DIGITAL ELECTRONICS

✓ CHAPTER TWO: *Gray Code

*Boolean Algebra & Logic Gates (Part I)

By J. Mathenge

Previously on NUMBER SYSTEMS AND CODES

1.8.3 Gray Code

*Read on Gray Code and its advantages.

- In gray code, only one bit changes in going from one number to the next.
- It is a non-weighted code.

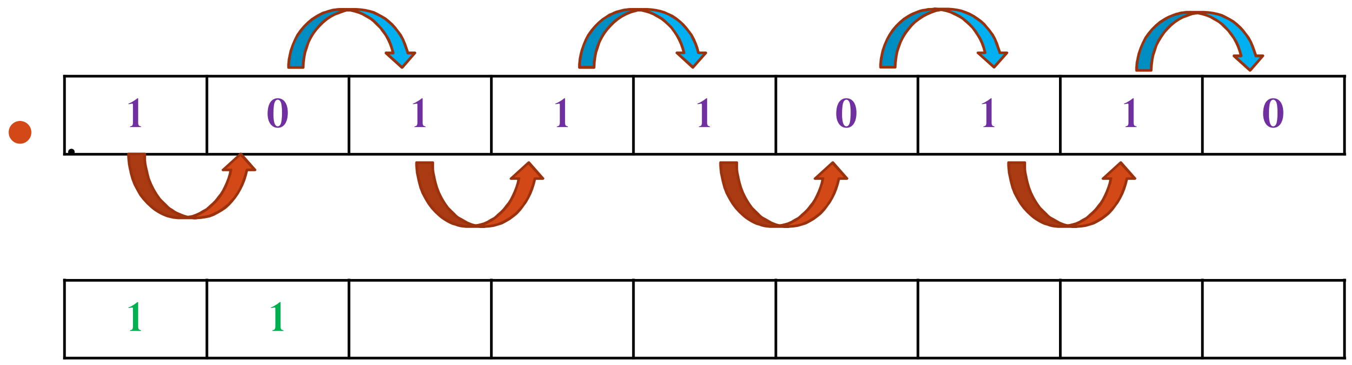
Converting from Binary to Gray Code

Example: Convert 101110110 to Gray Code.

i. Record the MSB of the binary number: $\textcircled{1}$

ii. Add the binary MSB to the next bit position, record the sum and neglect any carries.

1	0	1	1	1	0	1	1	0



iii. Record successive sums until completed.

1	1	1	0	0	1	1	0	1

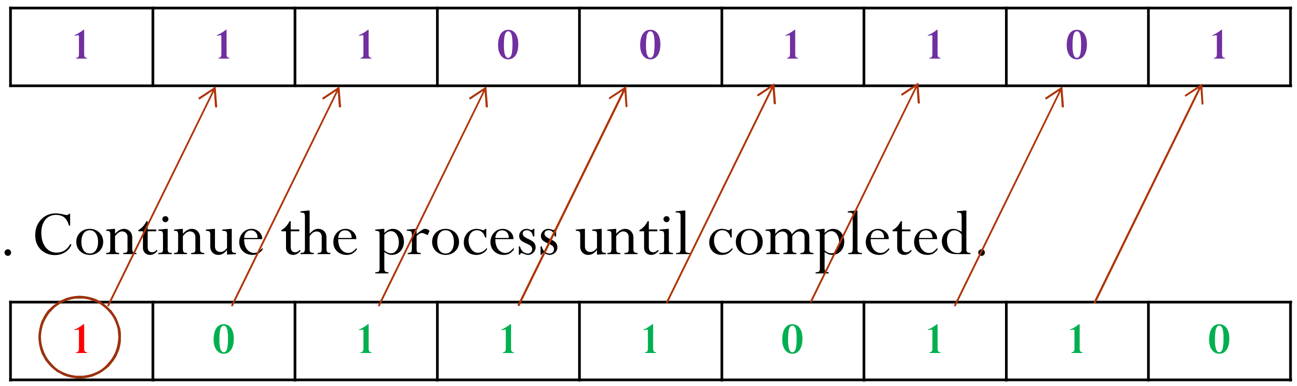
Converting from Gray to Binary

Example: Convert 111001101 to Binary

i. Record the MSB of the given number: 1

NUMBER SYSTEMS AND CODES

ii. Add the binary MSB to the next significant bit position of the Gray Code number you are converting; again recording the sum and ignoring any carries.



iii. Continue the process until completed.

NUMBER SYSTEMS AND CODES

- The table below shows the Gray codes for the first 16 decimal digits:

DECIMAL	BINARY	GRAY
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Advantage of Gray Code

- Only one (1) bit change in the gray code in moving from one number to the next. Gray code is often used in situations where other codes might produce erroneous or ambiguous results during those transitions in which more than 1 bit of the code is changing e.g. in the transition from 7 to 8 in binary, all bits change.

