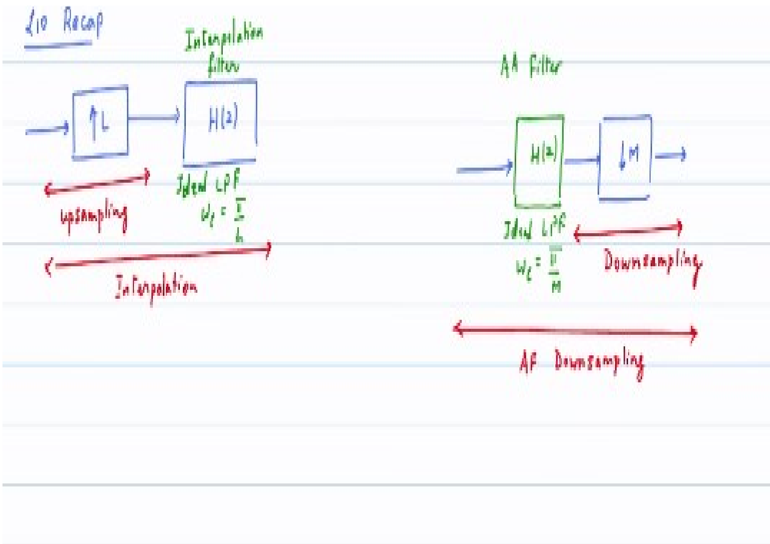
Multirate Digital Signal Processing
Prof. David Koilpillai
Department of Electrical Engineering
Indian Institute of Technology – Madras

Lecture – 12
Multiplexer/Demultiplexer Interpretation

Good morning. We begin lecture 12, the plan for today's lecture is to introduced some very interesting results from multirate signal processing which you will find are very powerful and also give us the basis for a lot of the interesting results and applications that we develop in multirate signal processing. So let me just summarise what we have said so far in lecture number 11 and the one earlier than that so that we can get our lecture number 12 content moving forward.

(Refer Slide Time: 00:55)

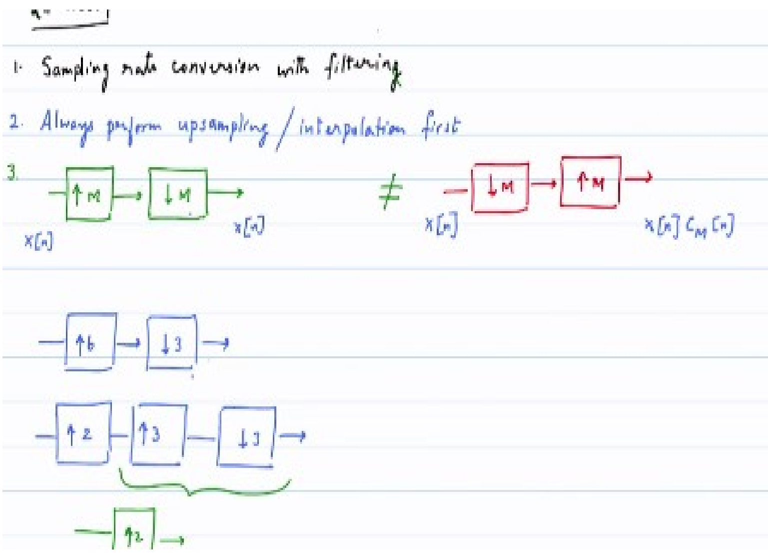


So the main element from lecture 11, lecture 11 recap would be 3 points, which I would like you to keep in mind. One is that the sampling rate conversion always goes with some form of filtering. Sampling rate conversion + filtering and the important point to note is that both of our sampling rate blocks, upsampling and downsampling actually are useful with the presence of the filtering.

And that again is a slide from the last lecture where we said that upsampling gives you multiple copies of the images which you remove through ideal low pass filter. So the combination of up sampling followed by filtering gives us a useful signal which is at a higher sampling rate and likewise when we look at the downsampling, the down sampling process can be done without any reference to the input signal or it is bandwidth.

