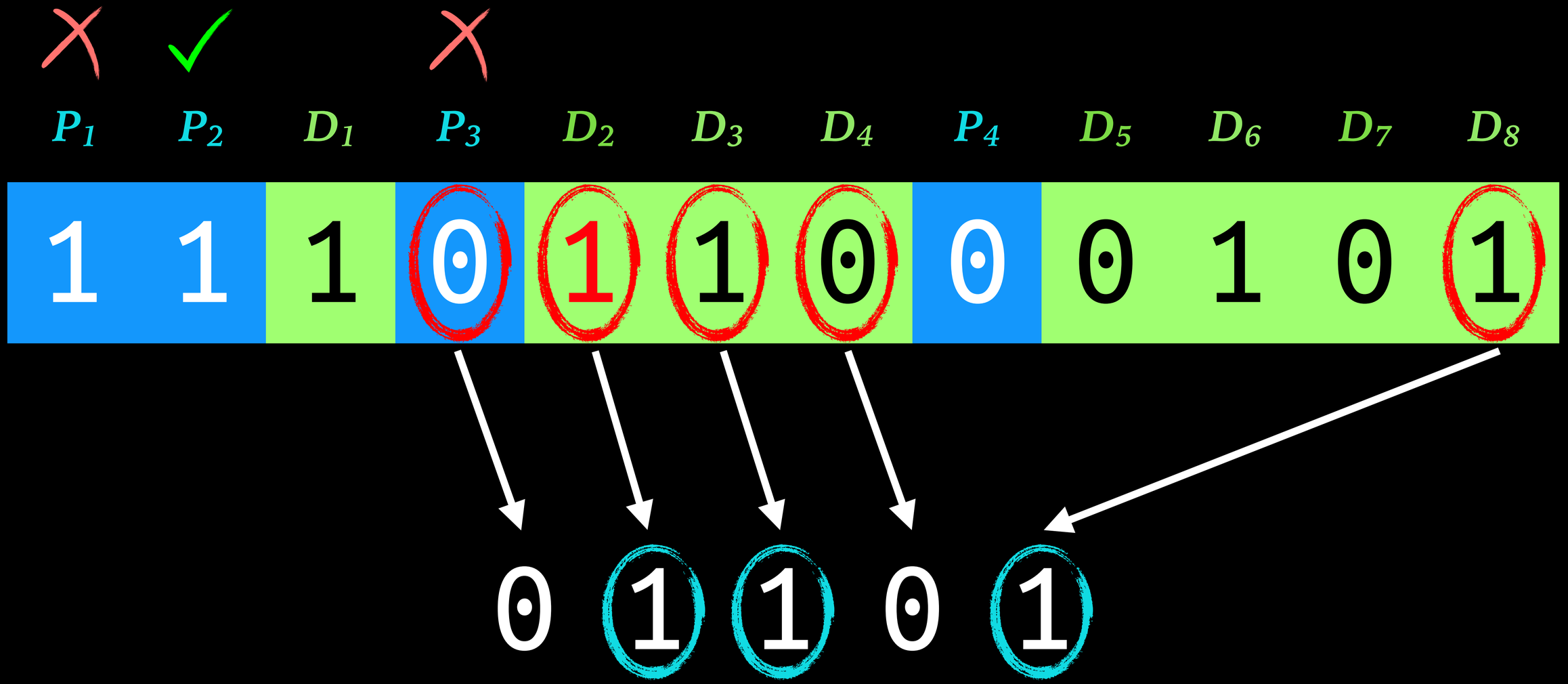
P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6	D_7	D_8
1	1	1	0	1	1	0	0	0	1	0	1



There is an even number of 1’s, so what P_2 is storing are correct.

