UPS 2012/2013

Cvičení 7

Obsah

- Chyby
- Hammingova vzdálenost
- Parita
- CRC

Chyba přenosu

- Dojde ke ztrátě či záměně dat
  - Zkreslení signálu, rušení, šum
- Bezpečnostní kódy
  - Detekce chyb x oprava chyb
- Uvažuje symetrický binární přenosový kanál bez paměti
  - Symetrický: 0/1 se přenáší se stejnou pravděpodobností
  - Binární: Přenáší se 0/1
  - Bez paměti: Nezáleží co se přeneslo v předchozím kroku

Chyba při přenosu

- Pravděpodobnost přenosu 1 bitu $P_1 = p_1$
- Pravděpodobnost přenosu N bitů $P_N = p_1^N$
- Příklad:
  - máme SBPKBP, kolik bitů můžeme přenést, aby pravděpodobnost bezchybného přenosu byla 0,9, když pravděpodobnost přenosu 1 bitu je 0,9999 ?

Příklad I

- $P_1 = 0.9999$
- $P_n = 0.9$
- N=?

Příklad II

- $P_1 = 0.9999$
- $P_n = 0.9$
- N=?
- $0.9 = 0.9999^N$

Příklad III

- $P_1 = 0.9999$
- $P_n = 0.9$
- $N = ?$
- $0.9 = 0.9999^N$
- $\ln(0.9) = N \ln(0.9999)$

Příklad IV

- $P_1 = 0.9999$
- $P_n = 0.9$
- $N = ?$
- $0.9 = 0.9999^N$
- $\ln(0.9) = N \ln(0.9999)$
- $N = \ln(0.9) / \ln(0.9999)$

Příklad V

- $P_1 = 0.9999$
- $P_n = 0.9$
- $N = ?$
- $0.9 = 0.9999^N$
- $\ln(0.9) = N \ln(0.9999)$
- $N = \ln(0.9) / \ln(0.9999)$
- $N = 1\,053$