However, it is always advantageous and we have shown that this is a sufficient condition to show that if you want to have alias free down sampling then you should limit your signal to within a bandwidth that is pi/M. So these are the 2 combinations, sampling rate conversion combined with the filtering. So that is with the corresponding filtering process. So keep that picture in mind.

(Refer Slide Time: 02:19)



Then we also did a fractional sampling rate conversion. So where they was both upsampling and downsampling to be done. So in this case we also made the observation that it is always better and the right thing to do is to perform the upsampling first, upsampling by itself is not complete actually with the filtering you should actually call it interpolation. So the combination of upsampling and filtering is what we are referring to and that should be done ahead of any downsampling okay.

So that is the second observation and we showed that if we did it in the other order then it actually gave us some erroneous results and the third observation that we made was the following. It is actually a very important result in multirate signal processing, though might seem like a fairly simple and obvious result, but actually becomes a very powerful result in our applications.

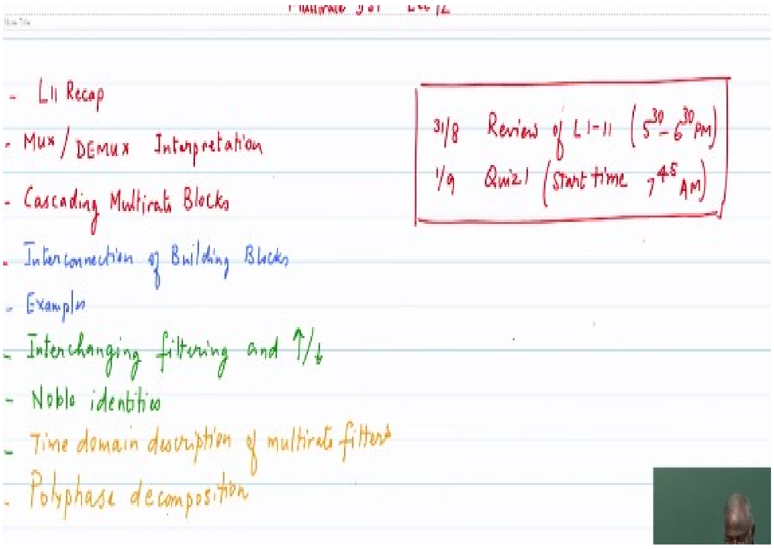
So this is a combination of, cascading of an up sampler followed by a down sampler. No filtering, only upsampling and downsampling. So if I had X of N at the input, we showed that the output was also X of N. This sequence operation basically they cancel each other. So anytime you have a upsampler followed by downsampler they cancel each other. This also leads to an interesting observation.

So if I have upsampling by a factor of 6, downsampling by a factor of 3 okay and as you will agree, that up sampling by a factor of 6 is I am going to insert 5 zeros between every pair of samples. Now this can also be done as 2 stages, upsampling by a factor of 3, upsampling by a factor of 2 or maybe let me do it in this order. Upsampling by a factor of 2, upsampling by a factor of 2, upsampling by a factor of 3.

And followed by downsampling by a factor of 3 and the result that we have shown basically says that the combination of these 2 cancel each other. So effectively this block can be represented or the upsampling and downsampling by a factor of 3 is effectively upsampling by a factor of 2. Again in this particular example it is probably very obvious to see, but there are structures where when you see these multiple operations and you can find that you can take advantage of this result, it is a good thing.

We said that this combination is not equal to, not equal to sampling rate conversion in the other direction. If I do the downsampling first followed by the upsampling, in this case if I had X of N as my input, the output is the input signal multiplied by the comb sequence and therefore we made the statement that these 2 are not the same, but this is a useful way for us to simplify, multirate structures when the 2 of these blocks come side by side, it is an advantage for us to use these results okay. So that is a summary of lecture number 11.

