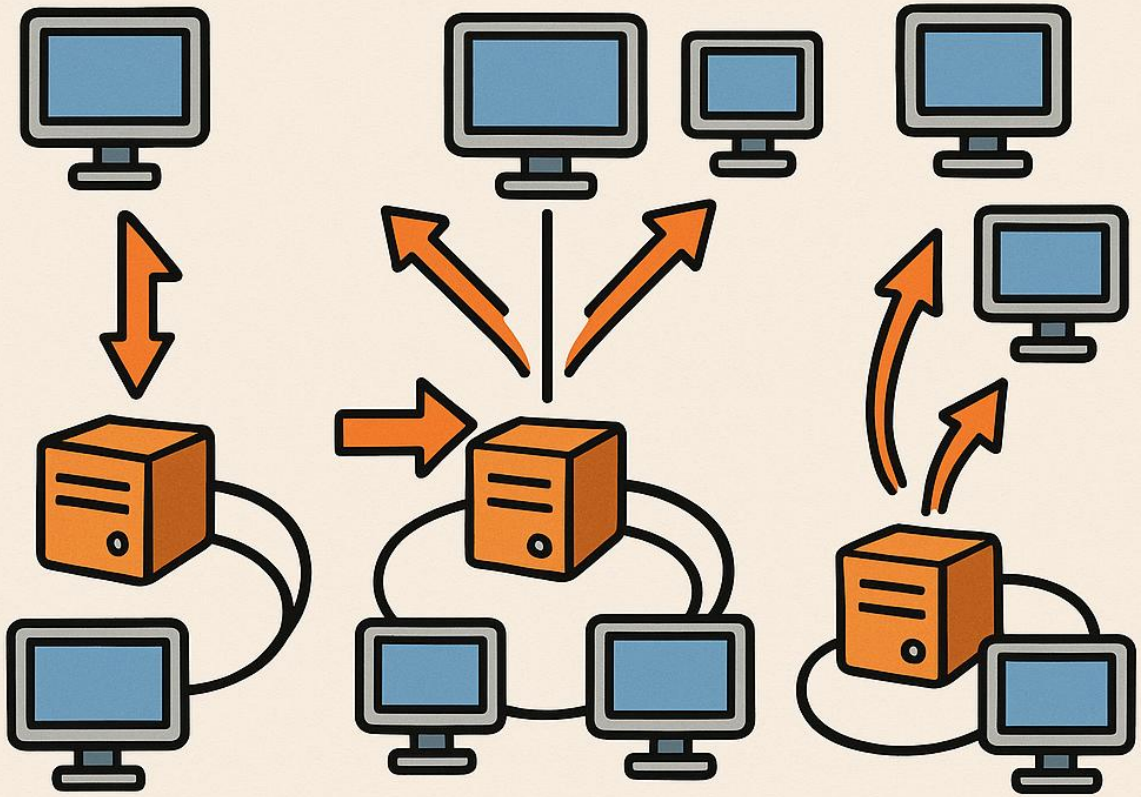
DRUHY PŘENOSŮ DAT V SÍTI



Ing. Petr Orvoš

SOŠ a SOU NERATOVICE

ÚVOD DO PŘENOSU DAT - ROZDĚLENÍ

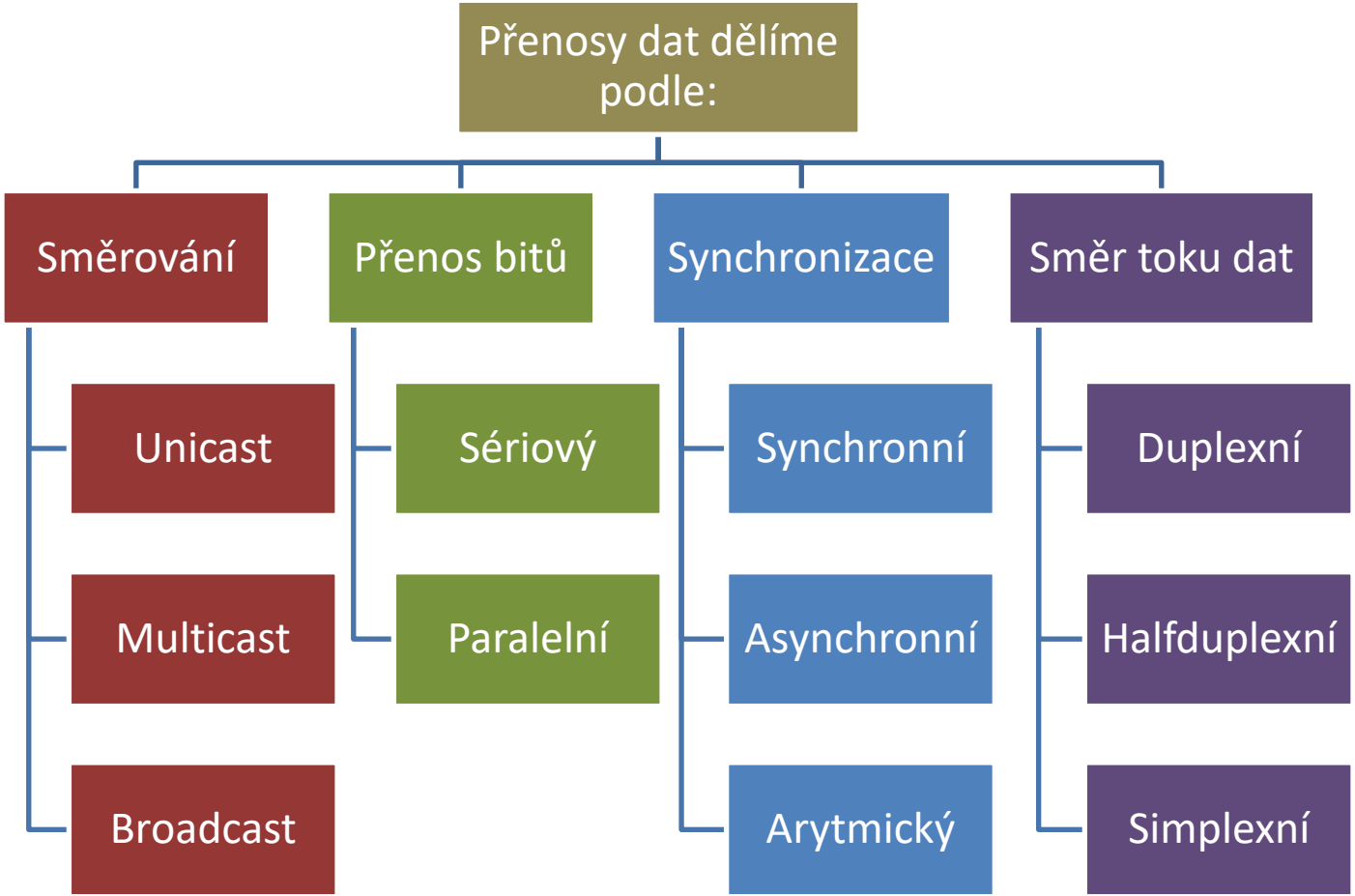
Každá síť – od jednoduchého propojení dvou počítačů až po celý internet – funguje na principu **přenosu dat**. Data se přenášejí **v bitech (0 a 1)** pomocí **elektrických, optických nebo bezdrátových signálů**.

Který přenos je nejlepší v dnešních reálných sítích?

Unicast - je to „nejčastěji používaný typ“ komunikace mezi dvěma uzly (klient–server, web, e-mail).

