Digital Signal Processing

Prof: S. C. Dutta Roy

Department of Electrical Engineering

Indian Institute of Technology, Delhi

Lecture - 3

Digital Signals (Contd…..) Digital Systems

This is the 3rd lecture on Digital Signal Processing and today we continue our discussion on digital signals and also introduce digital systems. In the previous lecture, yesterday, we had discussed why we require Digital Signal Processing even if the signal is analog, and what the processing of an analog signal by DSP actually involves. It involves sampling, hold, A to D, DSP, D to A and finally an analog low pass filter.

We had enumerated the major advantages of DSP and mentioned a dominant advantage, namely realization of linear phase. For communication engineers it is a boon because linear phase implies that there is no delay distortion in processing a signal. We also said that the same digital signal processor can be used for handling more than one signal by a process known as time multiplexing. DSP is less sensitive to element tolerances and environmental changes. The accuracy and range of DSP (dynamic range) can be increased almost without limit if you are prepared to spend more money. Storage of digital signals is no problem. It can be done on magnetic tapes, disks and other media.

Low frequency processing is no problem because there is no inductance to bother about. The characteristics of DSP can be changed very conveniently by changing one or more coefficients, that is, simply changing some numbers. But then we also said that DSP is not all advantageous; it has its own demerits. First is that the hardware complexity is much more than in analog signal processing. Then there is a limitation on the highest frequency that can be used because the sampling frequency is limited for sample hold and A to D conversion, the current status being about 10 MHz. You cannot go beyond that. So the signals that you can handle can go only up to 5 MHz, that is half of the maximum sampling frequency. Power dissipation is a problem in DSP but I have told you that research is on to reduce the power as much as possible and ultimately to

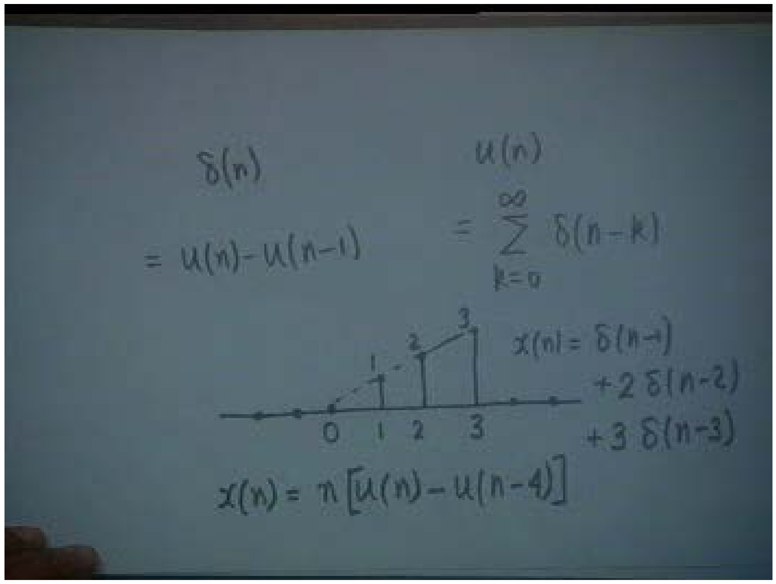
0, particularly for implantable devices; the aim is that the device should work from body temperature.

We introduced sequences like right sided, left sided, general sequences, even sequence and odd sequence. We said that any sequence can be broken up into its even part and odd part. Then we talked about bounded sequence, absolutely summable sequence, and finite energy or square summable sequence. We also defined a quantity called average power. For a periodic signal, average power needs to be calculated over one period only, because every period is a repetition of another period.

Then we introduced two elementary digital signals, namely delta (n), the unit impulse, and u(n), the unit step delta(n) is the most elementary signal and it differs from delta (t) of analog signals in that it is much simpler; its amplitude is simply 1, and it occurs at n = 0. u(n) can be represented as a combination of delta (n – k) where k = 0 to ∞. And delta (n) similarly can be represented as u(n) – u(n – 1). This is the relationship between delta (n) and u(n). Any arbitrary sequence can be expressed in terms of u(n) and delta (n).

Let us take an example: Suppose you have a sequence which starts from 0 at n = 0 and then 1 at n = 1, 2 at n = 2 and 3 at n = 3. Obviously we can write this as follows. Suppose we call this signal as x(n). What is the length of this signal? It is 4 because the sample at 0 also has to be taken into account. We can obviously write x(n) as delta (n – 1) + 2 delta (n – 2) + 3 delta (n – 3), in terms of deltas. x (n) can also be expressed in terms of u(n) – u(n – 4). What is this? This is a gate from 0 to 3 because u(n – 4) is subtracted from u(n). We see that the amplitude is proportional to n so you simply multiply by n. Thus x(n) = n[u(n) – u(n – 4)]. There are a large number of problems in the text book and you should work them out. Any arbitrary signal can be expressed in terms of u(n) and delta (n).

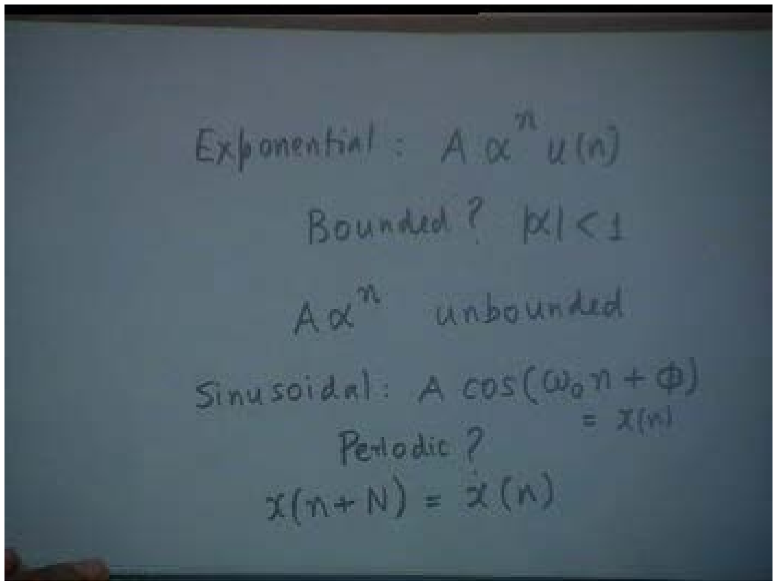
(Refer Slide Time: 05:18 - 08:25)