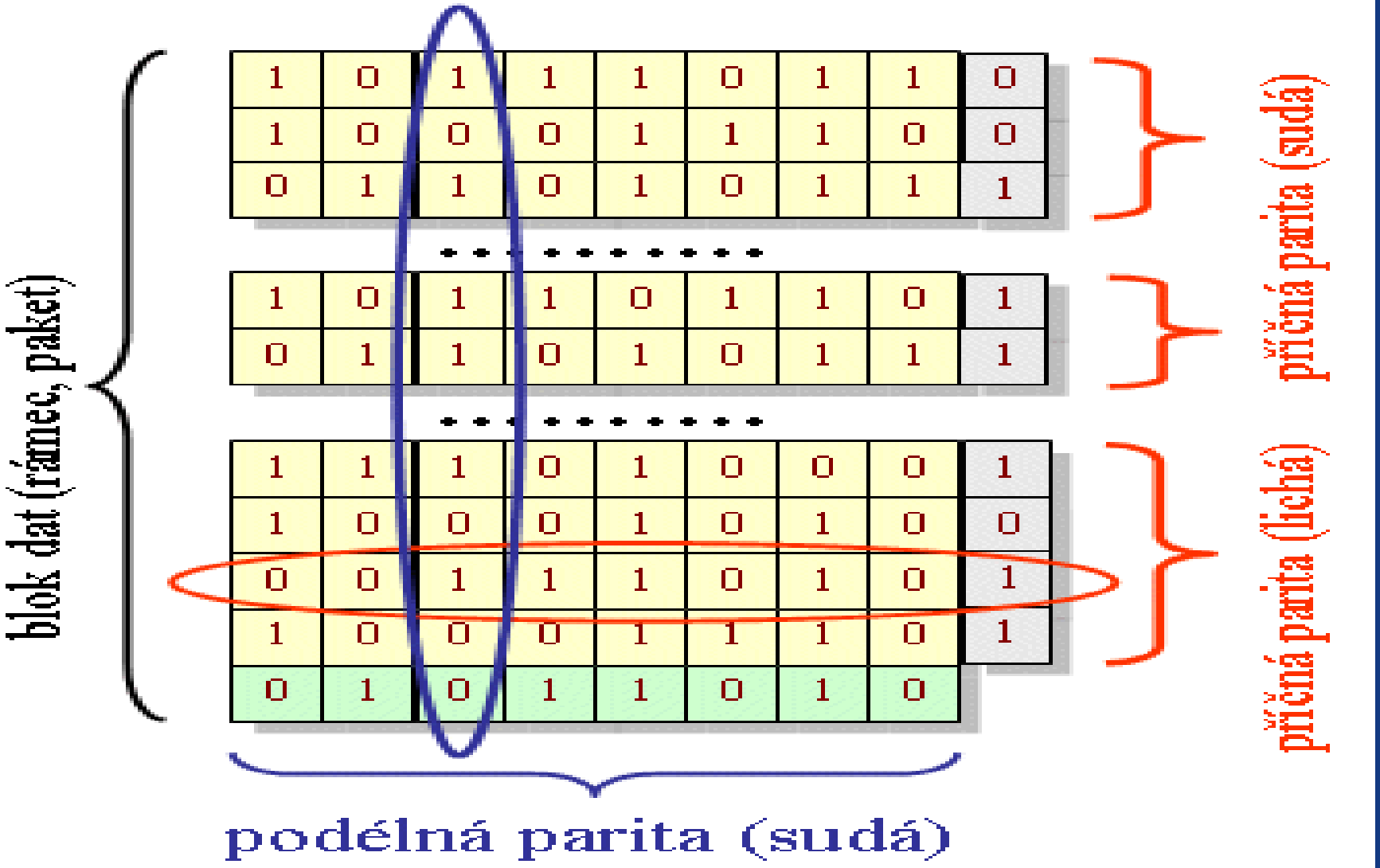
Bezpečnostní kódy

- Přidáme nějaké bity navíc nebo pozměníme data
- Čím více bitů navíc tím účinnější metoda
- Detekční – kontrola zda jsou data správně
- Samoopravné – chybu rozpoznají a opraví

Parita

- Přidáváme jeden paritní bit
- Sudá 0 = sudý počet 1, 1 = lichý počet 1
  - Vždy sudý počet 1 ve zprávě
  - Umí jen detekovat, nevíme co je špatně
- Lichá parita je analogie k sudé
- Příčná parita – paritní bit ke každému slovu
- Podélná parita – přidáváme paritní slovo, zabezpečuje celý blok, lze vyhodnocovat průběžně
- Křížová – kombinace příčné a podélné

Parita


blok dat (rámec, paket)
příčná parita (sudá)
příčná parita (lichá)
podélná parita (sudá)

Checksum

- Kontrolní součet – pro celý blok dat
- Jednotlivé znaky chápeme jako čísla bez znaménka
- Provádíme sčítání modulo 2^8 nebo 2^{16}
- Výsledek je číslo o délce 1 nebo 2 bytů
- Výpočet probíhá postupně
- Po přijetí kontrolní sumy se provede kontrola
- V případě chyby je nutné vyžádat přenos znovu

Hammingův kód (7,4)

- Dovoluje detekovat dvojitou a opravit jednoduchou chybu
- Všechny bitové pozice, jejichž číslo je rovné mocnině 2, jsou použity pro paritní bit (1, 2, 4, 8, 16, 32, …).
- Všechny ostatní bitové pozice náleží kódovanému informačnímu slovu (3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, …).
- Každý paritní bit je vypočítán z některých bitů informačního slova. Pozice paritního bitu udává sekvenci bitů, které jsou v kódovém slově zjišťovány a které přeskočeny.

