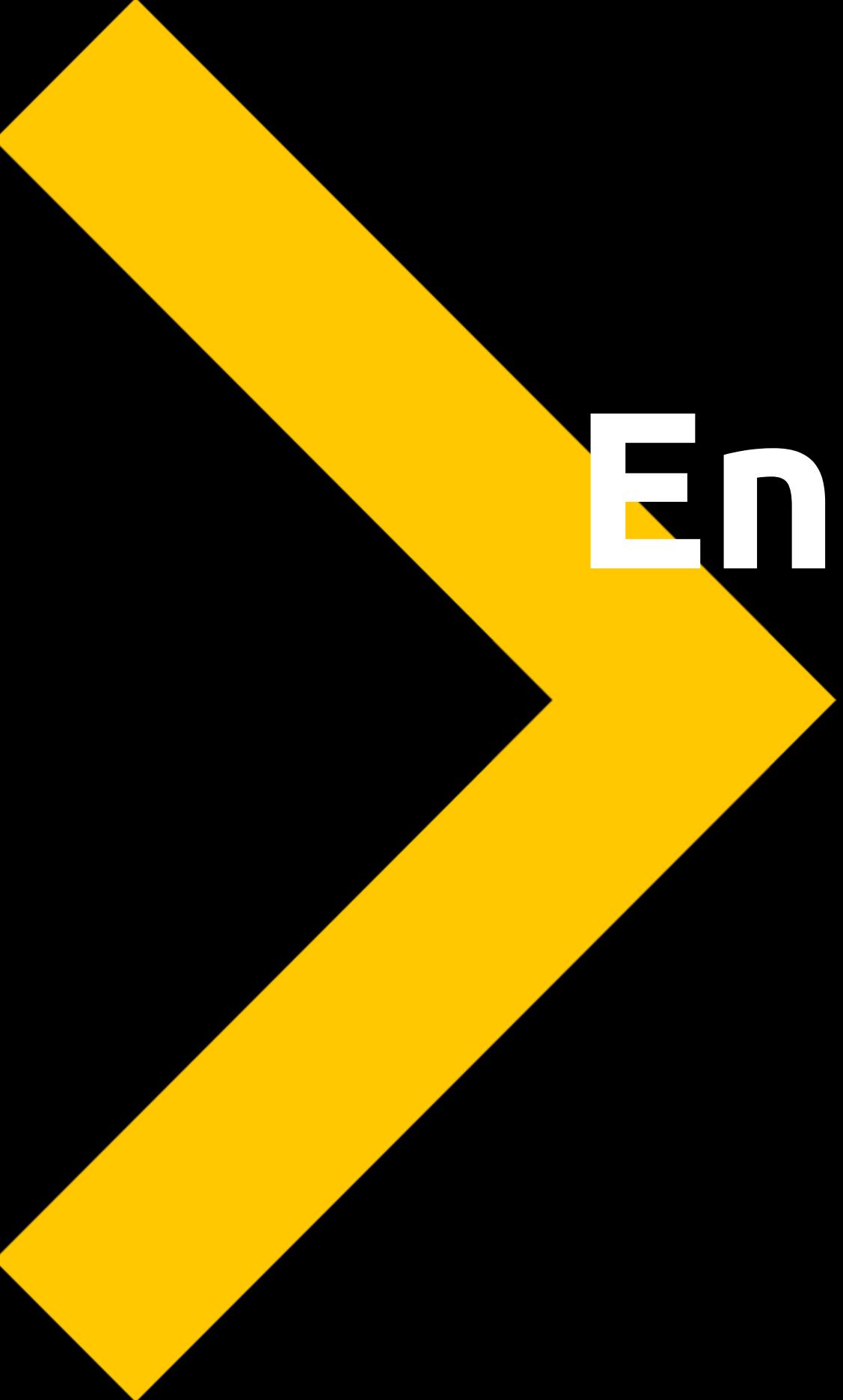


Hamming Code

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



Encoding

Given that our data is

1 0 1 0 0 1 0 1

8 bits

$$2^3 = 8 \text{ bits}$$

Number of parity bits:

$$3 + 1$$

Number of parity bits:

