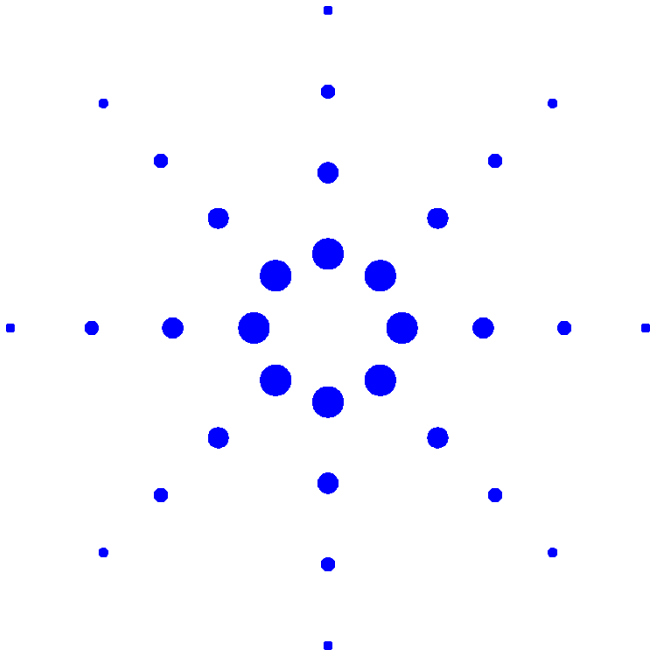
2021



Zabezpečení datových přenosů pomocí CRC

NÁVOD K ÚLOZE

POČÍTAČOVÉ SÍTĚ

ZABEZPEČENÍ DATOVÝCH PŘENOSŮ POMOCÍ CRC

ÚVOD

Cílem úlohy je seznámit se s funkčními principy využití CRC (Cyclic Redundancy Check), tedy s jeho matematickým základem, vlastnostmi a detekčními schopnostmi a praktickou implementací.

CRC je algoritmus výpočtu redundantních dat z dat užitečných (tedy těch, jejichž přenos chceme zabezpečit). CRC kód je odesílán spolu s daty a po jejich přijetí znovu nezávisle vypočítán. Pokud se liší vypočtený CRC kód od přijatého, je jisté, že při přenosu došlo k chybě. Pokud jsou přijatý a vypočtený CRC kód shodné, považujeme data za správně přijatá. Existuje však nízká (ale nenulová) pravděpodobnost, že přijatá data jsou chybná.

Přenášenou datovou posloupnost lze interpretovat například jako polynom s binárními koeficienty. Posloupnost 11000101 lze pak reprezentovat binárním polynomem $x^7 \oplus x^6 \oplus x^2 \oplus 1$. CRC se počítá jako zbytek po dělení datové posloupnosti tzv. generujícím polynomem. V průběhu výpočtu se na koeficienty uplatňuje operace sčítání modulo 2 (tj. platí $1 \oplus 1 = 0$). Určení CRC pouze generujícím polynomem je nejednoznačné, různé algoritmy ho implementují různě – z historických nebo technických důvodů může docházet prohození bajtů, změně pořadí bitů v bajtu, nebo přidání různých bitových posloupností před nebo za vstupní data.

- Někdy se před výpočtem před vstupní data přidává jednička.
- Někdy se před výpočtem za vstupní data přidává počet nul odpovídající stupni generujícího polynomu. CRC vypočtený z posloupnosti data + odeslaný CRC (bez chyb při přenosu) je potom vždy rovný nule.

Číslo za písmeny CRC znamená obvykle stupeň generujícího polynomu, například CRC5 znamená, že generující polynom má nejvyšší mocninu x^5 a tedy délku odpovídající 6 bitům, zbytek po dělení (tedy vlastní CRC) má pak stupeň maximálně o jedničku nižší – v našem případě tedy x^4 a délku odpovídající 5 bitům.

MĚŘENÍ A SIMULACE

V první části úlohy se prostřednictvím ověřování hypotéz seznámíte s detekčními vlastnostmi CRC. Ve druhé části tyto znalosti využijete k ověření jeho spolehlivosti.

DETEKČNÍ VLASTNOSTI

V rámci domácí přípravy se seznamte s detailním postupem výpočtu CRC v kapitole Dodatky. Spusťte program Matlab a otevřete simulační schéma zobrazené na obr. 1. Proveďte následující operace.

