Digital Design

Lecture 2 & 3: Number systems Part 1 & 2

Number systems

General use Decimal numbers : 8956

Digits used are 0 - 9

$$8956 = 8 \times 10^3 + 9 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$$

Can be generalized to any decimal number

$$a_3a_2a_1a_0 \,.\, a_{-1}a_{-2}$$

$$= a_3 \times 10^3 + a_2 \times 10^2 + a_1 \times 10^1 + a_0 \times 10^0 + a_{-1} \times 10^{-1} + a_{-2} \times 10^{-2}$$

Number systems

Decimal number system: Base is 10 Numbers used : 0-9

Base also called radix

Binary number system : Base is 2 Numbers used : 0-1

For example: 101.11

$= 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$

$= 4 + 0 + 1 + 0.5 + 0.25$

$= 5.75$

Number systems

For Base - r system $(a_n a_{n-1} \ldots a_1 a_0 . a_{-1} a_{-2} .. a_{-m})_r$

$a_n \times r^n + a_{n-1} \times r^{n-1} \ldots\ldots a_1 \times r^1 + a_0 \times r^0 + a_{-1} \times r^{-1} + \ldots a_{-m} \times r^{-m}$

Find the decimal equivalent of

$(123.4)_8$ [Octal] $= 1 \times 8^2 + 2 \times 8^1 + 3 \times 8^0 + 4 \times 8^{-1} = 83.5$

$(B2.4)_{16}$ [Hexa decimal] $= 11 \times 16^1 + 2 \times 16^0 + 4 \times 16^{-1} = ??$

$(110101)_2$ [Binary]

$= 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = ??$

Number Base conversions

Typical conversions

Base-10 to Base-r

Convert $(49)_{10}$ to $()_2$

		Remainder	
2	49		
2	24	1	LSB
2	12	0	
2	6	0	
2	3	0	
2	1	1	
	0	1	MSB

$(110001)_2$

Number Base conversions

Typical conversions

Base-10 to Base-r

Convert $(50)_{10}$ to $()_2$

		Remainder	
2	50		
2	25	0	LSB
2	12	1	
2	6	0	
2	3	0	
2	1	1	
	0	1	MSB

$(110010)_2$

Number Base conversions

Typical conversions

Base-10 to Base-r (fraction)

Convert $(0.125)_{10}$ to $()_2$

	Integer
0.125x 2 = 0.25	0
0.25 x 2 = 0.5	0
0.5 x 2 = 1.0	1

$(0.125)_{10} = (0.001)_2$

Limited to required number of digits

Number Base conversions

Typical conversions

Base-10 to Base-r (fraction)

Convert $(0.49)_{10}$ to $()_2$

	Integer
0.49 x 2 = 0.98	0
0.98 x 2 = 1.96	1
0.96 x 2 = 1.92	1
0.92 x 2 = 1.84	1

$(0.49)_{10} = (0.011111....)_2$ Limited to required number of digits

Number Base conversions

Typical conversions

Base-r to Base-10

Convert $(110110)_2$ to $(\ )_{10}$

$$=1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

$$=1 \times 32 + 1 \times 16 + 0 \times 8 + 1 \times 4 + 1 \times 2 + 0 \times 2^0$$

$$=1 \times 32 + 1 \times 16 + 0 \times 8 + 1 \times 4 + 1 \times 2 + 0 \times 2^0$$

$$=54$$

Number systems

$(B65F)_{16}$

Express the following numbers in decimal

$(10110.0101)_2$

$(1010.1010)_2$

$(26.24)_8$

$(16.5)_{16}$

$(FAFA)_{16}$

Problems

(1) Use binary expansion to convert binary fractions into decimals

(i) $(101.1101)_2$ (ii) $(1101.0111)_2$
(iii) $(111.111)_2$ (iv) $(101.01011)_2$

(2) Convert
(i) $(13.6875)_{10}$ (ii) $(32.45)_{10}$
(iii) $(28.555)_{10}$ (iv) $(7.0202)_{10}$ into binary fraction

(3) Convert the following numbers with the indicated base to decimal :

