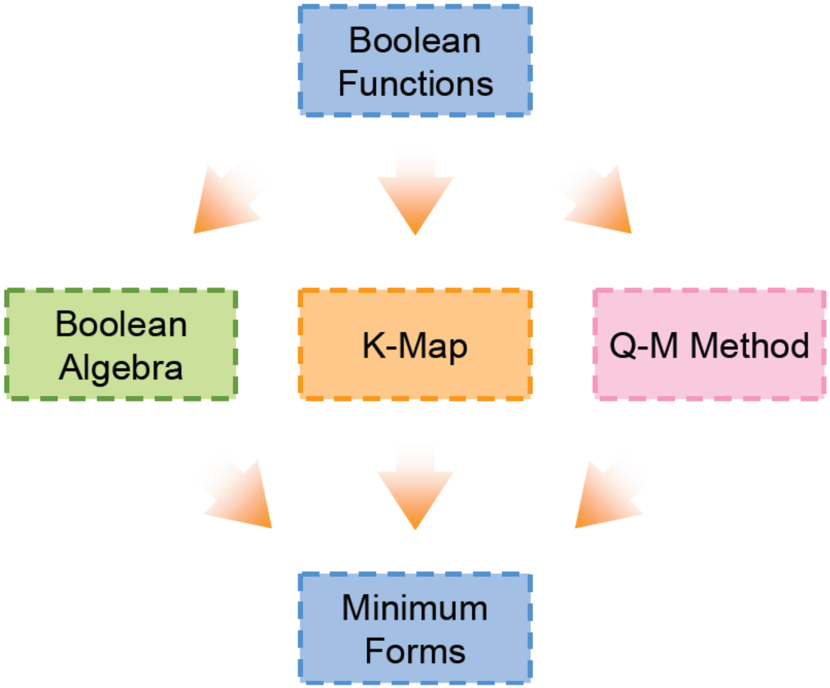
EE2000 Logic Circuit Design

Lecture 3 – Combinational System Design



What will you learn?

3.1 Learn Binary Coded Decimals

3.2 Learn the design procedure of combinational system with examples

3.3 Identify a timing hazard of a combinational system and learn to solve the problem

3.4 Learn various schemes of error detection and correction for binary data transmission

3.4 Error Detection and Correction

- Data is transmitted in the form of binary bits (1 or 0)
- Noise might cause an error in the transmitted data (0 to 1 or 1 to 0)
- Error detection codes
  - Constant-weight code, e.g. 2-of-5 code
  - Parity bit
  - Hamming code

Constant-weight Codes (***m***-of-***n*** Code)

- A separable error detection code with a code word length of ***n*** bits, whereby each code word has exactly ***m*** instances of a "one"

Decimal numbers	**3-of-6 code**	
	3 data bits	Appended bits
0	000	111
1	001	110
2	010	110
3	011	100
4	100	110
5	101	100
6	110	100
7	111	000

- 3-of-6 code: 6 bits with 3 "1"s
- Can detect certain errors but not all (Single bit error)
- E.g. Original: 011100

1) 011100 -> Correct
2) 011101 -> Error detected
3) 011000 -> Error detected
4) 101100 -> Incorrect

Parity Code

- The simplest method for error detection is using parity bit
  - An additional bit (LSB) attaches to the original code
  - Two kinds of party bit (even or odd parities)

- The value of parity bit is defined by the total no. of "1"s in the resulting codeword either even or odd

Example of Parity Code

Decimal numbers	Binary code	Number of ‘1’	Even Parity Bit	Even Parity Code	Odd Parity Bit	Odd Parity Code
0	0000	0	0	00000	1	00001
1	0001	1	1	00011	0	00010
2	0010	1	1	00101	0	00100
3	0011	2	0	00110	1	00111
4	0100	1	1	01001	0	01000
5	0101	2	0	01010	1	01011
6	0110	2	0	01100	1	01101
7	0111	3	1	01111	0	01110
8	1000	1	1	10001	0	10000
9	1001	2	0	10010	1	10011
10	1010	2	0	10100	1	10101
11	1011	3	1	10111	0	10110
12	1100	2	0	11000	1	11001
13	1101	3	1	11011	0	11010
14	1110	3	1	11101	0	11100
15	1111	4	0	11110	1	11111

