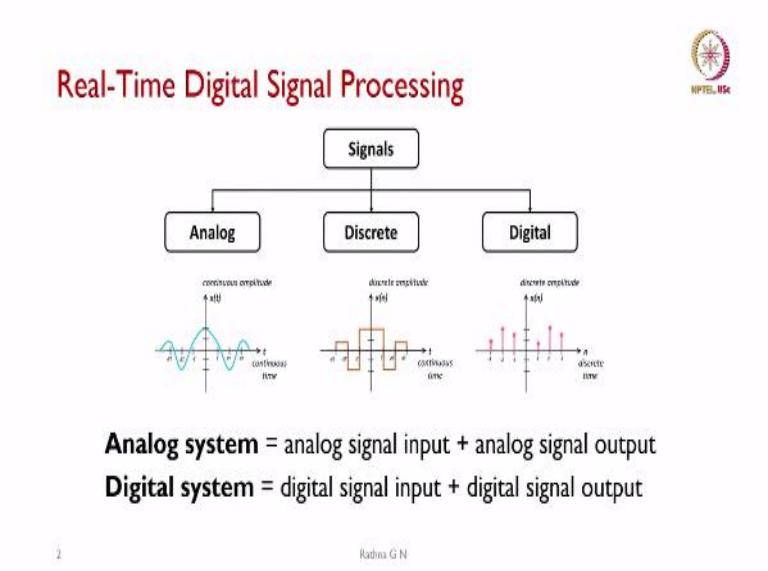
Real – Time Digital Signal Processing
Prof. Rathna G N
Department of Electrical Engineering
Indian Institute of Science - Bengaluru

Lecture – 02
Basics of Signal Processing

So, last class we discussed about what are the basics of signal processing, what course is going to discuss about it. Today we will discuss about a little bit on basics of signal processing.

(Refer Slide Time: 00:37)



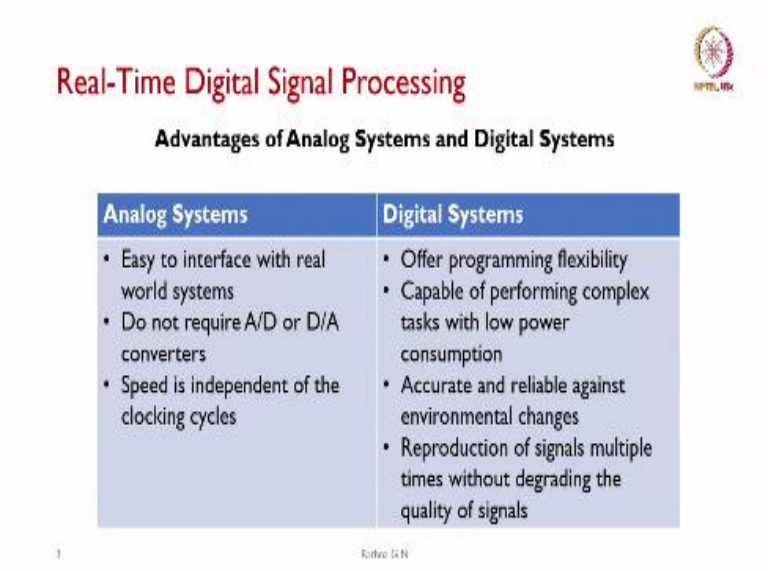
So, the real time signal processing has 3 components basically what you will see that some of the signals are represented with. The first part is the analog signals. So, what you will be seeing is the continuous signal, you will be seeing that the amplitude is continuous as well as the time is going to be continuous. So, coming to the discrete signals how it is going to be representing it is shown in this figure. So, the frequencies as you can see continuous or continuous time what we call it and then the amplitude is going to be discrete in this.

So, when we come to represent the signals in the digital domain that is in the digital form. So, we will see that the amplitude is discrete, as well as it is going to be discrete in time. So, as it is represented with n. So, these are some of the examples of the signals when we come to the system how we are going to represent that. Analog system is represented with along with the signals

whatever we are processing input is going to be an analog signal basically input and then the even the output after performing the processing.

So, the output is going to be represented in the analog domain itself that we call it as an analog signal output. So, when we come to represent the system or digital systems basically, input is going to be digital signal input and then we will be doing the processing, the output again it will be in the digital signal output what we are going to get from the system.

(Refer Slide Time: 02:21)



So, coming to some of the advantages of analog systems or digital systems, so, we will say that analog systems the first one is it is easy to interface with real world systems. And then we say from the point of view of digital system, they offer programming flexibility basically, and then capable of performing complex tasks with low power consumption. So, whereas, in the case of analog systems we do not need any A to D or D to A converters because input is in the analog format and even output what we want it in the analog domain.

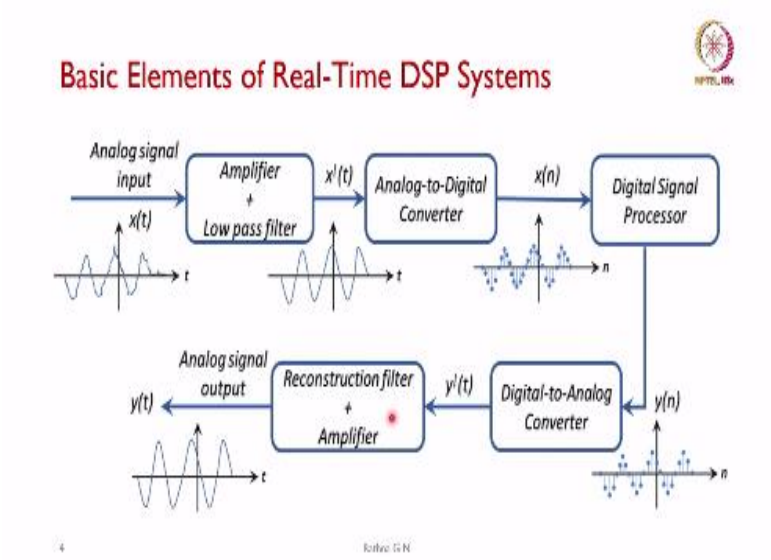
So, we need not have to convert it into signals in from analog to digital or digital to analog. And then whatever we are talking about the speed, is going to be independent of the clocking cycles. So, there is no need to sample the signals whereas, in the digital domain, so, we have to do sampling has to be done and then it depends on the sampling frequency and the frequency component present

in the input signals. So, these are, one of the advantage in this case is going to be accurate and reliable against environmental changes.

So, we know that analog when we do the processing, so, some of the components may be resistors or capacitors or other hardware what we will be using it for processing. So, we know that for the weather conditions, as we say that precision basically, percentage of the precision, so, which may vary depending on the temperature and other environmental changes that is going to occur. We know that in the digital systems they are not going to be affected by this environmental changes.

And one more advantage of the digital system is reproduction of signals. So, that is going to be multiple times what we can do it without degrading the quality of signals. In this case what we say is if I have to little bit modify or the same signal what I want to get it in different places which is possible, whereas from the analog domain, we know that because the environment is going to play a havoc so, if we are not taken care of these things, the multiple outputs for the same input data, we may get it differently. So, this is one of the disadvantages of analog systems.