One sequence that is very often used is called the exponential sequence. Exponential sequence is of the form, $A\alpha^n$. This sequence may start from n = 0 and go as A, Aα, $A\alpha^2$, $A\alpha^3$ and so on. The way to ensure starting at n = 0 is to multiply $A\alpha^n$ by u(n). u(n) is 0 for n less than 0 and it is 1 for n greater than or equal to 0. Therefore $A\alpha^n u(n)$ is a right sided signal starting from n = 0. Is this a bounded sequence? Obviously not, if |α|>|. If |α|<1 then it shall be a bounded sequence. α can be complex, real, positive or negative. But if magnitude of α is greater than 1, then this right sided signal is not bounded because it increases indefinitely to ∞. If it is not right sided, if you take the general signal α^n, then is this bounded? Are there some conditions which will make it bounded sequence? No, whether |α|>1 or |α|<1. If |α| magnitude is less than 1, then the right side is bounded but the left side will become unbounded. So this signal in general is unbounded, irrespective of the magnitude of α.

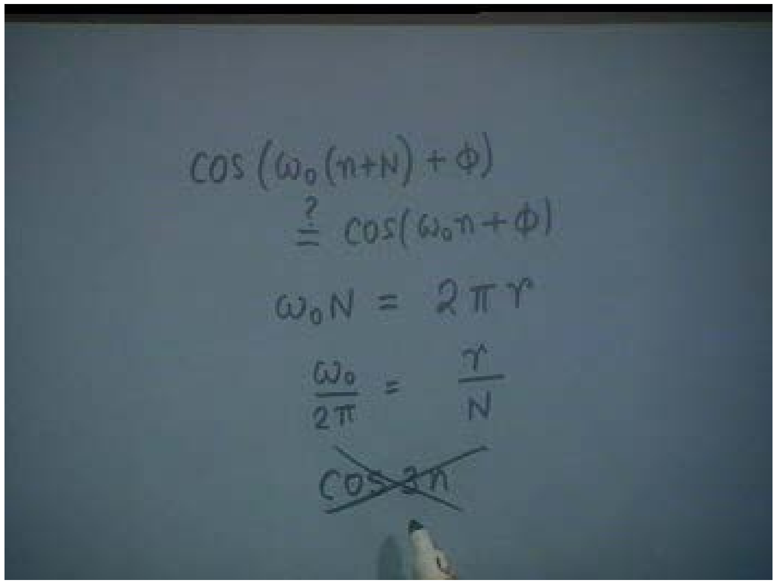
Then we considered the most important signal in practice, and information is transmitted only by a combination of them, that is the sinusoidal signal. The most general sinusoidal signal is A cosine ($\omega_0 n + \Phi$). Is sinusoidal signal a bounded signal? Yes it is a bounded signal because the maximum value is A. Is it a periodic signal? The answer is not an unconditional yes. It may be periodic or it may be non-periodic. If it is to be periodic and if we denote this by x (n), then there must exist a quantity N such that it is equal to x(n + N).

(Refer Slide Time: 08:42 - 13:06)



Let us see why. Forget about A. A does not affect the periodicity or otherwise. Suppose you take cos (ω_0 (n + N) + Φ); if this is to be equal to cos (ω_0 n + Φ) then obviously $\omega_0 N$ must be equal to an integral multiple of 2π. So, it must be 2πr where r is an integer. In other words $\omega_0/(2\pi) = r/N$. Now this is not guaranteed; $\omega_0/(2\pi)$ may be a rational number that is an integer divided by an integer or it may not be. For example cos (3n) is not a periodic function because in 3/(2π) is not a rational number. Therefore cos (3n) is not a periodic signal. In other words, while all analog sinusoidal signals are periodic, a digital sinusoidal signal may or may not be periodic. Is that point clear?

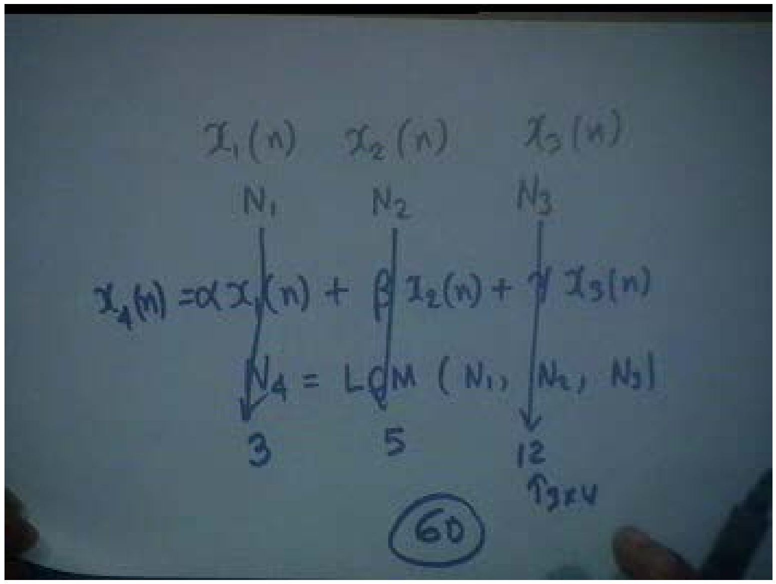
(Refer Slide Time: 13:12 - 14:35)



