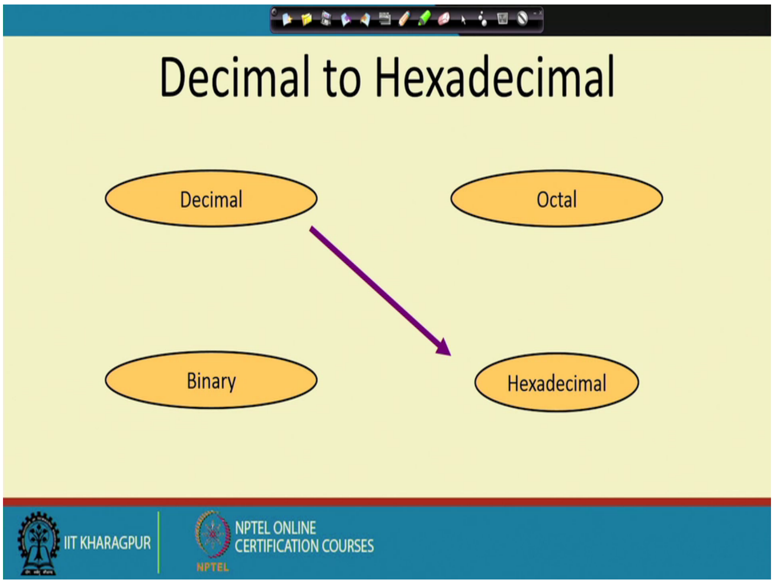
Digital Circuits

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

Lecture – 04
Number System (Contd.)

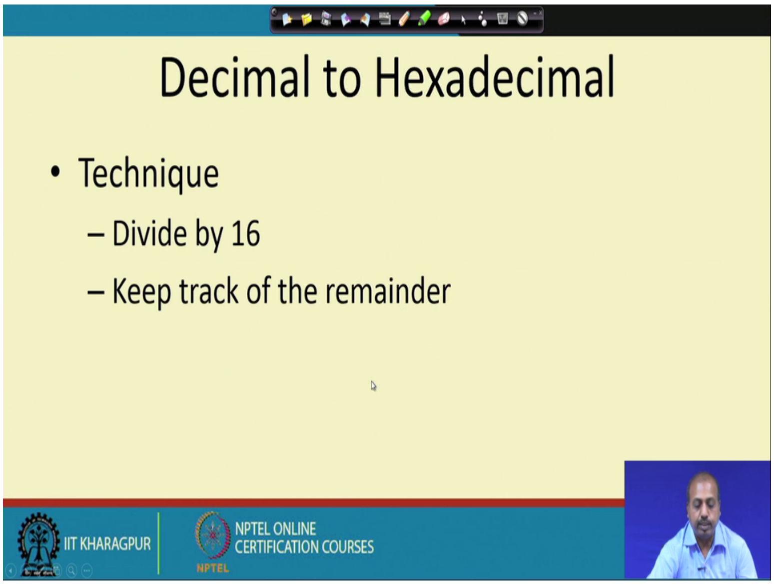
Decimal to hexadecimal conversion so, this is also similar in nature.

(Refer Slide Time: 00:20)



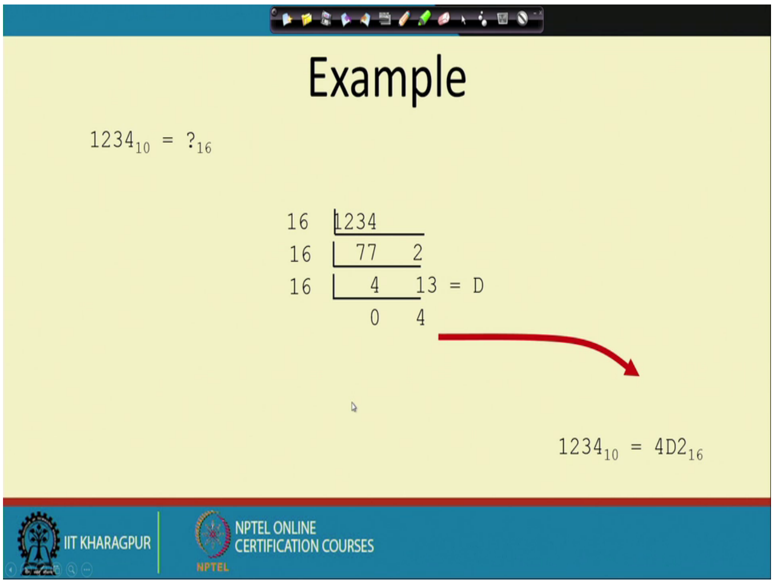
So, here we have to divided by 16 as you can understand by this time.

(Refer Slide Time: 00:29)



So, this base conversion so, will be dividing it by 16. So, divide by 16, and keep track of the remainder.

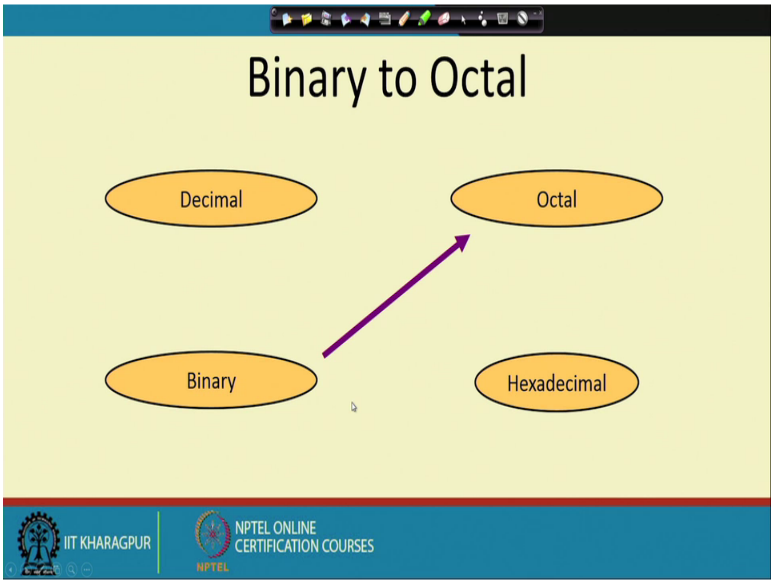
(Refer Slide Time: 00:33)



So, 1 2 3 4 to the base 10 converting to hexadecimal. So, only thing is that division becomes a bit complex. So, divide by 16 so, remainder is 2, then divide by 16, remainder is 13. And 13 the corresponding hexadecimal symbol is D, so, 13 is basically D. So, this is the integer value, and this is the hexadecimal symbol corresponding to that. And then is 4 divided by 16, remainder is 4 so the number that you get is 4 D 2. So, this 4 D 2 so, this is the number corresponding to this.

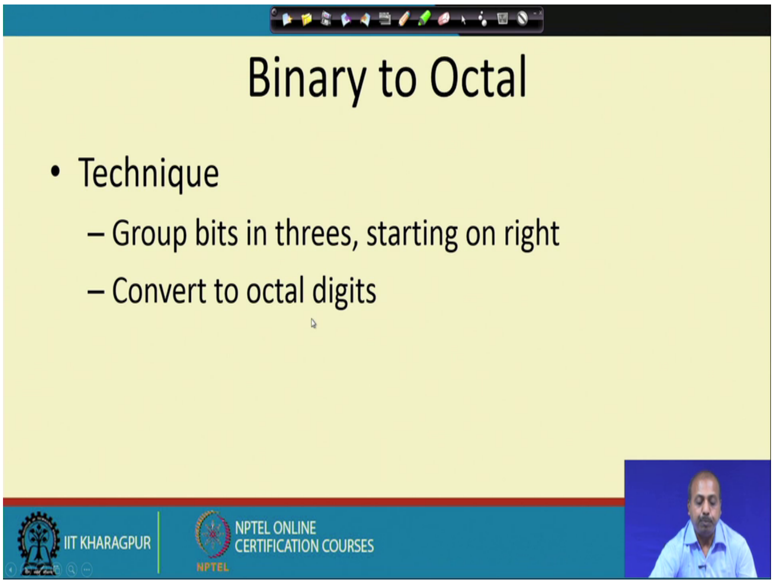
So, this way we can convert by a decimal number to different hexadecimal numbers.

