

## L01 - Communication Basics (v6.0)

# Communication Basics

Serialization, Bit Synchronization, Physical Aspects of Transmission,  
Transmission Frame, Frame Synchronization, Error Control

## L01 - Communication Basics (v6.0)

### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

## L01 - Communication Basics (v6.0)

## Information

- **What is information?**
  - Represented and carried by **symbols**
  - **Recognized** by receiver (hopefully)
  - *Interpretation* is the key...



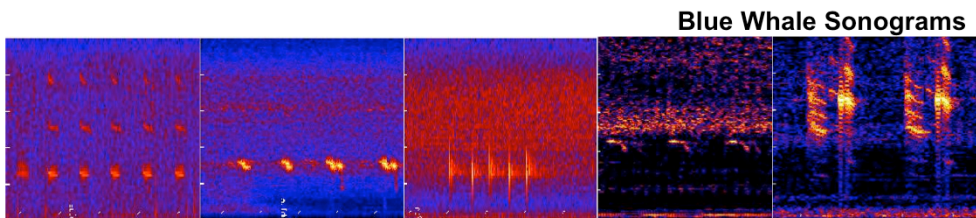
What is information? This question may sound quite easy but think a bit about it. Obviously we need symbols to represent information. But these symbols must also be recognized as symbols by the receiver. In fact, philosophical considerations conclude that information can only be defined through a receiver. The same problem is with art. What is art? Several decades and centuries had their own definitions. Today most critics use a general definition: art can only be defined in context with the viewer.

In the following chapters – throughout the whole data communication – we will deal with symbols representing information. A symbol is not a 0 or a 1. But this binary information can be represented by symbols. Be patient...

**L01 - Communication Basics (v6.0)**

## Symbols

- **Symbols (may) represent information**
  - Voice patterns (Speech)
  - Sign language, Pictograms 🌐 ♿ 🔊 🗣️
  - Scripture
  - Voltage and current levels
  - Light pulses



© 2016, D.I. Lindner / D.I. Haas

Communication Basics, v6.0

4

What is a good information source? From a theoretical point of view a random pattern is the best because you'll never know what comes next. On the other hand, if you receive a continuous stream of the same symbol this would be boring. More than boring: there is no information in it, because you can predict what comes next! From this we conclude that a sophisticated coding – representing the information as efficient as possible using symbols – is a critical step during the communication process.

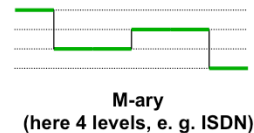
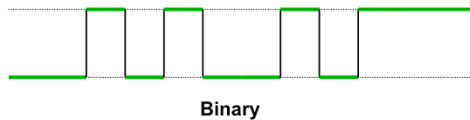
Throughout these chapters we will mainly deal with symbols such as voltage or electrical current levels or light pulses.

Look at the Blue Whale Sonograms. The x-axis represents time, the y-axis frequency and the color represents power density. This communication pattern is very complex (those of dolphins is even more complex). It is known that each herd has their own traditional hymn. And: they like to communicate!

## L01 - Communication Basics (v6.0)

## Representation of Symbols for Information Processing, Storage and Exchange

- In the context of computer systems and data communication
- Discrete levels = "Digital"
  - Resistant against noise
- How many levels?
  - Binary (easiest)
    - Bit (binary digit), values 0 and 1
  - M-ary: *More information per time unit!*



What symbols do we encounter on wire? Digital binary symbols are commonly known and widely in use. Why? Consider information transmissions in groups of symbols (for example the group of 8 binary symbols is called a byte). We have two parameters: the number base  $B$  and the group order  $C$ . If you calculate the "costs" that you get for arbitrary variations of  $B$  and  $C$ , and if we assume a linear progress (so that  $\text{cost} = k \times B \times C$ ) then for any given (constant) cost the perfect base would be  $B=e$ , that is  $B=2.7182\dots$

In other words: the perfect base is a number between 2 and 3. The technical easiest solution is to use  $B=2$ . Note that these considerations assume a linear cost progression.

In many cases we pay the price of higher efforts and use a larger base. This leads us to m-ary symbols and later to PAM and QAM.

Discrete / digital levels are physically represented:

Electrical transmission systems (using copper e.g. coax-, twisted-pair cables)

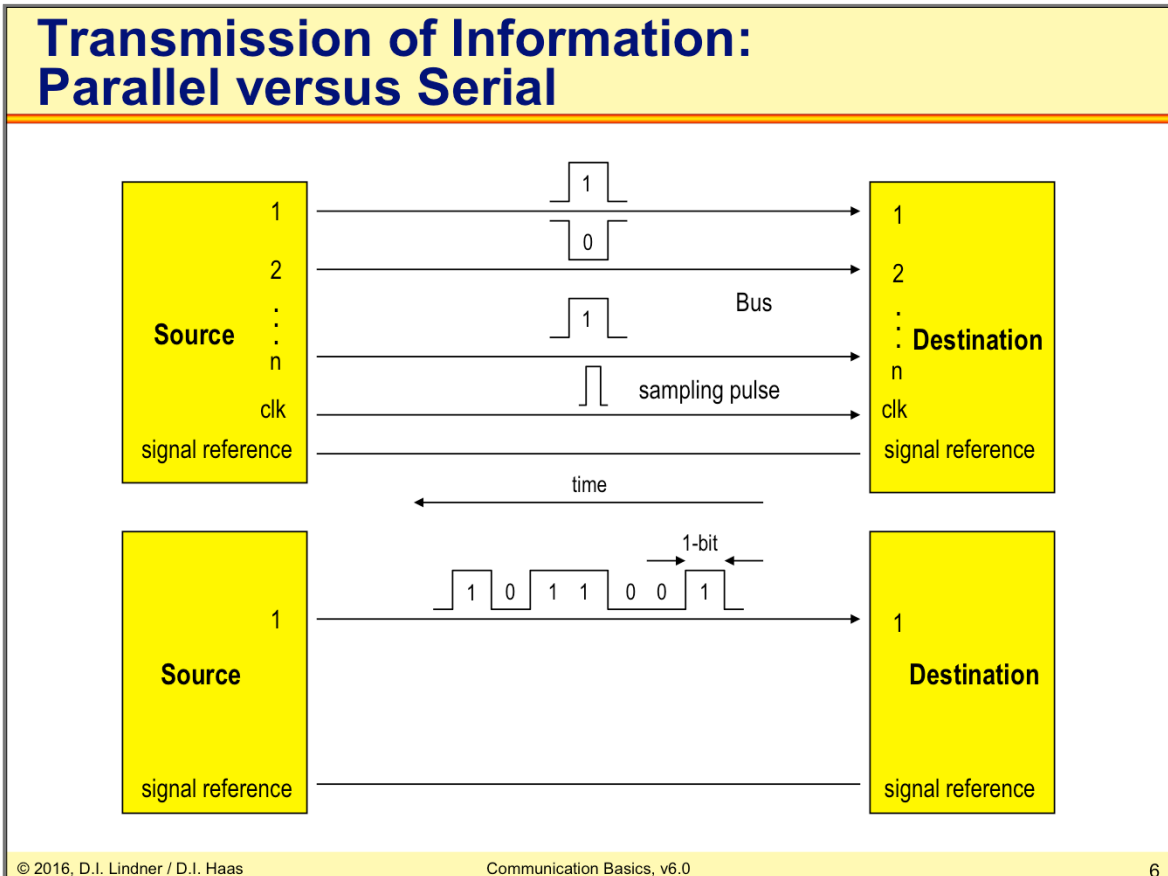
voltage level

current level

Optical transmission systems (using fiber e.g. multi-mode, single-mode fiber)

light on / off

**L01 - Communication Basics (v6.0)**



Within a computer system -> Parallel transfer mode

A data word (8-bit, 16-bit, ...) is transferred at the same time using several parallel lines called "Bus"

Data-bus for transferring data words

Address-bus for addressing memory location

Control-bus for signaling direction of transfer (read/write), clock (clk.), interrupt,

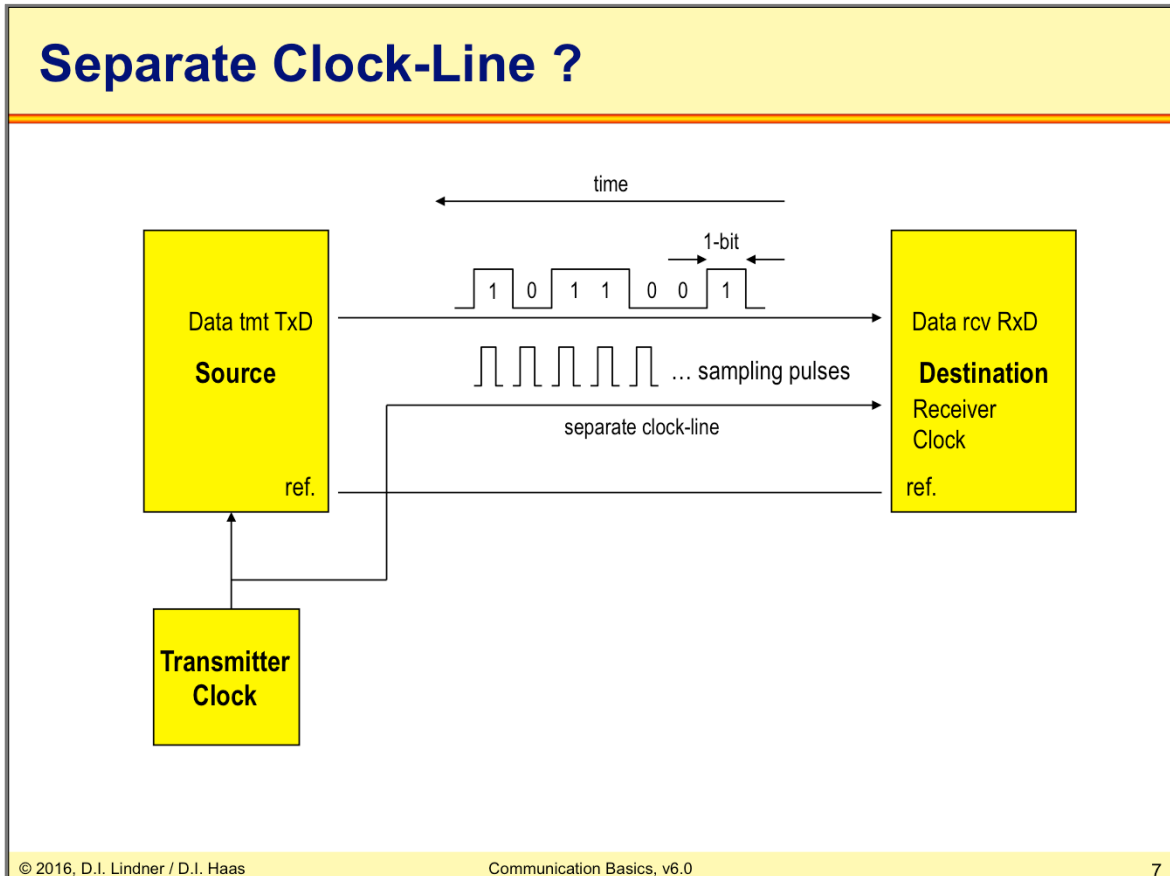
...

Between computer systems -> Bit-serial transmission

Bits are transferred bit by bit using a single line.

Basic transmission technique used in data communication networks

## L01 - Communication Basics (v6.0)



What does serial transmission mean?

Bits are transmitted on one physical line a single bit at a time using a constant time interval (bit-cell) for each bit.

The receiver of a serial transmission line must sample bits at the right time in order to interpret the bit pattern correctly .

Receiver clock must be synchronized to transmitter clock.

One way is to use a separate clock line as it is done by parallel transmission technique.

In case of WAN a separate clock line is not acceptable for reasons of cost.

Therefore bit (clock) synchronization techniques are used.

**L01 - Communication Basics (v6.0)**

## Parallel versus Serial

- **Parallel transmission**
  - Multiple data wires (fast)
  - **Explicit clocking wire**
  - Simple synchronization but not cost-effective
  - Only useful for small distances
  
- **Serial transmission**
  - Only one wire (-pair)
  - **No clocking wire**
  - **Most important** for data communication for long distances
  - Bit (clock) synchronization is necessary

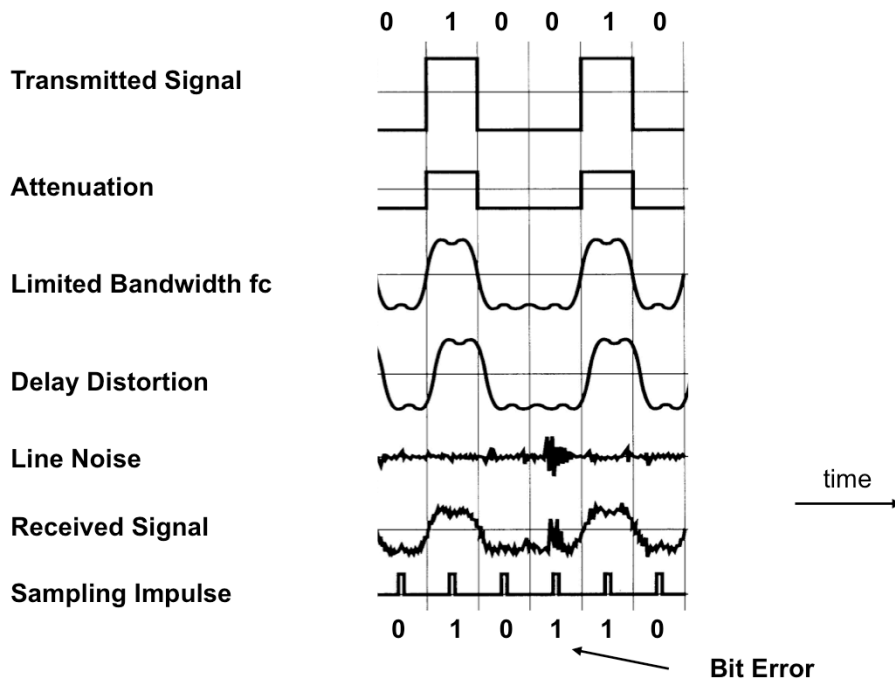
In case of parallel transmissions there is always a dedicated clock line. This is a very comfortable synchronization method. A symbol pattern on the data lines should be sampled by the receiver each time a clock pulse is observed on the clock line. But unfortunately, parallel transmissions are too costly on long links. In LAN and WAN data communication there are practically no parallel lines.

The most important transmission technique is the serial. Data is transmitted over a single fiber or wire-pair (or electromagnetic wave). There is no clock line. How do we synchronize sender and receiver?