Let’s check P_3

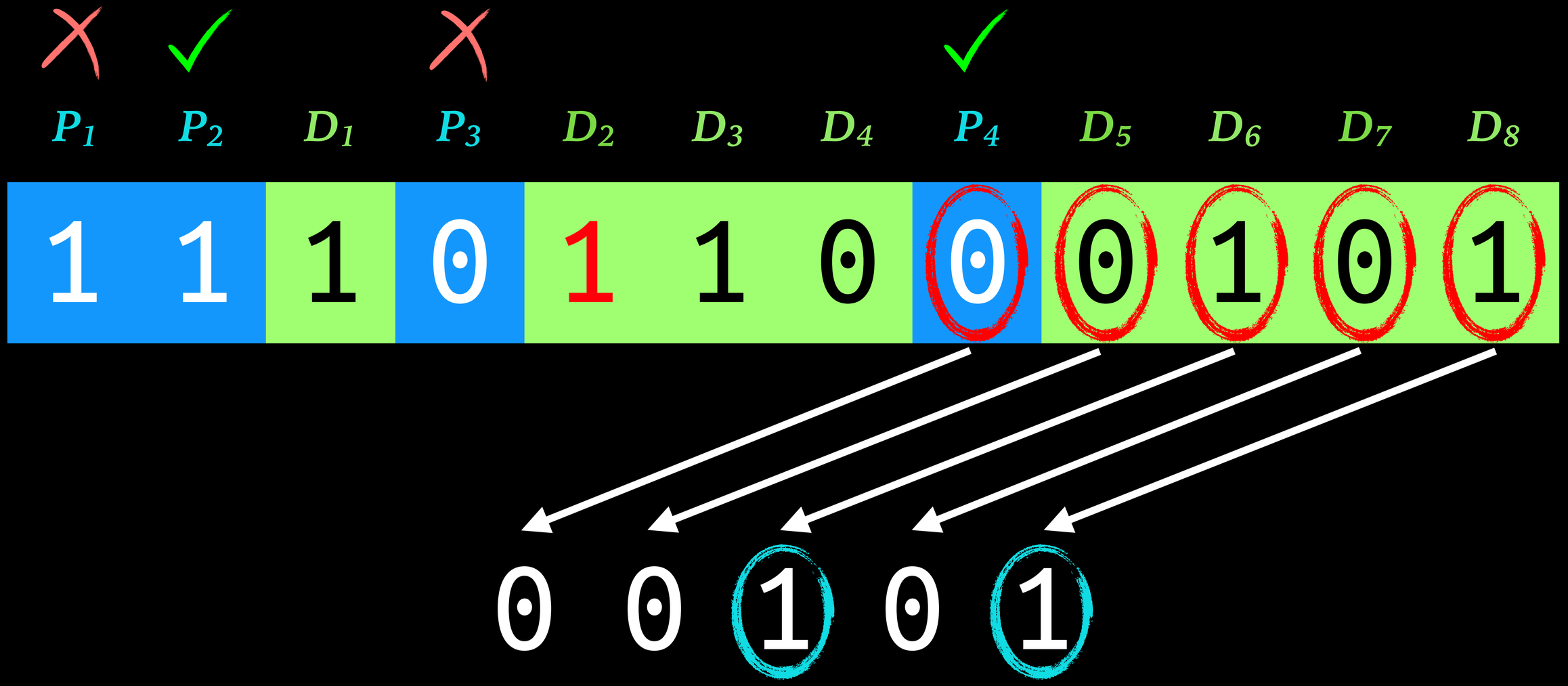
P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6	D_7	D_8
1	1	1	0	1	1	0	0	0	1	0	1



There is an odd number of 1's, so something related with P_3 is incorrect.

