LECTURE 2

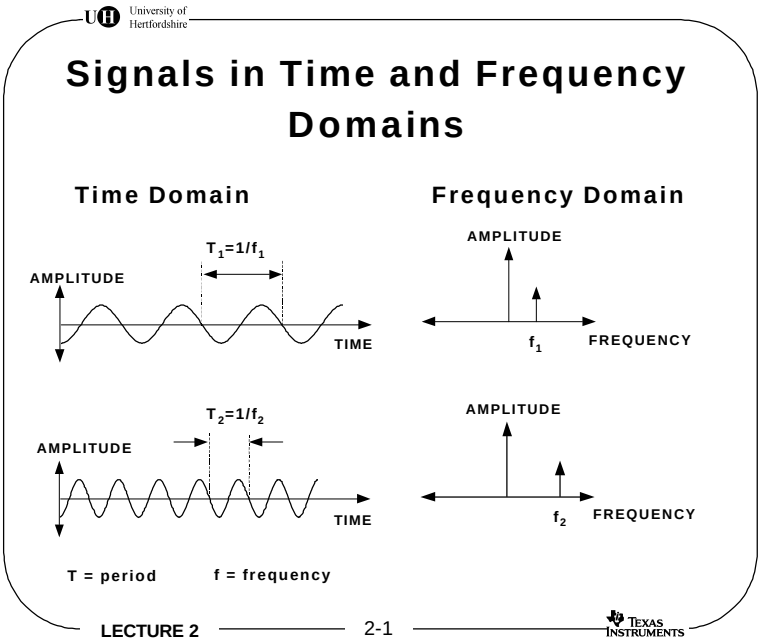
FROM ANALOG TO DIGITAL

OBJECTIVES

The objectives of this lecture are to:

- Introduce sampling, the *Nyquist Limit* (*Shannon's Sampling Theorem*) and representation of signals in the frequency domain
- Introduce basic concepts of analog-to-digital conversion (ADC) and quantization noise
- Consider practical ADC and DAC devices
- Consider the functional blocks of the C54x DSP Starter Kit board

This lecture covers the sampling and analog-to-digital conversion processes.



All electronic signals can be visualized (signal representation) using two basic methods – the *time domain* and the *frequency domain*. If we have an electronic signal running along a copper wire, we need some method of displaying what that signal is.

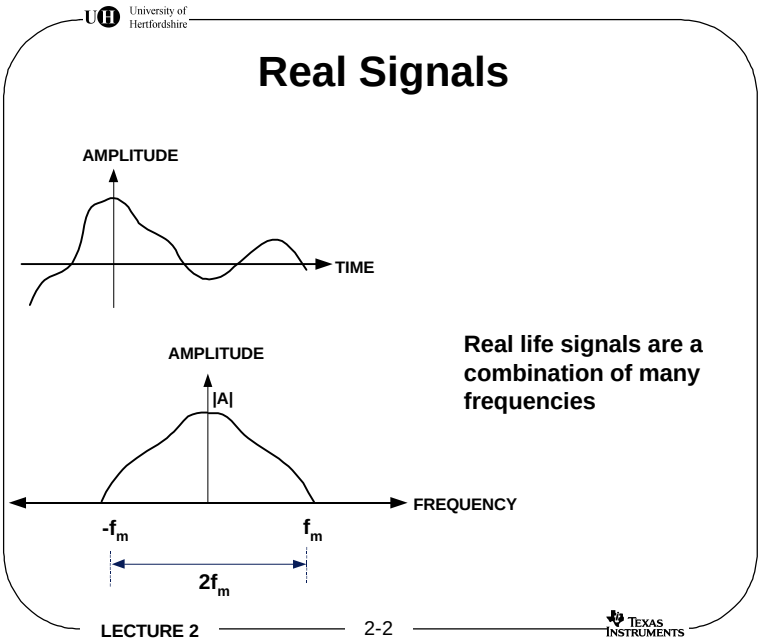
- **Time Domain**

The *time domain* is the form of visualization that most people are familiar with. This method shows variations of a signal with time. The most common time domain instrument is the oscilloscope, which has graduations of volts on the Y-axis and graduations of time on the X-axis.

A pure sine wave is shown on the left side of the above diagram. The signal has a *frequency* (the number of times that the signal repeats itself in a second) and a *period* (the time duration of one complete signal cycle). Frequency and period are not different quantities, but different methods of describing the time measurement of the same signal. The diagram shows two waveforms. The lower one has a higher frequency (more cycles per second) and a shorter period (less time to complete one cycle). Both signals have the same amplitude.

- **Frequency Domain**

The *frequency domain* is very important in the field of DSPs. Instead of showing the variation of a signal with respect to time, we show the variation of the signal with respect to frequency. Most people are familiar with the graphical displays in music systems that show how much bass or treble is in the music. This is a frequency domain display. The most common instrument for displaying the frequency domain is the spectrum analyzer. The right side of the diagram shows a frequency domain display. In the top graph, we see a single line representing a single pure frequency (f_1). In the lower graph, we see a higher-frequency signal (f_2). This line is further to the right on the frequency axis.



- **Real Signals**

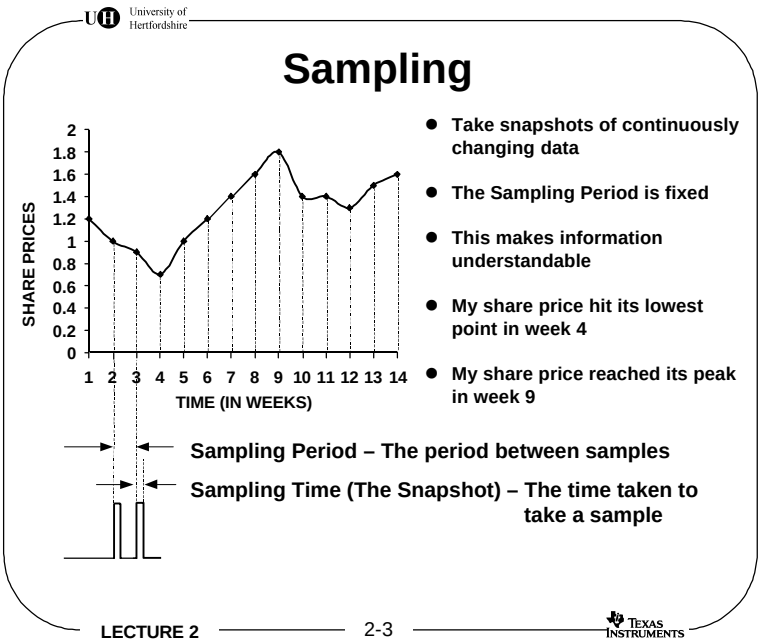
Real signals, i.e., sound or speech, are typically more complicated than a simple sine wave. They consist of many frequencies of differing amplitudes that combine to a composite signal. For instance, if you spoke the single letter "F," your voice would produce a range of higher-frequency signals than if you said the letter "W." Likewise, if you spoke the letter "E," you would find that it consists of one main frequency and a few lesser frequencies. You will have an opportunity to see these frequency patterns for yourself in later demonstrations.

