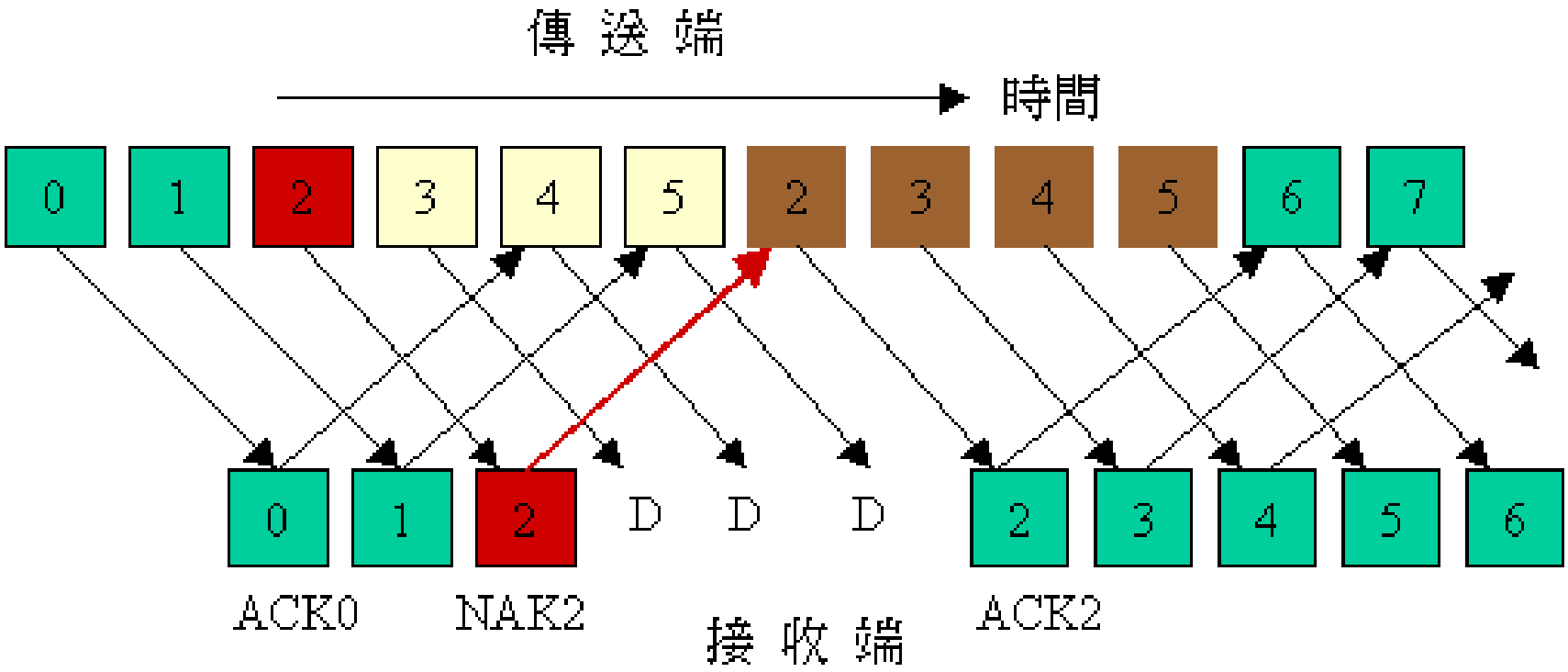


Mechanismy přenosu digitální komunikace

O čem přednáška je?

Děje a mechanismy implementované na úrovni fyzické vrstvy:

- Synchronní a asynchronní přenosy
- Chyby, detekce a opravy chyb
  - ✓ Parita, CRC (Cyclic Redundancy Check)
  - ✓ blokové kódy (detekční, opravné)
- Konfigurace spoje
  - ✓ plný duplex, poloviční duplex, simplex
- Komunikační rozhraní
  - ✓ V.24 / EIA-232-F – standard rozhraní na modem pro hlasové pásmo na veřejných komutovaných telekomunikačních (analogových) sítích
  - ✓ Rozhraní fyzické vrstvy ISDN,
    ISDN – *Integrated Services Digital Network*

Digitální vysílání v základním pásmu

- žádná nosná frekvence
- elektrické pulsy se vysílají přímo do přenosového média
- typicky 2-úrovňové pulsy, 1 bit/puls
- příklady používání
  - ✓ ISDN spojení po krouceném dvoudrátu (64 kb/s)
  - ✓ PCM hlas nebo data po speciálních kabelech (1,544 Mb/s a více)
  - ✓ LAN (kroucený dvoudrát, koaxiál, Ethernet)