(Refer Slide Time: 05:03)



So, coming to when we have talked about so much of advantage of digital systems, so we will see what are the basic elements of real time system DSP systems are going to have it. So, one size the input, if it is an analog signal input as we will see that it is both varying in time as well as amplitude,

which is represented as $x(t)$. So, we have, if it is the amplitude very low, then we may have to do a little bit of amplification.

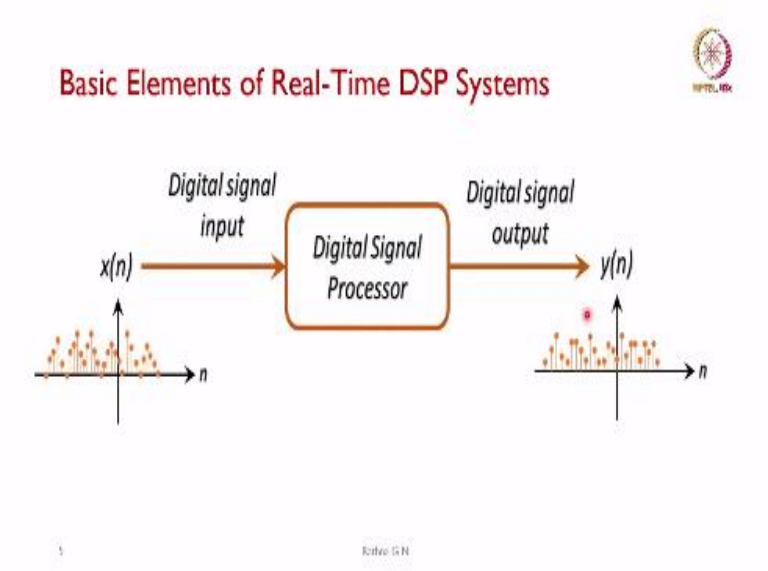
And then to curtail the frequency component present in the analog signal, we have to pass it through the low pass filter. So, we call in the analog domain as the anti aliasing filter. So, we will come to the aliasing part of it in a while, then once we have converted we call it as $x'(t)$t. So, as you can see that some of the amplitude is going to be discrete and then it may be continuous in time domain, then we pass it through the analog to digital converter. So that ADCs are going to convert analog signals into digital domain.

So, we call it as $x(nT)$ in this case, as you will be seeing that the samples what we are going to get from the analog is represented in this figure. Then once this is in the digital signals, it is both in amplitude as well as in time that being discretized will fit it as input for our digital signal processor. So, for the digital signal processor input as we know that it is going to be discretize signal both in amplitude as well as in time what we will be feeding it.

And will do the processing whatever application we are intended to then the output is available as a digital output, as we will be seeing it $y(n)$ in this case, after processing of it, then if it has to be converted back to analog domain, then we have to pass it through the digital to analog converter, so that we will be getting $y'(t)$ as the output then what we have to do is we need analog output in the continuous domain. So, we have to do a little bit of amplification and then we will use the reconstruction filter.

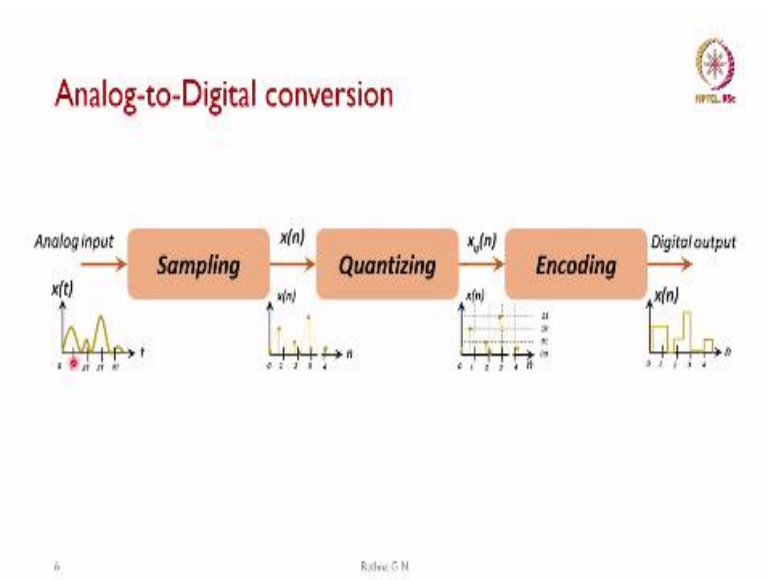
Reconstruction filters are going to be usually sync filters so when time permits, we will take it up and then we will be getting the analog signal output. So, if we want to store digital signals in the digital form, so in the compression or whatever, we can store it in that domain itself. Whenever we want to have the analog output that time we can retrieve the signals and then convert it back into analog domain.

(Refer Slide Time: 08:04)



So, coming to the extending the basic elements of the real time DSP systems, so we have the $x(n)$ as the input. And then this is the digital signal input what we are going to provide it to the DSP processor and then we are going to process it and then we will be storing digital signal output as I have mentioned in the previous slide, so we can store it after computation what has been done in the digital domain itself.

(Refer Slide Time: 08:34)



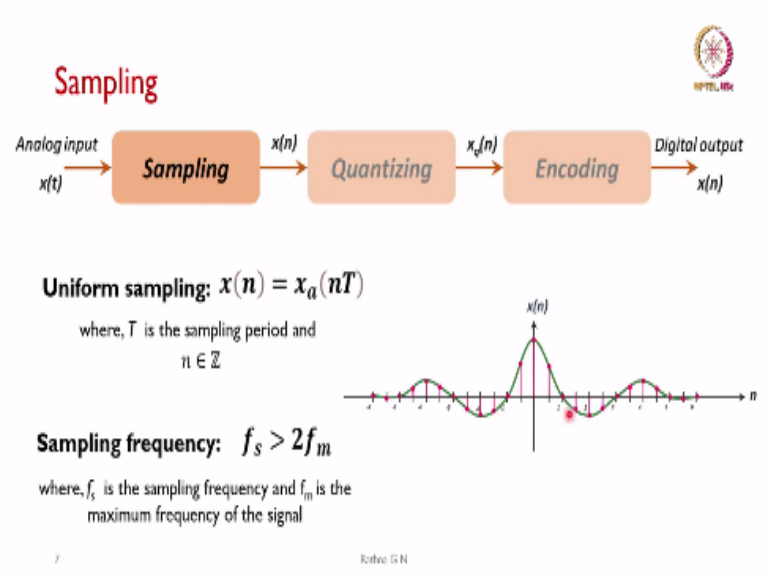
So, now we will see a little bit of what we will be doing how to do analog to digital conversion that is ADC what we say it. So, we know that first is the analog input as it is shown here, basically, which is continuous in time as well as in amplitude. So, the first step in this going to be, we have to do the sampling, what we have to do it. Sampling frequency has to be fixed, which depends on

the nyquist criteria basically, and then $x(n)$ is the input to our quantize because we know that although we have the discrete amplitude and discrete in time.

We know amplitude we cannot represent it in the digital domain as we want it. It depends on number of bits what we are going to use it. So, we have to quantize our input signal, and then after quantization, so we will be storing it as $x_q(n)$. So, as the quantization process little bit is shown and then a little more will be showing in the next slides actually. So, it will be seeing that number of bits. This is the digital output which has to be encoded, and then we will be storing it as a digital input as it is shown.

So, we will see that analog signal was this when it comes to the digital domain. So, this is how the representation with respect to amplitude as well as time what we will be seeing it, it is discretized.

