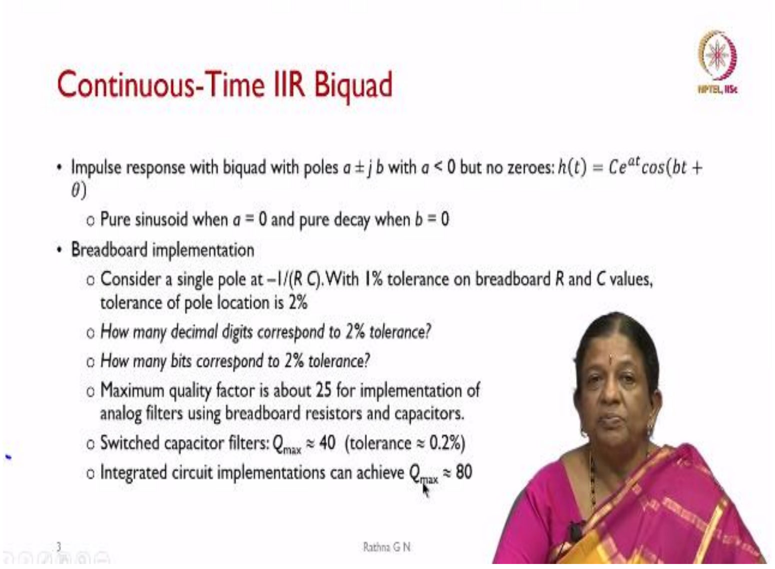
Real - Time Digital Signal Processing

Prof. Rathna G N
Department of Electrical Engineering
Indian Institute of Science - Bengaluru

Lecture - 13
IIR Filters 2

Welcome back to real time digital signal processing course. So, we discussed about IIR filters little bit in the last class today we will continue with the same thing. So, as a recap, as I said we have discussed little bit IIR filter, how we have to design from analog domain to digital domain, what we have seen the thing? So, today we will continue on that.

(Refer Slide Time: 00:45)



Continuous-Time IIR Biquad

- Impulse response with biquad with poles $a \pm j\,b$ with $a < 0$ but no zeroes: $h(t) = Ce^{at}cos(bt + \theta)$
  - Pure sinusoid when $a = 0$ and pure decay when $b = 0$
- Breadboard implementation
  - Consider a single pole at $-1/(R\,C)$. With 1% tolerance on breadboard R and C values, tolerance of pole location is 2%
  - *How many decimal digits correspond to 2% tolerance?*
  - *How many bits correspond to 2% tolerance?*
  - Maximum quality factor is about 25 for implementation of analog filters using breadboard resistors and capacitors.
  - Switched capacitor filters: $Q_{max} \approx 40$ (tolerance $\approx 0.2\%$)
  - Integrated circuit implementations can achieve $Q_{max} \approx 80$

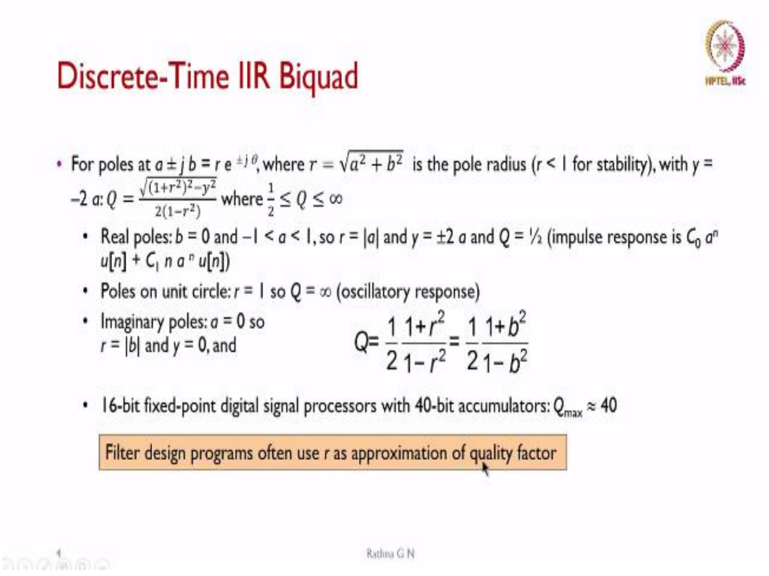
3 Rathna G N

So, what we said was, it was the structure 1 what we had taken the thing or direct form 1 so, here we will see how Biquad has to be designed. In the continuous domain first we will see the thing and later on we will go back to the digital domain. So, we know that impulse response with Biquad with poles $\sigma \pm jb$ were with $\sigma < 0$ or I can take it as a which is less than 0, but no 0s then the impulse response is given in the analog domain $h(t) = Ce^{at}cos(bt + \theta)$.
So, pure sinusoid when $a = 0$ and pure decay when $b = 0$. So, how we can implement in the breadboard what it is given nowadays, nobody does the breadboard design but one can test it how it is going to be and if you are satisfied you can go for implementation in the regular PCB. So, we consider a single pole where is it, it is at $-1/(RC)$. So, we consider 1% tolerance for breadboard R and C values and then C tolerance for the pole location what we assume is 2%.

So, we will say that how many decimal digits corresponds to 2% tolerance one has to look at it and how many bits corresponded 2% tolerance one has to look at it. So, what we want is the maximum quality factor is about what we say is 25 for implementation of analog filters using breadboard resistors and capacitors. So, if we use this switched capacitor filters the quality factor what we said was maximum what we can achieve is approximately 40 that is for the tolerance approximately 0.2%.

And then integrated circuit implementation can achieve maximum up to 80. So breadboard, you can see that it is going to be half of that of the ICs design.

(Refer Slide Time: 03:15)



Discrete-Time IIR Biquad

- For poles at $a \pm j\,b = r\,e^{\pm j\theta}$, where $r = \sqrt{a^2 + b^2}$ is the pole radius ($r < 1$ for stability), with $y = -2\,a$: $Q = \frac{\sqrt{(1+r^2)^2 - y^2}}{2(1-r^2)}$ where $\frac{1}{2} \le Q \le \infty$
  - Real poles: $b = 0$ and $-1 < a < 1$, so $r = |a|$ and $y = \pm 2\,a$ and $Q = ½$ (impulse response is $C_0\,a^n\,u[n] + C_1\,n\,a^n\,u[n]$)
  - Poles on unit circle: $r = 1$ so $Q = \infty$ (oscillatory response)
  - Imaginary poles: $a = 0$ so $r = |b|$ and $y = 0$, and $Q = \frac{1}{2}\frac{1+r^2}{1-r^2} = \frac{1}{2}\frac{1+b^2}{1-b^2}$
  - 16-bit fixed-point digital signal processors with 40-bit accumulators: $Q_{max} \approx 40$