L01 - Communication Basics (v6.0)

# What Happens To A Signal On The Wire?



There are no digital (rectangle) signals on a physical line. Physical signals are attenuated, bandwidth limited, distorted and even background noise is put on them.

## L01 - Communication Basics (v6.0)

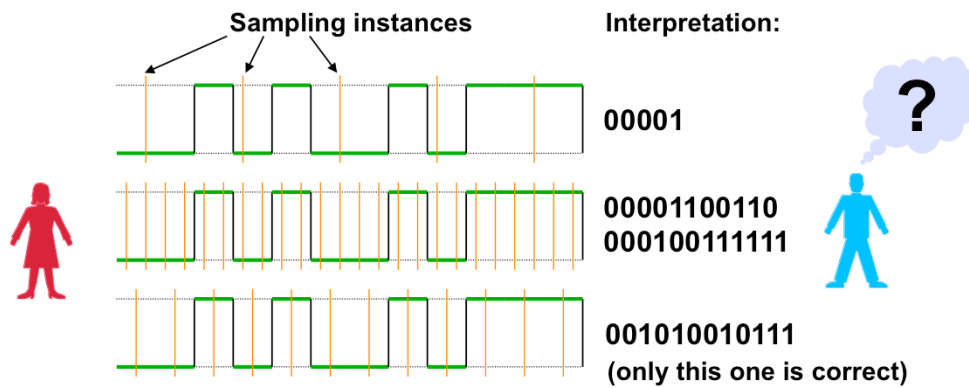
### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

## L01 - Communication Basics (v6.0)

## Synchronization

- **Sender sends symbol after symbol...**
- **When** should receiver pick the signal samples?  
 – => Receiver must **sync** with sender's clock !

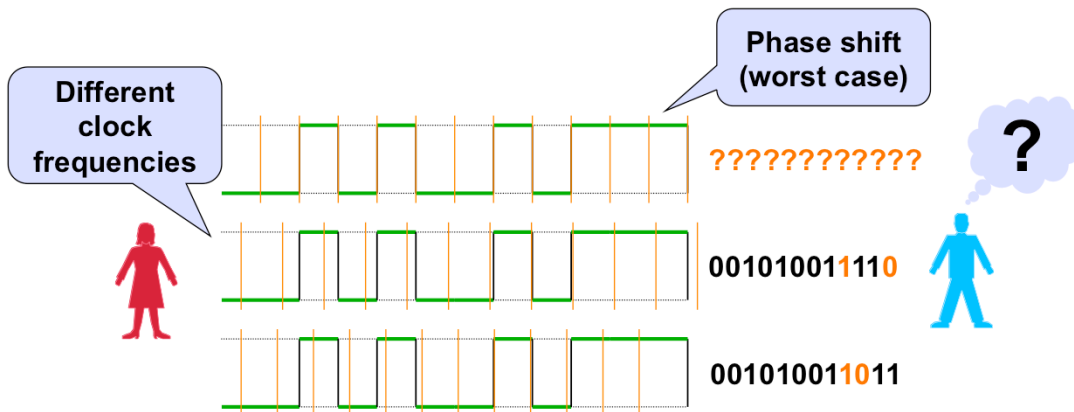


One of the most important issues among communication is that of synchronization. Nature forbids absolute synchronization of clocks. Suppose you are a receiver and you see alternating voltage levels on your receiving interface. If you had no idea about the sending clock then you would never be able to interpret the symbols correctly. When do you make a sample?

## L01 - Communication Basics (v6.0)

## Synchronization

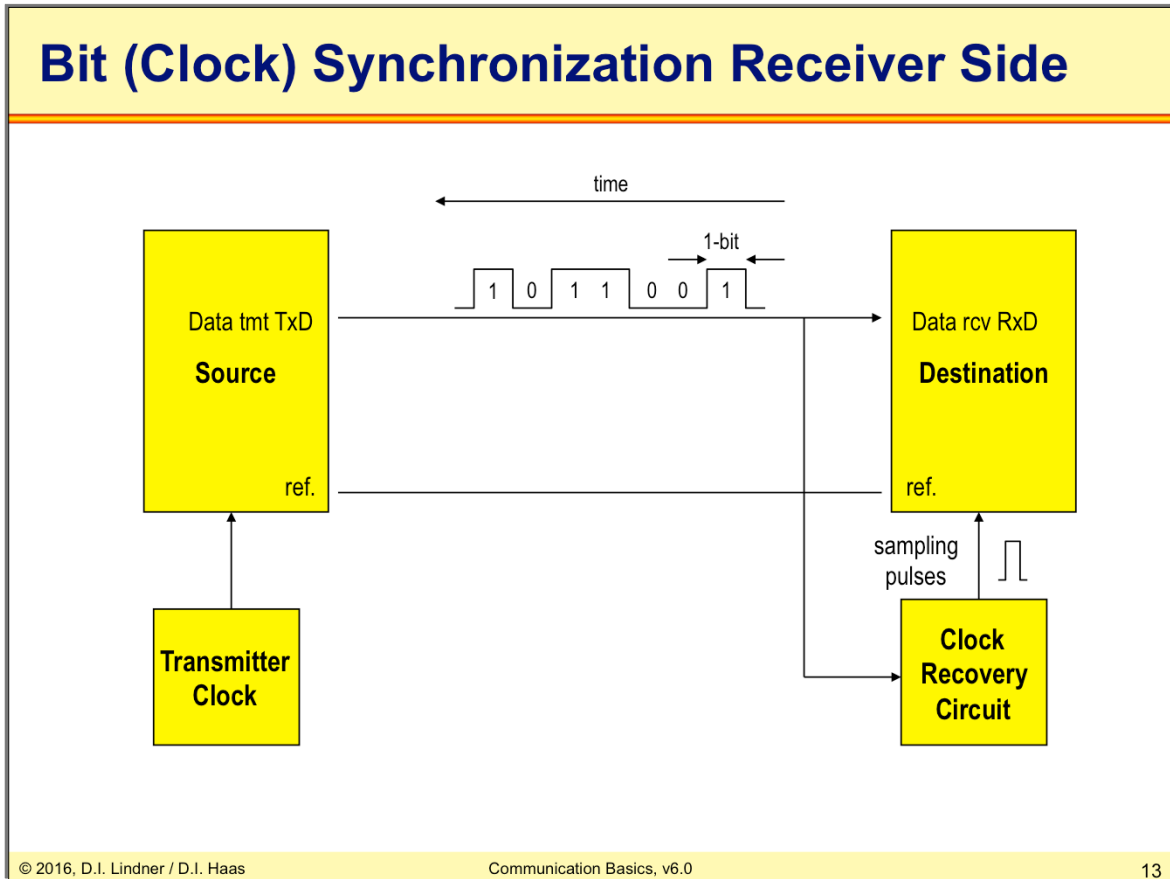
- In reality, two independent clocks are **NEVER** precisely synchronous
  - We always have a frequency shift
  - But we must also care for phase shifts



So we must assume that the receiver's clock is approximately identical to the sender's clock. At least we must deal with small phase and frequency gaps. As you can see in the slide above, we still cannot be sure when to make samples.

What we need is some kind of synchronization method.

## L01 - Communication Basics (v6.0)



Bit synchronization deals with synchronization of the receiver clock to the transmitter clock for serial transmission

Principle of bit synchronization:

Signal changes are used by the receiver for clock recovery

Recovered clock generate pulses which are used to sample the bit stream to decide if 0 or 1

Sampling should occur in the center of bit-cell because signal attenuation, bandwidth limitation, delay distortion will modify signal form

Depending on duration of bit synchronization we can differentiate between

asynchronous and synchronous transmission method

## L01 - Communication Basics (v6.0)

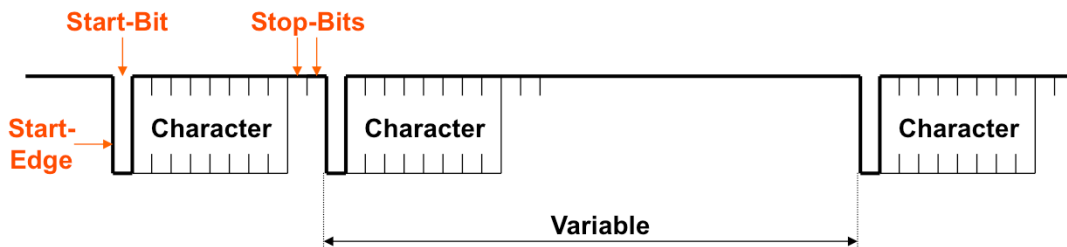
### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

## L01 - Communication Basics (v6.0)

## Asynchronous Transmission

- **Independent clocks at transmitter and receiver**
  - Oversampling at the receiver: Much faster than bit rate
- **Only phase is synchronized**
  - Using Start-bits and Stop-bits
  - Variable intervals between characters
  - Synchronicity only during transmission of a data word
- **Inefficient**
  - 8 bits data need additional 3 bits for bit synchronization



One synchronization method is the asynchronous transmission. Actually this method cannot provide real synchronization (hence the name) but at least a short-time quasi-synchronization is possible. The idea is to frame data symbols using start and stop symbols (lets sloppy call them start- and stop bits). Using oversampling, the receiver is able to get a sample approximately in the middle of each bit – but only for short bit sequences.

Asynchronous transmission is typically found in older character-oriented technologies.

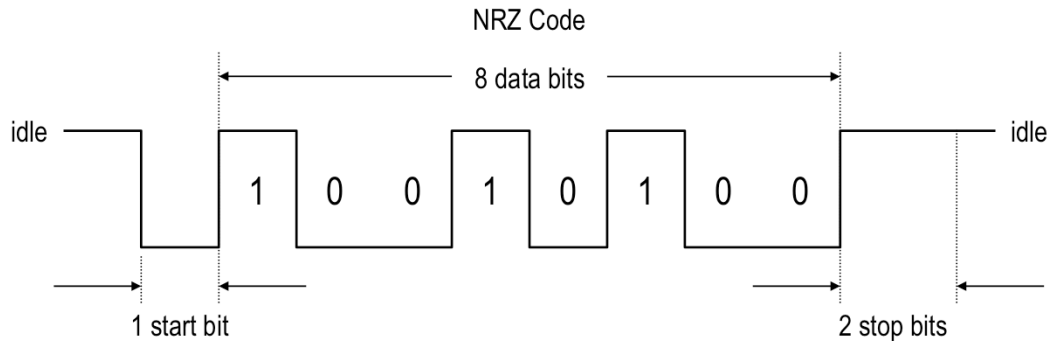
Example: RS-232C / V.24-V.28 (com1 on your PC)

Bit synchronization lasts only for the time needed to transmit one data word.

Data words could be sent independently and are synchronized independently from each other, therefore we call it asynchronous

## L01 - Communication Basics (v6.0)

## Data Word Framing by Start / Stop Bits



- NRZ (non return to zero) describes the encoding of bits where level 1 refers to logical 1 and level 0 refers to logical 0
- Idle .... no data is transmitted, no change of signal level

Technique of start/stop bit is used by asynchronous transmission:

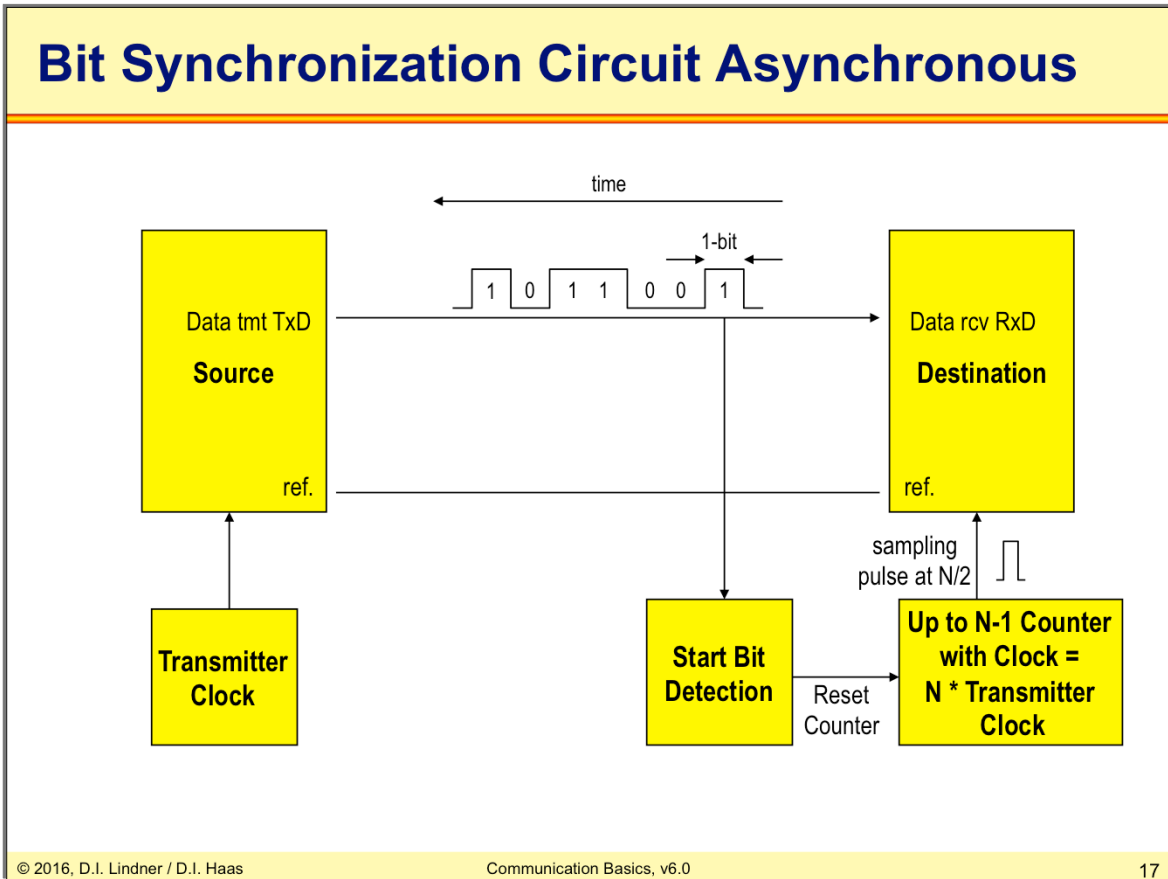
Start bit is indicated by a binary change from 1 to 0 and synchronizes the following 8-bit data word with over sampling

Stop bit(s) are one or two bits being binary 1. They make sure that every following start bit is recognized correctly regardless of the transmitted data and force the line into idle state after transmission of one data word.

Idle line means there are no signal changes on the line.



**L01 - Communication Basics (v6.0)**



The slide shows a possible implementation of clock recovery circuit for asynchronous transmission.

