ECC Algorithm

(Error Checking & Correction)

Memory Product &
Technology Division

1997.8.30

ECC Code Generation

- **ECC code consists of 3bytes per 256bytes**
  - **Actually 22bit ECC code per 2048bits**
  - **22bit ECC code = 16bit line parity + 6bit column parity**
- **Data bit assignment table with ECC code**

I/O 7	I/O 6		I/O 1	I/O 0	
D(00000000,111)	D(00000000,110)		D(00000000,001)	D(00000000,000)	1st Byte
D(00000001,111)	D(00000001,110)		D(00000001,001)	D(00000001,000)	2nd Byte
D(00000010,111)	D(00000010,110)	• • • •	D(00000010,001)	D(00000010,000)	2rd Byte
D(11111110,111)	D(11111110,110)		D(11111110,001)	D(11111110,000)	255th Byte
D(11111111,111)	D(11111111,110)		D(11111111,001)	D(11111111,000)	256th Byte

- **ECC code assignment table**

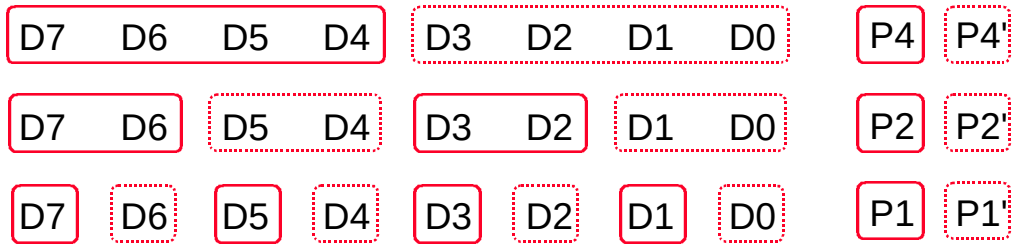
I/O 7	I/O 6	I/O 5	I/O 4	I/O 3	I/O 2	I/O 1	I/O 0
P64	P64`	**P32**	P32`	**P16**	P16`	**P8**	P8`
P1024	P1024`	**P512**	P512`	**P256**	P256`	**P128**	P128`
P4	P4`	**P2**	P2`	**P1**	P1`	1	1