- Navrhněte datový rámec s délkou 8 až 12 bitů a vložte jej do bloku *Frame*.
- Zvolte jednoduchý zabezpečovací polynom (např. CRC3, CRC4 – viz Dodatky) a uložte koeficienty shodně do bloků *CRC Generator* a *CRC Syndrome Detector*.
- V pracovním prostředí Matlabu definujte proměnné frame_*length* a *CRC_degree* a uložte do nich hodnoty odpovídající délce dat a stupni generujícího polynomu.
- Pro zvolený datový rámec a generující polynom vypočtěte odpovídající CRC.
- Chyby přenosového kanálu generuje blok *Error vector*. Jako první zvolte vektor bez chyb tj. 000……0, poté s jednou chybou. Délku chybového vektoru určuje součet délek přenosového rámce a stupeň generujícího polynomu (nezaměňujte stupeň polynomu a jeho délku).

- Na zobrazovacích blocích se ujistěte o správné činnosti simulačního schématu.

Simulačně ověřte platnost následujících hypotéz pro všechny relevantní kombinace chyb tam, kde je to možné. Dále se platnost těchto hypotéz můžete pokusit obecně dokázat v rámci přípravy protokolu.

- Je-li chybový vektor posunem generujícího polynomu (násobení obecnou mocninou), chyba není detekována.
- Obecněji, je-li chybový vektor beze zbytku dělitelný generujícím polynomem, chyba není detekována.
- Každá jednonásobná chyba je detekována, pokud má generující polynom koeficient 1 u x^0 a zároveň má alespoň jeden další člen.
- Pokud je generující polynom beze zbytku dělitelný polynomem $x \oplus 1$, pak detekuje jakýkoliv lichý počet chyb.
- Pokud $x^i \oplus 1$ není beze zbytku dělitelné generujícím polynomem pro všechna $i \in \langle 1, n-1 \rangle$, kde n je délka kódového slova, pak jsou detekovány všechny dvojnásobné chyby.
- Pokud je chybový vektor typu $x^j(x^t \oplus \ldots \oplus 1)$, kde t je menší nebo rovno stupni generujícího polynomu (shluk chyb s délkou menší nebo rovnou počtu bitů CRC), pak jsou všechny takovéto chyby detekovány.

V rámci přípravy protokolu obecně dokažte platnost následujících tvrzení.

- Pokud je chybový vektor shlukovou chybou s délkou rovnou délce generujícího polynomu (vektor typu $x^j(x^{t-1} \oplus \ldots \oplus 1)$, kde t-1 = k je rovno stupni generujícího polynomu), pak pravděpodobnost, že chyba nebude detekována, je $2^{-(k-1)}$.
- Pokud je chybový vektor shlukovou chybou s délkou t vyšší než k+1, k je stupeň generujícího polynomu, pak pravděpodobnost, že chyba nebude detekována, je 2^{-k}.

SPOLEHLIVOST

V Simulinku otevřete přenosový řetězec zobrazený na obr. 2. Blok *Bernoulli Binary Generator* generuje náhodnou posloupnost dat. Blok *Binary Symmetric Channel* generuje chyby bitů s nastavenou pravděpodobností. Ve vámi zvolené konfiguraci přenosové cesty a pravděpodobnosti chyby bitu (zkuste např. 0,1‰, 1‰ a 1%) určete četnost jevu, kdy není detekován příjem chybného rámce (přijat pozměněný rámec se správným CRC).

DODATKY

V rámci domácí přípravy se seznamte s detailním postupem výpočtu CRC, se schématy simulačních modelů a s jejich nastavením.

VÝPOČET CRC

Data: 10011101 Reprezentace: $x^7 \oplus x^4 \oplus x^3 \oplus x^2 \oplus 1$

Generující polynom: $x^3 \oplus x \oplus 1$ (CRC3)

Nejprve posuneme data o stupeň generujícího polynomu (vynásobíme koeficientem x^3, tedy $x^7 \rightarrow x^{10}$ atd.). Poté vydělíme datový polynom generujícím polynomem. Zbytek po tomto dělení je hledané CRC.