## L01 - Communication Basics (v6.0)

### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

## L01 - Communication Basics (v6.0)

## Synchronous Transmission

- **Synchronized clocks**
  - Most important today!
  - Phase and Frequency synchronized
- **Receiver uses a Phased Locked Loop (PLL) control circuit**
  - Requires frequent signal changes
  - => *Coding* or *Scrambling* of data necessary to avoid long sequences without signal changes
    - Encoding / Scrambling at the sender side
    - Decoding / Descrambling at the receiver side
- **Continuous data stream possible**
  - Large frames possible (theoretically endless)
  - Receiver remains synchronized
  - Typically each frame starts with a short "*training sequence*" aka "*preamble*" for the PLL to lock in (e. g. 64 bits)

The most important method is the synchronous transmission. Don not confuse this with synchronous multiplexing – we are still on the physical layer! Two things are necessary: a control circuit called Phased-Locked-Loop (PLL) and a signal that consists of frequent transitions. How do we ensure frequent transitions in our data stream? Two possibilities: coding and scrambling our data.

Synchronous transmission is found in most modern bit-oriented technologies.

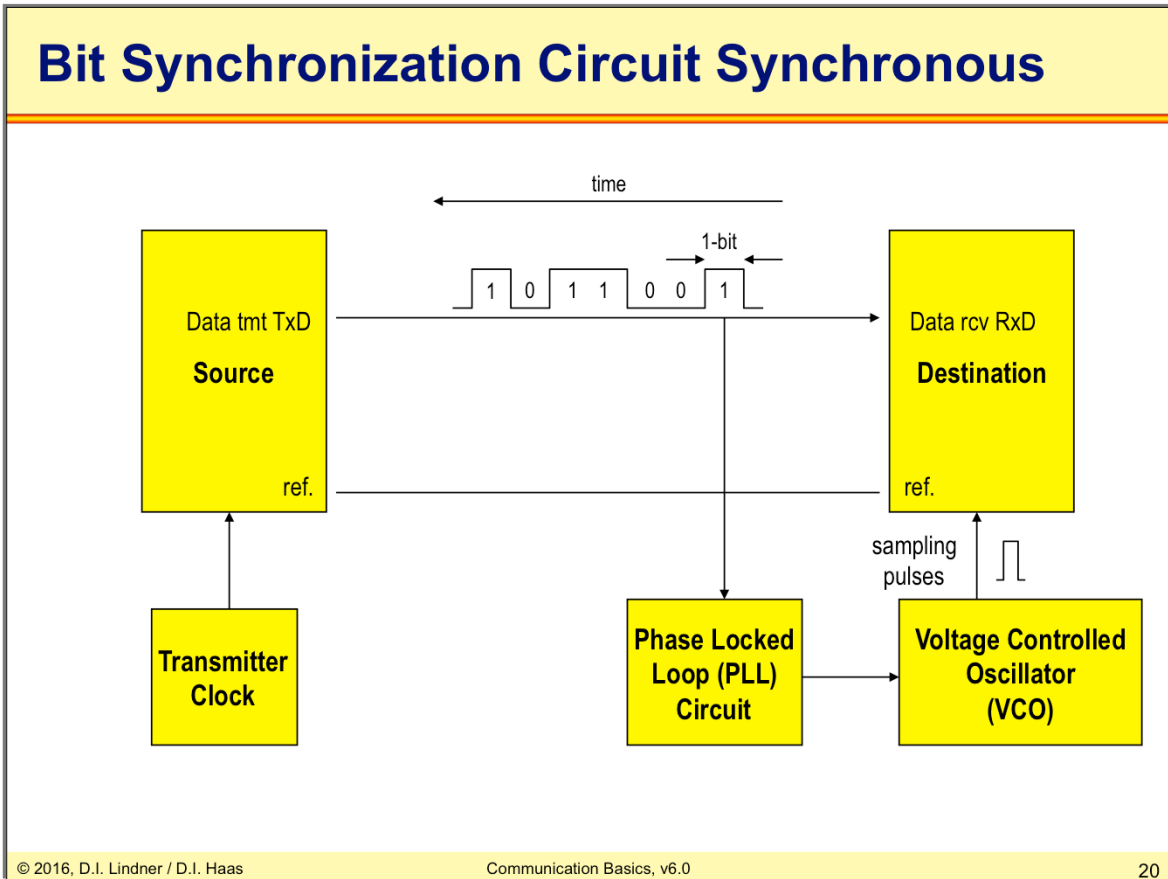
Bit synchronization lasts at least for the time to transport a block of data. Some implementations even keep the synchronization ongoing by sending periodic sync-bits in times no data has to be sent.

Requirement for this kind of transmission is sufficient changes of signal levels to enable clock recovery at the receiver.

Phased Locked Loop (PLL) technique is used to freeze the receiver clock in times where no signal changes are present on the line.

In contrast to asynchronous transmission bit overhead is reduced. Only at the beginning of a data block additional synchronization bits are necessary, later bit stream itself will keep bit synchronization going on.

**L01 - Communication Basics (v6.0)**



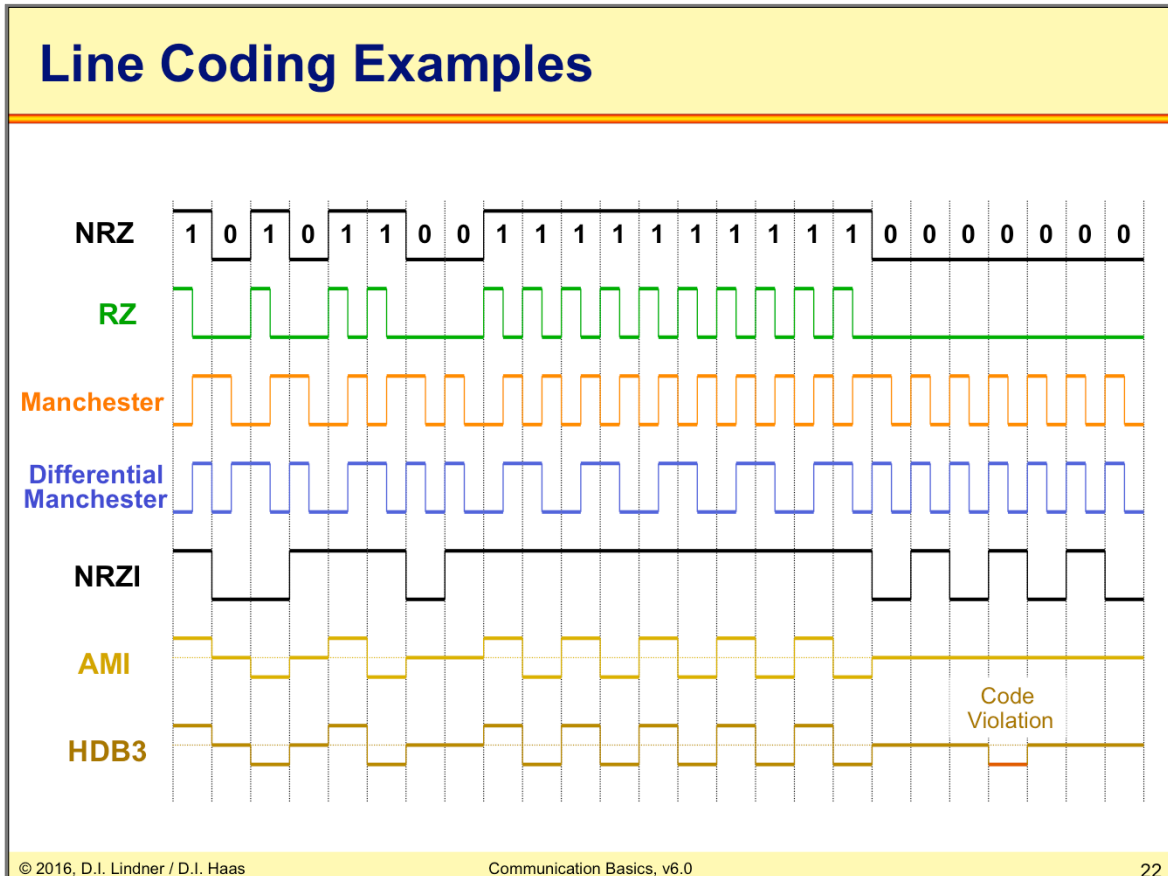
The slide shows a possible implementation of clock recovery circuit for synchronous transmission.

## L01 - Communication Basics (v6.0)

### Synchronous Transmission

- **Bit synchronization depends on sufficient signal changes within the bit stream**
  - For long series of 0s or 1s simple NRZ encoding is not able to provide this changes
- **Two basic methods are used to guarantee signal changes**
  - Encoding of bits that every bit contains a signal change
    - Manchester-code (Biphase code), Differential-Manchester-code, commonly used in a LANs
  - Encoding of bits in such a way that there are enough signal changes in the bit stream
    - NRZI (with bitstuffing), RZ and AMI (with scrambler)
    - HDB3 (with code violations), commonly used in a WANs

## L01 - Communication Basics (v6.0)



The trivial code is Non Return to Zero (NRZ) which is usually the human naive approach.

**RZ (Return to Zero):** Positive impulse (half bit length) describes a logical 1, logical 0 does not trigger any signal change. Scrambler prevents large numbers of 0's in bit stream. Bandwidth requirements are twice of NRZ. Has a dc component

RZ codes might also use a negative level for logical zeroes, a positive level for logical ones and a zero Volt level in between to return to. RZ is for example used in optical transmissions (simple modulation).

**Manchester:** Bit is divided into two half-bits. First half-bit is the complement of the data bit, second half-bit is identical to data bit. Change of signal level occurs in the center of each bit. Change from 1 to 0 describes a logical 0. Change from 0 to 1 describes a logical 1.

**Differential Manchester:** Logical 0 is defined by a signal change at the beginning and at the center of the bit. Change of signal only at the center identifies a logical 1. No signal change at the center of a bit can be used for code violation (J and K symbols)

Principle characteristics of Manchester and Differential Manchester codes: Bandwidth requirement is twice of NRZ. They have no or constant dc (direct current) component

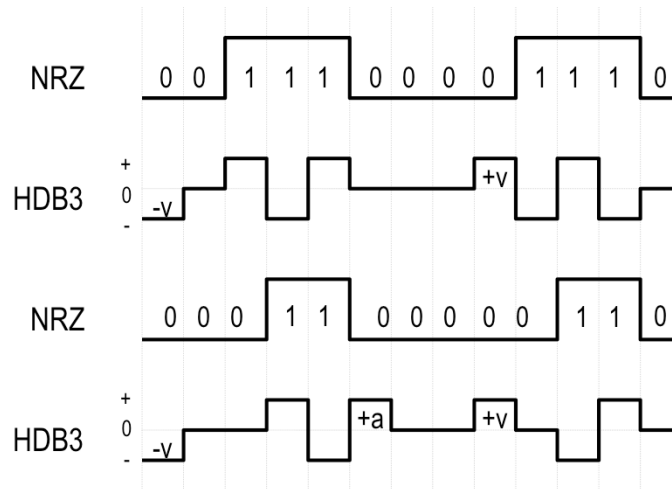
**NRZI (Non Return to Zero Inverted):** Logical 0 is defined by change of signal level at beginning of bit, logical 1 does not produce any change of signal. Bit stuffing prevents large numbers of 1's in bit stream. Bandwidth requirements are identical to NRZ. Has a dc component. NRZI codes either modulate for logical ones or zeros. In this slide we modulate the zeroes, that is each logical zero requires a transition at the beginning of the interval.

**AMI (Alternate Mark Inversion):** Three level encoding (+, 0, -). Pulses (length = 1 bit) with changing polarity describe logical 1's, no pulse characterizes a logical 0. Scrambler prevents large numbers of 0's in bit stream. Bandwidth requirements are identical to NRZ. Has no or constant dc component

Manchester is used with 10 Mbit Ethernet. Token Ring utilizes Differential Manchester. Telco backbones (PDH technology) use AMI (USA) or HDB3 (Europe). Of course there are many other coding styles.

**L01 - Communication Basics (v6.0)**

**HDB3 (High Density Bipolar 3) Code**



	polarity of last pulse	amount of pulses since last violation	
		odd	even
bit pattern	plus	0 0 0 +v	-A 0 0 -v
	minus	0 0 0 -v	+A 0 0 +v

Encoding Rules for HDB3:

Logical 1's are encoded using pulses with alternate polarity, a logical 0 never generates a pulse.

Exception in case of a continuous sequence of 0's:

Four 0's are encoded by a special pattern consisting of one or two impulses (A and V-bits)

V-bits are code violations, breaking the rule of alternating pulses

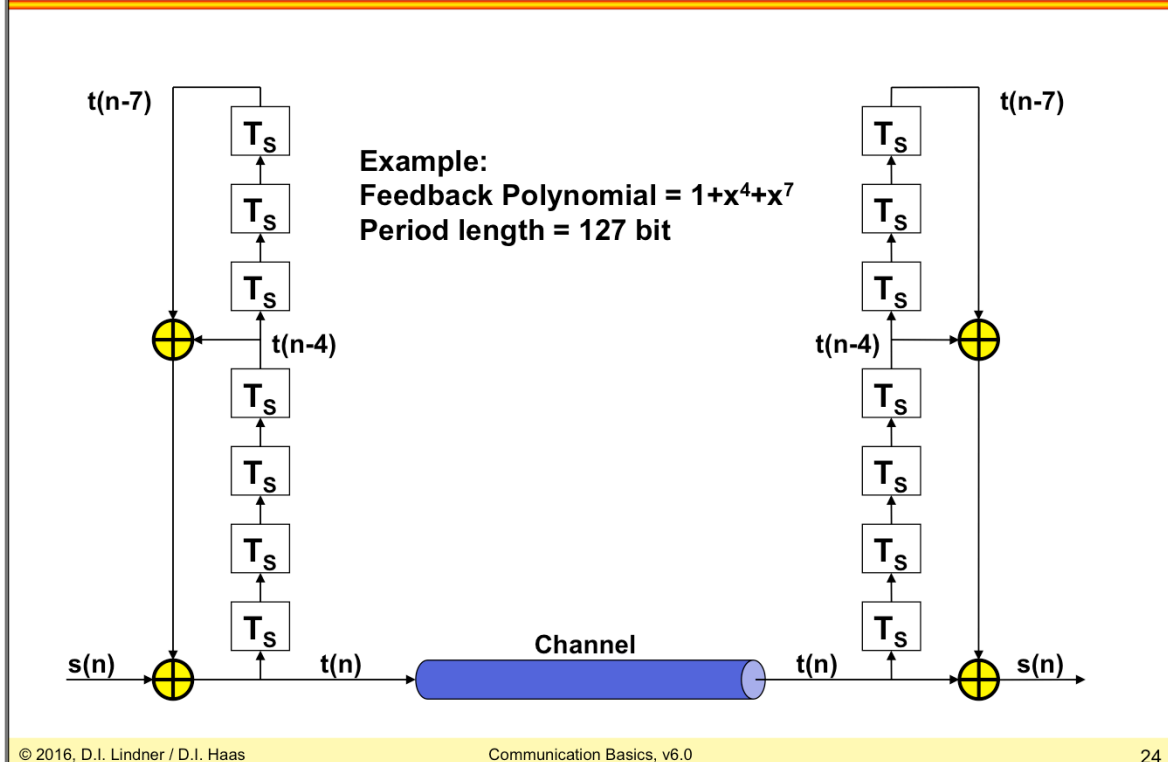
The following rule avoids DC portion using A- and V-bits

Bandwidth requirements are identical to NRZ

Has no or constant dc component

## L01 - Communication Basics (v6.0)

## How Does a Scrambler Circuit Look Like?



Another method to guarantee frequent transitions is scrambling. Scramblers are used with ATM, SONET/SDH for example.