Hammingův kód

- Pro paritní bit p1 (pozice 1) se ve zbylém kódovém slově 1 bit přeskočí, 1 zkontroluje, 1 bit přeskočí, 1 zkontroluje, atd.
- Pro paritní bit p2 (pozice 2) se přeskočí první bit, 2 zkontrolují, 2 přeskočí, 2 zkontrolují, atd.
- Pro p3 (pozice 4) se přeskočí první 3 bity, 4 zkontrolují, 4 přeskočí, 4 zkontrolují, atd.
- http://www.uai.fme.vutbr.cz/~matousek/TIK/flashB5.html

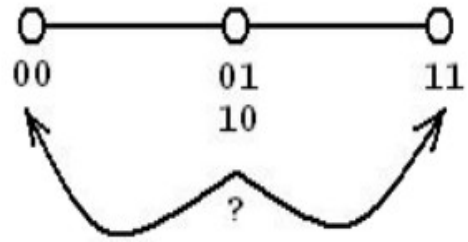
Rozšířený Hammingův kód (8,4)

- Na začátek každého slova přidáme paritu pro celé slovo
- Používá se sudá parita
- Dovoluje opravit jednu chybu, ale detekovat dvě

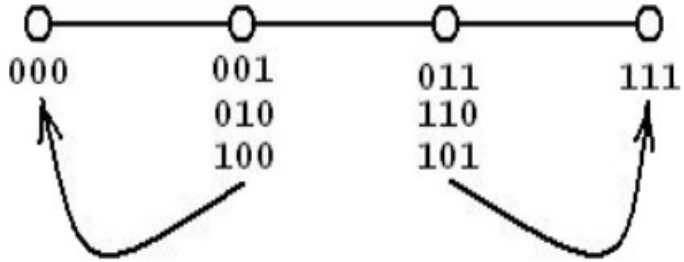
Hammingova vzdálenost I.

- Počet míst v němž se dvě kódová slova liší
  - příklad: 000 a 001 mají vzdálenost 1bit, 010 a 101 mají vzdálenost 3bity
- Charakterizuje odolnost kódu proti poruchám a schopnost identifikovat a případně opravit chyby
- Minimální Hammingova vzdálenost = minimální vzdálenost mezi všemi možnými páry vektorů

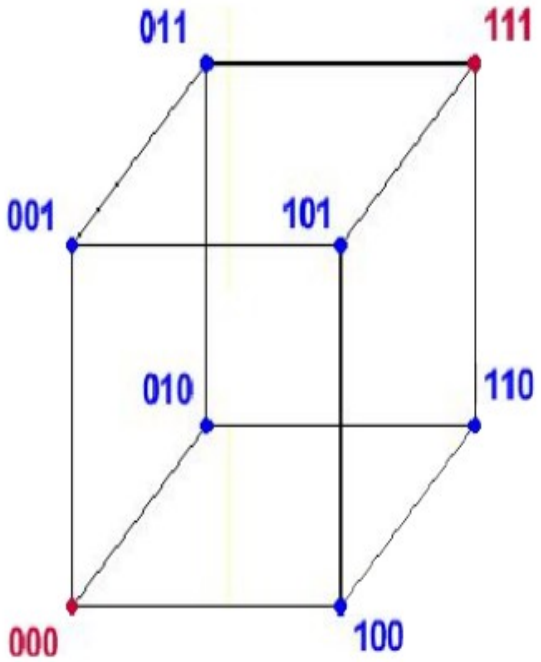
Hammingova vzdálenost II.



Minimální Hammingova vzdálenost kódu je 2.
Jednobitová chyba jde detekovat, ale nelze opravit.



Minimální Hammingova vzdálenost kódu je 3.
Jedno a dvoubitová chyba jdou detekovat.
Opravit lze pouze jednobitovou chybu.



Hammingova vzdálenost III.