$x^{10} \oplus x^7 \oplus x^6 \oplus x^5 \oplus x^3 : x^3 \oplus x \oplus 1 = x^7 \oplus x^5 \oplus 1$

$x^{10} \oplus x^8 \oplus x^7$

$x^8 \oplus x^6 \oplus x^5 \oplus x^3$

$x^8 \oplus x^6 \oplus x^5$

x^3

$x^3 \oplus x \oplus 1$

$\boldsymbol{x \oplus 1}$

Délka CRC je rovna stupni generujícího polynomu, výsledné CRC je tedy **011**.

Odeslaná posloupnost bude 10011101011, tedy $x^{10} \oplus x^7 \oplus x^6 \oplus x^5 \oplus x^3 \oplus x \oplus 1$.

Ověření správnosti příjmu probíhá obdobně.

$x^{10} \oplus x^7 \oplus x^6 \oplus x^5 \oplus x^3 \oplus x \oplus 1 : x^3 \oplus x \oplus 1 = x^7 \oplus x^5 \oplus 1$

$x^{10} \oplus x^8 \oplus x^7$

$x^8 \oplus x^6 \oplus x^5 \oplus x^3 \oplus x \oplus 1$

$x^8 \oplus x^6 \oplus x^5$

$x^3 \oplus x \oplus 1$

$x^3 \oplus x \oplus 1$

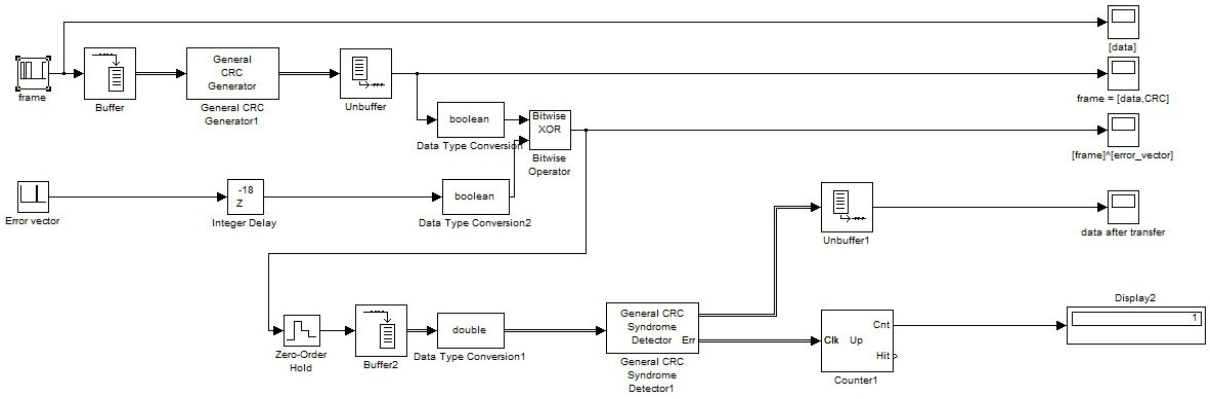
$\boldsymbol{0}$

Výsledný zbytek po dělení je 0, chyba tedy nebyla detekována.