The feedback polynomial above can be written as

$$t(n) = s(n) \text{ XOR } t(n-4) \text{ XOR } t(n-7)$$

The descrambler recalculates the original pattern with the same function (change  $s(n)$  with  $t(n)$ )

Period length =  $2^R - 1$ , where  $R$  is the number of shift registers

That is, even a single 1 on the input (and all registers set to 0) will produce a 127-bit sequence of pseudo random pattern.

This scrambler is used with 802.11b (Wireless LAN).



## L01 - Communication Basics (v6.0)

### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

## L01 - Communication Basics (v6.0)

## Theoretical Basis for Data Transmission

- **How can a digital signal be represented?**
  - Fourier analysis proves that any periodic function  $g(t)$  with period  $T$  can be constructed by summing a (infinite in case of rectangle pulses) number of sinus and cosines functions

$$g(t) = (1/2)c + \sum_{n=1}^{\infty} a_n \sin(2\pi nft) + \sum_{n=1}^{\infty} b_n \cos(2\pi nft)$$

- With  $f = 1/T$  and  $a_n$  and  $b_n$  as amplitudes of the  $n^{\text{th}}$  harmonics and  $c$  as the dc component
- Such a decomposition is called Fourier series

## Fourier Coefficients

- How can the values of  $c$ ,  $a_n$  and  $b_n$  be computed?

$$c = (2/T) \int_0^T g(t) dt$$

$$a_n = (2/T) \int_0^T g(t) \sin(2\pi nft) dt$$

$$b_n = (2/T) \int_0^T g(t) \cos(2\pi nft) dt$$

## L01 - Communication Basics (v6.0)

### Imperfect Real Data Transmission

- 1. No transmission systems can transmit signals without losing some power (attenuation)**
- 2. No transmission systems can transmit different Fourier components with the same speed (delay distortion)**
- 3. No transmission systems is free from noise**

ad 1)

If all harmonics would be equally diminished the signal would be reduced in amplitude but not distorted.

Unfortunately all transmission systems diminish different harmonics by different amounts.

Usually amplitudes from 0 up to certain frequency  $F_c$  are transmitted undiminished with all frequencies above  $F_c$  are strongly attenuated.

$F_c$  may be caused by a physical property of the transmission medium.

$F_c$  may be caused by filter function introduced intentionally in the transmission system (Pupin).

$F_c$  is synonymous for useable bandwidth  $B$  of a given transmission system.

ad 2)

For digital data it may happen that fast components from one bit may catch up and overtake slow components from the bit ahead and hence bits are mixed

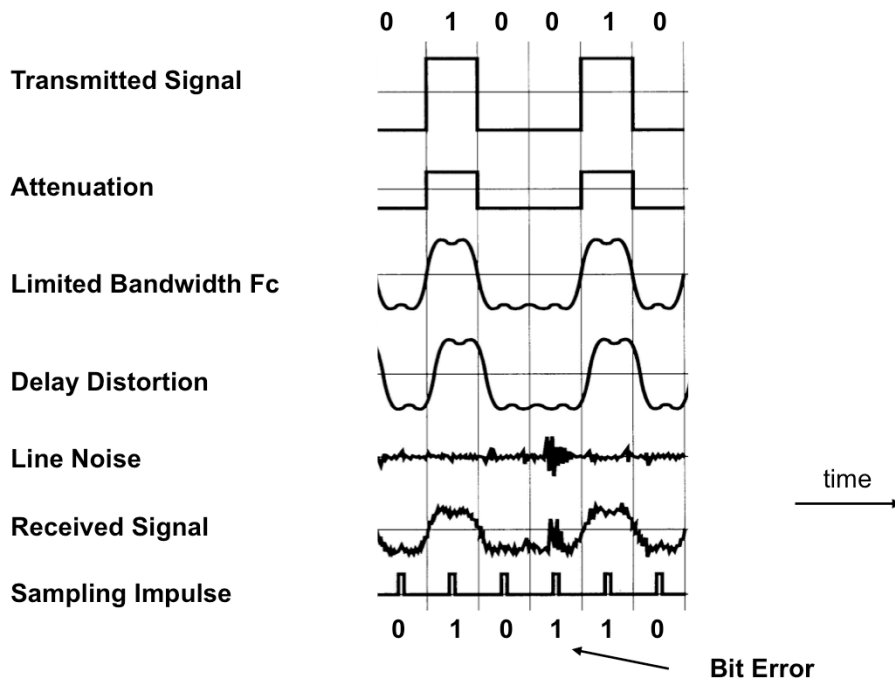
Inter-symbol interference. Eye-diagram for visualization of delay distortion.

ad 3)

Noise is unwanted energy from sources other than from the transmitter

## L01 - Communication Basics (v6.0)

### That Happens To A Signal !!!



## L01 - Communication Basics (v6.0)

### Real Data Transmission

- **In real transmission systems**
  - The original signal will be attenuated, distorted and influenced by noise when traversing the transmission line
  
- **By increasing the bit rate**
  - Bit synchronization even in middle of a bit becomes more and more difficult because of these impairments
  - Above a certain rate bit synchronization will be impossible
  
- **Relationship**
  - Between bandwidth  $F_c$ , line length and maximum achievable bit rate on a certain transmission line (system)

## L01 - Communication Basics (v6.0)

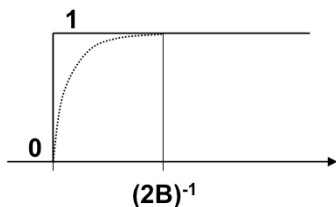
## Maximal Information Rate (Theoretical)

- **What is the maximal information rate of an ideal (noiseless) but bandwidth limited transmission channel ?**
  - Nyquist law:  $R = 2 * B * \log_2 V$ 
    - valid for a noiseless channel
    - R ... maximum bit rate (bits/sec)
    - B ... bandwidth range of a bandwidth limited transmission
    - V ... number of signal levels (e.g. 2 for binary transmission)
  
  - example analogue telephone line
    - B = 3000 Hz (range 400 – 3400 Hz)
    - R = 6000 bits/sec for V = 2
    - R = 18000 bits/sec for V = 8

## L01 - Communication Basics (v6.0)

## Nyquist Law Rationale

- **Maximal data rate proportional to channel-bandwidth B**
  - Rise time of Heavyside  $T=1/(2B)$
  - So the maximum rate is  $R=2B$ , also called the Nyquist Rate
  - Note: We assume an ideal channel here – without noise!
- **Bandwidth decreases with cable length**
  - As a dirty rule of thumb:  $BW \times \text{Length} \cong \text{const}$
  - But note that the reality is much more complex
  - Solitons are remarkable exceptions...



**Maximum signal rate: At least the amplitude must be reached**

Since each channel is a low-pass, and some channels even damp (very) low frequencies, data can only be transmitted within a certain channel bandwidth B.

If we put a 0 to 1 transition on the line (with ideally zero transition time), the receiver will see a slope with a rise time of  $T=1/(2B)$ .

So the maximal signal rate is  $T=1/(2B)$  – in theory. In practice we need some budget because there is noise and distortion and imperfect devices.

The longer the cable the more dramatically the low-pass behavior. In other words: on the same cable type we can transmit (let's say) 1,000,000,000 bits/s if the cable is one meter in length, or only 1 bit/s if the cable is one million kilometers in length.

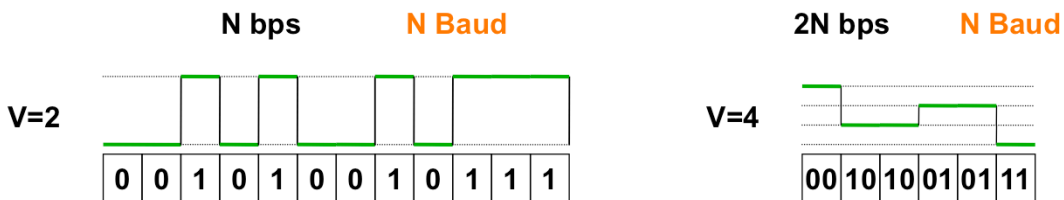
It is very interesting to mention that some modern fiber optic transmission methods violate this basic law. This methods base on so-called Soliton-Transmission.



## L01 - Communication Basics (v6.0)

## Bitrate versus Baud

- **The rate of changes of a symbol**
  - is called signaling rate  $R_s$  or **Symbol Rate**
  - is measured in Baud
- **The rate of bits transported**
  - is called bit rate  $R_i$  or **Information Rate**
  - and is measured in bit/sec (bps)
- $R_i = R_s * \log_2 V$ 
  - V ... number of signal levels
- $R_i = R_s$ 
  - for binary transmission where  $V = 2$
- **The goal is to send many (=as much as possible) bits per symbol**
  - => QAM (see next slides)



Baud is named after the 19th century French inventor Baudot, originally referred to the speed a telegrapher could send Morse Code.

Today the symbol rate is measured in Baud whereas the information rate is measured in bit/s.

## L01 - Communication Basics (v6.0)

## Maximal Information Rate (Reality)

- **What about a real channel? What is the maximum achievable information rate in presence of noise?**
  - Disturbance caused by crosstalk, impulse noise, thermal or white noise
- **Answer by C. E. Shannon in 1948**
  - Even when noise is present, information can be transmitted *without errors* when the information rate is below the *channel capacity C*
  - Channel capacity depends only on channel bandwidth and SNR (signal to noise ratio)
  - $\max R = C = B \cdot \log_2(1+S/N)$ 
    - S ... signal power, N ... noise power
    - SNR ... measured in decibel (db)
    - $SNR = 10 \cdot \log_{10} S/N$
  - example analogue telephone line
    - B = 3000 Hz
    - SNR = 30 db means  $30 = 10 \cdot \log_{10}(S/N) \rightarrow S/N = 1000$
    - $\max R = 3000 \cdot \log_2(1+1000) = 3000 \cdot (9,967226259)$
    - $\max R =$  approximately 29902 bits/sec

The great information theory guru Claude E. Shannon made a great discovery in 1948. Before 1948, it was commonly assumed, that there is no way to guarantee an error-less transmission over a noisy channel. However, Shannon shows that transmission without errors is possible when the information rate is below the so-called channel capacity, which depends on bandwidth and signal-to-noise ratio. This discovery is regarded as one of the most important achievements in communication theory.

## L01 - Communication Basics (v6.0)

### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

## L01 - Communication Basics (v6.0)

## Communication Channels

- **Usually *Low-Pass* behavior**
  - Higher frequencies are more attenuated than lower
- **Baseband transmission**
  - Signal without a dedicated carrier
  - Example: LAN technologies (Ethernet etc)
- **Carrierband / Narrowband transmission**
  - The baseband signal modulates a carrier to match special channel properties
- **Broadband transmission**
  - Different baseband signals modulate different carriers
  - Medium can be shared for many users / channels e. g. WLAN and cable networks

Each communication channel exhibits a low-pass behavior—at least beyond a very high frequency. Not only is the signal attenuated; phase shifts occur and even nonlinear effects sometimes rise with higher frequencies. The result is a smeared signal with little energy.

In most cases the signals do not need to be modulated onto a carrier. That is, all the channel bandwidth can be used up for this signal. We call this baseband transmission.

Baseband transmission / Baseband mode:

All the available bandwidth of the serial line is used to derive a single transmission path.

Signals travel as rectangle pulses.

Physical property of transmission medium, power of sender, sensitivity of receiver and S/N ratio are the limiting factors for the achievable bit rate.

Appropriate encoding ensures bit synchronization, avoids dc component, keeps electromagnetic radiation low

Carrierband transmission put the baseband signal onto a carrier with higher frequency. This is necessary with radio transmissions because low frequencies have a very bad radiation characteristic. Another example is fiber optics, where special signal frequencies are significantly more attenuated and scattered than others.

Broadband transmission / Broadband mode:

The available bandwidth of the serial line is divided to derive a number of lower bandwidth transmission paths on one serial line.

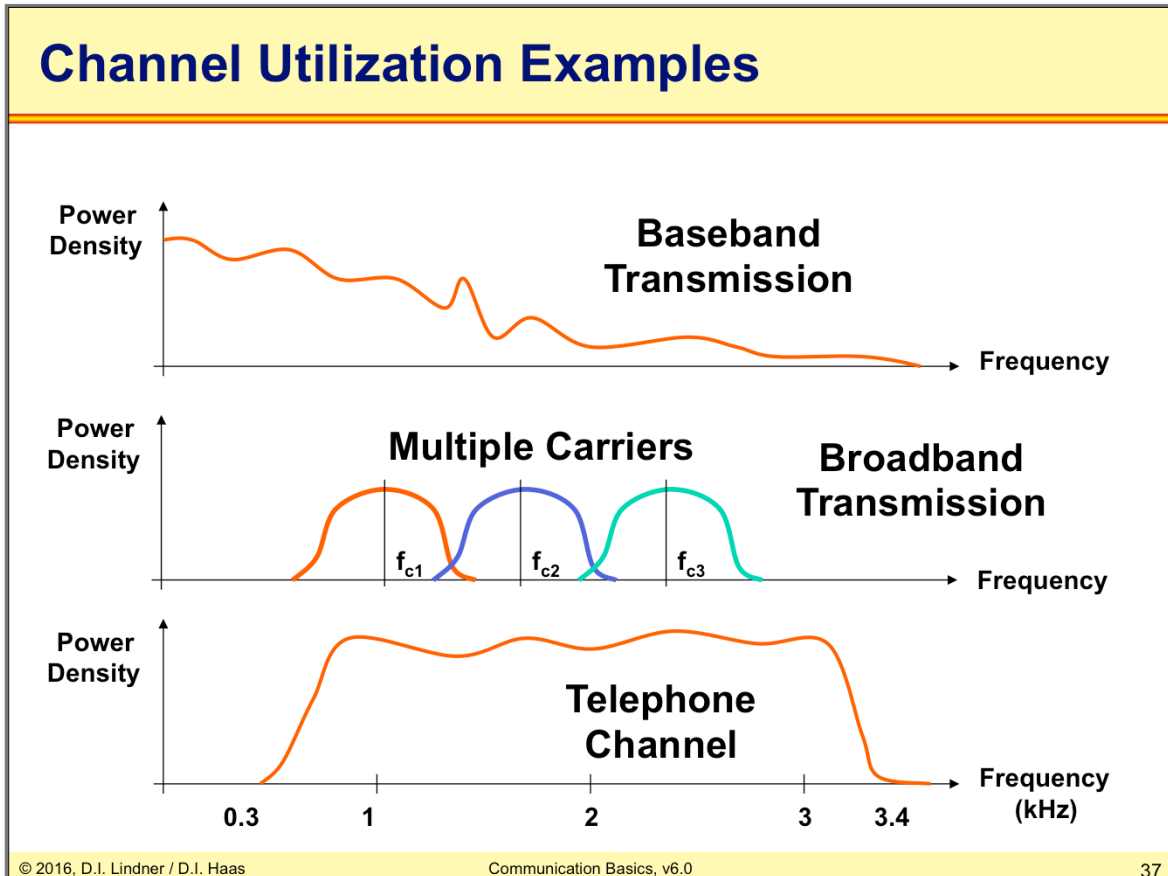
In analogue systems every path is modulated by a unique carrier.

