CS 225: Switching Theory

S. Tripathy IIT Patna

Previous Class

- Number Systems and Codes
 - Different Number systems (positional)
 - Conversion
 - Representation (complement)
 - Binary Arithmetic

This Class

Number Systems and Codes

- Codes
 - BCD, cyclic code etc.
 - Gray code
 - Parity and Error correcting code

Answer the following

```
(1010100)_2 - (1000011)_2 =? Use 2'complement method
                 Ans.: 10010001 =0010001 (17=84-67)
(1000011)_2 - (1010100)_2 =? Use 2'complement method
               Ans.: _{1101111} = -0010001(-17=67-84)
1100101_2 \times 111101_2 = ?
               Ans.: 1100000010001 (101 x61 =6161)
100101_2 \div 101_2 = ?
               Ans.: Q= 111, rem= 10 (37_{10} \div 5_{10} = Q(7), rem(2))
```

Binary Coded Decimal (BCD)

To code a number with n decimal digits, we need 4n bits in BCD e.g. $(365)_{10}$ = $(0011\ 0110\ 0101)_{BCD}$

This is different to converting to binary, which is $(365)_{10} = (101101101)_2$

- Use 4-bit binary to represent one decimal digit
- Easy conversion
- Wasting bits (4-bits can represent 16 different values, but only 10 values are used). Clearly, BCD requires more bits. BUT, it is easier to understand/interpret

BCD Addition

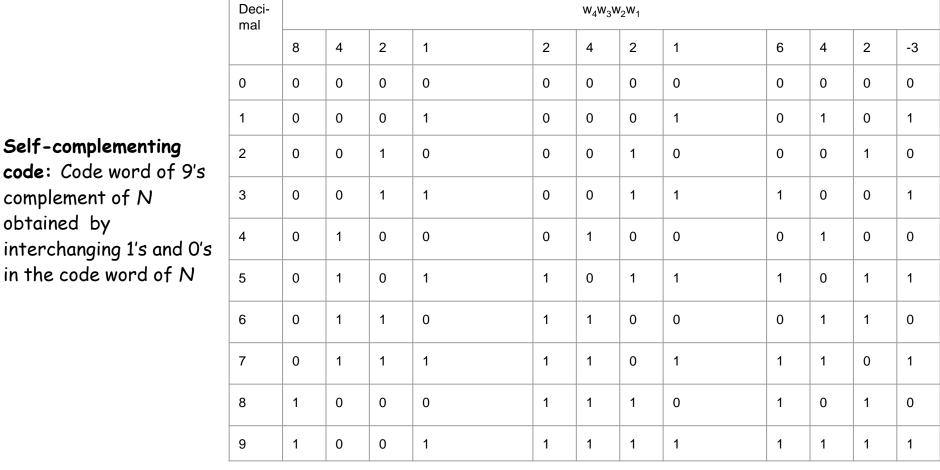
- When 2 BCD codes are added:
 - If the sum is less than 1010 (10 in decimal), the corresponding BCD sum digit is correct
 - o If the sum is equal or more than 1010 (10), must add 0110 (6 in decimal) to the corresponding BCD sum digit in order to produce the correct carry into the digit to the left
- Example: Add 448 and 489 in BCD.
 0100 0100 1000 (448 in BCD)
 0100 1000 1001 (489 in BCD)

Decimal Codes

Self-complementing

complement of N

obtained by



BCD

Self-complementing Codes

Non-weighted Codes

Decimal Digit	Excess-3	Cyclic
0	0011	0000
1	0100	0001
2	0101	0011
3	0110	0010
4	0111	0110
5	1000	1110
6	1001	1010
7	1010	1000
8	1011	1100
9	1100	0100

Add 3 to BCD

Successive code words Differ in only one digit

Gray Code

Decimal number	Gray	Binary			
	g3 g2 g1 g0	b3 b2 b1 b0			
0	0 0 0 0	0 0 0 0			
1	0 0 0 1	0 0 0 1			
2	0 0 1 1	0 0 1 0			
3	0 0 1 0	0 0 1 1			
4	0 1 1 0	0 1 0 0			
5	0 1 1 1	0 1 0 1			
6	0 1 0 1	0 1 1 0			
7	0 1 0 0	0 1 1 1			

Decimal number		Binary						
	g3	g2	g1	g0	b3	b2	b1	b0
8	1	1	0	0	1	0	0	0
9	1	1	0	1	1	0	0	1
10	1	1	1	1	1	0	1	0
11	1	1	1	0	1	0	1	1
12	1	0	1	0	1	1	0	0
13	1	0	1	1	1	1	0	1
14	1	0	0	1	1	1	1	0
15	1	0	0	0	1	1	1	1

Conversion

Binary to Gray:

Start from right side LSB as: gi = bi+bi+1, gn = bn

Gray to Binary:

Start from left side MSB as: bn = gn and bi-1 = bi + gi-1

Convert

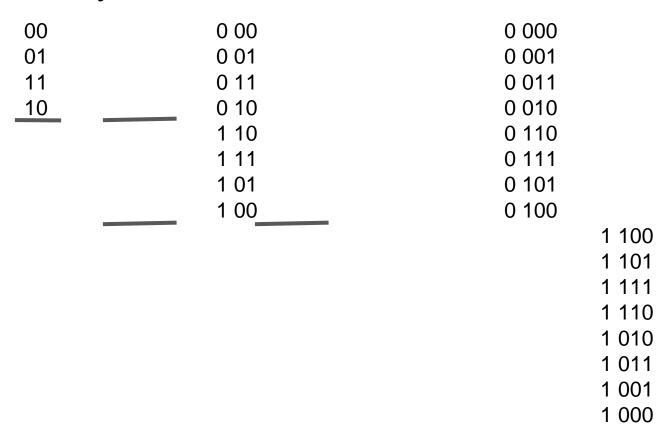
• Binary (1001) to gray

1 1 0 1

• Gray (1 1 0 0) to binary

1 0 0 0

Reflection of Gray Codes



Error-detecting Codes

p: parity bit;

Even parity used in codes.

Distance between codewords: no. of bits they differ in

Minimum distance of a code: smallest no. of bits in which any two code words differ

Minimum distance of above single errordetecting codes = 2

Decimal Digit	Even-parity BCD				2-out-of-5			;		
	8	4	2	1	р	0	1	2	4	7
0	0	0	0	0	0	0	0	0	1	1
1	0	0	0	1	1	1	1	0	0	0
2	0	0	1	0	1	0	1	1	0	0
3	0	0	1	1	0	0	1	1	0	0
4	0	1	0	0	1	1	0	0	1	0
5	0	1	0	1	0	0	1	0	1	0
6	0	1	1	0	0	0	0	1	1	0
7	0	1	1	1	1	1	0	0	0	1
8	1	0	0	0	1	0	1	0	0	1
9	1	0	0	1	0	0	0	1	0	1

Hamming Codes: Single Error-correcting

Minimum distance for SEC or double-error detecting (DED) codes = 3 Example: {000,111} Minimum distance for SEC and DED codes = 4

No. of information bits = m

No. of parity check bits, p1, p2, ..., pk = k No. of bits in the code word = m+k

Assign a decimal value to each of the m+k bits: from 1 to MSB to m+k to LSB

Perform k parity checks on selected bits of each code word: record results as 0 or 1

Form a binary number (called position number), c1c2...ck, with the k parity checks

Hamming Codes (Contd.)

No. of parity check bits, k, must satisfy: $2^k \ge m+k+1$

Example: if m = 4 then k = 3

Place check bits at the following locations: 1, 2, 4, ..., 2k-1

Example code word: 1100110

- Check bits: p1= 1, p2 = 1, p3 = 0
- Information bits: 0, 1, 1, 0

Hamming Code Construction

Select p_1 to establish even parity in positions: 1, 3, 5, 7

Select p_2 to establish even parity in positions: 2, 3, 6, 7

Select p_3 to establish even parity in positions: 4, 5, 6, 7

Error position	Position number						
	c1	c2	c3				
0 (no error)	0	0	0				
1	0	0	1				
2	0	1	0				
3	0	1	1				
4	1	0	0				
5	1	0	1				
6	1	1	0				
7	1	1	1				

Hamming Code Construction (Contd.)

Position:	1 p ₁	2 p ₂	3 m ₁	4 p ₃	5 m ₂	6 m ₃	7 m ₄
Original BCD message:			0		1	0	0
Parity Check in positions $1,3,5,7$ requires $p_1=1$	1		0		1	0	0
Parity Check in positions							
2,3,6,7 requires $p_2=0$	1	0	0		1	0	0
_							
Parity Check in positions 4,5,6,7 requires p ₃ =1	1	0	0	1	1	0	0
Coded message	1	0	0	1	1	0	0

Hamming Code Construction

Ex: If the original message is to be send is 0010

• The message to be send is

0 1 0 1 0 1 0

If the received message is 0 1 0 1 0 1 1

Error position is:

111 (7)

Hamming Code for BCD

Position: 1 2 3 4 5 6 7 Intended message: 1 1 0 1 0 0 1 Message received: 1 1 0 1 1 0 1 4-5-6-7 parity check: 1 1 0 1 1 0 1 c_1 =1 since parity is odd 2-3-6-7 parity check: 1 0 0 1 c_2 =0 since parity is even 4-5-6-7 parity check: 1 0 1 1 c_3 =1 since parity is odd

Decimal digit	Position							
	p ₁	p_2	m_1	p_3	m_2	m_3	m_4	
0	0	1	0	0	0	0	0	
1	1	1	0	1	0	0	1	
2	0	1	0	1	0	1	0	
3	1	0	0	0	0	1	1	
4	1	0	0	1	1	0	0	
5	0	1	0	0	1	0	1	
6	1	1	0	0	1	1	0	
7	0	0	0	1	1	1	1	
8	1	1	1	0	0	0	0	
9	0	0	1	1	0	0	1	

SEC/DED Code

Add another parity bit such that all eight bits have even parity

- Two errors occur: overall parity check satisfied, but position number indicates error double error (cannot be corrected)
- Single error occurs: overall parity check not satisfied
 - o Position no. is 0: error in last parity bit
 - Else, position no. indicates erroneous bit
- No error occurs: all parity checks indicate even parities

. Thanks