Filter design programs often use r as approximation of quality factor

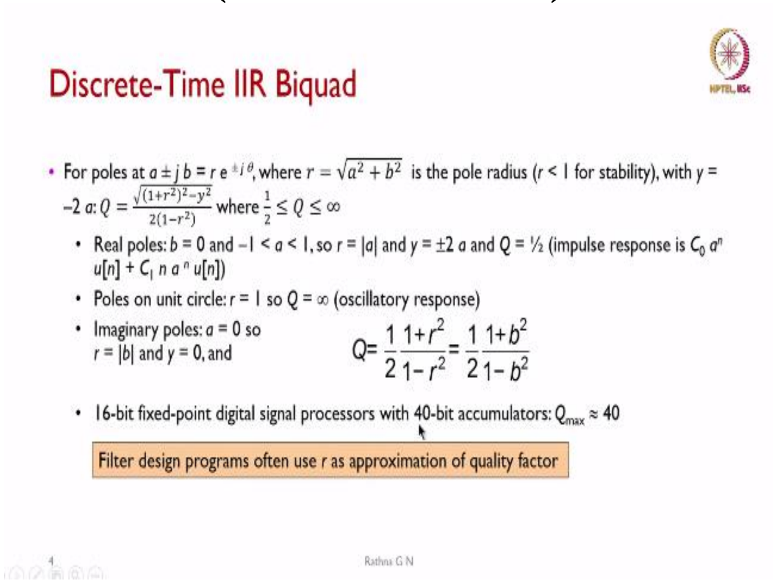
Rathna G N

So, coming to the discrete part of it. So, we will be representing $a \pm jb = re^{\pm j\theta}$. So, where $r = \sqrt{a^2 + b^2}$ is the pole radius. So, assume because it is within the unit circle, we assume $r <$ 1 for stability with $y = -2a$, then what happens to quality factor $Q = \frac{\sqrt{(1+r^2)^2 - y^2}}{2(1-r^2)}$, where we assumed that $\frac{1}{2} \le Q \le \infty$.

So, what happens when the poles are real, we know that $b = 0$, a is in between -1 and then 1. So, $r = |a|$ and $y = \pm 2\,a$ and what we call that is $Q = ½$ that is impulse responses $C_0\,a^n\,u[n] + C_1\,n\,a^n\,u[n]$. So, if the poles are on the unit circle, then $r = 1$. So, $Q = \infty$ that is we call it as oscillatory response and imaginary poles. If we have it, then a becomes 0.

What happens to r$r = |b|$ and $y = 0$. And we call $Q = \frac{1}{2}\frac{1+r^2}{1-r^2} = \frac{1}{2}\frac{1+b^2}{1-b^2}$. So, in the 16 bit fixed point digital signal processor, so we say 40 bit accumulator will achieve $Q_{max} \gg 40$. So, you will be wondering, when we took up the architecture, we said the maximum adder length is going to be 40 bit accumulator what we will consider, so, filter design programs often use r as an approximation of quality factors.

(Refer Slide Time: 05:47)



Discrete-Time IIR Biquad

- For poles at $a \pm j\,b = r\,e^{\pm j\theta}$, where $r = \sqrt{a^2 + b^2}$ is the pole radius ($r < 1$ for stability), with $y = -2\,a$: $Q = \frac{\sqrt{(1+r^2)^2 - y^2}}{2(1-r^2)}$ where $\frac{1}{2} \le Q \le \infty$
  - Real poles: $b = 0$ and $-1 < a < 1$, so $r = |a|$ and $y = \pm 2\,a$ and $Q = ½$ (impulse response is $C_0\,a^n\,u[n] + C_1\,n\,a^n\,u[n]$)
  - Poles on unit circle: $r = 1$ so $Q = \infty$ (oscillatory response)
  - Imaginary poles: $a = 0$ so $r = |b|$ and $y = 0$, and $$Q = \frac{1}{2}\frac{1+r^2}{1-r^2} = \frac{1}{2}\frac{1+b^2}{1-b^2}$$
  - 16-bit fixed-point digital signal processors with 40-bit accumulators: $Q_{max} \approx 40$

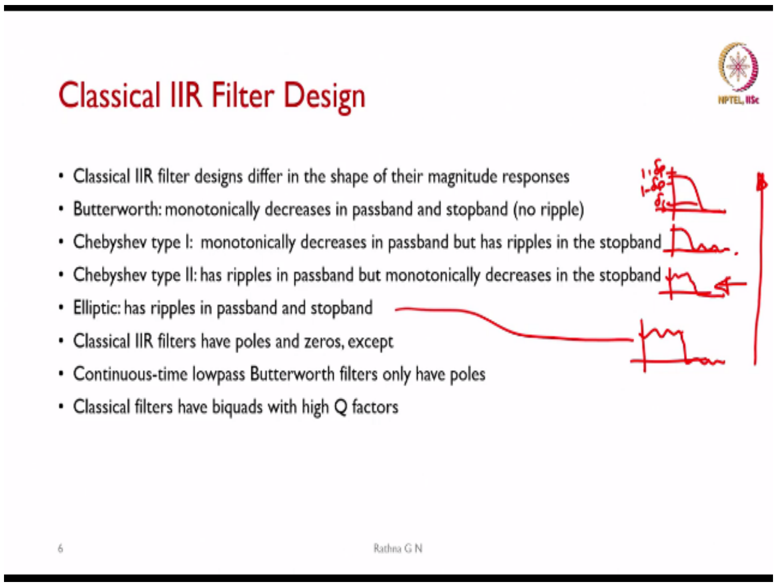
Filter design programs often use r as approximation of quality factor

4 Rathna G N

So, how to implement IIR filter? Same approach in discrete and continuous time filter what we will be considering it, so the classical IIR filter designs what is going to be considered. So, filter order of n will have $n/2$ conjugate roots if n is even, or one real root and then $(n-1)/2$ conjugate roots if n is odd. So, response is very sensitive to perturbations in pole locations, rule of thumb for implementing IIR filter, what is it? Decompose IIR filter into second order section that is Biquads what we call it.

And cascade these Biquads from input or output in order of ascending quality factors. We will see little later also how we will be working out what happens to pole positions and then how it will be affecting the cut off frequency we will see with little problems later. So, for each pair of conjugate symmetric poles in a Biquad conjugate 0s should be chosen as those closest in Euclidean distance to the conjugate poles. This we will see it in the next class, how it is going to be chosen?

(Refer Slide Time: 07:11)



Classical IIR Filter Design

- Classical IIR filter designs differ in the shape of their magnitude responses
- Butterworth: monotonically decreases in passband and stopband (no ripple)
- Chebyshev type I: monotonically decreases in passband but has ripples in the stopband
- Chebyshev type II: has ripples in passband but monotonically decreases in the stopband
- Elliptic: has ripples in passband and stopband
- Classical IIR filters have poles and zeros, except
- Continuous-time lowpass Butterworth filters only have poles
- Classical filters have biquads with high Q factors

6 Rathna G N

Classical IIR filter design how it is done, differ in the shape of their magnitude responses. So, the first one what we call it as the Butterworth filter, which is monotonically decreases in passband and stopband will not have any ripple in that. And when we consider Chebyshev type 1, which is monotonically decreases in passband, but has ripples in the stopband, the Butterworth you will be seeing it this is what we call it as monotonically decreasing and then will not have any ripples in the passband.