(Refer Slide Time: 01:13)



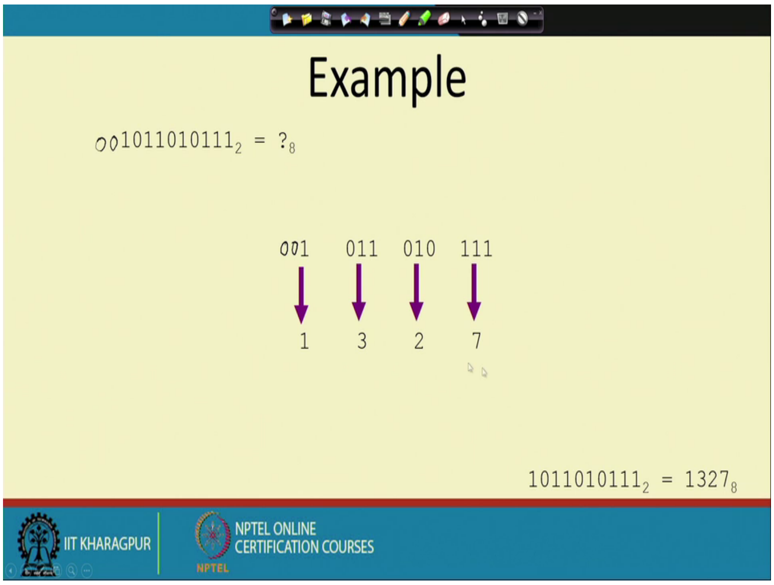
Now, the other way binary to octal so, how can you do this thing? As I already said the one possibility is that is binary number, you convert to decimal and from this decimal you go to octal. So, that is one avenue, but we can say that that is 2 conversion will be necessary in the process. But that is not required so, you can do it directly.

(Refer Slide Time: 01:35)



So, you can group bits in 3s starting on the right and convert to octal digits so, will explain it with an example.

(Refer Slide Time: 01:44)

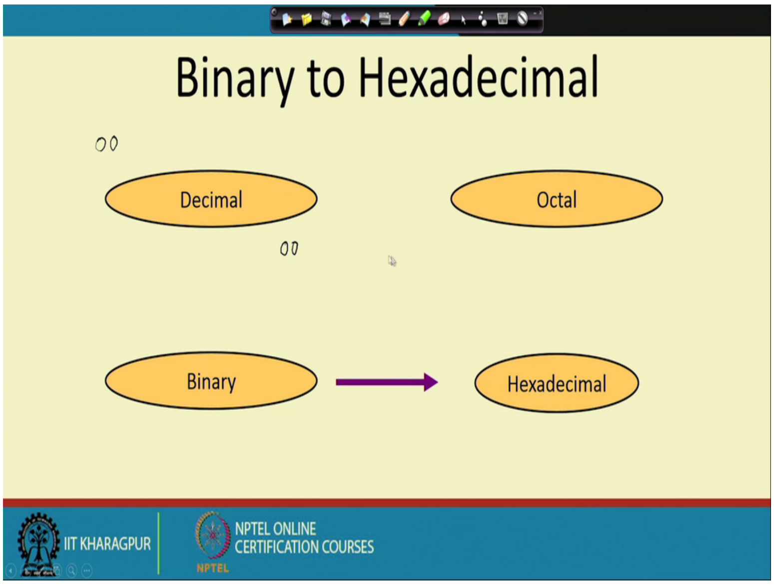


Suppose this is the binary number that you have, and we it does the so, I want to get the corresponding octal number. So, what is done? So, I make groups of 3 starting from the right side. So, you must keep in mind so, should not start from the left side so, you should start from right side.

Then this first 3 bits 1 1 1, they from the first group then this 0 1 0, from the next group. Then this 1 1 0 from the third group, and here I do not have 3 bits. So, you can safely assume that we have got this bits has 0 0's. 2 0's are there so, as a, because I can always assume that there are 2 0's at this point. And because they will not contribute anything to the number. So, I can do that so, that way I can have this.

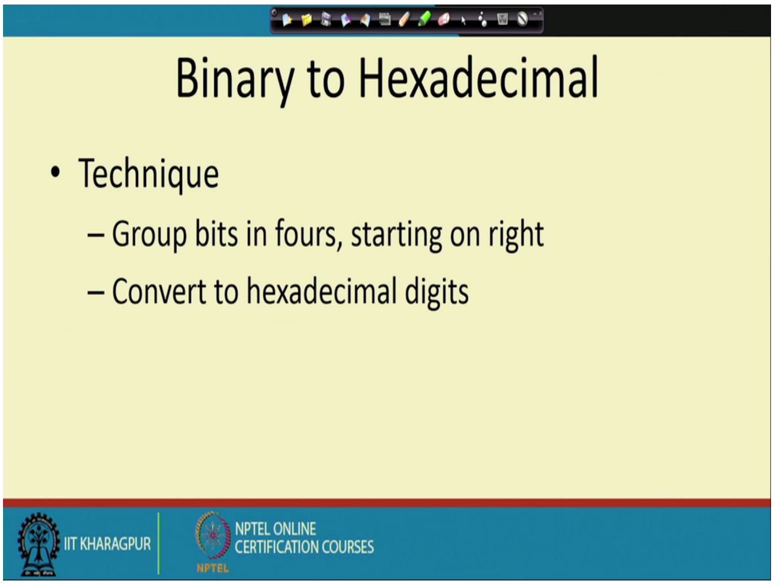
So now you convert the bits the 3 bits pattern into corresponding octal numbers. So, this 1 1 1 is 7, 0 1 0 is 2, 0 1 1 is 3 and 1 0 0 0 1 is 1. And the corresponding number becomes, this one 3 2 7.

(Refer Slide Time: 02:53)



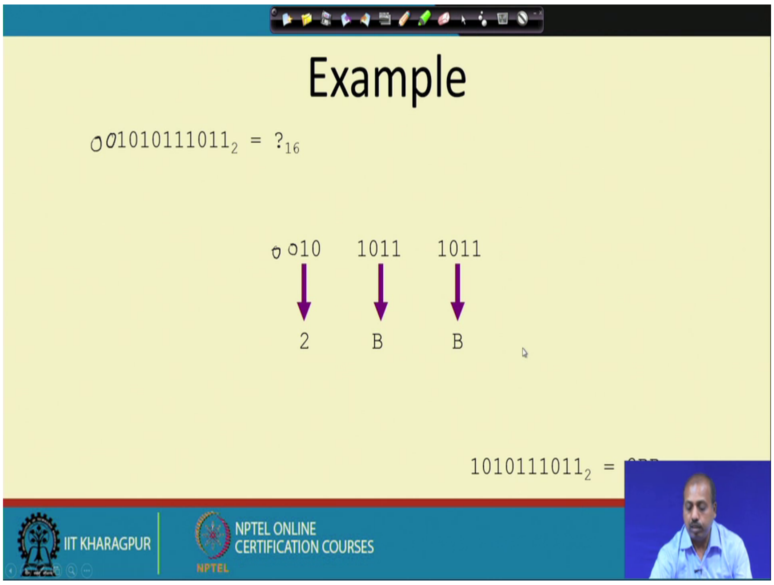
So, this is the corresponding number so, if you are going for this binary to hexadecimal conversion. So, here also the same thing, so, but now the grouping that will do will be in terms of 4 bits.

(Refer Slide Time: 03:08)



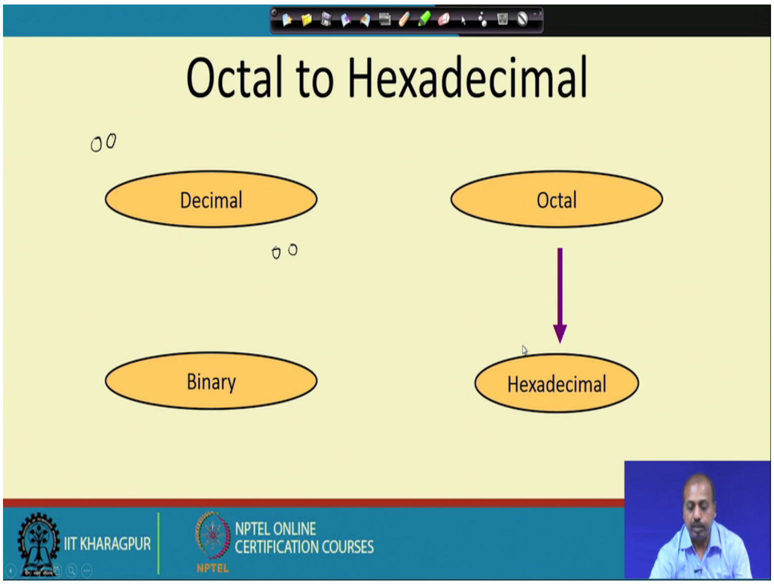
So, it will be group in terms of 4, 4 bits starting from the right side, and then that the groups the bit pattern group that we have so, they will be converted to hexadecimal digits.