- **Bandwidth**

The bandwidth of a signal is the difference between the maximum and minimum frequencies above a certain amplitude (in the above case it is f_m). Knowing the bandwidth of a signal is very useful, as without this information we might either design equipment which cannot process all of a signal, or is over specified and costs too much for the required purpose. For instance, we wouldn't use an audio amplifier to amplify a 1MHz signal because an audio amplifier is only designed to amplify signals that we can hear (the highest frequency we can hear is around 20kHz). In the diagram above, instead of a single line of a set frequency and amplitude, we see a spread of frequencies and amplitudes, which illustrate the highest and lowest frequencies that exist in the signal. The bandwidth of a signal tells us nothing about the contents of the signal. For example, we could have two signals, both with a bandwidth of 10kHz. One signal could have a signal ranging from 5kHz to 15kHz, while the other might have signals ranging from 500kHz to 510kHz.

- **Spectrum**

The shape of the signal in the lower diagram shows the distribution of signal energy with frequency or the *spectrum* of the signal. In this case a high proportion of low-frequency energy is present. Note that the above diagram is also an *even* function, showing both positive and negative frequencies. This common representation of a signal is used because the frequency component of a signal can be expressed as the sum of two exponentials (this will be explained later).



- **Sampling**

The conversion of analog signals to digital and back again was mentioned in Lecture 1. We will now look at the conversion process in more detail. One of the stages in the process of converting analog signals to a digital pattern is known as *sampling*. To illustrate the operation of sampling, we will use the example of variations in stock market prices over a period of several weeks.

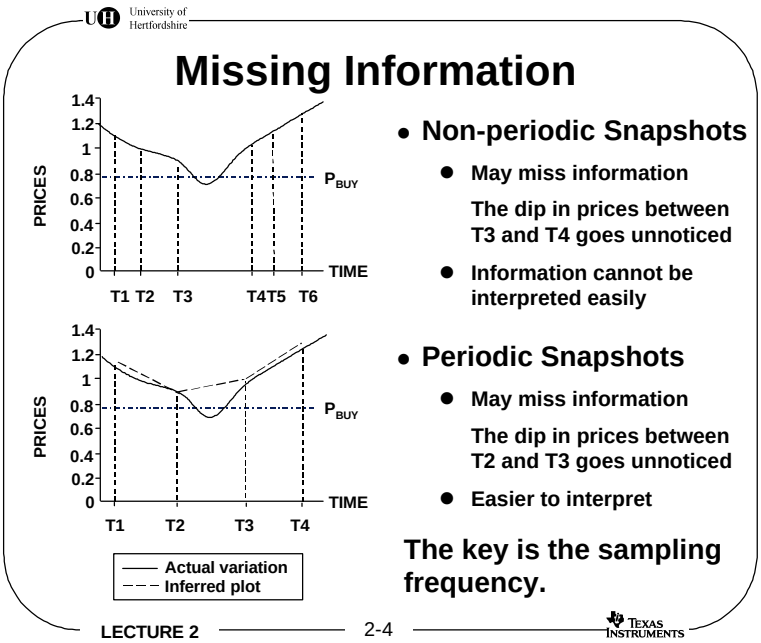
- **Sampling Period and Sampling Time**

A *sampling period* is the time between samples. *Sampling time* is ideally an instant in time when a sample is taken. In reality, taking a sample always takes a finite amount of time, so the sampling time (duration) is not zero. In our example, the sample is taken each Monday between 5:00 and 6:00 oClock. The sampling time is therefore one hour. Usually the sampling time is much smaller than the sampling period.

In the above diagram, the share prices are recorded at one-week intervals. When we plot these values against time, we end up with a set of snapshots of the stock market prices. These are the *samples* of our input signal. In other words, sampling is a method of recording the instantaneous value of the signal. In the above diagram, we take samples at weekly intervals to ascertain the performance of the stock market.

We must decide how often we need to take samples to gain an accurate representation of changes in the stock market. There are several different alternatives. We could sample the stock market every day instead of every week. This would give us a much more precise representation of the stock market fluctuations. But do the stock market prices change significantly enough to warrant the more frequent *sampling rate*? We could take it to the extreme and sample the stock market every minute, but the current values of the shares may fluctuate and produce unwanted noise. We could take the opposite approach and sample the prices once each month or year, but then we might miss when the shares reached their highest and lowest values.

Alternatively, we could take samples sometimes daily, sometimes weekly, or sometimes with a gap of ten days, but how useful would our information be from such samples? Likewise, can we guarantee with the samples shown in our diagram that the prices of the shares did not fall even more between weeks 4 and 5, and perhaps more importantly, are we interested if they did?



This section builds upon our stock market example. We want an automatic system for determining when we should buy our shares. If the price of the shares falls below a set level (P_{Buy}), we want to increase our investment.