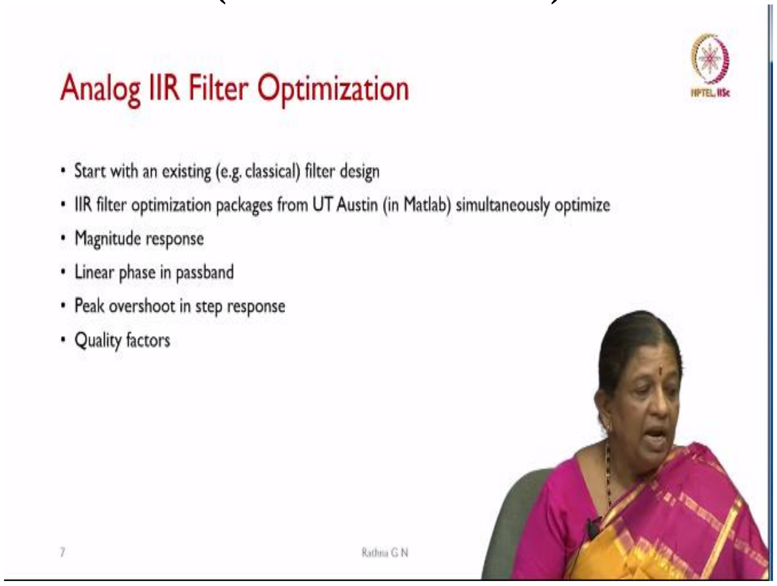
So, when we come to the Chebyshev type 1, so we say I am considering the lowpass filter. So here it is, what it says is monotonically decreasing, but we will be having the ripples in this. So, when you come to Chebyshev type 2 will have ripples in the passband then it is going to be a flat response in the case of stopband when I come to elliptical. So, you will be seeing that we will be having both triples in passband and then stopband. So, which one do we prefer? So, that is one of the challenge one has to look at it.

So, when we consider the order of the filter, it is an increasing order in this way. And when you see the passband attenuation, whether you are going to meet a stopband attenuation which one you want to meet it. So, we call this as my delta s basically where it is coming. And this is my $1 + \delta_p$ what we had represented this is $1 - \delta_p$. So, in this case, whether we are going to select Butterworth, I said the order is going to be more whereas elliptical order is going to be very less compared to other filters.

But I will be having the ripple in passband as well as stopband whereas Chebyshev 1 has ripple in the stopband, whereas Chebyshev 2 have ripple in the passband. So, compared to which frequency we want to meet most of the time, when we use the impulse invariance method, it is better to go for the Chebyshev 2 which is going to be in between order. So, I can allow a little bit of ripples in the passband but I want to meet the flat response in my stopband so that no aliasing is going to happen.

So, coming to order of the filter it is as we said that with respect to Butterworth and elliptic, it stands in between. So, what it says is classical IIR filters have poles and 0s, except continuous time lowpass butter filters only have poles and all classical filters have Biquads with high Q factors. So, we will be seeing why we go for the Biquad in a while.

(Refer Slide Time: 10:26)



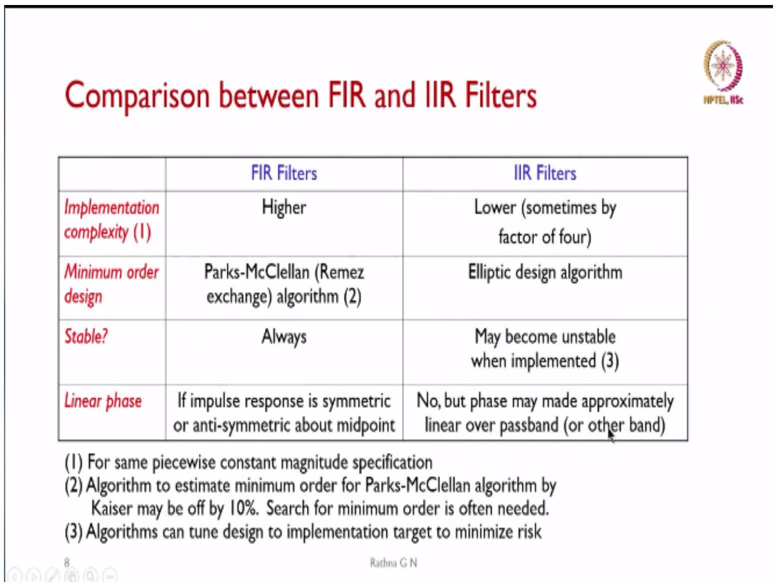
Analog IIR Filter Optimization

- Start with an existing (e.g. classical) filter design
- IIR filter optimization packages from UT Austin (in Matlab) simultaneously optimize
- Magnitude response
- Linear phase in passband
- Peak overshoot in step response
- Quality factors

7 Rathna G N

How we are going to do the optimization? So we will be starting with an existing basically classical filter design we are going to use it and we will do IIR filter optimization packages from UT Austin which is developed in MATLAB simultaneously, you can optimize on the filter order. So, you will be optimizing on the magnitude response linear phase in passband, what we can achieve using IIR filter also and then how we are going to control the peak overshoot in step response and how we are going to take care of the quality factors one has to decide and then do the design.

(Refer Slide Time: 11:06)


Comparison between FIR and IIR Filters

	FIR Filters	IIR Filters
Implementation complexity (1)	Higher	Lower (sometimes by factor of four)
Minimum order design	Parks-McClellan (Remez exchange) algorithm (2)	Elliptic design algorithm
Stable?	Always	May become unstable when implemented (3)
Linear phase	If impulse response is symmetric or anti-symmetric about midpoint	No, but phase may made approximately linear over passband (or other band)

(1) For same piecewise constant magnitude specification
(2) Algorithm to estimate minimum order for Parks-McClellan algorithm by Kaiser may be off by 10%. Search for minimum order is often needed.
(3) Algorithms can tune design to implementation target to minimize risk

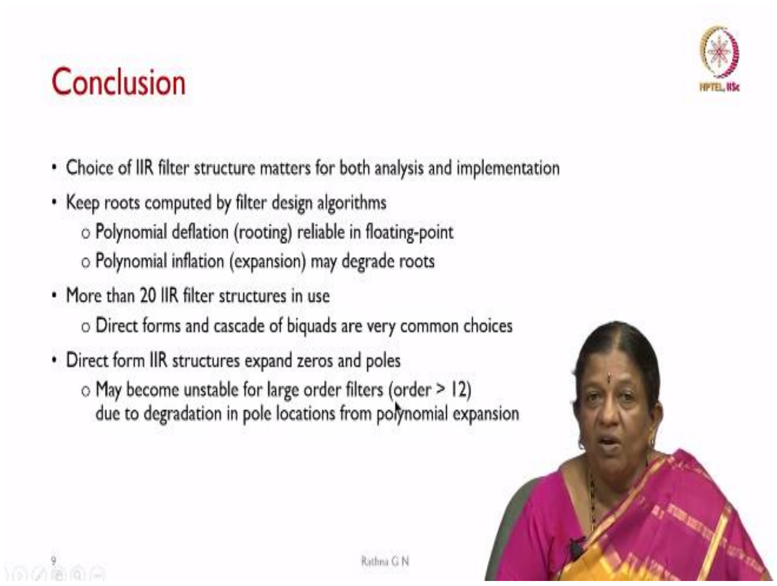
8 Rathna G N

So, we will do a little bit comparison of FIR and IIR filter in this slide. So, when we talk about the implementation complexity, so we will be comparing the same piecewise constant magnitude specification. So, for that FIR filters will have higher order whereas IIR filters what we can achieve is with the lower so, the factor may be 4 lower compared to higher FIR filters. Coming to minimum order design when we want to do the thing so, we can use Parks-McClellan or Remez exchange algorithm. So, what is the consequence of it?

Algorithm to estimate minimum order for this algorithm by Kaiser may be off by 10%. So, we have to search for minimum order is often needed in this case. So, whereas in the IIR filter design, we know that if we allow deviation both in the passband and then stopband, so we can achieve elliptic design algorithm gives us the minimum order. So, whether it is stable, when we are putting a question mark, we know that FIR filter is always stable. So, we say that IIR filter may become unstable, when implemented.