(i) $(4310)_5$

(ii) $(198)_{12}$

(iii) $(735)_8$

(iv) $(525)_6$

Number conversions

Other conversions

Binary to octal

Octal: base 8 digits used 0 - 7

1110001010101

For Octal- 2^3, 8bit: 2^3

001 110 001 010 101

1 6 1 2 5

$(1110001010101)_2 \longrightarrow (16125)_8$

Number conversions

Other conversions

octal to binary

Octal: base 8 digits used 0-7

6373

6	3	7	3
↓	↓	↓	↓
110	011	111	011

$(6373)_8 \longrightarrow (110011111011)_2$

Number conversions

Other conversions

Binary to Hexadecimal Hexa: base 16 digits used 0-F

1110001010101 4 bit, then Hexa decimal-2^4

0001 1100 0101 0101

1 C 5 5

$(1110001010101)_2 \longrightarrow (1C55)_{16}$

Representation of Negative Numbers

- Signed Magnitude
- Diminished radix complement
- Radix complement

Representation of Negative Numbers

- Signed Magnitude

3-bit numbers	Signed magnitude
0 0 0	+0
0 0 1	+1
0 1 0	+2
0 1 1	+3
1 0 0	-0
1 0 1	-1
1 1 0	-2
1 1 1	-3

MSB indicates Sign : 0 indicates positive, 1 indicates negative

Limitations

1. Two Zeros

2. Add +2 & -1

```
  0 1 0
  1 0 1
  -----
  1 1 1
```

Complements

Diminished radix complement

Given a number N in base r having n digits (r-1)'s complement is defined as (r^n-1-N)

In case of decimal it is called 9's complement

9's complement of 865 is 10^3 -1 - 865 = 999 - 865 = 134

In case of binary it is called 1's complement for 1011

1's complement of 1011 is 2^4 - 1 - 1011 = 1111-1011 = 0100
(or you can simply use the complement ~ 1 for 0 and 0 for 1)

Complements

Decimal	S.M.	1's comp.
7	0111	0111
6	0110	0110
5	0101	0101
4	0100	0100
3	0011	0011
2	0010	0010
1	0001	0001
0	0000	0000
-0	1000	1111
-1	1001	1110
-2	1010	1101
-3	1011	1100
-4	1100	1011
-5	1101	1010
-6	1110	1001
-7	1111	1000
-8	–	–