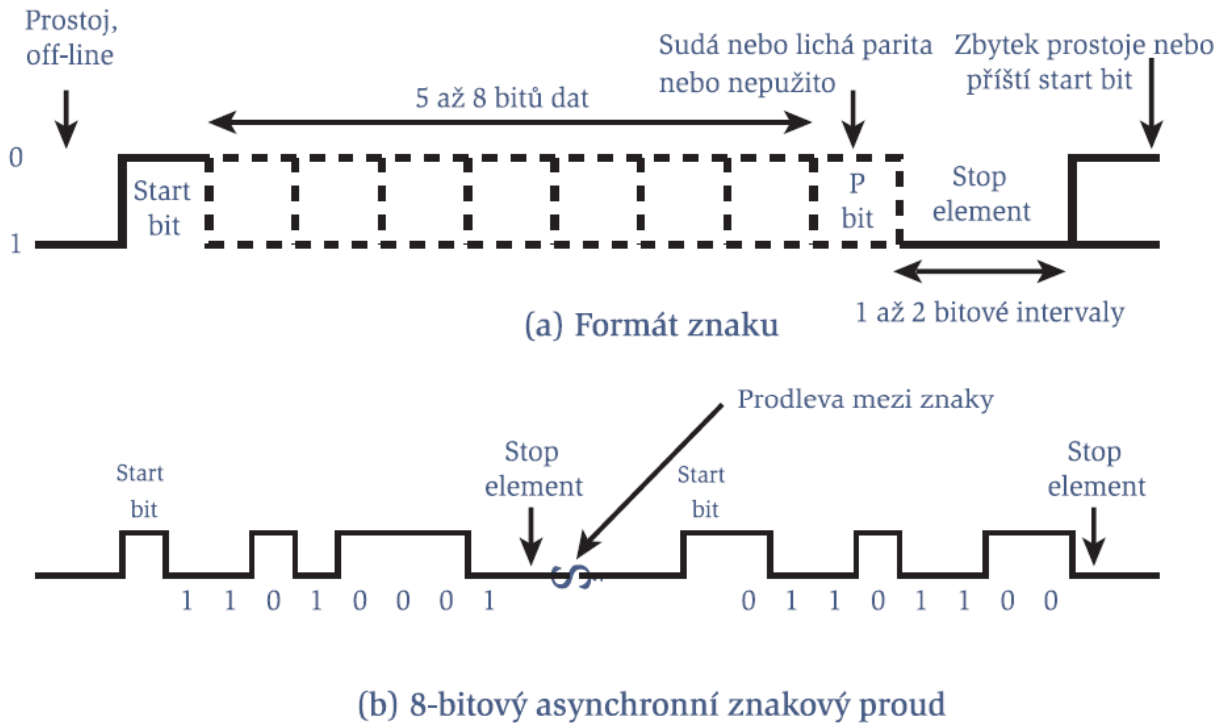
Synchronní a asynchronní přenosy

- Digitální data jsou přenášena sériovým proudem bitů
- Problém synchronizace času
  - ✓ přijímající strana musí být s vysílající stranou z hlediska vymezování časových intervalů, ve kterých se přenáší signálové prvky, synchronní
  - ✓ signálový prvek reprezentuje
    - – bit, např. $NRZ\text{-}L$ (digitální signál), FSK (analogový signál)
    - – několik bitů, např. $QPSK$
    - – část bitu, např. $Manchester$ – 1 bit = 2 po sobě jdoucí prvky signálu
  - ✓ při 10 Mb/s je přenosový interval 0,1 μs;
    drift časovače přijímače 1%, tj. 10 ns/bit, posouvá střed bitového intervalu posouvá po 50 bitech na jeho okraj
- 2 řešení ztráty synchronnosti
  - ✓ Synchronní přenos
  - ✓ Asynchronní přenos

Asynchronní přenos

- data se vysílají po znacích (5-ice, 7-ice, 8-ice, . . . bitů)
- resynchronizace se zajišťuje na počátku přenosu znaku,
  po dobu přenosu znaku se synchronnost běhu času udrží
  - ✓ mezi přenosy znaků se vkládají přenosy synchronizačních značek
  - ✓ pokud přijímač nepřijímá znak, vyhledává přechod $1 \longrightarrow 0$
  - ✓ jakmile ho rozpozná,
    vzorkuje signál po dobu 5 (7, 8, . . .) bitových intervalů
  - ✓ pak vyhledává přechod $1 \longrightarrow 0$, . . .
- jednoduché řešení, levné řešení, typicky do desítek Kb/s,
  ex. obvod UART (*Universal Asynchronous Receive/ Transmit*)
- na přenos 1 znaku se přidává
  cca 20% režijního času (2 až 3 bitové intervaly)
  - ✓ efektivní metoda při „řídkém" přenosu znaků –
    např. vstup z klávesnice

Asynchronní přenos, 2

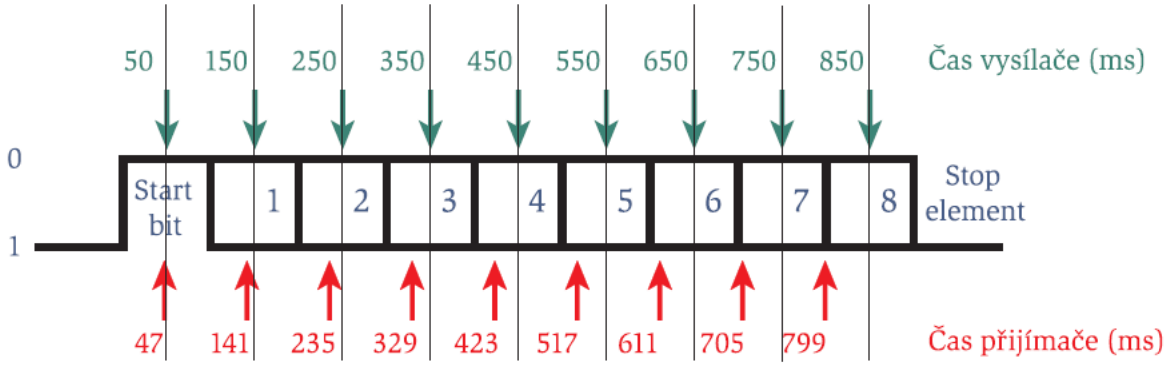


- ✓ je definovaná minimální délka stop bitu (≡ prostoj):

  1 nebo 1,5 nebo 2 bitové intervaly
- ✓ není definovaná maximální délka stop bitu (prostoje)

Asynchronní přenos, dopad chyby časování

- znak 8 bitů, o 5 % pomalejší čas přijímače než vysílače
  - ✓ poslední bit bude u přijímače posunut o 45%, ještě se dobře navzorkuje
- znak 8 bitů, o 6 % rychlejší čas přijímače než vysílače
  při rychlosti přenosu 10 kb/s bude bit. interval = 100 μs,
  přijímač ale bude vzorkovat každých 94 μs:



  - ✓ poslední vzorkovaný bit nebude mít správnou hodnotu a navíc,
  - ✓ pokud bit 7 = 1 a bit 8 = 0, pak bit 8 bude chápán jako *start bit* – chyba formátování (*framing error*)

Asynchronní přenos, bloková (rámcová) synchronizace

- Data se vysílají po částech – rámce, bloky (*frames*)
  - ✓ důvod - chybové řízení, protokolové řízení, . . .
- Vymezování rámců při asynchronním přenosu – STX/ETX, . . .
  - ✓ řídicí znaky v abecedě (ASCII, . . .)
- Jedinečnost hranic bloků – vkládání DLE
  - ✓ STX/ETX se nevyskytují v datech
  - ✓ pokud se vyskytují (binární data),
    jako hranice bloku se vysílají dvojice znaků DLE STX/DLE ETX
  - ✓ pokud se vysílá ekv. znaku DLE, vysílá se dvojice znaků DLE DLE a přijímač dvojici znaků DLE DLE nahrazuje znakem DLE

Synchronní přenos

- Blok dat (rámec, *frame*) se přenáší bez start bitů a stop bitů
- hodiny vysílače a přijímače musí běžet synchronně
- Bitová synchronizace (synchronnost během přenosu bloku)
  - ✓ lze zajistit samostatným časovacím spojem
    – použitelné na krátké vzdálenosti (metry), citlivé na poruchy
  - ✓ zabudováním hodinového signálu do dat
    – kódování typu Manchester nebo
    – odvozování z frekcence nosného (analogového) signálu
- Bloková synchronizace (rozpoznání začátku a konce rámce)
  - ✓ prefixové (*preamble*) a postfixové (*postamble*) příznaky (*flags*)
    např. bitový vzorek 01111110
  - ✓ synchronní přenosy se v současnosti řeší téměř výhradně bitově orientovanými protokoly (data nejsou strukturovaná do znaků)

Rámec, příklad z HDLC

- Typický rámec pro synchronní přenos:

- příznak definovaný protokolem HDLC (*High Data Link Control*):
  01111110
  - ✓ jedinečnost příznaku se zajišťuje při vysílání vkládáním 0 po pěti 1 v poli dat
  - ✓ při příjmu se jedna 0 vyskytující se po pěti 1 vynechává
  - ✓ přijetí vzorku 01111110 indikuje konec rámce
  - ✓ synchronnost mimo přenos rámců – opakované vysílání příznaku
- Synchronní přenos je efektivnější než asynchronní
  - ✓ režie asynchronního přenosu – cca 20%
  - ✓ režie HDLC – 2 příznaky + řídící pole – 6 oktetů, typické datové pole 1000 datových oktetů ⇒ režie 0,6 %

Rámec, příklad z LAN

- preambule – 1010101010. . . 101011
- pevně strukturované záhlaví
  - ✓ adresová část (kdo, komu)
  - ✓ typová část (protokolární informace, data)
  - ✓ délka dat (v oktetech, . . .)
- data
- pevně strukturované ukončení
  - ✓ redundantní data pro detekci chyb

Chyby přenosu dat, jejich detekce a opravy

- fyzická vrstva je vždy předmětem závad, chyb
  (šum, přeslechy, chyby synchronizace, zkresleni, . . .)
- jednotlivé přenášené bity nelze přijímat s nnulovou
  pravděpodobností výskytu chyby
- chyba –
  změna hodnoty bitu v době mezi jeho vysláním a přijetím
- *bit error rate* (BER) = 10^{-6}
  - ✓ zdánlivě velmi dobrý výsledek, ale
  - ✓ při rychlosti přenosu dat 10 Mb/s se vyskytuje chyba každou 1/10 s
- BER = 10^{-7}
  - ✓ při přenosu 1,25 MB binárních dat se vyskytne alespoň 1 chyba
  - ✓ neakceptovatelné pro spolehlivý přenos

