Рачунарске мреже (вежбе – термин 1)

Слој везе података (Data Link Layer)

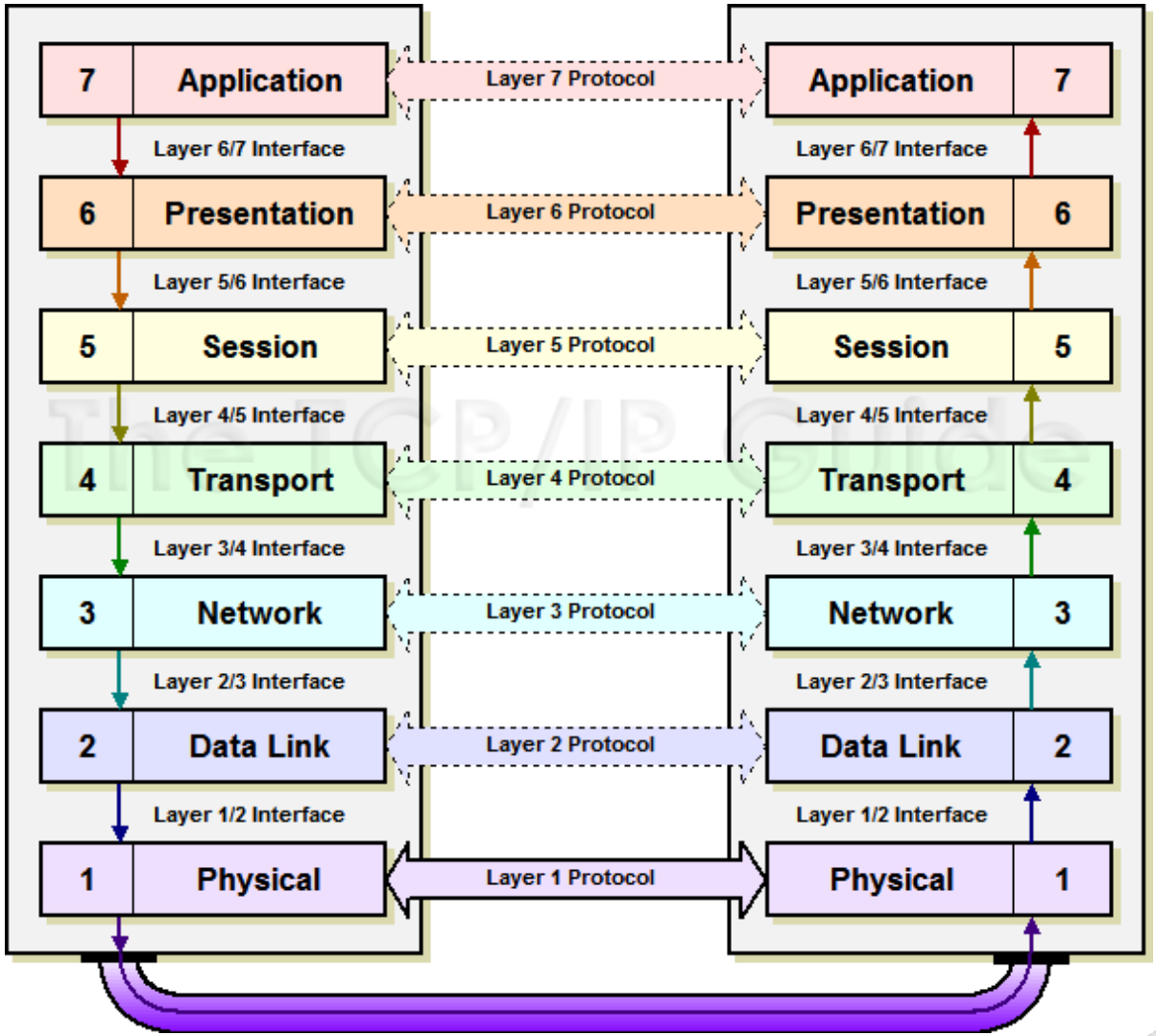
Садржај

- 1.час
  - Упознавање са садржајем вежби из предмета Рачунарске мреже и начином полагања предиспитних обавеза
  - Слој везе података:
    - Намена слоја везе података
    - Уоквиравање
- 2.час
  - Откривање и исправљање грешака:
  - Хамингов код
  - CRC код