A certain base-frequency - which together with the necessary bandwidth range for that channel - occupies a certain frequency band of the given transmission system.

Cable television is an example for that.

In digital systems broadband means sometimes high speed only e.g. B-ISDN = ATM - but no modulation is used to achieve these.

## L01 - Communication Basics (v6.0)



The above slide shows some examples for baseband and carrierband transmission. In case we use multiple carriers we may also call it broadband-transmission.

The third picture (bottom of slide) shows the spectral characteristic of a telephony channel (signal). The ITU-T defined an "attenuation-hose" in great detail (dynamics, ripples, edge frequencies, etc). As a rule of thumb we can expect low attenuation between 300 Hz and 3400 Hz.

Carrierband- Narrowband transmission / Carrierband-Narrowband Mode:

Bandwidth is intentionally limited and hence binary signals (rectangle pulses) must be adapted before using the line.

Adaptation is done by modulation. Modem originally used for transport of data over telephone network

Several modulation techniques were developed:

Amplitude modulation (amplitude-shift-keying ASK)

Frequency modulation (frequency-shift-keying FSK)

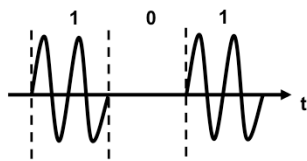
Phase modulation (phase-shift-keying PSK)

Combination of above methods used in modern high speed modems today (e.g. QAM - Quadrature Amplitude Modulation).

## L01 - Communication Basics (v6.0)

## Analogue Modulation Overview

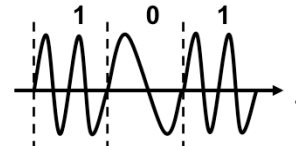
- **EVERY transmission is analogue – but there are different methods to put a base-band signal onto a high-frequency carrier**
- **The most simple (and oldest) is ASK**
  - The illustrated ASK method is simple "On-Off-Keying" (OOK)
- **FSK and PSK are called "angle-modulation" methods (nonlinear => spectrum shape is changed!)**
- **For digital transmission, almost always QAM is used**
  - The BER of BPSK is 3 dB better than for simple OOK



Amplitude Shift Keying (ASK)



Phase Shift Keying (PSK)



Frequency Shift Keying (FSK)

$$g(t) = A_t \cdot \cos(2\pi f_t t + \varphi_t)$$

These three parameters can be modulated

The slide shows a general modulation equation. The 3 parameters of the equation describe the 3 basic modulation types. All 3 parameters, the amplitude  $A_t$ , the frequency  $f_t$  and the phase  $\varphi_t$ , can be varied, even simultaneously. In nature, there is no real digital transmission; the binary data stream needs to be converted into an analog signal. As first step, the digital data will be "directly" transformed into an analog signal (0 or 1), which is called a baseband signal. In order to utilize transmission media such as free space (or cables and fibers) the base signal must be mixed with a carrier signal. This analog modulation shifts the center frequency of the baseband signal to the carrier frequency to optimize the transmission for a given attenuation/propagation characteristic.

### Amplitude Shift Keying:

A binary 1 or 0 is represented through different amplitudes of a sinus oscillation. Amplitude Shift Keying (ASK) requires less bandwidth than FSK or PSK since *natura non facit saltus*. However ASK is interference prone. This modulation type also used with infrared-based WLAN.

### Frequency Shift Keying:

Frequency Shift Keying (FSK) is often used for wireless communication. Different logical signals are represented by different frequencies. This method needs more bandwidth but is more robust against interferences. To avoid phase jumps, FSK uses advanced frequency modulators (Continuous Phase Modulation, CPM).

### Phase Shift Keying:

The 3rd basic modulation method is the Phase Shift Keying (PSK). The digital signal is coding through phase skipping. In the picture above you see the simplest variation of PSK, using phase jumps of 180°. In practice, to reduce BW, phase jumps must be minimized, and therefore PSK is implemented using advanced phase modulators (e. g. Gaussian Minimum Shift Keying, etc). The receiver must use same frequency and must be perfectly synchronized with the sender using a Phase Locked Loop (PLL) circuit. PSK is more robust as FSK against interferences, but needs complex devices.

After understanding these modulation methods QAM shall be introduced, which is the most important

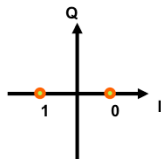
**L01 - Communication Basics (v6.0)****QAM: Idea**

- **"Quadrature Amplitude Modulation"**
- **Idea:**
  1. Separate bits in groups of words (e. g. of 6 bits in case of QAM-64)
  2. Assign a dedicated pair of Amplitude and phase to each word ( $A, \varphi$ )
  3. Create the complex amplitude  $Ae^{j\varphi}$
  4. Create the signal  $\text{Re}\{Ae^{j\varphi} e^{j\omega t}\}$   
 $= A (\cos \varphi \cos \omega t - \sin \varphi \sin \omega t)$  which represents one (of the 64) QAM symbols
  5. Receiver can reconstruct ( $A, \varphi$ )

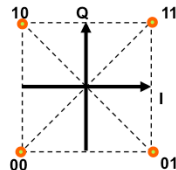
L01 - Communication Basics (v6.0)

## QAM: Symbol Diagrams

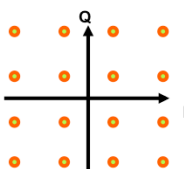
**Standard PSK**



**Quadrature PSK (QPSK)**

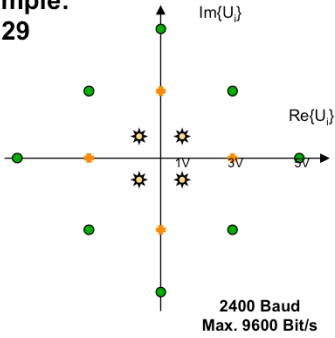


**16-QAM**



**Other example: Modem V.29**

- For noisy and distorted channels  
4800 bit/s
- ✱ For better channels  
7200 bit/s
- For even better channels  
9600 bit/s



2400 Baud  
Max. 9600 Bit/s

© 2016, D.I. Lindner / D.I. Haas Communication Basics, v6.0 40

The standard PSK method only use phase jumps of  $0^\circ$  or  $180^\circ$  to describe a binary 0 or 1 (two symbols). In the left picture above you see a enhanced PSK method, the Quadrature PSK (QPSK) method. While using Quadrature PSK each condition (phase shift) represent 2 bits instead of 1 (four symbols). Now it is possible to transfer the same data rate by halved bandwidth.

The QPSK signal uses (relative to reference signal)

- $45^\circ$  for a data value of 11
- $135^\circ$  for a data value of 10
- $225^\circ$  for a data value of 00
- $315^\circ$  for a data value of 01

Usually the assignment of bit-words to symbols is such that the error probability due to noise is minimized. For example the Gray-Code may be used between adjacent symbols to minimize the number of wrong bits when an adjacent symbol is detected by the receiver.

The above slide at the left side bottom shows the symbol distribution over the complex plane for the V.29 protocol (QAM-16) which is/was used by modems. Depending on the noise-power of the channel, different sets of symbols are used.

- 4800 bit/s requires 4 points (2 bits per symbol)
- 7200 bit/s requires 8 points (3 bits per symbol)
- 9600 bit/s requires 16 points (4 bits per symbol)

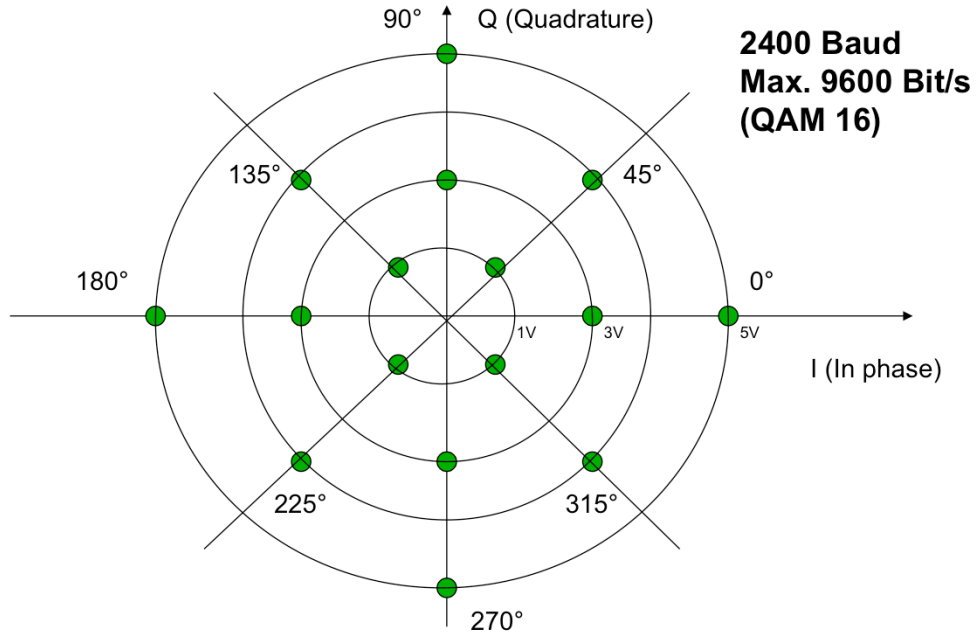
Note:

- 14,400 bit/s would require 64 points (6 bits per symbol) -> QAM-64
- 28,800 bit/s would requires 128 points (8 bits per symbol) -> QAM-128



L01 - Communication Basics (v6.0)

# Modem: V.29 (QAM) for TELCO Lines



**L01 - Communication Basics (v6.0)****Example QAM Applications**

- **One symbol represents a bit pattern**
  - Given N symbols, each represent  $\log_2(N)$  bits
- **Modems (Telco – 200-3500Hz limited),**
- **1000BaseT (Gigabit Ethernet)**
- **WiMAX, GSM, ...**
- **WLAN 802.11a and 802.11g:**
  - BPSK @ 6 and 9 Mbps
  - QPSK @ 12 and 18 Mbps
  - 16-QAM @ 24 and 36 Mbps
  - 64-QAM @ 48 and 54 Mbps

It is important to understand that spread spectrum (or OFDM) techniques are always combined with a symbol modulation scheme. Quadrature Amplitude Modulation (QAM) is a general method where practical methods such as BPSK, QPSK, etc are derived from.

The main idea of QAM is to combine phase and amplitude shift keying. Since orthogonal functions (sine and cosine) are used as carriers, they can be modulated separately, combined into a single signal, and (due to the orthogonality property) de-combined by the receiver.

And since  $A \cdot \cos(\omega t + \phi) = A/2 \{ \cos(\omega t) \cos(\phi) - \sin(\omega t) \sin(\phi) \}$  QAM can be easily represented in the complex domain as  $\text{Real}\{ A \cdot \exp(i \cdot \phi) \cdot \exp(i \cdot \omega t) \}$ .

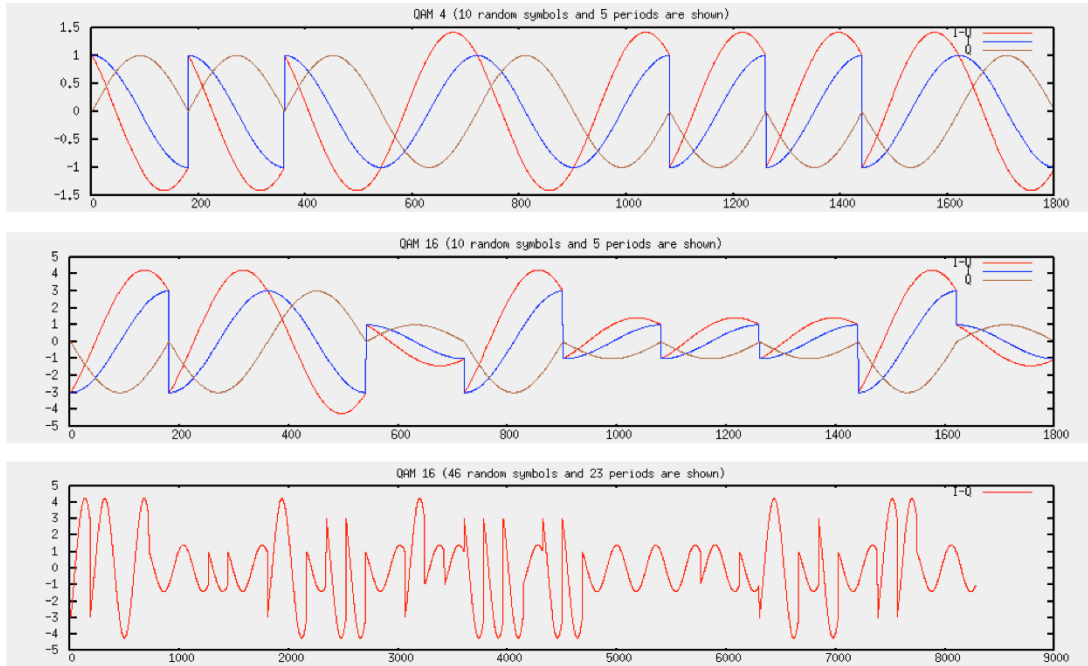
To reconstruct the original data stream the receiver need to compare the incoming signal with the reference signal. The synchronization is very important.

Why not coding more bits per phase jump ?