Chyby přenosu dat, jejich detekce a opravy, 2

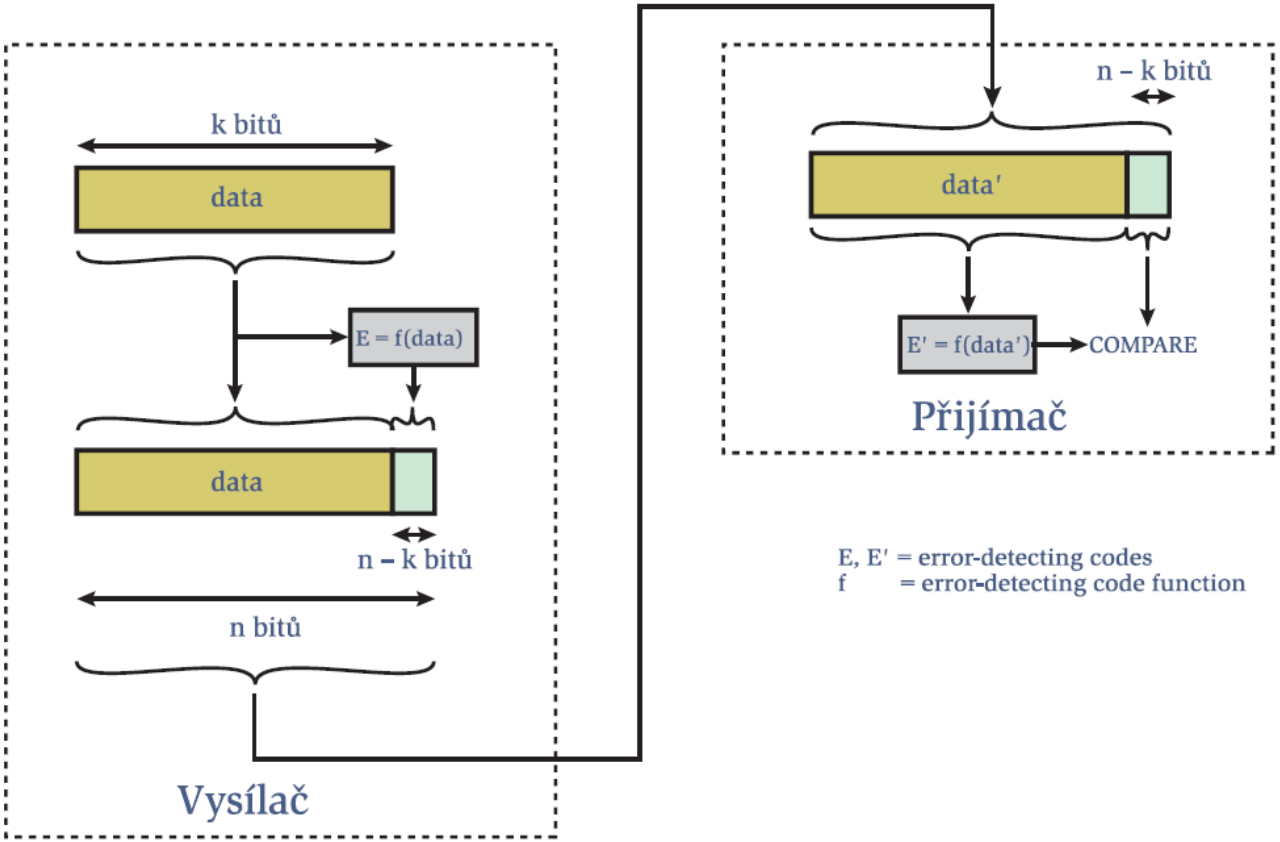
- Nad úrovní fyzické vrstvy musí existovat protokolová vrstva, která chybovost ošetří
  - ✓ typicky, podle RM OSI, vrstva 2 – data link control
- ošetří chybovost =
  - ✓ detekuje chyby a zajistí opakování přenosu
    - – *Error Detection, Automatic Request for Retransmission* (ARQ)
    - – např. test parity a potvrzování
  - ✓ detekuje chyby a pokusí se je (s velkou pravděpodobností úspěšně) opravit při příjmu
    - – *Forward Error Correction*, FEC
    - – např. násobná repetice každého bitu a uplatnění majoritní funkce
    - – vhodné pro případy, kdy zpětná vazba ARQ je nedostupná (simplex) nebo použitelná pouze omezeně při příjmu

Klasifikace chyb

- 1bitové chyby
  - ✓ neovlivňují se sousední bity, bílý šum
- Dávkové chyby (*burst*) délky B
  - ✓ posloupnost B bitů ovlivněná poruchou,
  - ✓ mezi posledním bitem chybové dávky a prvním bitem další chybové dávky je alespoň x bitů přenesených bezchybně
  - ✓ při přenosu rychlostí 10 Mb/s způsobuje impulsní šum délky 1 μs dávkové chyby délky 10 b
  - ✓ při přenosu rychlostí 100 Mb/s způsobuje impulsní šum délky 1 μs dávkové chyby délky 100 b
  - ✓ dávkové chyby jsou důležitým faktorem při vyšších rychlostech přenosu dat

Proces detekce chyb, generické schéma

- Vysílač přidává k datům bity kódu pro detekci chyb(y)



Kódy pro detekci chyb - parita

- výsledný kód znaku s paritním bitem
  musí mít sudý/lichý počet 1 – sudá parita / lichá parita
- paritní zabezpečení nedetekuje sudý počet 1-bitových chyb
- paritní bit se typicky počítá aritmetikou mod 2 (XOR)
- příčná parita – (sudá/lichá) parita znaku (oktetu, . . .)
- podélná parita (sudá/lichá) parita
  (,,po řádech znaků přes celý blok" –
- použití podélné parity v kombinaci s příčnou paritou příp.
  umožní detekci i sudého počtu 1-bitových chyb ve znaku
- obecně platí, že paritní zabezpečení je velmi slabý, málo
  průkazný nástroj pro detekci chyby

Blokové kódování, blokový kód (n,k)

- přidávání paritního bitu k symbolu je příklad techniky blokového kódování
- blokový kód (n, k), $n > k$
  - ✓ přiřazuje k-bitovému datovému slovu n-bitové (vysílané) kódové slovo
  - ✓ z množiny 2^n bitových kombinací vybírá 2^k (validních) kódových slov a tato přiřazuje jednotlivým datovým slovům (těch je 2^k)
  - ✓ kódové slovo některých algoritmů zobrazuje originální datové slovo (má pak formát „původní data + opravný kód"), tzv. systematické kódy
  - ✓ kódové slovo jiných algoritmů nezobrazuje originální datové slovo (má formát jedinečného kódového slova)
  - ✓ přidávání paritního bitu k ASCII znaku je příklad techniky blokového kódu typu (8,7)

Lineární blokové kódy

- Blokový kód (n,k) je lineární, když platí $c = dG$, kde
  - ✓ c je n-bitové slovo kódovaných dat
  - ✓ d je k-bitové slovo dat
  - ✓ G je definiční matice 1 a 0
  - ✓ binární sčítání se řeší v aritmetice $mod\ 2$
- Příklad, tzv. blokový Hammingův kód (7,4)
  - ✓ příslušná definiční matice G = $\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$
  - ✓ kódové slovo odpovídající např. datovému slovu [1010] je $[1010001] = [1010]G$
  - ✓ blokový Hammingův kód (7,4) je systematický kód , datové slovo / kódové slovo: [1010] / [1010001]

Cyklické kódy

- speciální případ lineárních blokových kódů
- Cyklický kód je lineární blokový kód (n,k) s vlastností:
  každý cyklický posuv kódového slova takového kódu je opět kódové slovo tohoto kódu
  - ✓ cyklické kódy jsou velmi oblíbené:
    – efektivně implementují detekci výskytu chyb
    – jsou snadno implementovatelné v hardware
- Příklad: lineární blokový kód (3,2)
  - ✓ s definiční maticí $G = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$
  - ✓ pro datová slova 00, 01, 10 a 11
    generuje kódová slova 000, 110, 101 a 011
  - ✓ každé generované kódové slovo po cyklickém posuvu vlevo či vpravo dává opět validní kódové slovo – kód je cyklický

