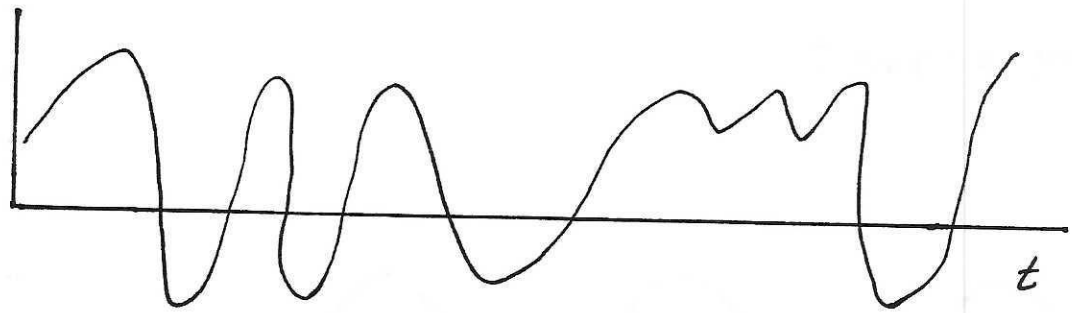
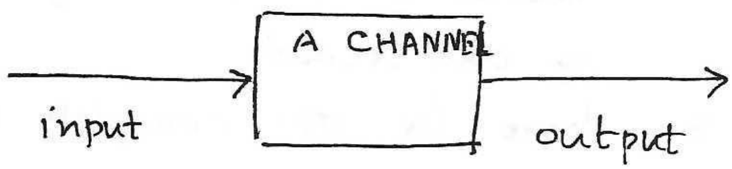
BASICS OF ANALOG SIGNALS



A signal is function of time $x(t)$



Interested in the reponse to an input.

Linear System: response is linear

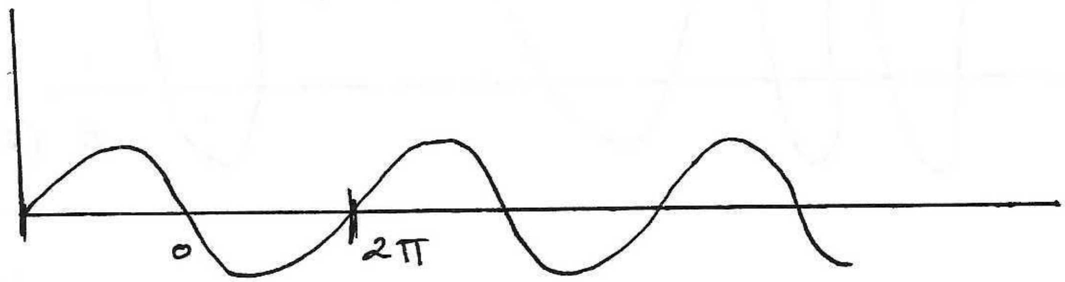
$$\left.\begin{array}{l} V_1(t) \longrightarrow V_2(t) \\ U_1(t) \longrightarrow U_2(t) \end{array}\right\} \therefore A V_1 + B U_1 \longrightarrow A V_2 + B U_2$$

Many circuits & idealized channels are linear in their reponse.

A sinusoidal signal input to a linear system has the property:

$$A\cos(\omega t) + \varphi_1 \longrightarrow B\cos(\omega t) + \varphi_2$$

$x = \sin t$



Frequency f $\equiv$ number of complete cycles in one second.

