

Digital Circuits
Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 22
Arithmetic Circuits (Contd.)

So, we were discussing on this code conversion techniques.

(Refer Slide Time: 00:23)

FOUR BIT BINARY TO GRAY CODE CONVERTER –DESIGN (1)...

INPUT (BINARY)				<i>decimal</i>	OUTPUTS (GRAY CODE)			
B3	B2	B1	B0		G3	G2	G1	G0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	1
0	0	1	0	2	0	0	1	1
0	0	1	1	3	0	0	1	0
0	1	0	0	4	0	1	1	0
0	1	0	1	5	0	1	1	1
0	1	1	0	6	0	1	0	1
0	1	1	1	7	0	1	0	0
1	0	0	0	8	1	1	0	0
1	0	0	1		1	1	0	1
1	0	1	0		1	1	1	1
1	0	1	1		1	1	1	0
1	1	0	0		1	0	1	0
1	1	0	1		1	0	1	1
1	1	1	0		1	0	0	1
1	1	1	1		1	0	0	0

Binary code
0 1 0 1 1

Gray code

$HD(x,y) = \text{No. of bit changes from } x \text{ to } y$

1

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, today we will first look into another code converter which will convert binary code to gray code. So, gray code is a special type of code where this successive decimal values, so they differ by only 1 bit. So, take an example. Here say, we have we take this example. So, say the value 0 0 0 0; so you in this column, so if you just write the corresponding decimal value. Then for 0 0 0 0, so the value is 0; so is 1, 2, 3, 4. So, it grows go goes like this.

Now, you see that if you look into the binary pattern corresponding to successive decimal values, so a 0 to 1, only 1 bit is changing; the LSB is changing. But when we are going from 1 to 2, so you see the 2 bits are changing; the LSB b 0 and b 1. So, both are flipping. Similarly when we are going from 2 to 3, again only the LSB is flipping. But when we are going from 3 to 4 here, so here, you see that a 3 bits are flipping

Now, this actually sometimes not very much advisable; particularly when you have got applications in communication and all where we want that the successive values. They should not flip many a time. Also when we have got some integrated circuit chips, VLSI chips and we have a set of signal lines are running, the signal lines in successive time instance is. So, it will assume successive decimal values.

So, it is expected that those values, those bits will not flip much from 1 1 value to the next value. This is particularly true when we look into the microprocessors where there is address bus connecting to memory. So, address bus, it will generate the successive addresses and then if I have following binary pattern then there will be a large number of bit flips. So, this leads to a good amount of power consumption, bus power consumption.

So, what we do? We convert them into something called a gray code. So, in a gray code that is shown here in the right side. So, here this, if you look into the value from 0 to 1, only the LSB is flipping; from 1 to 2, only the bit g 1 is flipping. So, bit g 0 continuous to be 1. Similarly from 2 to 3 only bit g 0 is flipping, other bits remain unaltered. So, in this way if you look into any two successive bit patterns, say successive decimal values, their gray codes, they differ only by a 1 position; only 1 bit will be flipping. So, others will remain unaltered.

So, we define something called a hamming distance. So, hamming distance between two patterns say x and y is equal to the number of bit changes from x to y. So, this is defined as the number of bit changes between x and y. So, in case of, in case of binary number system; so you see that it is not a fixed value. So, many sometimes only 1 bit flips sometimes particularly if you look into this 7 to 8, this flipping; you see that all the 4 bits are flipping. So, there a hamming distance becomes high.

On the other hand, if you look into the gray code pattern from 7 to 8 also, you see only this bit is flipping, others are not flipping. So, in case of hamming, in case of gray code; this hamming distance is always equal to 1 between 2 successive decimal values. So, this has got many application in digital systems and so, this code conversion, is this decimal to binary to gray code conversion is sometimes very useful

So, the way it is done; so this is the logic. So, it is a MSB is copied directly. So, from this m MSB, it is XORed with the next bit to produce the next one, produce the next most significant bit. And again, so this binary code; so these are this is the binary code that we

are trying to convert into gray code ok. The 5 bit binary code we are trying to convert into gray code. So, this is binary digits, they are XORed to get the next code.