- **Non-Periodic Snapshots**

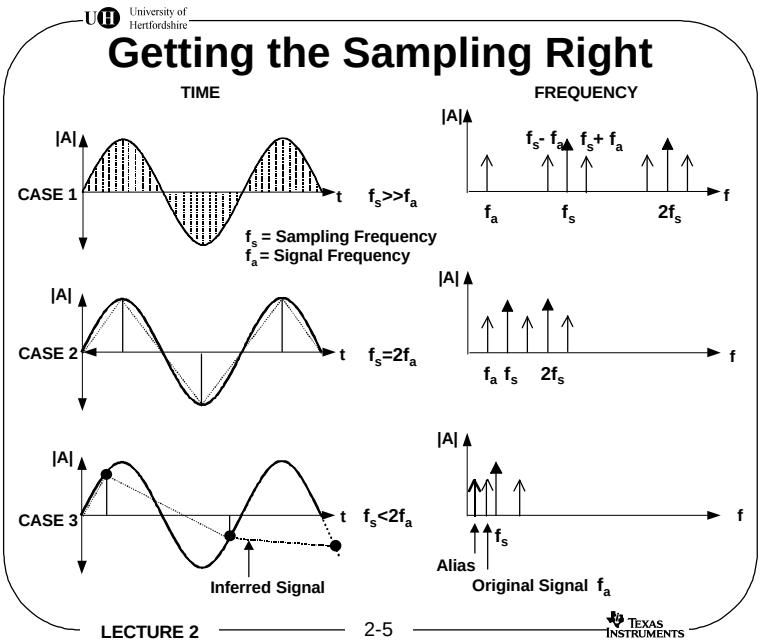
In this case, we assume that the stock index is going to vary continuously over time as shown above. We also assume that the stock market index is only published at irregular times (T1 to T6) in the upper half of the above diagram. We can plot these published values against time, but since we have no information other than the stock index at the six intervals, we must infer what happened between the published values. The easiest way to do this is to join the values with a simple curve. In our example, the implication of this is that we don't see the dip in the stock index that actually occurred between T3 and T4. We missed the point in time when we meant to buy our shares. From this, we can see that non-periodic sampling has two drawbacks: First, it is not easy to interpret the data, and second, we may miss important information.

- **Periodic Sampling**

The frequency at which the samples are taken is critical to capturing meaningful information about an analog signal. In our example, even if we sampled at regular intervals, if our *sample rate* was too infrequent, we could still miss important data such as the dip below the *buy* level (see above diagram). Although this second graph is easier to interpret, we have still missed the all-important price dip between T2 and T3. So, periodic sampling by itself does not guarantee that we don't miss important data. If we plot the graph using the samples available, the inferred plot misses information.

- **The Key**

How can we ensure that we do not miss the dip in prices? The key is sampling frequency. If there were more samples, we would certainly be able to see the dip. Is there a rule by which we can determine a safe sampling period so that we do not miss information? Fortunately, the answer to this question is yes.



- **Sampling at a Very High Frequency**

By sampling at a very high frequency compared with the signal frequency, we can ensure that we do not miss important information. Another benefit of sampling at a high frequency is that from the samples, we can rebuild the signal almost as well as the original. As shown on the diagram above, if we plot the inferred signal from the samples, we will get a waveform very similar to the original.

When an analog signal is sampled, multiple images of the spectrum occur in the frequency domain. These images are centered around the base band (original signal) and multiples of the sampling rate. This is comparable to having multiple analog-mixer circuits.

- **Sampling at the Nyquist Rate**

What would happen if we reduced the sampling frequency? In the time domain, we would be taking less samples per period. In the frequency domain, we would see the multiple images of our signal centered at 0Fs, 1Fs, 2Fs, etc … The lower the sample rate (Fs), the closer the images are to each other. A limit is reached when $f_s = 2f_a$, where an overlap occurs. The time domain picture looks like our Case 2 plot, where we are taking two samples per period.

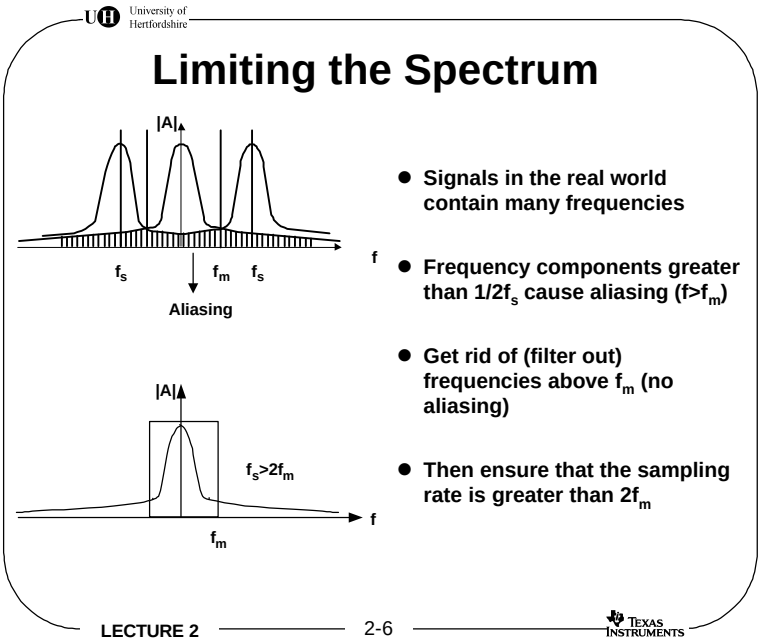
- **Case 2 (Blank box on the slide should contain $f_s = 2f_{a)}$**

Case 2 is a special case, when ($f_s = 2f_{a)}$. There are only two samples per period. The inferred plot of the signal from the samples is still a good approximation to our original signal. The caveat here is that if the signal were shifted by 90 degrees we would loose all amplitude information because the samples would occur on the zero crossings. Thus it is important to sample at least *slightly over* the Nyquist rate.

- **Sampling Below the Nyquist Rate**

A further reduction of the sampling frequency will cause one sample to be taken per period. The Case 3 plot shows the effect of this in both time and frequency domains. In the time domain, the inferred signal does not look like our original signal. In the frequency domain, an *alias* of our original signal spectrum appears near the frequency of our original. We are now in a situation similar to that of the share prices. It is not possible to reconstruct the original signal from our samples. This effect is called *aliasing* in signal processing. It should be noted that aliasing a signal is not *always* a problem and can, in fact, be beneficial in some cases where exact reconstruction of the signal is not required.

As you may have already gathered, the minimum sampling rate must be twice that of the highest frequency component of the signal. This frequency is called the Nyquist Limit**.** The theory behind this originated from *Nyquist's Sampling Theorem*. *Shannon's Sampling Theorem* also states the same fact. We cannot reconstruct the original signal if it was sampled at a rate below the Nyquist limit. We will end our stock market example by saying that one of the keys to being a winner in the stock market is to ensure that you sample the share prices at twice the frequency that they change!



