Mathematical Methods and Techniques in Signal Processing - I

Prof. Shayan Srinivasa Garani
Department of Electronic Systems Engineering
Indian Institute of Science, Bangalore

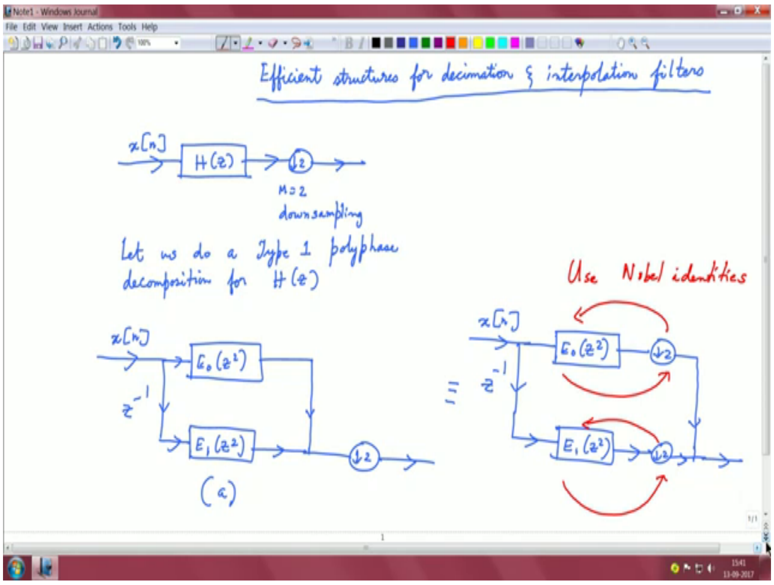
Lecture - 41
Efficient architectures for interpolation and decimation filters

So, in the last lecture we discussed the idea behind polyphase decomposition. We also looked in to noble identities and interconnecting systems. So, now we have a bunch of identities and useful tools and using these can we develop efficient structures towards decimation filters that is you have filtering followed by decimation or for interpolation filters that is you up sample then you filter or by for doing an m by l conversion right and how do we realise efficient architectures for sampling rate conversion; so, this is the idea.

But two importance things you have to consider. So, when you are down sampling you are throwing away samples. So, therefore, is there a way in which you can efficiently make use of the computation and in the case of interpolation you know you are up sampling first; that means, you are inserting 0s and then filtering so; that means, you are doing a lot of filtering of these 0s.

So, still waste in computation; so, is there a way to overcome these limitations through efficient design. And we will see how polyphase filtering and the user mobile identities and interconnecting systems and all these ideas that we discussed in the last lecture will be useful towards this purpose.

(Refer Slide Time: 01:53)



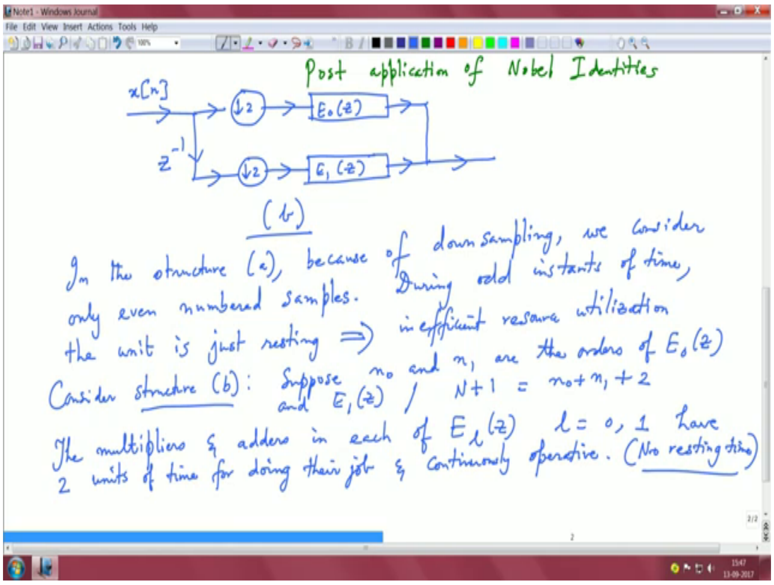
So, let us consider the signal x of n; now we are doing filtering first followed by decimation. So, let us say we are filtering through some H of z and then we are sat down sampling by 2 right; let us say m equals two down sampling. We will decompose this filter H of z using polyphase type 1 polyphase.

So, let us do a type 1 polyphase decomposition for H of z. So, they get this and this form; so, basically we have x of n we have a filter which is E naught z square through the delay element here we have E 1 of z square. And this signal we are down sampling by 2 right this is the normal architecture; I mean at least we have done one important step of simplification that is we have decomposed the filter into a type 1 polyphase decomposition.