Cyklické kódy s kontrolní redundancí, CRC

- CRC, *Cyclic Redundancy Check*
- CRC jsou cyklické (tedy i lineární blokové) kódy
- CRC jsou určeny pro detekci chyb v přijaté informaci
- pro blok k bitů dat (zprávu) se vygeneruje $(n\text{-}k)$ bitová posloupnost (FCS, *Frame Check Sequence*) přidávaná ke k bitům zprávy,
  kde n je délka generovaného kódového slova
- data + FCS tvoří kódové slovo o délce n bitů,
  CRC je systematický kód

Cyklické kódy s kontrolní redundancí, CRC

- FCS se generuje se tak, aby vytvořené kódové slovo bylo číslo bezezbytku dělitelné jistým předem daným číslem – klíčem, resp. dělitelem
- FCS se generuje jako zbytek po dělení zprávy klíčem
- klíč hraje roli definiční matice lineárních blokových kódů
- přenáší se kódové slovo, rámec, o délce n bitů
- přijímač přjaté kódové slovo vydělí klíčem (dělitelem)
- je-li zbytek dělení nulový, přenos proběhl bez chyb
- postup ,,trochu odlišný" od principu ze snímku

  Proces detekce chyb, generické schéma

CRC, polynomový model

- zpráva (D) se chápe jako polynom s binárními koeficienty rovnými hodnotám bitů zprávy
  - ✓ pokud $D = 11001_2$,
    pak její reprezenatce v polynomovém modelu je
    $D(X) = X^4 + X^3 + 1$
  - ✓ polynom odpovídající k-bitovému slovu je řádu $k - 1$
- dělitelé (klíče) jsou definované několika standardy, např.

CRC-12: $G(X) = X^{12} + X^{11} + X^3 + X^2 + X + 1$

CRC-16: $G(X) = X^{16} + X^{15} + X^2 + 1$

CRC-CCITT: $G(X) = X^{16} + X^{12} + X^5 + 1$

CRC-32: $G(X) = X^{32} + X^{26} + X^{23} + X^{22} + X^{+6} + X^{12} +$
$+ X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 +$
$+ X^2 + X + 1$

CRC, polynomový model, 2

- vysílaný n-bitový rámec obsahující k-bitovou zprávu D s připojeným $n-k$-bitovým FCS reprezentuje polynom $T(X) = X^{n-k}D(X) + FCS$,
  - ✓ násobení X^{n-k} posouvá $D(X)$ o $n-k$ bitů vlevo
- Příklad
  - ✓ 10-bitová posloupnost dat [1010100101] má polynomiální reprezentaci $D(X) = X^9 + X^7 + X^5 + X^2 + 1$
  - ✓ nechť FCS je 3-bitová posloupnost [111]
  - ✓ odpovídající kódové slovo [1010100101111] má polynomiální reprezentaci $X^3(X^9 + X^7 + X^5 + X^2 + 1) + X^2 + X + 1 = X^{12} + X^{10} + X^8 + X^5 + X^3 + X^2 + X + 1$.

CRC, polynomový model, 3

CRC proces tvorby n-bitového kódového slova:

- k-bitová zpráva $D(X)$,
  $(n-k)$-bitový klíč (dělitel, zabezpečovací polynom) $G(X)$
- $X^{n-k}D(X)/G(X) = Q(X) + R(X)/G(X)$, podíl + zbytek
- úprava násobením $G(X)$: $X^{n-k}D(X) = Q(X) \times G(X) + R(X)$
- v aritmetice $mod\ 2$ je součet a rozdíl ekvivalentní a tudíž platí
  $X^{n-k}D(X) + R(X) = Q(X) \times G(X)$
- tj. pokud se použije $FCS = R(X)$,
  pak kódové slovo $T(X) = X^{n-k}D(X) + FCS$
  je dělitelné klíčem $G(X)$ beze zbytku

CRC, polynomový model, příklad

- $D = 10011_2$, tedy $D(X) = X^4 + X + 1$
- nechť $G(X) = X^2 + X + 1$
- nechť FCS je 2-bitový:
  - ✓ $x^2(x^4+x+1)/x^2+x+1=$, tj.

$$
\begin{array}{l}
x^6 + 0 \quad + 0 + x^3 + x^2 + 0 + 0 \;/\; x^2 + x + 1 = x^4 + x^3 + 1 \\
x^6 + x^5 + x^4 \\
\hline
\qquad x^5 + x^4 + x^3 + x^2 + 0 + 0 \\
\qquad x^5 + x^4 + x^3 \\
\hline
\qquad\qquad\qquad x^2 + 0 + 0 \\
\qquad\qquad\qquad x^2 + x + 1 \\
\hline
\qquad\qquad\qquad\qquad x + 1
\end{array}
$$

  - ✓ $x + 1$ je reprezentace 11_2

CRC, polynomový model, příklad, 2

- takže rámec $T = D + FCS = 1001111_2$ pak reprezentuje polynom $T(X) = X^6 + X^3 + X^2 + X + 1$
- a polynom $T(X) = X^6 + X^3 + X^2 + X + 1$ je dělitelný klíčem $G(X) = X^2 + X + 1$ bezezbytku