- Pro detekci n bitovych chyb platí
  - $d_{min} => n+1$; tj $n <= d_{min} - 1$
- Pro detekci a korekci n bitovych chyb platí
  - $d_{min} => 2n+1$; tj $n <= (d_{min} - 1)/2$
  - D(000,001) = 1, nevíme nic
  - D(000,101) = 2, poznáme jednu chybu
  - D(000,111) = 3, 2 poznáme, 1 opravíme

Cyklické kódy CRC

- Cyklický redundantní součet
- CRC s počítá před operací kde čekáme chybu
- Odesílá se společně s daty
- Po přenosu se spočítá znovu a rozhodne se
- Někdy je možné chybu i opravit
- Např. Generující polynomy $G(x)=x^4+x+1$, tedy $(10011)_2$
- Délka zabezpečení se rovná stupni generujícího polynomu

Cyklické kody CRC

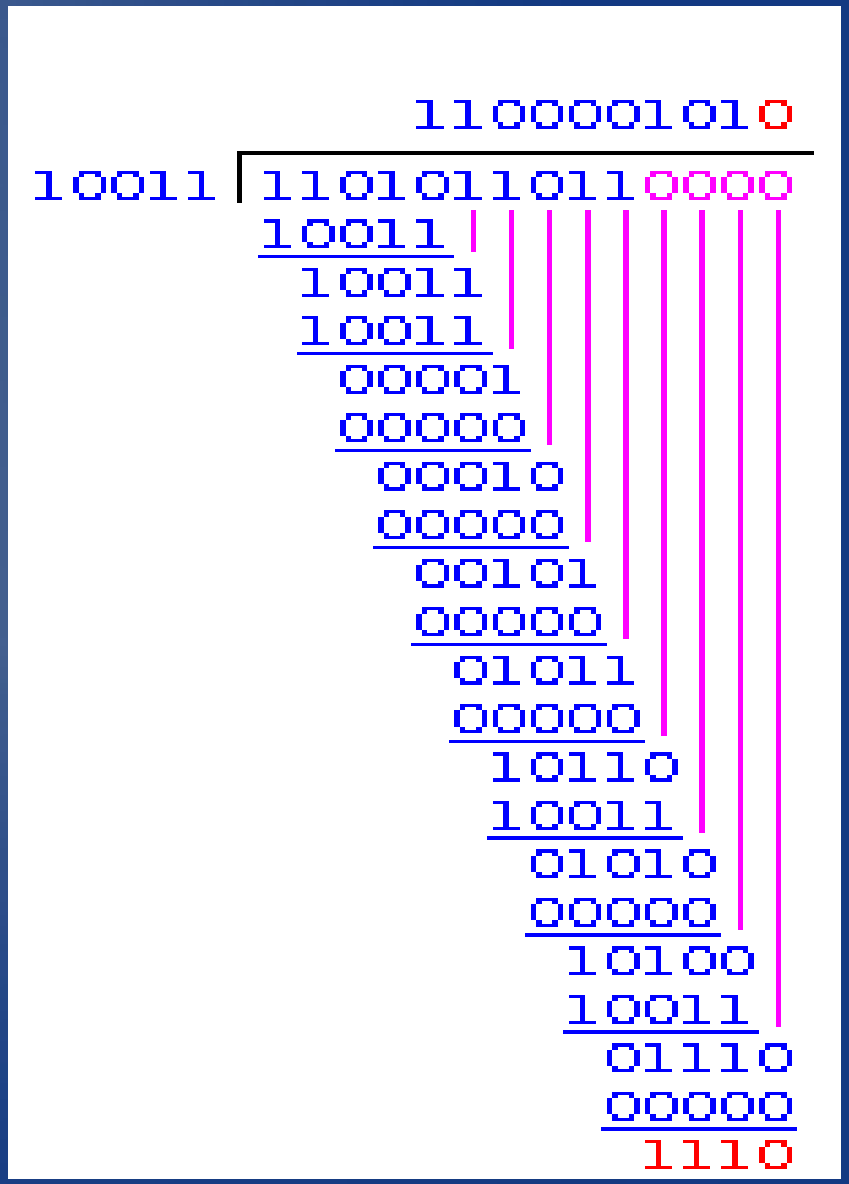
- Vypočteme zbytek po dělení R(x)=M(x)/G(x)
- Odesíláme T(x) = M(x) | R(x)
- Po přijetí provedeme T(x)/G(x)
- Pokud je výsledek (zbytek) nula, je přenos v pořádku
- Označení jako CRC 16, 32 atp. podle stupně polynomu G(x)
- http://en.wikipedia.org/wiki/Cyclic_redundancy_check

