Another Method is **Syndrome word Calculation:**

Table 5.2 Increase in Word Length with Error Correction

	Single-Error Correction		Single-Error Correction/ Double-Error Detection	
Data Bits	**Check Bits**	**% Increase**	**Check Bits**	**% Increase**
8	4	50	5	62.5
16	5	31.25	6	37.5
32	6	18.75	7	21.875
64	7	10.94	8	12.5
128	8	6.25	9	7.03
256	9	3.52	10	3.91

To generate a 4-bit syndrome for an 8-bit data word with the following characteristics:

- If the syndrome **contains all 0s**, **no error** has been detected.
- If the syndrome **contains one and only one bit set to 1**, then an **error has occurred in one of the 4 check bits**. No correction is needed.
- If the syndrome **contains more than one bit set to 1**, then the numerical value of the **syndrome indicates the position of the data bit in error**. This data bit is inverted for correction

check bit

0000 0010 0011

Data Bit and Check Bit Positions:

$8 + 4 = 12$

- Following diagram dissipates 12 bit word… which includes check bits and data bits.
-
- The **check bits are calculated** as follows, where the **symbol** $\oplus$ **designates the exclusive-OR operation**:

$$
\begin{aligned}
C1 &= D1 \oplus D2 \oplus D4 \oplus D5 \oplus D7 \\
C2 &= D1 \oplus D3 \oplus D4 \oplus D6 \oplus D7 \\
C4 &= D2 \oplus D3 \oplus D4 \oplus D8 \\
C8 &= D5 \oplus D6 \oplus D7 \oplus D8
\end{aligned}
$$

12

C1 -> position 1: , C2-> Position 2, C4->position 4, C8->position 8,

Bit position	12	11	10	9	8	7	6	5	4	3	2	1
Position number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check bit					C8				C4		C2	C1

Figure 5.9 Layout of Data Bits and Check Bits

- **Calculating the Hamming Code**
- The key to the Hamming Code is the use of extra parity bits to allow the identification of a single error. Create the code word as follows:
- **Mark all bit positions that are powers of two as parity bits.** (positions 1, 2, 4, 8, 16, 32, 64, etc.) 128, 256 ....
- **All other bit positions are for the data to be encoded.** (positions 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, etc.)

$D7 \oplus D5 \oplus D4 \oplus D2 \oplus D1$

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

- **Each parity bit calculates the parity for some of the bits in the code word. The position of the parity bit determines the sequence of bits that it alternately checks and skips.**
- **Position 1:** check 1 bit, skip 1 bit, check 1 bit, skip 1 bit, etc. (1,3,5,7,9,11,13,15,...)
- **Position 2:** check 2 bits, skip 2 bits, check 2 bits, skip 2 bits, etc. (2,3,6,7,10,11,14,15,...)
- **Position 4:** check 4 bits, skip 4 bits, check 4 bits, skip 4 bits, etc. (4,5,6,7,12,13,14,15,20,21,22,23,...)
- **Position 8:** check 8 bits, skip 8 bits, check 8 bits, skip 8 bits, etc. (8-15,24-31,40-47,...)
- **Position 16:** check 16 bits, skip 16 bits, check 16 bits, skip 16 bits, etc. (16-31,48-63,80-95,...)

C_1 C_2 C_4 C_8 D_1 D_2 D_3 D_4 D_5 D_6 D_7 D_8

$C_1 \rightarrow D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7$

- **Position 32:** check 32 bits, skip 32 bits, check 32 bits, skip 32 bits, etc. (32-63,96-127,160-191,...) etc.
- **Set a parity bit to 1** if the total number of ones in the positions it checks is odd. **Set a parity bit to 0** if the total number of ones in the positions it checks is even.

8 bit

- Assume that the 8-bit **input word is 00111001**,
- The calculations are as follows:

$$C1 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$C2 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$C4 = 0 \oplus 0 \oplus 1 \oplus 0 = 1$$

$$C8 = 1 \oplus 1 \oplus 0 \oplus 0 = 0$$

-

Bit position	12	11	10	9	8	7	6	5	4	3	2	1
Position number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check bit					C8				C4		C2	C1

Figure 5.9 Layout of Data Bits and Check Bits

12 Bit → 8 Bit + 4 Bit

① while — writing the data

② which — Reading the data

① $C_8 C_4 C_2 C_1$ ⊕ ② $C_8 C_4 C_2 C_1$

Writing Reading

$$
\begin{aligned}
C1 &= 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1 \\
C2 &= 1 \oplus 1 \oplus 1 \oplus 1 \oplus 0 = 0 \\
C4 &= 0 \oplus 1 \oplus 1 \oplus 0 = 0 \\
C8 &= 1 \oplus 1 \oplus 0 \oplus 0 = 0
\end{aligned}
$$

When the new check bits are compared with the old check bits, the syndrome word is formed:

$$
\begin{array}{rcccc}
 & C8 & C4 & C2 & C1 \\
 & 0 & 1 & 1 & 1 \\
\oplus & 0 & 0 & 0 & 1 \\
\hline
 & 0 & 1 & 1 & 0
\end{array}
$$

The result is 0110, indicating that bit position 6, which contains data bit 3, is in error.

