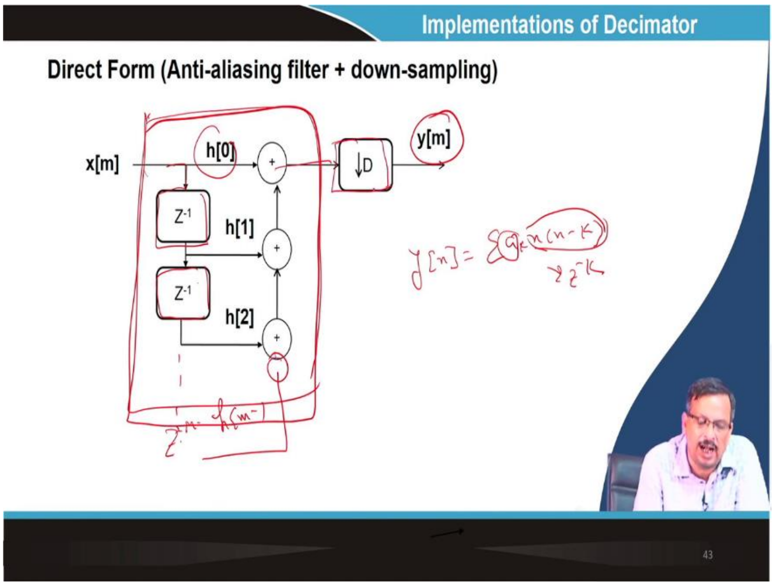
Signal Processing Techniques and its Applications
Dr. Shyamal Kumar Das Mandal
Advanced Technology Development Centre
Indian Institute of Technology, Kharagpur

Lecture - 53
Implementation of Decimator and Interpolator

In this class, I will talk about how to implement this multi-rate conversion system, whether it is down sampling or upsampling. So, now, we talk about how to implement it.

(Refer Slide Time: 00:33)

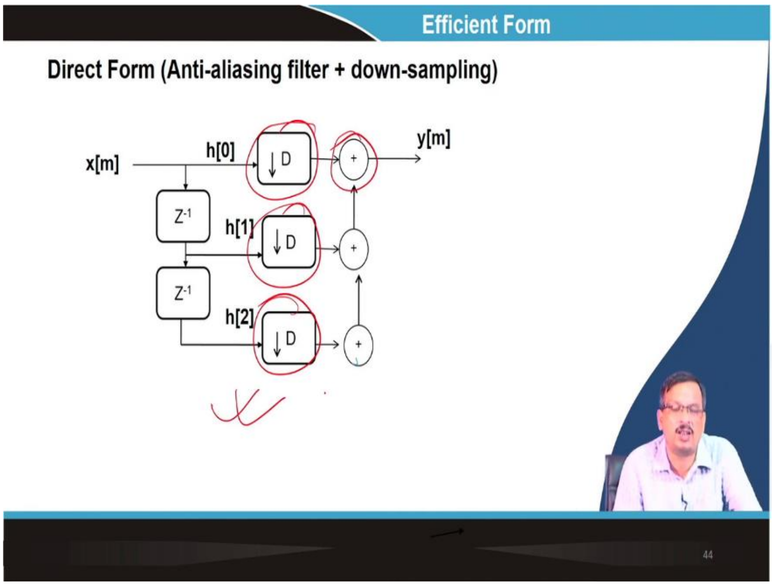


So, let us. So, what I said is that whether it is down-sampling or upsampling, this side is nothing but a filter. So, this filter is nothing but an FIR filter. If I say that, then the FIR filter can be implemented this way: the delay. So, what is an FIR filter? y[n] is equal to the summation of that a k into x[n-k].

So, I know the convolution of the filter coefficient along with that signal. So, in the z domain, it is nothing but a z^{-k}. So, if I want to implement using direct structure form, I can say that x[m] passes through a low pass filter, which can be implemented using direct structure form. So, let us say I said mth order filter, then I can say $z^{-(M-1)}$ because M equals h[0], which is equal to there. So, h[m-1] multiply and add together.

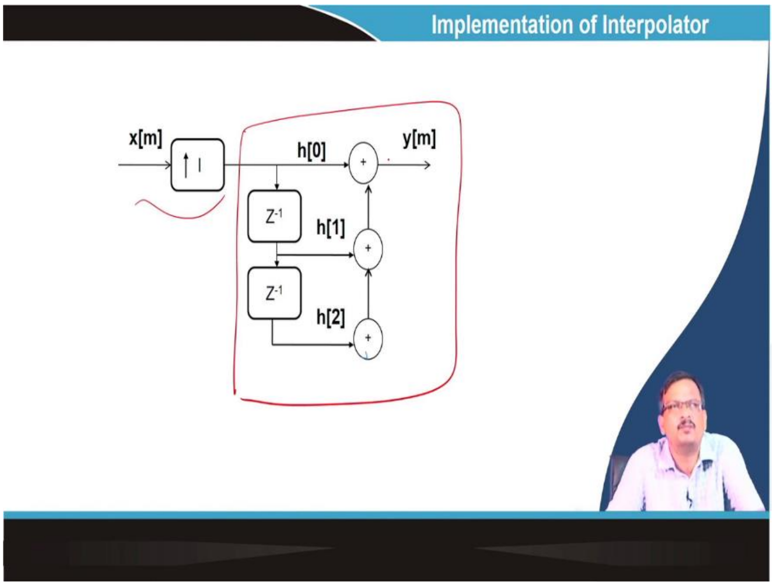
Then, I down-sample the signal and get y[m]. So, I can implement it using an FIR low pass filter using a simple direct form structure now. If you see that, I can use it instead of after filtering.