Some Examples on Binary Arithmetic.

Perform: 39-25 in binary.

i. Rewrite the problem as: 39+(-25)

ii. Get the unsigned binary equivalents of 39 and 25.

			32	16	8	4	2	1
39			1	0	0	1	1	1
25			0	1	1	0	0	1

		Sign Bit	32	16	8	4	2	1
+39		0	1	0	0	1	1	1
+25		0	0	1	1	0	0	1

Some Examples on Binary Arithmetic.

		Sign Bit (-64)	32	16	8	4	2	1
+25		0	0	1	1	0	0	1
OCN		1	1	0	0	1	1	0
Add one								1
-25 (TCN)		1	1	0	0	1	1	1

		Sign Bit (-64)	32	16	8	4	2	1
+39		0	1	0	0	1	1	1
+(-25)		1	1	0	0	1	1	1
Solution	$\textcircled{1}$	0	0	0	1	1	1	0

Carry Over/Over flow. Ignore it

$8+4+2 = 14$

Some Examples on Binary Arithmetic.

- Perform: 659+825.

		1024	512	256	128	64	32	16	8	4	2	1
659			1	0	1	0	0	1	0	0	1	1
825			1	1	0	0	1	1	1	0	0	1

		1024	512	256	128	64	32	16	8	4	2	1
659		0	1	0	1	0	0	1	0	0	1	1
825		0	1	1	0	0	1	1	1	0	0	1

		1024	512	256	128	64	32	16	8	4	2	1
659		0	1	0	1	0	0	1	0	0	1	1
825		0	1	1	0	0	1	1	1	0	0	1
Solution		1	0	1	1	1	0	0	1	1	0	0

1024 + 256 + 128 + 64 + 8 + 4 = 1484

Some Examples on Binary Arithmetic.

- Perform: 659-825.
- Rewrite the problem as: 659+(-825)

		Sign Bit (-1024)	512	256	128	64	32	16	8	4	2	1
659			1	0	1	0	0	1	0	0	1	1
825			1	1	0	0	1	1	1	0	0	1

		Sign Bit (-1024)	512	256	128	64	32	16	8	4	2	1
+659		0	1	0	1	0	0	1	0	0	1	1
+825		0	1	1	0	0	1	1	1	0	0	1

		Sign Bit (-1024)	512	256	128	64	32	16	8	4	2	1
+825		0	1	1	0	0	1	1	1	0	0	1
OCN		1	0	0	1	1	0	0	0	1	1	0
Add 1												1
-825		1	0	0	1	1	0	0	0	1	1	1

Some Examples on Binary Arithmetic.

- Perform: 659-825.

		Sign Bit (-1024)	512	256	128	64	32	16	8	4	2	1
+659		0	1	0	1	0	0	1	0	0	1	1
+(-825)		1	0	0	1	1	0	0	0	1	1	1
		1	1	1	0	1	0	1	1	0	1	0

$$-1024 + 512 + 256 + 64 + 16 + 8 + 2 = -166$$

2. BOOLEAN ALGEBRA AND LOGIC GATES

2.1 Introduction

- In Boolean algebra, the variables (known as Boolean variables) are allowed to have only two possible values (0 or 1).
- The expression x =f(A;B) means x is a function of variables A and B where and B are Boolean variables and can only take on two possible values 0 or 1 .

2.2 Basic Operations of Boolean Algebra

- Boolean algebra has only 3 basic operations:

i. Logical addition (the OR operation), Symbol "+"

ii. Logical multiplication (the AND operation), Symbol "."

iii. Logical complementation (the NOT operation), Symbols $*$, $-$, $'$

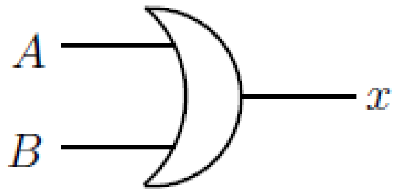
2. BOOLEAN ALGEBRA AND LOGIC GATES

2.2.1 The OR operation

- Operates on two or more variables. It is expressed as:

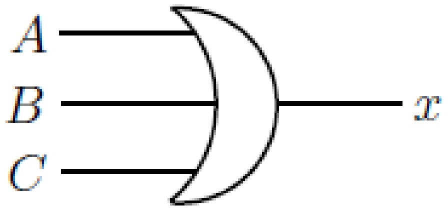
Example: x = A + B

A	B	x
0	0	0
0	1	1
1	0	1
1	1	1



2–input OR gate

Example: x = A + B + C



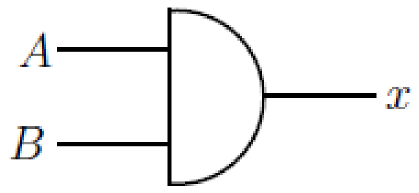
3–input OR gate

A	B	C	x
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

2. BOOLEAN ALGEBRA AND LOGIC GATES

2.2.2 The AND operation

- Example: x = A . B



2–input AND gate

A	B	x
0	0	0
0	1	0
1	0	0
1	1	1

2.2.3 NOT operation

$$x = \overline{A}$$

- Is read as .x = NOT A.