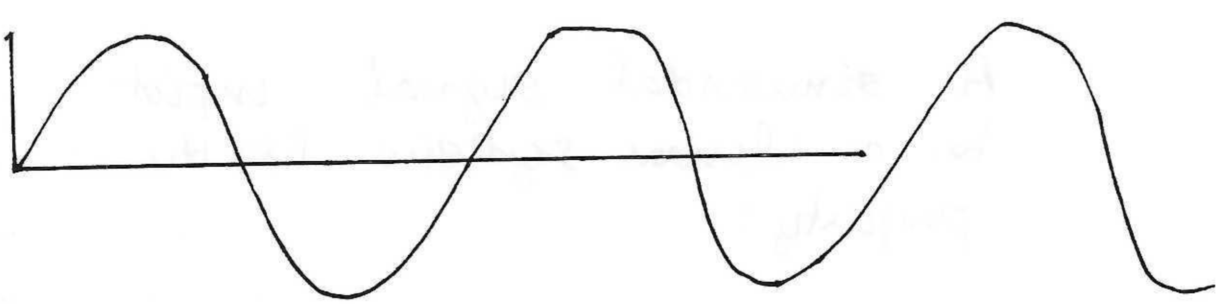
Period $\equiv$ time for one complete cycle.

$x = \sin t$: period $= 2\pi$

$\therefore$ f = frequency $= 1/2\pi$ $=$ 1/period Hz

$x = \sin(2\pi f t)$

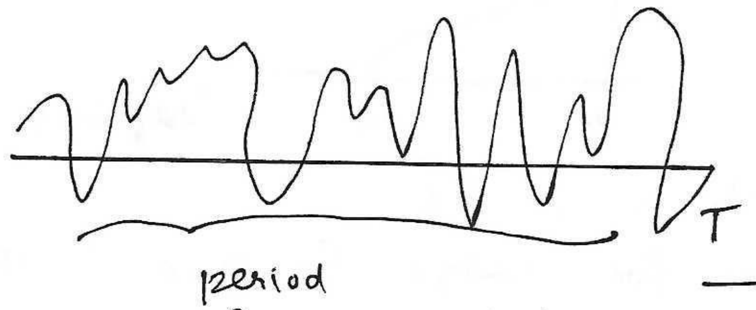
Let $f = 5$ $\quad x = \sin(2\pi \cdot 5 \cdot t)$



$t = 1/5 \Rightarrow$ full period $\therefore$
frequency $=$ 5 Hz.

similarly for $\cos(2\pi(4000)t)$
frequency $=$ 4000 Hz or 4 kHz.

Consider a "complicated" periodic signal of period T



fundamental frequency = $1/T$

Remarkably:

We can represent a waveform $x(t)$ over the $-T/2$ to $T/2$ as

$$x(t) = \sum_{f=0}^{\infty} A_n \cos(2\pi f t/T) + B_n \sin(2\pi f t/T)$$

This works for (1) periodic functions
[(2) waveforms that are non-zero over a finite time] ?

Can also use

$$x(t) = \sum_{k} A_k \cos(2\pi k f_0 t + \theta_k)$$

From Signal $x(t)$
to Frequencis & Power

$B_0 = 0$

$$A_0 = \frac{1}{T} \int_{-T/2}^{T/2} x(t)\, dt$$

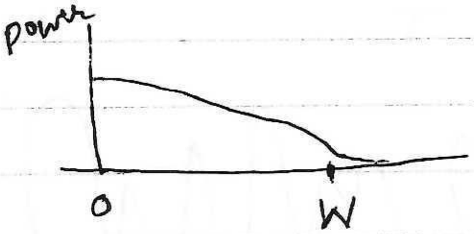
$$n>0 \quad A_n = \frac{1}{2T} \int_{-T/2}^{T/2} x(t) \cos\left(\frac{2\pi n}{T} t\right) dt$$

$$n>0 \quad B_n = \frac{1}{2T} \int_{-T/2}^{T/2} x(t) \sin\left(\frac{2\pi n t}{T}\right) dt$$

$$x(t) = A_0 + \sum_{k=1}^{\infty} A_n \cos\left(\frac{2\pi n t}{T}\right) + B_n \sin(\;)$$

Generalized Fourier Transform.