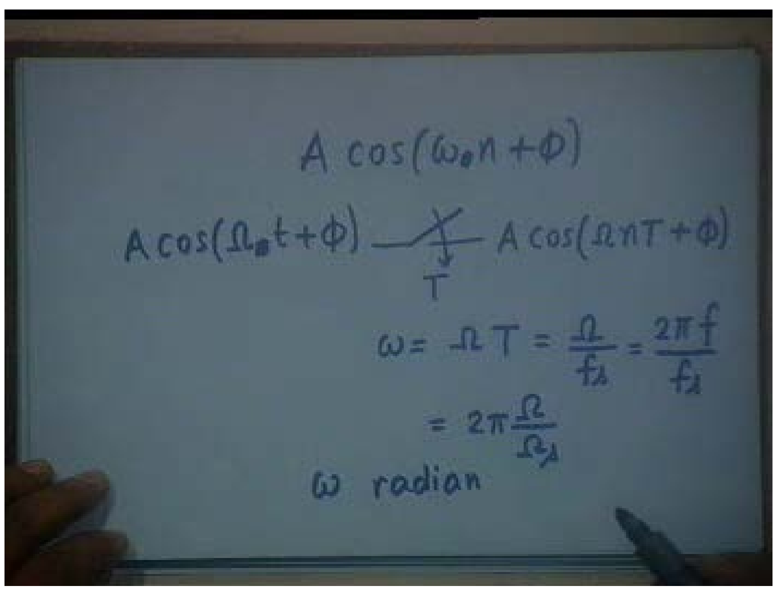
Suppose, we have three signals $x_1(n)$, $x_2(n)$ and $x_3(n)$. Suppose each of them is periodic, of periods N_1, N_2 and N_3, respectively. Then let's make a combination of these three signals. That is multiply x_1 by α, x_2 by β and x_3 by gamma. Let us call $x_4(n)$ as the signal obtained by superposition of these three signals. The question I ask is, is $x_4(n)$ periodic? It is surely periodic, but the period N_4 the least common multiple (LCM) of (N_1, N_2, N_3). Let us take an example. Suppose N_1 is 3, N_2 is 5 and N_3 is 12. The LCM of 3, 5 and 12 is 60. After 60 cycles x_1 executes 20 complete cycles, x_2 executes 12 complete cycles and x_3 executes 5 complete cycles. At the end of 60 samples of $x_4(n)$, each of the components have completed an integral number of cycles.

Therefore the combination is periodic with the period of 60. In general, the sum of digital periodic signals is also periodic. If the periods of component signals are prime numbers, then the new signal has a period equal to the product of the individual periods. For example; if one is 3 and the other is 5, it is simply 3 times 5, equal to 15. But if they are not primes with respect to each other then we have to take the LCM of the component periods to get the period of the resulting signal.

(Refer Slide Time: 14:54 - 17:32)

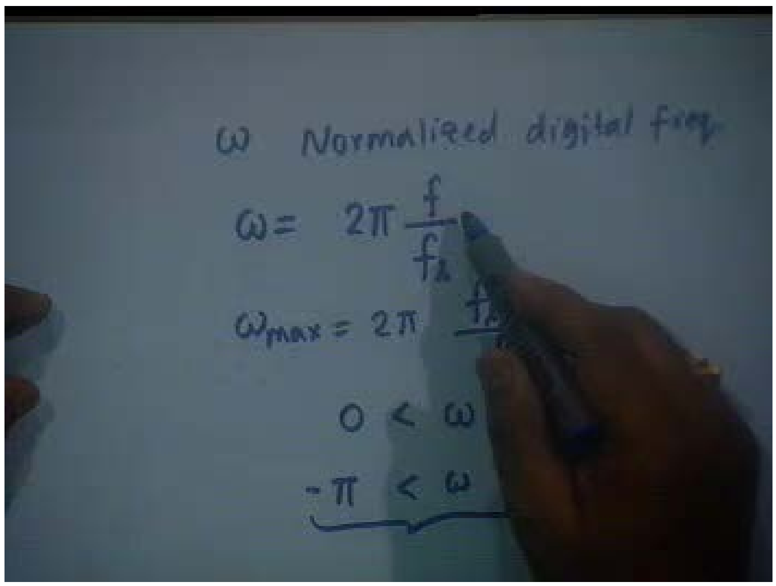


(Refer Slide Time: 17:46 - 20:26)



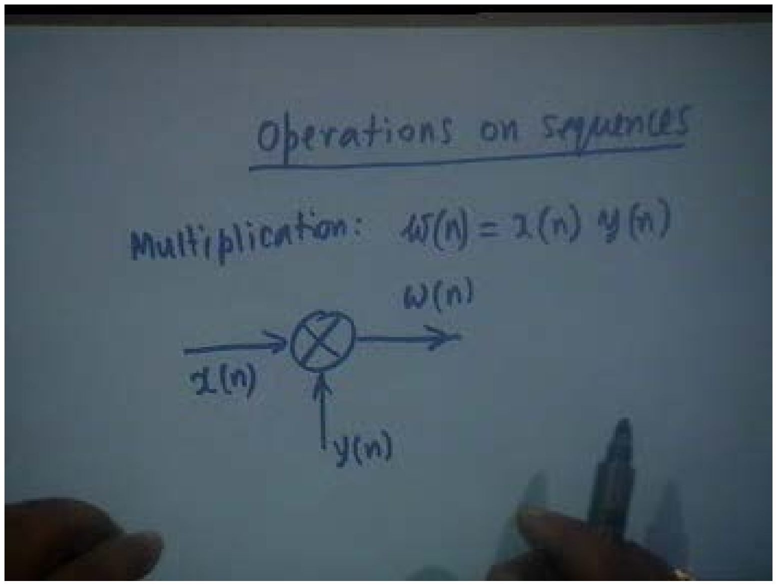
Now we go a little more into the interpretation of this signal A cos (ω_0n + Φ). How does one obtain such a signal? Suppose you have an analog signal A cos (Ωt + Φ) and sample it at every T seconds; then what you get is A cos (Ω nT + Φ). After A to D conversion, you get A cos (ω n + Φ). What is ω? ω is Ω multiplied by T which I can write as Ω/f_s, f_s being the sampling frequency that is equal to 1/T. The actual frequency f is in Hz, Ω is in radians per second, i.e. $\Omega = 2\pi f$, 2f. Therefore you can write ω as 2πf/fs. You notice that Ω was in radians per second but ω has lost its unit because f/f_s is dimensionless. ω is a dimensionless number and we express this in radian. Does radian have a dimension? No, an angle is arc divided by radius and therefore ω the digital frequency is in radian, not radians per second.