(Refer Slide Time: 05:49)



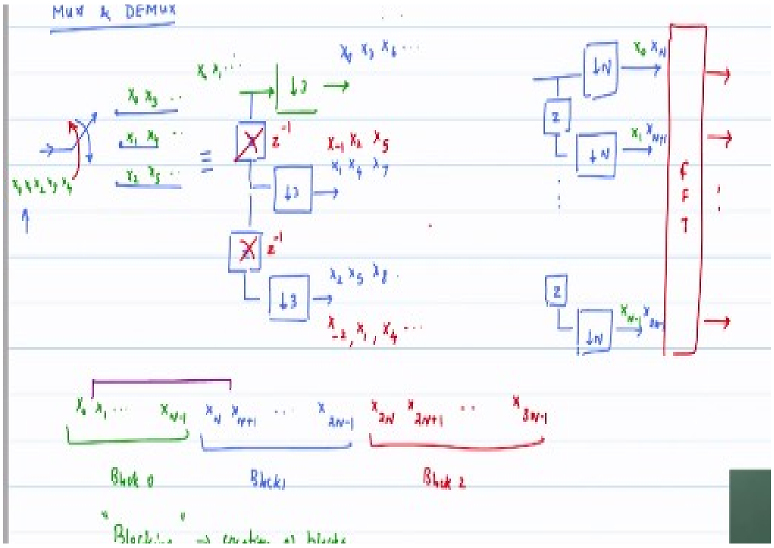
We will now move into looking at some of the newer aspects of lecture number 12. So lecture number 12, I would like to first and foremost introduce you to an equivalent representation of upsampling, downsampling using blocks that you are familiar with from communications, multiplexer and the demultiplexer. Then we already have started to talk about cascading of multirate blocks.

So this is what we would referred to as cascading of multirate blocks. So we will look at the more general case where you're upsampling by L, downsampling by M, we will look at the inter, and then will also look at interconnections. What happens if you have multiplier, what happens if you have an adder. Some simple examples.

The interesting result start to come towards the latter half when we start saying okay, if I have filtering and upsampling, downsampling okay. What are some of the ways in which multirate signal processing can help me take advantage of that, that leads to a very interesting result called noble identities and then time permitting we will also develop this concept further.

So let us dive into the content for today's lecture. The first one that I would like to get into is a discussion of the multiplexer, demultiplexer okay.

(Refer Slide Time: 07:03)



So I am sure that from communications you are very familiar with the basic building block of a multiplexer and demultiplexer. So basically a multiplexer, let us take the demultiplexer. Demultiplexer takes a high rate stream and splits it into multiple lower rate streams. So typically the way you would draw a demultiplexer is the high rate stream is coming in and then I have several outputs.

And each of these are at lower rate. So basically the demultiplexer will feed these, will split the input signal in the time domain into multiple parallel streams. So basically what have you done you have gone from a higher rate input signal to a lower rate signal, okay, multiple of them. So now which of the building blocks gives us the higher rate to lower rate conversion? It is the downsampler.

So is the downsampler in some way, in some way again maybe not in an obvious way but in some way equivalent of a demultiplexer. So basically if we had a stream coming in, we use a different colour. If I had stream coming in which was let us say labeled in the following way, X0, X1, X2, X3, X4 and so on and let me take the case where there were only 3 blocks, 3 channels.

So I will erase this also, draw a third line which is here okay. So basically it is a demultiplexer, 1:3 demultiplexer and so the way you would expect is at X0 gets fed on one branch. X1 gets fed on other branch, X2 then you go back to the first line. This is X3, X4, X5 and so on, okay, and it is very easy to see that yes this is exactly what will happen if I did downsampling process as well.

So basically if I took a input signal and downsampled by a factor of 3. So if this is X0, X1, the same input signal as before, this output would be X0, X3, X6 and so on, okay. But in a demultiplexer case, I need to get the other signals as well. So in fact we did look at this particular example. How do I get X1 in the lower branch, is it a delay or an advance? Let us try both.

So if I do an advanced operation, advanced operation, so then what happens the originally this was the origin. If I do an advance operation the origin shifts to the right. So which means what comes into the downsampler at the zero'th index is X1 and if I now downsample by a factor of 3 and look at this, this will be X1, X4, X7 and so on. Okay so it is not a delay element, but actually an advanced element.