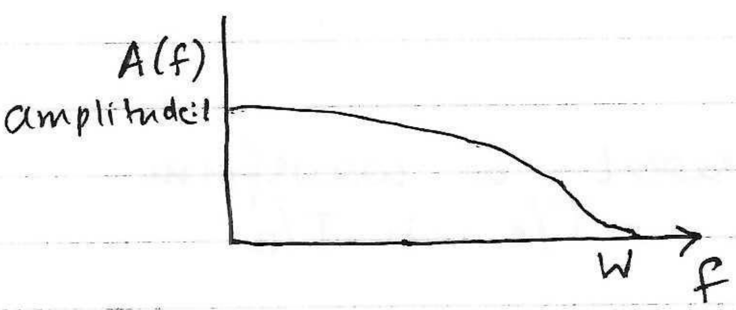
Bandwidth of a signal
Range of frequencies containing non-negligible power or amplitude.

Example 

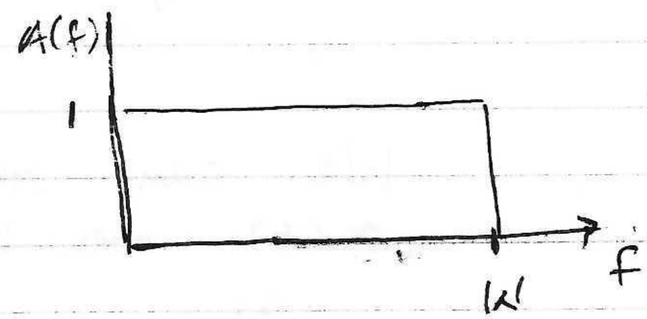
The frequency spectrum

Bandwidth is W Hz.
Similarly for range f_1 to f_2 it is $W = f_2 - f_1$.

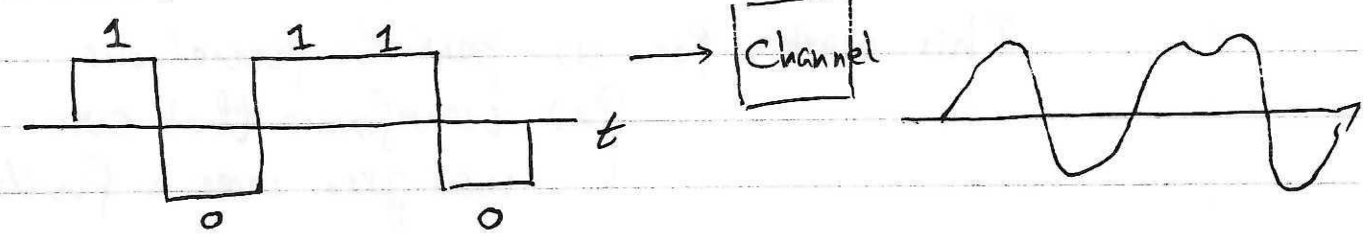
Low pass channel



ideal low pass



Thus typical response of a channel degrades the signal
Signal as pulses



Trying to get short pulse 

Need higher bandwidths

Shannon Capacity Theorem

Maximum transmission capacity of a channel of bandwidth W and signal to noise ratio SNR is:

$$C = W \log_2 (1 + SNR)$$

in bits/second.

$$SNR = \frac{\text{average signal power}}{\text{average noise power}}$$

Example: on a good telephone line it can be 10,000.

Telephone channel bandwidth = 3.4 kHz

$$\Rightarrow \quad C = 3.4 \times 10^3 \log_2 (1 + 10{,}000)$$
$$= 45.2 \text{ kbits/sec.}$$