P8 ~ P1024 : Line parity
P1 ~ P4 : Column parity

Fail bit address offset
=> **(P1024,P512,P256,P128,P64,P32,P16,P8, P4,P2,P1)**

ECC Code Generation Simple Example

- **Parity Generation (In case of 8bit input)**



P4=D7(+)D6(+)D5(+)D4 P4`=D3(+)D2(+)D1(+)D0 ** (+) means XOR (Even parity)*

P2=D7(+)D6(+)D3(+)D2 P2`=D5(+)D4(+)D1(+)D0

P1=D7(+)D5(+)D3(+)D1 P1`=D6(+)D4(+)D2(+)D0

- **For example (In case of 1 bit fail)**

Original data : 1 0 1 0 1 0 1 0

P4=1(+)0(+)1(+)0=0, P4`=1(+)0(+)1(+)0=0
P2=1(+)0(+)1(+)0=0, P2`=1(+)0(+)1(+)0=0
P1=1(+)1(+)1(+)1=0, P1`=0(+)0(+)0(+)0=0

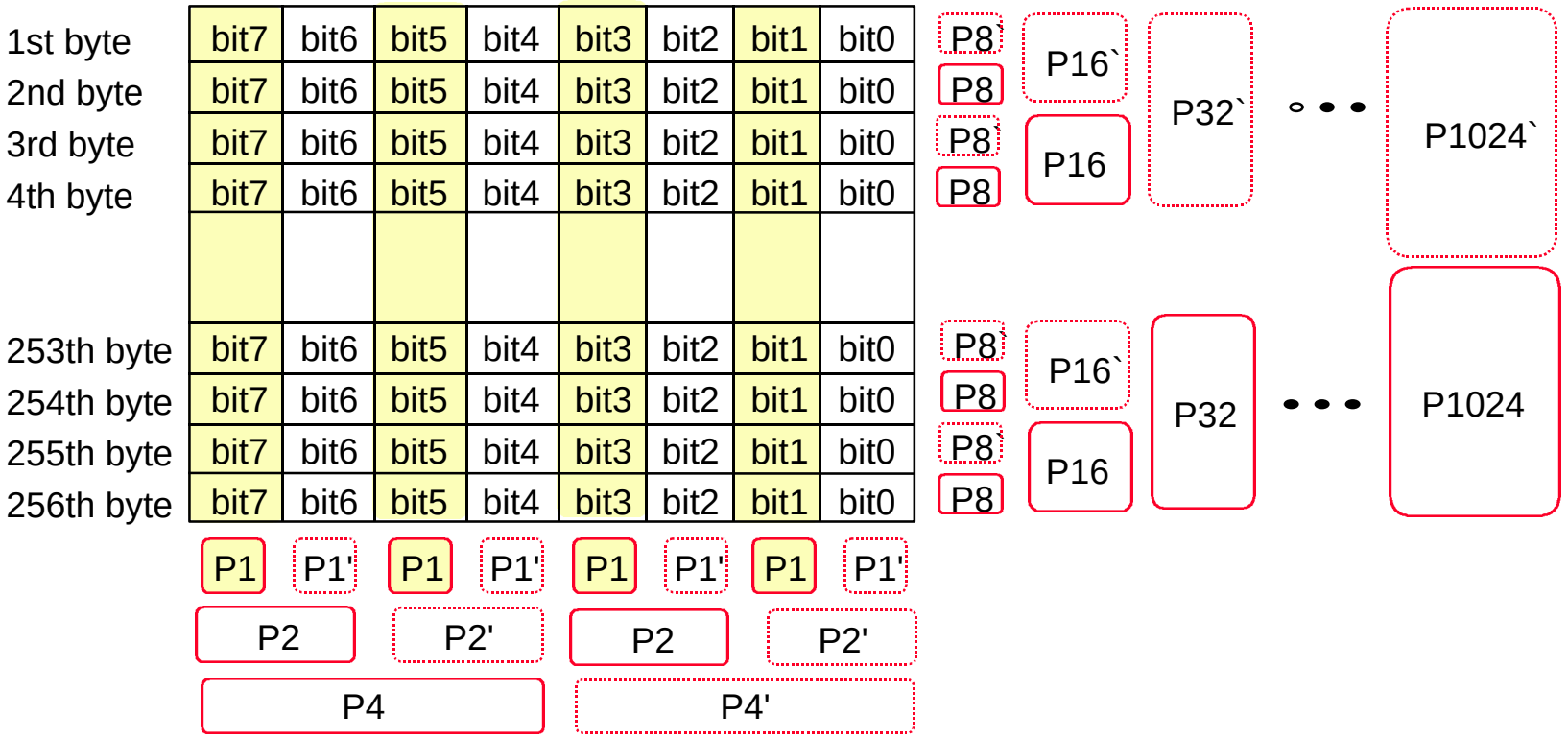
Changed data : 1 0 1 **1** 1 0 1 0

P4=1(+)0(+)1(+)**1**=**1**, P4`=1(+)0(+)1(+)0=0
P2=1(+)0(+)1(+)0=0, P2`=1(+)**1**(+)1(+)0=**1**
P1=1(+)1(+)1(+)1=0, P1`=0(+)**1**(+)0(+)0=**1**

In here, fail bit locate is column address offset *(P4,P2,P1=100=I/O4)*

ECC Code Generation Methode (1)