- **způsob přenosu bitů:** v sítích se téměř vždy používá **sériový přenos**
- **časování:** kvůli efektivitě se v praxi **preferuje synchronní** rámcování bloků (bez režie start/stop bitů)
- **směr toku:** moderní linky běží **plně duplexně** (současně oběma směry), např. Ethernet
- **kontrola**: **CRC**

PŘENOS DAT - ROZDĚLENÍ



PŘENOS PODLE SMĚROVÁNÍ DAT

UNICAST

- přenos od jednoho odesílatele k jednomu příjemci
- nejčastější typ komunikace

Příklad: webový prohlížeč ↔ server (HTTP, e-mail).

adresace cílového zařízení probíhá pomocí **MAC nebo IP adresy**

MULTICAST

- přenos od jednoho odesílatele ke skupině vybraných příjemců
- efektivní pro skupinovou komunikaci.

Příklad: IP televize, online přednášky, streamování.

využívá rozsah adres **224.0.0.0–239.255.255.255** (IPv4)

BROADCAST

- přenos od jednoho odesílatele ke všem v síti (v rámci jedné podsítě).
- výhoda: rychlé rozeslání všem
- nevýhoda: zatěžuje síť, protože všechna zařízení musí zprávu zpracovat

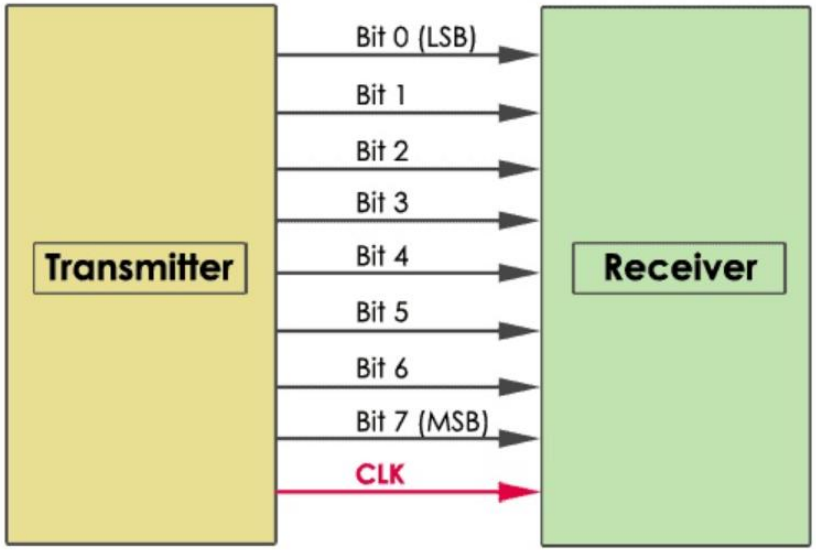
adresace cílového zařízení probíhá pomocí **broadcast MAC nebo IP adresy**

PŘENOSY PODLE ZPŮSOBU PŘENOSU BITŮ - VIDEO

https://www.youtube.com/watch?v=myU2x27FIIc&t=29s

PŘENOSY PODLE ZPŮSOBU PŘENOSU BITŮ - PARALELNÍ

- najednou se přenáší **více bitů současně** (např. 8 bitů = 1 byte)
- rychlý, ale **pouze na krátkou vzdálenost** (cca do 20 m)
- používal se mezi PC a tiskárnou (LPT port)
- předpokládá více-žilovou kabeláž
- nevhodný pro sítě – problém se **synchronizací a přeslechy**



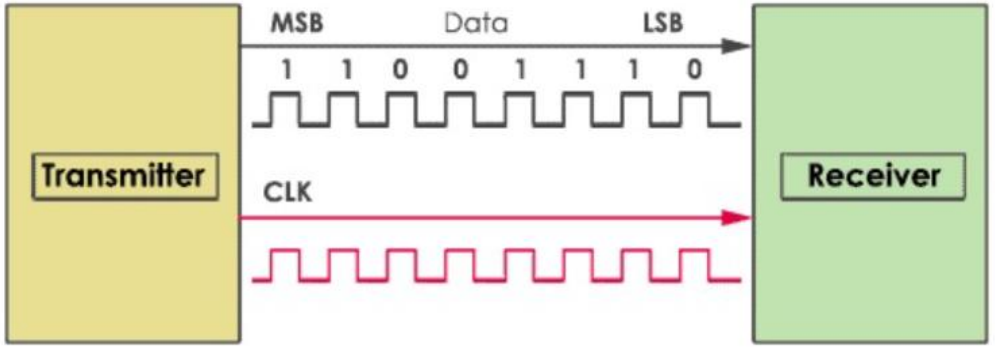
https://www.voxcafe.cz/

Paralelní komunikace přenáší 8, 16 nebo 32 bitů dat najednou.