(Refer Slide Time: 05:09)

FOUR BIT BINARY TO GRAY CODE CONVERTER –DESIGN (2)...
Simplification using K-maps:

$G_3 = B_3$

$G_2 = \overline{B_3} B_2 + B_3 \overline{B_2}$
 $G_2 = B_3 \oplus B_2$

$G_1 = B_2 \overline{B_1} + \overline{B_2} B_1$
 $G_1 = B_2 \oplus B_1$

$G_0 = \overline{B_1} B_0 + B_1 \overline{B_0}$
 $G_0 = B_1 \oplus B_0$

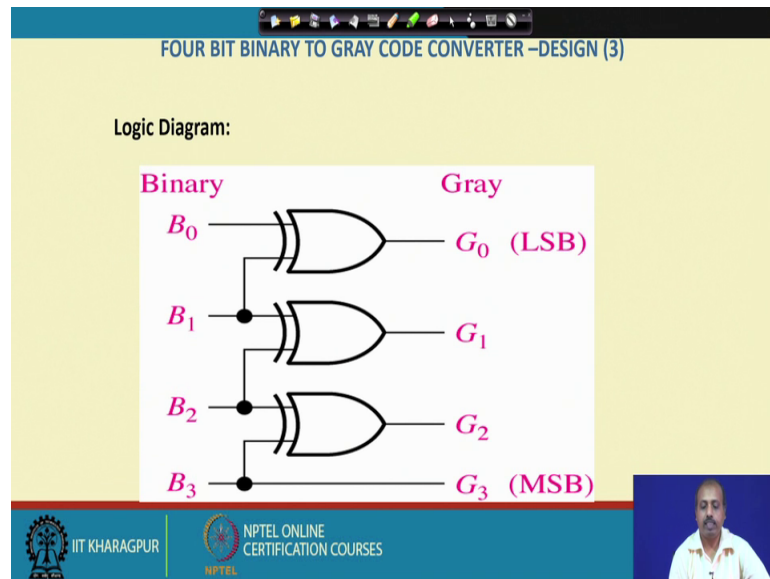
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, we will come to this definition via this truth table. So, we will start with the truth table here and then we will show you that this actually leads to this functionality. So, let us take, let us take this simplification of this code. So, if we look into the corresponding Karnaugh maps for G 3, G 2, G 1 and G 0; that is G 3 is this column, G 2 is this column.

So, if you look into the corresponding Karnaugh maps. So, G 3 is 1 from whenever B 3 is equal to 1. So, whenever B 3 is equal to 1 G 3 is equal to one. So, here you see in the karnaugh map, we have done it like this and if you do a grouping and do a minimization then G, the equation that we get is G 3 equal to B 3. Similarly for G 2, the function turns out to be B 3 bar B 2 plus B 3 B 2 bar which is nothing but B 3 XOR B 2.

Similarly, G 1 gives me B 2 XOR B 1 and G 0 gives B 1 XOR B 0. So, this is as per the definition that we have here. You see that here also the basic definition of gray code is like that and you see that the corresponding Boolean functions. So, that is also leading to the same thing.

(Refer Slide Time: 06:11)



So, you can very easily draw a logic diagram which can convert a 4 bit binary code into a gray code where this B₃ will be directly copied to G₃ as per this equation $G_3 = B_3$. $G_2 = B_3 \oplus B_2$. So, G_3 for getting G_2 so, B₃ and B₂ they are XORed. So, get $G_1 = B_2 \oplus B_1$. Similarly, G_1 is B₂ XOR B₁. So, this is XORed here and G_0 is B₁ XOR B₀. So, that gives the LSB. So, this way you can have a very simple gray code converter using XOR gates.

Next we consider the other way that is starting with the gray code. How can we reach the binary codes? So, this is the standard code conversion that we have. So, if I again, if we say that these are the decimal values; so, we say that here I am writing down the decimal, then this is 0, 1, 2, 3, 4, 5 etcetera.