Limitations of 1's Complement

- Two Zeros
- End-around-carry-bit addition

Add 4 & -7

```
 0100
 1000
 ----
 1100
```

Add 4 & -3

```
   0100
   1100
 -------
 1 0000
      1
 -------
   0001
```

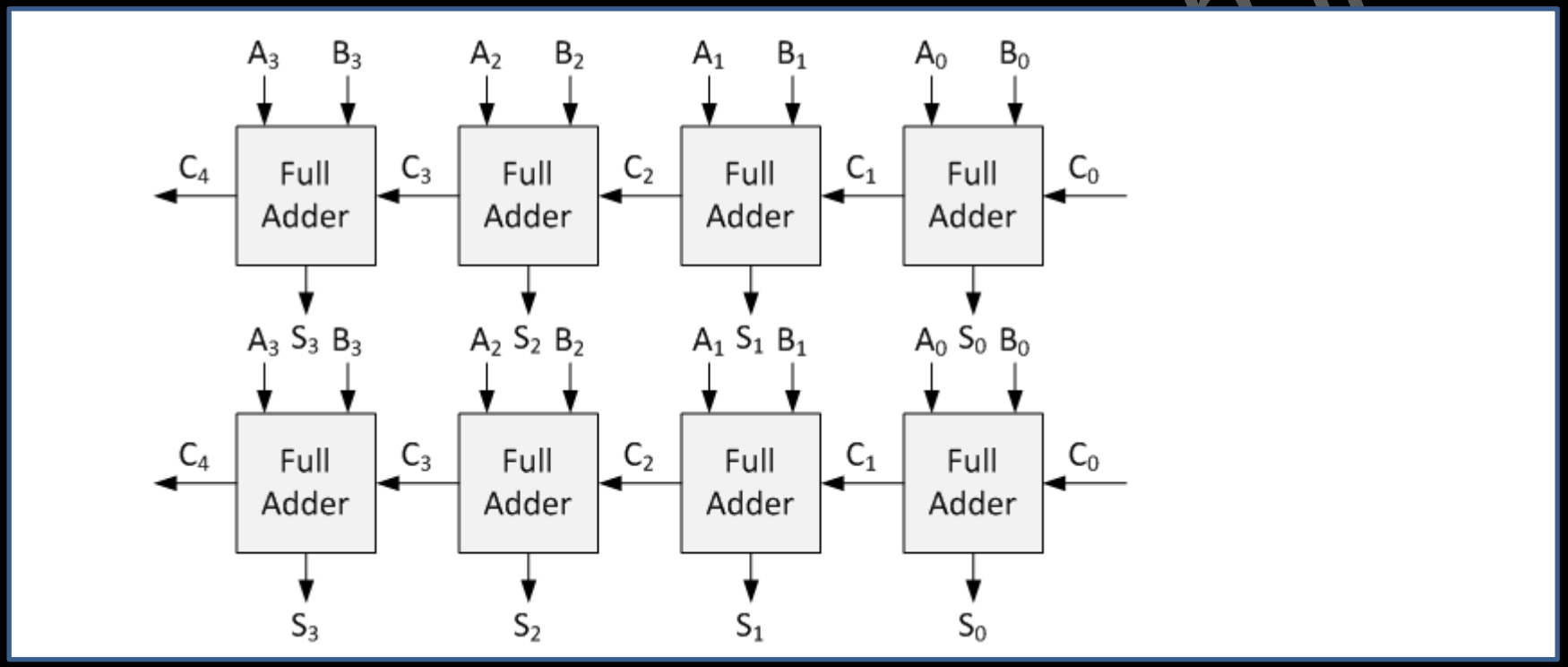
Complements

Add 4 & -3

```
  0100
  1100
 ------
1 0000
     1
 ------
  0001
```

Limitations of 1's Complement

- **Two Zeros**
- **End-around-carry-bit addition**



Complements

Radix complement

Given a number N in base r having n digits r's complement is defined as (r^n-N)

In case of decimal it is called 10's complement

10's complement of 865 is 10^3 -865 = 1000-865 = 135

10's complement = 9's complement + 1

In case of binary it is called 2's complement

2's complement of 1011 is 2^4 -1011 = 10000-1011 = 0101

2's complement = 1's complement + 1

Complements

Decimal	S.M.	1's comp.	2's comp.
7	0111	0111	0111
6	0110	0110	0110
5	0101	0101	0101
4	0100	0100	0100
3	0011	0011	0011
2	0010	0010	0010
1	0001	0001	0001
0	0000	0000	0000
-0	1000	1111	–
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	–	–	1000