So, we can tune design to implementation target to minimize the risk of it. So, we have to see that all the poles are inside the unit circle so that we will get a stable filter. Coming with the linear phase with can we achieve so we know that impulse response is symmetric or anti symmetric about midpoint then we always achieve a linear phase using FIR filters whereas in IIR filters will not achieve it, but phase may be made approximately linear over passband or other band in this case.

(Refer Slide Time: 13:15)



Conclusion

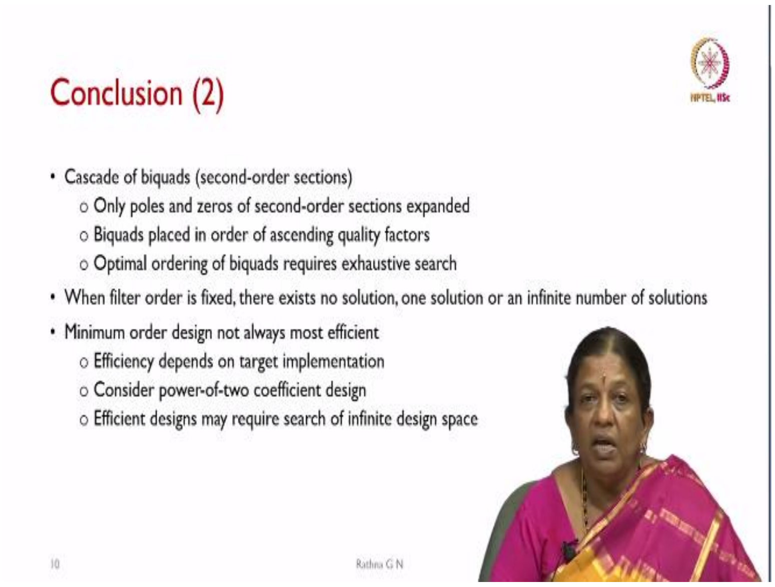
- Choice of IIR filter structure matters for both analysis and implementation
- Keep roots computed by filter design algorithms
  - Polynomial deflation (rooting) reliable in floating-point
  - Polynomial inflation (expansion) may degrade roots
- More than 20 IIR filter structures in use
  - Direct forms and cascade of biquads are very common choices
- Direct form IIR structures expand zeros and poles
  - May become unstable for large order filters (order > 12) due to degradation in pole locations from polynomial expansion

9 Rathna G N

So to conclude on the IIR filter Biquad section, we will see that choice of IIR filter structure matters for both analysis and implementation. So, keep roots computed by filter design algorithms and then a polynomial deflation that is rooting reliable in floating point. So, we will it is going to have polynomial inflation that is expansion may degrade the roots of it. So, more than 20 IIR filter structures in use. So that is direct forms and cascade of Biquads are very common choices. So, why we go for the cascade in IIR filter?

So, we will discuss it in with the problem later so that you will become convinced that why how to go for the cascade section. In the direct form IIR structures expand 0s and poles it may become unstable for large order filters usually are greater than 12th order what we call it due to degradation in pole locations from polynomial expansion. So, most of the MATLAB when we design our filter, we use the MATLAB FTA tool box. So, it provides as we call it as stable filter coefficients, which we will be using it in implementation.

(Refer Slide Time: 14:50)


Conclusion (2)

- Cascade of biquads (second-order sections)
  - Only poles and zeros of second-order sections expanded
  - Biquads placed in order of ascending quality factors
  - Optimal ordering of biquads requires exhaustive search
- When filter order is fixed, there exists no solution, one solution or an infinite number of solutions
- Minimum order design not always most efficient
  - Efficiency depends on target implementation
  - Consider power-of-two coefficient design
  - Efficient designs may require search of infinite design space

10 Rathna G N

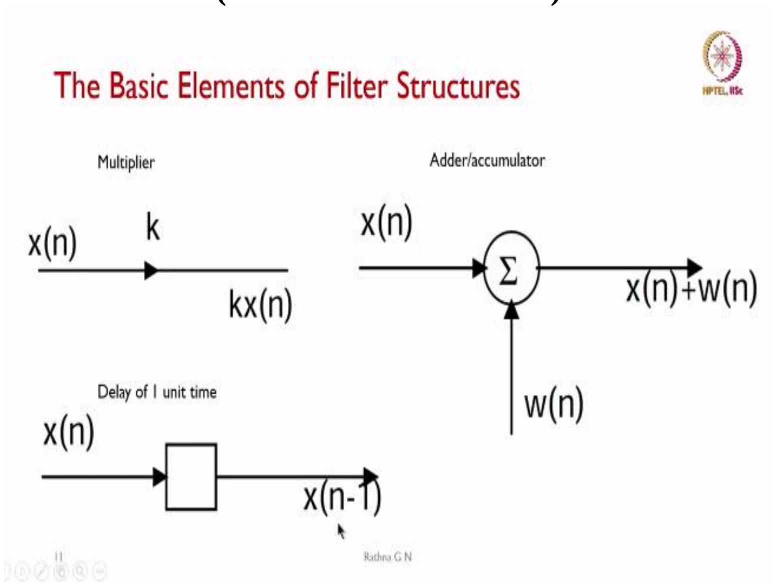
Coming with the further conclusion. So, cascade of Biquads that is second order sections, only poles and 0s of second order sections expanded. So, what happens to this if it is a direct form if there is any quantization in my pole are 0, so the complete structures a may become unstable. Whereas, in the Biquads if any of this poles 0s quantized only it is limited to that structure and it will not affect the other second order sections. Biquads placed in the order of ascending quality factors.

So, this is going to become an optimization problem one has to look at it and then see that how I can achieve this and that is the way that Biquads section will be connected. Optimal ordering of Biquads requires that is what, what it says exhaustive search and when filter order is fixed, there exists no solution, 1 solution or an infinite number of solutions. So, if you fix the order, we may not have a solution or I may have 1 solution, this is the way I have to connect all of them.

If it is a second order I do not have any choice only one Biquad section I can have it but if it is more order is more than 2 then I can have if it is equal to 4 I may have in that case 2 options this can be first that can be later, but still my poles and 0s I can alter from the sections and then see that I achieve the what I will say optimal filter structure as well as my unstability is going to be much more reduced.

So, minimum order designs not always the most efficient in this case. So, efficiency is going to depends on the target implementation. Consider power of 2 coefficient design always and efficient designs may require search of infinite design space. So, these are the drawbacks of IIR filter design.

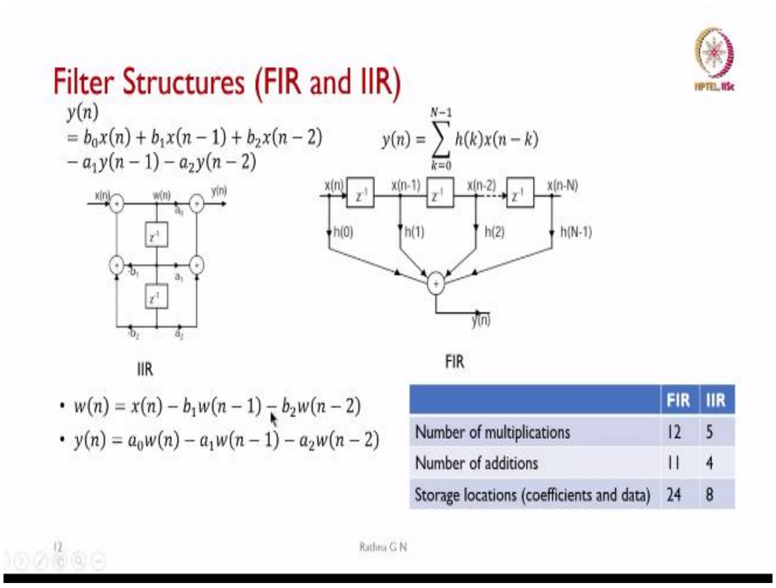
(Refer Slide Time: 17:16)


The Basic Elements of Filter Structures

So with this, we will be still going ahead to design IIR filter. So, what are the basic structures we need for designing filters? So, as previously we have seen the thing but still to give a feel of that how a multiplier looks, so we will be putting an arrow and then if I say k, then it is nothing but $kx(n)$. And when we say adder or accumulator is there, it can be represented with sigma or plus sign also. Then $x(n)$ and then $w(n)$ is the weight, then we will be adding both of them $x(n) + w(n)$.