So and likewise just draw one more block that will actually be the one that gives you the third demultiplexer branch that would be X2, X5, X8 and so on. So these actually are equivalent blocks. So 1:3 demultiplexer is the same as doing a multirate structure again it is something that actually is part of the tool kit that we are developing. So the representation in multirate is useful and interesting.

So just since we also made an observation that if it was Z power -1. So if I had deleted this and made it Z power -1, this also as Z power -1 can you tell me what would be the output sequence. So now what would happen the origin would have moved 1 unit to the left that means X of -1 will now come into the, so X of -1 will be the first output. X of -1 then X of 2, X of 5 right dot dot dot.

This would be X of -2, am I right, X of 1, X of 4 dot dot dot. Is it multiplexer, demultiplexer? Answer is yes. The only difference is instead of going clockwise the demultiplexer is going counterclockwise. So basically starts of at the upper branch then swings around to the lower branch because X of 0 goes on the upper branch. X of 1 comes on the lowest, then comes to X2, X3 and.

So this chain can be advanced operators or it can be delay operators, does not matter it gives you a very elegant interpretation and useful representation as well. Now one of the most important or useful applications of this particular interpretation comes in the following context. Now very often when you do data analysis, you will do FFTs, that is usually very common in signal processing.

So I have date of this form X0, X1, all the way to X of N - 1 then X of N, X of N +1, dot dot dot. X of 2 N-1, X of 2N, X of 2N +1, all the way to X of 3N -1 dot dot dot okay. So now we treat this as block 1 or block 0. The second one as block 1, okay. We actually do this, we take blocks of data and then we do the data analysis okay.

And this is block 2 and of course when you write code in matlab or C you know that you have to be, you have to take care of it very carefully in terms of the indexing so that you do not make sure that you are breaking at the correct point. So this operation is actually has a name it is called the blocking of data. Now very different from blocking in communications. Blocking means the call does not go through.

This has not nothing to do with calls or connecting calls, this is actually breaking up data, creation of blocks. So this is creation of blocks and the reason we do that is because we want to run the FFT okay. Now when you actually implemented in hardware, it is very useful to actually drop on the multirate equivalent of this because you can actually implement it in the hardware in a very elegant manner.

So how this would be done? Basically create blocks of data and compute it is FFT if you wish. So the way we would interpret it is you take your input signal downsample by a factor of N, because you are going to now do blocking by a factor of N. So then the first one would be an advance operator of Z followed by downsampling by a factor of N and like this you put N -1 blocks, all of them down sample by a factor of N okay.

Now what is coming out at this point just we will write down of the first block, if this is X of 0, this will be X of 1, this will be X of N-1 okay. So what you do is put the DFT block here, DFT or FFT that will produce for you the corresponding outputs which you can then compute. Next instant of time what you need is the block starting from with X of N, X of N +1, and that is exactly what will happen.

Because the next instant of time this will be X of N, this will be X of N+1 and this will be X of 2N-1 okay. So every tick of the clock you will find that one vector comes to the input, you compute FFT and then you push it out. So again from a hardware implementation point of view. From a software point of view, this is fine, we can just index it and compute FFT, but you actually want to implement it.