(Refer Slide Time: 03:21)



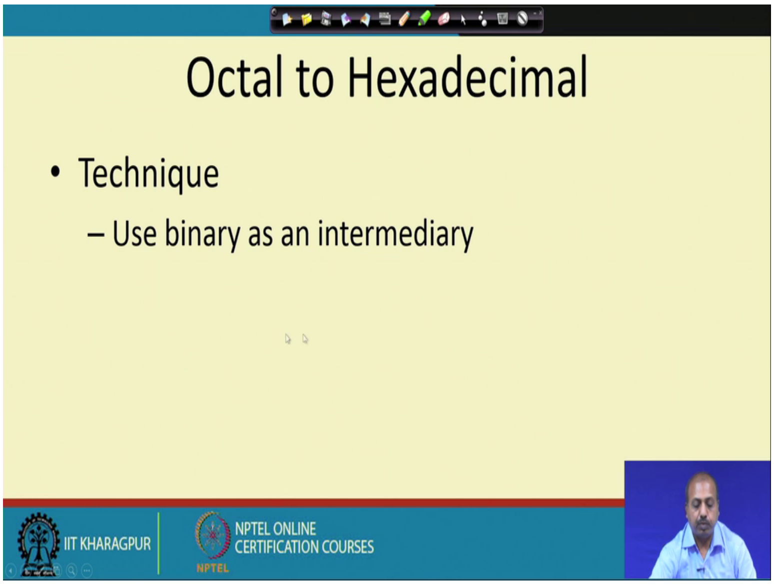
So, here o, your grouping in terms of 4 bits from the right. So, this 1 1 0 1 they from the first group then this 1 1 0 1 from the next group and then I have got only 2 bits left 0 and 1. So, 0 1 and as I did in the previous case so, you can safely assume that we have got 0's here, and those 0's are brought here ok. So, you can have this 0 0 1 0. So, ultimately the pattern becomes 2 BB, and this 2 BB is the hexadecimal number corresponding to the decimal number the binary number that we have.

(Refer Slide Time: 03:57)



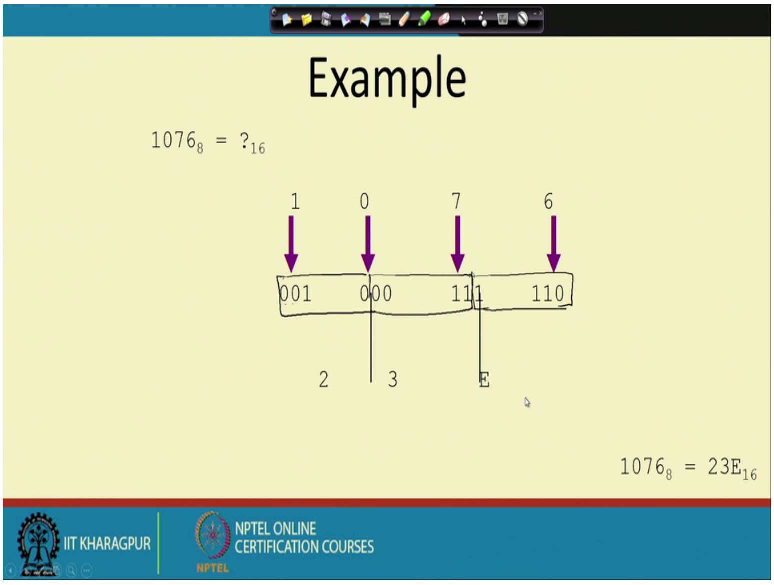
So, then how to convert octal number to hexadecimal numbers.

(Refer Slide Time: 04:08)



So, they it is it says that you can use a binary number as the intermediary. So, we take what we do is we convert the octal number to binary number first and from the binary number you convert to hexadecimal number.

(Refer Slide Time: 04:25)



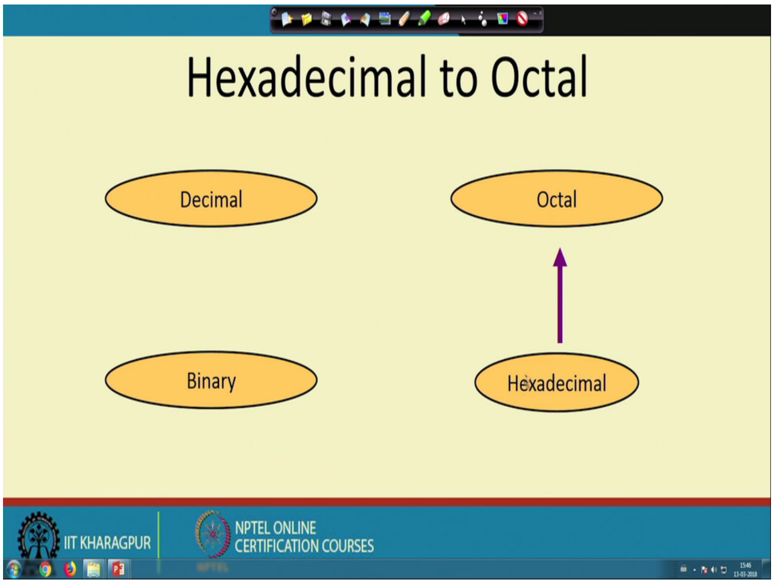
Like say this is an octal number 1 0 7 6 to the base 8, what is the value in the hexadecimal number.

So, first this 1 0 7 6 so, since this is an octal number so, individual digits I can convert it into 3 bit patterns and getting, the corresponding binary bits 0 0 1 0 0 0 1 1 1 1 1 0. And now I will be grouping then in terms of 4 bits from the right side. So, this say you see that this part so, this portion up to this much. So, it is can be it is taken as one group, then this part is taken as another group, this part is taken as another group, and then this part is taken as another group.

And again for making it ah 4 so, the 2 extras 0's have been added, at the beginning to make it a group of 4. And then you convert it into the corresponding hexadecimal numbers. So, this 1 1 1 0 so, this is 14 so, 14 in hexadecimal number system is E. So, that is E, similarly this is 0 0 1 1 that is 3. So, that is in hexadecimal also this is 3, and 0 0 1 0 that is that is 2 in hexadecimal system it is true 2.

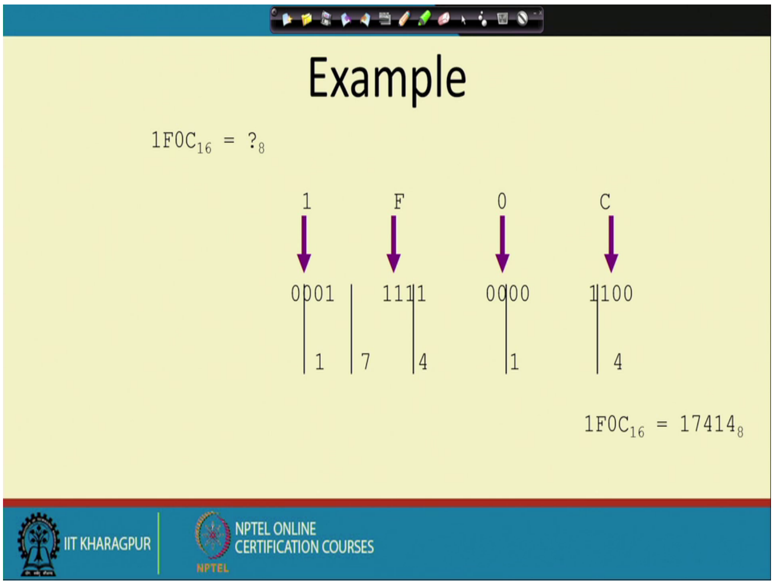
So, you can do this conversion, and accordingly you get this 1 0 7 6 in hexadecimal in octal number system is equivalent to 23 E in the hexadecimal number system.