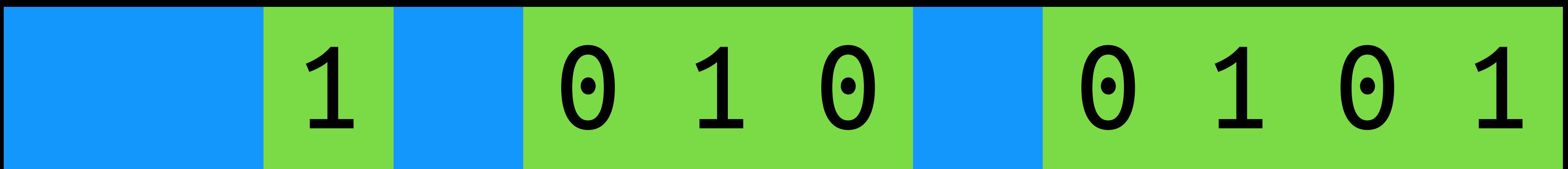
4

Arranging bits

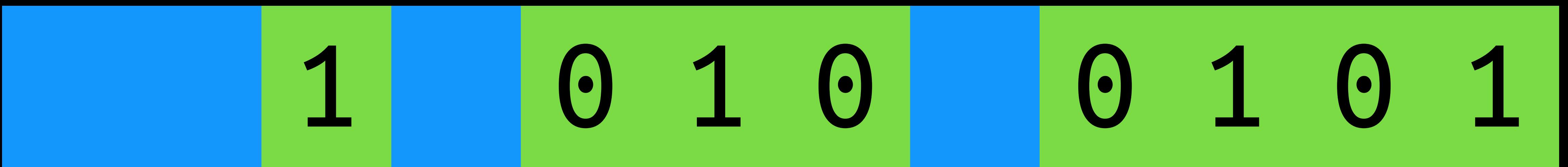
1 0 1 0 0 1 0 1



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



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

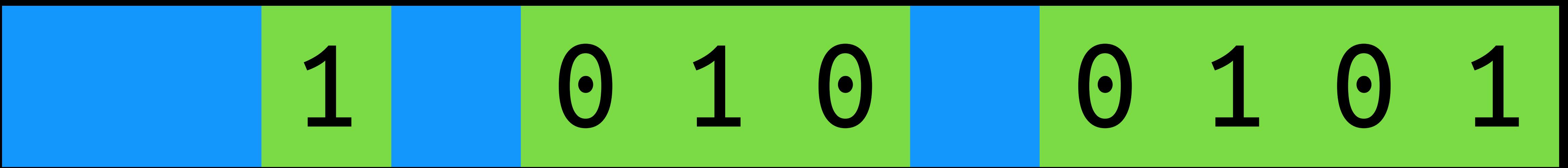


P_1 P_2

P_3

P_4

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



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

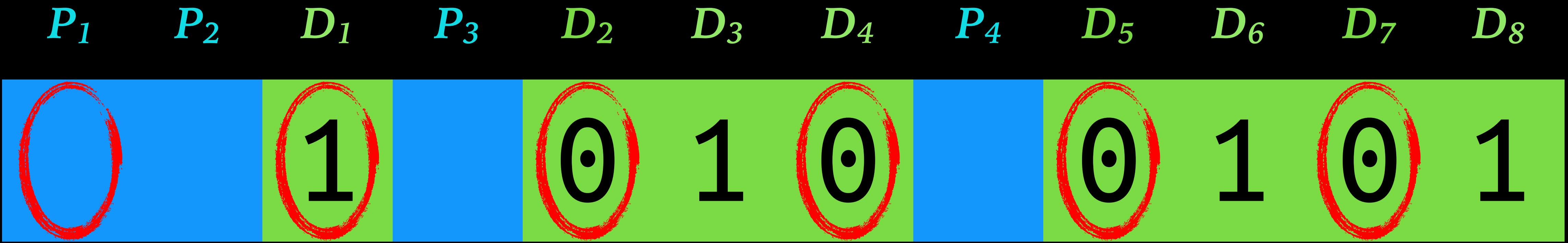
Find parity 1 (P_1)

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

	1		0	1	0		0	1	0	1
--	---	--	---	---	---	--	---	---	---	---

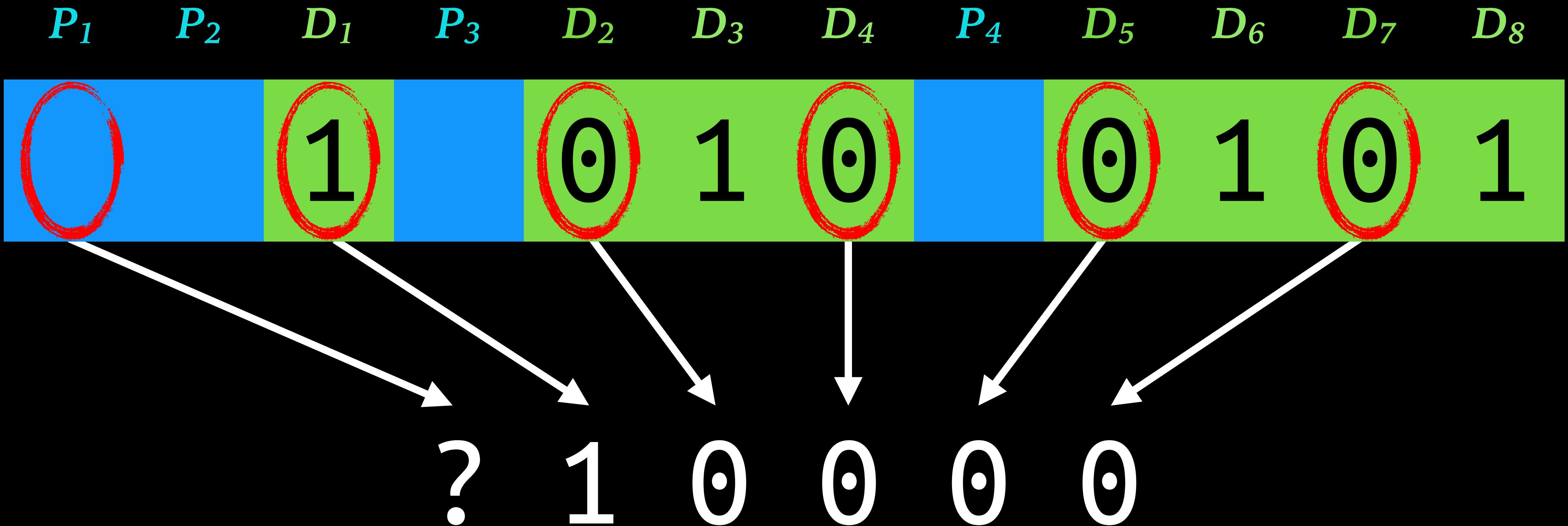
Pick 1, skip 1
(because the bit we are finding is at position 1)

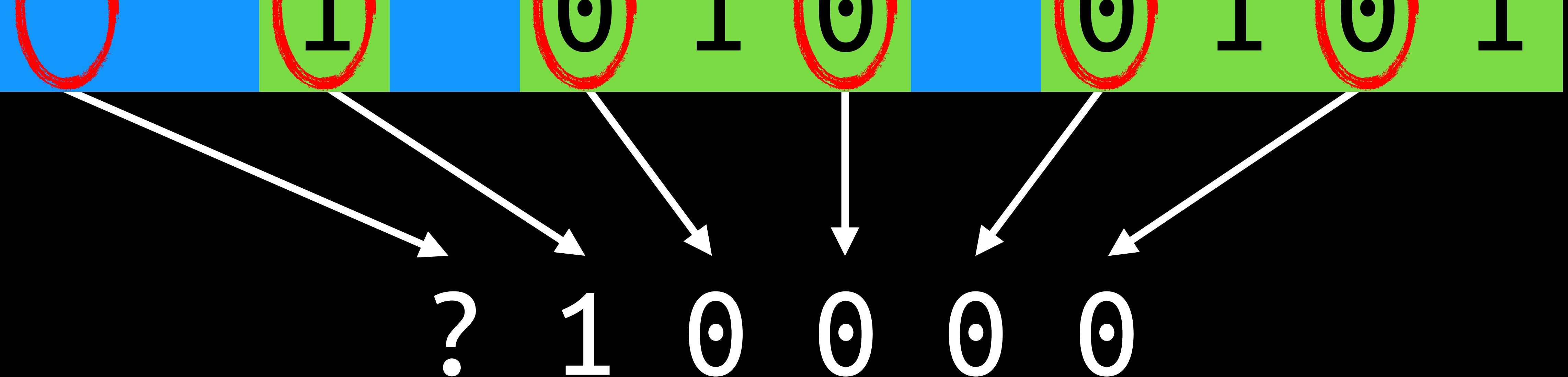
Find parity 1 (P_1)



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

Find parity 1 (P_1)



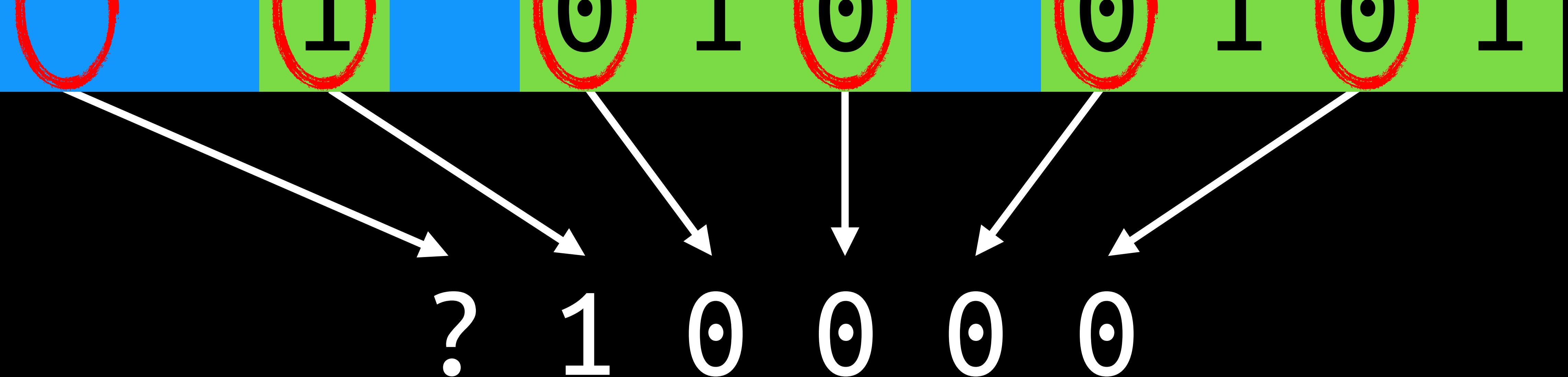


Count 1's

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.