(Refer Slide Time: 07:07)

FOUR BIT GRAY CODE TO BINARY CONVERTER –DESIGN (1)...

INPUT (GRAY CODE)				dec	OUTPUTS (BINARY)			
G3	G2	G1	G0		B3	B2	B1	B0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	1
0	0	1	0	2	0	0	1	1
0	0	1	1	3	0	0	1	0
0	1	0	0	4	0	1	1	1
0	1	0	1	5	0	1	1	0
0	1	1	0		0	1	0	0
0	1	1	1		0	1	0	1
1	0	0	0		1	1	1	1
1	0	0	1		1	1	1	0
1	0	1	0		1	1	0	0
1	0	1	1		1	1	0	1
1	1	0	0		1	0	0	0
1	1	0	1		1	0	0	1
1	1	1	0		1	0	1	1
1	1	1	1		1	0	1	0

Gray code

Binary code

IIT KHARAGPUR
 NPTEL ONLINE CERTIFICATION COURSES

Now, here also these values are known. For gray code, the values are like this for binary code; the values are like this. Now the way it is done is this MSB will be directly copied and then so this is the gray code that I have 1 0 1 0 0. Now that is 1 is copied directly to the binary MSB. So, that 1 is copied. Then this bit is XORed with the next gray coded bit 0 to produce the next bit. So, this way it goes.

So, this binary code produced, binary bit produced, so that is XORed with the next gray code bit to produce the next binary bit ok. So, that is the logic. So, we will see that this really happens, if we are looking into the truth table.

(Refer Slide Time: 08:08)

FOUR BIT GRAY CODE TO BINARY CONVERTER –DESIGN (2)...

B_3

G_3, G_2	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$B_3 = G_3$

B_2

G_3, G_2	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$B_2 = \overline{G_3} G_2 + G_3 \overline{G_2}$
 $B_2 = G_3 \oplus G_2$

B_1

G_3, G_2	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	0	0	1	1
10	1	1	0	0

$B_1 = \overline{G_3} \overline{G_2} G_1 + G_3 \overline{G_2} G_1 + \overline{G_3} G_2 \overline{G_1} + G_3 G_2 \overline{G_1}$
 $= G_1 (\overline{G_3} \overline{G_2} + \overline{G_3} G_2) + G_1 (G_3 \overline{G_2} + G_3 G_2)$
 $= G_1 (\overline{G_3} \oplus G_2) + G_1 (G_3 \oplus G_2)$
 $= G_1 \oplus G_3 \oplus G_2$
 $B_1 = G_1 \oplus B_2$

B_0

G_3, G_2	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	0	1	0	1
10	1	0	1	0

So, let us see that ok, this is the truth table for the function. So, B 3 equal to G 3. B 2 becomes equal to G 3 bar G 2 plus G 3 G 2 bar. So, B 3 is a B 2 is G 3 XOR G 2. Similarly B 1 becomes G 1 XOR B 2 ok. So, this way we can get the individual bits.

(Refer Slide Time: 08:33)

FOUR BIT GRAY CODE TO BINARY CONVERTER –DESIGN (3)...

B_0

G_3, G_2	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	0	1	0	1
10	1	0	1	0