Especially in the mobile communication there are too much interferences and noise to encode right. As more bits you use per phase jump, the signal gets more "closer". It is getting impossible to reconstruct the original data stream. In the wireless communication the QPSK method has proven as a robust and efficient technique.

**L01 - Communication Basics (v6.0)**

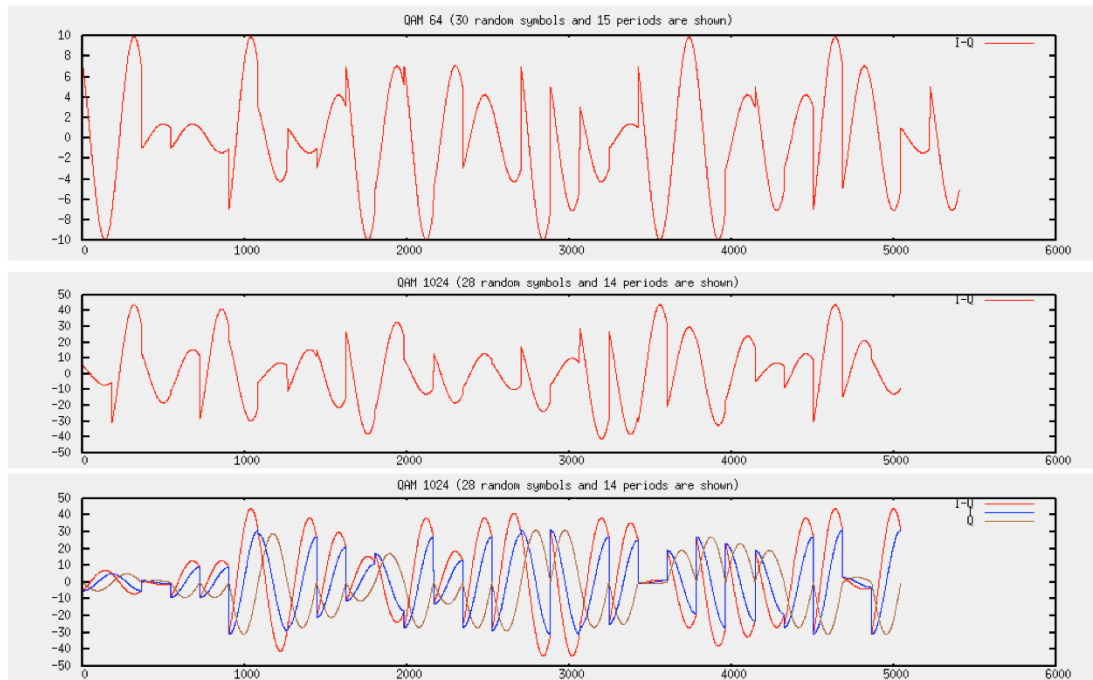
**QAM Example Symbols (1)**



Note that the above QAM signals show different successive QAM-symbols for illustration purposes. In reality each symbol is transmitted many hundred/thousand times

## L01 - Communication Basics (v6.0)

## QAM Example Symbols (2)



© 2016, D.I. Lindner / D.I. Haas

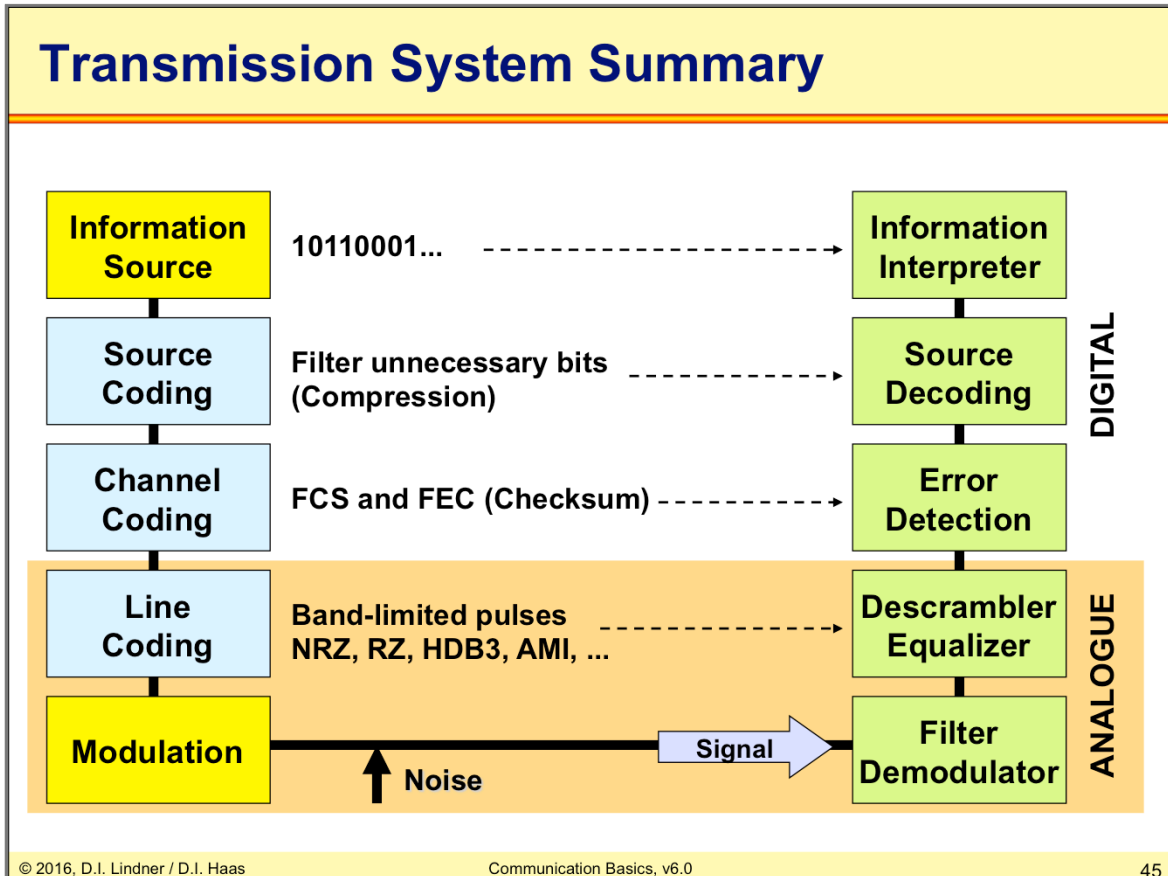
Communication Basics, v6.0

44

Note that the above QAM signals show different successive QAM-symbols for illustration purposes. In reality each symbol is transmitted many hundred/thousand times

These diagrams have been generated using Octave, a free Matlab clone.

## L01 - Communication Basics (v6.0)



Coding is not coding. The above slide gives you an overview about different coding purposes. Even modulation is sometimes called coding.

Source coding tries to eliminate redundancy within the information. Source coders must know well about the type of information that is delivered by the source.

Channel coding protects the non-redundant data stream by adding calculated overhead. Typically a Frame Check Sequence (FCS) is added. Only on very erroneous and/or long-delay links a Forward Error Correction (FEC) method might be useful. FEC requires too much overhead in most terrestrial applications.

Line coding focuses on the line, that is we want the symbols to be received correctly, even if noise and distortions are present. Furthermore line coding provides clock synchronization as discussed earlier.

Finally modulation might be necessary in case the channel has better properties at higher frequencies or the channel has only limited band of frequencies (good old telephone line -> limited to 50hz -> 3500kHz by Pupin coil inserted into the path in order to protect the analogue telephone system from high frequencies).

## L01 - Communication Basics (v6.0)

### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

## L01 - Communication Basics (v6.0)

## Time to Transmit A Given Number Of Bytes

$$\text{Serialization Delay (in ms)} = [ (\text{Number of Bytes} * 8) / (\text{Bitrate in sec}) ] * 1000$$

	Bitrate	9,6 kbit/s	48 kbit/s	128 kbit/s	2,048 Mbit/s	10 Mbit/s	100 Mbit/s	155 Mbit/s	622 Mbit/s	1 Gigabit/s
	Number of Byte	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )	Delay in msec ( $10^{-3}$ )
Bit	0,125	0,104167	0,020833	0,007813	0,000488	0,000100	0,000010	0,000006	0,000002	0,000001
Byte	1	0,833333	0,166667	0,062500	0,003906	0,000800	0,000080	0,000052	0,000013	0,000008
PCM-30	32	26,666667	5,333333	2,000000	0,125000	0,025600	0,002560	0,001652	0,000412	0,000256
ATM cell	53	44,166667	8,833333	3,312500	0,207031	0,042400	0,004240	0,002735	0,000682	0,000424
Ethernet	64	53,333333	10,666667	4,000000	0,250000	0,051200	0,005120	0,003303	0,000823	0,000512
X.25	256	213,333333	42,666667	16,000000	1,000000	0,204800	0,020480	0,013213	0,003293	0,002048
IP	576	480,000000	96,000000	36,000000	2,250000	0,460800	0,046080	0,029729	0,007408	0,004608
Ethernet	1.518	1.265,000000	253,000000	94,875000	5,929688	1,214400	0,121440	0,078348	0,019524	0,012144
FR	8.192	6.826,666667	1.365,333333	512,000000	32,000000	6,553600	0,655360	0,422813	0,105363	0,065536
TCP	65.534	54.611,666667	10.922,333333	4.095,875000	255,992188	52,427200	5,242720	3,382400	0,842881	0,524272

1kbit/s = 1000 bit/s !!!  
1KByte = 1024 Byte !!!

Serialization delay is the time which is necessary to put a block of bits on a serial line with a given bitrate.

## L01 - Communication Basics (v6.0)

## Propagation (Signal) Delay

$$T_p = \text{Propagation Delay (in ms)} = [ (\text{Distance in m}) / (\text{velocity in m/sec}) ] * 1000$$

	Distance	v=200.000km/s	v=300.000km/s
		Delay in msec (10 <sup>-3</sup> )	Delay in msec (10 <sup>-3</sup> )
CPU Bus	10 cm	0,0000005	0,0000003
	1 m	0,0000050	0,0000033
RS232, V24/V.28	15 m	0,0000750	0,0000500
LAN, Copper, RJ45	100 m	0,0005000	0,0003333
LAN, FO, X.21/V.11-V.10	1 km	0,0050000	0,0033333
Local Subscriber Line	2,5 km	0,0125000	0,0083333
WAN Link Repeater	10 km	0,0500000	0,0333333
WAN Link Repeater	100 km	0,5000000	0,3333333
WAN FO Link Repeater	1.000 km	5,0000000	3,3333333
WAN FO Link Repeater	10.000 km	50,0000000	33,3333333
Satellite Link	40.000 km	200,0000000	133,3333333
Satellite Link	50.000 km	250,0000000	166,6666667
	100.000 km	500,0000000	333,3333333
	300.000 km	1500,0000000	1000,0000000

$$\text{Total Delay (for a block of bits)} \\ = \text{Serialization Delay} + \text{Propagation Delay} + (\text{Switching Delay})$$

Propagation delay is the time which is needed for a electrical signal to propagate along a given length of line / transmission path. The upper limit for velocity is of course the speed of light.

Switching delay additionally occurs in case of the presence of an active component in the transmission path.

Examples for such active components:

Amplifiers / Repeaters

Synchronous TDM Switches; Circuit switching (PDH, SDH, ISDN) with low and constant delay

Asynchronous TDM Switches; Packet switching with variable delay

Some examples for packet switches:

X.25 switches, Frame Relay switches and ATM switches (WAN)

Ethernet switches (LAN)

IP router (LAN and WAN)



## L01 - Communication Basics (v6.0)

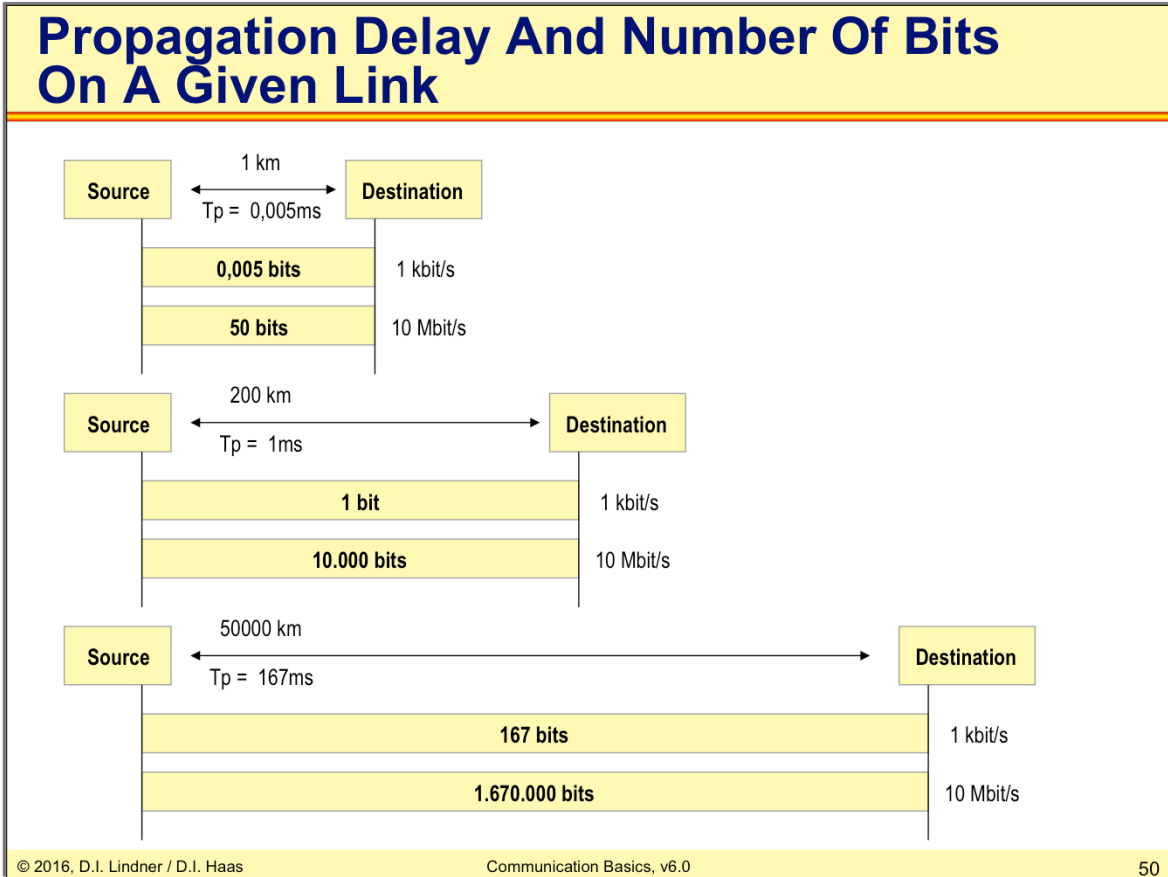
## How Long Is A Bit?

$$\text{Length (in m)} = [ ( 1 / ( \text{bitrate per sec} ) ] * [ ( \text{velocity in m/sec} ) ]$$

	Bitrate	Bit Length in meter	Bit Length in meter
Analogue Modem	9,6 kbit/s	20833,33	31250,00
Analogue Modem	48 kbit/s	4166,67	6250,00
DS0	64 kbit/s	3125,00	4687,50
ISDN (2B)	128 kbit/s	1562,50	2343,75
PCM-30, E1	2,048 Mbit/s	97,66	146,48
Token Ring 4	4 Mbit/s	50,00	75,00
Ethernet	10 Mbit/s	20,00	30,00
Token Ring16	16 Mbit/s	12,50	18,75
Fast Ethernet, FDDI	100 Mbit/s	2,00	3,00
ATM STM1, OC-3	155 Mbit/s	1,29	1,94
ATM STM4, OC-12	622 Mbit/s	0,32	0,48
Gigabit Ethernet	1 Gigabit/s	0,20	0,30
OC-48	2,5 Gigabit/s	0,08	0,12
10 Gigabit Ethernet	10 Gigabit/s	0,02	0,03
		Copper	LWL - Free Space
		200.000 km /sec	300.000 km / sec

If we combine these two attributes we can say that a bit – given with a certain bit rate on a line travelling with a certain speed along the line – has a certain length on the line.