(Refer Slide Time: 01:56)



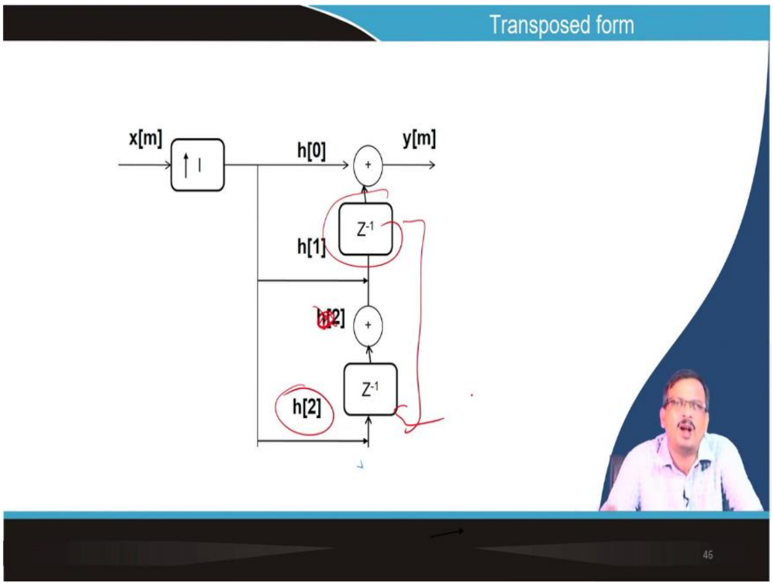
So, this D-down sample can also go inside. So, I can say that instead of the whole filter after downsampling, I can put this downsampling inside that direct structure. So, what will happen? That means it will reduce the sum complexity, ok? So, that is the efficient implementation of a direct form of the filter, or, I can say, a decimation filter similar to a downsampling filter.

(Refer Slide Time: 02:27)



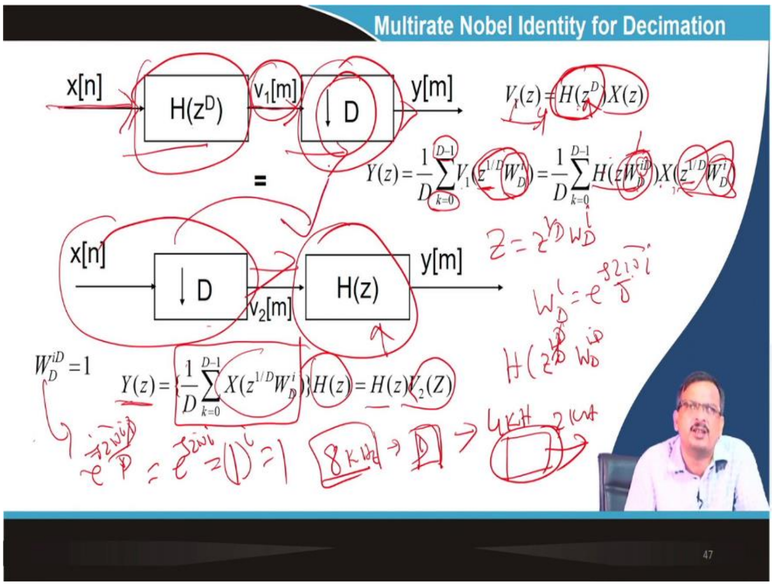
Similarly, upsampling followed by a filter can be implemented again using a direct form structure for interpolation or upsampling. So, it is nothing but a h[0] z^{-1}. So, delay and then multiply and sum.

(Refer Slide Time: 02:46)



It can be said that instead of z, I can also put the z here. So, this is not there. So, this will be there. So, I can put the z here. So this is called a transpose form of that implementation.

(Refer Slide Time: 03:01)



So, I can implement that FIR filter, whether it is HD or FIR filter, after that upsampling or before the downsampling, which is called an anti-aliasing filter and after it calls that removing of the multiple images of the signal. So, both can be implemented using the FIR filter.

Now, there is another example called multi-rate Nobel identity, so if I say the decimation, then a filter is followed by a downsampling. So, $H(z^D)$ is then followed by a down. So, if the x[n] signal is applied here, what is the V1(z)? V1(z) is nothing but the in multiplied by x(z) in the z domain.

So, what is y(z)? y(z) is nothing but a v 1 is the input, y is the output, and it 1D is the downsampling factor. So, it is nothing but a

$$Y(z) = \frac{1}{D}\sum_{k=0}^{D-1} V_1\left(z^{\frac{1}{D}} W_D^{k}\right)$$

What is w D i? It is nothing but an $e^{j2\pi/Di}$ ok. That we have already formed, we have already derived during the decimation or downsampling.

Now, if I put this one in v 1 value in this equation, H d. So, instead of this, it will become a z. So, instead of H(z), I can say H z to the power 1 by D and z D was there D. So, D D cancel z and w Di into D and x of z to the power 1 D w Di. z is equal to I put z is equal to z to the power 1 by D w Di. So, since it is z to the power, D already exists. So, it will be z into w Di D into $x(z^i)$1 by D into w Di D.