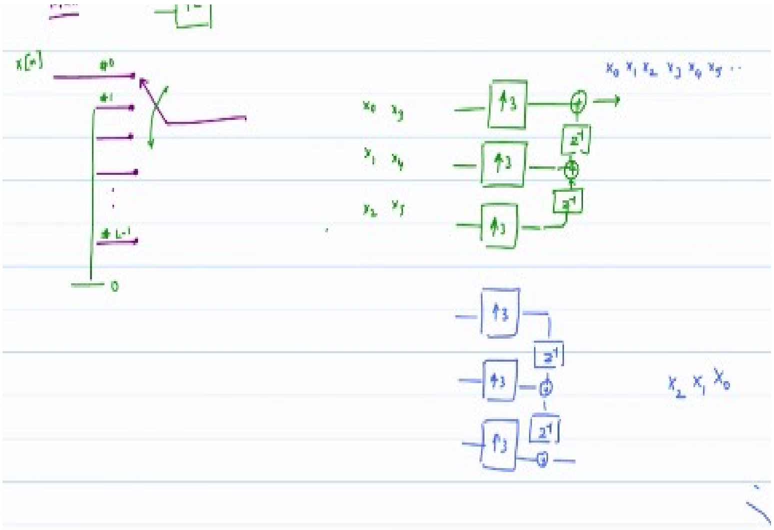
Down sample, this would be a very good way for us to visualize and also implement the down sampler okay. That's is a good question, the question is why do I split it as non-overlapping blocks, that is your question, okay. So yes the it is a very good question, basically the answer to that is, very often when we have large segments of data we say that okay I am interested in spectral analysis and so we take segments of data and analyse it.

Now nothing prevents the second block from being defined in the following way. You could have defined your block number 2 in this form, starting from X1 to XN, that is called a sliding FFT okay. So this is the block FFT versus sliding FFT. If you wanted to have more fine resolution of your data, if you for example thought that your signal is not stationary, if you thought that your signal was quasi-stationary that means things could change.

So between block 0 and block 1 something could have happened that could, you are trying to find out those non-stationary elements, in that case you will do a sliding FFT and that is equally permissible, again, do we have a multirate structure for that? Yes we do, but let me ask you to think about that, but in terms of applications very often if it is stationary data then we are quite happy to look at blocks which are non-overlapping, and of course we can look at ways of modifying the structure if we want to do the sliding FFT as well.

Okay but good question, it is more of what is your data like, what is the application are you looking at quasi-stationary or stationary, in which case then the approach or the way you do the blocking may vary.

(Refer Slide Time: 18:21)



Now let me also add one more element, the counterpart of the down sampler, and that is the upsampler. So if this was a demultiplexer then the other block must be the multiplexer, the multiplexer structure basically says that there many lines coming in, many lines coming in and you will have a rotating switch which connects all of them. So basically I have a rotating switch which will connect all of them.

And produce an output which is at a higher data rate, okay. So many parallel lines are available from which you will do the multiplexing okay. Now the upsampler by a factor of L says that there are L lines. So this is line number 0, line number 1, this is line number L-1, now this has the data X of N, those samples must go through and I have to insert L-1 zeros. So what I say is connect all of these and connect them to circuit notations like ground.

So basically that has a value 0, so basically this one can rotate counterclockwise you take the input sample from the input stream and then you insert L-1 zeros and then pick up the next sample against L-1 zeros again systematic way of operation. Now very often we also want to reconstruct the blocks, we want to reconstruct the blocks. So if you remember how were the blocks coming in after the blocking operation, we say that it was X0, X1, X2 and then the next instant of time it will be X3, X4, X5.

So basically blocks of data are coming in, if I wanted to reconstruct then the upsampling operation with the interpretation that we are looking at, you insert, do an upsampling operation okay that will make sure that you are inserting the number of zeros appropriately.

To insert X of 1 you up sample by a factor of 3 and then you have to pass through a delay. So that it actually gets into the right position okay.