(Refer Slide Time: 20:28 - 22:58)



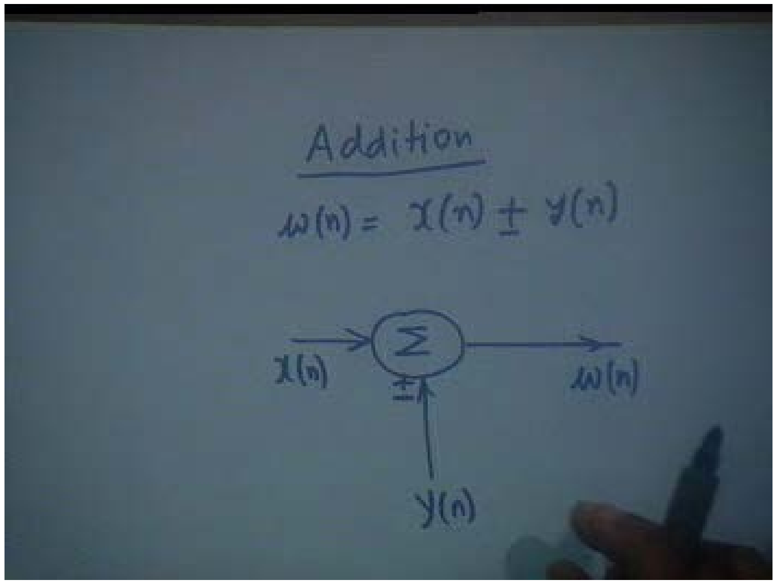
ω is given a name normalized digital frequency. Recall that ω is $2\pi f/f_s$. We shall soon see that in Digital Signal Processing, we cannot go beyond half the sampling frequency. If we do, there is distortion and therefore we have to limit to $\omega_{max} = 2\pi\ ((f_s/2)/f_s) = \pi$. Therefore the range of normalized frequency is 0 to π. For a real signal, for every positive frequency, there must exist a negative frequency also. Therefore the actual range of ω is – π to + π. This is called the base band or the band of vision. It is the band where you have to focus your attention in DSP. This is an advantage because our range of vision is limited; we don't have to go from – ∞ to + ∞ as in analog signals. If we want to draw magnitude or phase response, it suffices to consider only positive frequency because magnitude is an even function and phase is an odd function of frequency. The compression of the frequency from – π to + π makes life simple; you don't have to compute beyond π. But you must remember that ω is the normalized frequency. It is not the frequency in radians per second, it is in radians.

(Refer Slide Time: 23:07 - 24:57)



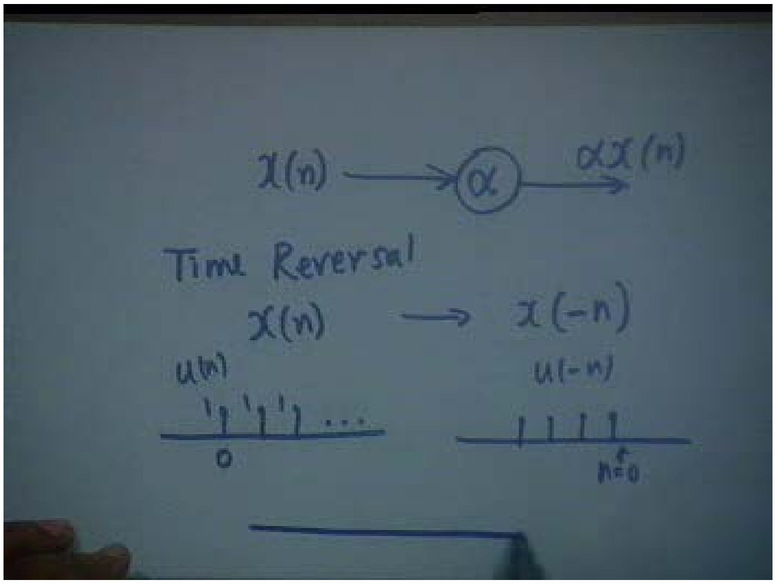
Now a digital signal or sequence can be subjected to some operations. First is multiplication; if a sequence x (n) is multiplied by another sequence y(n) to form a new sequence w(n), then w(i) = x(i) y(i) for all i. The symbol used for multiplication which is nothing but repeated addition is a cross within a circle. Multiplication is a non-linear operation and there are very few instances where we resort to a nonlinear operation like multiplication. However, as we shall see later in FIR filtering, we do use multiplication of two sequences (we multiply the impulse response sequence by a window function). One more point before we go to next operation. Suppose the two sequences are not of the same length. Suppose x(n) is length 3 and y(n) is of length 7. Then how do you multiply? Before doing any operation you make two lengths identical by adding 0's to the sequence which has less number of samples. This must be remembered.