Advantages of 2's Complement

- **Two Zeros**
- **No End-around-carry-bit addition**

Add 4 & -7	Add 4 & -3
0100	0100
1001	1101
1101	1 0001

Complements

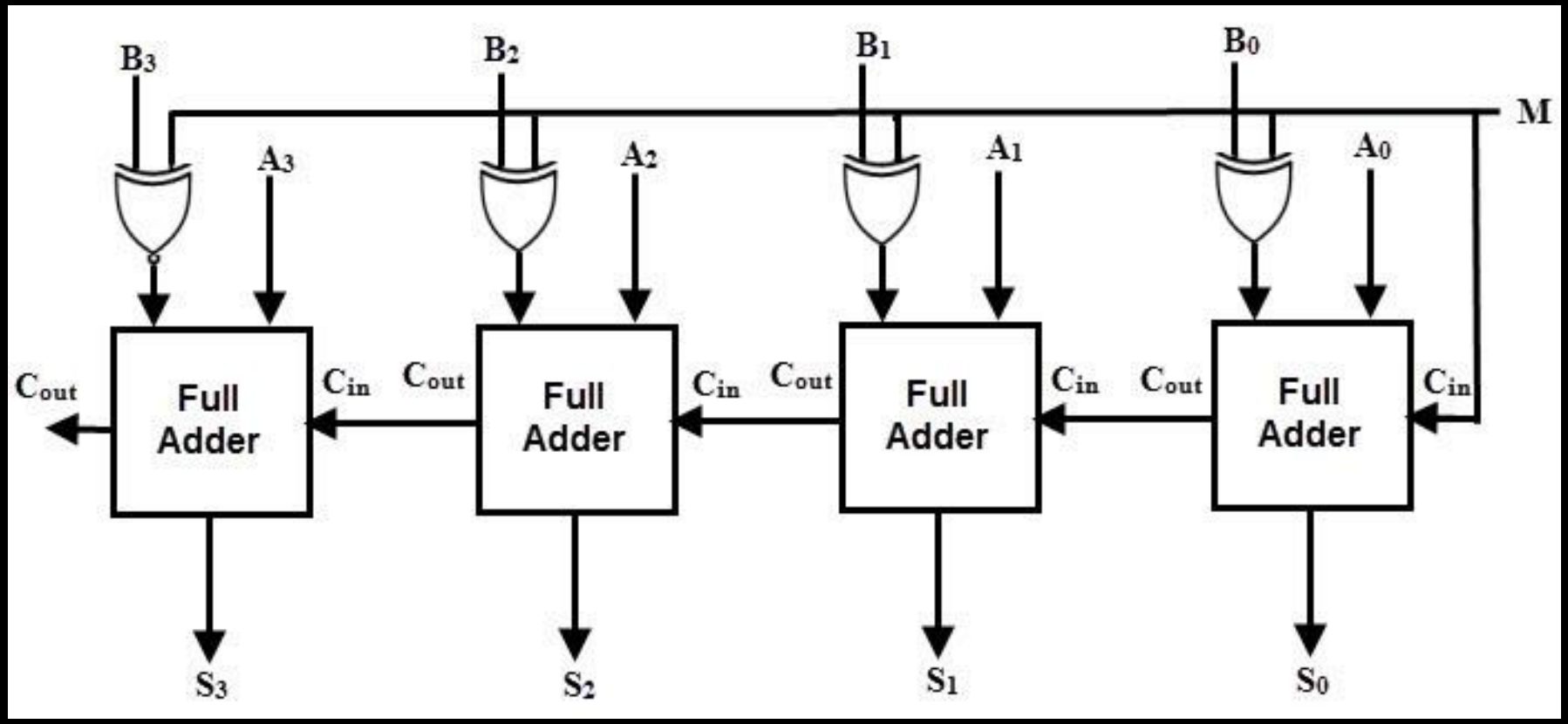
EX-OR Gate As Buffer and Inverter

Control	A	F	
0	0	0	Pass
0	1	1	
1	0	1	Invert
1	1	0	

Complements

Advantages of 2's Complement

- **Easy Implementation: Adder Subtractor M=0 adder, M=1 Subtractor**



Complements

Overflow in 2's Complement

Add 4 & -3

```
  0100
  1101
1 0001
```

Add -4 & -5

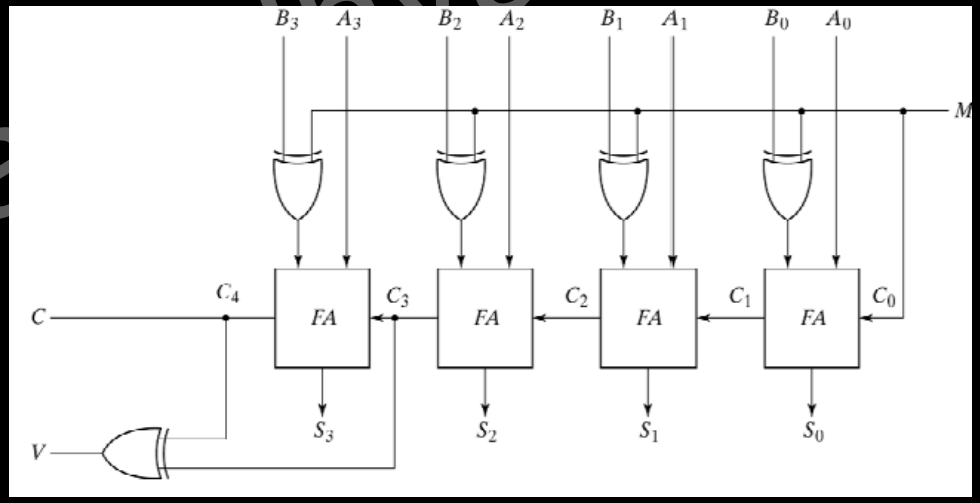
```
  1100
  1011
1 0111
```

Add -8 & 4

```
1000
0100
1100
```

Add 4 & 4

```
0100
0100
1000
```

Binary Codes - BCD

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