Hamming Code

- Insert extra bit at specific positions to enable error detection and correction

Step 1: Calculate extra bit (***k***) needed for a ***n*** bit of code.

$$2^k \geq n + k + 1$$

For a 4-bit data $d_4d_3d_2d_1$, *n* = 4

$$2^k \geq 5 + k$$

Therefore, minimum value of *k* is 3. We need **3 parity bits**!

Hamming Code

Step 2: Place Parity Bits in the positions of powers of 2.

$$d_4 d_3 d_2 \overset{2^2}{p_3} d_1 \underset{2^1}{p_2} \overset{2^0}{p_1}$$

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Bit	7	6	5	**4**	3	**2**	**1**

Hamming Code

Step 3: Calculate each parity bits based on odd or even parity.

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Bit	7	6	5	4	3	2	1
Binary Code	111	110	101	100	011	010	001
p_1	d_4		d_2		d_1		
p_2	d_4	d_3			d_1		
p_3	d_4	d_3	d_2				

p_1: Include all data bits in positions whose binary representation includes a 1 in the least significant position excluding Bit 1.

p_2: Include all data bits in positions whose binary representation includes a 1 in the position 2 from right excluding Bit 2.

p_3: Include all data bits in positions whose binary representation includes a 1 in the position 3 from right excluding Bit 4.

Hamming Code

Example: data $d_4d_3d_2d_1$ = 1000

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Bit	7	6	5	4	3	2	1
Binary Code	111	110	101	100	011	010	001
p_1	1		0		0		
p_2	1	0			0		
p_3	1	0	0				
Even Parity	1	0	0		0		
Odd Parity	1	0	0		0		

Even Parity

$$p_1 = H_7 \oplus H_5 \oplus H_3 = 1 \oplus 0 \oplus 0 = 1$$
$$p_2 = H_7 \oplus H_6 \oplus H_3 = 1 \oplus 0 \oplus 0 = 1$$
$$p_3 = H_7 \oplus H_6 \oplus H_5 = 1 \oplus 0 \oplus 0 = 1$$

Odd Parity

$$p_1 = (H_7 \oplus H_5 \oplus H_3)' = (1 \oplus 0 \oplus 0)' = 0$$
$$p_2 = (H_7 \oplus H_6 \oplus H_3)' = (1 \oplus 0 \oplus 0)' = 0$$
$$p_3 = (H_7 \oplus H_6 \oplus H_5)' = (1 \oplus 0 \oplus 0)' = 0$$

Hamming Code

Example: data $d_4d_3d_2d_1$ = 1000

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Bit	7	6	5	4	3	2	1
Binary Code	111	110	101	100	011	010	001
p_1	1		0		0		
p_2	1	0			0		
p_3	1	0	0				
Even Parity	1	0	0	**1**	0	**1**	**1**
Odd Parity	1	0	0	**0**	0	**0**	**0**

Even Parity

$$p_1 = H_7 \oplus H_5 \oplus H_3 = 1 \oplus 0 \oplus 0 = 1$$

$$p_2 = H_7 \oplus H_6 \oplus H_3 = 1 \oplus 0 \oplus 0 = 1$$

$$p_3 = H_7 \oplus H_6 \oplus H_5 = 1 \oplus 0 \oplus 0 = 1$$

Odd Parity

$$p_1 = (H_7 \oplus H_5 \oplus H_3)' = (1 \oplus 0 \oplus 0)' = 0$$

$$p_2 = (H_7 \oplus H_6 \oplus H_3)' = (1 \oplus 0 \oplus 0)' = 0$$

$$p_3 = (H_7 \oplus H_6 \oplus H_5)' = (1 \oplus 0 \oplus 0)' = 0$$

Exercise

Determine the Hamming code using both odd and even parity bit for a data code of 11100

Step 1: Calculate extra bit (***k***) needed for a ***n*** bit of code.