CRC, detekovatelné chyby

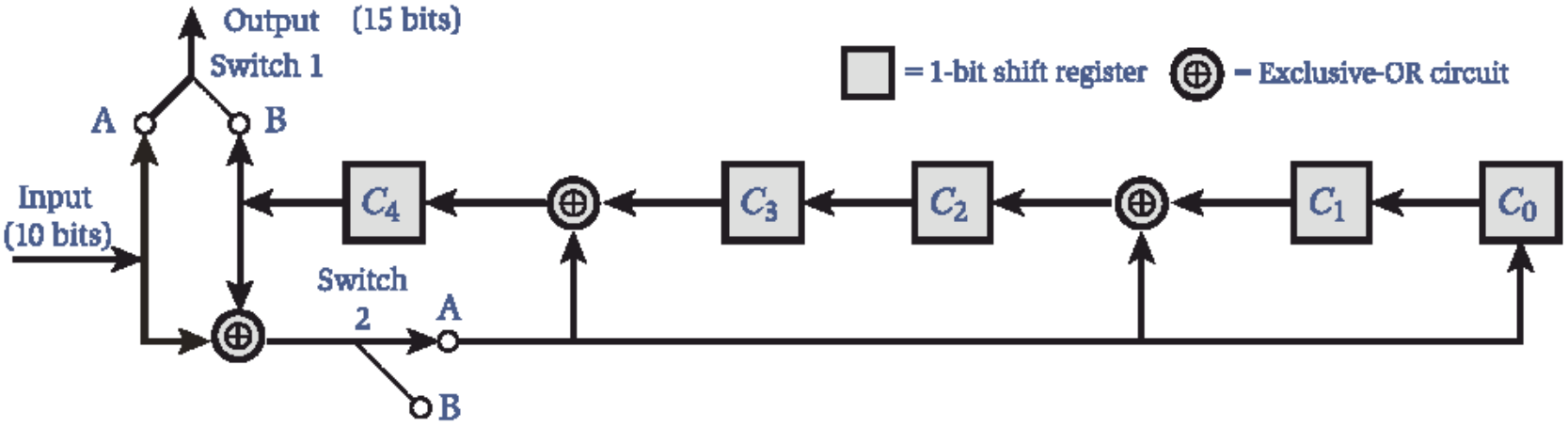
- Platí, že při použití CRC jsou detekovatelné (bez důkazu)
  - ✓ všechny 1bitové chyby pokud $G(X)$ má více než 1 nenulový term
  - ✓ všechny liché počty chyby pokud $G(X)$ obsahuje činitel $X + 1$
  - ✓ všechny dávky chyb pokud délka dávky nepřevýší $n - k$
  - ✓ ...
- CRC kódy jsou velmi užitečné při detekci chyb při přenosu dlouhých rámců

CRC, digitální logika

- řeší se digitální obvod fungující jako generátor FCS
- generátor FCS je dělící obvod tvořený hradly XOR a posuvným registrem
- posuvný registr — řetěz 1-bitových pamětí s výstupem a vstupem
- pamatovaná hodnota paměti = výstupní hodnota
- v pravidelných intervalech je pamatovaná hodnota nahrazovaná vstupní hodnotou
- výstup paměti = vstup sousední paměti, cyklické pořadí

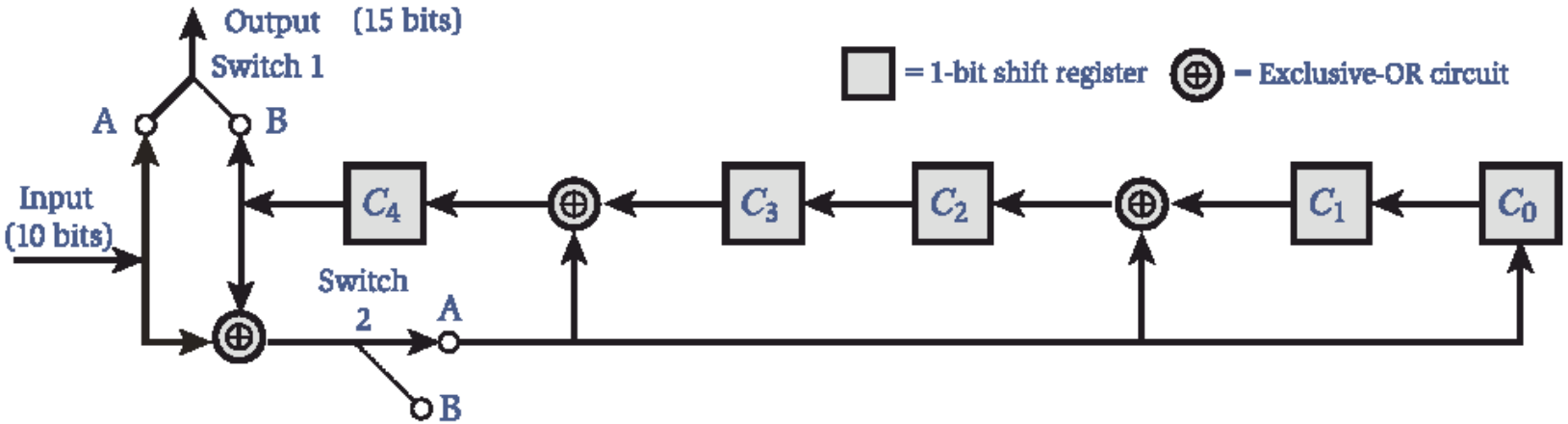
CRC, digitální logika

- ilustrace zapojení generátoru FCS (pro klíč $X^5 + X^4 + X^2 + 1$)
- počet pamětí v generátoru udává řád nejvyšší klíče (5)
- umístění hradel XOR udávají ostatní koeficienty klíče
  (X^4 a X^2 říkají, že jsou na výstupech hradel C_3 a C_1,
  1 říká, že hradlo se uplatňuje na každý vstup)



CRC, digitální logika, vysílač

- počátečně jsou paměti posuvného registru vynulované
- při vstupu k-bitové zprávy jsou přepínače v poloze A
- k-bitová zpráva, dělenec, vstupuje do registru krok po kroku
- po ukončení vstupu zprávy registr obsahuje FCS – zbytek po dělení zprávy dělícím polynomem (klíčem)
- přepínače se pro výstup FCS nastaví do polohy B