(Refer Slide Time: 25:47 - 27:46)



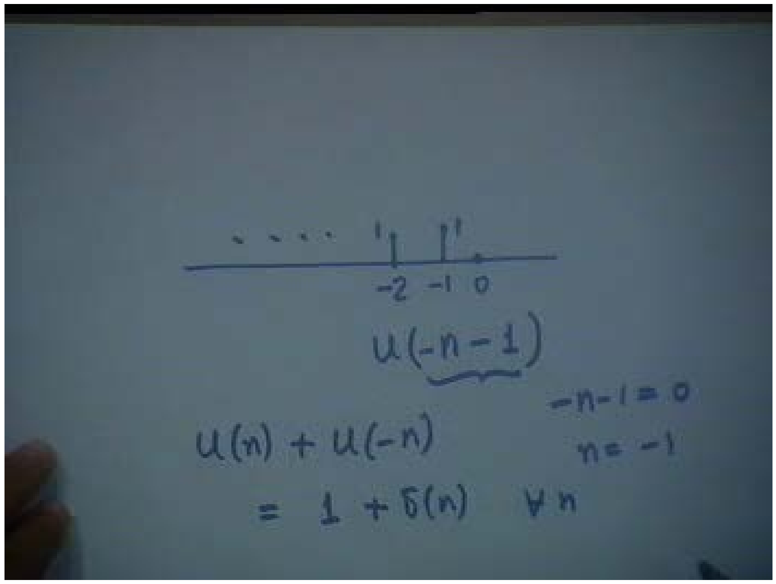
The next operation is addition or subtraction. When you add or subtract y(n) from x(n), then the corresponding samples of y(n) are added or subtracted to from x(n) to form the new sequence w(n). The symbol that we shall use is the summation sign within a circle and two inputs x(n) and y(n). If it is simply addition then we do not assign any sign to the arrows. But if it is subtraction of y(n) from x(n) then we assign a negative sign to y(n). So in general it is ± sign but plus sign we usually avoid. Negative sign simply means that this sequence is to be multiplied by – 1; – 1 is not a multiplier but it is simply changing the sign bit of this binary number. So, we do indicate a minus sign only when subtraction is involved but if addition is involved we do not do that. One more point that needs your attention in Digital Signal Processing is that we cannot add more than two numbers at a time. In an accumulator, we fill numbers in sequence and they accumulate. The new number is added to the stored number and the result is stored; then it takes another number, and so on. As a result, whenever we have to add three numbers, for example, we shall use two summers to indicate the steps that are involved.

(Refer Slide Time: 27:57 - 29:33)



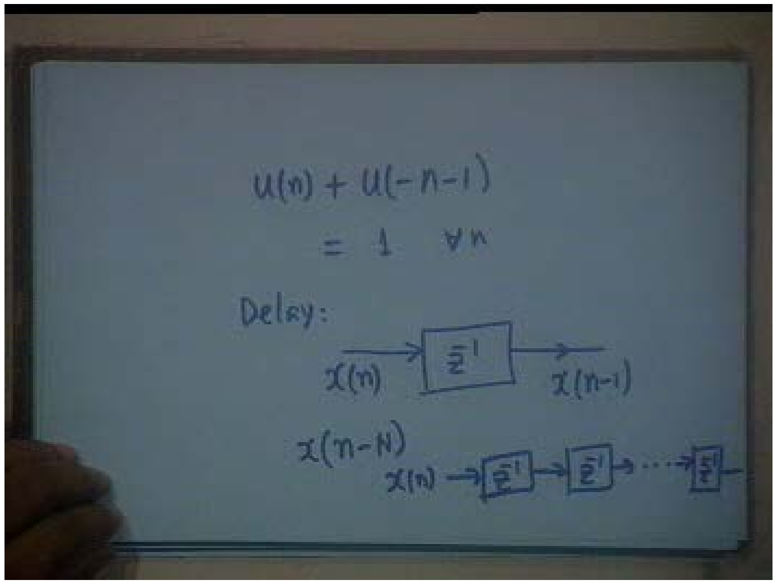
Multiplication can be of two signals or it can be by a scalar. For example, if you have x(n) and if you wish to multiply this by α, then we shall simply use α inside the circle and the output is α x(n); this is a scalar multiplication. It is not a multiplication of two sequences. If it is a multiplication of two sequences then we use a cross, as mentioned earlier. The operation of time reversal means that for a given x(n), the time index is reversed. That is you find an x(– n), which means simply that you flip x(n) with n = 0 as the fixed point. That is if it is a right sided signal then you flip over and make it a left sided one; if x(n) is a left sided signal, flip over and make it a right sided signal. For example as you know u(n) is 1 for n greater than or equal to 0. What is u(– n)? It starts at n = 0 because – 0 is same as n = 0. Then it goes to the left.

(Refer Slide Time: 29:34 - 31:11)



Suppose I have a signal which does not start at 0 but it starts at – 1 or – 2 and it is of amplitude 1 on the left hand side for all n after the starting value; what do you call this signal? Can it be u(– n – 1)? Where does this start? Obviously when (– n – 1) = 0, i.e. n = – 1. So this signal is u(– n – 1). u(– n – 2) will start at n = – 2. Suppose I take the sum u(n) + u(– n). What is this signal? It is 1 everywhere except at n = 0, where the amplitude would be 2. Therefore, we can write this as 1 + delta (n) for all n.