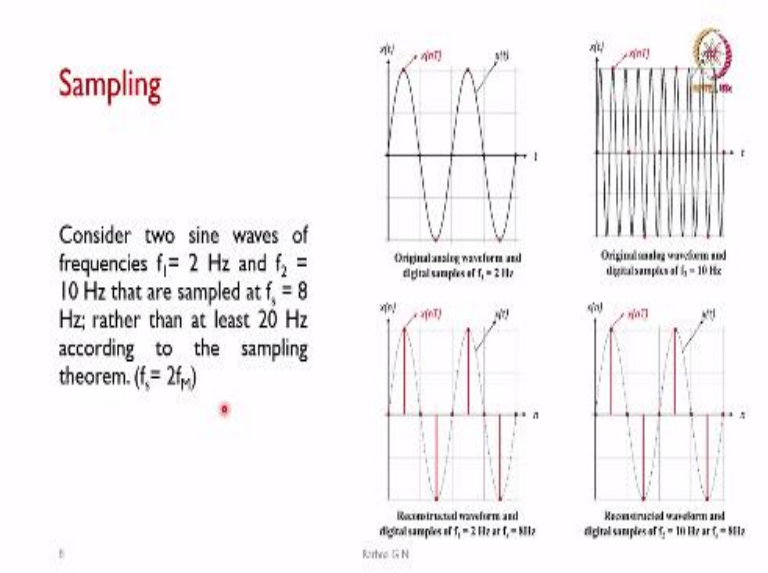
(Refer Slide Time: 10:16)



So, as a first step in this block diagram in the conversion is the sampling. So, we will be seeing that what we call it is uniform sampling. That is $x(n)$ is the output what we want after sampling. So, for that, this is from the analog signal x_a, so which is going to be sampled for every n and then T is the period what we will be considering. So, T is our sampling period and then n will be belonging it to our real world.

And then when we come to sampling frequency, how we are going to fix the thing. So, what we say is f_s is a sampling frequency it should be greater than 2 times of the f_m, $f_s > 2f_m$where f_m is the maximum frequency that is present in our analog input. So, as we can see, this is the analog signal. And then what is the duration here what we call it as delta t is the sampling period what we will be taking it here between 2 samples, and then the complete thing sampling frequency what f_s is going to be represented depending on the maximum frequency present in the signal.

(Refer Slide Time: 11:32)



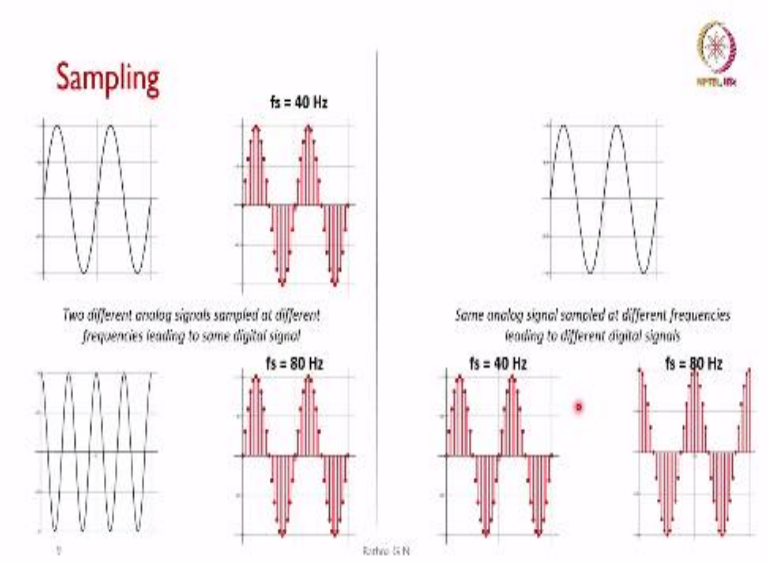
So, just as an example here we will consider 2 sine wave frequency i.e we have f_1 as 2 hertz and then the second frequency component is 10 hertz, and then we are going to sample this with 8 hertz sampling frequency. So, we said that according to the previous equation, sampling frequency should be greater than twice that of the maximum frequency whereas in this case it should be according to this maximum frequency 10 hertz it should have been 20 hertz instead of that what we are doing is sampling at 8 hertz.

So, then what happens to this. So, you will be seeing that this is the first frequency what I have it at 2 hertz, and then this is the 10 hertz frequency component which is represented when I try to reconstruct these 2 signals back to original after doing processing and everything. So, you will be seen that this is going to be 2 hertz basically, that is we have f_1 is at 2 hertz and then the sampling frequency is 8 hertz.

So, which has reconstructed properly so, what have in this case is how many points I am going to have there are 4 points, 2 points what it has in maximum what you are representing the thing in both domain that is in the positive and then the negative what it has it and then when I reconstruct this signal, 8 hertz signal, so, what happens to the thing is, it is going to have the aliasing effect.

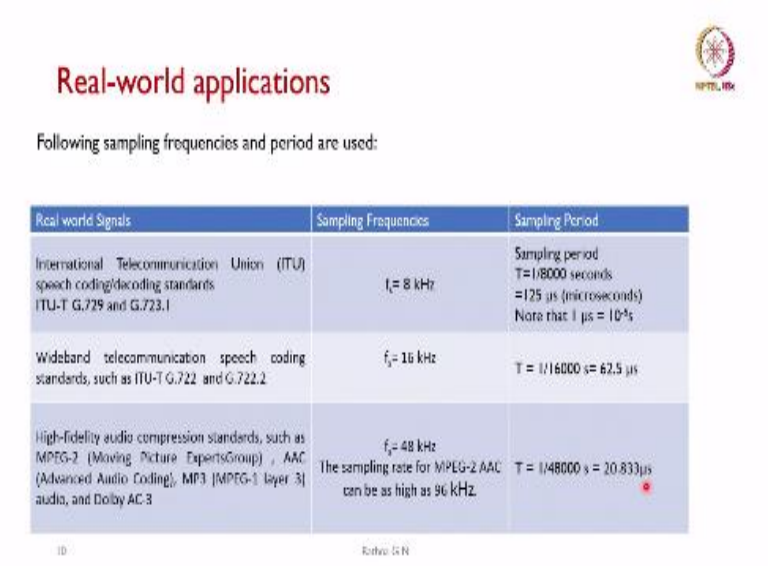
So, because I have not taken care of twice that of the maximum frequency, so, we will be seeing that 10 - 8 hertz is the one reflected basically what I will be getting both as a 2 hertz frequency in the domain. So, some of these examples, we will be doing it in the lab, so that you will be able to figure it out, what will be the reconstruction is going to happen if normal whatever Nyquist criteria frequency is not met.

(Refer Slide Time: 14:02)



So, coming to the other part of the sampling, so, you will be seeing that 2 different analog signal sampled at different frequencies leading to some digital signal what we are looking at. So, we have sampled at 40 hertz here. And then this one is sampled at 80 hertz, what you will be seeing it, so, when I do the reconstruction, so the 40 hertz if the sampling frequency is 80 hertz, so, you will be representing both the signals in their domain reconstructed properly.