If it is a delay of 1 unit time, basically it can be represented by a square box. So, you can either specify it as D here or $Z\ -\ 1$. So, then we know that it is a delay element. So, $x(n)$ is the input output will be $x(n-1)$.

(Refer Slide Time: 18:21)



Filter Structures (FIR and IIR)

$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) - a_1y(n-1) - a_2y(n-2)$

$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$

- $w(n) = x(n) - b_1w(n-1) - b_2w(n-2)$
- $y(n) = a_0w(n) - a_1w(n-1) - a_2w(n-2)$

	FIR	IIR
Number of multiplications	12	5
Number of additions	11	4
Storage locations (coefficients and data)	24	8

12 Rathna G N

So, coming to filter structures, just now, we said that comparison we did how the order of the filter is going to be. So here, we are seeing the second order Biquads section for IIR filter. So, you will be seeing that it is a second order for the same magnitude response we need here, FIR filter section is shown, the equation for this is $y(n) = \sum_{k=0}^{N-1} \square$. I as an example, in the table we have taken $N = 30$, then how many structures and other things what it is required we will be seeing in a while.

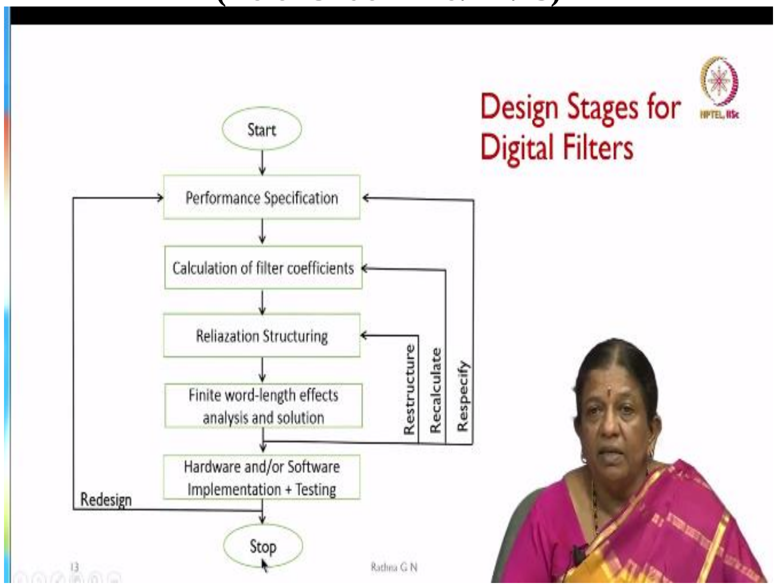
So, whereas for the IIR filter so we will be separating, this is my $w(n)$ centre point. And then second order section, what we have considered here. So, $w(n) = x(n) - b_1w(n-1) - b_2w(n-2)$. What happens to my $y(n)$, $y(n) = a_0w(n) - a_1w(n-1) - a_2w(n-2)$. So, since we have considered this is my feedback section and this is my feed forward section. So, this is my whole equation of second order IIR filter. So, we will be seeing how many multiplications are required?

So, for the same magnitude what we said so IIR filter needs order 2, whereas FIR filter needs 12th order then number of multiplications in this is going to be 12, $h(0)$ to $h(11)$. And then, whereas in the IIR filter because it is a second order, so we will be seeing that number of multiplications in this is going to be 1, 2, 3, 4 and then 5. So, that is what we represent how many additions do we need it? Whereas in this case, we need 11 additions, whereas IIR needs 4 additions you can count on the thing.

Because we assumed addition and subtraction as equivalent. So, you have 1 here 2, 3 and then 4 additions and storage locations that is including coefficients and data. So, how many we need it that is what 1 has to count, we have 12 multiplications and we have 12 what we say is coefficients for the $h(n)$, what we need it and 12 input what we need it and 1 output what we are going to have the thing if current sample is not considered $x(n)$.

Then we will be needing 24 locations to store both coefficient and data output we assume that it is going out of the memory or out of the port, so we need not have to have a storage for it. Whereas in the IIR filter, as you will be seeing that we have 5 coefficients here basically and then the delay elements $w(n-1), w(n-2)$ and then what we have 1 of the input $x(n)$ what we need it so you will be seeing that $5 + 2 = 7 + 1 = 8$ is the number of locations that is memory what we need for the structure, as the order goes very high in FIR filter, you will be seeing that number of multiplications addition increase even the storage will be increasing it.

(Refer Slide Time: 22:23)



Design Stages for Digital Filters

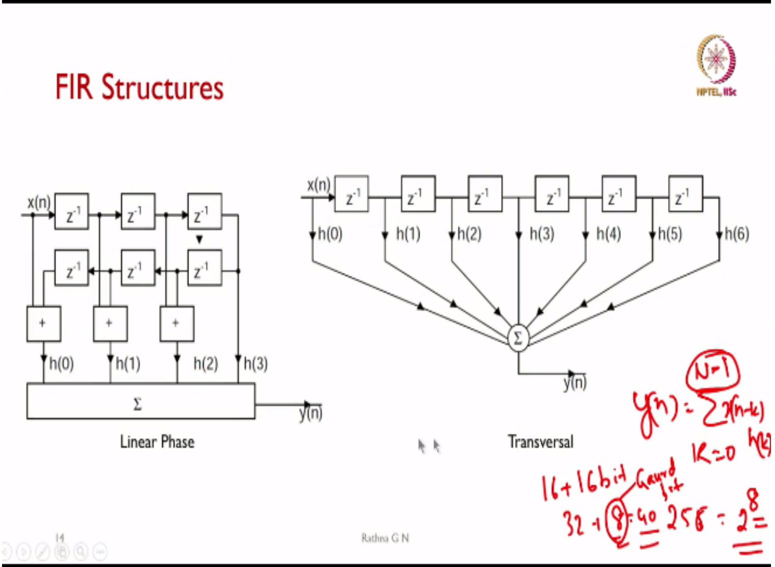
$$\begin{tikzcd}
\text{Start} \arrow[d] \\
\text{Performance Specification} \arrow[d] \\
\text{Calculation of filter coefficients} \arrow[d] \\
\text{Reliazation Structuring} \arrow[d] \\
\text{Finite word-length effects analysis and solution} \arrow[d] \arrow[u, bend right, "\text{Restructure}"'] \arrow[uu, bend right=60, "\text{Recalculate}"'] \arrow[uuu, bend right=80, "\text{Respecify}"'] \\
\text{Hardware and/or Software Implementation + Testing} \arrow[d] \arrow[uuuu, bend left=80, "\text{Redesign}"] \\
\text{Stop}
\end{tikzcd}$$

So, how we are going to consider the design stages for digital filter? So, we are going to start first will be specifying the performance first and then calculate the filter coefficients. And then if everything is according to our wish, then we will go for the realizing structure, what we have to select and then see that finite word length effects analysis and solution. Once we have selected that, whether it is going to affect input and output, magnitude and then frequency responses, it becomes an unstable filter.