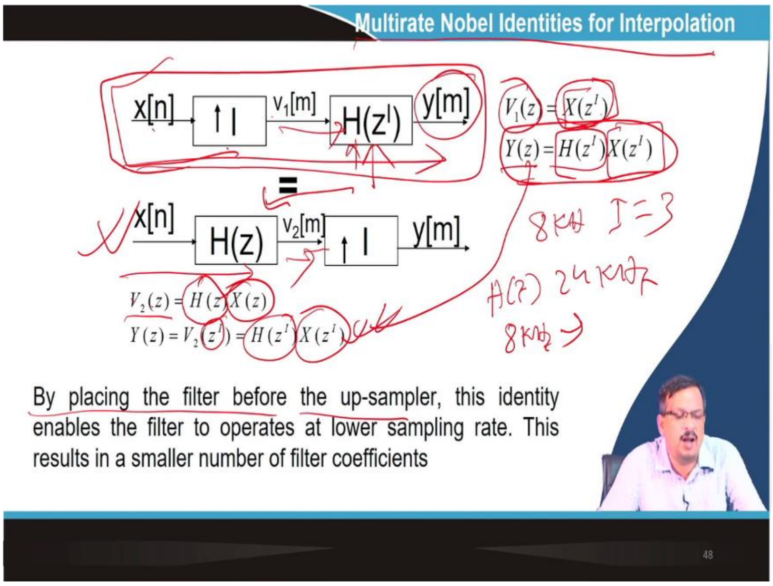
Now, since what is w Di D is it is nothing but $e^{j(2\pi/iD)/D}$, D D cancels. So, $e^{j2\pi i}$. So, i is an integer. So, I can say it is nothing but a 1 e to the power 2 π cos 2 π plus j sin 2 pi. So, once it is 2 π, I equal to 1, I equal to 2, I equal to 3. So, every value will be 1 because $e^{j2\pi}$ is nothing, but a 1 to the power i is equal to 1 i is an integer.

So, in that case, I can say that y(z) is equal to 1 by D. So, this becomes 1. So, this becomes H(z) and will remain ok. So, what is H z? So, it is nothing but an H(z) in V2(z). Let us consider this as a V two. So, I can say instead of first implementing an aliasing filter and then sampling, I can down-sample it and then pass it through an anti-aliasing filter. Both will be the same.

So, suppose I have an 8-kilohertz signal; instead of passing it first FIR filter, which is a limited FIR filter and then downsample, I can downsample it. Let us say I down-sample by 2. So, the first, 8-kilo hertz sampling rate was converted to 4-kilo hertz, then I restricted the filter π by D. It is nothing but 2 kilohertz. I implement the 2-kilohertz filter output; I will only get 0 to 2 kilohertz, which I desire.

So, why do we do the down-sampling? What is the advantage of doing the filter after downsampling? Because it can reduce the order of the filter since the sampling rate is reduced, then the order of the filter can be reduced. So, multi rate noble identity can be used for efficient computation similarly for upsampling.

(Refer Slide Time: 07:34)



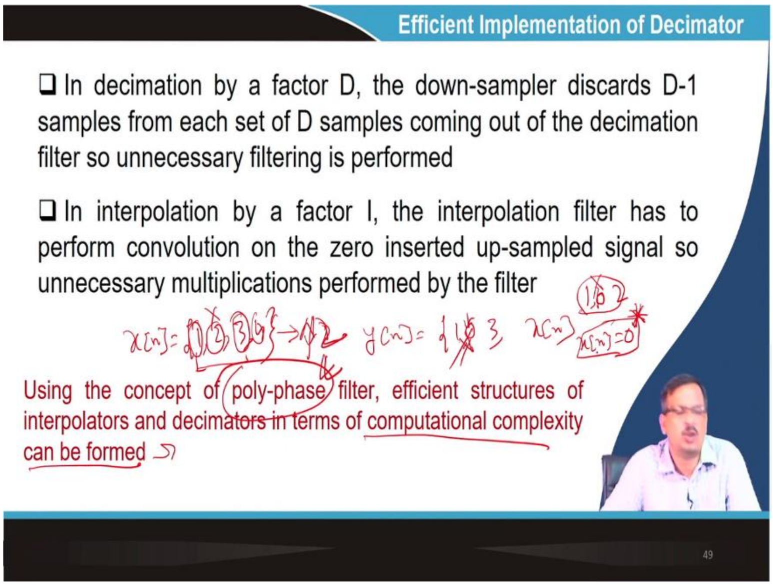
Let us say x[n]. So, this is the up sampler block diagram of an up sampler. So, I up the sample, then pass remove that image, multiple images keep the single image and then I get the y m. So, if I do this block diagram. So, what is V1(z)? V1(z) is nothing but an $x(z^I)$ if it is up-sampling by factor i. And then what is y(z)? y(z) is v 1 x. So, x of z to the power I pass through the $H(z^I)$.

So, I can say H(z) I multiply by x of z to the power I. Now consider I first do the filtering and then do the up-sampling. So, what is first do the filtering? So, I can say V2(z) is nothing but the H(z) into x(z) when it passes through the up sampler. So, it becomes V2(z) to the power I. So, I put z to the power I H(z) to the power I x(z) to the power I. So, this one and this one are the same y(z) s are the same. So, this structure and this structure are the same. This is called the noble identity of interpolation or upsampling.

So, why do I require it? So, if I say I place the filter before the up sampler, the up sampler means sampling frequency increases. So, my filter, here my I have to operate the filter at high frequency. Let us say I have a signal at 8 kilohertz and an up-sampling filter that is equal to 3. So, I know H(z) has to be operated at 24 kilohertz, but in this case, I can operate the filter at 8 kilohertz, and the filter output can be up-sampled to factor 3. So I can efficiently implement that filter.

So, that is called multirate Nobel identity for interpolation. So, multi rate Nobel identity for decimation or down-sampling multi rate Nobel identity for interpolation or upsampling both exist ok.

(Refer Slide Time: 09:49)



Now, I said efficient in the implementation of the decimation. So, what is the problem? Interpolator, upsampling, or down-sampling- whatever is the efficient implementation of up-sampling and down-sampling. So, what is the problem? So, when I say the upsampling or when I say the downsampling, let us say x[n] is equal to 1 2 3 4.

Now, if I say down sample by factor 2 D is equal to 2, that is what we select? We select sample number 1 and sample number 0, then discard sample number 1 and select sample number 3. So, x n. So, y[n] I can say that it is nothing but a 1, 3 and 4 will be discarded, and 5 will be selected.