(Refer Slide Time: 14:45)



So, coming to a real world application so, we will be seeing a few of the sampling frequencies and then what are the sampling periods associated with each one of them. So, one of the real world is our telecommunication. So, we know that International Telecommunication Union that is ITU what it is going to be representing speech coding and decoding standards, which is defined by ITU T what we call it G 729 and then 723.1 standards, there the sampling frequencies what it is used is 8 kilohertz.

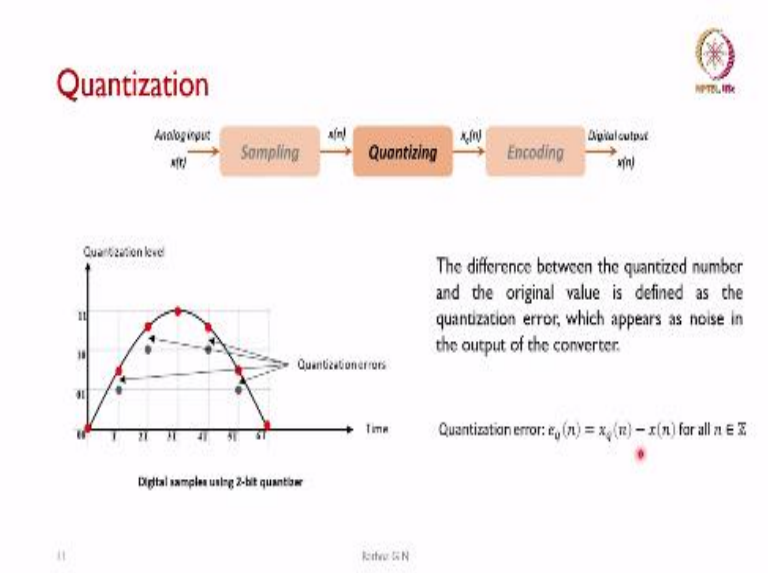
So, we know that 8 kilohertz when we say the thing what will be the sampling period is T is going to be given by $1/f_1$ which is going to be 1 by 8000 seconds. So, which constitute 125 microseconds basically what will be representing that, so, how we are going to represent 1 microsecond is equal to 10^{-6} seconds, the same thing so, if we consider the wideband telecommunication speech coding standards such as ITU T G 722 and then 722.2.

Here the sampling frequency is 16 kilohertz then we know that sampling period is going to be $1/16000$ which is nothing but 62.5 microseconds coming to the third real world signal. So, what we call it as high fidelity audio compression basically standards such as MPEG 2 that is moving picture basically processing what we will be looking at it express group basically, and then AAC standards that is Advanced Audio Coding and then mp3 we know that coding technique, so, which is we call it as MPEG 1 layer 3 what it is going to be considered for the audio.

And then we know that Dolby AC 3 systems also.

In these applications, the sampling frequency that is used is 48 kilohertz, and then we know that for the sampling rate for MPEG 2 AAC, it can be as high as 96 kilohertz. So, in that case, what happens our sample period is if we consider 48 kilohertz, which is 1 / 48000 which is going to be 20.83 microseconds is the sampling period at which we have to sample the signals basically. So, these are some application shows that how we have to do the sampling correctly.

(Refer Slide Time: 17:32)

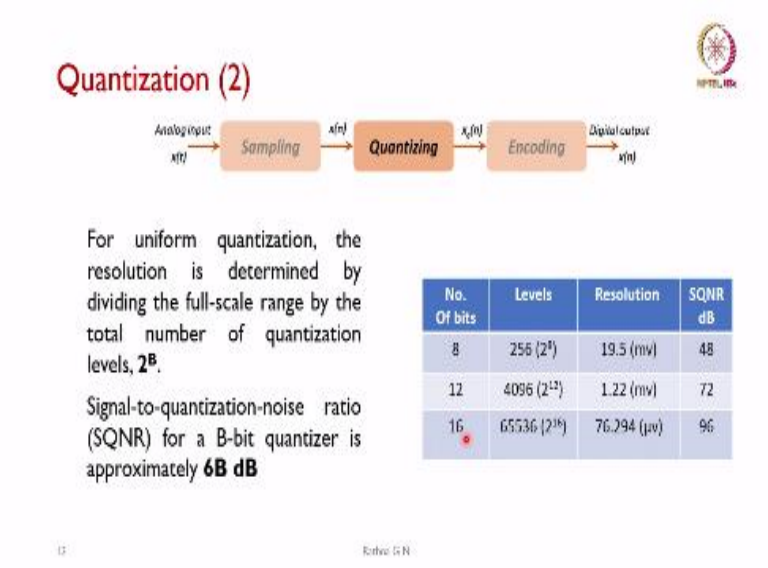


So, now, once we have sampled the signals, then the signal is available in the digital domain which we call it as $x(n)$ then we have to go for the quantization. So, it depends on as I said number of bits, in this case as an example, we are taking 2 bit quantizer that is whatever the input which is coming, so, we will be representing with 2 bits, then I know in the digital domain how we will be representing the 2 bit number system.

So, that is 00 and then 01 10 & 11. So, this is what my input signal. So, in the continuous domain, so, our sample signal what I have got it so I have to map them to the nearest numbers basically whatever I will be represented with the number of bits. So, we call whatever the difference between the 2 as we can see one of the example this number which is between 10 and 11, which I will be representing as 10.

So, which we call all these errors as quantization errors, that is basically the difference between the quantized number and the original value is defined as the quantization error which appears as noise in the output of the converter. So, how do we define the quantization error which is given by equation like this $e_q(n)$ which is nothing but $x_q(n)$ minus that is quantized value minus the $x(n)$ the original value of it, so, for all n in my real world basically.

(Refer Slide Time: 19:29)



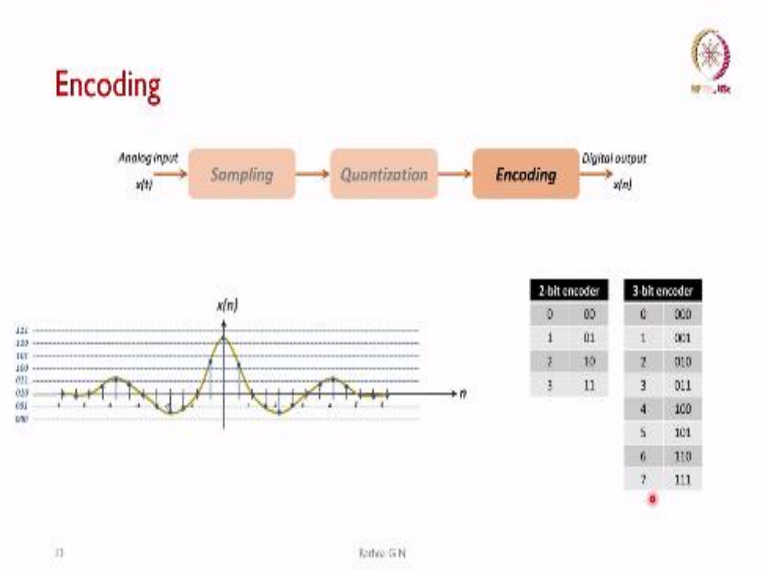
So, coming to next one what is the thing is going to happen so, we took it as an example as 2 bit quantizer and then we saw the thing. So, for uniform quantization, the resolution is going to be determined by dividing the full scale range by the total number of quantization levels. That is if I am representing with 8 bit number, then the resolution for each value is going to be represented as 2^8. So, B can be any number of bits what it is shown here.