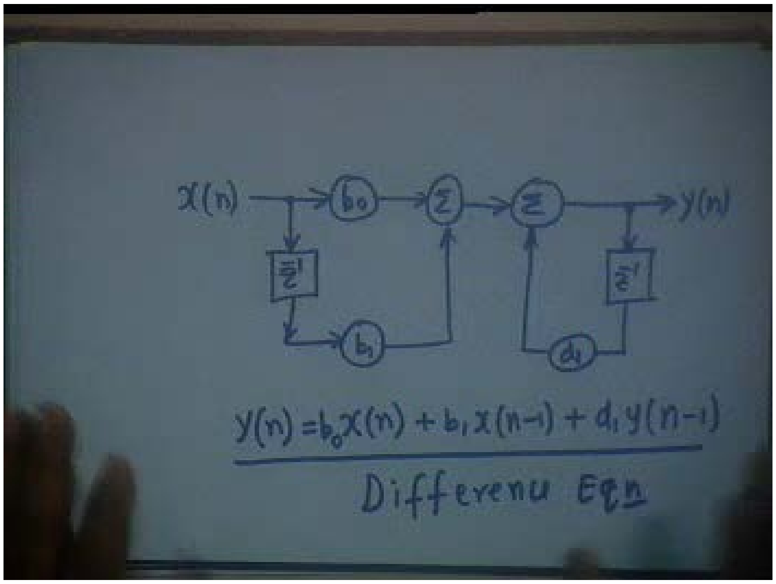
(Refer Slide Time: 31:13 - 33:13)



If I add u(n) to u(– n – 1), then this would be equal to 1 at all n. So u(– n – 1) has a special significance, we shall come back to this later. And then finally we have delay. We represent delay by a special symbol z^{-1}; we shall discuss the significance of z later. It simply means that if x(n) is the input, then the output is x(n – 1). In terms of hardware, it is a one sample delay and in terms of software, it means retrieving the immediate past sample. The immediate past sample is stored somewhere. If you require x(n – 1) you retrieve it on bring it back. This process is symbolically represented by a block whose transfer function is z^{-1}. For example, if you require x(n – N) then you have to feed x(n) to a chain of N such delays. z^{-N} would be represented by a chain of N z^{-1} blocks. This is also a discipline that we shall follow.

A Digital system handles digital signals and operates on digital signals to produce another digital signal, which in some ways is better than what you had at the beginning. Let us take a very simple example of a digital system.

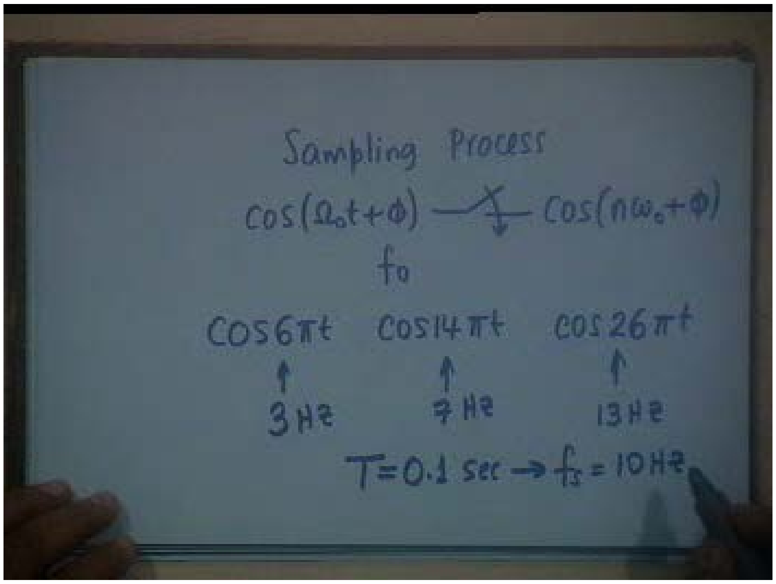
(Refer Slide Time: 33:53 - 36:50)



Suppose, you have an x(n) which is scalar multiplied by b_0 and added to a delayed version of x(n). Look at the diagram. The input to the z^{-1} block is x(n) and the output is x(n – 1). In such diagrams, you must indicate arrows because arrow shows the direction of flow of the signal. You multiply x(n – 1) by b_1 and add it in the first summer. To make things a little more involved, let us say that we have another summer. We could have combined these two summers but we did not do that. We use them separately, because no summer should have more than two signals. The output of the second summer is y(n) after another signal is added. This last signal is derived from y(n) through another delay z^{-1} and multiplication by d_1. Then you can see that $y(n) = b_0x(n) + b_1x(n-1) + d_1y(n-1)$. This is the symbolic diagram for this digital system. In this digital system, the describing equation is a difference equation, which is the exact counterpart of differential equation in analog signal processing. Any dynamic digital system will be described by a difference equation and a difference equation can always be represented by a schematic diagram. This diagram represents hardware as well as software. In terms of hardware, it indicates delay elements, multipliers and summers. In terms of software, it shows what things are to be multiplied by what and which signal has to be retrieved from the storage. So it is a hardware as well as software description of a digital system.

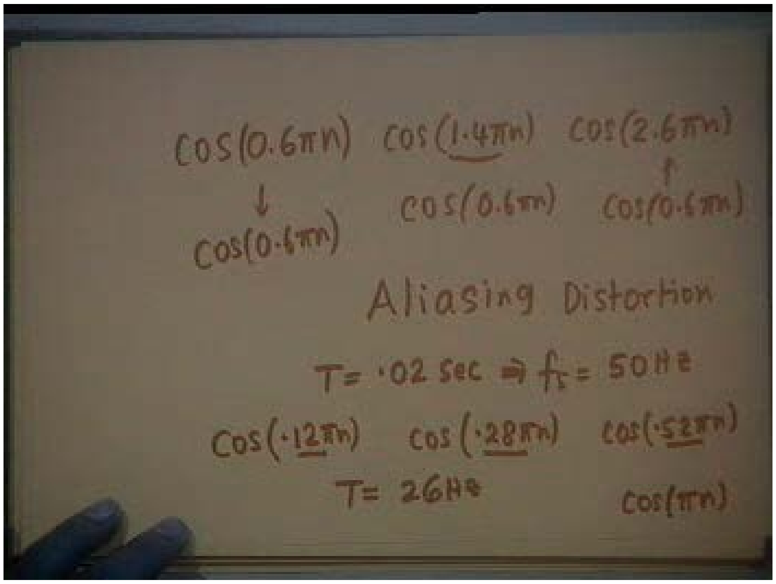
We next look at the sampling process. We already introduced ω as the normalized digital frequency. We now look up the sampling process in a little more detail so as to appreciate why ω is restricted to lie between $-\pi$ and $+\pi$.