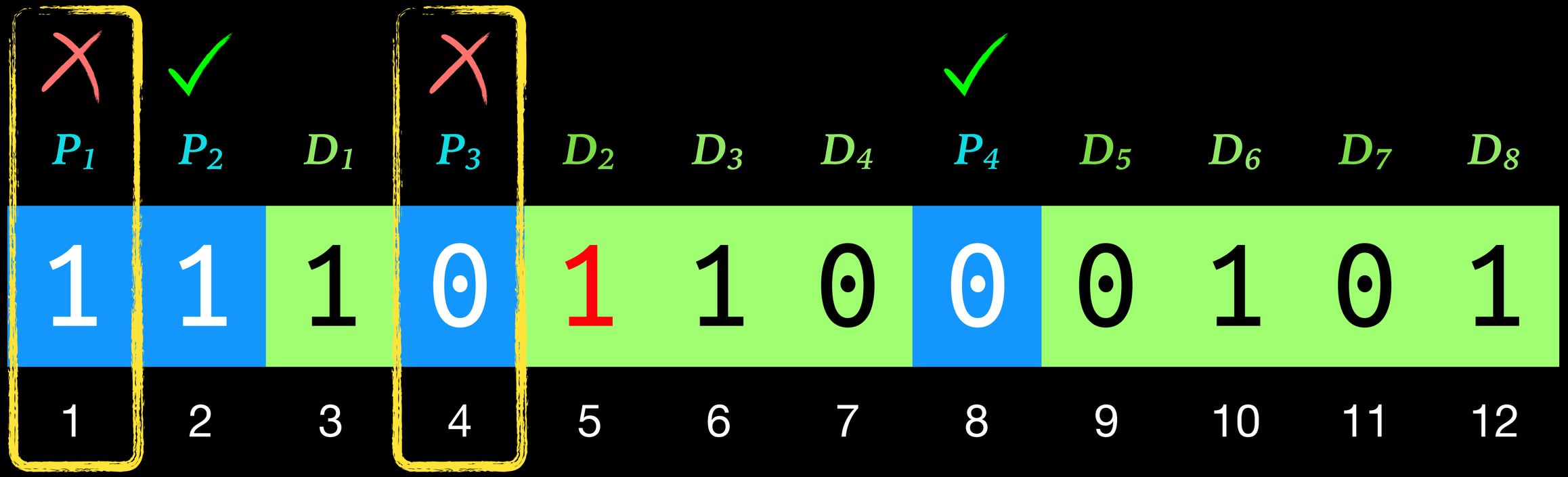
Let's check P_4

P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6	D_7	D_8
1	1	1	0	1	1	0	0	0	1	0	1

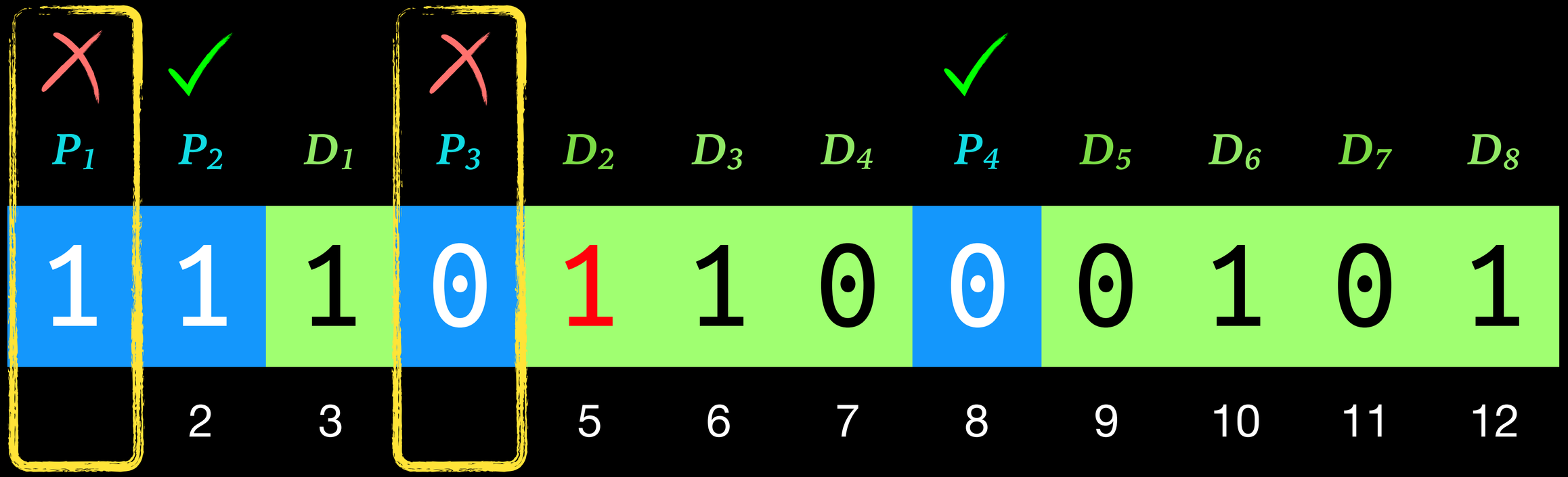


There is an even number of 1’s, so what P_4 is storing are correct.