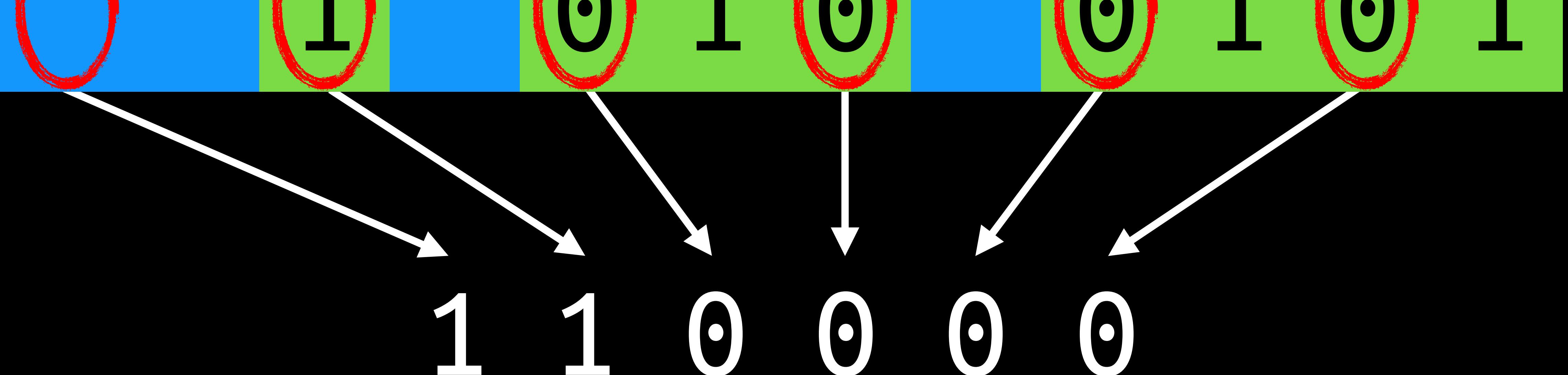


Count 1's

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.



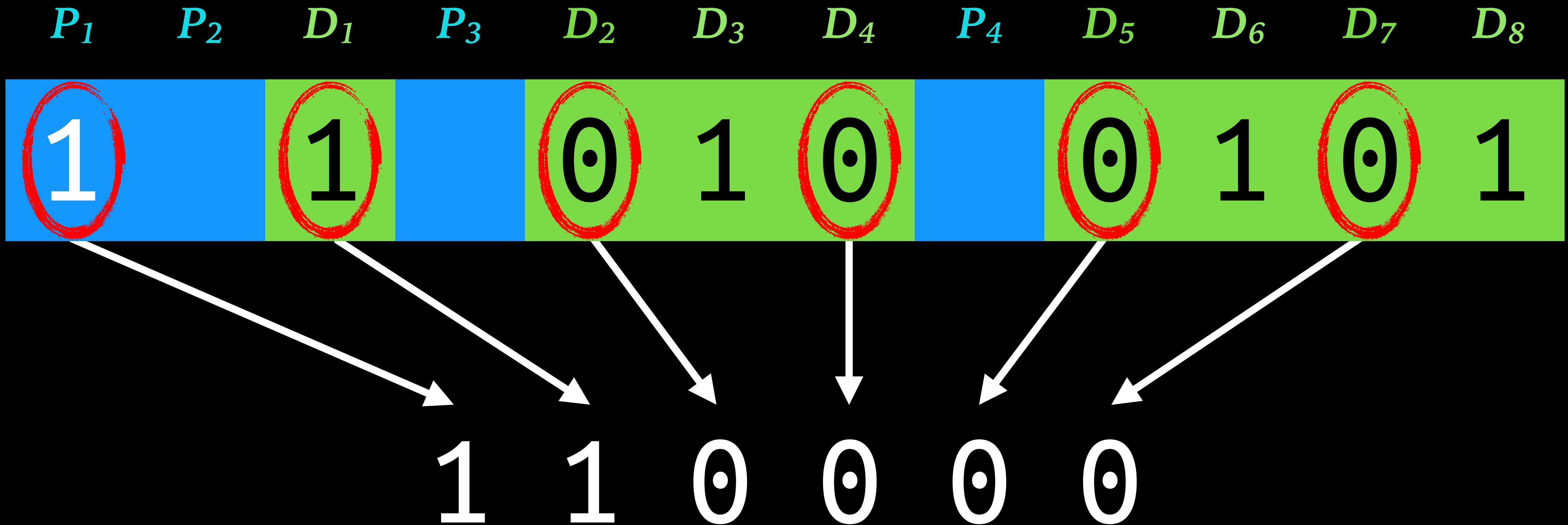
Count 1's

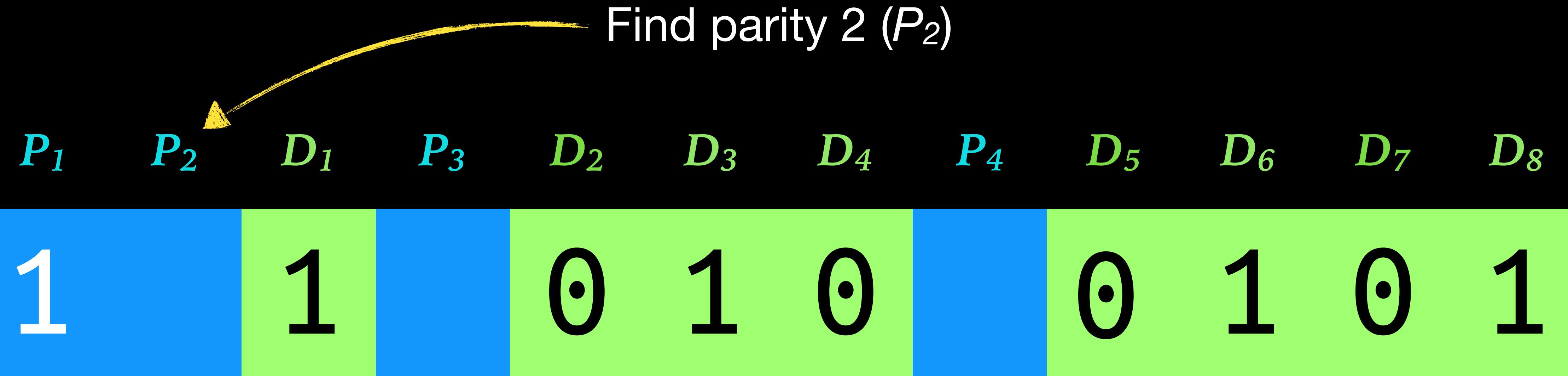
If the amount of 1's is odd, parity bit equals 1.