(Refer Slide Time: 06:02)



Now, hexadecimal to octal so, this also you can guess what are we trying to do what will be doing. So, will be converting hexadecimal to binary, and from binary you will be converting in to octal. So, will be following this avenue, ok, so, let us see how it is done using binary as intermediary.

(Refer Slide Time: 06:25)



So, suppose we have got this hexadecimal number 1F0C to the base 16, and we want convert into corresponding octal number.

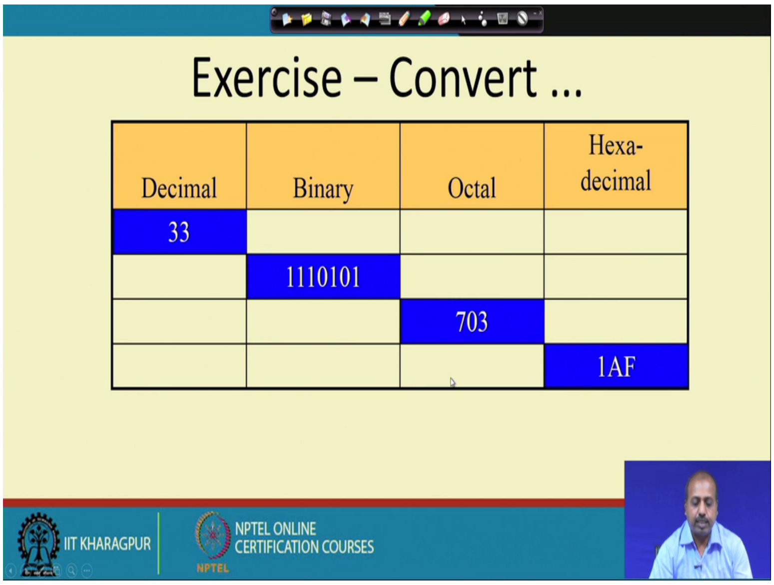
So, this so, first of all this 1F06 so, that we right down the corresponding the binary representation. And we know that straight way we can convert these individual digits to binary 4 bit binary patterns, and then that gives us the binary representation of this number.

Now, after that since I am trying to go to octal so, I have to make the groups of 3 so, I make groups of 3 from the right. So, the 0 0 1, that is there, next group is 1 0 0, next group is again 0 0 1, next group is 1 1 1 and the last group is 1 0 0. So, 3 extra 0 have been added so, that we can make it we can get 4 bit pattern here.

So, that way the number that we get in the octal number system is 17414 in the octal number system. So, this way we can convert very easily between this octal and hexadecimal number systems using this binary number system as the intermediary,.

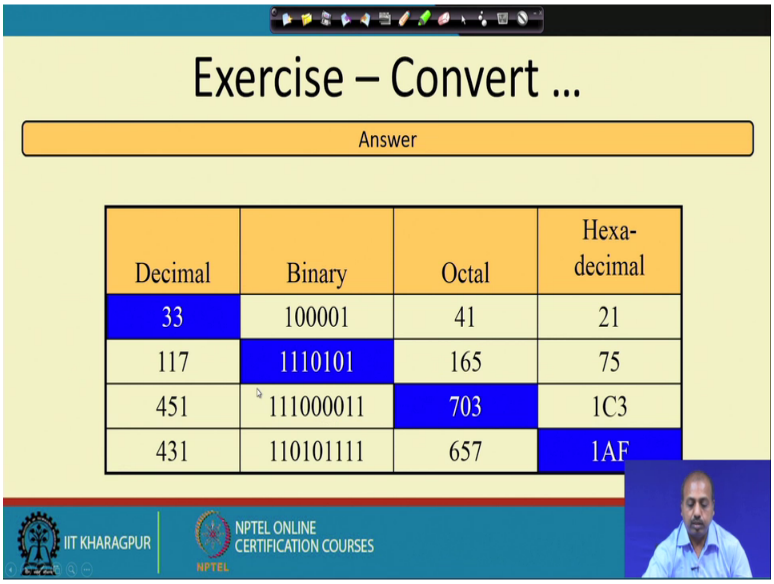
So, otherwise we have to convert to decimal, and then by from multiplying by powers of 8, and or dividing by power after the dividing by powers of 16. If you are trying for octal to hexadecimal conversion or hexadecimal to octal conversion you have to convert to decimal by multiplying by powers of 16, and then from that decimal number to octal by dividing by powers of 8.

(Refer Slide Time: 08:05)



And do a very simple exercise like the we have got assume numbers, and how to what the values will be in different number system so, this is the answer.

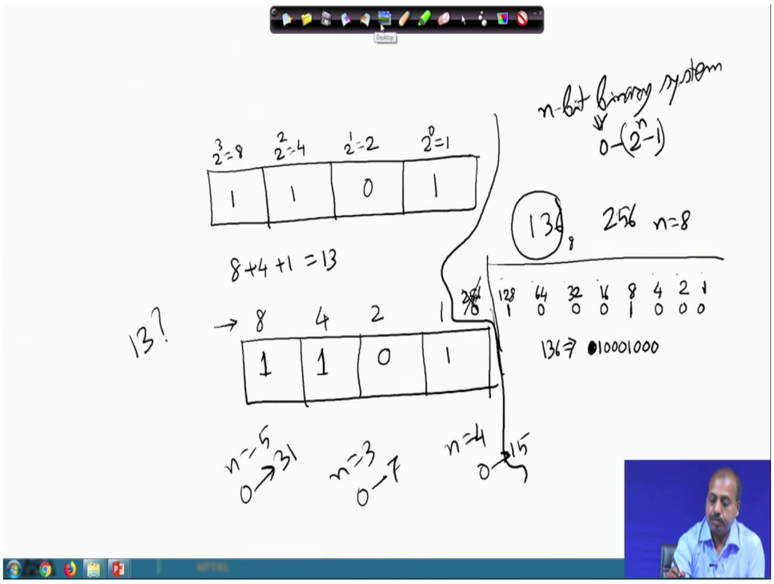
(Refer Slide Time: 08:15)



So, 33 in a hexa in binary number system so, this will be 1 0 0 0. Actually whenever you are doing these operations this binary number system so, that can be and easier way of doing this conversion so, you did not always go on dividing by 8 ok. So, will looking to a technique, and that will make it simple ok.

So, let us say that if will looking to this individual digits of a binary number system. So, the first so, if this is a if suppose this is the this is a 4 bit number; for example 4 bit binary number,.

(Refer Slide Time: 08:53)



So, what is the contribution of first number? So, that is 2 2 the power 0 equal to 1. Contribution of the next digit is 2, 2 the power 1 that is equal to 2. This is 2, 2 the power 2 equal to 4, and this is 2 2 the power 3 equal to 8.

So, any number that you are representing so, they will be summing of the values of this 8 4 2 and 1 so, if I take the number say 1 1 0 1 so, the number is 8 plus 4 plus 1 so, that is 13.

So now if I ask you the question in the other direction, I tell you what is the representation of 13 in the binary number system. And you have in your mind this particular scale, ok, that on this scale I have got this values. So, this is 8, this is 4, this is 2, this is 1, this are the weights. So, what you do from 13? So, since the since the most significant value, the weight is 8 so, this has to be given to reach 13 so, from 13 I have given 8. So, I am left with 4, sorry I am left with 5 next value is 4 so, I have to take it because I need to reach 5. So, I have to take it so, this 1 also I take.

So, what is as now I am left with 1, and but the next weight is 2. So, I cannot take this so I put a 0 here, and the next one I am I am still left with 1 so, I take a one here. So, you see that if I just remember this particular scale then I can just put them 1's and 0's to convert it into the binary number system. So, any arbitrary number let us takes a say 136. First of all we have to see how many bit representation will be required for 136.