(Refer Slide Time: 37:08 - 39:33)



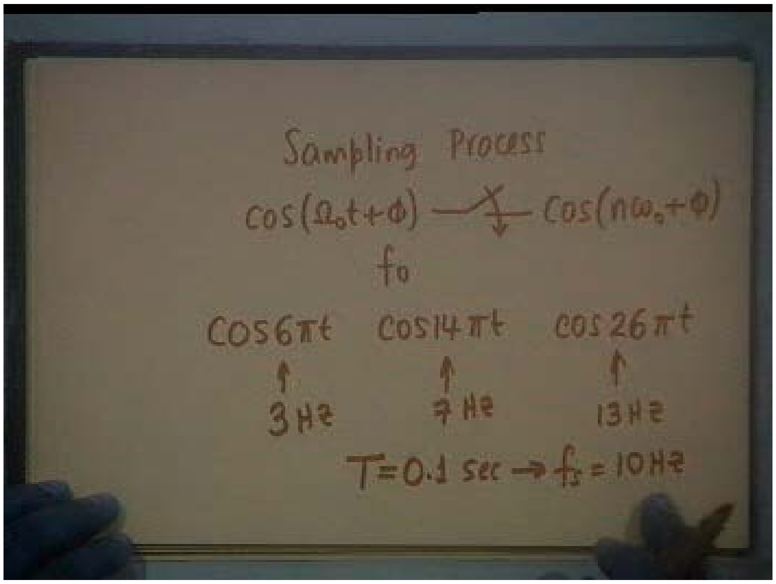
If you have a signal cos (Ω_0 t + Φ) then after sampling, it is given to A to D. We do not show A to D because the mathematical techniques are the same whether you have done quantization or you have not done quantization. What you get after sampling is cos ($n\omega_0$ + Φ), where ω_0 is $2\pi(f_0/f_s)$. I have told you that f_0 must be less than or equal to $f_s/2$. What happens if this is not the case? We will illustrate with a simple example. Let us say we have three signals: cos 6πt, cos 14πt and cos 26πt. We have avoided Φ also and made the signals as simple as possible. What are the actual frequencies? These are 3 Hz, 7 Hz and 13 Hz. Suppose the sampling interval is 0.1 second. It correspond to f_s = 10 Hz. Obviously, 10 Hz is not the correct sampling frequency for the second and third signals but let us see what happens.

(Refer Slide Time: 39:35 - 44:10)



When sampled the signal cos (6πt) will become cos (6πnT).

(Refer Slide Time: 39:41 - 39:49)

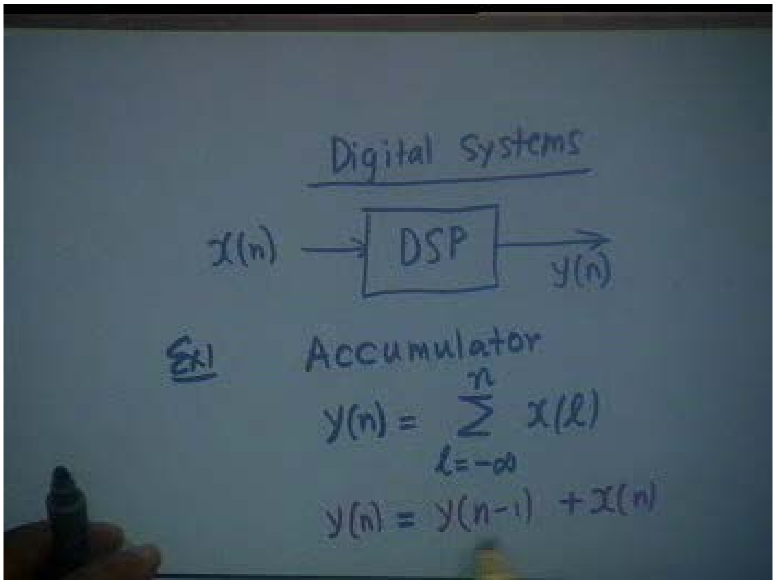


T is 0.1. Therefore, the first signal will become cos (0.6π n), the second signal will become cosine (1.4π n) and the third signal will become cosine (2.6π n). But 1.4π n is 2π n – 0.6π n.

What is cos (2π n – 0.6π n)? It is the same as the cosine (0.6π n). Similarly 2.6 is 2 + 0.6. Therefore this signal also becomes cos 0.6π n. In other words the three signals become indistinguishable. This is what inadequate sampling frequency does. The 7 Hz signal now poses as if it is a 3 Hz signal. A high frequency poses as a low frequency. Similarly 13 Hz signal poses as 3 Hz. Therefore if the initial signal was a combination of 3, 7 and 13 Hz, the signal after digitization or sampling will become simply 3 Hz. Higher frequencies pose as low frequencies. This process is known as Aliasing, commonly used in criminal language, like a terrorist's name is so and so, alias so and so. And the distortion due to this process is known as Aliasing distortion.