Step 2: Place Parity Bits in the positions of powers of 2.

Hamming Code	H_9	H_8	H_7	H_6	H_5	H_4	H_3	H_2	H_1
Bit	9	8	7	6	5	4	3	2	1

Exercise

Step 2: Place Parity Bits in the positions of powers of 2.

data code: 11100

Hamming Code	H_9	H_8	H_7	H_6	H_5	H_4	H_3	H_2	H_1
Bit									
Binary Code									
p_1									
p_2									
p_3									
p_4									
Even Parity									
Odd Parity									

Step 3: Calculate the number of '1' in each parity bits

Step 4: Place '1' if odd number of '1' for even parity; else '0'; Place '0' if odd number of '1' for odd parity; else '1'

Error Detection and Correction

Example: data $d_4d_3d_2d_1$ = 1000

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Bit	7	6	5	4	3	2	1
Binary Code	111	110	101	100	011	010	001
p_1	1		0		0		
p_2	1	0			0		
p_3	1	0	0				
Even Parity	1	0	0	1	0	1	1
Odd Parity	1	0	0	0	0	0	0

Consider even parity and if we receive a code of 1001111, check the parity bits

$$c_1 = H_7 \oplus H_5 \oplus H_3 \oplus H_1 = 1 \oplus 0 \oplus 1 \oplus 1 = 1$$
$$c_2 = H_7 \oplus H_6 \oplus H_3 \oplus H_2 = 1 \oplus 0 \oplus 1 \oplus 1 = 1$$
$$c_3 = H_7 \oplus H_6 \oplus H_5 \oplus H_4 = 1 \oplus 0 \oplus 0 \oplus 1 = 0$$

$$c_3c_2c_1 = (011)_2 = 3$$

Error Detection and Correction

Example: data $d_4d_3d_2d_1$ = 1000

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Bit	7	6	5	4	3	2	1
Binary Code	111	110	101	100	011	010	001
p_1	1		0		0		
p_2	1	0			0		
p_3	1	0	0				
Even Parity	1	0	0	1	0	1	1
Odd Parity	1	0	0	0	0	0	0

Consider even parity and if we receive a code of 1001111, check the parity bits

$$c_1 = (H_7, H_5, H_3, H_1) = (1, 0, 1, 1) = 1$$
$$c_2 = (H_7, H_6, H_3, H_2) = (1, 0, 1, 1) = 1$$
$$c_3 = (H_7, H_6, H_5, H_4) = (1, 0, 0, 1) = 0$$

$$c_3c_2c_1 = (011)_2 = 3$$

Exercise

Example: data $d_4d_3d_2d_1$ = 1000

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Bit	7	6	5	4	3	2	1
Binary Code	111	110	101	100	011	010	001
p_1	1		0		0		
p_2	1	0			0		
p_3	1	0	0				
Even Parity	1	0	0	1	0	1	1
Odd Parity	1	0	0	0	0	0	0

Consider odd parity and if we receive a code of 1001000, check the parity bits

c_1

c_2

c_3

$c_3c_2c_1 =$

Limitation?

Example: data $d_4d_3d_2d_1$ = 1000

Hamming Code	H_7	H_6	H_5	H_4	H_3	H_2	H_1
	d_4	d_3	d_2	p_3	d_1	p_2	p_1
Odd Parity	1	0	0	0	0	0	0

If we receive a code of 1001100, check the parity bits

$$c_3c_2c_1 = (111)_2 = 7?$$

If we receive a code of 1011100, check the parity bits

$$c_3c_2c_1 = (010)_2 = 2?$$