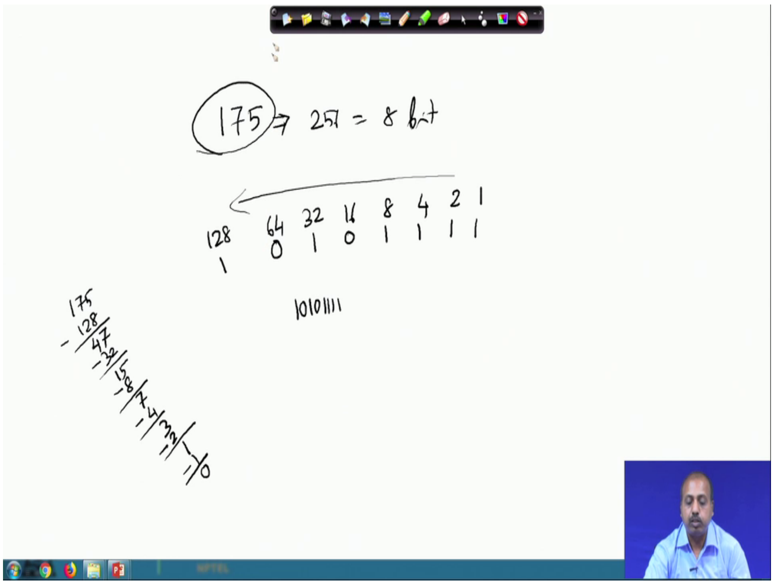
See if I have got n bit so, if I have got n bit binary number system, n bit binary number system, then it can represent numbers in the range of 0 to 2 2 the power n minus 1. So, n bit binary numbers for example, if you take n equal to 4. So, you can represent numbers from 0 to 15. If you take n equal to 3, then you can represent the number 0 to 7 if you take n equal to 5 you can represent numbers from 0 to 31.

So, for 136 what will be the value of n? So, value of n will be the next power of 2 after 136 ok. So, the next power of 2 a, 2 is 256 that is n equal to 8. So, I will need an 8 bit numbers system to represent 136. So, in a 8 bit number system what happens to the values like I have got the first value is if I right from the LSB side, first is one, next is 2. So, I am just writing this position that we have written here say 1 2 4 like that. So, if you are writing like this 1 2 4, then 8, then 16 32 64, 128 256. This is that 8 bit range that we have.

Now, for the number 136, we do not need this 256 bits. So, this is 0, fine? Now 128 I need because I want represent 136, 128 I need. So, after I have taken 128, I am left with only 8 ok. So, 64 I do not take I do not take 32, I do not take 16, I have to take 8 after that value has become 0 so, none of this bits are necessary.

So, 136 is actually represented as the bit pattern 0 1 0 0 0 1 0 0 0. So, this. So, 8 bit so, I am sorry this 120 after, this 256 will not come here, because I am going up to n equal to 8. So, this is 1 2 3 4 5 6 7 8 so, after this much will be necessary. So, this will be up to 128 will be necessary. And in that 128 so, this oneth for representing 136 128 will be necessary so, this 0 is also not there. This 136 will be necessary, after that I will be taking this only 8 will be left so, this 64 32 16. So, they will be not be necessary only, this 8 will be necessary again, ok.

(Refer Slide Time: 14:21)



So, let us take another example let us take another example say 175. So, for 175 again the same thing that the next power is 256 so; that means, I need an 8 bit representation. So, 8 bit if I take so, 1, 2, 4, 8, 16, 32, 64, 128. So, these are the 8 bit weights.

So, for 175 I will need this 128, after I have taken 128. So, 175 minus 128 so, that gives me 7 47 so, more 47 more has to be represented. So, I cannot take this 64 so, that is 0. I have to take this 32 so, if you have subtract 32. So, you are left with 15 so, I do not take this 16, I take this 8 so, minus 8. So, left with 7 so, I have to take this 4, minus 4 and left with 3. So, I take this 2, minus 2 and left with one, I take this one, minus 1 left with 0.

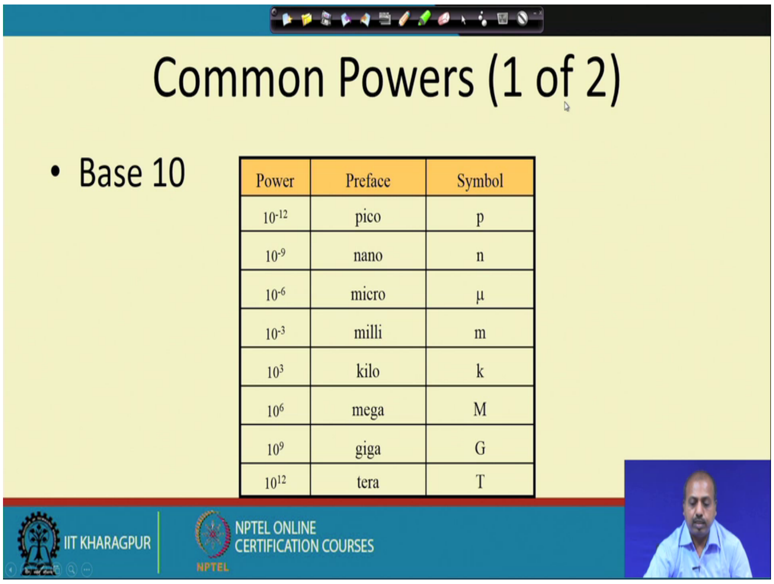
So, the 170, the value 175 in the binary number system will be given by 1 0 1 0 1 1 1 1. So, you can check the corresponding decimal value to be equal to 175. So, this way we do not need to always do that deviator division by 2 and all. So, if we just remember these powers of 2, and we can just go and assigning we can take go on taking those positions by the turning on the corresponding bits to 1, ok.

So, here also is 33, say 33 means, I will requiring say 6 bits representation. And in that 6 bit representation so, this is these value is 1. So, this is 2, this is 4, this is 8, this is 16, this is 32 so, 32 plus 1 33, 1 1 7. So, I will need 7 bit representation, because 2 2 the power 7 is 128. And then this is again the thing 1, 2, 4, 8, 16, 32 64. So, say 64 plus 32 is 96 plus 16 is 112, plus 4 116 plus 117 so, that gives us 170.

So, this way I can do the conversion, and once you have convert it into binary, then octal and hexadecimal conversion conversions are very simple, because for octal conversion what I need to do is I have to make groups of 3. So, I make a groups of 3 here I make a another group of 3 here. So, the first group is one and the second group is 4. For hexadecimal I make a group of 4, and then I make a group of 4; like this so, that way it is 21, ok.

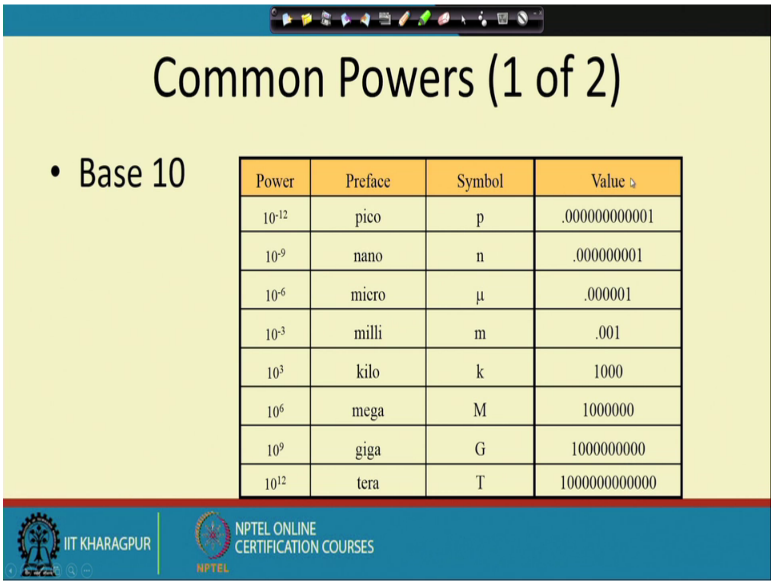
Similarly, here I make groups of 3 so, this is one group, this is another group and this is another group. So, I get 165 in octal and if you are taking hexadecimal, then this is one group and this is another group. So, the first group is 1 1 1 0 so, sorry 0 1 1 1 so, that is 7, and the next one is 0 1 0 1 that is 5. So, this way you can oh 1 1 you can very easily to a conversion from decimal to binary by taking help of that number scale that powers of powers of 2 scale. And then from their you can go to octal and hexadecimal numbers very easily. Do not need to do those multiplication divisions and all.