A	x
0	1
1	0

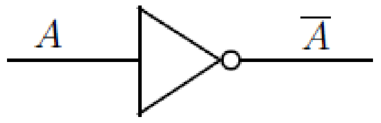


Figure 2.3: A NOT gate (inverter)

2. BOOLEAN ALGEBRA AND LOGIC GATES

2.3 Other Logic Gates

2.3.1 The NOR gate

- For two variables A and B, the NOR operation is defined as:

$$x = \overline{A + B}$$

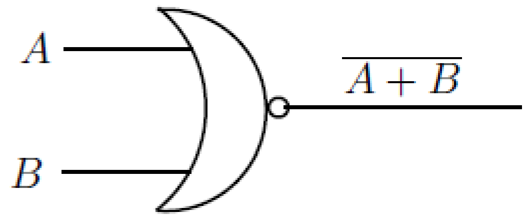


Figure 2.4: A 2-input NOR gate

A	B	x
0	0	1
0	1	0
1	0	0
1	1	0

2. BOOLEAN ALGEBRA AND LOGIC GATES

2.3.2 The NAND gate

$$x = \overline{A \cdot B \cdot C}$$

- Read as .x equals NOT (A AND B AND C).

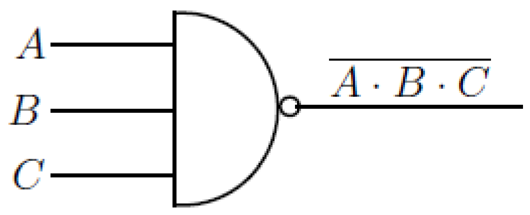


Figure 2.5: A 3-input NAND gate

A	B	C	x
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

2. BOOLEAN ALGEBRA AND LOGIC GATES

2.3.3 The Exclusive-OR gate

- Defined as:

$$x = A \oplus B$$

A	B	x
0	0	0
0	1	1
1	0	1
1	1	0

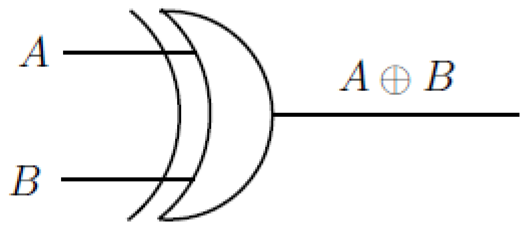


Figure 2.6: An Exclusive-OR gate

- The Exclusive-OR operation is sometimes abbreviated as XOR or EXOR.
- *Note that the XOR gate has only two inputs.*

2. BOOLEAN ALGEBRA AND LOGIC GATES

2.3.4 The Exclusive-NOR gate

- This is usually abbreviated as XNOR or EXNOR gate. It is the complement of the XOR operation.

$$x = A \odot B = \overline{A \oplus B}$$

A	B	x
0	0	1
0	1	0
1	0	0
1	1	1

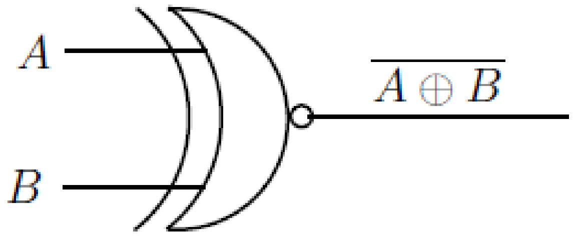


Figure 2.7: An Exclusive-NOR gate

2. BOOLEAN ALGEBRA AND LOGIC GATES

2.4 Laws of Boolean Algebra

Basic Theorems

$A + 0 = A$	$A \cdot 1 = A$
$A + 1 = 1$	$A \cdot 0 = 0$
$A + A = A$	$A \cdot A = A$
$A + \bar{A} = 1$	$A \cdot \bar{A} = 0$

- Looking at the above table, we can see that the corresponding laws on either side are related by:

i. Interchanging + and . Symbols

ii. Interchanging 0 and 1

- Theorems which are related to another by this double interchange are known as duals.

End of session

Questions….?