Figure 5.10 illustrates the preceding calculation. The data and check bits are positioned properly in the 12-bit word. Four of the data bits have a value 1 (shaded in the

Bit position	12	11	10	9	8	7	6	5	4	3	2	1
Position number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check bit					C8				C4		C2	C1
Word stored as	0	0	1	1	0	1	0	0	1	1	1	1
Word fetched as	0	0	1	1	0	1	1	0	1	1	1	1
Position number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Check bit					0				0		0	1

Figure 5.10 Check Bit Calculation

D_8	D_7	D_6	D_5	C_8	D_4	D_3	D_2	C_4	D_1	C_2	C_1
12	11	10	9	8	7	6	5	4	3	2	1

$C_1 \quad D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7$

$C_2 \quad D_1 \oplus D_3 \oplus D_4 \oplus D_6 \oplus D_7$

$C_4 \quad D_2 \oplus D_3 \oplus D_4 \oplus D_8$

$C_8 \longleftarrow D_5 \oplus D_6 \oplus D_7 \oplus D_8$

Data Bit = 00111001

	12	11	10	9	8	7	6	5	4	3	2	1
	0	0	1	1	C_8	1	0	0	C_4	1	C_2	C_1

C_1 : $D_7 \oplus D_5 \oplus D_4 \oplus D_2 \oplus D_1$

C_2 : $D_7 \oplus D_6 \oplus D_4 \oplus D_3 \oplus D_1$

C_4 : $D_8 \oplus D_4 \oplus D_3 \oplus D_2$

C_8 : $D_8 \oplus D_7 \oplus D_6 \oplus D_5$

Data Bit → Reading — Data — 8 — 4

writing — 1

While writing → $C_8\ C_4\ C_2\ C_1$ (1) →

While Reading → $C_8\ C_4\ C_2\ C_1$ (2) →

$C_8 \oplus C_8 \quad C_4 \oplus C_4 \quad C_2 \oplus C_2 \quad C_1 \oplus C_1$

$S_8\ S_4\ S_2\ S_1$ → !

Syndrom

1 0 0 0 0

2 0 0 1 0 → Check bit

3 0 1 1 0 → data Bit

(labels above the two 1s: 1, 2)

→ Data Bit error

→ Position of the data bit

Data Bit : 00111001

$D_8' \; D_7' \; D_6' \; D_5' \; C_8 \; D_4' \; D_3' \; D_2' \; C_4 \; D_1' \; C_2 \; C_1$

$0 \; 0 \; 1 \; 1 \; ? \; 1 \; 0 \; 0 \; ? \; 1 \; ? \; ?$

$$C_1 \quad D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7$$

$$C_2 \quad D_1 \oplus D_3 \oplus D_4 \oplus D_6 \oplus D_7$$

$$C_4 \quad D_2 \oplus D_3 \oplus D_4 \oplus D_8$$

$$C_8 \quad D_5 \oplus D_6 \oplus D_7 \oplus D_8$$

$$c_1 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$c_2 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$$

$$c_4 = 0 \oplus 0 \oplus 1 \oplus 0 = 1$$

$$c_8 = 1 \oplus 1 \oplus 0 \oplus 0 = 0$$

Writing the data =

$c_8 \; c_4 \; c_2 \; c_1$

$0 \;\; 1 \;\; 1 \;\; 1$

D_8 D_7 D_6 D_5 C_8 D_4 D_3 D_2 C_4 D_1 C_2 C_1

0 0 1 1 0 1 0 0 1 1 1 1

Memory

00 111001

0111

+

→ 12 Bit

Data – while Reading

0 0 1 1 0 0 0 1 →

C_8 C_4 C_2 C_1 0 C_8 C_4 C_2 C_1

Syndrom word

4 3 2 1

0 0 1 1 0 0 0 1 – ω

00110001 — R

$C_1 \rightarrow$

$C_2 \rightarrow$

$C_4 \rightarrow$

$C_8 \rightarrow$

0111

Syndrome

$$
\begin{array}{r}
0\;1\;1\;1 \\
0\;0\;0\;1 \;\oplus \\
\hline
0\;1\;1\;0 \\
\hline
\end{array}
\longrightarrow 6 \quad X \quad 7
$$

$$
R \longrightarrow
\begin{array}{r}
0\;1\;1\;1 \\
0\;0\;0\;0 \;\oplus \\
\hline
0\;1\;1\;1
\end{array}
\longrightarrow
$$

Sy $\longleftrightarrow$