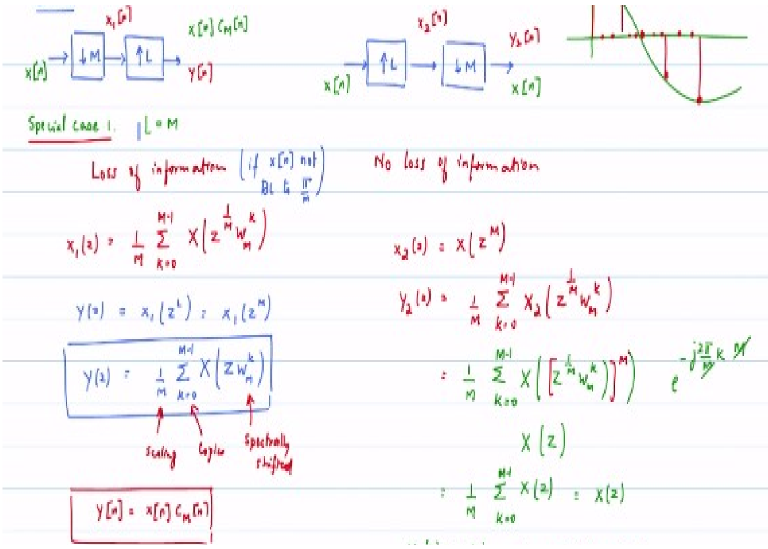
So this is an addition sign and the third data point also, upsample by a factor of 3 will make this also an addition with a another delay element and what you get here, what you will get here is X0, X1, X2, X3, X4, X5 and so on. Okay, so basically this is the unblocking, basically it creates, on the blocks it creates these stream of data that we are interested in. Same thing that we would like to look at little bit more carefully.

Now if these signals had gone in the other direction basically the arrows have gone in the other direction. This is upsampling by a factor of 3, upsampling by a factor of 3, upsampling by a factor of 3 okay and it goes in this direction; Z inverse okay, add okay, what would this produce X0, this also produces the right thing? No it actually X0 gets shifted by 2 units of time right.

X0 gets shifted by 2 units of time X1 gets shifted by 1 unit of time, X2 does not get any shift correct. X2 does not get any shift. This is not the right sequence in which we want and therefore this is not a useful form, but there are structures which we will see where instead of the arrows going up we actually will do the arrows coming down and when we come at this point, if you do blocking in this manner.

If you do blocking in this fashion, the correct structure is this one, this is not the right structure, but if you do blocking in a slightly different way, this turns out to be a good one may be just think about that okay. So I think the relationship between downsampler and the corresponding demultiplexing, the upsampler and the multiplexing, I hope is clear and definitely we will utilise this in some of the structures that we will build up okay.

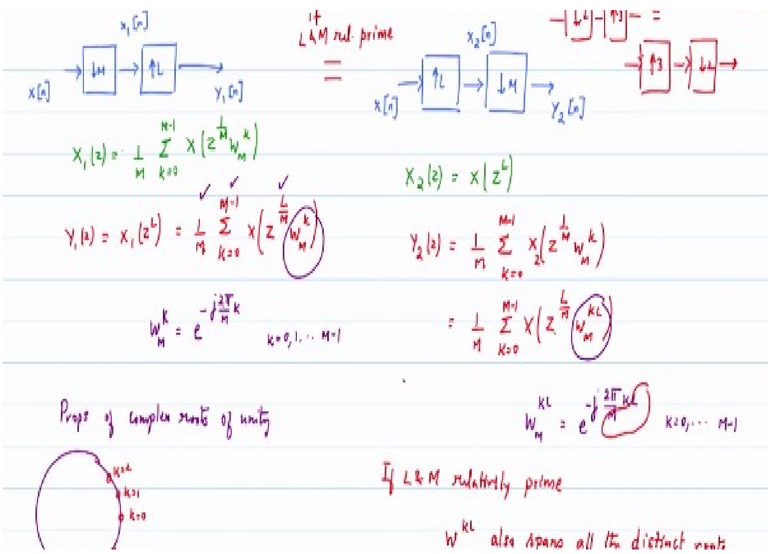
(Refer Slide Time: 22:58)



Just to refresh your memory this is the slide from the last lecture, just wanted to highlight, this was the case where we took L = M, L =M and what we would now like to do is look at the general case where L not equal to M. So basically follow the same method but L not equal to M, but rather than ease this one and cause confusion, I thought I will just write the results down and I think you may be able to do it very easily as well.

So the key things that we wanted to look at are the interconnections of these 2 blocks when L and M are different okay. And we have not said anything about whether they have common factors. We will make an statement about that towards the end.

(Refer Slide Time: 23:45)



Downsample by a factor of M, upsample by a factor of L. No filtering only the sampling rate conversion and if this is X of N, X1 of N, Y1 of N okay. So write it in the Z domain, or want to look at the transform expressions, X1 of Z, downsampling, I get 1/M summation K = 0 to M-1, X of Z power 1/M WMK. Please make sure that the W is related to M, because the downsampling factor is M.