L01 - Communication Basics (v6.0)



The picture shows how many bits are stored on a line / transmission path depending on the distance between and the bitrate used.

## L01 - Communication Basics (v6.0)

### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

**L01 - Communication Basics (v6.0)****Requirements & Facts  
Serial Transmission System**

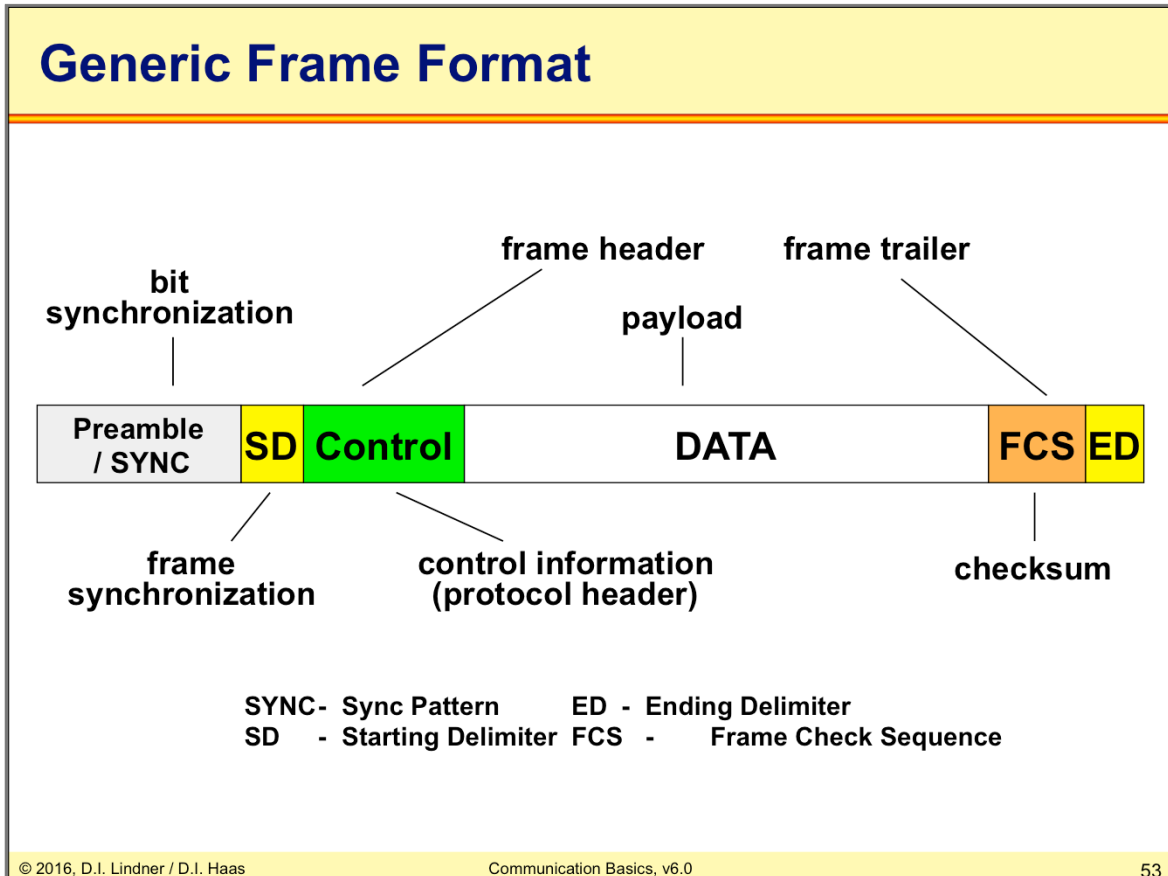
- **Information between systems is exchanged in blocks of bits**
  - Every block is carried in as so called transmission frames
- **The recognition of the beginning and the end of a block in the received bit stream is necessary**
  - Frame synchronization
- **Errors on physical lines may lead to damage of digital information**
  - 0 becomes 1 and vice versa
  - The longer the block the higher the probability for an error
- **Methods necessary for error checking**
  - Frame protection
  - Error detection and recovery

Framing is the task of packing the information of higher layers to provide for example start and end of packet detection plus some optional features which will be discussed on the following slides.

Frame protection is used to detect possible errors during data transmission.

Error recovery can be used to allow packet retransmissions if data errors are detected by the frame protection mechanism. It is based on a successful error detection.

## L01 - Communication Basics (v6.0)



Generic Frame Consists of Data and Metadata (Header or "Overhead") and requires synchronous physical transmission (PLL) to allow arbitrary frame lengths

- **Preamble / SYNC** - is used to provide synchronization between the sender and the receiver transmission clock. This is necessary to allow the detection of the single bit borders.
- **SD** - Start Delimiter is needed to detect the actual beginning of the transmission frame. From this point on data is fed from the physical layer into the receive buffer.
- **Control Field** - provides optional addressing, connection establishment, error recovery and flow control
- **Data** - is the payload provided by higher OSI layers
- **FCS** - Frame Check Sequence is used for error detection
- **ED** - End Delimiter is used to determine the end of the frame

**L01 - Communication Basics (v6.0)****Preamble**

- **Preamble / SYNC is a special bit pattern**
  - Used for bit synchronization after an idle period (Preamble)
  - Can be used as fill pattern during idle times to keep the receiver clock synchronized (SYNC)
- **Enables PLL synchronization**
  - Typically a 0101010...-pattern
  - Example: 8 Byte preamble in Ethernet frames



The purpose of the Preamble is to lock the receiver clock towards the sender clock by the help of Phase Locked Loop (PLL) circuits. The Preamble is different depending on the type of Data-link technology that is used.

In Ethernet technology for example the bit pattern consists of 62 clock changes between a logical 1 and a logical 0, followed by two logical 1's to indicate the start of frame.

The Preamble is obviously only needed for synchronous physical layers.

In the case that an asynchronous physical layer is used, e.g. COM port on PC or async serial interface on a router, the Preamble / SYNC is not needed. Still the rest of the generic frame format may be used to envelop blocks of information.

**L01 - Communication Basics (v6.0)****Control Field**

- **Is used for implementing protocol procedures**
- **Contains information such as**
  - Frame type, protocol type
    - Data, Ack, Nack, Connect, Disconnect, Reset, etc.
    - IP, IPX, AppleTalk, etc.
  - Sequence numbers for identification of frame sequence
    - Necessary for error recovery and flow control with connection oriented services
  - Address information of source and destination in case of a multipoint line
  - Frame length, etc.



The contents of the control field depends on the tasks that need to be performed by the Data-link protocol.

So the control field could contain:

- Address information for addressing especially in point to multipoint environments
- Sequence numbers that can be seen like serial numbers for each single frame
- Acknowledgement Flags to indicate that the data was received properly
- Frame Type information to indicate whether it's a frame that carries data or control information
- Service Access Point (SAP) or payload type information to indicate what is transported by the frame
- Signaling information in case of connection oriented protocols to build up an connection

Details about that will be covered in other chapters of the lecture.

## L01 - Communication Basics (v6.0)

### Agenda

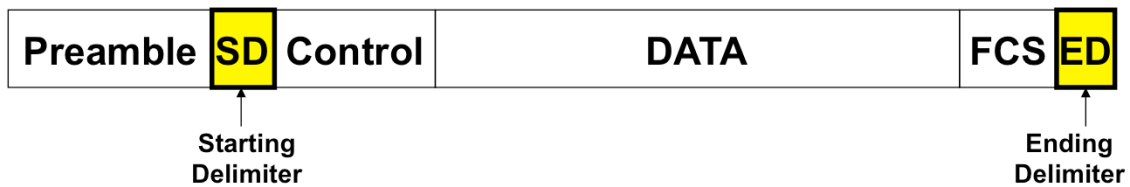
- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control



## L01 - Communication Basics (v6.0)

## Frame Synchronization

- **Beginning and ending of a frame is indicated by SD and ED symbols**
  - Bit-patterns or code-violations
  - Length-field can replace ED (802.3)
  - Idle-line can replace ED (Ethernet)
- **Also called "Framing"**



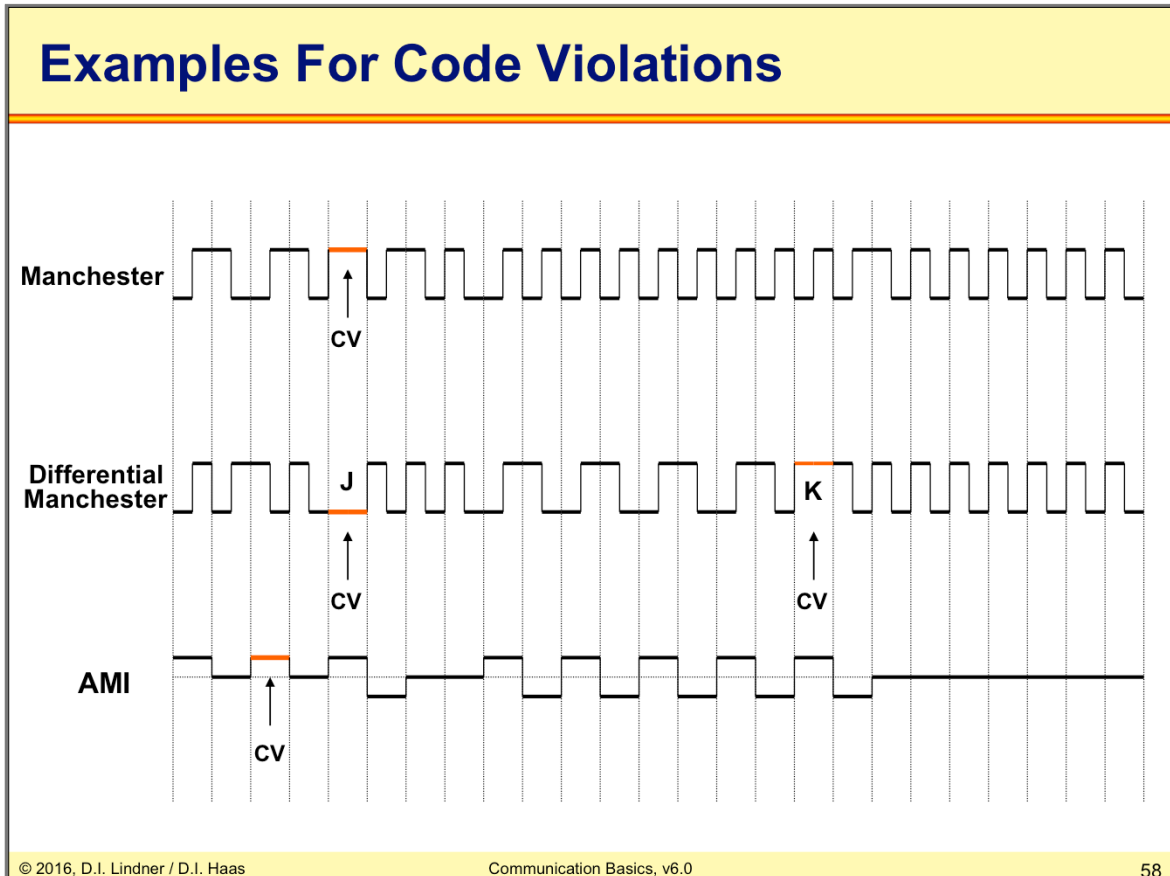
There are different methods available to indicate the start and the end of a data frames. The simplest method is the use of a special bit pattern. In the HDLC protocol and its derivatives the bit pattern "01111110" is used to indicate start and end of frame.

In Token Ring technology code violation is used. Code violation is an intended brake in the rules of coding.

In Ethernet technology the SD is indicated by the bit pattern "11" following the Preamble. But the end of the frame is indicated by an idle line, this means silence on the wire for a specified period of time.

Optionally the ED can be omitted if an length field is present inside the Data-link frame. In this case the end of frame can be calculated by counting the number of bytes received.

## L01 - Communication Basics (v6.0)



Code violation is an intended hurt of the rules of coding. It can be used for signaling purposes.

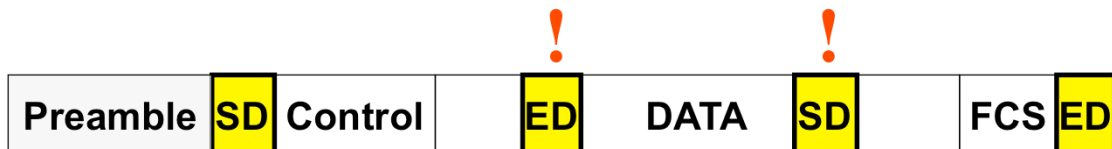
In Token Ring systems for example the differential Manchester code is used. Violations of the differential Manchester code are used for the SD and ED patterns to indicate the start and the end of frame.

The differential Manchester code violation symbols are called J and K. The code violation in the differential Manchester code is achieved by omitting the change from 1 to 0 or from 0 to 1 in the middle of a pulse.

## L01 - Communication Basics (v6.0)

## Protocol Transparency

- **What, if delimiter symbols SD, ED occur within frame?**
- **Solution:**
  - Byte-Stuffing
  - Bit-Stuffing



In the case that a special bit pattern is used to indicate the start and end of a frame, it is necessary to prevent this pattern inside the data portion of the frame. Otherwise this would lead to frame misinterpretation.