And then I have to specify a signal to quantization noise ratio for B bit quantizer is approximately what we call it as 6B, B is the number of bits dB is represented in decimal. So, as it is shown in the table here, number of bits, if I use 8, B=8, then the levels what I can represent is 2^8, which is a 256 levels what I can represent the values and then the resolution is going to be as we have seen now that full scale range, if we take it as 5 volts, then I will be getting each sample is represented as 19.5 millivolts level.

And then in that case, what I am going to achieve for 6×8 is going to be my signal to quantization noise ratio, which is going to be 48 dB. So, if I increase the number of bits to 12 bits, what I am going to look at it, then I will be representing the number of levels is 4096. And then, which is 2^{12}. And then the resolution as we will be seeing that it will be 1.22 millivolts. So, compared to this, I have a more resolution between the bits and my SQNR that is quantization noise ratio, what signal to quantization as ratio is going to increase to 72 dB.

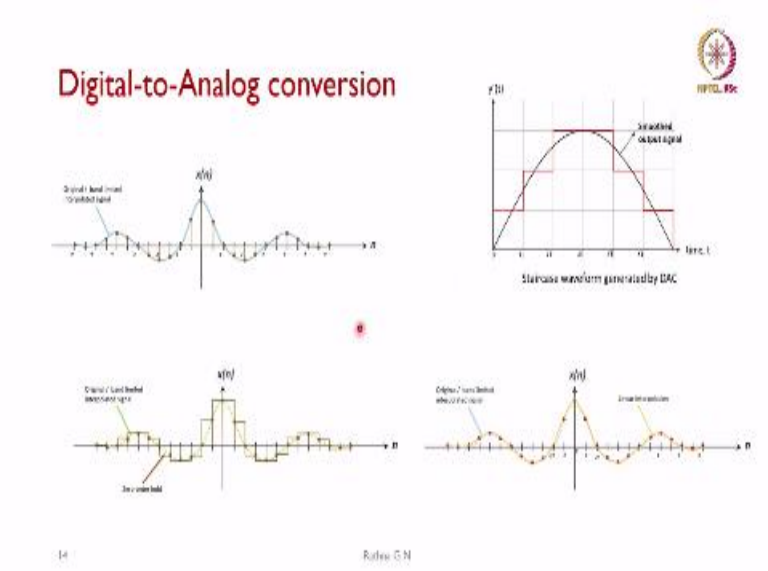
So, for the 16 bit conversion or quantization, so I will be achieving somewhere around 96 dB. So, all of us know that CD quality although it is extinguished now today, so which had a 91 dB of resolution what that is signal to quantization noise ratio as the standard being used for it. So, you will be seeing that somewhere between these 2 what you are supposed to have number of bits, but we know that it depends on ADC what we are going to use it so which we will be discussing it in a while.

(Refer Slide Time: 22:16)



So, now once we have done quantization, then we have to represent this quantization bits, according to the number of bits what we have been using it. So, then we have to do the encoding of the number or the signal. So, this is 2 bit encoder, we have seen it already. And then when I talk about 3 bit, then I will be representing the values between 0 to 7 this is integer, what I will be representing it.

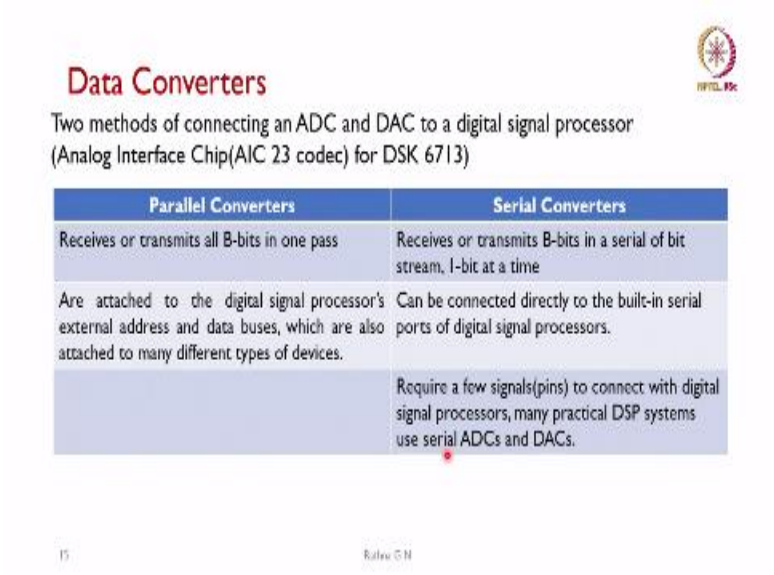
(Refer Slide Time: 22:54)



So, when I take up the number system, we will discuss about what I will call it as negative and then positive numbers, how we can represent it. So, once we have done the analog to digital conversion, then we want to do digital to analog conversion. So, we have these are the digital values what I have it, $y(n)$ is the thing, and then I will be seeing that how I can represent these in the continuous domains. So, this will be the digital output what I have it.

So, I will be doing the output, that is smooth output signal, what I want to represent in this digital output, so how to do the conversion. So, for that as we will be seeing that we will be using the this as the original or this thing, signal band limited signal what we had it so we will be using the zero order hold basically sync filter what will be presenting it, and then we will try to reconstruct the signal in this way so this is our linear interpolation what I can do it and then reconstruct the signal in this fashion. So, this is how we will be getting the analog signal output.

(Refer Slide Time: 24:14)



So, coming to data converters, so we need ADCs and DACs for both input analog input conversion to digital domain and from digital domain to analog. So, in this we will be using the DSK 6713 board basically, as I mentioned in the previous class, that most of the colleges have this so I will be taking this as the basic one. And then we will be going with the laboratory incorporating this board either in the simulator mode initially then we can use the hardware to test the real time signals.

So, in this case what we have call it as AIC chip which is analog interface chip in the DSK 6713 is going to be AIC 23 codec, what we call it both coding and then decoding is incorporated in one of the chip itself. So, the higher versions of a DSP processor from TI what I will be taking it, so we will see why we have gone with the TI later on. The thing is going to be 3 2 version AIC 32 versions what they will be using in the higher versions of the boards basically.

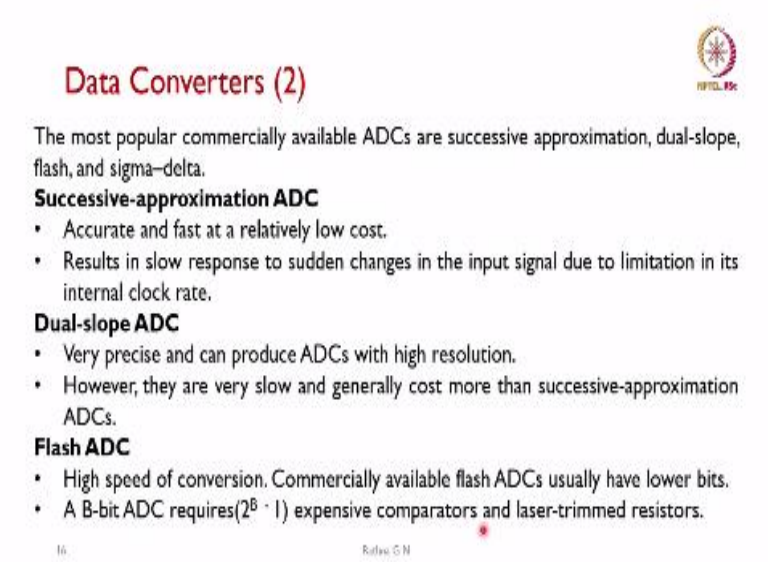
So, now when we want to do the data conversion, either I can have a parallel converters or serial converters, so we will see the advantage of a disadvantage of both parallel and serial converters. So, parallel converters, we know that it receives or transmit all B bits in one pass. So, whatever data I have considered, basically, then it will be converting or taking all the 8 bits at a time if it is 8 bit converter, and then it will be outputting all 8 bits output.