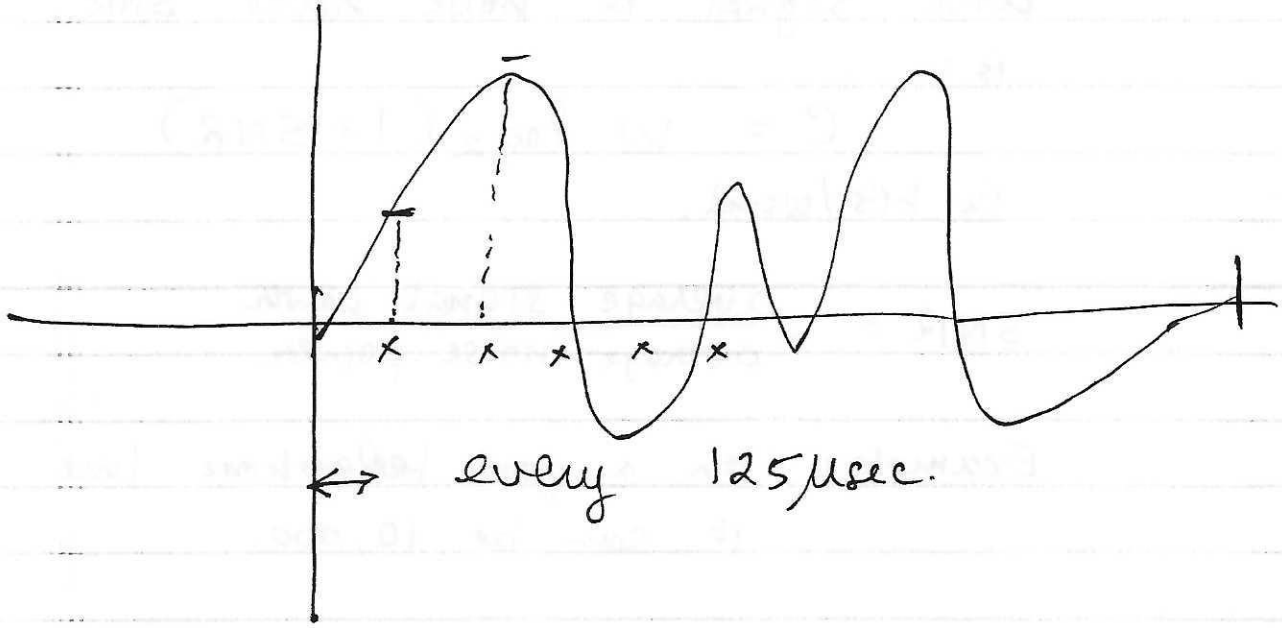
Note SNR sometimes given in decibels

$$SNR\,(db) = 10 \log_{10} SNR.$$

Example for SNR = 10,000 we get $10 \times 4 = 40$ decibels

Nyquist — Need to sample $2 \times W$ to accurately reproduce signal

Sampling an analogue signal to digitize it



Sample Points

For example

~~Sapu~~ Sample 8000 times a second for a 4 kHz voice channel

MP3 → want to get high quality audio range

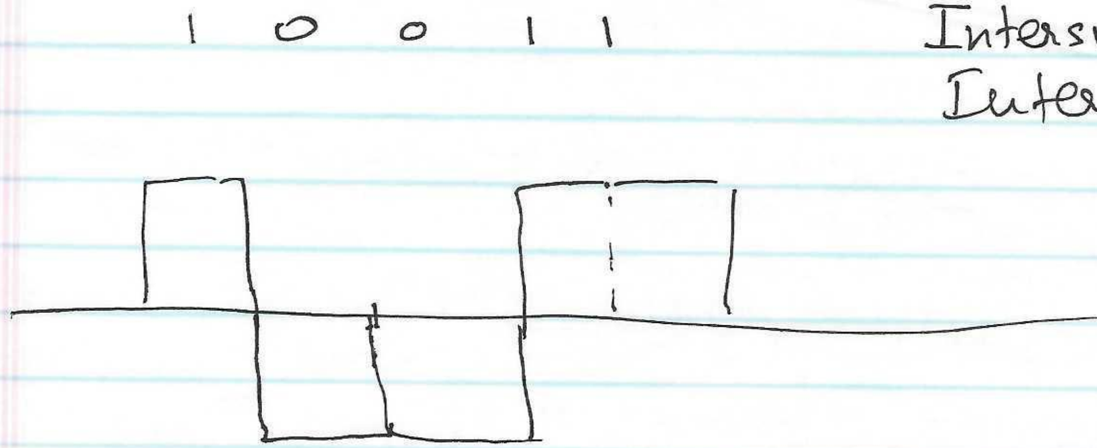
∴ 44,000 samples/second.