CRC, digitální logika, přijímač

- použije se stejná logika
- každý bit přicházející zprávy se vsouvá do registru
- pokud je přenos bezchybný,
  po přijetí dat obsahuje registr FCS
- přijetím bitů FCS se registr vynuluje

Kódy pro opravu chyb, Forward Error Correction, FEC

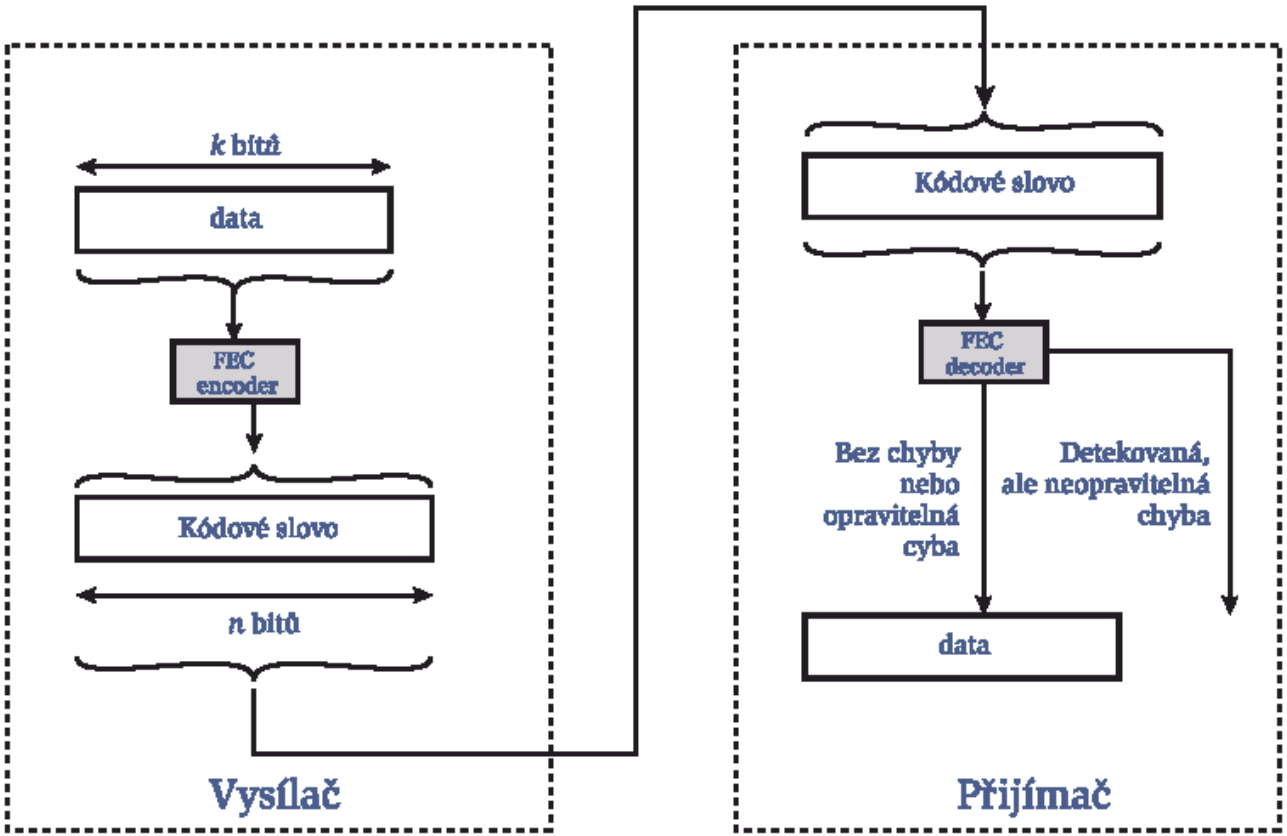
- Oprava detekované chyby se řeší opakováním vyslání (rámce)
  - ✓ opakování vysílání rámce musí být iniciováno vhodným protokolem (v datové vrstvě)
  - ✓ handshaking, sliding window, . . . (viz příští přednáška)
- Protokolární řešení opravy chyby retransmisí rámce je pro mnohé systémy neakceptovatelné, např.
  - ✓ frekvence výskytu chyb je vysoká, bylo by mnoho retransmisí
  - ✓ zpoždění přenosem rámce může být větší než doba vysílání rámce (satelity, řádově stovky ms), byla by nutná retransmise více rámců
  - ✓ pro bezdrátové přenosy v rámci mobilní telefonie
- V takových prostředích se přenosové chyby musí opravovat na základě přijaté informace

Kódy pro opravu chyb, Forward Error Correction, FEC

- Forward …
  - ✓ vysílač připravuje kódování pro chybové řízení před předáním bitů přijímači
- techniky pro přidávání redundance datových bitů (pro opravu, pro detekci chyb)
  - ✓ blokové kódování
  - ✓ konvoluční kódování (mimo naši sféru zájmu)

Proces opravy chyb, generické schéma

- FEC, *Forward Error Correction*, generický model:



Proces opravy chyb, příklad

- Repetiční kód
  - ✓ každý bit se zopakuje vícekrát – např. 2x
  - ✓ 0 se vysílá jako 000, 1 se vysílá jako 111,
  - ✓ předpokládá se výskyt 1-bitových chyb a hlasuje se
  - ✓ přijetí 001, 010, 100 se chápe jako přijetí 000, přijala se 0
  - ✓ přijetí 101, 110, 110 se chápe jako přijetí 111, přijala se 1
  - ✓ dvoubitové chyby způsobí chybnou opravu dat – 001 místo 111 značí 0 a ne 1
- Repetiční kód snižuje datovou kapacitu přenosového kanálu.
  - ✓ Při dvojnásobném opakování každého datového bitu, na 1/3