So, whereas in the serial case, it receives or transmits this B bits in a serial of bit stream, that is 1 bit at a time. So, what is the consequence of it, as we know that, whatever I can get, if I am talking about 8 bits of taken the thing, B = 8 I have considered. So, all the 8 values will be coming into my system at 1 clock cycle. Whereas when I am using the serial converter, I know that 8 bits will take 8 clock cycles because, I am taking 1 bit at a time, so I have to wait for 8 clock cycles to get complete data.

But what are the disadvantages of the parallel converters we will see. So, these are attached to the digital signal processor basically, external address and databases directly it will be connected to that. So, which are also attached to many different types of devices in this case. So that means to say that so many bits are required or what I will call it as IO pins to get the input data and any other devices also using it. So, so many IO pins has to be there.

Whereas in the case of serial converters, so we will be using built in serial ports of digital signal processors, and then we know that it requires few signals or pins to connect with our digital signal processor. So, most of the many practical DSP systems use serial ADCs and then DACs.

(Refer Slide Time: 28:04)



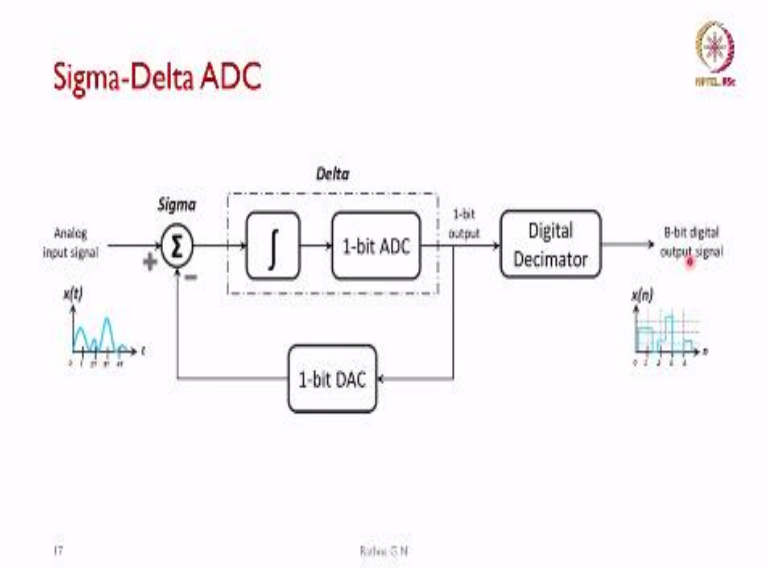
And then coming to the thing, what are the kinds of data converters available in the market. So, we know that some of them are successive approximation, dual slope, flash ADCs are there and sigma delta converters are there. So, we will see advantage and disadvantage of them few of them,

the first one is a successive approximation ADC. So, we know that it is very accurate and then olden days we were using this type of conversion and we know that it is fast at a relatively low cost, so very popular in the olden days.

So, this results in slow response to sudden changes in the input signal due to limitation in its internal clock rate. So, its internal clock rate is less. So, any changes in the thing so, the response is going to be slow, this is the disadvantage of our successive approximation ADC. Coming to dual slope ADC so, this is very precise and can produce ADCs with high resolution. What is the disadvantage they are very slow and generally cost more than successive approximation.

So, where the cost is not going to matter, then people are going to use the dual slope ADCs. The other one type is the flash ADC. So, we know that it is very high speed conversion and commercially available flash ADCs usually have lower bits because of the power consumption and then speed and other things coming into picture. So, they cannot have higher bits. So, an B bit ADC we know that it requires $2^B - 1$. So, bits basically so which is going to be expensive comparators and laser trimmed resistors required to implement this kind of ADCs in the flash mode.

(Refer Slide Time: 30:06)



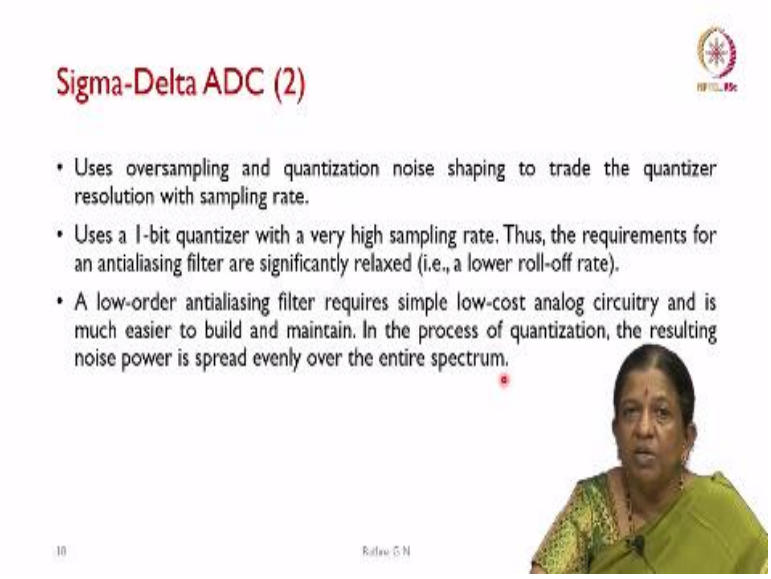
So, the other advantage of using the sigma delta, which we will be looking at in a while, so what the input signal we have the analog input signal. So, we will be feeding into the sigma delta, what

first is sigma basically, then you will be seeing that delta part of it is going to come which is 1 bit ADC, we will be taking it and then I will be getting 1 bit output and then which is going to be we call it as digital decimator.

So, why we need the digital Decimator is we will be sampling at a very high frequency here input signal and then the output has to be brought down to whatever the board is going to support. So, which will be a converting get into 8 bit digital output signal in this case, so when I want to reconstruct because this is what we call it as sigma delta, delta sigma basically, so when converting it from ADC mode, we will be having sigma delta when I want to convert it back to DAC that is my analog output.

So, I will be taking the delta output and then this is 1 bit DAC, I am going to have it which is going to provide it to sigma and then I will be getting the analog output from it. So, you will be seeing that this is a digital input, how 8 bit representation of the digital signal is shown in this figure.