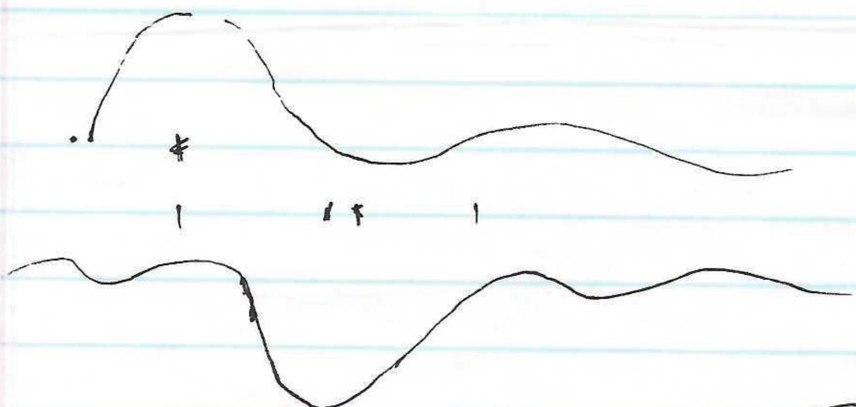
$44{,}000 \times 2$ ~~bits~~ channels $\times$ 16 bits/sample

$\approx$ 1.4 Mbits/sec

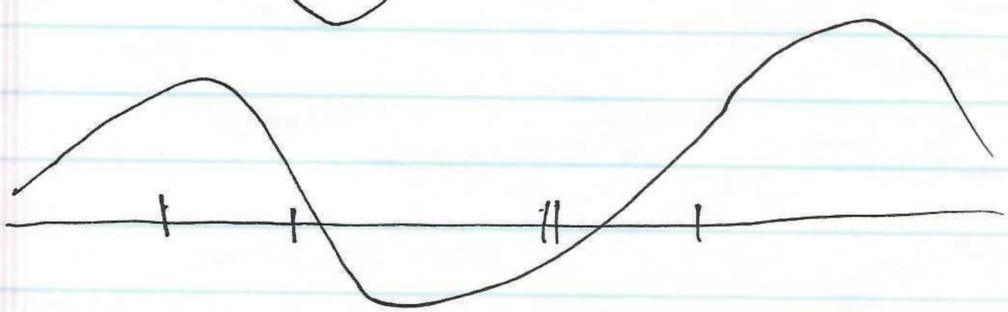
Error Detection & Correction

Intersymbol Interference





pulse
Square wave passed through filter



sum of pulse.

Can't exactly tell what the sample value should be accurately

Detecting / Not Detecting Error

"Random bit error model" n bits

$1 \quad 0 \quad 1 \quad 1 \quad \underline{\cancel{0}} \quad 1 \quad \underline{\cancel{1}}$ underlined errors

$$P(k \text{ errors}) = \binom{n}{k} p^k (1-p)^{n-k}$$

Binomial Theorem

$$(x+y)^n = \underbrace{(x+y)(x+y)\cdots(x+y)}_{}$$ n times

Example $(x+y)^3 = (x+y)(x+y)(x+y)$
$= x^3 + 3x^2y + 3xy^2 + y^3$

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k$$

Random Error Model: $x = p$ probability of error
$y = 1-p$ " " of no error

$P(k \text{ errors})$ as above.

$1 \quad 0 \quad 1 \quad 1 \quad \cancel{0} \quad 1 \quad \cancel{1}$

$p \cdot p = p^2$ specific two errors

Using Binomial we get a prob distribution.
Note summing all possibilities is summing
0 errors, 1 err, 2 errs, ... n errors

Note: $(1-p)^j = \underbrace{(1-p)(1-p)\cdots(1-p)}_{j \text{ times}}$

$= 1 - jp + \binom{j}{2} p^2 \ldots$

If $p = 10^{-6}$
$p^2 = 10^{-12}$

Two dimensional parity check.

↓ info ↓ check · column

1	1	0	1	1
0	0	1	1	0
1	0	1	1	1
1	1	0	0	0
1	0	0	1	0

→ check column.