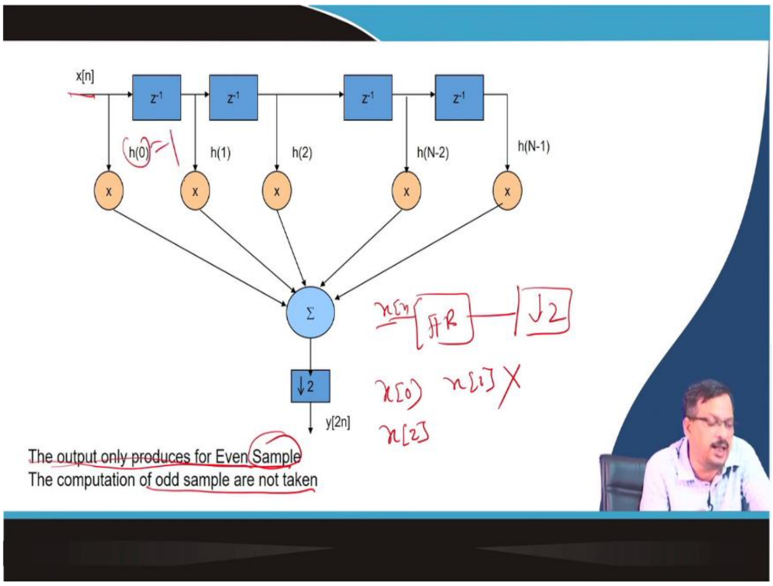
So, when I say the filtering. So, after downsampling or after upsampling, when I want to implement the filter. So, what am I doing? Basically, that discarded sample is also included in my computation if I implement the filter. So, why should I include that coefficient during filtering? Because it reduces the order of the convolution. So, what is the convolution? The complexity of the convolution depends on the length of the signal.

Due to the downsampling, the length of the signal is reduced. So, why should I use that whole length of the signal for filtering? So, how can I efficiently implement the filter that the sample I am discarding will not use? Similarly, in the case of upsampling, let us say the upsampling factor by 2, then what will happen? In between 1 and 2, I put a 0.

When I multiply convolution, x[n] multiplies when the filter coefficient if the x[n] value is 0, so why should I include that part in the convolution? So, in both cases, I want an efficient algorithm by which I can say that in the case of interpolation for sample value x[0], the filter convolution will not happen. In the case of downsampling the sample that I am discarding, I should not want to compute that value.

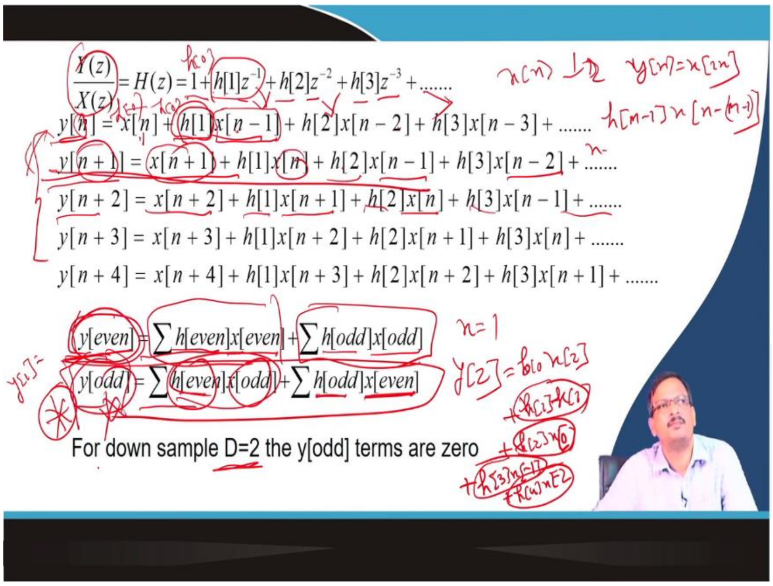
So, to implement that, there is a concept called polyphase filtering. So, we can say that an efficient structure of interpolation and decimation in terms of computational complexity can be formed using poly-phase filtering. First, I have to understand what is poly phase filtering and how this poly phase filtering can reduce the computational complexity or the redundancy calculation that exists in the case of up-sampling and down-sampling.

(Refer Slide Time: 13:05)



So, what is poly-phase implementation? So, let us say I have a filter x[n], x[n] that passes through an FIR filter and then downsamples by 2; let us say this is my system diagram. So, what I said is output is only produced for the even sample, and computation of the odd sample is not taken because I know I will take x[0], then I will take x[2], not x[1], because x[1]. I am discarding it as a downsample by 2. So, even sample output is produced, and odd sample output is not produced.

(Refer Slide Time: 13:52)



So, let us say write down that mathematical form. So, I can say y(z) by x(z), which is the filter called H(z), is nothing but a 1 plus z to the power 1 h[0] if the h[0] is equal to 1 then I can say it is 1 into h[1] into $x(z)^{-1}$ or I can write h[0] also here. So, here will be h[0] ok.

So, now, if you see that h[1] z^{-1} h[2] z^{-2} h[3] z^{-3}. So, when I say in the time domain, y[n] is nothing but the x[n] multiplied by h[1] is multiplied by x[n-1] delayed by 1 sample delayed by 2 samples delayed by 3 sample dot dot dot. Suppose the order of the filter is m. So, the last coefficient is h[m-1] x[n] minus m minus 1 ok.