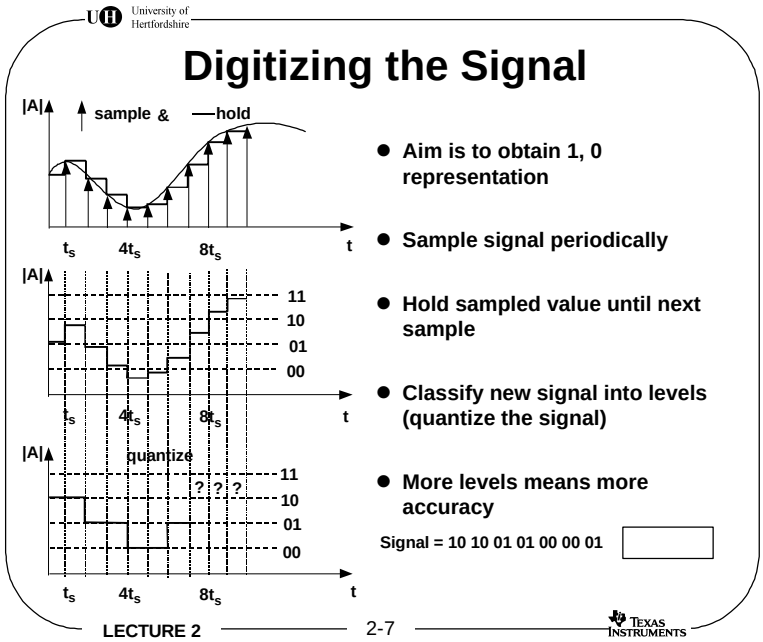
- **Spectrum of Real Signals**

In the real world, we rarely come across a pure sine wave as shown in the previous example. Typical signals have a wide spectrum as shown in the above diagram. Information above or below a certain frequency or amplitude is usually considered surplus to requirements. For example, hi-fi manufacturers assume that 20kHz is the highest frequency that the ear can detect. Assuming that f_m is the maximum frequency of interest in a signal, we would then like to sample at the Nyquist Rate, i.e. at $f_s = 2f_m$ (the sampling frequency = twice the maximum frequency). If we go ahead and sample at $2f_m$, we obtain a spectrum as shown in the top diagram. The bands overlap and we produce aliasing in the output signal. To avoid this, we must first use a low-pass filter on the input signal to remove any frequency components above f_m.

- **Limiting the Spectrum**

This low-pass filter is called an *anti-aliasing* filter. These filters are found at the input to most applications. Because they are needed before the signal is sampled, they are almost always analog filters. It is theoretically possible to use the DSP device to remove the aliasing, but this would require a high sample rate. In practice, this is rarely done because it would consume too much of the processing power of the DSP, which we need to perform other tasks.

The ideal anti-aliasing filter characteristics are shown in the lower half of the above diagram. It is flat with gain A over the pass band and zero at any other point. An ideal filter would also have a linear phase response. Variations from this ideal may have a significant effect on performance. For example, in control applications, a badly designed filter may mean that the system spends all of its time trying to compensate for the filter, rather than trying to maintain a constant motor speed. In audio applications, a poor phase response can lead to harmonic distortion and degradation that is often audible. For audio applications, commercially available switched-capacitor filters are easy to use and provide a reasonably linear response.



- **Digitizing the Signal**

We have now covered the concept of sampling a signal and filtering it with an anti-aliasing filter. The next stage is to convert the signal to a digital representation. In real world designs, the basic sampling function is achieved by using a *sample and hold* circuit, which maintains the sampled level until the next sample is taken. The result of this is the *staircase effect* shown in the upper section of the above diagram. It may not be apparent at first why we need to hold the input value at a particular level. After all, we have described sampling as taking a *snapshot* of the input signal. The reason is quite simple. It takes a finite amount of time for an analog-to-digital converter to execute its conversion process. During this conversion time, the input signal must not change significantly. If the signal is allowed to vary on the input of the ADC during the conversion, we might end up with a meaningless digital output, since the ADC performs different stages of its conversion with a different input voltage.

We must now devise a method of representing these sampled values of our analog waveform as a number that can be used by our DSP. This function, called *quantization*, is performed by an analog-to-digital converter.

Quantization can be thought of as classifying the signal into certain bands. In our example, we chose to use a four-level 2-bit quantizer. The quantizer determines where the amplitude of a particular sample falls within the four levels. Each level is assigned a 2-bit code in ascending order. It is useful to note at this stage that:

Digital	Decimal
00	0
01	1
10	2
11	3

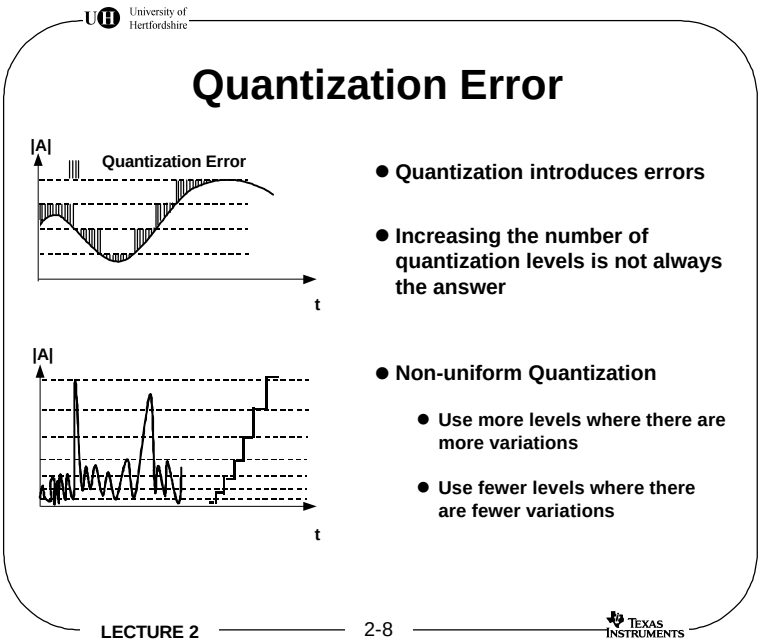
In this example, the quantizer rounds up, thus a signal level that *just* exceeds the level for 01 is sampled as 10. It is also possible for a quantizer to round down. In practice, ADCs generally *do not* round to the nearest sample. The quantization produces a sequence of 2-bit digital values of the input signal. From the input signal in the graphic on the previous page, the following digital output sequence is produced:

10 10 01 01 00 00 01 __ __ __

What are the final three values?