$B_0 = \overline{G_3} \overline{G_2} \overline{G_1} G_0 + \overline{G_3} \overline{G_2} G_1 \overline{G_0} + \overline{G_3} G_2 \overline{G_1} \overline{G_0} + \overline{G_3} G_2 G_1 G_0$
 $+ G_3 \overline{G_2} \overline{G_1} \overline{G_0} + G_3 \overline{G_2} G_1 G_0 + G_3 G_2 \overline{G_1} \overline{G_0} + G_3 G_2 G_1 G_0$
 $= \overline{G_3} \overline{G_2} (\overline{G_1} G_0 + G_1 \overline{G_0}) + \overline{G_3} G_2 (\overline{G_1} \overline{G_0} + G_1 G_0)$
 $+ G_3 \overline{G_2} (\overline{G_1} G_0 + G_1 \overline{G_0}) + G_3 G_2 (\overline{G_1} \overline{G_0} + G_1 G_0)$
 $= \overline{G_3} \overline{G_2} (G_1 \oplus G_0) + \overline{G_3} G_2 (G_1 \oplus G_0)$
 $+ G_3 \overline{G_2} (G_1 \oplus G_0) + G_3 G_2 (G_1 \oplus G_0)$
 $= (G_1 \oplus G_0) (\overline{G_3} \overline{G_2} + \overline{G_3} G_2) + (G_1 \oplus G_0) (G_3 \overline{G_2} + G_3 G_2)$
 $= (G_1 \oplus G_0) (\overline{G_3} \oplus G_2) + (G_1 \oplus G_0) (G_3 \oplus G_2)$
 $= G_0 \oplus G_1 \oplus G_2 \oplus G_3$
 $B_0 = G_0 \oplus B_1$

So, B 0 is big expression like this and then you can simplify and finally it boils down to this B 0 is G 0 XOR B 1. So, actually this B 1 is given by G 1 XOR B 2 and B 2 is given by G 3 XOR B 2. So, if I just expand it, so this can be in the previous state at this point. If you see, B 1 turns out to be G 1 XOR G 3 XOR G 2.

Now in the next one, you see that we have got this $G_1 \text{ XOR } G_2 \text{ XOR } G_3$. So, that part we can simplify to have $G_1 B_1$ directly. So, here this the logic diagram of this binary to the gray to binary code converter is something like this. So, we have got this G_0, G_1, G_2 and G_3 on the left side. So, G_3 is directly copied to B_3 and this G_3 and G_2 are XORed to get B_2 .

Now for getting B_1 , I need to XOR G_1, G_2 and G_3 . So that will require 3 input XOR gate. So, to simplify, so $G_2 \text{ XOR } G_3$ has already been computed in B_2 . So, this B_2 is taken here and we do $G_1 \text{ XOR } B_2$ to get B_1 . Similarly for G_0 , I need G_0 ; for getting B_0 , I need to XOR all this G_0, G_1, G_2, G_3 . So, what we do, B_1 is already having $G_1 \text{ XOR } G_2 \text{ XOR } G_3$. So, that is XORed with G_0 to get B_0 . So, this is possible because a XOR is a linear function that we have seen and so, it is associative in nature.

So, you can do any XOR at any you can do the XORs in either order. So, that way it is giving as the flexibility. So, you can do this for this gray code to binary code conversion.

(Refer Slide Time: 10:40)

Exercise

1. Convert the binary number 0101 to Gray code with XOR gates
2. Convert the gray code 1011 to binary with XOR gates

Solution:

(a) Binary to Gray: A circuit with four XOR gates. The inputs are Binary bits 1, 0, 1, 0. The outputs are Gray bits 1, 1, 1, 0. The first XOR gate takes inputs 1 and 0. The second XOR gate takes inputs 0 and 1. The third XOR gate takes inputs 1 and 0. The fourth XOR gate takes inputs 1 and 0.

(b) Gray to Binary: A circuit with four XOR gates. The inputs are Gray bits 1, 1, 1, 0. The outputs are Binary bits 1, 0, 1, 1. The first XOR gate takes inputs 1 and 1. The second XOR gate takes inputs 1 and 1. The third XOR gate takes inputs 1 and 0. The fourth XOR gate takes inputs 1 and 0.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, we can convert binary number 0 1 0 1 to gray code like this. So, this is 0 1 0 1, this is the binary number. Now for the gray, so this is the circuit for gray code converter. So, 0 will be copied here. Then this 0 and 1 are XORed to get 1; then this 0 and 1 are XORed to get 1. So, this way it goes. So, final solution it becomes so, 1 0 1 0. It is when it is converted to gray code, it is 1 1 0 0 or if we want to convert the gray code number 1 0 1 1 into binary, so this gray code bit is 1 0 1 1. So this B G B that this 1 is directly copied

here. Then the MSB is directly copied, then this MSB and this bit G 2, so they are XORed to get the bit b 2. Then B 2 XOR the G 1, so that gives me B 1. So, that way it goes. So, this is the pattern that I get is 1 1 0 1.