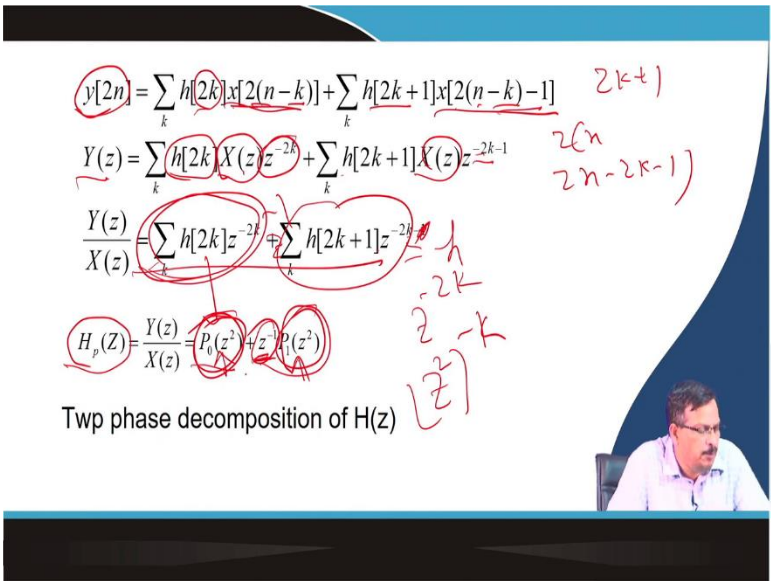
Now, what is y[n+1]? I increase; instead of n, I write n plus 1. So, x[n+1]. So, x[n] is replaced by n plus 1. So, h[1] into 1 minus 1. So, x[n] then x of then 1 will be reduced n minus 1 n minus 2. So, what is n plus 2? x[n+2] h[1] into x[n+1] h[2] into x[n] like that. Now, I can say if I select only the y[even] part. So, if you say y[even] is equal to h[even], multiply by x[even] plus h[odd] multiplied by x[odd].

Let us say that let us n equal to let us say 1, n equal to 1. So, y[2]. So, this one is nothing but a h[0], or I can say h[0] equals 1 then x 2 plus I can say this 1 n equals 1. So, x 2 plus h[1] into x 1 plus this line I am writing. So, h[2] into x[0]. So, 1 minus the x[0] plus h[3] into x minus 1 plus h[4] into x of I can say n minus 3. So, that is minus 2. So, if you see h[2] x 0 h 4 x 2 and x 1 h[1] h[3] x minus 1. So, I can say that y[even] is nothing but a h[even] multiplied by x[even] plus h[odd] multiplied by x[odd].

Similarly, y[odd] is nothing but a h[even] multiplied by x[odd] and h[odd] multiplied by x[even]. So, I checked it, and let us say for y equal to n equal to 1. So, y 1 is equal to x[n] x 1 h[1] x 0. So, h becomes odd, and x becomes even. So, that is why even h[odd] x[even], and if it is h[even], then x becomes odd.

Now, for D equal to 2, this y[odd] is not required. I only have to calculate y[even] because y[odd] does not exist because I know x[n] when it is downsampled by 2; that means y[n] is equal to x[2] n elsewhere, it is 0. So, it is either 0, 2, or 4. All those exist. So, I can say y[even] exists, and y[odd] does not exist.

(Refer Slide Time: 18:09)

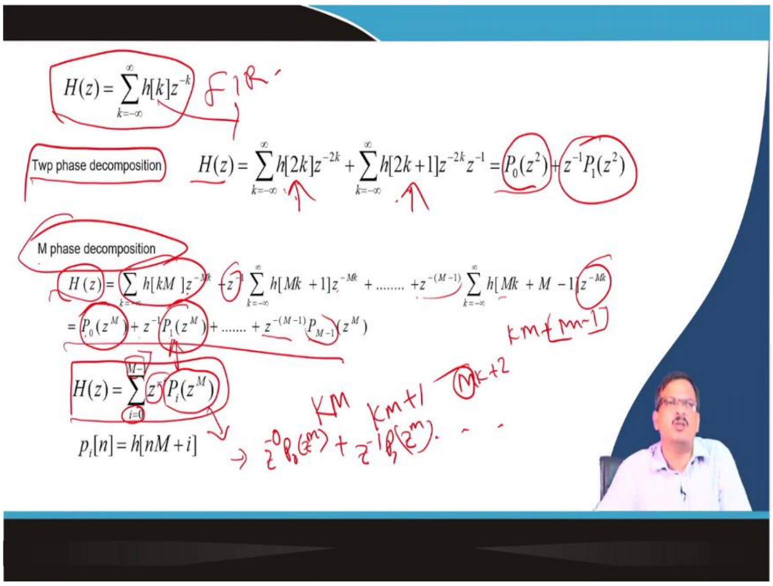


If I say, this is the y[even] y[2] n. So, it exists 2 k 2 n minus 2 k, and this is odd 2 k plus 1 is odd. So, it is nothing but a 2 into n 2 n minus 2 k plus 1. So, that is this one. So, I can say y(z) is equal to h[2k] z to x(z). This is nothing but a x(z) z^{-2} k. similarly, this is nothing but an x(z) z^{-2} k minus 1.

Now, if I say the minus 1 is outside. So, I can say that, or I can simplify this. So, minus 1 is outside. So, this is nothing but a. I can say this term will be there, and $H(z)^{-1}$ will be there and then z^{-2} k. So, I can let us say this is a filter, and this is also a filter. So, this is Po(z), and this is P1(z) square because 2 k z^{-2} k is nothing but a z to the power square^{-k}. So, k varies from minus infinity to infinity. So, I can say it is Po(z) square, and it is P1(z) square, delayed by z^{-1}.

So, I can say Hp(z) can be synthesised by 2 poly-phase filters, which is delayed by one sample. That is why it is called poly phase ok.