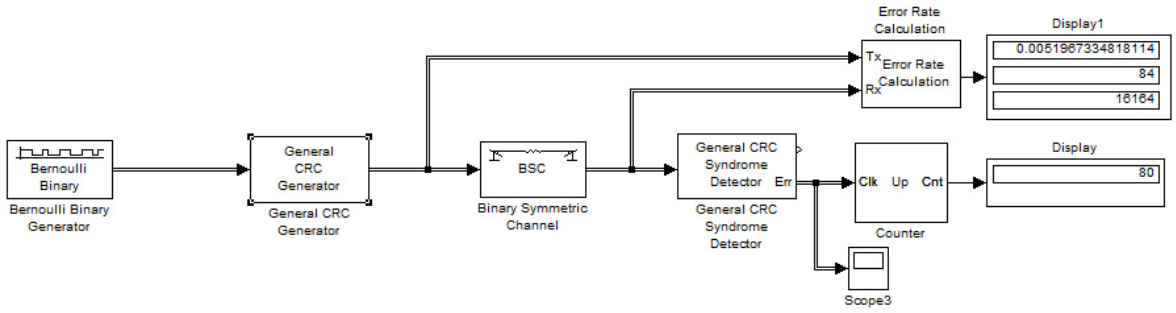
PŘÍKLADY POLYNOMŮ

Název	Polynom
CRC-1	x + 1 (most hardware; also known as parity bit)
CRC-3	x^3 + x + 1
CRC-4-ITU	x^4 + x + 1 (ITU-T G.704, p. 12)
CRC-5-EPC	x^5 + x^3 + 1 (Gen 2 RFID[15])
CRC-5-ITU	x^5 + x^4 + x^2 + 1 (ITU-T G.704, p. 9)
CRC-5-USB	x^5 + x^2 + 1 (USB token packets)
CRC-6-ITU	x^6 + x + 1 (ITU-T G.704, p. 3)
CRC-7	x^7 + x^3 + 1 (telecom systems, ITU-T G.707, ITU-T G.832, MMC, SD)
CRC-8-CCITT	x^8 + x^2 + x + 1 (ATM HEC), ISDN Header Error Control and Cell Delineation ITU-T I.432.1 (02/99)
CRC-8-Dallas/Max^im	x^8 + x^5 + x^4 + 1 (1-Wire bus)
CRC-8	x^8 + x^7 + x^6 + x^4 + x^2 + 1
CRC-8-SAE J1850	x^8 + x^4 + x^3 + x^2 + 1
CRC-8-WCDMA	x^8 + x^7 + x^4 + x^3 + x + 1[16]
CRC-10	x^10 + x^9 + x^5 + x^4 + x + 1 (ATM; ITU-T I.610)
CRC-11	x^11 + x^9 + x^8 + x^7 + x^2 + 1 (FlexRay[17])
CRC-12	x^12 + x^11 + x^3 + x^2 + x + 1 (telecom systems[18][19])
CRC-15-CAN	x^15 + x^14 + x^10 + x^8 + x^7 + x^4 + x^3 + 1
CRC-16-IBM	x^16 + x^15 + x^2 + 1 (Bisync, Modbus, USB, ANSI X3.28, many others; also known as CRC-16 and CRC-16-ANSI)
CRC-16-CCITT	x^16 + x^12 + x^5 + 1 (X.25, HDLC, XMODEM, Bluetooth, SD, many others; known as CRC-CCITT)
CRC-16-T10-DIF	x^16 + x^15 + x^11 + x^9 + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1 (SCSI DIF)
CRC-16-DNP	x^16 + x^13 + x^12 + x^11 + x^10 + x^8 + x^6 + x^5 + x^2 + 1 (DNP, IEC 870, M-Bus)
CRC-16-DECT	x^16 + x^10 + x^8 + x^7 + x^3 + 1 (cordless telephones)[21]
CRC-24	x^24 + x^22 + x^20 + x^19 + x^18 + x^16 + x^14 + x^13 + x^11 + x^10 + x^8 + x^7 + x^6 + x^3 + x + 1 (FlexRay[17])
CRC-24-Radix^-64	x^24 + x^23 + x^18 + x^17 + x^14 + x^11 + x^10 + x^7 + x^6 + x^5 + x^4 + x^3 + x + 1 (OpenPGP)
CRC-30	x^30 + x^29 + x^21 + x^20 + x^15 + x^13 + x^12 + x^11 + x^8 + x^7 + x^6 + x^2 + x + 1 (CDMA)
CRC-32-IEEE 802.3	x^32 + x^26 + x^23 + x^22 + x^16 + x^12 + x^11 + x^10 + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1 (V.42, Ethernet, MPEG-2, PNG[22], POSIX cksum)
CRC-32C (Castagnoli)	x^32 + x^28 + x^27 + x^26 + x^25 + x^23 + x^22 + x^20 + x^19 + x^18 + x^14 + x^13 + x^11 + x^10 + x^9 + x^8 + x^6 + 1 (iSCSI & SCTP, G.hn payload, SSE4.2)
CRC-32K (Koopman)	x^32 + x^30 + x^29 + x^28 + x^26 + x^20 + x^19 + x^17 + x^16 + x^15 + x^11 + x^10 + x^7 + x^6 + x^4 + x^2 + x + 1
CRC-32Q	x^32 + x^31 + x^24 + x^22 + x^16 + x^14 + x^8 + x^7 + x^5 + x^3 + x + 1 (aviation; AIXM[23])
CRC-64-ISO	x^64 + x^4 + x^3 + x + 1 (HDLC — ISO 3309, Swiss-Prot/TrEMBL; considered weak for hashing[24])
CRC-64-ECMA-182	x^64 + x^62 + x^57 + x^55 + x^54 + x^53 + x^52 + x^47 + x^46 + x^45 + x^40 + x^39 + x^38 + x^37 + x^35 + x^33 + x^32 + x^31 + x^29 + x^27 + x^24 + x^23 + x^22 + x^21 + x^19 + x^17 + x^13 + x^12 + x^10 + x^9 + x^7 + x^4 + x + 1 (as described in ECMA-182 p. 51)