And the next step is straight forward, Y1 of Z is X1 of Z power L, okay. I already have an expression for X of Z. So if I want to treat the new variable as Z power L, wherever there is Z, I replace it with Z power L. So this will come out to be 1/M scale factor summation K = 0 to M-1 X of 1/Z raise to the power L, Z power L/M WMK okay. Please be sure that no confusion with respect to the how we have obtained the expression.

On the other side if I move over and draw the interchanged sequence, upsampling by a factor of L, downsampling by a factor of M, again I will leave you to quickly write down the expressions, I will just write the final answer. If this is X of N same input, I will just give different labelling for the outputs because I do not know if they are equal or under what conditions they will be equal and the intermediate signal also I am going to call it as X2 to N okay.

So the first step X2 of Z is upsampling, X of Z power L, now the next equation is the downsampling Y2 of Z is 1/M summation K = 0 to M-1, to M-1 X of Z power 1/M WMK, X2 of sorry. So I know I have an expression for X2 of Z. So the operation now please pay close attention. The argument is Z power 1/M WMK, so X2 of the argument is X of the argument raise to the power L.

So then this becomes 1/M summation K = 0 to M-1, X of the argument of X2 of, X2 raise to the power L which will be Z power L/M WMKL. So let us do a comparison 1/M present on both sides, summation is present, Z power L of M is present. The only thing that is different is the shifts okay. Those are the shifts are different. So what are these shifts, WMK, WMK basically corresponds to E power – J 2 pi/M times K, K = 0, 1, all the way to M-1 okay.

Now when I look at this side expression, it is WM KL that will be E power –j 2 pi/M times K times L, still K goes from 0, 1 to M-1 okay, but except that it is already always being multiplied by the factor L. Now if you go back and look at the roots of unity, property of the roots of unity, complex roots of unity. Properties of the complex roots of unity. So this set what we have here this corresponds to K = 0, K = 1, K = 2 and so on.

And the last one, so this will be K=0, K=1, K=2, this is K=M-1. Basically you are spanning all the roots of unity okay. Now I am sure you would have come across the several interesting theorems and results with the complex roots of unity. Now if L and M are relatively prime. If L and M are relatively prime that means there is nothing that you can cancel between these two okay, L and M are relatively prime.

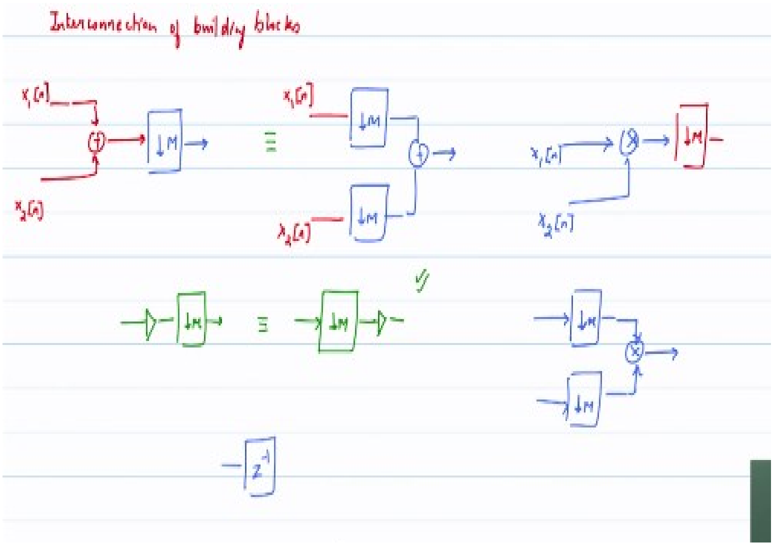
Then omega, sorry, WM raise to the power KL also spans all the roots of unity, also spans the roots of all the distinct roots of unity okay. So in other words that is the only condition under which they will span, because if L was a factor of or if they had a common factor you will find that it does not span all the roots of unity. It will only span a subset, okay. So very interesting result that emerges these 2 may be equal, okay.