CRC příklad

- M(x)=1101 0110 11
- G(x) = 10011 = $x^4 + x + 1$
- Délka zabezpečení je rovna stupni generujícího polynomu, tj. k=4. Vypočteme zbytek po dělení M(x) * x^4
- 11 0101 1011 0000 / 10011
- R(x) = 1110

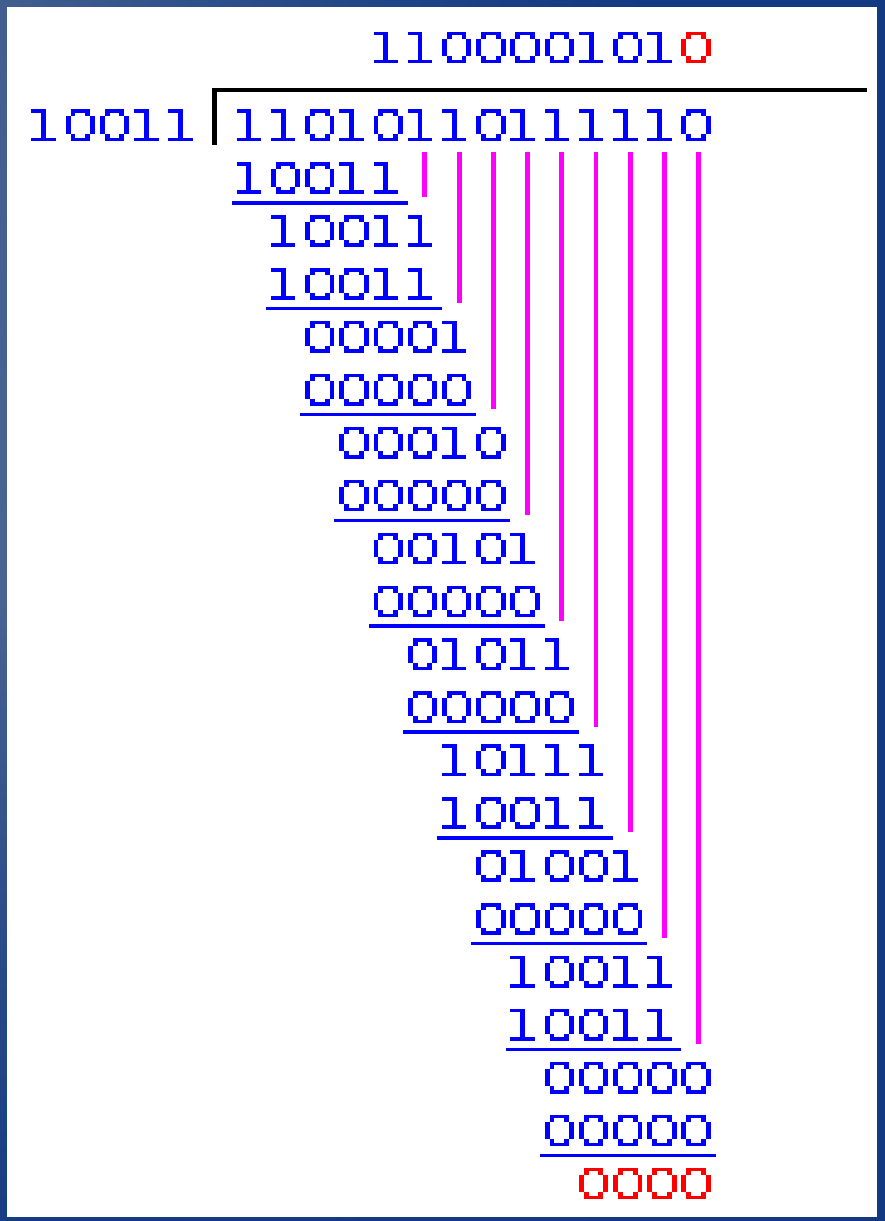
CRC příklad

- Postup dělení
  - Stejné jako dělení pod sebe
  - Operaci odečítání nahrazuje operace XOR
    - 1 XOR 1 = 0
    - 1 XOR 0 = 1
    - 0 XOR 1 = 1
    - 0 XOR 0 = 0
- Odesíláme M(x) | R(x)
  - 1101 0110 11 | 1110