PŘENOSY PODLE ZPŮSOBU PŘENOSU BITŮ - SÉRIOVÝ

- data se přenášejí **bit po bitu** po jednom vodiči nebo kanálu
- nejmenší položka dat přenášená sériově je označována jako znak a má obvykle rozsah 7 nebo 8 bitů
- pomalejší v jednom taktu, ale **spolehlivější a vhodný na větší vzdálenost kvůli synchronizaci času**
- dnes **převládající způsob přenosu** ve všech typech sítí

Příklad: Ethernet, USB, SATA, Wi-Fi



Například pokud se mají poslat 8bitová binární data **11001110** z vysílače do přijímače, tak jaký kousek se přenese jako první?

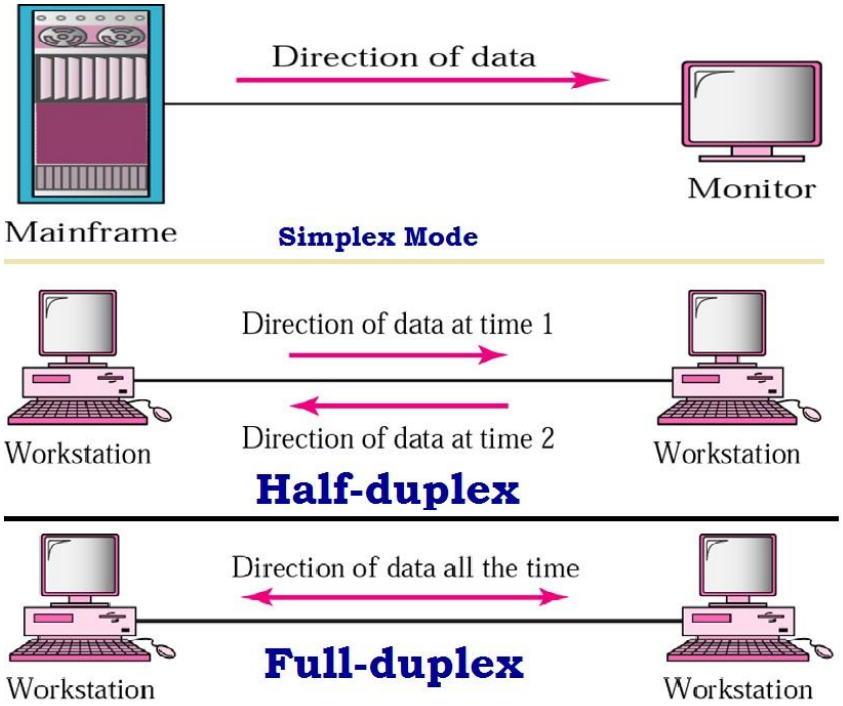
Z výše uvedeného diagramu pro každý hodinový impuls, vysílač posílá jeden bit dat do přijímače.

PŘENOSY PODLE ZPŮSOBU PŘENOSU BITŮ - SHRNUTÍ

Kritérium	Paralelní přenos	Sériový přenos
Počet přenášených bitů	Více bitů najednou (např. 8, 16, 32)	Jeden bit po jednom
Počet vodičů	Více vodičů (pro každý bit zvlášť)	Jeden vodič nebo kanál
Rychlost přenosu	Vyšší na krátké vzdálenosti	Vyšší efektivita na dlouhé vzdálenosti
Použití	Krátké vzdálenosti, uvnitř zařízení	Dlouhé vzdálenosti, propojení externích zařízení
Synchronizace	Může docházet k nesouladu (skew) mezi bity	Bez problému synchronizace jednotlivých bitů
Příklady	Připojení RAM, starší tiskárny (LPT port)	USB, Ethernet, SATA