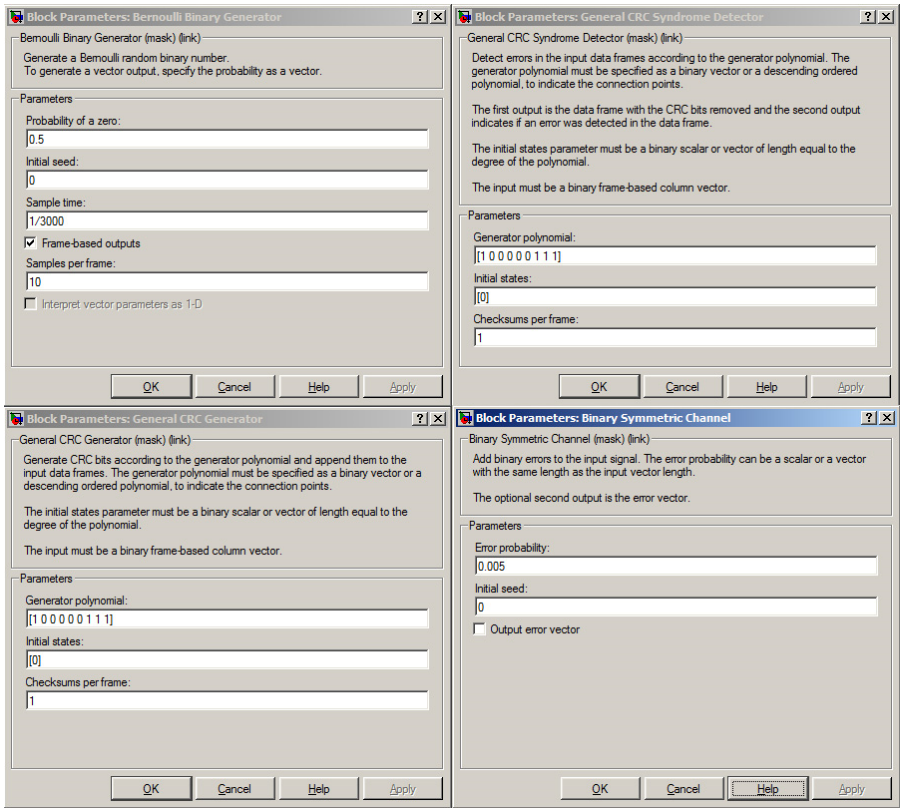
SIMULAČNÍ MODELY



Obr. 1: Simulace zabezpečení pomocí CRC



Obr. 2: Simulace generování náhodných chyb v přenosovém řetězci při zabezpečení pomocí CRC



Obr. 3: Nastavení bloků simulačního řetězce

Blok *Bernoulli Binary Generator*:

- *Probability of zero* – pravděpodobnost nuly v generovaných datech, nastavit 0.5.
- *Initial seed* – počáteční stav, nastavit např. 0.
- *Sample time* – délka trvaní jednoho vzorku → počet vzorků za sekundu, v kombinaci s délkou simulace nastavovat tak aby se přeneslo dostatečné množství rámců (aby se v přenesených datech vyskytly nedetekované chyby).
- *Samples per frame* – počet bitů v jednom datovém rámci.

Blok *Binary Symmetric Channel*:

- *Error probability* – pravděpodobnost chyby bitu v přenosu.

Blok *CRC Generator*:

- *Generator polynomial* – generující polynom, nutno nastavit stejně i v bloku *CRC Detector*.
- *Checksums per frame* – nastavit 1.