- **Accuracy**

Any number of levels is possible, and the more levels there are, the more accurate the digital representation. We shall look at this point more closely. As in all problems in engineering, this is another issue for compromise.



- **Quantization Error**

Quantization by its very nature introduces errors. There are two primary sources of errors. One is sampling, which only takes the amplitude of the signal at a point in time and holds it until the next sample. The second source of errors comes from the quantizer, which pulls up or pushes down the amplitude of the signal to its digital representation. The quantization graph above shows the kind of error levels involved in our case. This error produces an effect which is called quantization noise. In audio or speech applications, this error appears as noise on the output. If we take a typical DSP application with a 10- to 12-bit ADC, the quantization noise is usually negligible compared to other noise sources.

- **Reducing Quantization Error**

One way to reduce quantization error is to increase the number of quantization levels. This will certainly reduce the errors in some cases, since there will be a quantization level nearer to the actual sample.

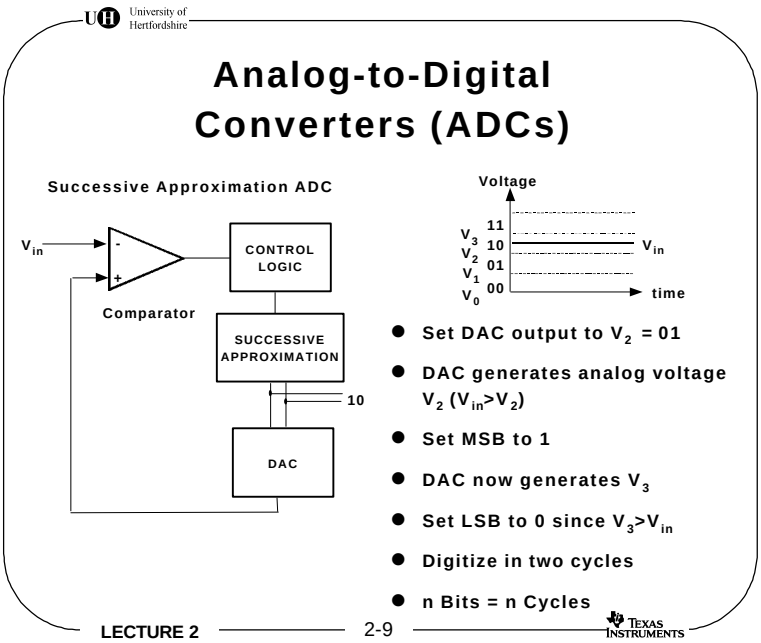
The more levels we have, the more accurately we can describe the analog signal. In general, a DSP system will use an ADC with 10 or 12 bit resolution. This means that the input signal will be measured against 1024 or 4096 levels, respectively. Therefore, if our input signal varies between 0 and 5V, the least significant bit (LSB), i.e., a single bit, would correspond to just 4.88mV for the 10-bit ADC ($5V/2^{10}$) and 1.22mV for the 12-bit ADC ($5V/2^{12}$) assuming a uniform quantization step.

Note that increasing the number of quantization levels is not always the best answer. Consider a case where signal amplitudes are grouped, as shown in our second graph above. The bottom portions of the graph contain more variations in amplitude. The top portion of the waveform does not change much. If we decided to apply a non-uniform quantization to this waveform by allowing more digitization levels where there are more variations, we can more accurately represent the waveform in the digital domain with fewer

ADC bits. A *step size* is used that varies according to the signal amplitude. In this case, we would ensure that there are more levels at the lower amplitudes.

In practice, the quantizer has a uniform step size, and it is the high input signal that is compressed. In our above example, the higher-amplitude signals can be compressed, leaving the lower amplitude signals unchanged. The overall effect is a non-uniform quantization. After processing, the signal is reconstructed at the output by expanding it. Performing this process of compression and expansion is known as *companding*.

The most widely used application of companding is in the public telephone system. There are two distinct companding schemes used in Europe and the USA. The method used in Europe is called A-law and the method used in the USA is called μ-law.



- **Practical ADCs**

We now know, in theory, how to convert an analog signal to digital. The steps are:

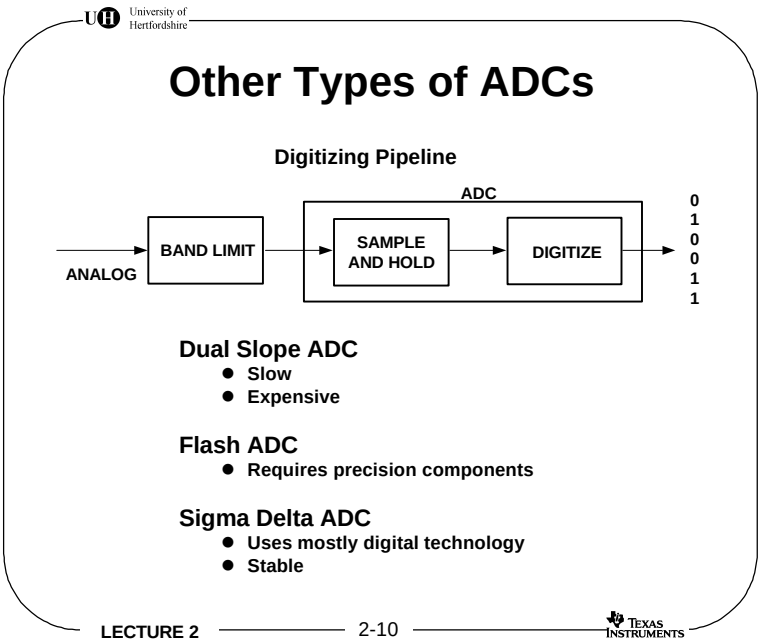
1. Limit the spectrum
2. Sample and hold
3. Quantize each sample
4. Obtain digital data stream

- **Successive Approximation ADC**

The successive approximation ADC is the most common type of ADC, with a typical resolution of 8 to 12 bits. We will use a 2-bit example for simplicity. The successive approximation ADC consists of four primary functional blocks. Its technique is to keep splitting the voltage range in half to determine where the input signal lies.