Now, this structure is still not giving insights to us because now we are computing something through this equivalent polyphase representation, but we are still throwing away some samples right. So, let us see if we can bring this down sampler into each of the branches here right. So, this is equivalent to the following structure sum of x of n.

Now, I have E naught of z square down sampler a delay element E 1 of z square this way I am pushing the down sampling down sampler into each of the two branches. So, this my equivalent form not much insight, but I think we would be seen something interesting if we could swap the down sampler and the filter in each of the parallel parts using noble identities ok.

(Refer Slide Time: 06:22)



So, we simplify this as follows we get x of n; we down sample this first this is followed by some filter of lower order which is E naught z delay down sampler followed by some filter E 1 and now we have the output. And I call this structure b this is post application of noble identities.

Now, in the structure a because of down sampling; we consider only even numbered samples because we are throwing away the odd samples in the process of the down sampling. Now which means during odd instance of time the unit is just resting which is basically in efficient resource utilisation because filtering and these operations can be costly right.

So, now we consider structure b right which is this structure suppose n naught and n 1 are the orders of the filters E naught of z and E 1 of z such that N plus 1 is n naught plus n 1 plus 2 let us say this equation holds right it means we have lower order filters for E naught and E 1 and the overall filter orders let us assume this is this is N right having N plus 1 coefficients.

Now, the multipliers and adders in each of E l of z; so, in this case l equals 0 and 1 have two units of time for doing their job. And there continuously operated that is there is no resting time which is important.

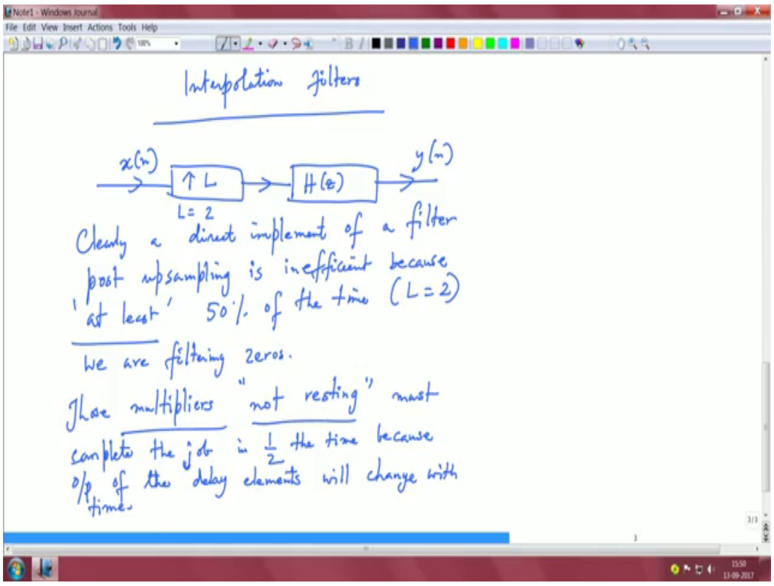
In the first case there was resting time because you are just throwing away the samples;

you are doing you know some computation, but you are throwing away the samples in this case you basically have two units of time for doing your job and you are continuously operated there is no resting time. So, therefore, this is definitely an efficient architecture for realisation right you are cutting down the filter order in the first in the first cut.

That means, your number of multiplications and additions would be lower the power branch basis and then you are continuously operating because you are you are not wasting any time here because you are keeping the odd samples here you are just basically none of the unit is not resting right. And if you think about an m phase down sampler you know you can imagine you are efficiency is m fold roughly m fold ok.

So, now having seen the advantages of polyphase decomposition and noble identities to efficient decimation filter design.

(Refer Slide Time: 12:27)



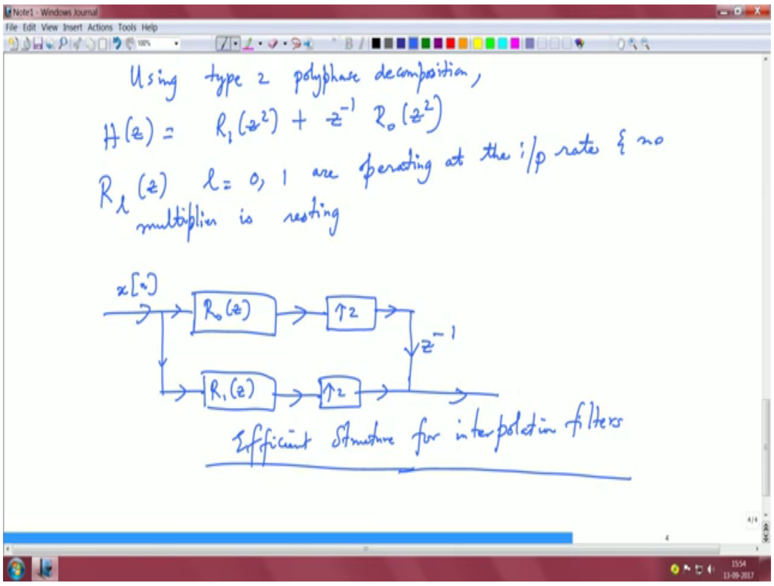
Let us look into the aspect of interpolation filters. So, we have a signal which is discrete time signal and we up sampled this by a factor L and then we filter this through some filter H of z; that means, we have this output.