(Refer Slide Time: 19:48)



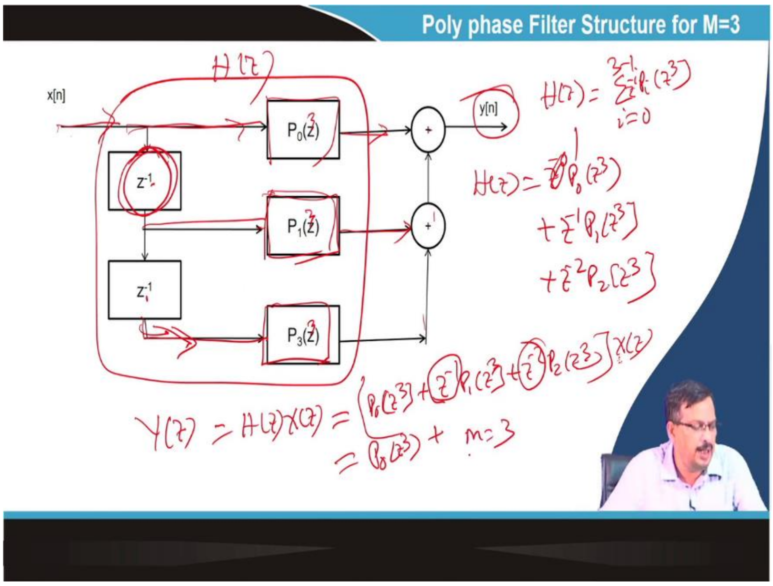
So, if I want to put it in a generalised form, let us say H(z). This is my FIR filter, ok? So, now, H(z) is equal to. So I can break even and odd terms. So, when I break it even and odd, it is nothing but a Po(z) square plus z^{-1} P1(z) square. So this is called 2-phase decomposition.

Let us say I want to M phase decomposition; I want to decompose h[0] H(z) in a M number of phases. So, if you see 2 phases, odd and even. So, I can say k index k is divided by m number of samples. So, it is k m, then k m plus 1, then k m plus k m plus 2 plus 2 like that, then k m plus 3 like that. So, M number of decomposition happens.

So, the last one is k m plus m minus 1, ok. So, this is nothing but a z^m minus 1 into z^m k. So, I can say that this is nothing but a poly phase $Po(z)^m$ This is nothing but a delay by $P1(z)^m$ If you see, why is it P 0? Because it is the 0th one. Why is it p one? Because it is 1. So, that is why it is. So, the second term is P 1, the third term is P 2, and the Mth term will be P M minus 1.

So, if I write it down in summation form, I can say I equals 0 to M minus 1, it is z to the power minus I P I z^m So if I derived it, what will happen? This i equals 0 z^{-0} $Po(z)^m$ plus z^{-1} $P1(z)^m$ So, dot dot dot dot. So, this is my polyphase decomposition of a filter H z. So, M phase poly means more than 1 M phase decomposition. So, now, if I want to implement the filter, what is the signal flow diagram of the filter?

(Refer Slide Time: 22:23)



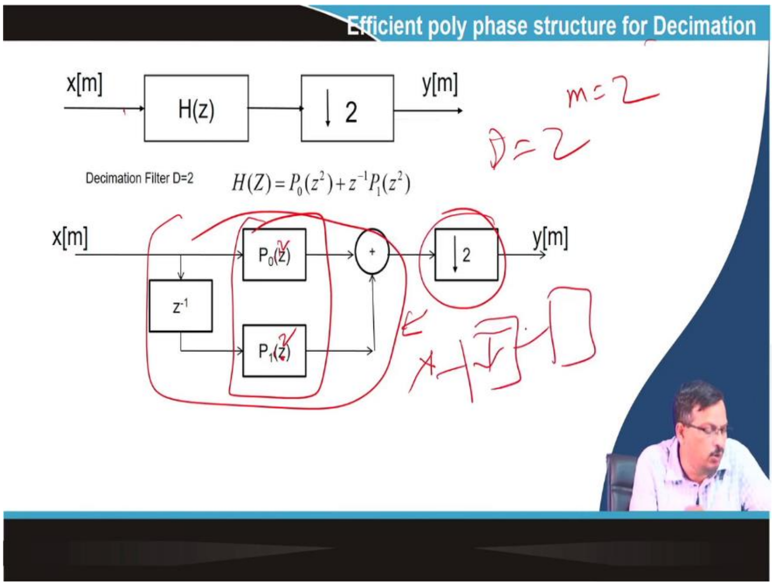
So, let us say I have a 3 phase, yeah. So, H(z) is equal to 3 phase decomposition I equal to 0 to 3 minus 1 z to the power minus I P I z to the power M, M is equal to 3. So, there will be a z-cube, ok? So, I can say H(z) is nothing but a z^{-0} $Po(z)^3$ plus z^{-1} $P1(z)^3$ plus z^{-2} $P2(z)^3$. So, how do I implement x[n]? So, this is x of. So, this is my H(z).

So, how do I implement it? That Po(z) will directly pass, z to the power 0 is nothing but a 1 and then delayed by one sample z^{-1} I implement P1(z) done then z^{-2} I. So, why is it coming? Because I know y(z) is equal to H(z) into X z.

So, I can say it is nothing but a Po(z) to the $power^3$ plus z^{-1} P1(z) plus $z^{-2}P2(z)^3$ whole multiplied by X z. So, it is nothing but a $Po(z)^3$ plus x(z) z^{-1}; that means sample x will be delayed by 1 sample. So, that is why I delayed by 1 sample.