(Refer Slide Time: 11:44)

BCD to XS 3 code converter- Design (1)...

Input (Std BCD code)				Output (XS3 Code)			
A	B	C	D	w	x	y	z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

Handwritten annotations on the table: A checkmark is next to the '0' in the 'D' column of the first row. The numbers 0, 1, and 2 are written next to the 'D' column of the first three rows. The numbers 3, 4, and 5 are written next to the 'z' column of the first three rows. A checkmark is also present in the 'D' column of the row corresponding to decimal 5 (1010).

Next we look into another code which is known as BCD to XS 3 code so many times what happens is that we want to add some base to the code. So, we do not allow some of the codes to appear. For example here, so XS 3 code means whatever be the value from there you need to subtract 3 to get the actual value. For example, so if I look into this pattern, you see that here the values that I am getting is 3. If I just take it, the binary representation so, this is 3.

So, XS 3 means from this I need to subtract 3. So, if I 3 minus 3 is 0, so this actually the corresponding decimal value is 0. Similarly, so this is 0 1 0 0 that is 4. So, to get the actual decimal value, I have to do 4 minus 3 that is 1.

Similarly 1 0 1, so this is 5; so 5 minus 3, 2. So, XS 3 code means it has got an excess of 3 from the value. So, if I am talking about BCD to XS 3 conversion then, so BCD can go up to the digit 9. So, it is going from 0 to 9. So, this is my zero and we have 9 here, 1 0 0 1. So, up to this much.

So, nine will be converted to 12 ok. So, that the 9 plus 3 twelve. So, this is the XS 3 representation of 9. For rest of the um bit pattern 4 bit pattern, I do not need to consider

because in BCD, we do not have this combinations 1 0 1 0 1 0 1 1 and all. So, for them I can take the output to be simply do not care because they will not appear in the conversion a in the conversion process ok. So, this co patterns will never come.

(Refer Slide Time: 13:48)

BCD to XS 3 code converter- Design (2)...