Now, clearly a direct implementation of a filter post up sampling is inefficient because at least say suppose L equals 2 right at least 50 percent of the time this is for L equals 2 we are filtering 0s the rest of the time you may have some 0s, but at least 50 percent of a time since you are inserted 0s 50 percent of the time. So, you are basically filtering these 0es through this filter.

So, this is inefficient and those multipliers not resting must complete the job in half the time. Because output of the delay elements will change with time; beautiful if you think about any filter basically you have delay elements you have some scalars basically to scale the magnitude at the output on b tap delay line.

So, if you look at direct form implementation just brief course away you are really not efficient. Because half the time you are just filtering out the 0s and those multipliers that are not resting right; so, they must complete the job in half the time for L equals to 2. So, now fortunately we can think about an efficient implementation for this and that is three type 2 decomposition.

(Refer Slide Time: 16:00)



Which is type 2 polyphase decomposition of the filter and then application noble identities; so, we will have this works. So, using type 2 polyphase decomposition; so, we can write H of z as some R 1 z square plus delay times R naught z square. I mean we are using the permitted version right we discussed in the last lecture about an exercise problem which is link in the type 1; polyphase decomposition to the type 2 polyphase decompositions. So, we will basically invoke type 2 polyphase decomposition for interpolation filter.

Now, each of these filters R l of z l equals 0 1 are operating at the input rate and no multiplier is resting I mean because no ideal multiplier. So, what we do is we have x of n feed this through this filter R naught of z this is post application of the noble identities. So, you can see the swap between the up sampler and the filter and also carefully observe the delay element that is appearing at the output here. So, this is R 1 this is also up sampled and this is our efficient structure ok.

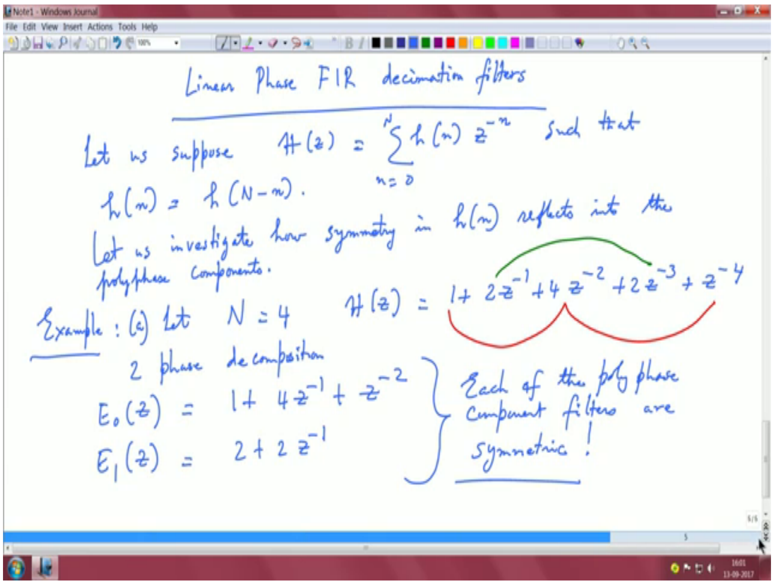
So, what we did is just for the input side I mean just we did. So, if we just recall here; so, we had an up sampling by 2 followed by filtering right and the filter we do a polyphase decomposition as follows. So, it is going to up sampling followed by filtering through polyphase and then what we do is basically group structure very carefully like what we did in the decimation filters. So, we swap using noble identities the up sampler and the reduced order filter and we have an efficient architecture for doing interpolation.

Now, you could imagine the scaling in terms of efficiency for l fold expansion right if we did an l fold expansion you are inserting l minus 1 0s into the into the signal right and therefore, a lot of filtering is happening through the 0s. So, therefore, we want to cut down you are you are drastically wasting your compute power and efficiency and the delays resting of the filters. So, therefore, using type three polyphase decomposition we can realise an efficient architecture for interpolation filters.

So, we will also study some important properties with linear says FIR decimation filter now we if we if I give you a general FIR filter possibly which is symmetric right. Do we see any structure in the polyphase components right this is this is another thing which we have to sort of investigate. So, one is we have a an arbitrary filter right now we have any H of z, but when you do filter design sometimes you can exploit properties in the construction of the filter itself right, you might want to bring in some symmetry in the filters for efficient computations.