If application of a sender must take care of avoiding this bit patterns in the data stream, the transmission is not data-transparent or protocol transparent. The goal is of course to achieve data / protocol transparency to put the burden of the application.

There are several principle methods to achieve this goal.

The most famous are bit-stuffing by modifying single bits of the data stream and byte-stuffing by replacing the whole byte.

Data/Protocol Transparency Techniques:

Byte-stuffing with character based method e.g. IBM BSC (Binary Synchronous Control) protocol, PPP over asynchronous line

Bit-stuffing with bit oriented method e.g. HDLC (High level Data Link Control), PPP over synchronous line

Code violations e.g. Token Ring J,K Symbols of Differential-Manchester-code

Byte count e.g. DDCMP (Digital Data Communications Message Protocol)

L01 - Communication Basics (v6.0)

# Character-Oriented Transmission ASCII-Code

American Standard Code for Information Interchange

Bit Positions	7	0	0	0	0	1	1	1	1
	6	0	0	1	1	0	0	1	1
5	0	1	0	0	1	0	1	0	1
0 0 0 0	Nul	DLE	SP	0	@	P	\	p	
0 0 0 1	SOH	DC1	!	1	A	Q	a	q	
0 0 1 0	STX	DC2	"	2	B	R	b	r	
0 0 1 1	ETX	DC3	#	3	C	S	c	s	
0 1 0 0	EOT	DC4	\$	4	D	T	d	t	
0 1 0 1	ENQ	NAK	%	5	E	U	e	u	
0 1 1 0	ACK	SYN	&	6	F	V	f	v	
0 1 1 1	BEL	ETB	`	7	G	W	g	w	
1 0 0 0	BS	CAN	(	8	H	X	h	x	
1 0 0 1	HT	EM	)	9	I	Y	i	y	
1 0 1 0	LF	SUB	*	:	J	Z	j	z	
1 0 1 1	VT	ESC	+	;	K	[	k	{	
1 1 0 0	FF	FS	,	<	L	\	l		
1 1 0 1	CR	GS	-	=	M	]	m	}	
1 1 1 0	SO	RS	.	>	N	^	n	~	
1 1 1 1	SI	US	/	?	O		o	DEL	

Transmission Control

Format Control

Printable Character

Information Separator

Others

In the old days of data communication character-oriented protocols require encoding of information to be character-based (most famous 7-bit ASCII-Code).

That is fine for printable information to be transmitted and leaves coding points to be used for other protocol purposes: Start of Header, Ack and so on.

In case of transmitting non-printable information such as measurement values (any bit pattern) either the application of such a protocol has to encode this bit patterns to printable ASCII values using an agreed mapping rule or use byte-stuffing to avoid getting in conflict with control patterns of ASCII codepoints.

## L01 - Communication Basics (v6.0)

## Byte-Stuffing

- **Some character-oriented protocols divide data stream into frames**
  - Old technique, not so important today
  - e.g. IBM BSC (Binary Synchronous Control) protocol
- **Data Link Escape (DLE) character indicates special meaning of next character**

Data to send:

A B C DLE E F G ETX H I STX H

DLE STX A B C DLE DLE E F G ETX H I STX H DLE ETX

Typically, character-oriented protocols use asynchronous transmission (start-stop bits), so the receiver gets a bunch of characters but has to find the frames in it. In this case special control characters Start of Text (STX, 0x02) and End of Text (ETX, 0x03) are used to indicate start/end of a transmission frame.

Obviously STX and ETX must not appear inside the data portion, so an additional special control character Data Link Escape (DLE, 0x10) is used.

STX and ETX are only interpreted as control characters if the DLE pattern is found in front of them. This means if STX or ETX bit pattern are found inside the data stream, with the DLE pattern in front of them, they are interpreted as data.

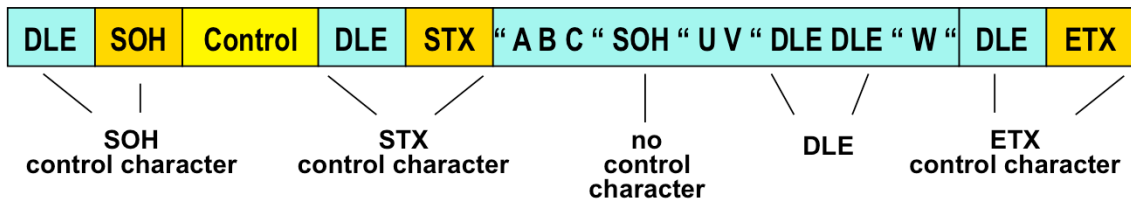
If the DLE pattern itself would appear inside the data portion it is doubled to indicate that it is only data.

L01 - Communication Basics (v6.0)

## Character Based Transmission With And Without Protocol Transparency



Transmission in non-data transparent mode; control character not allowed in data block



Transmission in data-transparent mode with byte-stuffing;  
control character allowed in data block

Byte-stuffing: DLE inside data portion will be doubled by sender;  
receiver deletes this doubled DLE

## L01 - Communication Basics (v6.0)

## Bit-Stuffing

- **Used in bit-oriented protocols**
  - Used by most protocols
  - Bits represent smallest transmission unit
- **HDLC-like framing: 01111110-pattern**
- **Rule:**
  - Transmitter-HW inserts a zero after five ones
  - Receiver rejects each zero after five ones

Data to send:

010011111000111111100101100110

01111110 01001111100001111101100101100110 01111110

In HDLC technology and its derivatives bit-stuffing is used to prevent the appearing of the SD, ED pattern inside the data frame. SD and ED equals 01111110 are called and also used for SYNC.

Any bit pattern different to flag will be interpreted as beginning of the frame. Flag should not occur inside the frame which would indicate the end of the frame. Bit-stuffing avoids the occurrence of the flag within a frame. There is a common rule between sender and receiver that after every fifth's 1 an additional logical 0 will be inserted by the sender in the data stream. The receiver itself removes all logical 0's following five logical 1's (that are zeros inserted by the sender). The appearance of sequence of 6 ones only occurs at the end of the frame.

One interesting side effect is that a given block of bits may need additional bits on the transmissions link hence additional bandwidth meaning based on the actual statistics of the to be transmitted bit patterns. If you have constant bit-rate on your link the actual delay in order to transport the complete block depends on the statistics.

## L01 - Communication Basics (v6.0)

### Agenda

- **Introduction**
- **Bit Synchronization**
  - Asynchronous
  - Synchronous
- **Physical Aspects**
  - Mathematical Background
  - Communication Channel / Modulation
  - Serialization / Propagation Delay
- **Transmission Frame**
  - Generic Format
  - Frame Synchronization
  - Error Control

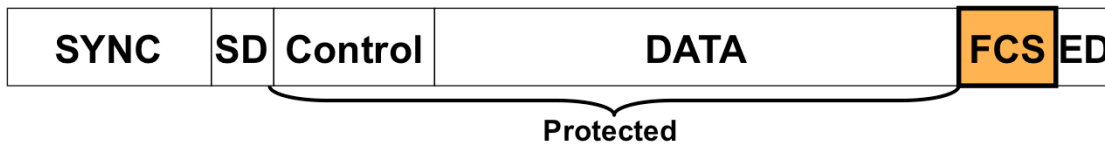


**L01 - Communication Basics (v6.0)**

## Error Control

- **Focus on error detection**

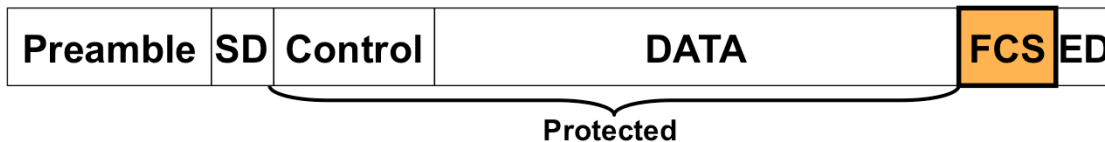
- Include enough redundant information with each block of data to enable receiver to detect only errors occurred -> error detecting codes -> Frame Check Sequence
- After error detection a retransmission of frame is initiated through protocol feedback to the sender
  - Area of ARQ-techniques
  - Feedback Error Control



## L01 - Communication Basics (v6.0)

## Frame Protection

- **A frame check sequence (FCS) protects the integrity of our frame**
  - From Sunspots, Mobile-Phones, Noise, Heisenberg and others
- **FCS is calculated upon data bits**
  - Different methods based on mathematical efforts: Parity, Checksum, CRC
- **Receiver compares its own calculation with FCS**



© 2016, D.I. Lindner / D.I. Haas

Communication Basics, v6.0

66

FCS principle:

Frame protection: Sender generates checksum (FCS) using an agreed rule in order to protect the data block. FCS is added at the end of the frame (FCS\_tmt)

Error detection: Receiver calculates its own FCS\_rcv and compares it with FCS\_tmt

FCS\_rcv = FCS\_tmt ... no error

FCS\_rcv not equal FCS\_tmt ... ERROR

After detection of an error frame is either silently discarded or an error recovery procedure based on retransmission the frame is initiated.

The Frame Check Sequence (FCS) allows the receiver to detect possible errors in the data stream.

The FCS is calculated by the sender and is attached at the end of the frame. The calculation of the FCS is typically performed in hardware.

There are many different technologies available to calculate an FCS like:

Parity bit calculation

XOR operation

Modulo operations

Cycle Redundancy Check (CRC)

Forward Error correction (FEC)

Complexity of checksum method determines types of errors that can be detected for 100% and the error probability for undetectable errors for a given frame size.

Different FCS methods were standardized depending on physical network type and expected line error patterns based on the experience made by monitoring of real transmissions systems.

**L01 - Communication Basics (v6.0)****FCS Methods**

- **Parity**
  - Even (10011101**1**) or odd (10011101**0**) parity bits
  - Examples: Asynchronous character-transmission and memory protection
- **Checksum**
  - Module 2 sum without carry bit (XOR operation)
  - Many variations and improvements
  - Examples: TCP and IP Checksums

The simplest method of FCS technologies is the parity bit calculation. This method is typically used in asynchronous character based transmission systems.

One parity bit is computed for each single character. The first two least significant bits are XORed together and the output of this operation is then XORed with the next more significant bit, and so on. The output of the final operation represents the required parity bit, which can be even (1) or odd (0).

Obviously the parity check can only protect against single bit errors. A two bit failure for example in the opposite direction 1 to 0 and 0 to 1 cannot be detected by the parity check.

For packet based transmission systems a similar method to calculate a checksum can be used. Instead of a single bit, 16 or 32 bit long words are used in combination with the XOR operation.

The XOR operation is also known as a modulo-2 adder, since the output of the XOR operation between two binary digits is the same as the addition of the two digits without a carry bit.

## L01 - Communication Basics (v6.0)

## Checksum Example: ISBN

- **100% Protection against**
  - Single incorrect digits
  - Permutation of two digits
- **Method:**
  - 10 digits, 9 data + 1 checksum
  - Each digit weighted with factors 1-9
  - Checksum = Sum modulo 11
  - If checksum=10 then use "X"

**ISBN 0-13-086388-2**

$$0*1+1*2+3*3+0*4+8*5+6*6+3*7+8*8+8*9 = 244$$

$$244 \text{ modulo } 11 = 2$$

ISBN stands for International Standard for Book Numbering.

The ISBN code which you may use to order your favorite books is protected by an checksum as well. This checksum is built on the basis of an modulo 11 operation.

The ISBN code itself consists of 10 digits where the first nine digits represent the book code and the tenth digit is the checksum used for error detection.

The ISBN checksum allows the detection of a single bit failure or the permutation of two bits with a probability of 100%.

## L01 - Communication Basics (v6.0)

## Cyclic Redundancy Check

- **CRC is one of the strongest methods**
- **Bases on polynomial-codes**
  - Protected bits are used as coefficients of polynomial
  - This polynomial is mod 2 divided by a generator-polynomial
  - The rest is the CRC-Checksum
  - Bit error burst with a maximal length of generator-polynomial are detected 100%
- **Several standardized generator-polynomials**
  - CRC-16:  $x^{16}+x^{15}+x^2+1$
  - CRC-CCITT:  $x^{16}+x^{12}+x^5+1$

Checksums which are built on basis of the XOR operation do not provide a reliable detection scheme against error bursts.

Therefore the Cycle Redundancy Check is based on mathematical polynomial equations and it is capable to detect even bursts of erroneous bits inside the data stream.

A single set of check bits is generated using an polynomial equation for each transmitted frame and appended to the tail of the frame by the sender. The receiver then performs the same computation on the complete frame and compares the received checksum with its own calculated checksum.

There are a lot of different standardized polynomial equations like the CRC-16, CRC-32 or the CRC-CCITT equations.

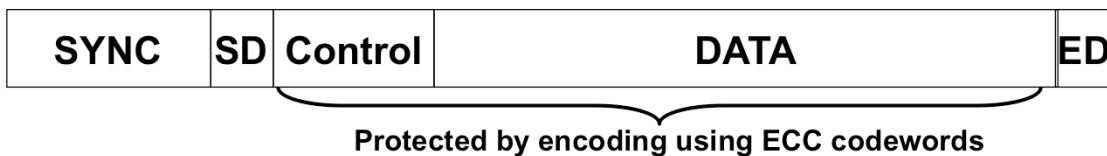
The CRC-16 check for example will detect all error bursts of less than 16 bits and most error bursts greater than 16 bits. The CRC-16 and the CRC-CCITT are mainly used in WAN technologies, while the CRC-16 is mainly used in LAN environments.

It is quite simple to implement these CRC checks in Hardware with the use of 16 or 32 bit long shift registers.

## L01 - Communication Basics (v6.0)

## Error Control

- **Focus on error correction**
  - Include enough redundant information with to enable receiver to correct errors occurred -> error correcting codes ECC (important -> “Hamming Distance”)
  - Forward Error Control (FEC)
  - Required for "extreme" conditions
    - High BER (Bit Error Rate), EMR
    - Long delays, space links
  - Examples: Reed-Solomon codes, Hamming-codes



All till now discussed error protection technologies lead to an drop or optional retransmission of the corrupted data fame. With the help of Forward Error Correction (FEC) technologies it is possible to fix the faults inside a data frame up to a certain extend.

One error correction coding scheme is the Hamming FEC scheme. In this scheme the data bits plus the additional check bits are called the codeword. The minimum number of bit positions in which two valid codewords differ is known as the Hamming distance of the code.

It can be shown that to correct  $n$  errors we need a code with a Humming distance of  $2n + 1$ .

In practice the number of check bits needed for error correction is much larger than the number of bits needed just for error detection. Therefore in most transport systems ARQ techniques are still used for error correction.

FEC technologies are mainly used in niche technologies, e.g. communication with space probes, where the RTT is very high and sometimes only an unidirectional communication channels exists.