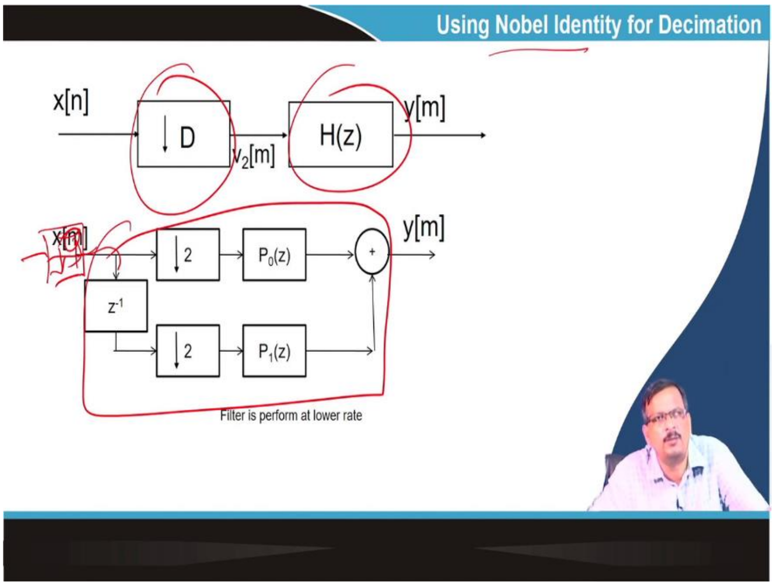
Some x will be delayed by 2 samples. So, z^{-1} and z^{-1} is delayed by 2 samples. So, some two sample delayed signals will pass through the P j P3 z^3; one sample delay is $P1(z)^3$, and now no delay is $Po(z)^3$. All are added together, and I get y n. So, this is called poly phase structure for M is equal to 3.

(Refer Slide Time: 24:42)



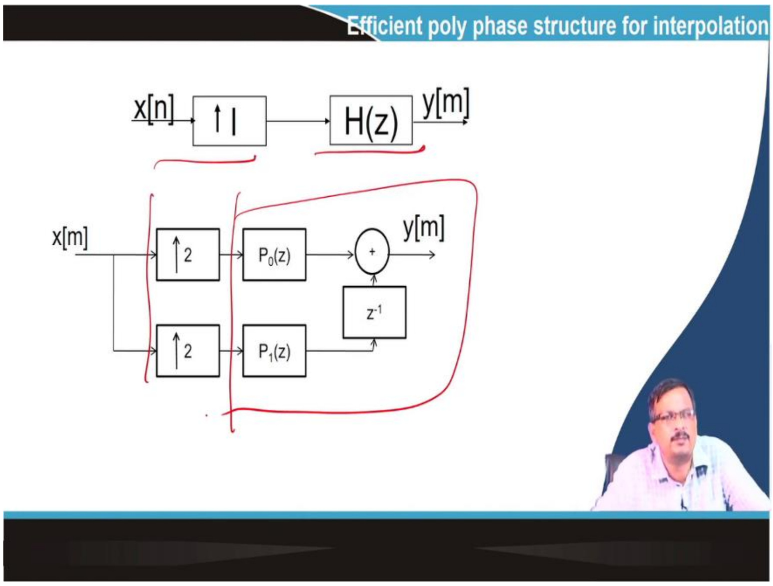
Similarly, let us say m is equal to 2. This one ok. So, this is a decimator, and decimation factor D is equal to 2. So, I decomposed that filter into 2 poly phase form and decimated it by factor 2. That means this filter will not use that discarded sample. So, this will be z square.

(Refer Slide Time: 25:24)



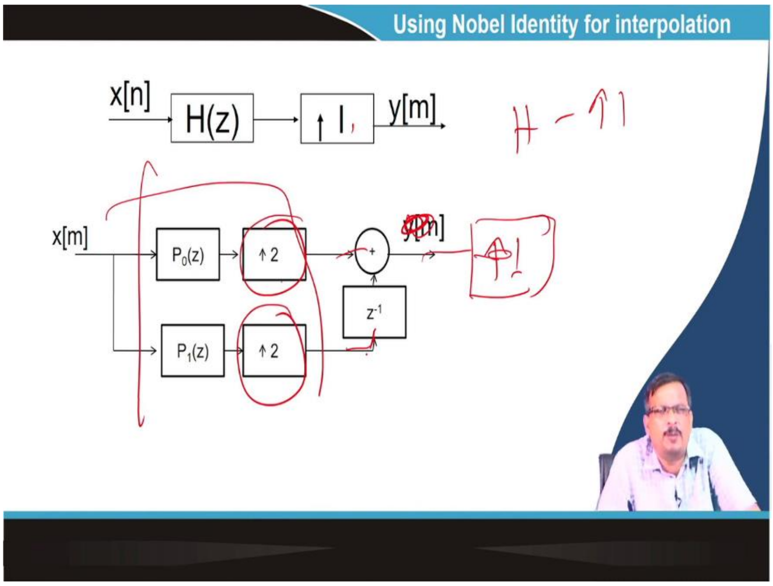
Similarly, decimation can be that if I use the Nobel identity here. So, I filtered and downsampled; the identity they said allowed me to downsample the signal and filter. So, I used Nobel's identity first down sampling, then filtering. So, I can do that with that one, first downsampling and then filtering. So, how do I do that? First downsampling this side. So, sorry there is a mistake. So, on this side, I will first sample and then implement that filter. Both are possible, first downsample and then filter.

(Refer Slide Time: 26:09)



Similarly, for interpolation, upsampling and filter H(z) is implemented this way, you are doing poly phase filtering ok.