In the above diagram, the input voltage is shown as V_{in}. This ADC is a 2-bit device, so there are four possible ranges in which the input signal can lie. On the first cycle, the output of the DAC is set to ½-scale, meaning the overall voltage range of the device is divided by two. The input signal is then compared with the DAC voltage. If it is higher, the most significant bit (MSB) is set to 1, and if lower, the MSB is set to 0. Assuming that the input signal is greater than ½-scale (i.e., MSB = 1), the DAC changes to ¾-scale and V_{in} is compared with this new value. Then, V_{in} and the DAC voltage are compared again to determine the second and final bit. This process can be extended to as many bits as needed.

Successive approximation ADCs are relatively inexpensive, and are generally accurate and fast. Conversion times vary from 1µs to 50µs with a resolution of 8 to 12 bits. This type of converter operates on only a brief sample of the input waveform and spikes in the input can produce flawed outputs. Its ability to follow changes in the input signal is limited by its internal clock rate, so it may be slow to respond to sudden jumps in the input signal.



- **Other Types of Practical ADCs**

We will only briefly cover the other types of ADCs available. For more information, refer to literature that covers analog-to-digital conversion.

- **Dual Slope ADCs**

Dual slope ADCs use a capacitor connected to a reference voltage. The capacitor voltage starts at zero and is charged for a set time by the output voltage of a sample-and-hold circuit. The capacitor is then switched to a known negative voltage reference, and charged in the opposite direction until it reaches zero volts again. This second charge is timed with a digital counter. The final count is proportional to the input voltage.

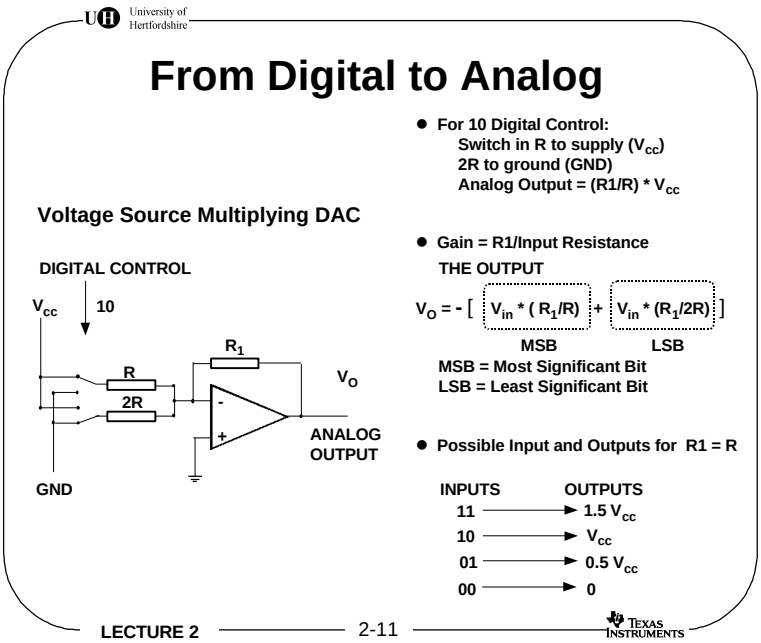
This technique is very precise and can produce ADCs with high resolution. The dual slope technique ensures that the greatest number of component variations is canceled out. For example, drifts or scale errors have no effect since we start and finish at the same voltage. These converters, however, are very slow and generally more expensive than successive approximation ADCs.

- **Flash ADCs**

Parallel flash ADCs convert the analog input voltage faster than other types of ADCs. They compare the voltage in parallel with a series of comparators that are attached to a resistive network. Their main advantage is that they work very fast since they produce a digital output in a single cycle. They are the only types of converters that can be used for very high-speed conversions such as video processing. Their disadvantage is that the resistors in the network must be matched and laser-trimmed for accuracy. Because of the exponential increase in components, flash ADCs are very expensive if more than 6 bits are required.

- **Sigma-Delta ADCs**

Sigma-Delta ADCs have the advantage of being implemented almost entirely in the digital domain. They are very reliable, highly stable, and extremely functional. Because of their largely digital composition, they can be integrated onto the same silicon as a DSP. Most ADCs are based on the principle of sampling and converting a signal directly at the sample rate; however, this requires precise component matching and low noise circuit design. Sigma-Delta ADCs use a low-resolution ADC (1-bit quantizer) with a sampling rate many times higher than the Nyquist frequency. This results in a conversion noise that is spread over a very wide bandwidth. Since the desired output bandwidth is a tiny fraction of the internal bandwidth, a filter can be used to remove the noise. Essentially, the digital filter trades high bandwidth for better noise performance.



- **Practical Digital-to-Analog Converters**

In many DSP applications, we must reconstruct an analog signal after the digital processing stage. This is done using a digital-to-analog converter (DAC), which is considerably less expensive than the ADC. Depending on their design and resolution, DACs normally have a settling time of between 100 ns to 1.2μs.

- **Voltage Source Multiplying DAC**

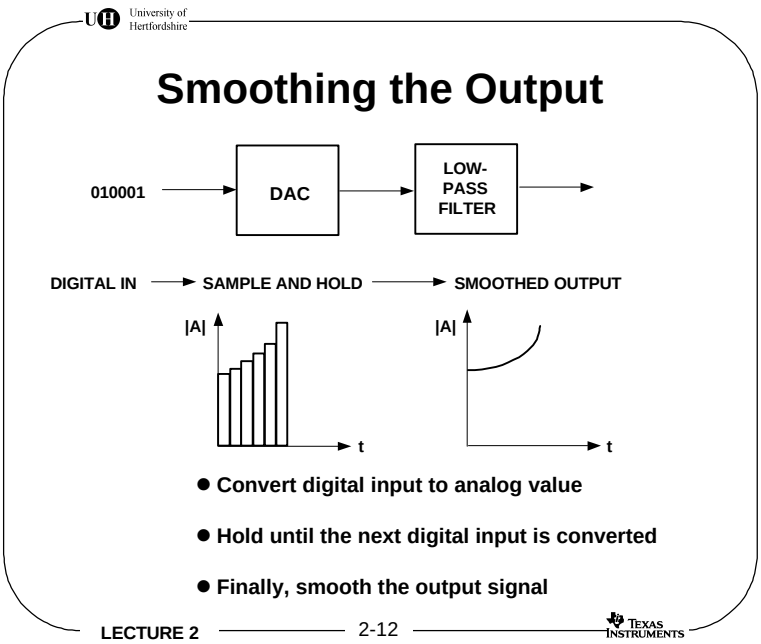
Voltage source multiplying DACs use a reference voltage which is switched in or out by the digital data. The converter is so-named because it multiplies a certain gain value with a source voltage (V_{cc} in this case).