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)

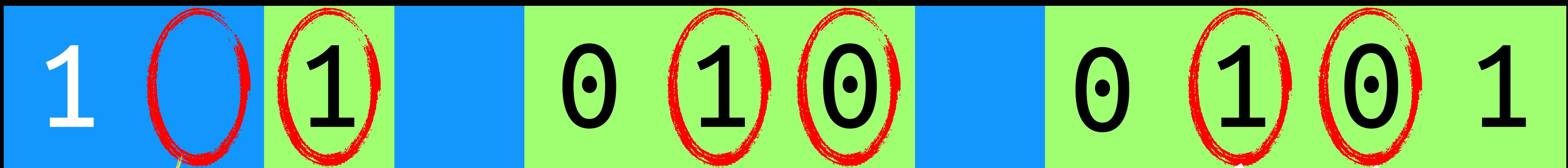




Pick 2, skip 2
(because the bit we are finding is at position 2)

Find parity 2 (P_2)

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



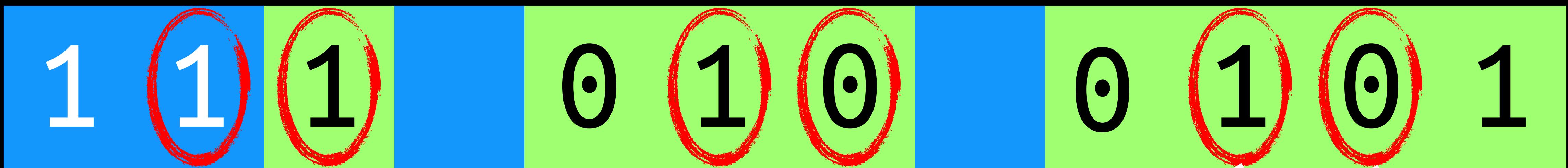
start here

? 1 1 0 1 0

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

Find parity 2 (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



? $\circled{1}$ $\circled{1}$ 0 $\circled{1}$ 0

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

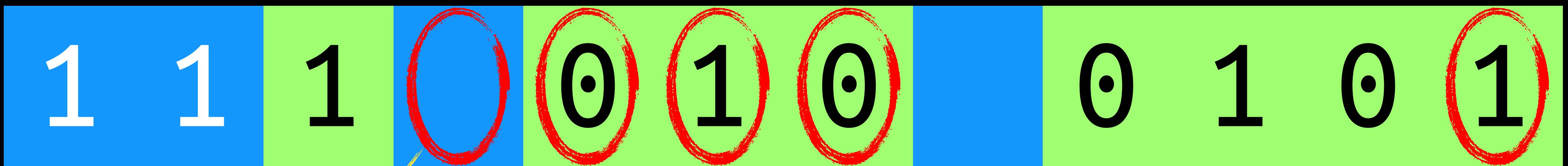
Find parity 3 (P_3)

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	0		0	1	0	1

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

Find parity 3 (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



start here

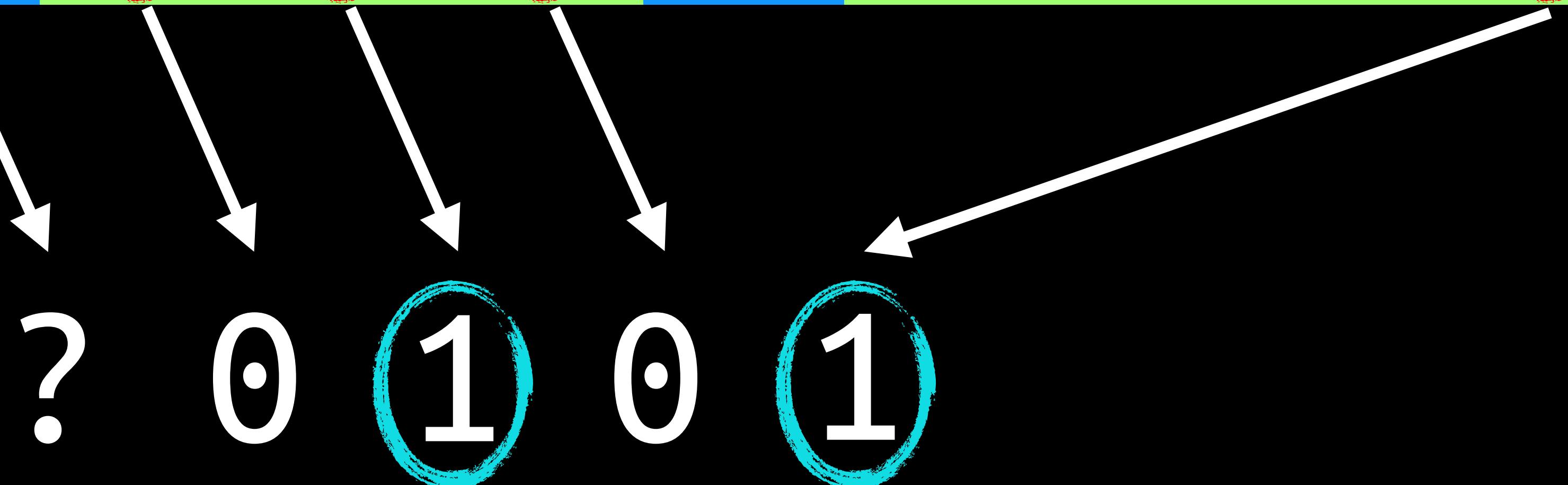
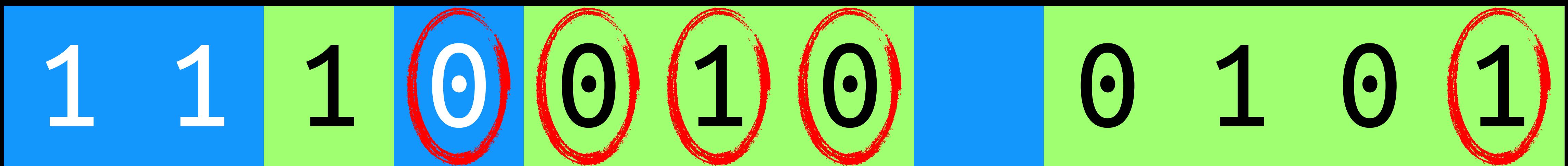


Pick 4, skip 4

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

Find parity 3 (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



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

Find parity 4 (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	0	1	0		0	1	0	1
---	---	---	---	---	---	---	--	---	---	---	---