We do not know but you have to check for different values of M and the equality holds if L and M are relatively prime. That is a very powerful result. So which basically says that for example if I have downsampling by a factor of 2, upsampling by factor of 3, it is the same as upsampling by a factor of 3 followed by downsampling by a factor of 2, upsampling okay, actually equal.

Then why did the sampling rate changed not work. When we did the sampling rate change 3/4 we did it in the different order and it did not work. If 3 and 4 are relatively prime by which argument I should have been able to do the operation in any sequence, the filtering made the difference. Basically this argument does not include any filtering, okay. If you include the filtering, then always you have to do the upsampling first because they are not the same when it comes to the.

So keep in mind again simple result but something that I believe will be a fairly powerful result, for us as we go forward. Let me quickly cover a couple of, these are all very simple results. So interconnection of building blocks.

(Refer Slide Time: 31:25)



Again this is more for completion that we are, you should never have a doubt okay, can I do this or can I not do this always with, interconnection of building blocks okay, maybe I will do it as a series of examples okay. I have X1 of N, which I am adding to X2 of N, X1 of N, adding to X2 of N that is my output which I am going to then downsample by a factor of M, okay, now this is identical to I can downsample X1 of N and X2 of N and then add them.

So in other words addition is a linear operation downsampling and upsampling are linear operations. So I can move things around and sometimes it is advantageous for us to do the following. So you take X1 of N, you downsample let me use the same colour, downsample by a factor of M and then you produce output signal, this is X2 of N downsampled by a factor of M, which you can then add and then produce the output; they are identical.

So this is X1 of N, in other words you can move the upsampling or downsampling, same argument holds. Upsampling or downsampling can be done and sort of trivially any scale factor can be done before or after a operation, is identically equal to, does not change anything, these are memoryless. You can downsample and then multiply. Now is there a difference to this two? Is there a difference in these two?

You may say well nothing because in one case you scaled and then multiply, actually what does the down sampler going to do, it is going to throw away samples. In that case why multiply. Throw the samples, only keep the samples that you are going to keep, and then do the computation, why waste multiplying obtaining results. So yes, this would be a preferred option.

So there are several times when you may want to optimise your structure and therefore do things little bit differently okay. Now it also extends to multiplication. There are times when you may do X1 of N multiplied with X2 of N, this is again a memory less operation sample by sample and if this you are going to do the downsampling here, downsampling or upsampling then it makes sense to do the downsampling first.

So in which case the better option would be take each signal downsample by a factor of M and then do the multiplication because anyway you are going to keep only these samples. So you start to think in terms of which is more efficient okay, so the notion of efficiency comes in and that is the pretty much what I wanted to leave you with saying that okay yes these are all equivalent, but you may want to think about okay, am I doing some wasteful computations if so how do I avoid it and of course all of these are equal.

You can change the downsampling to upsampling, all the results will hold. Notice that one thing that we have not done is to deal with the delay elements okay, because the upsampling and downsampling are delay variant blocks. So when delay comes into the picture we have to be very careful. So again we are going to develop some interesting and important results which pertain to use of delays in combination with upsampling and downsampling.

But those are going to be the results that we will develop next okay. So I assume, I believe we have already looked at the interchanging of results and interconnection. So if you go back to our, so we have pretty much come up to this point okay. We have come up to this stage, we looked at a couple of examples. Now when you introduce delays in combination with the multipliers and adders, what do you get? A filter. Delays, multipliers and adders.

So that is when you have filtering okay and you have either upsampling or downsampling, am I permitted to change, or under what conditions can I, how do I make the computations efficient. So one is we will look at, basically when can I do any sort of interchanging of the operations that leads to a very powerful result called the noble identities, which basically is one of the powerful results of multirate signal processing.

And then once you have that it leaves to probably the most powerful tool of multirate signal processing which is called polyphase decomposition again, that is something that we will pick it up in the next class.