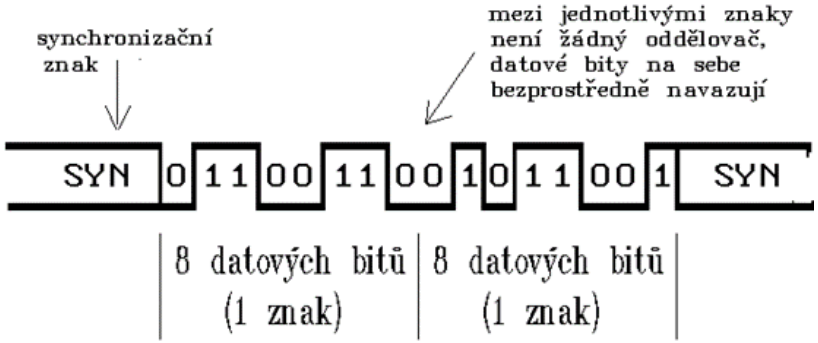
PŘENOSY PODLE SMĚRU TOKU DAT

typ přenosu	*směr*	*příklad*
simplexní	jednosměrný	staré TV vysílání
polo-duplexní	obousměrný, ale ne současně	vysílačky
Plně (full) duplexní	obousměrný současně	Ethernet, telefonní hovor



PŘENOS PODLE ČASOVÁNÍ - SYNCHRONNÍ

- data se posílají **v blocích (rámcích)** s pevným rytmem (taktem hodin)
- není potřeba start/stop bit – **vyšší rychlost** a efektivita, používá **SYN**
- odesílatel i příjemce se **synchronizují podle „hodinového signálu“**, ví kdy začíná a končí každý bit
- používá např. **kódování Manchester**, které kombinuje data s časovým signálem