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

Find parity 4 (P_4)

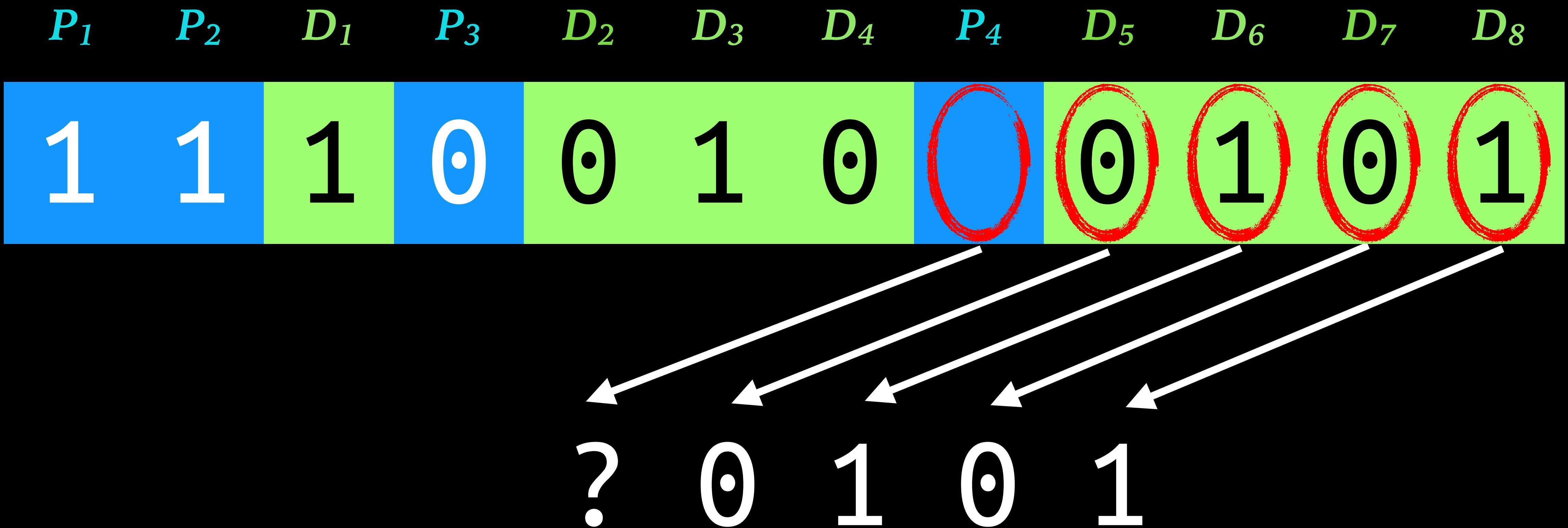


P_1 P_2 D_1 P_3 D_2 D_3 D_4 P_4 D_5 D_6 D_7 D_8

1	1	1	0	0	1	0		0	1	0	1
---	---	---	---	---	---	---	--	---	---	---	---

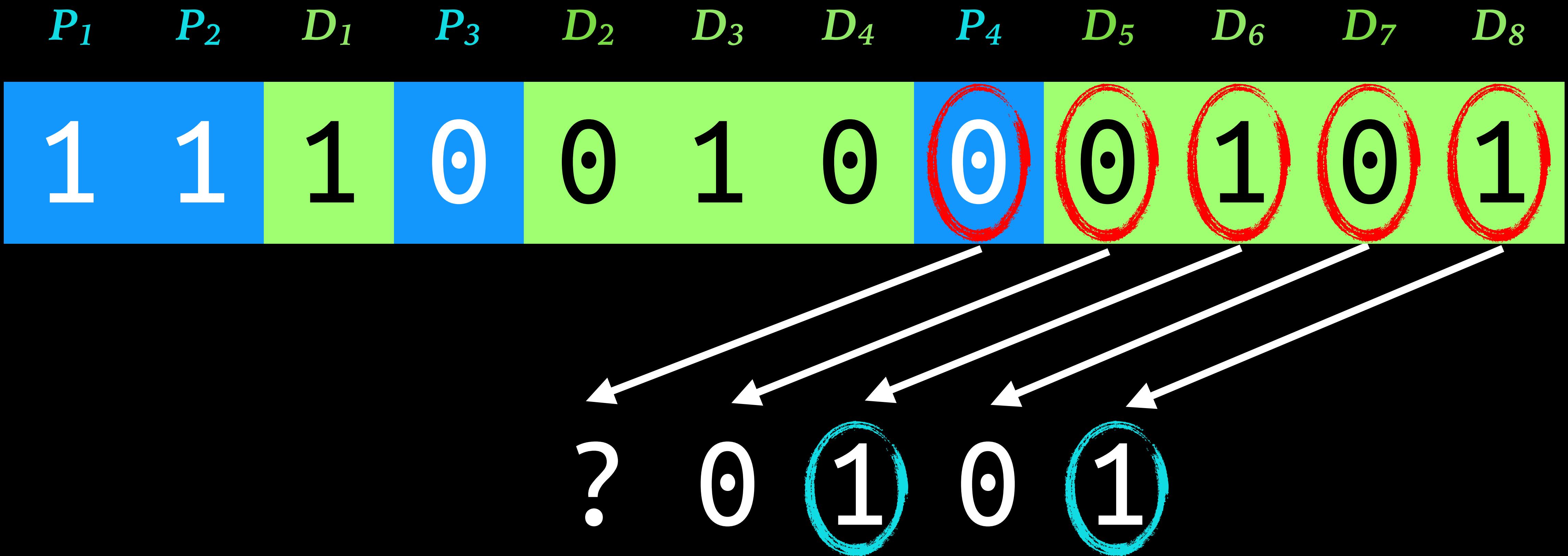
Pick 8, skip 8
(because the bit we are finding is at position 8)

Find parity 4 (P_4)



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

Find parity 4 (P_4)



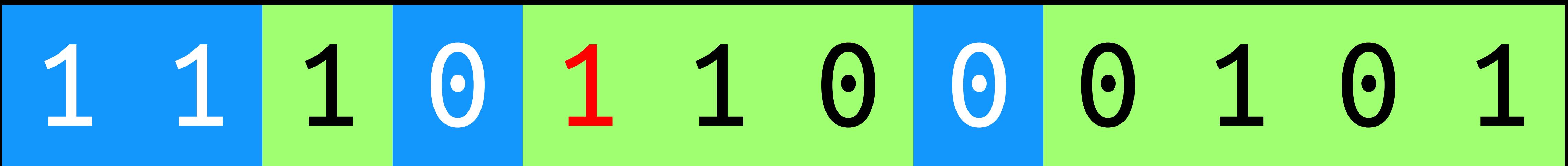
Pick 8, skip 8
(because the bit we are finding is at position 8)