So, if everything is met, then we will be going for hardware and or software implementation plus testing. And then if it is not met here, we will go for the redesign of it. So, if any one of the stage here, the finite word length is not met, we can go and then fine tune any one of these stages. So, if everything is feasible, then we will stop design.

(Refer Slide Time: 23:43)

FIR Structures



$y(n) = \sum_{k=0}^{\boxed{N-1}} x(n-k)\, h(k)$

16 + 16 bit Guard bit

$32 + \boxed{8} = \underline{40}$ $256 = \underline{2^8}$

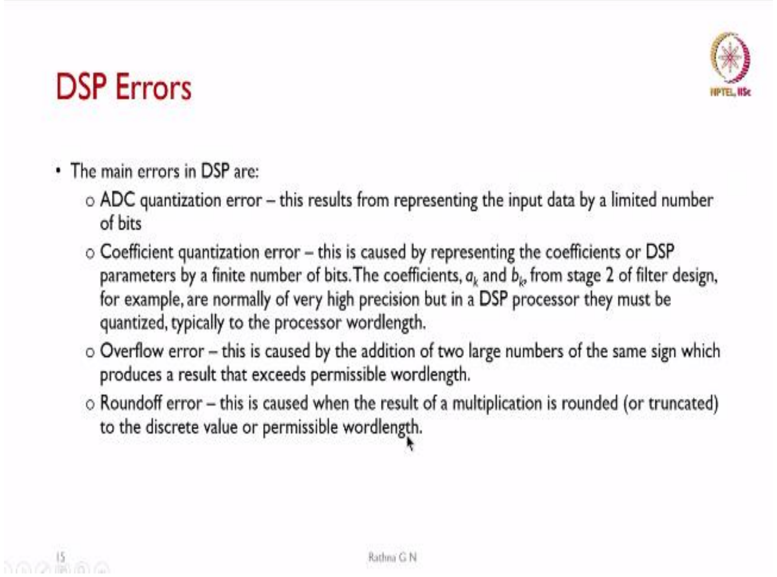
So, as we said some of the structures also matters. So, few FIR structures, you will be seeing it this one gives our linear phase as you will be seeing that we said it is a symmetric or anti symmetric what we are going to take the thing. So, here you will be seeing that this is $h(0)$ that is $x(n)$ and then x of what is the delay here you are going to get it so 1, 2, 3, 4, 5 and then 6. So, $x(n)$and $x(n-6)$ is combined and then you will be multiplying with $h(0)$.

The same way you can calculate which are ones getting multiplied and then you will be summing up then you will get a linear space. So, if you want to use the transversal as you will be seeing that. So, $h(0)$ to $h(6)$ are my impulse response. And then $x(n)$ to $x(n-6)$ will be the input and when you feed it into the thing, so you will be seeing a single summation. So, you may overflow or underflow in this case, so, $y(n)$ will be the output how much tolerance you are going to give it as I said we are using the 40 bit adder.

So, the order of the filter in sigma can usually be represented $y(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$ what we have it .So, this value N can be up to 255 so which is in power of 2 if you take it 2^8, 0 to 255 so it becomes 256 which is equal to 2^8. So, what we say if the adder is $16 + 16$ bit what we are going to add or registers are 32 bit, I can allow a overflow of 8 more bit that is $32 + 8$ we call this as the guard bits in accumulator.

So, this we call it as guard bit in DSP processors to take care of 40 bit addition so, up to 255 that we may not have a overflow beyond that, we may overflow in the FIR design. So, one has to take care as we said it is almost stable, but the order goes very high then it may become unstable.

(Refer Slide Time: 26:46)



DSP Errors

- The main errors in DSP are:
  - ADC quantization error – this results from representing the input data by a limited number of bits
  - Coefficient quantization error – this is caused by representing the coefficients or DSP parameters by a finite number of bits. The coefficients, a_k and b_k, from stage 2 of filter design, for example, are normally of very high precision but in a DSP processor they must be quantized, typically to the processor wordlength.
  - Overflow error – this is caused by the addition of two large numbers of the same sign which produces a result that exceeds permissible wordlength.
  - Roundoff error – this is caused when the result of a multiplication is rounded (or truncated) to the discrete value or permissible wordlength.

15 Rathna G N

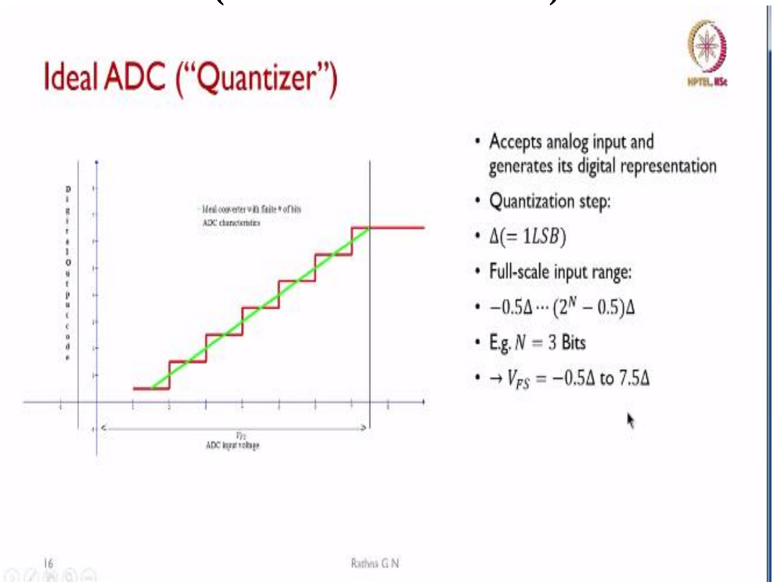
So, what are the DSP errors we are going to encounter in design. So, it has been listed main errors in DSP are first is ADC quantization error. So, this results from representing the input data by a limited number of bits. So, I can choose 12 bit ADC, 10 bit, 8 bit depends on what type of ADC you want to have it and then next is as we have to have a coefficient to be represented in fixed point, so I am going to have the quantization error here.

So, that is what, what it says represent the coefficients or DSP parameters by a finite number of bits. So, the coefficients a_k and b_k from stage 2 of filter design, for example are normally of very high precision, but in a DSP processor, they must be quantized typically to the processor wordlength. And then next is once I have done the coefficient quantization error I have looked in next is overflow error, as we are talking about addition and subtraction can give us overflow or underflow.

So, we say addition of 2 large numbers of the same sign, which produces a result that exceeds are permissible wordlength this is one more error, what we have to look it. The last one is the roundoff error. So, what is this? This is caused when the result of a multiplication is rounded or truncated to the discrete value or permissible wordlength. So, when I do multiplication, 16 bits into 16 bits, I will be resulting in 32 bit, but I would not be able to represent that complete 32 bit as output.

So, I will be truncating them or rounding them to 16 bit and then I will be storing so we have taken one example already in fixed point, multiplication and addition.