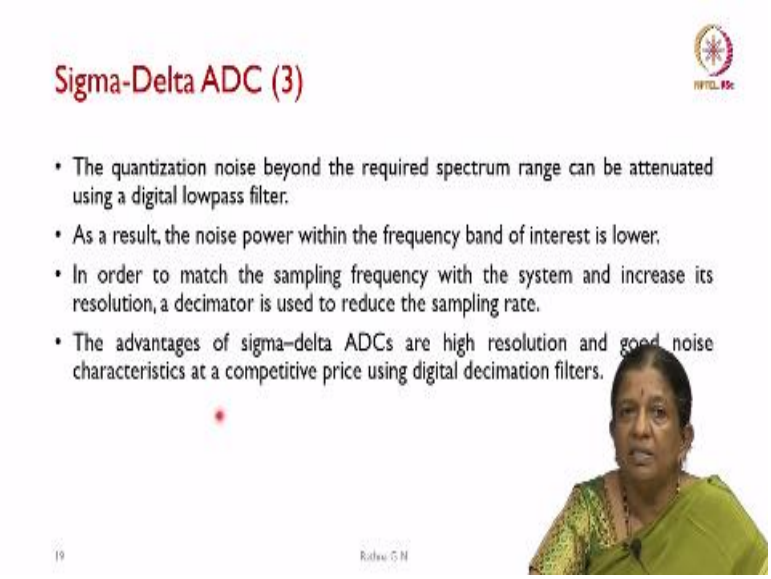
(Refer Slide Time: 31:39)



So, the coming to the, its usage so what it does is we use the over sampling and quantization noise shaping to trade the quantizer resolution with sampling rate. So, we are going to sample with high sampling rate, then what happens to this then we use 1 bit quantizer with a very high sampling rate. Thus, the requirements for an anti aliasing filter are significantly relaxed. So, we know that anti aliasing filter has to be designed in the analog domain.

So, we know the disadvantage of filter construction, there in analog domain. So that means to say that we are going to provide a lower roll off rate for the anti aliasing filter. So, we say that low order anti aliasing filter requires simple low cost analog circuit compared to a complicated one if we want to have a fast roll off and is much easier to build and then maintain them for longer period. So, in the process of quantization, what happens the resulting noise power is going to be spread evenly over the entire spectrum. Because we have sampled it at high speed, so the noise is going to be distributed over the whole spectrum.

(Refer Slide Time: 33:06)



So that is what it says the quantization noise beyond the required spectrum range can be attenuated using a digital low pass filter. So, we are relaxing on the analog low pass filter. And then we can use the digital low pass filter to remove the noise from the component that is present in this input signal. So, as a result, the noise power within the frequency band of interest is going to be low because it is distributed over the entire spectrum.

So, to match the sampling frequency with the system and increase its resolution, we use the decimator to reduce the sampling rate. So, the advantages of sigma delta ADCs are high resolution and good noise characteristics at a competitive price using digital decimation filters. So, the thing is happening in the digital domain. So, the cost of these ADCs are very less. So, that is how most of the DSP processors use the sigma delta ADCs in their hardware.

(Refer Slide Time: 34:13)

Now, once we have discussed about the ADC, so, we look into the DSP hardware. So, we can have a different as you will be seeing that 5 varieties of them are there, one is a special purpose that is usually we custom build the chips basically, such as application specific integrated circuit, we call that as ASICs, basically. So, one of the advantage is that, if you are going for a very huge volume, the cost of the units is going to be very less, we will be seeing in a while how there will be comparing.

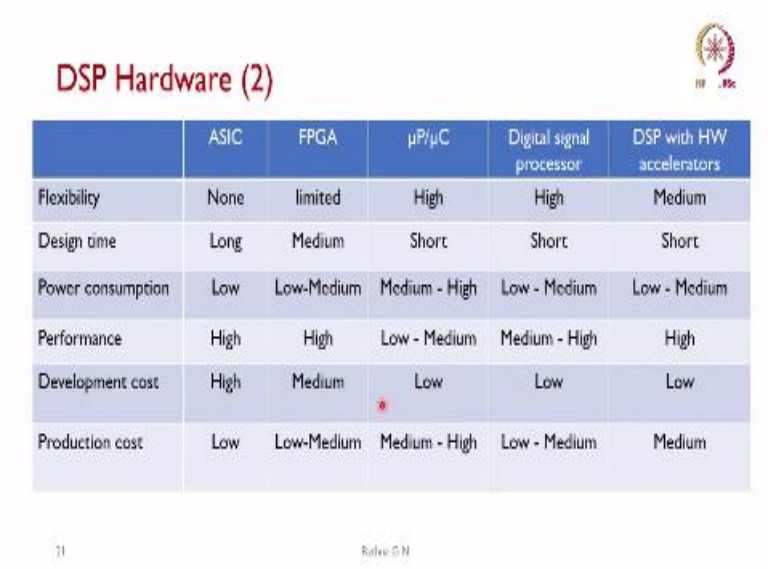
Then it is worthwhile going for the special purpose processors, but once we have built this to change any of the in between components or any circuitry then we have to redesign it, it is not possible to modify it, that is one disadvantage of using ASICs. The other one is field programmable gate arrays, we call it as FPGAs there it is before ASIC what we can use them. So, that is in the field, I can reprogram all the gates are present in the thing as an array, so we will be programming them.

And then I can later on once we feel that my circuit is capable of working very well. And then I want high productivity then I can go further ASICs, so it is easier to try it in this. So, compared to this, we will be seeing that in terms of the design time and other things, which will the first one will have the maximum time to build around and then this is little much easier and coming to some

of the levels where in classrooms and then some of the research we can use general purpose microprocessors or microcontrollers for specific applications.

And then we can go if the application is IO oriented, it is better to go with microcontrollers or if general purpose processor number of applications are much more what we have to test it, but when it comes to signal processing, so it is better to go for general purpose digital signal processors. So, we will see the advantage of them later. The other one is we can have DSP with application specific hardware that is we call it as hardware accelerators. So, with respect to general purpose process, we can have a hardware accelerators particularly meant for DSP applications also is a possible hardware one can consider.

(Refer Slide Time: 37:13)



So, this table gives you what are the advantages and then disadvantages in terms of that is from the flexibility, from the design time, power consumption, and then performance and then what will be the development cost and then the production cost. So, you will be seeing that ASIC is, I do not have any flexibility, as I have mentioned in the previous slide, that once you have designed it any change you have to incorporate you have to redesign so, I would not have any flexibility in change of design.

And then in the case of design time, you will be seeing that it is going to take a longer time and power consumption because according to my requirement application, I have designed it which is

going to consume low power because I will be using only few components what is much required for it and coming to the performance which is going to give very high performance because it is application specific what we have designed it. Coming to the development cost is going to be very high as time is going to be longer.

So, one who are very good in designing it they have to be paid heavily. That is why it is going to cost us very high development costs. Coming to the production cost because we think of high volume basically you can see your mobiles basically. So, as the customer increases, you will be seeing that cost is going to lower as the trend is if it is older versions so we will be getting it for low cost, but whenever it is newer you will see that there whatever the your design time is going to be much higher, but production cost will be kept as low as possible.