Let us consider a 2-bit example. To convert 10 to analog, resistance R is switched in by digital control logic, and the resistor 2R is grounded. The gain of the circuit is R1/R, which can be adjusted to a suitable value. For the sake of the example, let R1 = R. The overall gain of the circuit would then be unity, which would give an output of V_{cc}. The general conversion equation for this circuit is:

$$V_o = - [V_{in} * (R_1 / R) + V_{in} * (R_1 / 2R)]$$

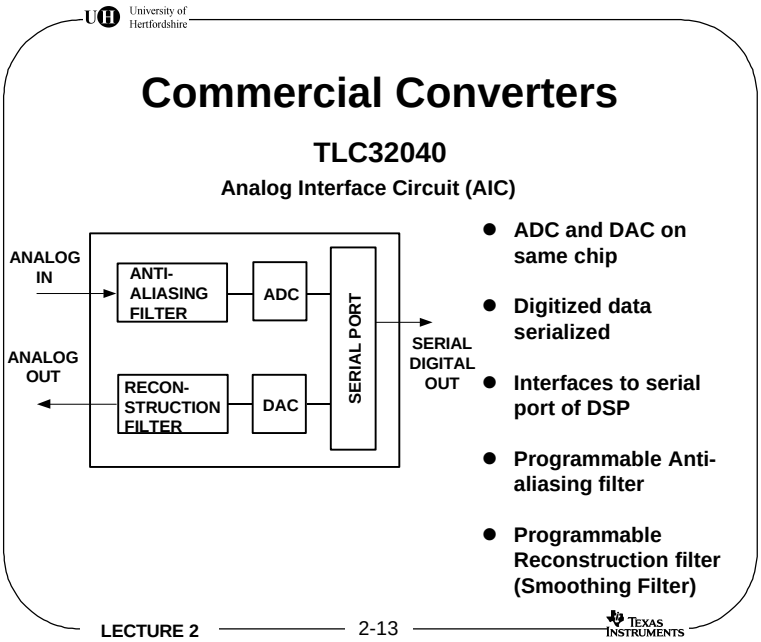
A 2-bit conversion table would be as follows:

Inputs	Gain	Analog Output
11	3/2	$(1.5)V_{cc}$
10	1	V_{cc}
01	½	$(0.5)V_{cc}$
00	0	0



- **Smoothing the Output**

The output of a DAC is stepped, just like the analog waveforms that were sampled and held. This stepped, or staircase effect is a distortion, and it may be desirable to reduce this effect. To this end, a low-pass smoothing filter is used. This filter is referred to as a *reconstruction* filter.



- **Commercial ADCs and DACs**

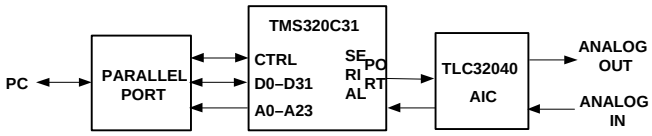
Analog interface chips (AICs), such as those in the Texas Instruments TLC32040 series, contain all of the analog components that we have previously described. They contain the ADC and DAC, plus anti-aliasing and reconstruction filters. This combined ADC and DAC is on the DSK board, so it has a direct relevance for the demonstrations and general use of the DSK board.

The above diagram shows a simplified functional block diagram of the device. The ADC path has an on-chip anti-alias filter. After digitization, the output is supplied in serial form with common microprocessor control signals. Both the ADC and DAC have 14-bit resolution.

The DAC path requires a serialized digital input. The output of the DAC is smoothed using an on-chip filter with programmable filter cutoff frequencies. The device is easily interfaced to the serial port of a DSP device.

University of Hertfordshire

C54x DSK



- 'C54x, 16-Bit Floating Point
  - 2K x 16 Bits On-Chip RAM
  - 64 x 16 Bits On-Chip Cache
  - 40 ns Single-Cycle Instruction Execution Time
- Analog Interface Unit (AIC) Contains
  - ADC (14-bit)
  - DAC (14-bit)
  - Filters
- Parallel Port
- JTAG and Expansion Connector

LECTURE 2 2-14

- **Functional Blocks of the DSK Board**

We now know some basics about DSP, ADC, and DAC chips. Most DSP systems have these components in common. Since we shall be using the C54x DSK board in our demonstrations, it will be good practice to learn about its design and functionality.

The DSK board consists of:

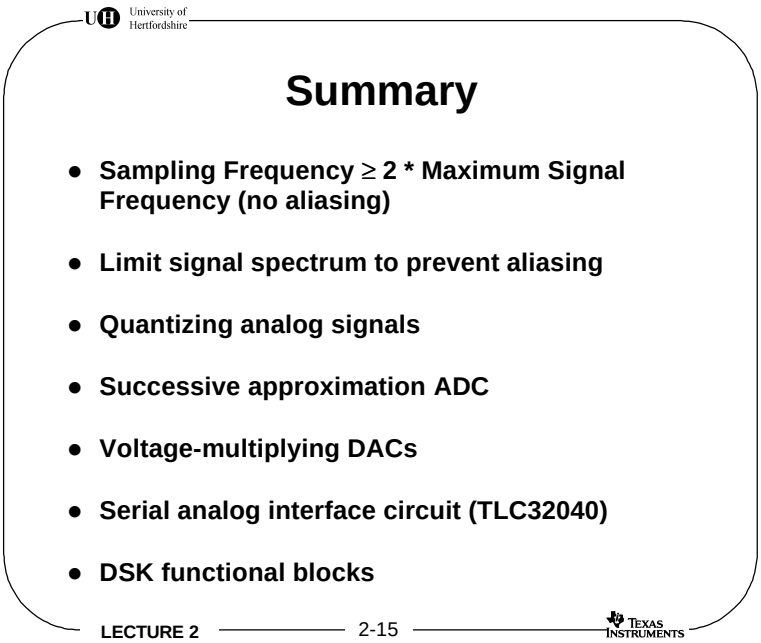
A C54x DSP running at 50MHz (25 MHz processor clock)
The TLC32040 Analog Interface Circuit
A parallel port

The C54x DSP has 2Kx16 bits of on-chip RAM, which is adequate to store both program and data. It has an on-chip cache unit of 64x32 bits to enhance instruction execution efficiency. The C54x has an instruction execution time of 40 ns.