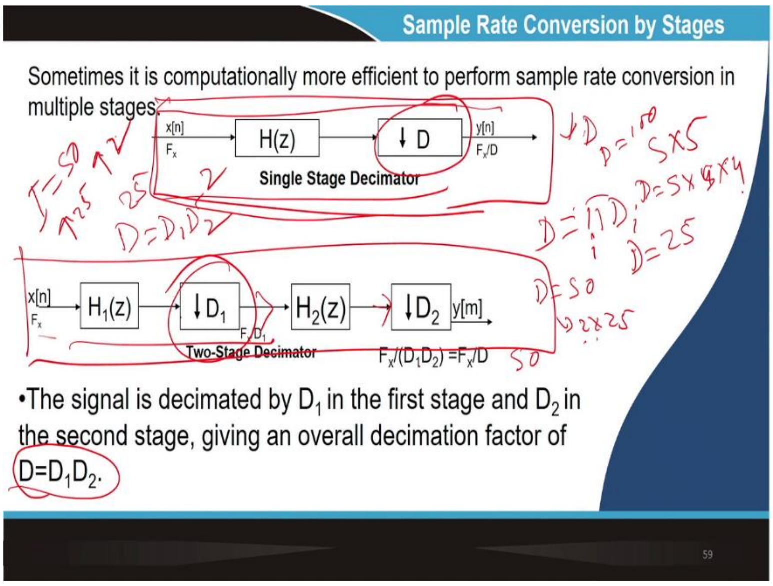
(Refer Slide Time: 26:23)



Now, I can use the new identity again. So, for the first filter, I can say the filter, and then, sorry, this will not be. This will be i or i. So, this. So, I can say that first filter, then I, but

I can be included inside also, then delay no problem. So, using Nobel identity, I can change the structure of the filter, but poly-phase implementation is mandatory.

(Refer Slide Time: 26:58)

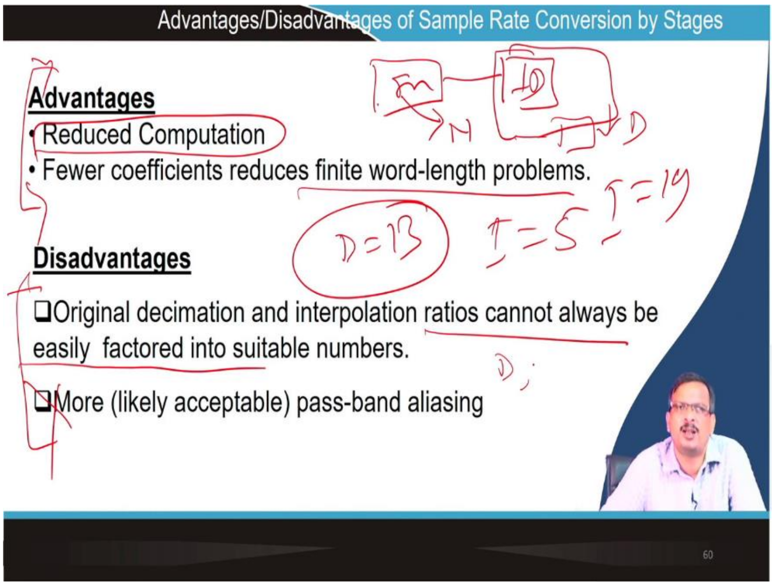


Now, there is another problem. That is called sampling rate conversion by stage. Instead of single, let us say this is a single-stage sampling rate conversion. This is a single-stage poly. So, single-stage sampling rate conversion. So, downsample by factor D in a single stage.

I can say that D is nothing but a product of many D. Let us say D is nothing but a product of Di. So, for example, let us say D is equal to 50. It is a product of 2 into 25, or let us say D is equal to 100, and D is equal to 25. I can say it is a product of 5 into 5. So, instead of a single decimation, I can say that D1 first decimated the signal. So, D is a product D equal to D1 in D2.

So, in this case, let us say that D1 equals 25, and D2 equals 2. So, first, I down the sample by D1. Again, I down the sample by 2 factor 2, and then ultimately, the total down sampling is 50. So, instead of a single stage, I can use multiple stages for down sampling. So, the overall decimation is D1 multiplied by D2. This can be done ok. Why do we do that? That is also an important factor.

(Refer Slide Time: 28:49)



Similarly, for interpolation also that I can instead of single interpolation, I can say that upsampling by, let us say, up the sample, I is equal to, let us say, 50. So, first up the sample by 25, then up the sample by 2 possible. So, multi-stage sampling rate conversion. So, why do we use multi-stage? What is the advantage of using a multi-stage system? The advantage is the reduction of computation. I will show you.

So, even in a single stage, if I want to implement the filter and then single stage 50, I have to I can show you the order of the filter, and then instead of a single stage, I can use multi-stage and then decimation by D, then I will show you the order of the filter. So, it reduces computation. In the second stage, I am computing using a lower sampling rate. Since the filter coefficient's filter number is reduced, the finite word length problem is solved.

Then what is the disadvantage? Many times, let us say D is equal to 13. Can I process in multi-stage? No, I cannot do that. So, let us say I am equal to 5. I cannot do that; I am equal to 19, so if that decimation interpolation ratio cannot always be easily factorised in a suitable number. Then, I cannot use multi-stage decimation or multi-stage interpolation. Then, every time, the problem is aliasing pass band aliasing.

So, there is a problem with pass band aliasing. Because a multistage multi-aliasing filter will be there and pass band aliasing every time, there will be an aliasing effect. So, that is the disadvantage part, but the advantage is the maximum.

So, if I design the filter in such a way that aliasing is not allowed. So, I can remove this portion, but if the D and I are factors that cannot be decomposed, I cannot use multi-stage decomposition or multi-stage interpolation. But yes, there is a definite advantage to using multi-stage decomposition, upsampling, or downsampling. So, what is that?

So, multi-stage means I am using 2 stages. It may be a n number of stages, z 1 D1 z 2 D2 z 3 D3 like that way. So that there is nothing called that I have to restrict. So, D can be, let us say D is equal to 100. I can say D is equal to D1; that means 5 into 4 into, let us say, 5 into 5 into 4 may be possible 5 5 and 4 or 25 in into 4.

So, multi-stage if the D is permitted to decompose if I decompose it, then the order of the filter will be lowered. So, in the next class, I will give you an example of how the order of the filter is going down and that advantage is taken.

Thank you.