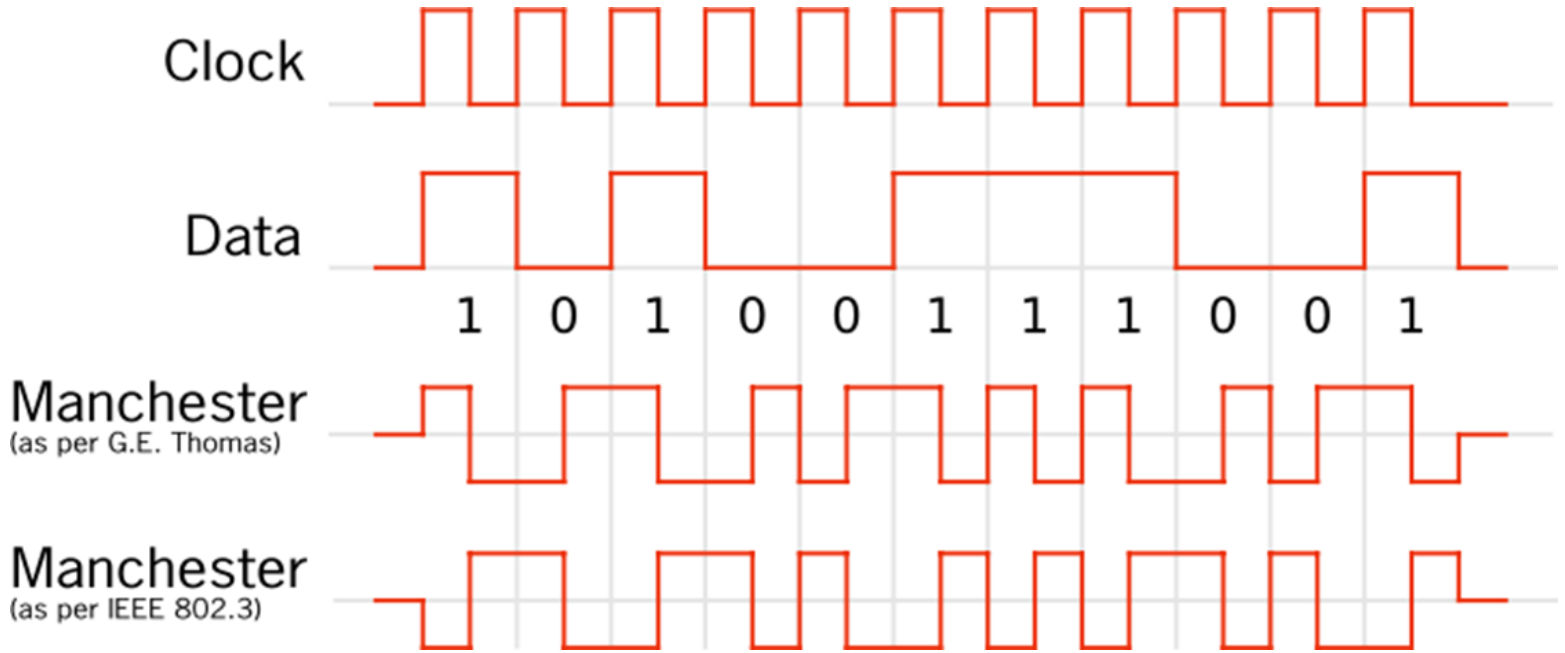
SYN = 00010110

Synchronizační znak (SYN) je speciální posloupnost bitů, která se na začátku synchronního přenosu posílá proto, aby si přijímač „srovnal hodiny“ s vysílačem. Binárně je to 22, HEX 0x16

Příklad: Ethernet

PŘENOS PODLE ČASOVÁNÍ - SYNCHRONNÍ

Konkrétních způsobů, jak sloučit hodinový a datový signál do jednoho výsledného signálu, existuje více. Poněkud nesprávně se tomu říká "kódování".

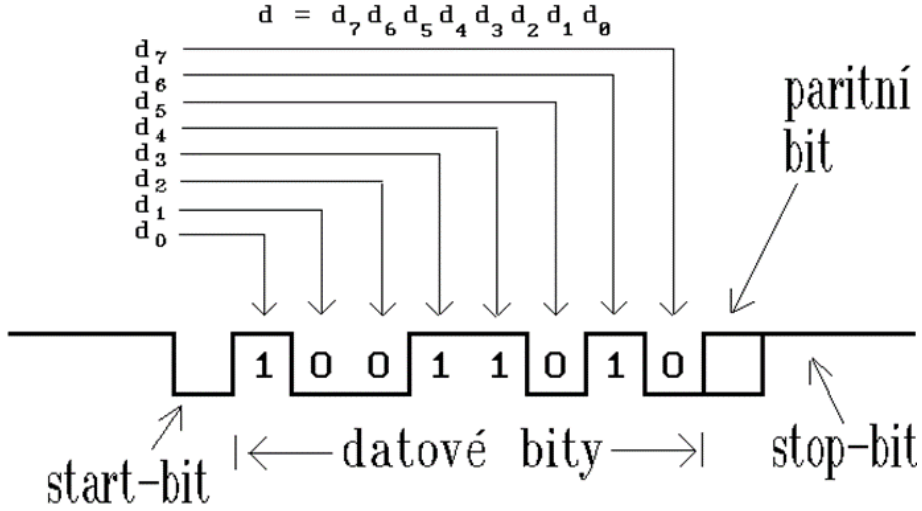


Příklad: kódování Manchester, a tzv. diferenciální Manchester.

PŘENOS PODLE ČASOVÁNÍ - ASYNCHRONNÍ

- data se posílají **nepravidelně**, znak po znaku – **nejsou časově sladěni!**
- každý znak má **start bit** a **stop bit,** příjemce tak pozná, kdy zpráva začíná a končí
- výhoda: jednoduché
- nevýhoda: nižší efektivita

Příklad: **RS-232**
(staré sériové porty).