The AIC is used to handle all analog-to-digital conversions. It communicates with the C54x via the synchronous serial port. It has a DAC and an ADC with 14-bit resolution. It also has on-chip anti-aliasing and smoothing filters. Two RCA connectors provide analog input and output on the board. It is also possible to disconnect the AIC completely and use an alternative AIC on an expansion board.

The C54x DSK board has a parallel port for connection to a PC for communication with the outside world. The parallel port can be run in standard or in enhanced mode. When communication is first established with host PC, a communications kernel is loaded. This software component manages all the communications with the PC and is used to load and execute user programs as well as for debugging. The debugger runs on the PC and displays contents of internal registers, internal RAM, and disassembled instructions that are useful for debugging programs.

Most signals are brought out to connectors on the side of DSK board to ease expansion. PCB holes for a JTAG header are provided. The JTAG port enables hardware emulation without removing the C54x from the circuit.



- **Sampling Frequency**

Because real analog signals contain a number of frequencies, the spectrum of input signals should be limited to ensure that no signal energy exists above a certain frequency. A safe sampling frequency can then be calculated.

- **Limit Signal Spectrum to Prevent Aliasing**

The sampling frequency must at least be twice that of the highest frequency in the incoming signal to prevent aliasing. This rate is commonly called the Nyquist limit.

- **Quantizing Analog Signals**

A typical analog-to-digital conversion pipeline consists of a spectrum limiter (a simple low-pass or bandpass filter), a sample and hold circuit, a quantizer, and a digitizer. Quantization inevitably introduces errors, but errors can be reduced by:

Increasing the number of quantization levels
Introducing non-uniform quantization

- **Successive Approximation DACs**

Successive approximation ADCs convert analog signals to digital data by comparing the incoming analog signal with a reconstructed *guess* from a DAC, which sequentially doubles each guess on each clock cycle. The feedback in a Sigma Delta ADC is created from averaging a 1-bit DAC sampled at a very high rate. Sigma Delta ADCs are widely used because of their reasonable pricing and digital reliability.

- **Voltage Multiplying DACs**

Multiplying DACs operate on the principle of producing a weighted sum analog signal, in which the sum is proportional to the digital input signal.

- **DSK Functional Blocks**

A DSK board has a serial in/out ADC/DAC, a serial port for external communications, and a boot-loader PROM.

REFERENCES

Bateman, A. and Yates, W. [1988]. *Digital Signal Processing Design*, Pitman Publishing, London, UK

Candy, James C. and Temes, Gabor C. (eds.) [1992]. *Oversampling Delta-Sigma Convertors*, IEEE Press, New York

Curtis, S. [March 1991]. "Bitstream Conversion," *Electronics World and Wireless World*, Vol 97, No 1661, pp. 205-208

Finck, R. [June 1989]. "High Performance Stereo Bit-Stream DAC With Digital Filter," *IEEE Transactions on Consumer Electronics*, Vol 35 No 4, pp. 793-796

Finck, R. and Schulze, W. [May 1990]. "Single Chip CD Decoder," *IEEE Transactions on Consumer Electronics*, Vol 36, No 2, pp. 89-91

Horowitz, P. and Hill, W. [1984]. *The Art of Electronics*, Cambridge University Press, Cambridge, UK

Inose, H., Yasuda, Y. and Marakami, J. [1962]. "A Telemetering System by Code Modulation," Delta-Sigma *IRE Transactions on Space, Electronics and Telemetry*, Set-8, pp. 204-209

Jeri, Abdul J. [1977]. "The Shannon Sampling Theorem - Its Various Extensions and Applications: A *Proceedings of the IEEE*, Vol 65, No 11, pp. 1565-1593, November 1977

Koch, R. et al [Dec 1986]. "A 12-bit Sigma-Delta Analog-to-Digital Convertor With a 15 MHz Clock Rate," *IEEE Journal of Solid State Circuits*, Vol SC-21, No 6, pp. 1003-1010

Lynn, P. A. and Fuerst, W. [1990]. *Introductory Digital Signal Processing*, John Wiley and Sons Ltd., UK

Matsuya, Y. et al [Dec 1987]. "A 16-bit Oversampling A-to-D Conversion Technology Using Triple *IEEE Journal of Solid State Circuits*, Vol SC-22, No 6, pp. 921-929

Naus, P. J. A et al [June 1987]. " A CMOS 16-bit D/A Convertor for Digital Audio," *IEEE Journal of Solid State Circuits*, Vol SC-22, No 2

Nyquist, H. [1928]. "Certain Topics in Telegraph Transmission Theory," *AIEE Transactions*, pp. 617-644

Oppenheim, A. V. and Schafer, R. W. [1975 and 1988]. *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ

Park, S. [1990]. *Principles of Sigma-Delta Modulation for Analog-to-Digital Convertors*, Motorola Inc.

Rebeschini, M. et al [1989]. "A High-Resolution CMOS Sigma-Delta A/D Convertor with 320 kHz Output *Proceedings of ISCAS*, pp. 246-249

Shannon, C. E. [1949]. "Communications in the Presence of Noise," *Proceedings of the IRE*, Vol 37, pp. 10-21, January 1949

Steele, R. [1991]. *Delta Modulation Systems*, Pentech Press, London

Stewart, R. W. [1991]. "Digital Signal Processing: Technology and Marketing for Audio Systems," *IEE Colloquium on Digital Audio Signal Processing*, Digest No 07, pp. 5/1-6

Thompson, C. D. [May 1989]. "A VLSI Sigma-Delta A/D Convertor For Audio and Signal Processing *Proceedings of ICASSP*, Glasgow, UK

Welland, D. R. et al [Nov 1988]. "A Stereo 16-bit Sigma-Delta A/D Convertor For Digital Audio," *Proceedings of the 85th Convention of the Audio Engineering Society*, Vol 2724, H-12, California

Zaks, R. [1981]. *From Chips to Systems*, Sybex, California