AB		C			
		00	01	11	10
A	00	1			1
	01	1			1
	11	X	X	X	X
	10	1		X	X

$z = D'$

AB		C			
		00	01	11	10
A	00	1		1	
	01	1		1	
	11	X	X	X	X
	10	1		X	X



$y = CD + C'D'$

AB		C			
		00	01	11	10
A	00		1	1	1
	01	1			
	11	X	X	X	X
	10		1	X	X

$x = B'C + B'D + BC'D'$

AB		C			
		00	01	11	10
A	00				
	01		1	1	1
	11	X	X	X	X
	10	1	1	X	X

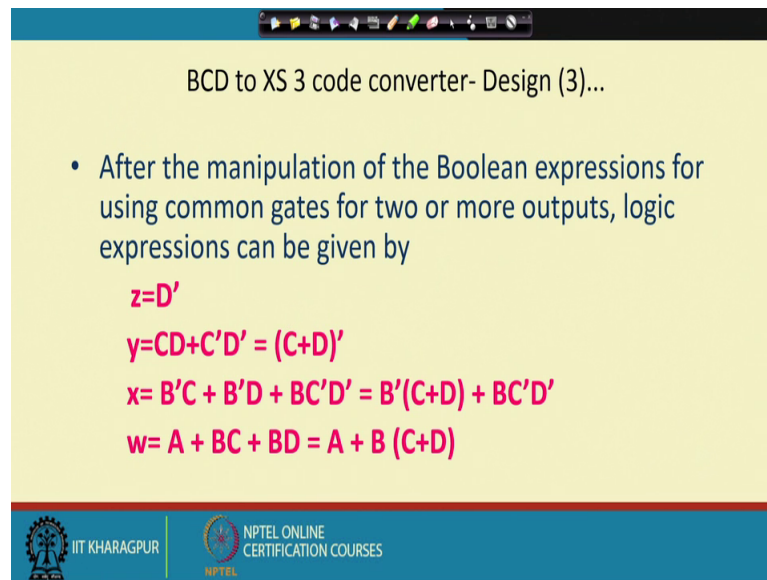
$w = A + BC + BD$


IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

So, with this, we will be we can go to the circuit for getting the converter BCD to XS 3 converter. So, it is like this that we have this say so, the first the z bit; so xyz and w, so this z is the first one. So, z is this LSB. So, corresponding to that if you look into the circuit, it becomes z equal to D bar or D dash.

Similarly y, this column; so, this will give us CD plus C bar D bar. So, this is the XNOR function. Then this part x; so this x, so, this gives us the function B bar C plus B bar D plus B C bar d bar. So, then for w we get A plus BC plus BD. So, I can design a logic circuit for this w x y and z and get this BCD to XS 3 conversion.

(Refer Slide Time: 14:43)



BCD to XS 3 code converter- Design (3)...

- After the manipulation of the Boolean expressions for using common gates for two or more outputs, logic expressions can be given by

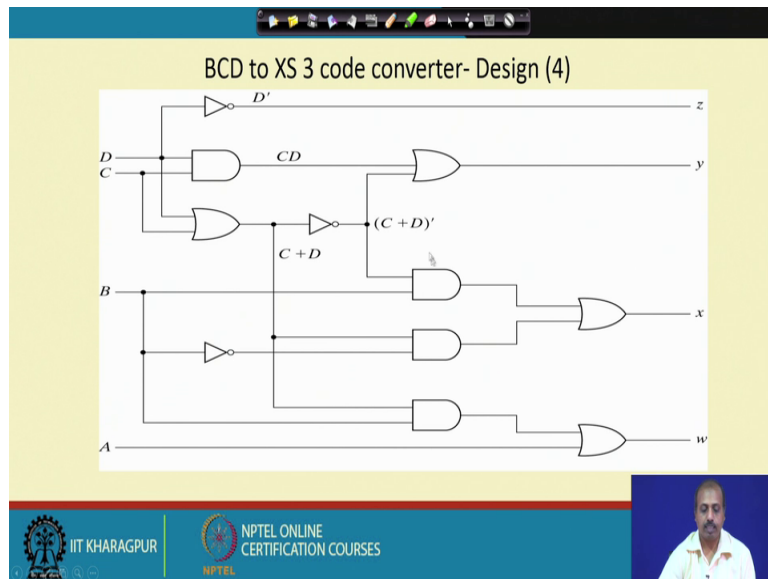
$$z = D'$$
$$y = CD + C'D' = (C+D)'$$
$$x = B'C + B'D + BC'D' = B'(C+D) + BC'D'$$
$$w = A + BC + BD = A + B(C+D)$$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, after manipulation of these Boolean expressions using these common gates for two or more outputs, logic expressions are given by this. So, this is this y is CD plus C bar D bar that is a 1 NOR NOR gate. So, we can sorry CD plus C bar d bar is an XOR gate, so XNOR gate. So, this is actually though written as plus, but this is actually 1 XNOR function. So, this is the XNOR function.

Similarly here, so B bar C plus B bar D plus B C bar D bar. So, this is XNOR function and then so, this B bar common. So, it is C plus D plus B bar C. So, this C plus D is an OR gate and then with that we can take AND with D bar. Then A plus BC plus BD so, we can take B common and C plus D comes as a common function. So, this C plus D can be taken as this C plus D can be taken as the common function for doing the manipulation and then we can realize the circuit ok.

(Refer Slide Time: 16:00)



So, this is the function for x y z and w and in fact, in this case this XNOR gate and NOR gate so, they are becoming similar. So, we can use this y, so instead of doing XNOR so we can do NOR also. So, that will also give us the similar result because of the presence of do not cares.