And if you bring in some symmetry then what is impact on polyphase filters and then on top of it coupled with this efficient architecture for realisation your much better in terms of your realisation ok.

(Refer Slide Time: 21:28)



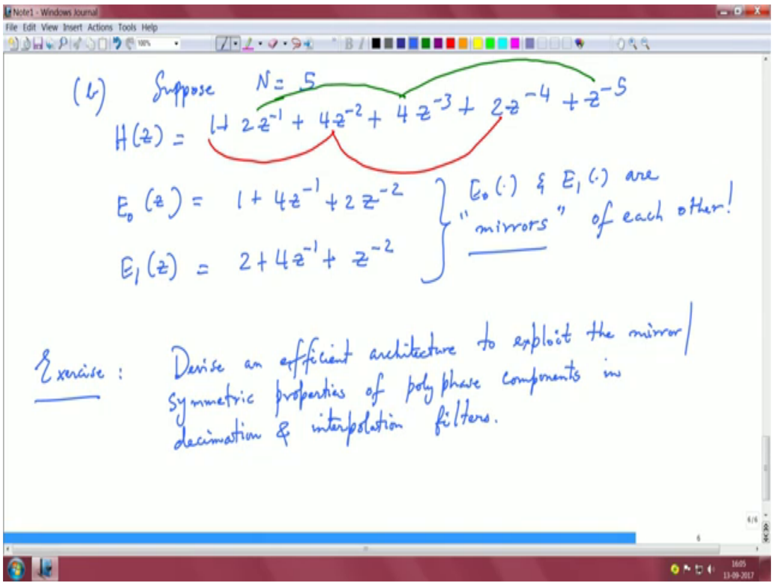
So, let us look into the next aspect which is linear phase FIR decimation filters. So, let us suppose we have a filter with impulse response small h of n such that H of n is H of n minus capital N minus small n; I mean this is symmetric right this is symmetric.

Let us investigate how symmetry in H of n reflects into the polyphase components. Now for any N capital N there only two possibilities N could be odd, N could be even right. So, let us just take some examples here sort of get a feeling for what is happening; let us choose capital N to be equal to 4. So; that means, I have five coefficients right N plus 1 coefficients. So, let us assume that H of z of the form 1 plus 2 z power minus 1 plus 4 z power minus 2 plus 2 z power minus 3 plus z power minus 4.

So, now we are familiar basically if you want E naught we take 1 4 z power minus 2 then you know if you choose this as one part. And then we could choose this and this as our second part right for L equals 2 polyphase. Let us take you know two phase decomposition; so, m here m equals 2 in this case now I have E naught of z is 1 plus 4 z power minus 1 plus z power minus 2 recall 1 plus 4 z power minus 2 plus z power minus 4 is z naught of z square; if you want to get reduce the order and you just look at in order z which is which is going to be this. And similarly you have E 1 of z which is 2 plus 2 z power minus 1.

And if we observe this carefully right basically these filters are symmetric right. So, each of the polyphase component filters are symmetric and one can think about how we can exploit symmetry to do your filtering operations efficiently right.

(Refer Slide Time: 26:08)



Let us look at the other case which is the odd case suppose N equals say 5; let us consider H of z to be 1 plus 2 z power minus 1 plus 4 z power minus 2 plus 4 z power minus 3 plus 2 z power minus 4 plus z power minus 5 even observe this symmetricity because we have started with the constraint that H of z should have a symmetric impulse response right.

Now, when we do the polyphase decomposition we get just recall you take this and then you take this. And for the other case you take this, you take this, you take this ok. Now E naught of z is basically 1 plus four z power minus 1 plus 2 z power minus 2 and E 1 of z is basically 2 plus 4 z power minus 1 plus z power minus 2.

Now, we observe E naught of E 1 unlike even case these are not symmetric, but E naught and E 1 are mirrors of each other. You look at this 1, 4, 2 and you see the mirror of this which is 2, 4, 1. So, there are some nice structures which are hidden into this polyphase filters and we should be clever to exploit these structures to basically save our multiplications using the some of these components.

So, sort of a this point would give you an exercise to ponder upon; device an efficient architecture to exploit the mirror slash symmetric properties of polyphase components in decimation and interpolation filters ok.

So, we looked at decimation filters, we looked at interpolation filters, we saw how we can use polyphase decomposition and noble identities for efficient realisation. Then if the original filter itself of course, that did not assume what the structure of the original filter has to be, but if we assume that we start with an FIR filter with symmetric impulse response right; how can we, what can we say about the properties of the polyphase components?

And now we know that there could be there could mirrors or there could be symmetric filters in that is polyphase filters could be mirrors when N equals odd and it could be symmetric when N equals even. Now considering these other additional properties can we build in device an efficient architecture to exploit these properties?