(Refer Slide Time: 28:58)



Ideal ADC ("Quantizer")

- Accepts analog input and generates its digital representation
- Quantization step:
- $\Delta (= 1LSB)$
- Full-scale input range:
- $-0.5\Delta \cdots (2^N - 0.5)\Delta$
- E.g. $N = 3$ Bits
- $\rightarrow V_{FS} = -0.5\Delta$ to 7.5Δ

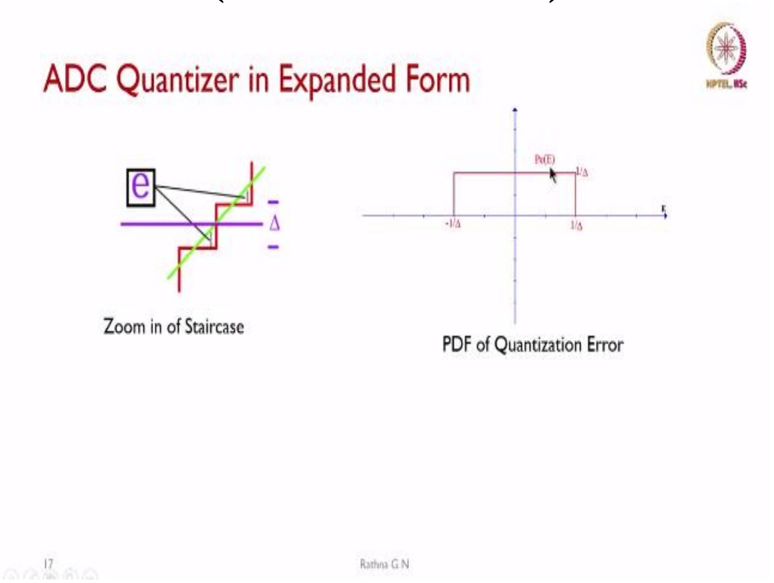
16 Rathna G N

So, these are the errors. So first, we will see how we are going to have ADC is going to cause as an error. So, we know that in the analog domain, the values are represented in continuous way. So, my ADC input voltage, what we have it on the X axis and Y axis will represent what is the value or that depends on the number of bits what I have it, so I will be representing two one of them. So, we know that so from this 1 voltage to 2 voltage I will be representing with the same value and I will be changing it.

So, this is the staircase, what you can look at it. So, what we want ideal converter with finite number of bits, we have a say that ADC characteristics is one of the thing what it required. So, what we say is it accepts analog input and generates a digital representation. So, what is the quantization step we call it as delta. So, it is going to be -1 LSB bit what it is going to be. So, if we have full scale input range is represented as this thing $-0.5\Delta \cdots (2^N - 0.5)\Delta$. So, the total will be -1 what we are going to have LSB representation.

So, N is the number of bits that one processor has, as an example, if we assume it is a 3 bit processor, then the full voltage scale what we can represent with this representation is -0.5Δ to 7.5Δ. So, this is the full scale range of ADC.

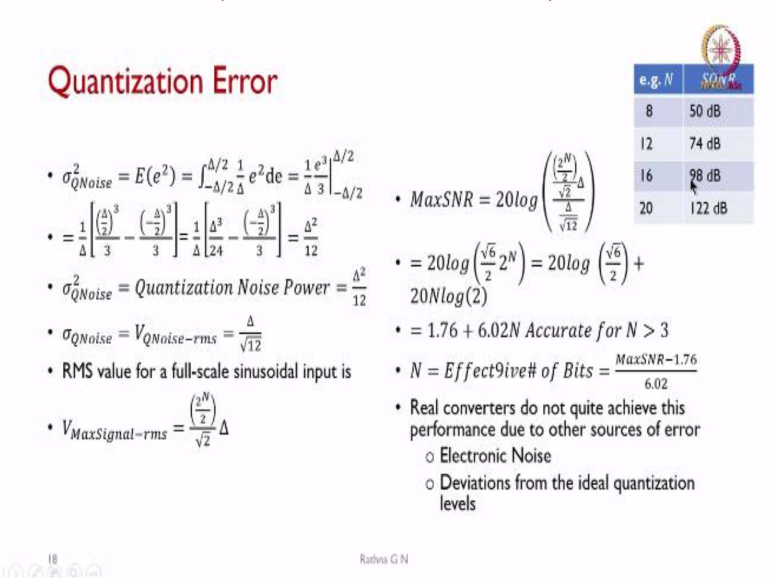
(Refer Slide Time: 31:03)



ADC Quantizer in Expanded Form

To show that how the quantizer in expanded form? So, what is this delta? So, you will be seeing that from this point to this point, this is we assume it is – 0.5 and this side is – 0.5. So, which will be giving me Δ so, that is what, what we say that is the error what we will be encountering from ADC quantization. So, this is represents a zoom in staircase thing. So, if we consider the PDF of quantization error, then I will be representing it as $-\frac{1}{\Delta}$ to $\frac{1}{\Delta}$. And then this is your expected value what you will be getting it here P_0E here, which is what we call it as $\frac{1}{\Delta}$.

(Refer Slide Time: 31:54)



Quantization Error

- $\sigma^2_{QNoise} = E(e^2) = \int_{-\Delta/2}^{\Delta/2} \frac{1}{\Delta} e^2 \text{de} = \frac{1}{\Delta}\frac{e^3}{3}\Big|_{-\Delta/2}^{\Delta/2}$
- $= \frac{1}{\Delta}\left[\frac{\left(\frac{\Delta}{2}\right)^3}{3} - \frac{\left(-\frac{\Delta}{2}\right)^3}{3}\right] = \frac{1}{\Delta}\left[\frac{\Delta^3}{24} - \frac{\left(-\frac{\Delta}{2}\right)^3}{3}\right] = \frac{\Delta^2}{12}$
- $\sigma^2_{QNoise} = Quantization\ Noise\ Power = \frac{\Delta^2}{12}$
- $\sigma_{QNoise} = V_{QNoise-rms} = \frac{\Delta}{\sqrt{12}}$
- RMS value for a full-scale sinusoidal input is
- $V_{MaxSignal-rms} = \frac{\left(\frac{2^N}{2}\right)}{\sqrt{2}}\Delta$

- $MaxSNR = 20log\left(\frac{\frac{\left(\frac{2^N}{2}\right)}{\sqrt{2}}\Delta}{\frac{\Delta}{\sqrt{12}}}\right)$
- $= 20log\left(\frac{\sqrt{6}}{2}2^N\right) = 20log\left(\frac{\sqrt{6}}{2}\right) + 20Nlog(2)$
- $= 1.76 + 6.02N\ Accurate\ for\ N > 3$
- $N = Effect9ive\#\ of\ Bits = \frac{MaxSNR-1.76}{6.02}$
- Real converters do not quite achieve this performance due to other sources of error
  - Electronic Noise
  - Deviations from the ideal quantization levels

e.g. N	SQNR
8	50 dB
12	74 dB
16	98 dB
20	122 dB

18 Rathna G N

So, coming with the quantization error, how we are going to calculate it? So, we have to can take it as a noise. So, we will be noise variance sigma quantization noise square, what we will be calculating, so which is nothing but the expected value of error square. So, when we equate it, it is $\int_{-\Delta/2}^{\Delta/2} \frac{1}{\Delta} e^2 \text{de}$. So, which is equivalent to $\frac{1}{\Delta}\frac{e^3}{3}\Big|_{-\Delta/2}^{\Delta/2}$.