Checking

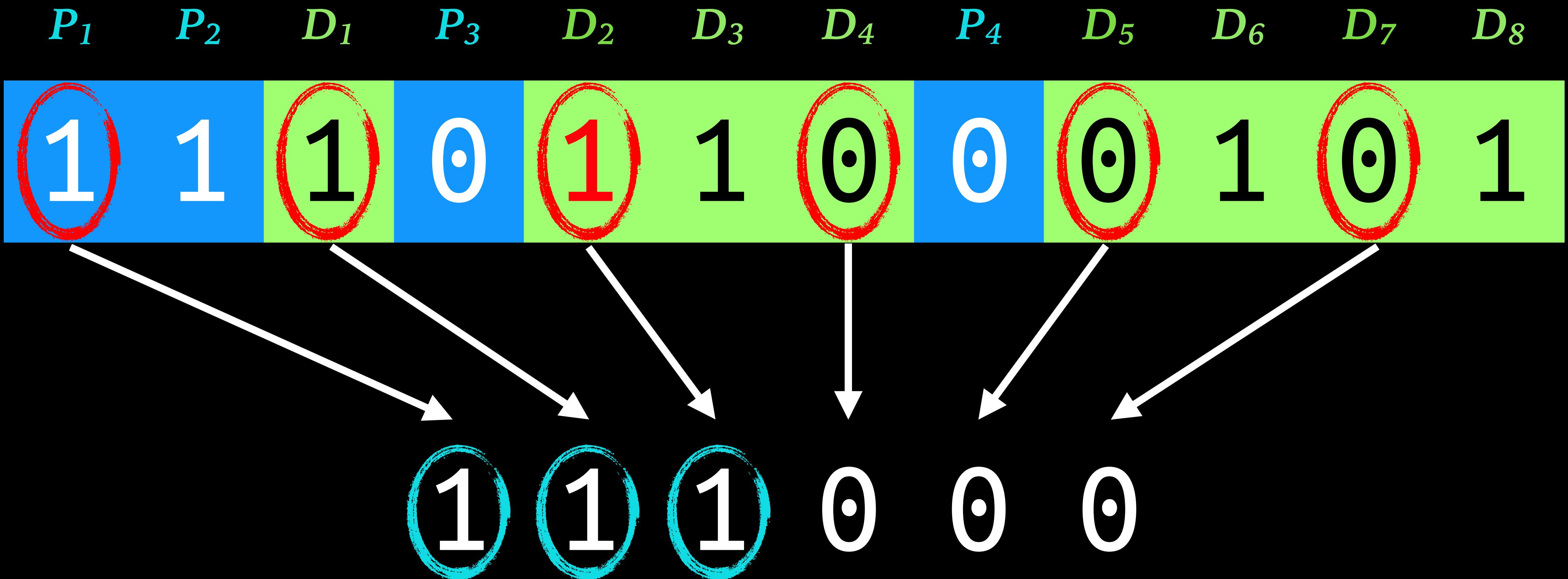
Given that D_2 is a faulty bit

D_2



it was
originally 0

Let's check with P_1



X

P_1

P_2

D_1

P_3

D_2

D_3

D_4

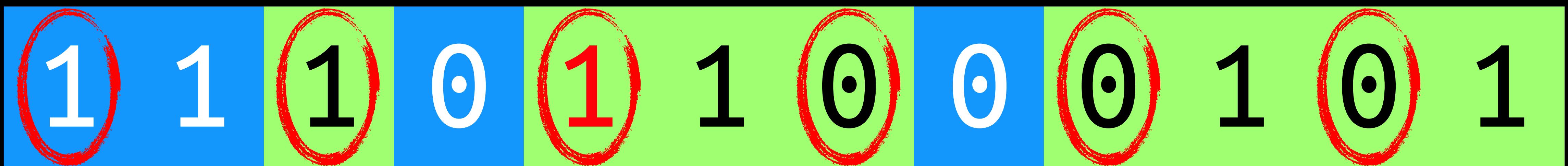
P_4

D_5

D_6

D_7

D_8



1	1	1	0	0	0
---	---	---	---	---	---

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

~~X~~

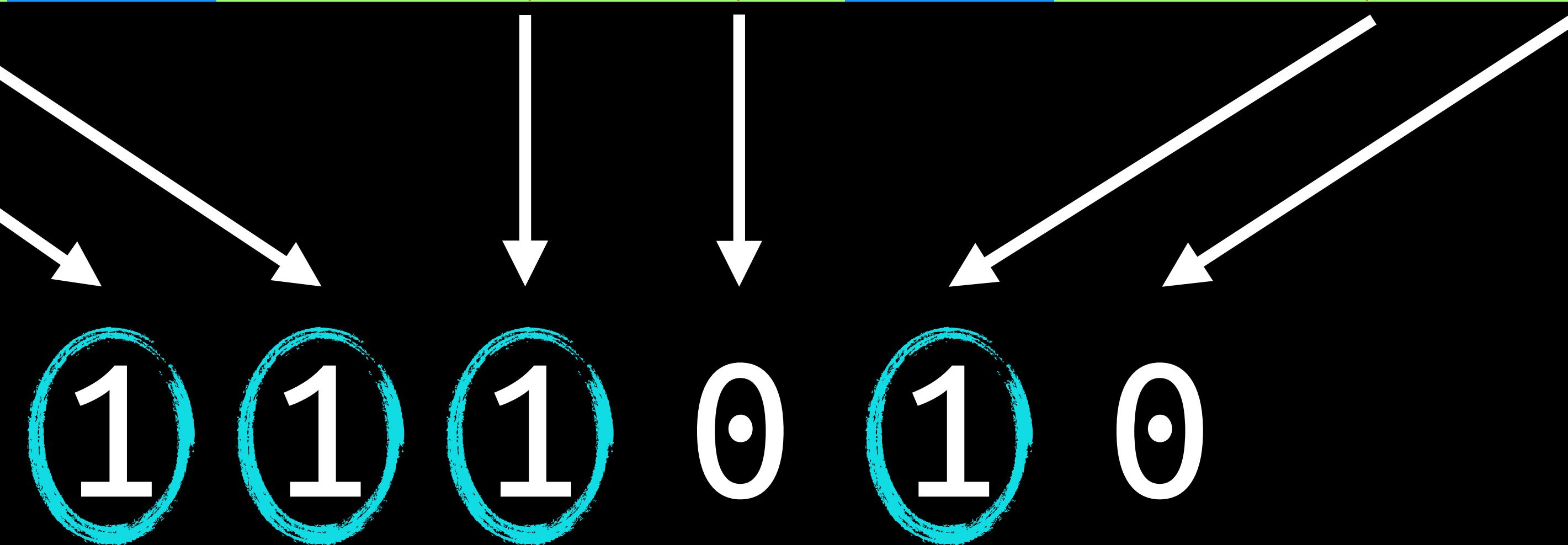
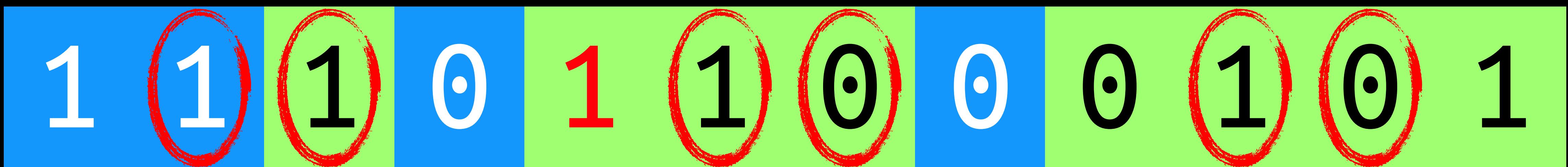
Let's check with P_2