(Refer Slide Time: 18:25)



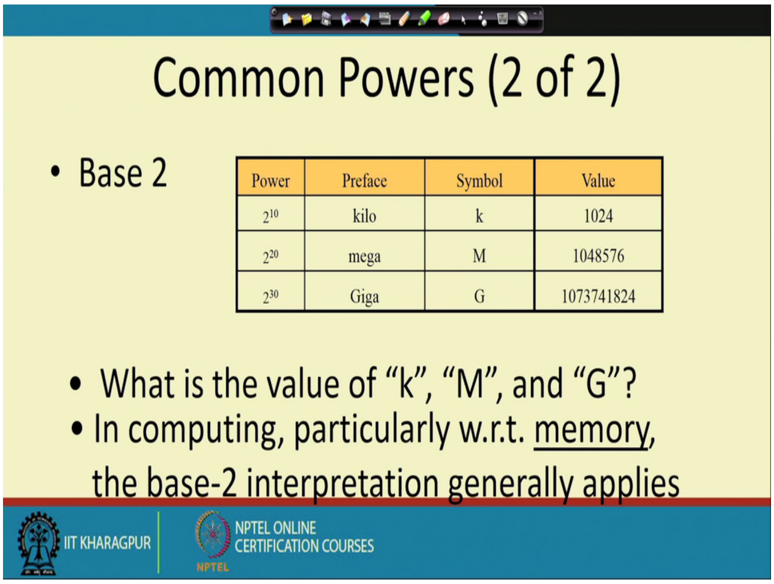
So, next we common powers so, this is these are the different names that are given base 10. So, 10 power minus 12 is called pico 10 power minus 9 is called nano minus 6 is micro. So, these are the standard thing up to 10 power 12 is tera. So, in representation we represent 10 by small p small n, I have then micro small m, small k for kilo then capital M for mega, capital G for giga, capital T for tera. So, this is so, this is common powers of base 10.

(Refer Slide Time: 18:59)



And the corresponding values are like this ok, so, 10 power minus 12 will be this on so, this is known to us.

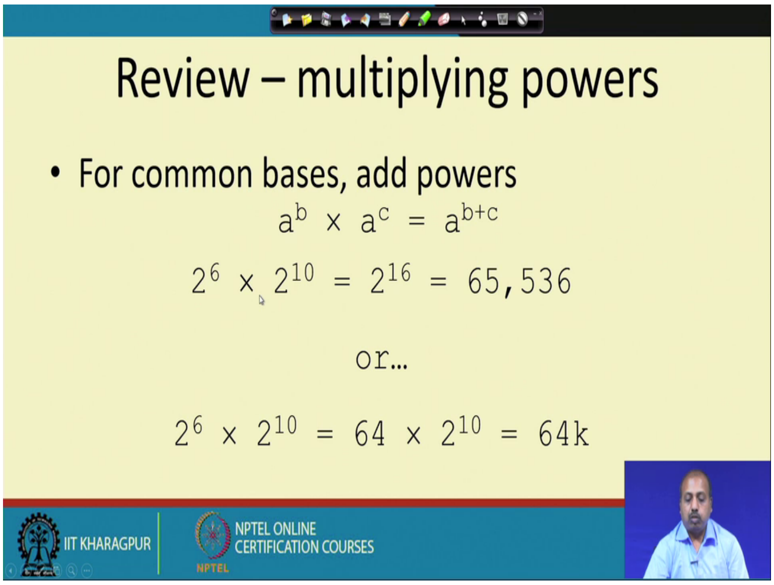
(Refer Slide Time: 19:06)



So, if a base 2 so, we have got 2 power 10, 2 power 20, 2 power 30, they are called kilo mega and giga. So, those are the corresponding symbol and the values are 1024, then this one and this 1 2 power 20 2 power 30 like that very big numbers. What is the value of kilo mega and giga? So, these are the values actually.

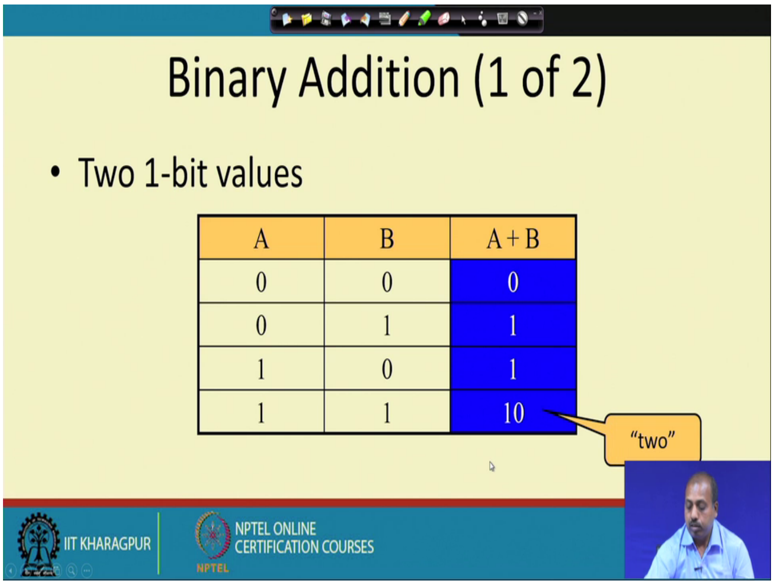
In computing so, particularly with respect to memory so, will be using base 2 interpretation. So, in incase of by incase of decimal number system so, kilo means thousand, but in case of binary number system. So, the kilo is 1024, and in computer systems or in digital processors. So, whenever we are talking with respect to memory so, will be talking about kilo as 1024, and then mega is 104857 that 2 power 20, 2 power 30; like that, they are not the powers of 10, ok.

(Refer Slide Time: 20:05)



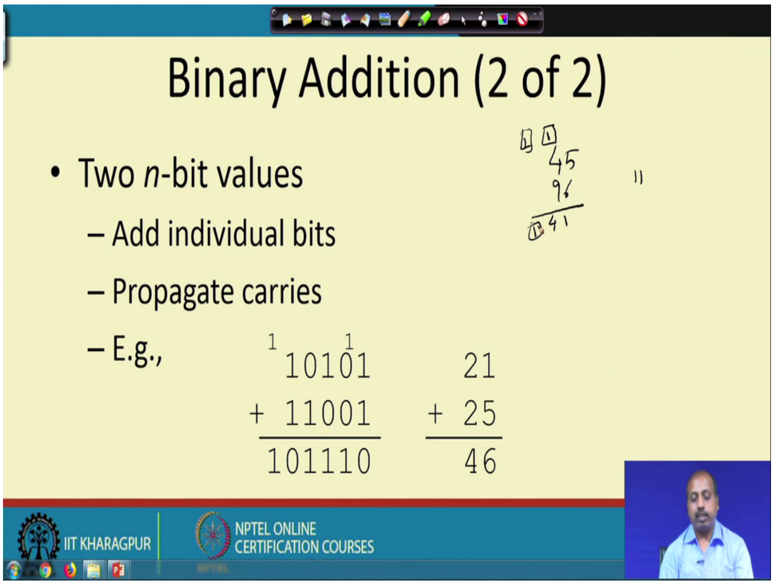
So, some common rules, like say for common basis we can add the powers, like if you are multiplying 2 numbers a to the a to the base a to the power b and a to the power C, then we can add the power so, we can say a to the power b plus C so, 2 power 6 into 2 power 10 is 2 power 16. So, 65536 or say 2 power 6 into 2 power 10 is 64 to the power 10. So, I can, I the convert it in this and then you can say it is 64 k. So, either we can visualize it like this 65536 or we take it to the power of 2, ok. So, it say that it is 64 k, where the actual value is 65536.

(Refer Slide Time: 20:50)



So, when you are adding binary numbers to add 2 one bit values so, if this is the rule. So, we both the bits are 0 a plus b 0 plus 0 is 0 0 plus 1 is 1, 1 plus 0 is 1, and 1 plus 1 is 2. So, which will be requiring 2 bits for representing it so, that is 1 0 so, that is 2.