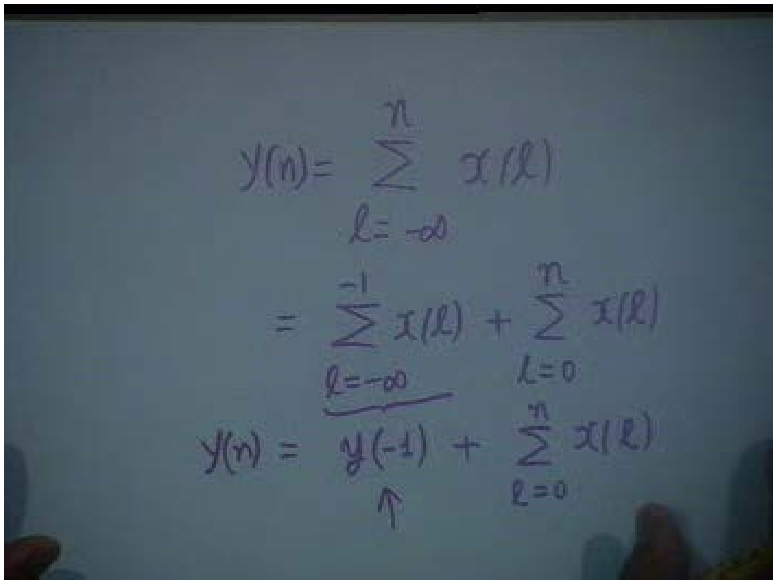
Now the way to contain aliasing distortion is to ensure that this quantity ω_0 in $\cos(\omega_0 n)$ is less than or equal to π. For example, suppose T is 0.02 second, that is f_s = 50 Hz. Then what will be the sampled signals? Signals would be cos (0.12π n), cos (0.28π n) and cos (0.52π n). Each ω_0 is less than π. These three signals are distinct and they retain their original identity. What is the lowest sampling frequency which shall make the highest frequency $\omega_0 = \pi$? Obviously, 26 Hz that is, T = 1/26 will make the last signal as the cos (π n). This essentially is sampling theorem, illustrated here with the help of a very simple example, If the sampling frequency is not adequate, i.e. it is not at least twice the highest frequency content of the signal, then there occurs aliasing distortion and the message would be distorted. There is no way that you can recover the original signal. Now we come back to digital systems.

(Refer Slide Time: 44:40 - 47:37)



The digital system or digital signal processor takes a digital signal x(n) and makes calculations with x(n) and its previous samples, and possibly previous values of the output to produce a new output y (n). Before we go into the formal characterization of digital systems, let us take a few examples. The first example that we take is that of an accumulator, which simply adds up the signals. The output is the sum of the signal x(ℓ) where ℓ goes from – ∞ up to the present instant n. Now I can write this in many different ways. One of the ways is to recognize that y(n) is y(n – 1) + x(n). So this accumulator can be described by a difference equation where we have avoided the infinite summation. We have expressed it in terms of finite quantities, but with recursion, i.e. feedback. y(n) is computed by adding x(n) to the immediate past output. This immediate past output has to be brought back from the storage, so it is a feedback. Whenever there is a recursion or feedback, there is a possibility of oscillation or instability. On the other hand, the original equation is non recursive; it does not require past output to be brought back.

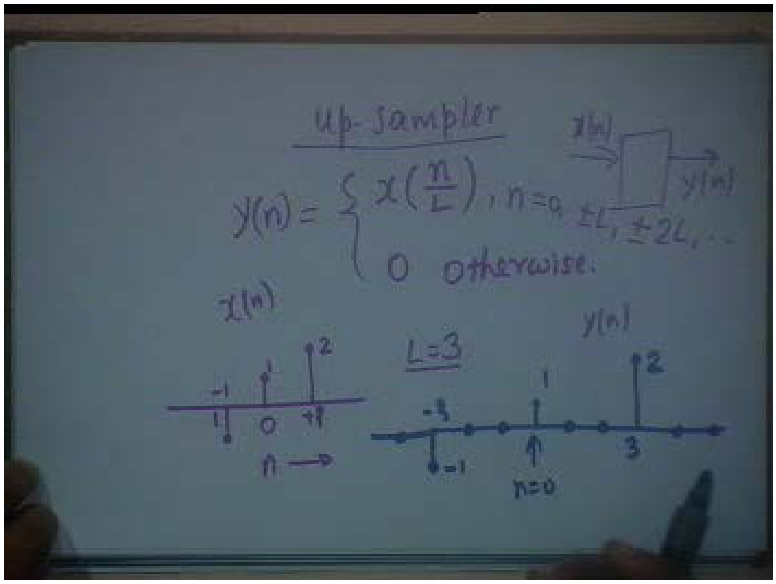
(Refer Slide Time: 47:42 to 49:16min)



This equation can also be written in a different form as y(n) equal to summation of x(ℓ) from – ∞ to – 1 added to summation of x(ℓ) from ℓ = 0 to n. Now, if you notice carefully, the first summation simply is y(– 1), which can be treated as the initial condition. That is, we start computation from l = 0, but we have to take care of what existed before n = 0 that is the accumulated content at n = – 1. To this, we keep on adding. Depending on how we characterize the accumulator, you will see that the system characteristics change. The first two characterizations make the system linear, while the last one makes it nonlinear.

Let us take another example: the up sampler. That is, you want to increase the rate of sampling. This is very commonly used in Digital Signal Processing. Similarly, one also uses down sampling.

(Refer Slide Time: 49:25 - 53:11)



In up sampling, the relationship between y(n) and x(n) is y(n) = x(n/L). Obviously y(n) will exist when n is an integral multiple of L. Digital signal exists only when argument is an integer, positive or negative, so y(n) exists for n = 0, ± L, ± 2L, and so on, and it is 0 otherwise.

What does an up sampler actually do? Let us look at a typical signal and see what an up sampler does. Suppose at 0, we have a sample – 1 at n = – 1, + 1 at n = 0, and we have 2 at n = + 1. Suppose L = 3. L has to be an integer number. What happens to y (n)? Let us try to draw this, at 0 it is 1 (n = 0/3 is 0), then at n = 1, x(1/3) does not exist so at n = 1 it is 0. Similarly, at n = 2, it does not exist. At n = 3 it exists and the value is 2. At n = – 1, it does not exist, at n = – 2 it does not exist, but at n = – 3 it is – 1, and the rest of the samples are 0. So what has the up sampler done? It has not changed the picture; the picture is the same but it has expanded. How? It has added two 0’s in between two consecutive samples of x(n); so up sampling by a factor of 3 means filling up 3 – 1 i.e. two 0’s in between consecutive samples. Later, we shall see that this changes the nature of the system. We shall also see that it is a time varying system.