CS 8351 DIGITAL PRINCIPLES AND SYSTEM DESIGN

UNIT – V : MEMORY AND PROGRAMMABLE LOGIC

ERROR DETECTION AND CORRECTION

The dynamic physical interaction of the electrical signals affecting the data pam of a memory unit may cause occasional errors in storing and retrieving the binary information. The reliability of a memory unit may be improved by employing error-detecting and error- correcting codes. The most common error detection scheme is the parity bit. A parity bit is generated and stored along with the data word in memory. The parity of the word is checked after reading it from memory. The data word is accepted if the parity of the bits read out is correct. If the parity checked results in an inversion, an error is detected, but it cannot be corrected.

An error-correcting code generates multiple parity check bits that are stored with the data word in memory. Each check bit is parity over a group of bits in the data word. When the word is read back from memory; the associated parity bits are also read from memory and compared with a new set of check bits generated from the data that have been read. If the check bits are correct, no error has occurred. If the check bits do not match the stored parity they generate a unique pattern called a syndrome that can be used to identify the bit that is in error. A single error occurs when a bit changes in value from 1 to 0 or from 0 to 1 during the write or read operation. If the specific bit in error is identified, then the error can be corrected by complementing the erroneous bit.

Hamming Code

One of the most common error-correcting codes used in RAMs was devised by R. W. Hamming. In the Hamming code, k parity bits are added to an n-bit data word, forming a new word of n + k bits. The hit positions are numbered in sequence from 1 to n + k. These positions numbered as a power of 2 are reserved for the parity bits. The remaining bits are the data bits. The code can be used with words of any length.

Consider, for example, the 8-bit data word 11000100. We include 4 parity bits with the 8-bit word and arrange the 12 bits as follows:

Bit position:	1	2	3	4	5	6	7	8	9	10	11	12
	P_1	P_2	1	P_4	1	0	0	P_8	0	1	0	0

The 4 parity bits, P1, P2,P3 and P4 are in positions 1, 2, 4 and 8 respectively. The 8 bits of the data word are in the remaining positions. Each parity bit is calculated as follows:

P_1 = XOR of bits (3, 5, 7, 9, 11) = $1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 = 0$

P_2 = XOR of bits (3, 6, 7, 10, 11) = $1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 = 0$

P_4 = XOR of bits (5, 6, 7, 12) = $1 \oplus 0 \oplus 0 \oplus 0 = 1$

P_8 = XOR of bits (9, 10, 11, 12) = $0 \oplus 1 \oplus 0 \oplus 0 = 1$

The 8-bit data word is stored in memory together with the 4 parity bits as a 12-bit compos ite word. Substituting the 4 P bits in their proper positions, we obtain the 12-bit composite word stored in memory:

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

When the 12 bits arc read from memory, they are checked again for errors. The parity is checked over the same combination of bits, including the parity bit. The 4 check bits are evaluated as follows:

C_1 = XOR of bits (1, 3, 5, 7, 9, 11)
C_2 = XOR of bits (2, 3, 6, 7, 10, 11)
C_4 = XOR of bits (4, 5, 6, 7, 12)
C_8 = XOR of bits (8, 9, 10, 11, 12)

A 0 check bit de signates e ven parity over the checked bits and a 1 designates odd parity. Since the bits were stored with even parity, the result, C= $C_8C_4C_2C_1$ = 0000, indicates that no error has occurred. However, if C ≠0, then the 4 -bit binary number formed by the check bits gives the position of the error bit. For example, consider the following three cases:

Bit position:	1	2	3	4	5	6	7	8	9	10	11	12	
	0	0	1	1	1	0	0	1	0	1	0	0	No error
	1	0	1	1	1	0	0	1	0	1	0	0	Error in bit 1
	0	0	1	1	0	0	0	1	0	1	0	0	Error in bit 5

In the first case, there is no error in the l2·bit word. In the second case, there is an error in bit position number I because it changed from 0 to I. The third case shows an error in bit position 5, with a change from 1 to 0. Evaluating the XQR of the corresponding bits, we determine the 4 check bits to be as follows:

	C_8	C_4	C_2	C_1
For no error:	0	0	0	0
With error in bit 1:	0	0	0	1
With error in bit 5:	0	1	0	1

Thus, for no error, we have C = 0000; with an error in bit 1, we obtain C = 0001; and with an error in bit 5, we get C = 0101. When the binary number C is not equal to 0000, it gives the position of the bit in error. The error can be corrected by complementing the corresponding bit. Note that an error can occur in the data word or in one of the parity bits.