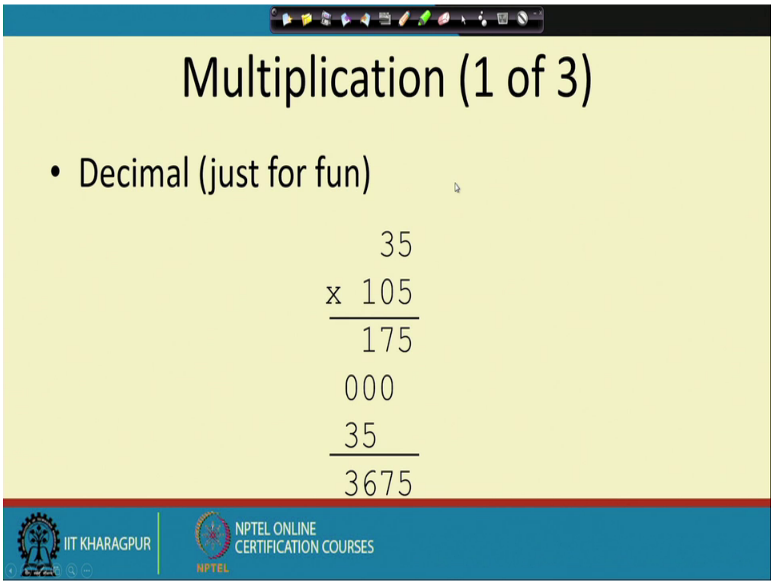
(Refer Slide Time: 21:12)



So, if you are adding 2 n bit values so, you have to add individual bits and propagate the carries just like in decimal addition so, we are propagating the carries, here also the same thing, like in decimal system; so, if you are adding say 45 and 96, what are doing? So, 6 plus 5 is 11 so, we write 1 here, and this 1 carry is taken to the next position. And then 9 plus 4 plus 1 so, that is 14 so, 4 is written here, and one is taken has a carry and then that carry appears at this point.

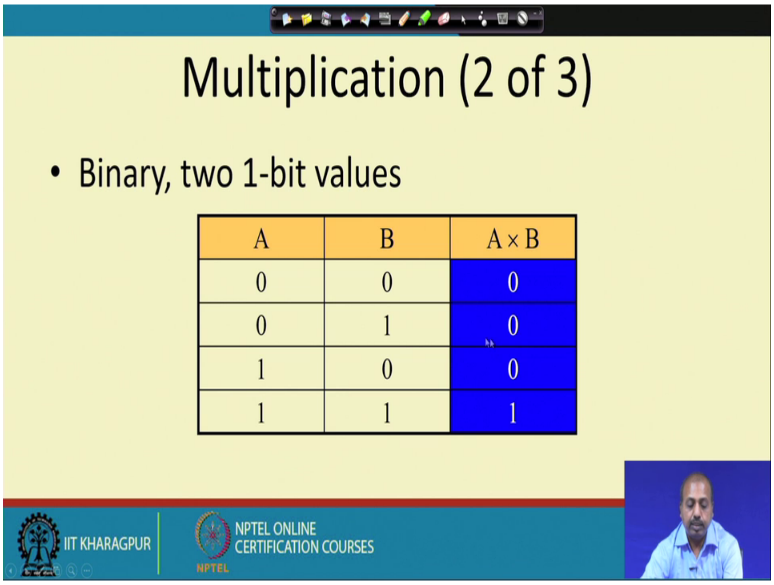
So, that way we can have the same thing in case of binary number system as well. So, we can say that we can add individual bits and propagate the carries. So, here 1 plus 1 is 0 and carry is propagated then 1 plus 0 plus 0 is 1. So, there is no carry, then 1 plus 0 is 1 0 plus 1 is 1 1 plus 1 is 0 and this one is propagated so, this is one. So, this gives us the number 40 6 so, that is you can check it that this is really 46.

(Refer Slide Time: 22:21)



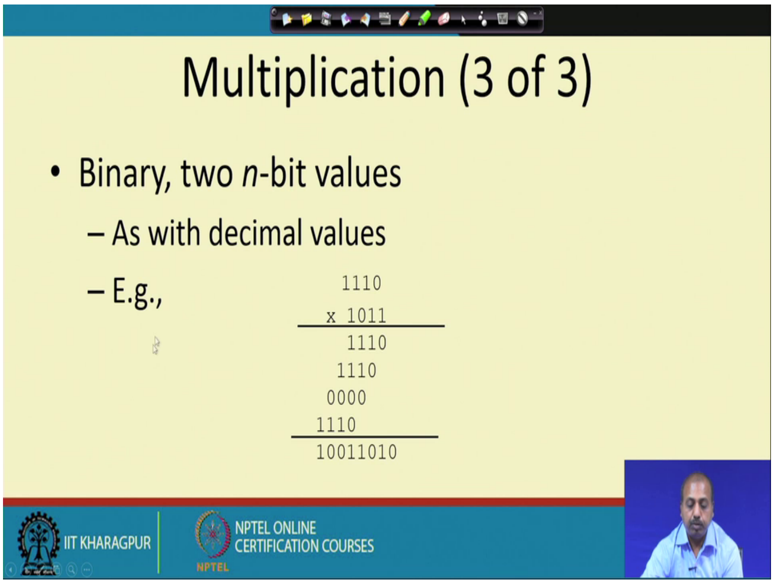
Multiplication so, you can multiply decimal number, like this so, this is the way you do multiply. So, first the we multiply whether first digit, then do a shifting multiplying whether second digit, then do a another shifting multiplying where first digit, and then we just some all this partial result.

(Refer Slide Time: 22:45)



So, to get the overalls overall multiplication value, here also in case of binary number system. So, the rule is like this 0 into 0 is 0 0 into 1 is 0 1 into 0 is 0 and 1 into 1 is 1.

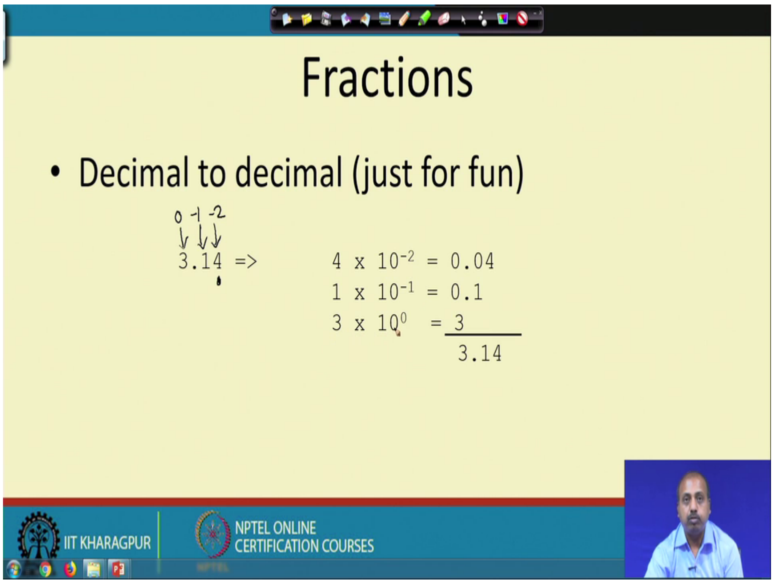
(Refer Slide Time: 22:55)



And then if you are doing some multi bit multiplication, then we have to do the similar rules that we do for the decimal 1 so, first this is multiplied by once. So, if it is a one basically you get back the other numbers so, this 1 1 0. So, then after that there is a shift, for the second bit multiplication so, you get again get 1 1 0. Then this is another shift, but this is all 0 this multiplying by 0. So, all are 0's, and then another shift and multiplying by 1 so, you are getting 1 1 0. So, ultimately you are getting this one as the result.

So, this way can the rules of multiplication addition the remain same.

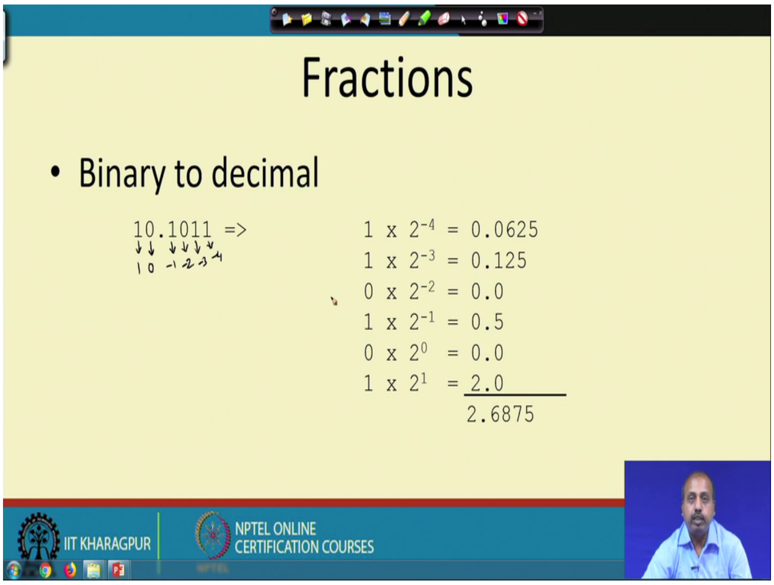
(Refer Slide Time: 23:31)



Now, for the fractions ok. So, how are the fractions represented? So, just for fun so, you can look into the decimal fraction, likes how they are actually what is the interpretation of this fractional numbers in decimal numbers system. Like, we represent something like 3.14, and then what we do is that we say if we are trying to assign weight so, you start assigning weights from this less significant position. So, so, this is this position is 0, this position weight is 0. So, after that so, after the decimal point, the first one this weight is minus 1, the second one weight is minus 2, so, it goes like this.