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
---	---	---	---	---	---	---	---	---	---	---	---

✗ ✓

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



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

\times \checkmark

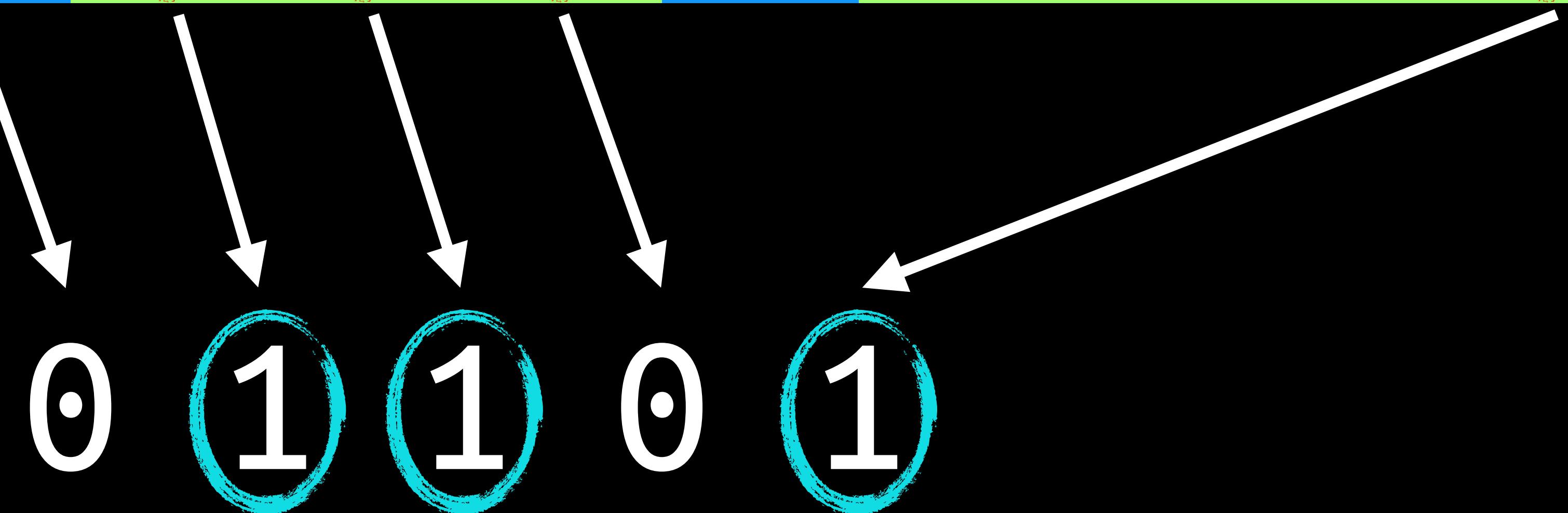
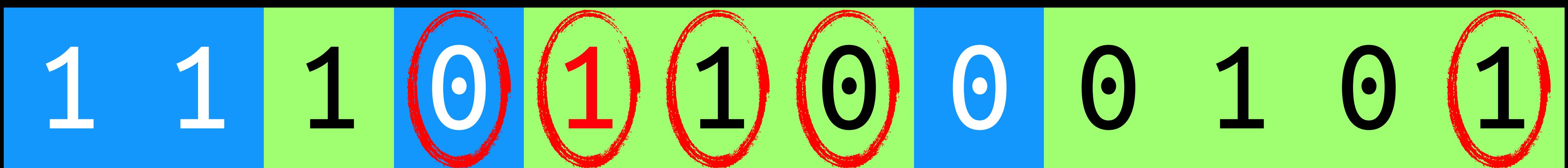
Let's check P_3

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
---	---	---	---	---	---	---	---	---	---	---	---

\times \checkmark

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



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

\times

\checkmark

\times

P_1

P_2

D_1

P_3

D_2

D_3

D_4

P_4

D_5

D_6

D_7

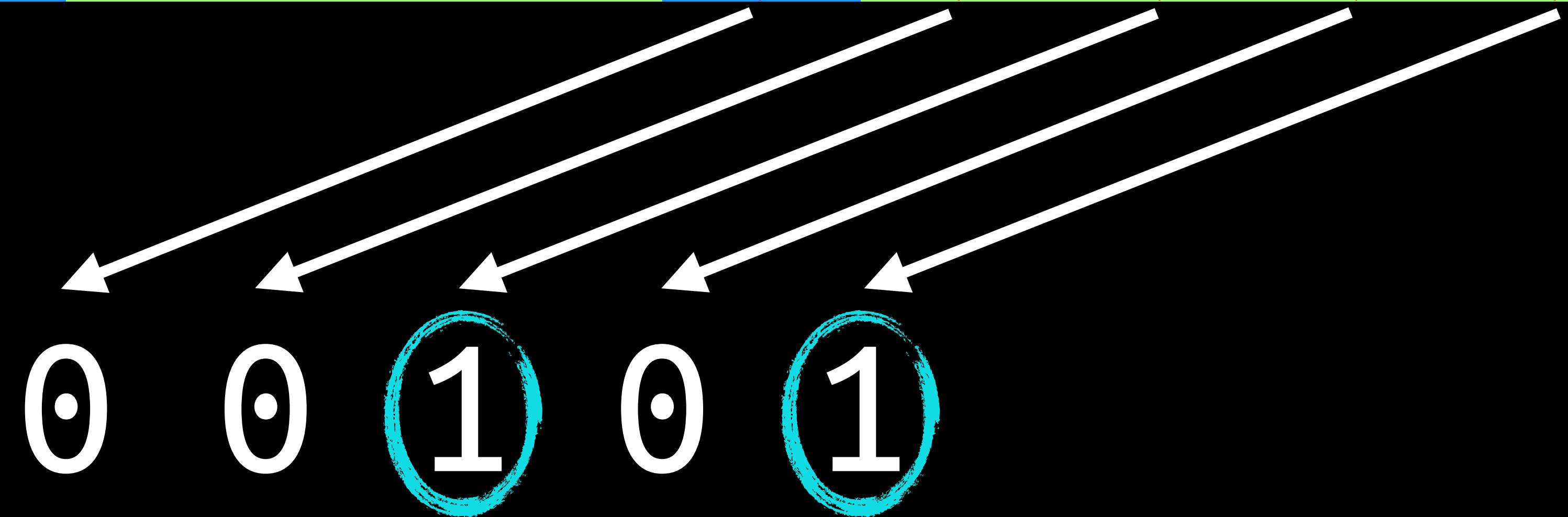
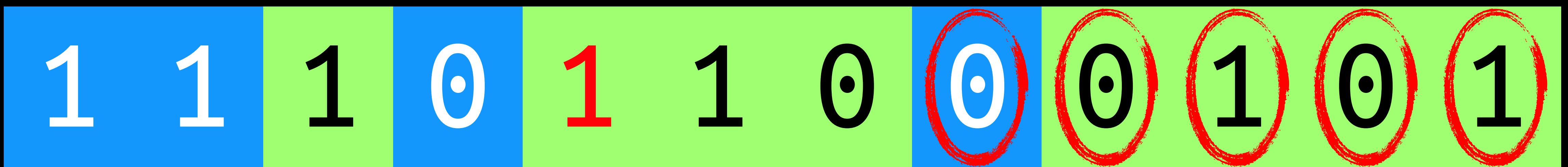
D_8