Start-bit
- označuje **začátek přenosu znaku**
- je vždy logická **0**
- přijímač podle ní pozná, že přichází nový znak, a začne měřit čas pro jednotlivé bity.

Stop-bit
- označuje **konec přenosu znaku**
- je vždy logická **1** a trvá 1 – 2 datové periody
- dává přijímači čas „vydechnout“ a připravit se na další znak.

PŘENOS PODLE ČASOVÁNÍ - PŘÍKLAD

Každé písmeno má svůj **ASCII kód** (8bitový), celkem 32 znaků:

```
A → 01000001
H → 01001000
O → 01001111
J → 01001010
```

Asynchronní přenos (bez parity):

```
| start | 8 datových bitů | stop |
```

```
A : 0 01000001 1
H : 0 01001000 1
O : 0 01001111 1
J : 0 01001010 1
```

```
↓ začátek          konec znaku      pauza
|0 01000001 1|    |0 01001000 1|    |0 01001111 1|    |0 01001010 1|
```

Při čtyřech znacích se přenáší 4×10 bitů = 40 bitů (místo 32 datových).

PŘENOS PODLE ČASOVÁNÍ - PŘÍKLAD

Každé písmeno má svůj **ASCII kód** (8bitový), celkem 32 znaků:

```
A → 01000001
H → 01001000
O → 01001111
J → 01001010
```

Synchronní přenos:

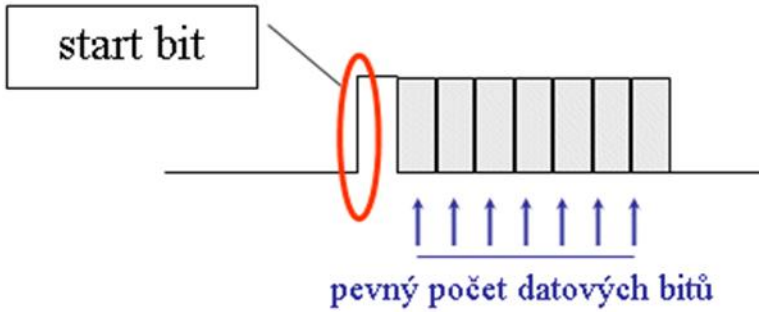
```
SYN : 00010110
DATA: 01000001 01001000 01001111 01001010

|00010110|01000001 01001000 01001111 01001010|
```

Při čtyřech znacích se přenáší 4×10 bitů = 40 bitů (místo 32 datových). Ale pokud bude přenos delší, SYN se použije jen jednou na začátku a poté to již bude jen 32 bitů za 4 znaky.

PŘENOS PODLE ČASOVÁNÍ - ARYTMICKÝ

- mezistupeň mezi synchronním a asynchronním
- vysílač i přijímač mají **vlastní hodiny**, které se při startu synchronizují
- nesdílejí společný takt, proto se **každý znak** rámuje zvlášť: začíná **start bitem** (0), následují **datové bity** (typicky 5–8), volitelný **paritní bit** a končí **stop bitem** (1), mezi znaky mohou být mezery.
- používá se tam, kde není nutná plná synchronizace.



CHYBY PŘI PŘENOSU

V komunikačním kanálu na zprávy působí **šum**, který může data poškodit — na některém místě v datovém proudu se místo 0 objeví 1 nebo naopak. Proto často využíváme **redundance** — posíláme spolu se zprávou nějakou další informaci, která nám pomůže odhalit chybu, nebo ji rovnou opravit.