Find the position of the faulty bit

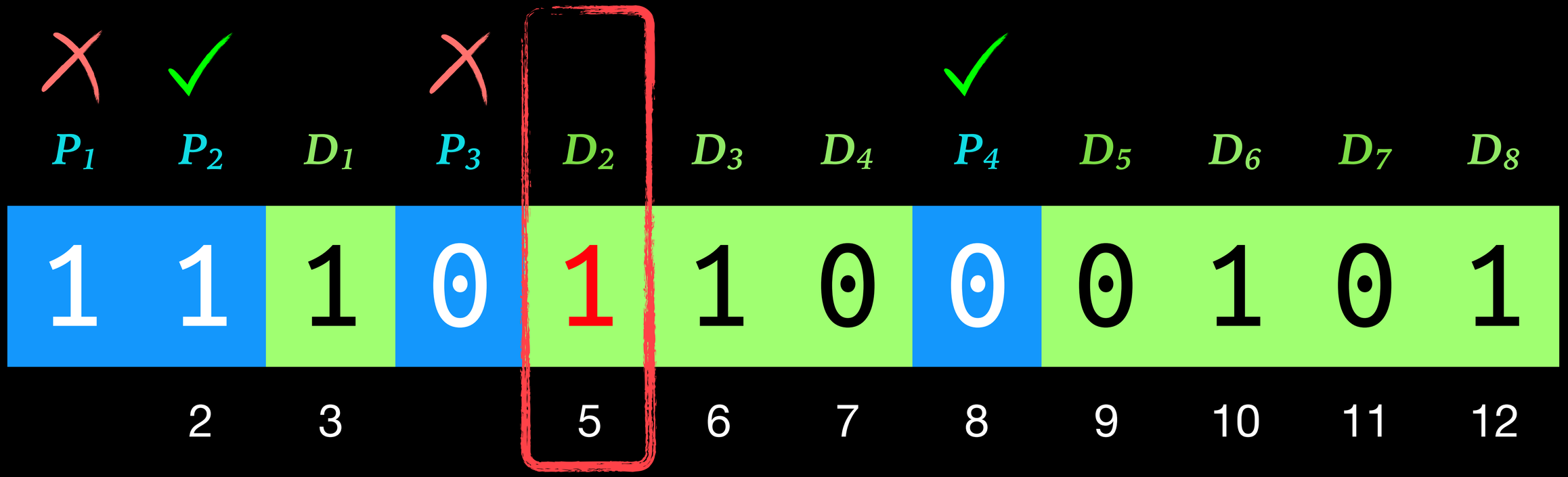


Find the position of the faulty bit



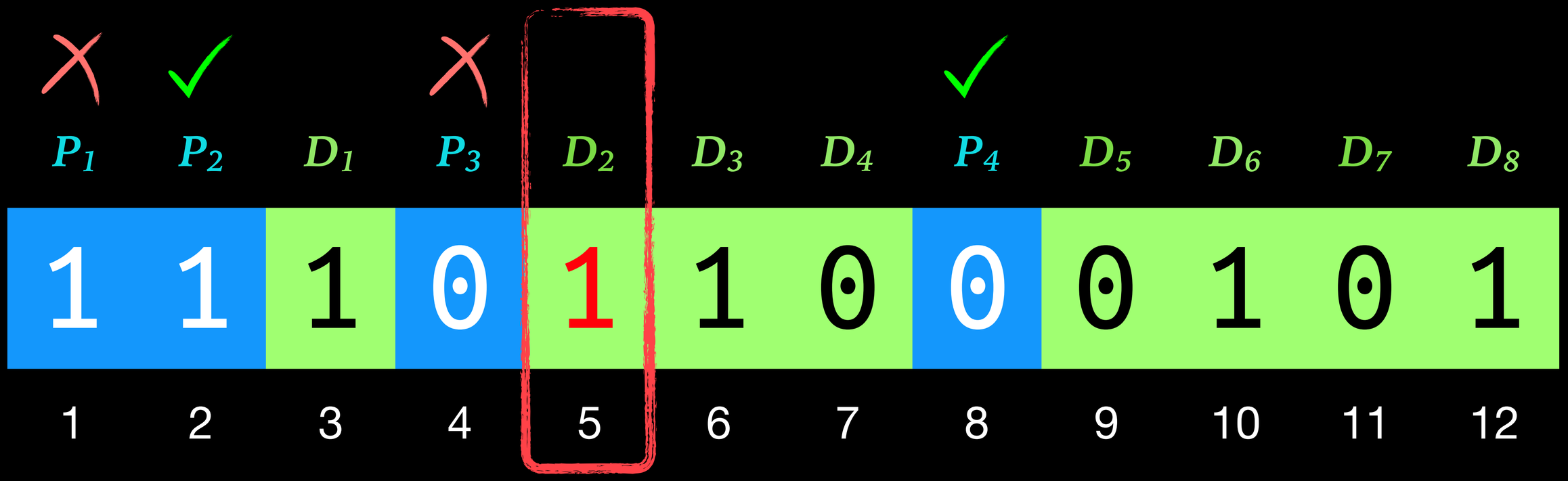
1 + 4

Find the position of the faulty bit

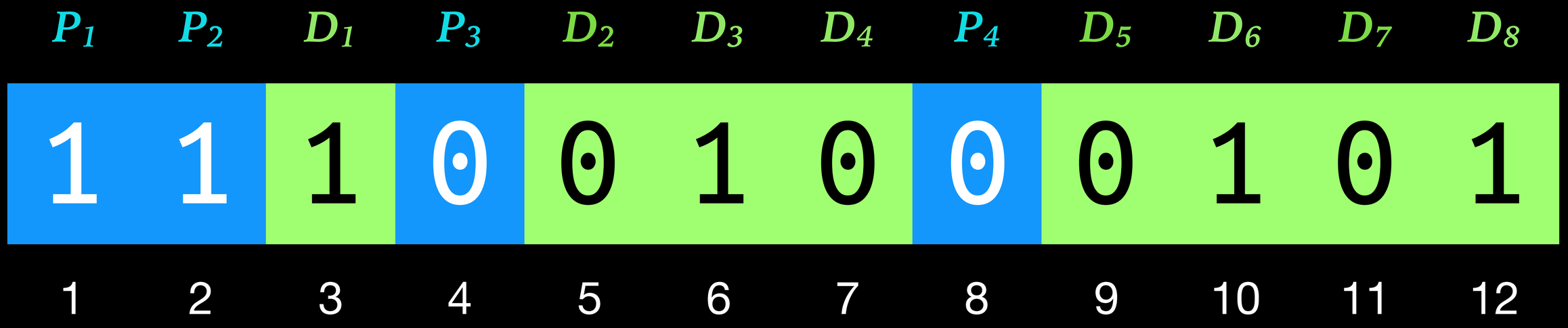


1 + 4 = 5

Invert the bit back



Invert the bit back



Original data:

1 0 1 0 0 1 0 1

Pros

Easy and fast recovery of a 1-bit error

Takes less space for keeping parity
(compared to full-backup methods)

Cons

Can only recover up to 2 bits
(without making recovery errors)

Suitable for rarely-breaking devices
(as it cannot recover more than a 2-bit error)

Demonstration

Members

59070022	คุณานนต์ ศรีสันติโรจน์
59070043	ฐิติภัทร วรรัตน์
59070058	ติณณภพ ไชยเขตต์
59070087	นาธาร เยี่ยงศุภพานนทร์

เซคชั่น 3 (พฤหัสบดี–เช้า)