Let's check P_4

1	1	1	0	1	1	0	0	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---

\times \checkmark \times \checkmark

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



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

Find the position of the faulty bit

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

Find the position of the faulty bit

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

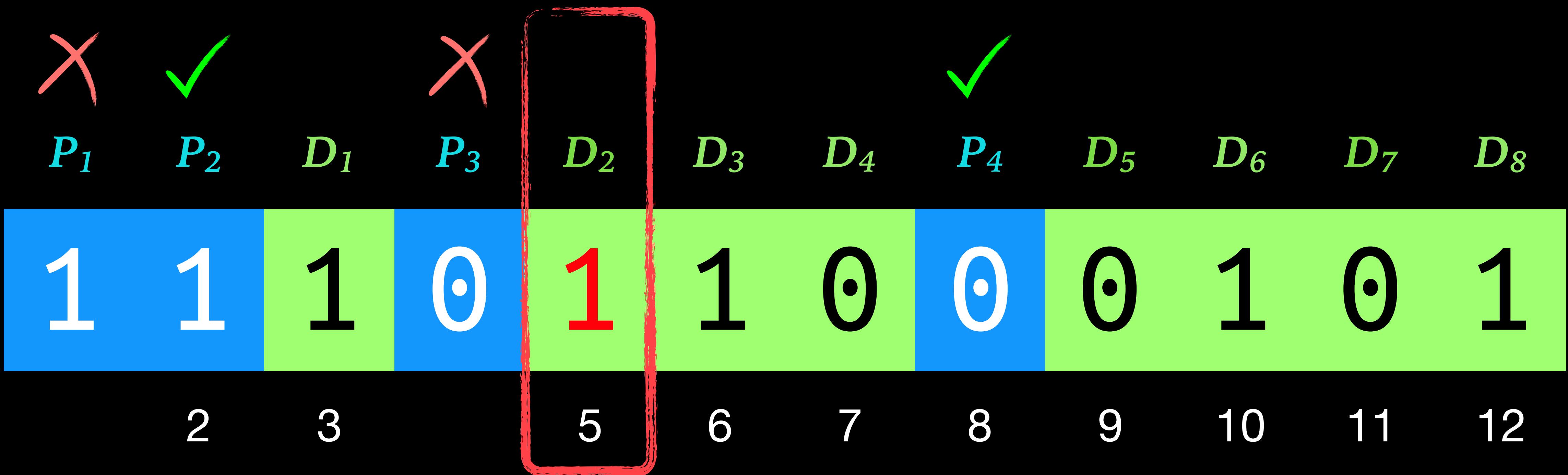
2 3 5 6 7 8 9 10 11 12

$$1 + 4$$

Find the position of the faulty bit

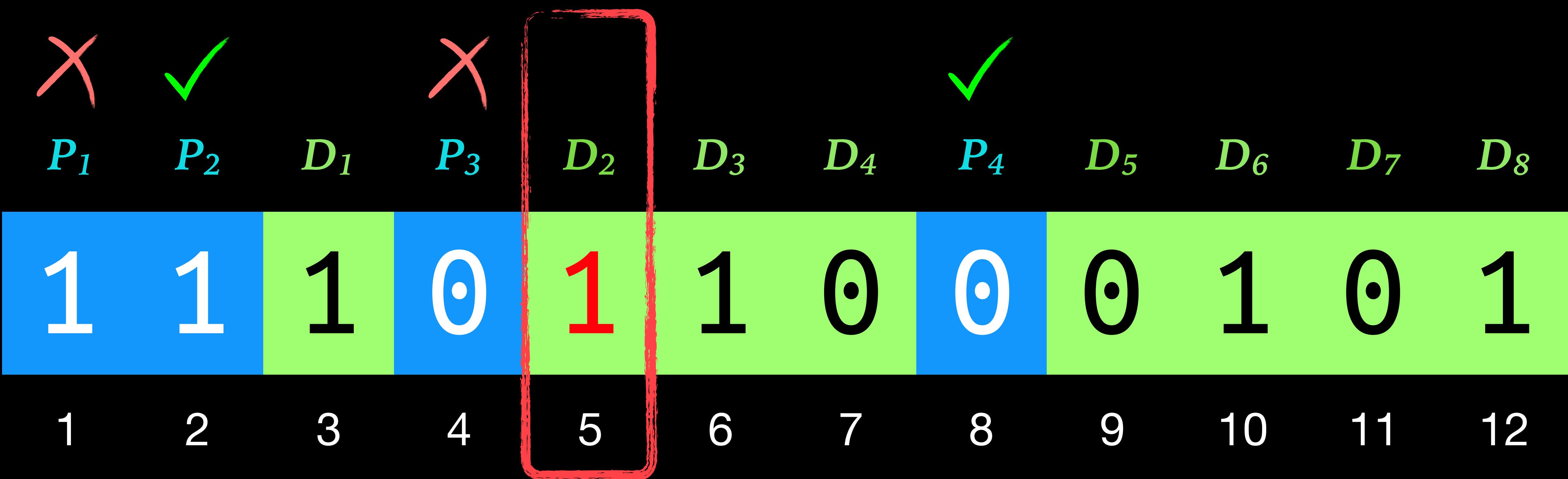
P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6	D_7	D_8
✗	✓		✗				✓				
1	1	1	0	1	1	0	0	0	1	0	1

2 3 5 6 7 8 9 10 11 12



$$1 + 4 = 5$$

Invert the bit back



Invert the bit back

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

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)

A color photograph of an older man with grey hair, wearing a light-colored suit, white shirt, and striped tie. He is leaning over a vintage computer setup, looking towards the camera with a slight smile. The computer consists of a light-colored monitor unit with a CRT screen and a keyboard unit below it. A small circular badge with a portrait is visible on the front of the keyboard unit. The background is a plain, light-colored wall.

Demonstration

Members

59070022	คุณานนต์ ศรีสันติ โรจน์
59070043	ธัญtip ภัทร วรรตนา
59070058	ติณณภาพ ไชยเขตต์
59070087	นางาร เยี่ยงคุภานนท์

เชคชั้น 3 (พฤหัสบดี-เช้า)