Příklady redundancí jsou:

- **kontrolní součet** (checksum)
- **paritní bit** (parity)
- **kontrolním číslo**
- **poslání celé zprávy vícekrát**

CHYBY PŘI PŘENOSU - PARITA

- je **dohodnutý kontrolní bit**, který hlídá, zda je počet „jedniček“ v posílaném bajtu „správně“ – podle dohody buď **sudý (even), nebo lichý (odd). Podle toho sudá nebo lichá parita.+**
- **v Ethernetu se již nepoužívá**

Je to nejjednodušší způsob detekce chyby: **přidá se 1 bit, který „dorovná“ počet jedniček. Pokud se při přenosu otočí (znehodnotí) jeden bit, parita nesedí a přijímač pozná chybu.**

Příklad (8 datových bitů):
Data: 10110010 → mají 4 jedničky (sudý počet).
Sudá parita (even): paritní bit = 0 (celkem zůstane sudý počet jedniček).
Lichá parita (odd): paritní bit = 1 (celkem bude lichý počet jedniček).

Když se při přenosu jeden bit převrátí, součet jedniček se změní a parita nesouhlasí → chyba odhalena. Když se ale převrátí dva bity naráz, součet může zůstat „správný“ → parita chybu nemusí odhalit (je to metoda „nejméně účinná“).

CHYBY PŘI PŘENOSU - CHECKSUM

- kontrolní součet (checksum) je **číslo vypočtené ze všech posílaných dat** (z „bloku“). **Příjemce si ho spočítá znovu a porovná s přijatou hodnotou. Pokud nesedí, data jsou poškozená.** V jednoduché verzi se sčítají bajty „modulo 2^8“ nebo „modulo 2^{16}“, takže výsledkem je 1 nebo 2 bajty navíc.

Příklad:
Máme data (bajty): 10, 20, 30, 40.
Součet = 10+20+30+40 = 100.
Kontrolní součet (CS) = 100 mod 256 = 100.
Odešle se: 10,20,30,40,CS=100.

Na druhé straně:
Součet přijatých dat = 10+20+30+40 = 100 → shoda s CS ⇒ data OK.
Kdyby se při přenosu jeden bajt změnil (např. 30 → 31), nový součet by byl 101 → neshoda ⇒ detekce chyby.

Výhoda: jednoduchý, levný, rychlý na výpočet.

Nevýhoda: není tak „silný“ — některé vícenásobné či vzájemně se kompenzující chyby může neodhalit.

CHYBY PŘI PŘENOSU - CRC

- metoda detekce chyb pro celé bloky dat. CRC je **dělání se zbytkem** nad bity. **Zbytek** přibalíme k datům; když znovu **vydělíme**, musí vyjít **nula**. Když ne, data cestou utrpěla chybu.

Příklad:
Data: **1011**, Generátor: 1101 (polynom $x3+x2+1$ → **„CRC-3“; šířka zbytku = 3 bity**, (při CRC-16) je to 16 nul.)
Kroky:

1. K datům připojíme tři nuly: **1011 000**.
2. Dělíme polynomicky: posouváme „1101“ a vždy, když nahoře začíná jedničkou, provádíme XOR.
3. Po dokončení vyjde zbytek 100.
4. Odeslaný kódový slovo je: **1011 100**.
5. Příjemce vezme 1011100, vydělí 1101 → zbytek 000 ⇒ blok je v pořádku. Kdyby se při přenosu převrátil bit, zbytek by nebyl nulový ⇒ chyba odhalena.

CRC je nejúčinnější forma zabezpečení bloku dat s typickou přesností detekce chyb až ~99,99 %

Závěrem

Co dodat? Naučte se to, budeme z toho psát.

A to je vše přátelé.

POUŽITÁ LITERATURA a ZDROJE

PETERKA, Jiří. Archiv článků a přednášek [online]. [cit. 2025-04-24]. Dostupné z: http://www.earchiv.cz

CISCO Networking Academy. NetAcad [online vzdělávací portál]. [cit. 2025-04-24]. Dostupné z: http://www.netacad.com

https://www.voxcafe.cz/mindblog/clanky/embedded-systemy/seriova-komunikace.html

http://www.rikovsky.cz/programy/binarni.htm

Části této prezentace byly vytvořeny s využitím generativní umělé inteligence (ChatGPT, verze z roku 2025) jako podpůrného nástroje pro získávání informací a formulaci textu. Výsledky byly následně editovány a ověřeny autorem."