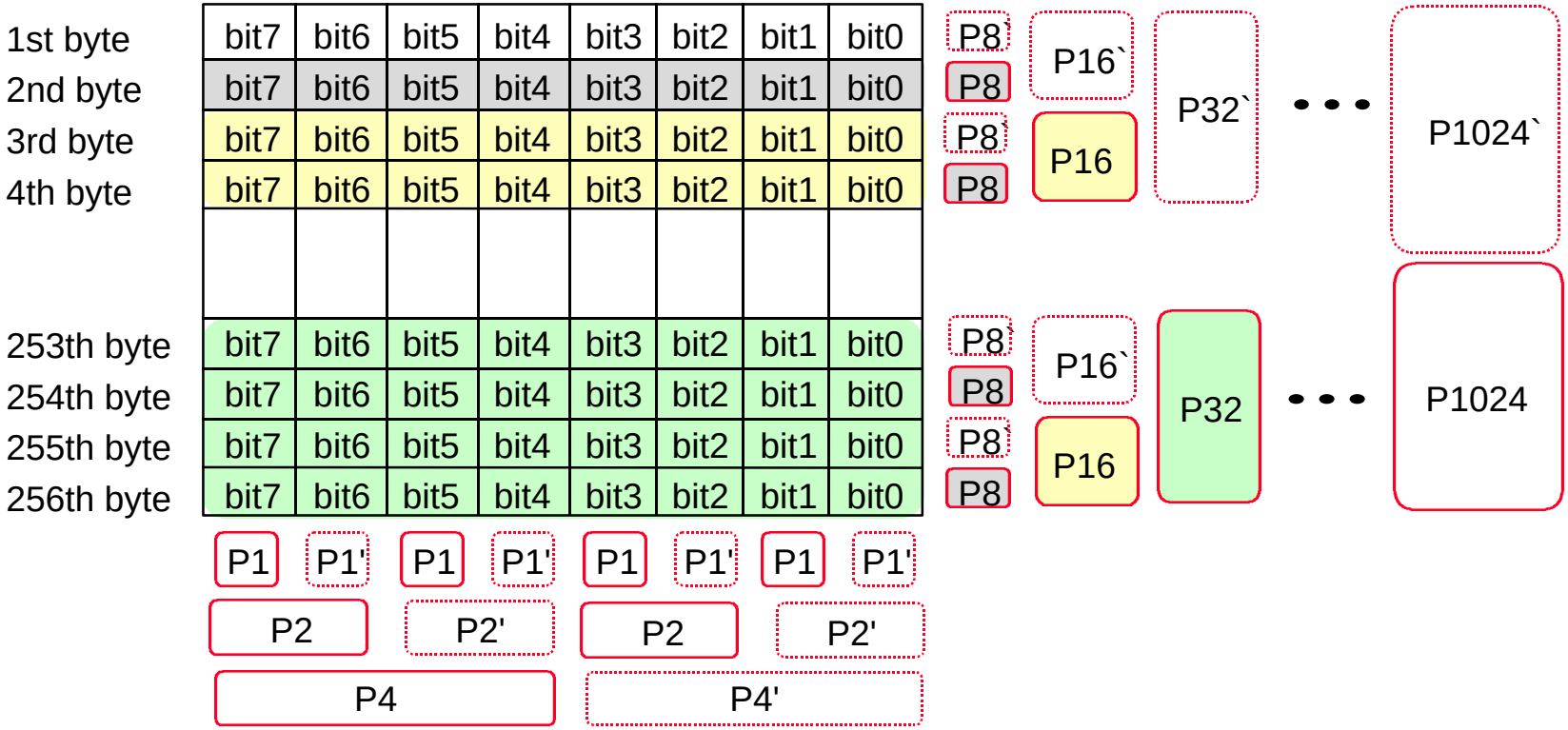
♦ **Parity Generation (In case of 256 byte input)**



P1=bit7(+)bit5(+)bit3(+)bit1(+)P1
P2=bit7(+)bit6(+)bit3(+)bit2(+)P2
P4=bit7(+)bit6(+)bit5(+)bit4(+)P4

ECC Code Generation Methode (2)

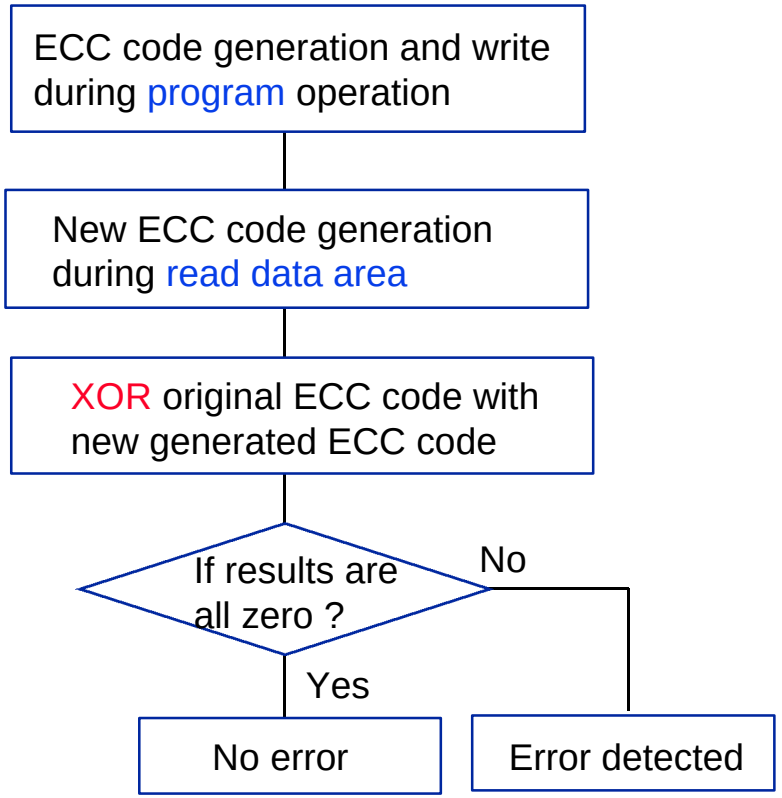
♦ **Parity Generation (In case of 256 byte input)**