CRC příklad

- Ověření přijaté zprávy

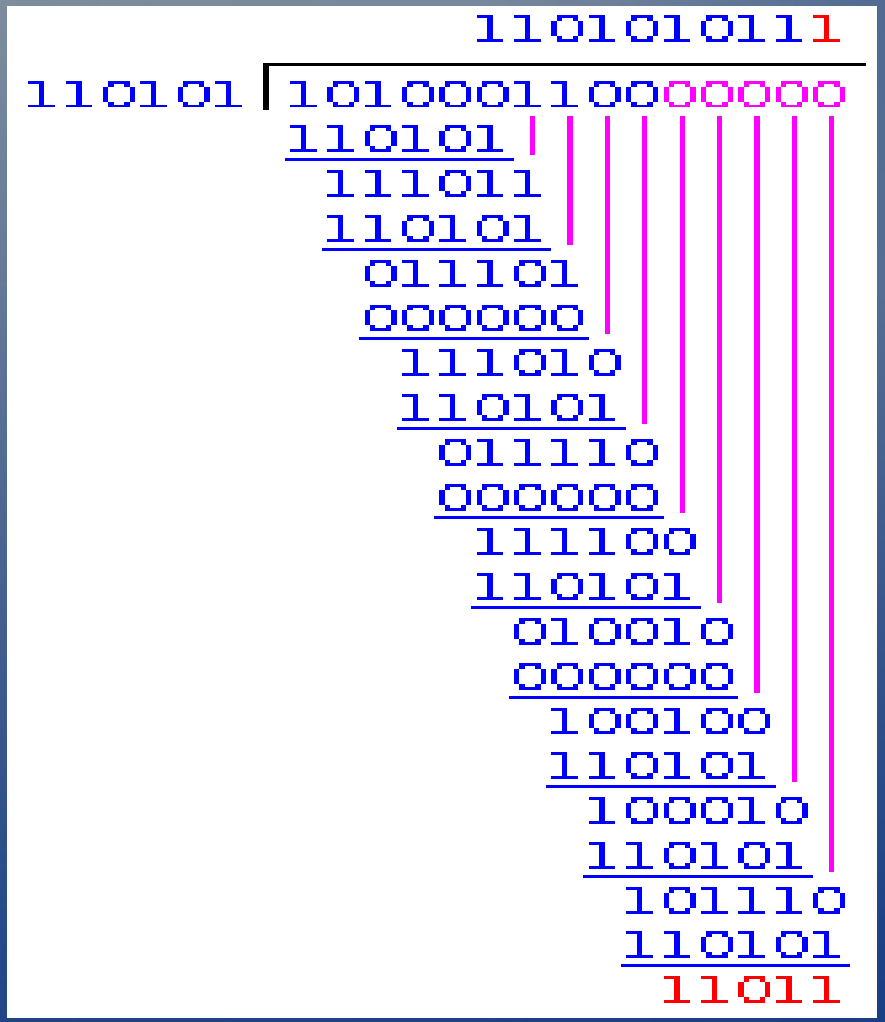


CRC samostatně

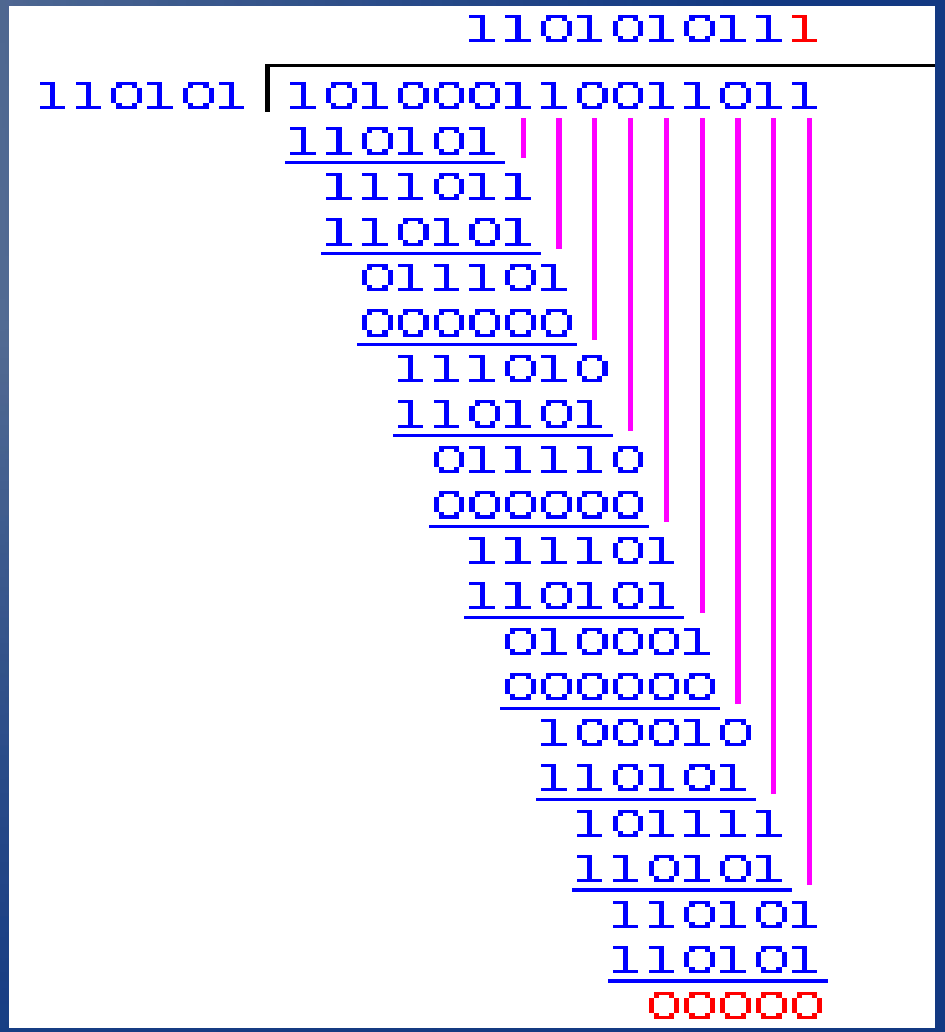
- M(x) = 10 10 00 11 00
- M`(x) = 10 10 00 11 00 00 00 0
- G(x) = 11 01 01 = $x^5 + x^4 + x^2 + 1$
- R(x) =
- T(x) =

CRC samostatně

Zabezpečení



Kontrola



CRC samostatně

- Zkoušejte si na
  - http://www.macs.hw.ac.uk/~pjbk/nets/crc/