Consider example 7698

7 6 9 8

BCD code

BCD and Binary comparison

$(185)_{10}$

BCD = (0001 1000 0101)

Binary = $(10111001)_2$

BCD = 12 bits, Binary = 8 bits

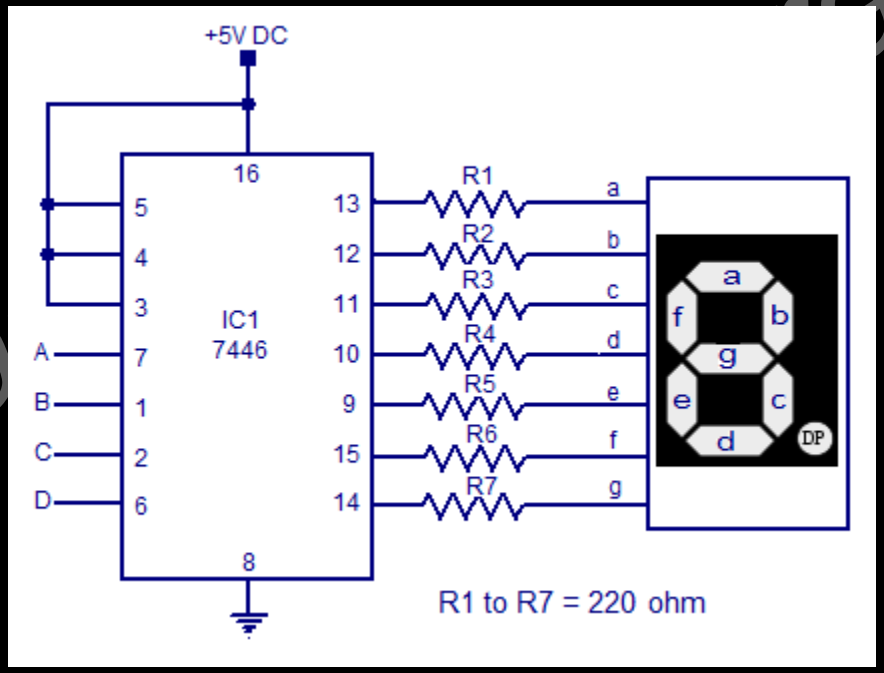
Some systems work directly on BCD (**IBM** Power6)

User enters decimal → BCD i/p→ compute in BCD → BCD o/p → Decimal output shown to user

Binary Codes - BCD

General digital systems

User enters decimal → BCD i/p→ Binary i/p → compute in binary → Binary o/p → BCD o/p → Decimal output shown to user



Binary Codes - BCD

BCD addition

4 + 5

4	0 1 0 0
5	0 1 0 1
9	1 0 0 1

Expected Result

4 + 8

4	0 1 0 0
8	1 0 0 0
	1 1 0 0

Is this expected Result ?