So, in case of for the sake of conversion so, you can say it is 4 into 10 power minus 2 plus for 1 into 10 power minus 1 plus 3 into 10 power 0. So, ultimately you get 3 point so, this is the decimal interpretation of this fractional numbers. So, whenever we converting binary to decimal so, the same thing has to be done.

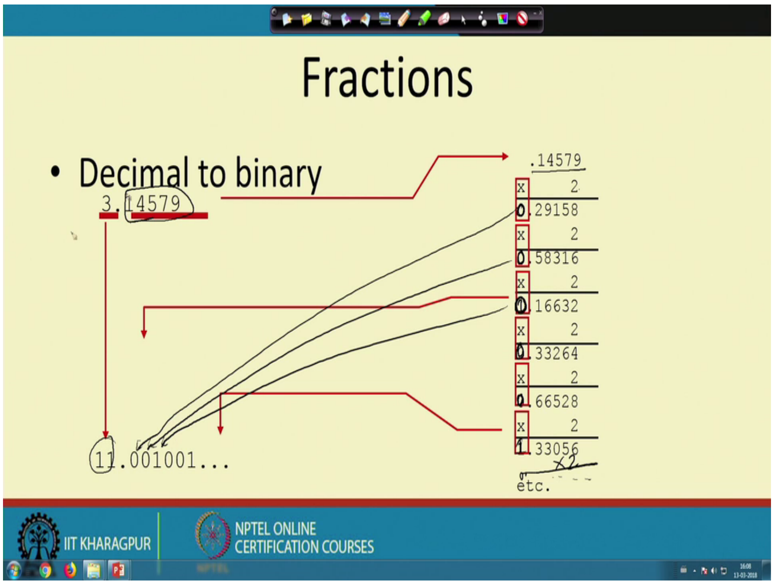
(Refer Slide Time: 24:38)



Because for so the, you can say that for this 1 0 1 1 part. So, this one so, what is the weight? So, if if you just right down so, this is the weights are; so, this is this is this weight is 0 so, this is minus 1, minus 2, minus 3 and minus 4, so, this weight is 1.

So now the rule is straight forward so, one into 2 power minus 4, plus 1 into 2 power minus 3, plus 0 into 2 power minus 2, plus 1 into 2 power minus 1, plus 0 into 2 power 0, plus 1 into 2 power 1. So, from right to left so, you just some them up. So, it then after getting the values so, you just take the sum so, the value is 2.6875. So, this way you can convert the binary numbers so, which is a fractional number into the corresponding decimal values.

(Refer Slide Time: 25:47)



For decimal to binary conversion the rule is just the reverse. So, you have to take the 2 parts separately so, what we do is for this decimal part, for the integer part so, we take it we convert it separately, and for the fractional part we do it separately.

So, for this 3 ah, if we if we are doing it if we are doing for 3. So, 3 is represented as 1 1, 3 is represented as 1 1. And for this 0.14579 so, we go on multiplying it by 2. So, 0.14579 multiplied by 2 it gives 0.29158, the portion that comes before this decimal point is 0. And then this 0.29158 is again multiplied by 2. So, you get point 0.58316, again this value that that is getting is 0. And again you have multiplying it by 2 so, this is getting 1 1.16 something.

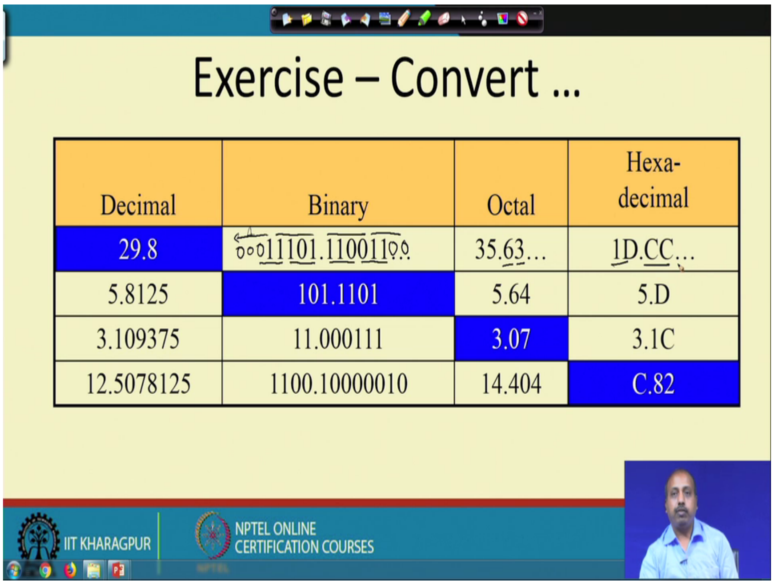
So, this is so, this value is 1. So, the portion that is coming before the fractional point after doing the multiplication is taken into consideration. And this value we go on multiplying so, this is multiplied by 2 again. So, you get a 0 here after that point 3 3 2 6 4 multiplied by 2 so, you get 0 point something. Again multiplied by 2 so, you get one point something so, it is not ending.

So, after this so, you can continue further, you can continue further multiplying again by 2 so, you will get is this before decimally you will get a 0. So, this way it can continue so, it is not ending here. But ah, but so, you so, the so, if it ns at some point of time everything become 0. So, you can stop at that point, otherwise it can go till infinity, but definitely for computer system. So, there is a finite storage. So, you cannot representing infinitely all the bits after the decimal point. So, we have to stop at some point of time.

So, if you say that will stop after so many bits. So, this is so, this is point 0 0 1 0 0 1. So, for the portion before decimal point so, you divide by 2 for the portion after decimal point, you divide by a multiplied by 2 ah. So, this 0 that you have got, it is coming into this 0 is coming into say this 0, then this 0 is coming into the next 0. This one is coming as the next one so, it comes like this successive bits they will be coming to the binary system like that.

So, in this way you can represent fractions in the binary in the binary number system.

(Refer Slide Time: 28:43)



So, this is an exercise so, 29.8 so, for 29.8 so, if you convert 29 into binary. So, you this is the number, and 0.8 if you go on multiplying by 2 so, 0.8 into 2 is 1.6. So, you get the one here now 0.6 that remains. So, multiplied by 2 so, you will get 1.2's again get you will get a one here. Now 0.2 is remaining, 0.2 multiplied by 2 is 0.4. So, you get a 0 here that 0.4 multiplied by 2 you will get again 0.8. So, that 0 comes here, the 0.8 multiplied by 2 you will get again point 1.6 the one comes here so, that way it goes on.

So, this way we can convert this decimal numbers into binary numbers. Now once the decimal number has been converted into binary so, you can do the conversion to other number systems; like, here so, to convert into octal number system so, what we are doing? So, we are making groups of 3. So, you make a group of this 3 bits, and then we make a group of this 3 bits for the integer part. So, this is 35, for the fractional part also we do grouping, but here grouping starts from this side.

So, grouping should starts from left side. So, this is one group, this is the next group. So, first group is giving the number digit 6 next digit may be giving as 3. For hexadecimal so, you take groups of 4. So, that is the first one so, for the integer part, you start from the less significant bit position. And here you do you take 2 more you take 2 more 0's. So, that way you get this part is one and this part is 8 plus 4 plus 1 that is 13; which is D, and for this fractional part you start grouping from this side. So, this part we group this 4 bits then you take 2 0's and you group this part.

So, this is 8 plus 4 12, that is C so, you get one D CC. So, this way you can do this conversions, so, for you can convert the decimal number to binary number first, and then from there using the conversion rules. So, you can go to octal number or hexadecimal numbers.