How to detect 1, 2, or 3 errors. ?

Do some examples

How to correct 1 error ?

When do you know you have 1 or 2 or 3 ... errors?
(you don't.)

1 Error Row / Column detects

2 Errors Same row or different rows

3 Errors Same row, detect -

Two different rows detect -

1 Error Correction Correcting Error ? row detects
colum detects

Two's Complement & Ones' Complement

Two's Complement

Example with 8 bits

	Positive #s Leading bit = 0		Negative #s	Leading bit = 1
0	00000000	128	1000 0000	−128
1	00000001	129	10000001	−127
	⋮		⋮	
127	01111111	255	11111111	−1

Note. Negative #'s can be viewed as an 8-bit positive number with a leading bit of 1 as a mod 256 number For example 129 mod 256 = −127 mod 256. To get a negative # complement (invert bits) and add 1.

Thus: In this representation, addition and subtraction is straight forward in a 16-bit regsiter. Just look at results mod 256 and add as usual.

Internet CheckSum

Why: <u>Ones' Complement</u> (Better error Detecting.)

Example with 8 bits

	Positive #s leading bit = 0	Negative #s	leading bit = 1
0	0000 0000	10000000	−127
1	0000 0001	1000 0001	−126
	⋮	⋮	
127	01111111	1111 1110	−1
		11111111	0

Note Invert bits to get a negative #. Equivalent to mod 255 for adding & subtracting. Adding somewhat more complex.

Internet Checksum. ①

① summing up the headers

16 bits	16 bits	16 bits

32 bit

SUM | | | |

$$(b_1 + b_2 + b_3 + \cdots b_L) \bmod (2^{16}-1)$$

4 bit example. 4 bit $\boxed{1011}$ $\boxed{0001}$ $\boxed{0101}$ $\boxed{1001}$

	4 bit.	4 bit
sum = 26	0001	1010
	$x_3 x_2 x_1 x_0$	$a_3 a_2 a_1 a_0$

$\triangleq$.

1011	11
0001	1
1100	12
0101	5
10001	17
1001	9
11010	26

$$(x_3 \cdot 2^7 + x_2 \cdot 2^6 + x_1 \cdot 2^5 + x_0 2^4)$$
$$+ \quad a_3 \cdot 2^3 + a_2 2^2 + a_1 \cdot 2 + a_0$$

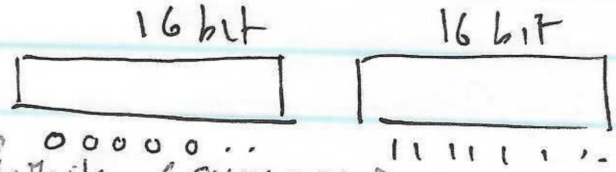
We want the result $\bmod (2^4-1)$

$(x_3 \cdots) \bmod 2^4-1$
$+ (\quad) \bmod 2^4-1 \quad = \quad a_3 a_2 a_1 a_0$

$$(x_3 \cdot 2^3 + x_2 \cdot 2^2 + x_1 2 + x_0) \cdot 2^4 \bmod (2^4-1)$$
$$= (x_3 x_2 x_1 x_0) \cdot 1 \qquad [\text{Note } 2^4 \bmod (2^4-1) = 1]$$

$\therefore$ right shift by 4 bits & add.
(need to keep doing it) until upper 0000.

(2)



fold 32 bit into 16 bit sum

mask
```
while (sum>>16)
sum = (sum & 0xffff)
      + (sum >> 16);
```
lower 16 bits of sum

This is the result we want.

$b_1 \cdots b_L \bmod 2^{16}-1 = x$

Checksum is by definition $-x$

In Example $x = 1011$
$= 11$ (decimal) (not this already mod (2^4-1))

What is $-x$

Claim: it is the 1's complement.

Why: $-x \bmod 15$

$= (15-x) \bmod 15$

To get $15-x$ need to simply do

$1111 - x$ is 1's complement.

$\therefore \quad -x = 0100 = 4.$