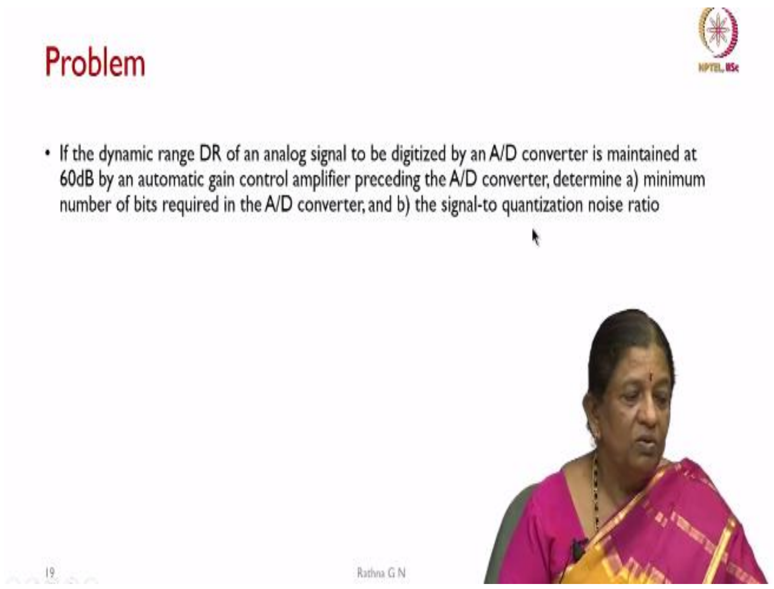
So, when you expand it, you will get it as $\frac{\Delta^2}{12}$. So, we say that quantization noise sigma square is given by or power is given by $\frac{\Delta^2}{12}$. And then we assume quantization noise V that is voltage quantization noise rms value is given as $\frac{\Delta^2}{12}$. So that is we say RMS value of a full scale sinusoidal input is given by if your maximum signal RMS value is represented as $\frac{\left(\frac{2^N}{2}\right)}{\sqrt{2}}\Delta$.

Then we will be calculating maximum signal to noise ratio, this is my signal and then this is my noise. So, it is $20log$ base what will be calculated it this is my signal this is my noise. So, when we equate this, it approximately comes out as $20log\left(\frac{\sqrt{6}}{2}2^N\right) + 20Nlog(2)$. So, if we approximate it, it will be $1.76 + 6.02\,N$ accurate, this equation is for $N > 3$. So, N is the effective number of bits, what we are going to have it in representation.

So, it is number of bits what it says is maximum signal to noise ratio $-\ ^{1.76}/_{6.02}$ what we will be getting the thing. So, what it says is real converters do not quite achieve this performance due to other sources of error. So, only we have taken signal to noise ratio that may mean other errors we may a like electronic noise or deviations from the ideal quantization level, which will cause the error mode.

To calculate for $N = 8$. So, what we are going to get signal to quantization noise ratio is going to be 50 dB what I can achieve 8 bit this one if I have $N = 8$, then what I will be achieving is the 50 dB if $N = 12$, I can achieve 74 dB signal to quantization noise ratio when it is 16 bits N then it will be 98 dB what I can achieve. And if it goes to 20 bits, what I can achieve is 122 dB CD quality what I was telling is approximately 90 dB what we want to have it so we can achieve with 16 bit that quality signal to noise ratio.

(Refer Slide Time: 35:36)



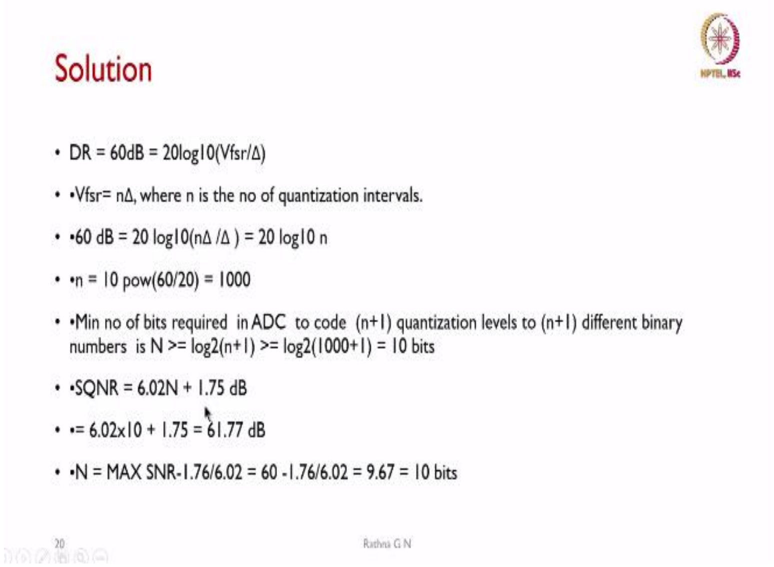
Problem

- If the dynamic range DR of an analog signal to be digitized by an A/D converter is maintained at 60dB by an automatic gain control amplifier preceding the A/D converter, determine a) minimum number of bits required in the A/D converter, and b) the signal-to quantization noise ratio

19 Rathna G N

So, coming will work out a problem. So, whether you have understood dynamic range and then what should be number of bits required. So, if the dynamic range of that is DR what we call it as an analog signal to be digitized by an A to D converter is maintained at 60 dB by an automatic gain control amplifier preceding the A to D converter. So, what are the things we had to determine one is minimum number of bits required in A to D converter and what will be a signal to quantization noise ratio. So, this is what, what we have to find out so, just will solve the problem.

(Refer Slide Time: 36:18)



Solution

- DR = 60dB = 20log10(Vfsr/Δ)
- •Vfsr= nΔ, where n is the no of quantization intervals.
- •60 dB = 20 log10(nΔ /Δ) = 20 log10 n
- •n = 10 pow(60/20) = 1000
- •Min no of bits required in ADC to code (n+1) quantization levels to (n+1) different binary numbers is N >= log2(n+1) >= log2(1000+1) = 10 bits
- •SQNR = 6.02N + 1.75 dB
- •= 6.02x10 + 1.75 = 61.77 dB
- •N = MAX SNR-1.76/6.02 = 60 -1.76/6.02 = 9.67 = 10 bits

20 Rathna G N

So, our dynamic range is given a 60 dB what we have to achieve. So, we have the equation $20\ \log 10(\mathrm{Vfsr}/\Delta)$. So, in this case we will take it as $n\Delta$ where n is the number of quantization intervals what do I want to have it so, when we substitute this it is going to be $20\ \log 10(\mathrm{n})$. So, $n\ =\ 10$ power substitute them 60/20 which is going to give me 1000.

So, you know that minimum number of bits required in ADC to code $n + 1$ will always be take it quantization levels to $n + 1$ different binary numbers is N should be greater than or equal to $\log 2(\mathrm{n} + 1)$. So, if we do that $\log 2(1000 + 1)$. So, we say that what I need is 10 bits what I need it, so 2 power 10 we know that it is 1024 bits what I will be representing with that is the nearest So, now we had to calculate signal to quantization noise ratio with number of bits selected as 10.

So, if you substitute 6.02 into N + 1.75 dB, so which is substitute the thing what I will be getting a signal to quantization noise ratio of 61.77 dB. So, the maximum of what we can achieve what it says is number of bits if we substitute in this equation maximum signal to noise ratio -1.6 divided by 6.02. So, if you see with this equation also you will be getting it as 9.67 bits what I need it so, the nearest one is 10 bits.

So, both this way working out and using this equation what we had it so both of them are giving us 10 bits. So, you can try which is going to be less than 3 bits whether they are going to match or not. So, this is what, what we cover in today's class so we will see how the filter structure is going to affect frequency component part of it centre frequency in the next class. Thank you.