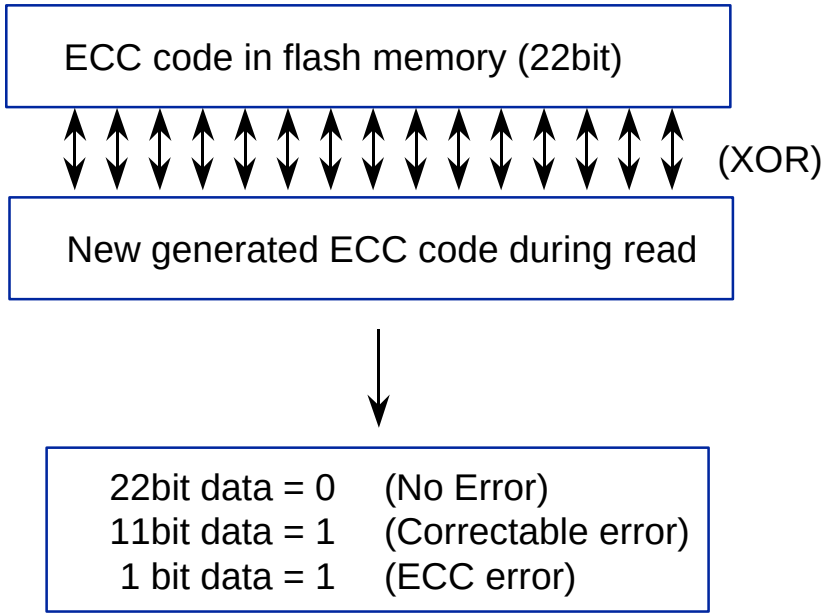


P8 = bit7(+)bit6(+)bit5(+)bit4(+)bit3(+)bit2(+)bit1(+)bit0(+)P8

Error Detection Methode

♦ **Error Detecton Sequence**





Error Detect Result

♦ **No Error**

- The result of XOR : all ECC code is `0`

♦ **Correctable Error**

- The result of XOR : total 11bits are `1`
- When main area have 1 bit error, each parity pair (ex. P8 & P8`) have 1 & 0 or 0 & 1.

♦ **ECC Error**

- The result of XOR : only 1bit is `1`
- When ECC area have an error, call it ECC error

♦ **Uncorrectable Error**

- The result of XOR : random data
- When flash have more than 2 bits error, could not correction data.

Error Correction

- For the Correctable Error, we can correct wrong data.
- Error Correction Example

  - After excute XOR, we can get final ECC code.

  If final ECC code P1024,P512,P256, ...P8,P4,P2,P1 is 01001100 110.

  The fail data : (01001100)th data (76th data)
  The fail bit : (110)th bit (I/O6)

  If change fail bit data, it will be correct data.

ECC Algorithm Excuted Result Example

Orginal Data : 123456789 . . . . ECC code : 01000011 001

Changed Data : 123457789 . . . . ECC code : 01000110 001

ECC code : 00000101 000 (5h 0h)

=>Fail bit is *6th byte 1st bit*

	P1024	P1024`	P512	P512`	P256	P256`	P128	P128`	P64	P64`	P32	P32`	P16	P16`	P8	P8`	P4	P4`	P2	P2`	P1`	P1
Orginal ECC	**0**	0	**1**	1	**0**	0	**0**	0	**0**	0	**0**	0	**1**	1	**1**	1	**0**	0	**0**	0	**1**	1
New ECC	**0**	1	**1**	0	**0**	1	**0**	1	**0**	1	**1**	0	**1**	0	**0**	1	**0**	1	**0**	1	**1**	0
Result of XOR	**0**	1	**0**	1	**0**	1	**0**	1	**0**	1	**1**	0	**0**	1	**1**	0	**0**	1	**0**	1	**1**	0

- The result data of XOR , each pair have alternative data that means correctable error.
- Because we know fail bit address, can correct fail bit to original data.