Expected answer is BCD of 12 0001 0010

Binary Codes - BCD

BCD addition

4 + 8

4	0 1 0 0	
8	1 0 0 0	
	1 1 0 0	Add correction of +6
	0 1 1 0	
0 0 0 1	0 0 1 0	
1	2	

Greater than 9

= To skip 6 invalid states (10 - 15) BCDs

Binary Codes - BCD

BCD addition

9 + 9

```
        9  1 0 0 1
        9  1 0 0 1
        ----------
1 0 0 1 0
  0 1 1 0
  -------
0 0 0 1 1 0 0 0
  -------
    1     8
```

Carry out generated

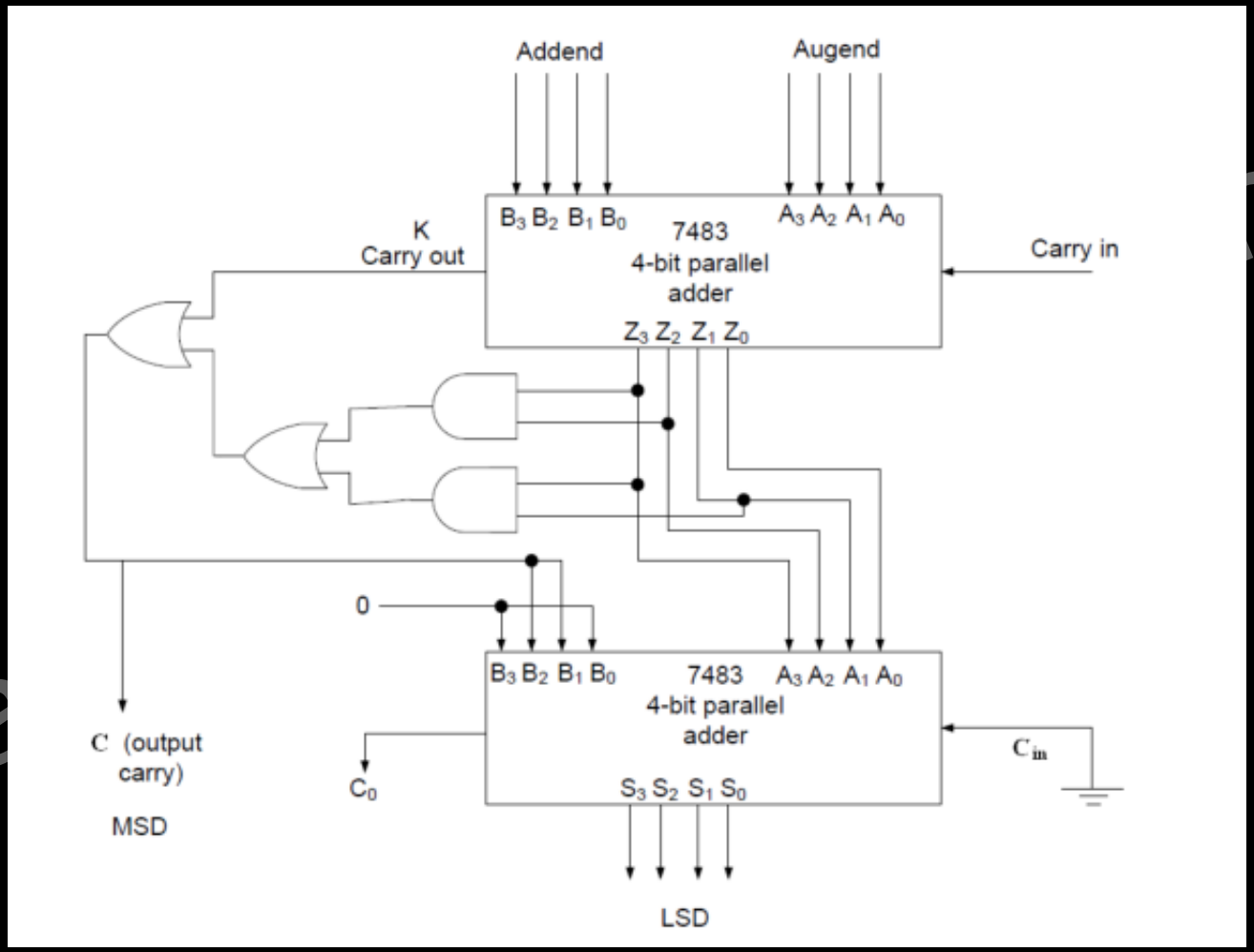
Expected result ?