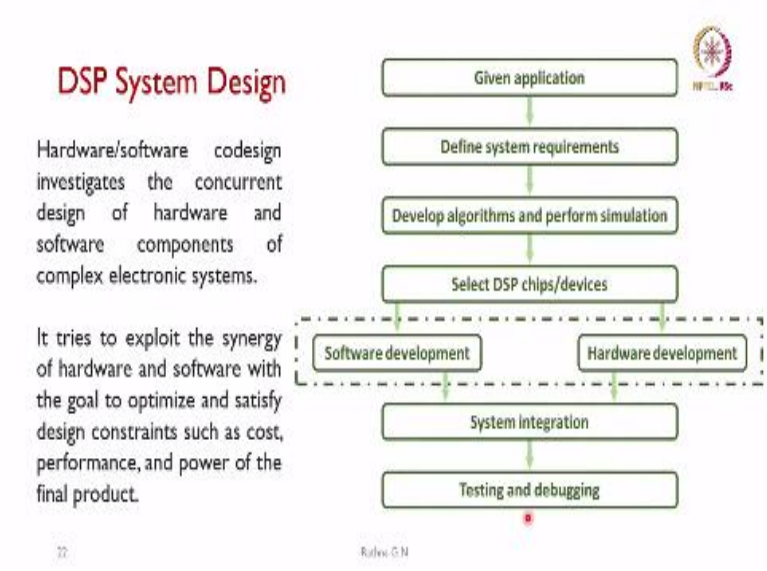
Coming to FPGA field programmable gate arrays so, we have limited flexibility in this compared to ASIC we have some of the flexibility and then our design time is medium and then the power consumption is going to be low medium in this case and performance is going to be high as equivalent to ASIC and development cost is going to be medium in this and production cost, it will be in the low medium case.

Coming to your microprocessor and microcontrollers as we will be seeing that our flexibility in digital signal processor and microprocessor and microcontroller are very high and then design time in both the cases is short whereas, in the power consumption here it can be medium to high, but in the digital signal processor we can have a low power consumption processors are available or I can have some of the medium processors, which I can use it for my applications.

So, when it comes to performance, there will be low to medium what it is going to be, but whereas, in the case of digital signal processor it is from medium to high performance what I will be getting it. Coming to the development costs both stand at low and then production cost is going to be this one is medium to high whereas, digital signal processor low to medium. Coming to with hardware accelerators so, the flexibility is going to be medium because my accelerator based on my requirement what I have designed it or what I am using it.

So, that way I will not have much flexibility, just like our FPGA we may have the medium flexibility, whereas the design time in this going to be short and then I can have the power consumption between low to medium what I can select and then performance is going to be very high and then development cost is going to be low in this and production cost is going to be medium. So, if I had the hardware and I want to have increase the speed of my DSP computations, then it is better to go with accelerators. So, these are the table what it decides what kind of hardware one has to select for their basic applications.

(Refer Slide Time: 41:45)



Coming to the design, how we are going to proceed. So, signal process system design, what we call it. So that is if you have been given an application. So, I will be defining my system requirements and develop algorithms and then perform simulation first to see that my algorithms are working fine and then my requirements are correct, then I can select DSP chips or devices or whether I have to go for the full custom or FPGA based or hardware accelerator, I can decide it here.

Then what I will do is I will be deciding on initially earlier cases was software development was separate and then hardware development was separate as you will I was pointing it out in the last class hardware cost was very less, software cost is going to be very high because they were designing it independently. So, hardware a lot of functions are built into our chips and then which

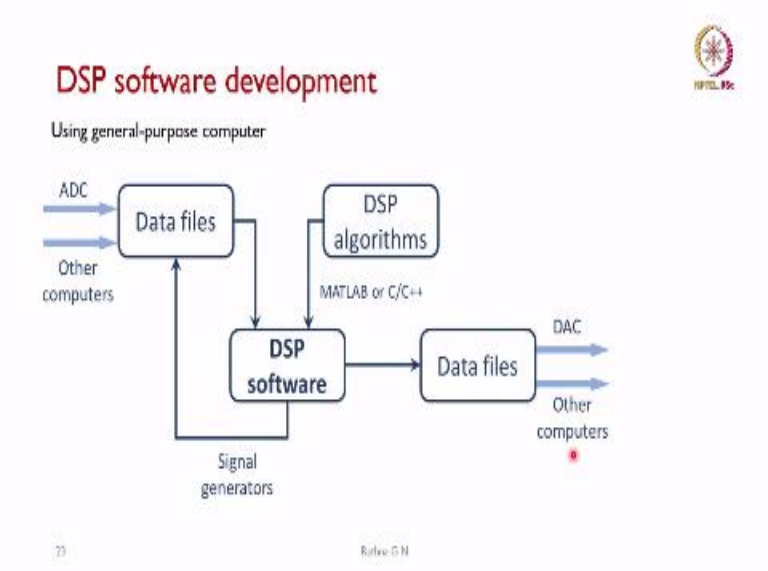
is going to come out to make them use software has to be developed in such a way that all the units are used which is going to cost us very high.

So, but now the trend is as we will be calling it as hardware software codesign. So, they have to go in hand in hand that is why it is put in dotted line. Now, recent trends they have to go in concurrent design of hardware and software components of complex electronics systems basically. So that will be in line with whatever hardware is getting added software also will be able to use that hardware.

So that is what, what it says it tries to exploit the synergy of hardware and software with the goal to optimize unsatisfied design constraints such as cost, performance and power of the final product. So, once this has been done, so we will be doing the integration system integration has to happen, then we know that testing and debugging is one of the most critical one. So, how one is going to do testing and debugging, it depends on what application we are going to use for, and then how much we have to do it.

Whether for real time systems or that is what I am talking we are talking about real time signal processing, it should not fail, then we should have taken care of testing and debugging for all possible cases one has to do it.

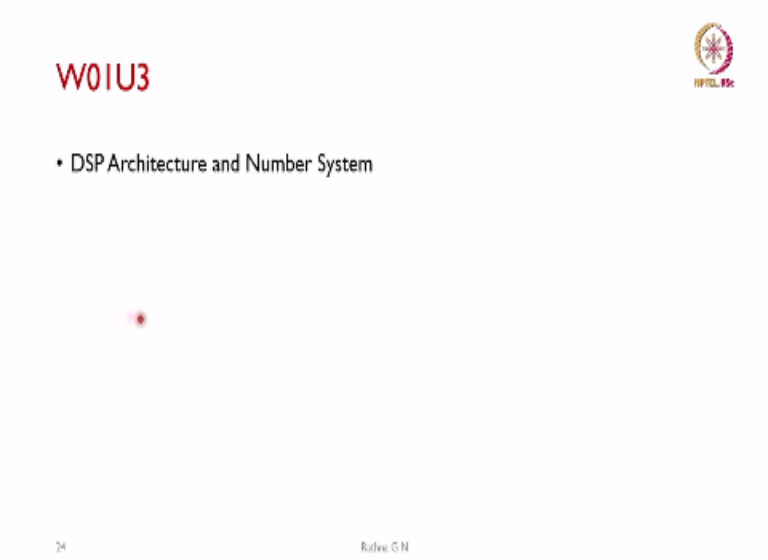
(Refer Slide Time: 44:29)



So, in the case of software development, so we will be using mostly general purpose computer. In that case, we will be externally we have to provide analog to digital converted signal into the system, which will be doing it storing it in the data files, or we can take it from other computers, if we have the converted data. So, we will be storing it as data files. And then we can run the thing DSP algorithms either using C or C++ or MATLAB and then those are the software what we can use it DSP software.

And then we can generate some of the signals and then we if we want, we can store it back in the data files, and then the output of it applying the algorithm we can store it in the data files, and then externally we can use the digital to analog converter to convert it and then send it out or if it has to be given to other computers, which has to process it, we can give it to other computers.

(Refer Slide Time: 45:36)



So, this is the end of the second lecture. So, in the next lecture, we will be taking DSP architecture and then number system.