Princip blokových (opravných) kódů

- každý k-bitový blok dat je kódovaný na n-bitový blok – kódové slovo (*codeword*)
- přenese se kódové slovo
- přijímač přijaté kódové slovo dekóduje a platí
  - ✓ pokud byl přenos bezchybný dešifrováním se dostává původní blok dat
  - ✓ některé chybové vzorky lze detekovat a opravit (na původní blok dat)
  - ✓ některé chybové vzorky lze detekovat, ale ne opravit
  - ✓ velmi málo chybových vzorků nelze detekovat, získávají se chybná data

Princip opravy chyb – blokové opravné kódy

- blokové opravné kódy jsou jednou z více tříd FEC kódů
- (n, k) binární blokový kód doplňuje přenášenou k-bitovou zprávu o $(n - k)$ bitů redundantních dat na kódové slovo
- n-bitová kódová slova mají obvykle, ne však nutně, formát „původní data + opravný kód" (tzv. systematický kód)
- na příjmové straně se z analýzy přijatého kódového slova dedukuje originální vzorek dat (až do výskytu jistého počtu chyb)
- Platí:
  - ✓ počet možných kódových slov = 2^n
  - ✓ počet validních kódových slov = 2^k, reprezentují originální data

Blokové kódy, příklady

- blokový kód $(3,1)$: $k = 1$, $n = 3$
  - ✓ viz Repetiční kód
  - ✓ každý bit se pro vysílání doplní 2 svými kopiemi
  - ✓ 0 se vysílá jako 000, 1 se vysílá jako 111,
  - ✓ předpokládá se výskyt 1-bitových chyb a hlasuje se
  - ✓ přijetí 001, 010, 100 se chápe jako přijetí 000, přijala se 0
  - ✓ přijetí 101, 110, 110 se chápe jako přijetí 111, přijala se 1
  - ✓ dvoubitové chyby způsobí chybnou opravu dat – 001 místo 111 značí 0 a ne 1
- blokový kód $(8,7)$: $k = 7$, $n = 8$
  - ✓ vysílání ASCII znaků s paritním bitem
  - ✓ validní kódové slovo – původní data se správnou paritou

Princip opravy chyb – blokové opravné kódy

zjednodušený princip blokových opravných kódů:

- zavádí se pojem Hammingova vzdálenost, d,
  mezi 2 n-bitovými posloupnostmi –
  – počet bitů, ve kterých se posloupnosti liší,
  (pro 011011 a 110001 je $d = 3$)
  – resp. počet 1 v XOR obou posloupností
- přijatá validní kódová slova se nahrazují originálními daty
- pokud přijaté kódové slovo nepatří mezi validní kódová slova,
  nahrazuje se nejbližším validním kódovým slovem
- pokud určení validního kódového slova není jednoznačné,
  chyba se pouze detekuje, neopravuje se

Princip opravy chyb – blokové opravné kódy, příklad (5,2)

- $k = 2$, $n = 5$
- data (zvolená validní) kódová slova

data	kódová slova
00	00000
01	00111
10	11001
11	11100

- validní kódová slova jsou $4(2^2)$, kódových slov je 32 (2^5)
- přijala se data: 00100, tj. nikoli validní kódové slovo
- nechť nejpravděpodobnější jsou 1-bitové chyby:
  - takže k 00100 ve vzdálenosti 1
  - existuje jediné validní kódové slovo – 00000,
  - přijatý datový blok je tudíž 00
- pokud by existovalo více takových validních kódových slov,
  - chyba by se pouze detekovala a neopravovala

Hammingovy kódy

- navržené pro opravu (nejčastějších) 1-bitových chyb
  - ✓ *Single-error-correcting* (SEC) *code*
  - ✓ *SEC double-error-detecting* (SEC-DED) *code*
- rodina (n, k) blokových opravných kódů s parametry
  - ✓ kódují k-tice datových bitů do n-bitových kódových slov
  - ✓ délka bloku kódového slova: $n = 2^m - 1$
  - ✓ počet datových bitů v bloku: $k = 2^m - m - 1$
  - ✓ počet kontrolních bitů v bloku: $n - k = m$
  - ✓ minimální vzdálenost mezi validními kódovými slovy: $d_{min} = 3$
- např. Hammingův kód (7,4)
  - ✓ $k = 4$, $n = 7$
  - ✓ $n = 2^m - 1$, tj. počet kontrolních bitů $m = 3$
  - ✓ $k = 2^m - m - 1$, počet datových bitů $4 = 8 - 3 - 1$
  - ✓ ex. definice předpisu, jak odvozovat validní kódová slova (viz matice G, snímek „Lineárné blokové kódy")

Hammingovy kódy, další vlastnosti

- kódová slóva by měla být ,,co nejdále od sebe''
  - ✓ chyba během přenosu by získané kódové slovo měla co nejméně přiblížit jinému validními kódovému slovu
- V Hammingovu kódu (7,4) je $d_{min} = 3$, takže se opravují všechny $\lfloor (3-1)/2 \rfloor = 1$-bitové chyby
  - ✓ lze ukázat, že pravděpodobnost správného přenosu 4-bitového bloku dat je $21p^2$, což při $p_e = 10^{-4}$ je cca $2,1.10^{-7}$
  - ✓ to je $\doteq$ výkonu repetičního kódu (3,1)
  - ✓ poměr 4/7 je však výhodnější než poměr 1/3
- -blokové Hammingovy kódy $(2^m - 1, 2^m - m - 1)$ s $d_{min} = 3$ lze nalézt pro všechna m
  - ✓ lze tudíž v kódových slovech délky $2^m - 1$ opravovat 1-bitové chyby
  - ✓ přitom se vzrůstem m poměr k/N se blíží k jedné a režijní náklady relativně klesají