Add correction of +6

After addition if carry out is generated or if sum is greater than 9 there is need for correction

Binary Codes - BCD

BCD addition



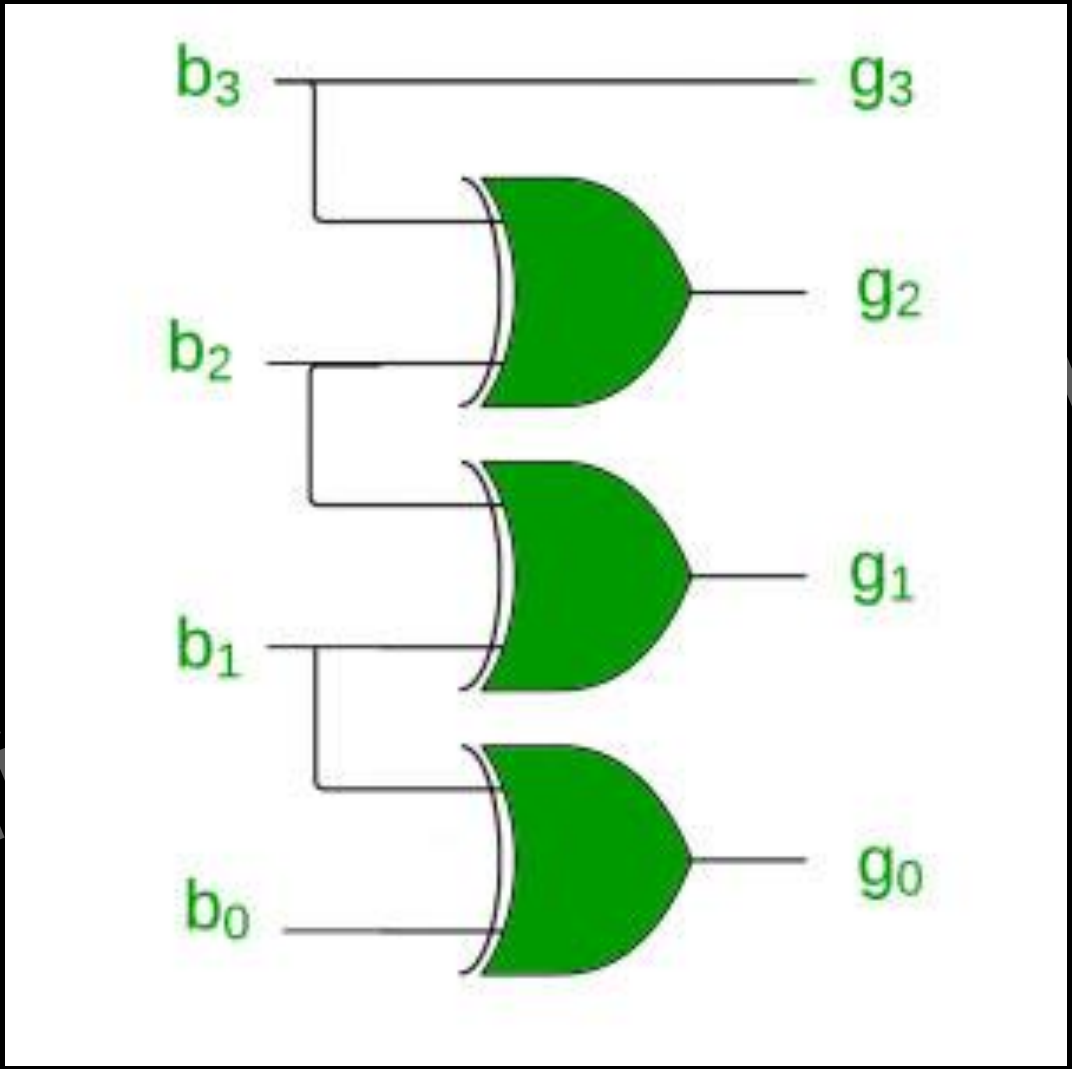
Binary Codes – Gray Code

THE GRAY CODE

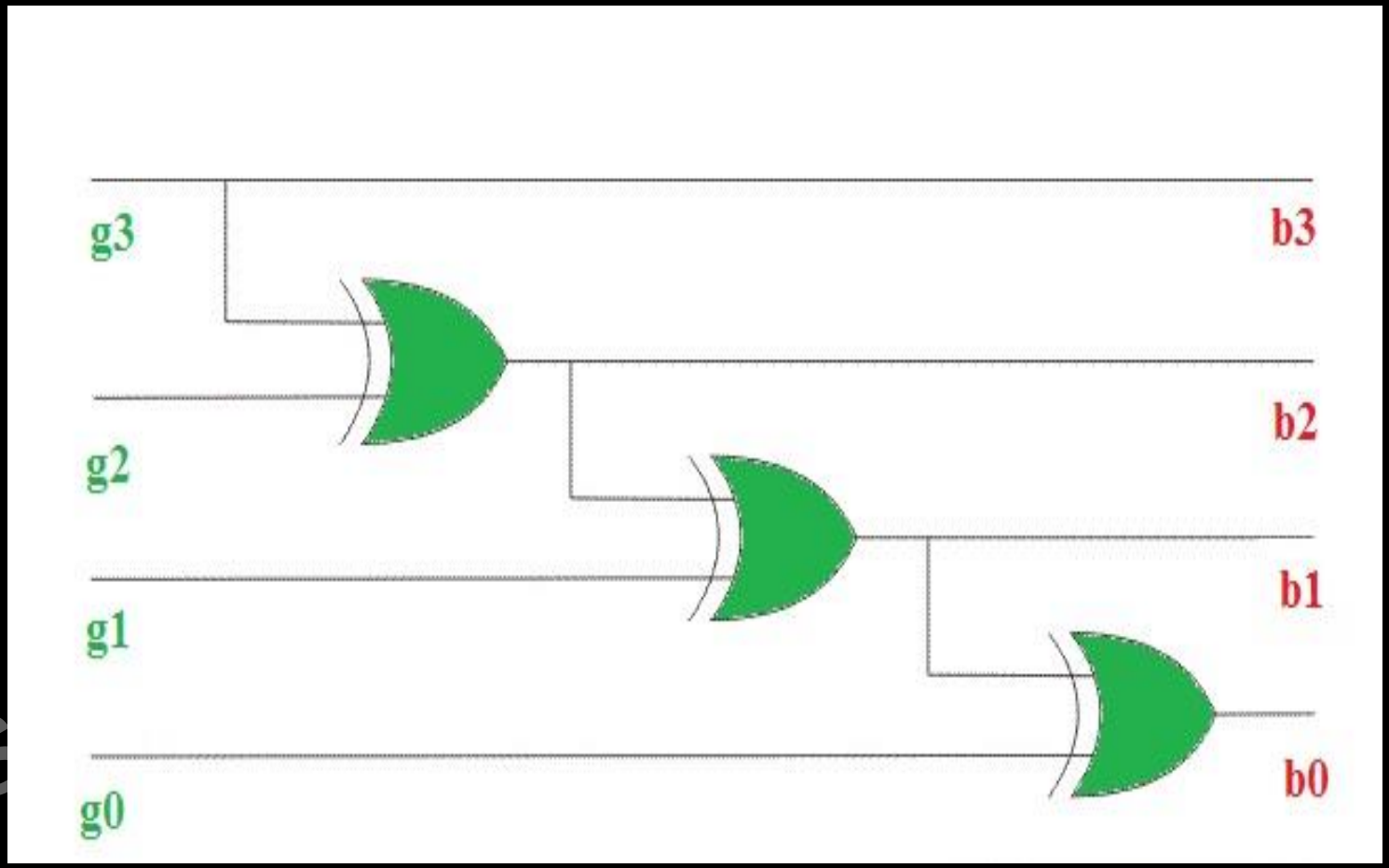
Decimal	Binary	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100

Decimal	Binary	Gray Code
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Binary Codes – Gray Code



Binary Codes – Gray Code



Thankyou