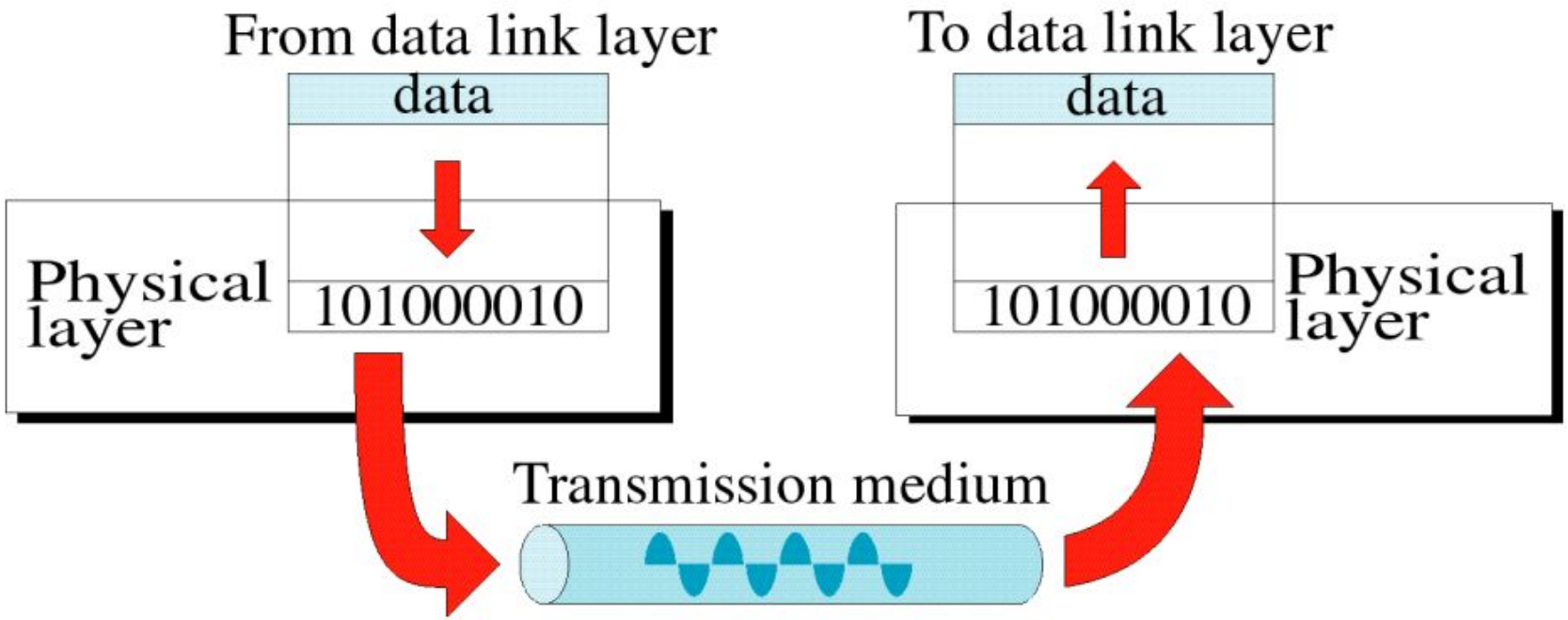
Начин полагања предмета

- 1. колоквијум (писани тест) – 23 поена
- 2. колоквијум (рад за рачунаром)– 23 поена
- Присуство на предавањима и вежбама – 4 поена
- Укупно 50 поена (услов за полагање испита 26 поена)
- Литература: imiE-Learning - ОАС информатике – 2.година - Летњи семестар - Рачунарске мреже

OSI модел


7
Application
Layer 7 Protocol
Application
7
Layer 6/7 Interface
Layer 6/7 Interface
6
Presentation
Layer 6 Protocol
Presentation
6
Layer 5/6 Interface
Layer 5/6 Interface
5
Session
Layer 5 Protocol
Session
5
Layer 4/5 Interface
Layer 4/5 Interface
4
Transport
Layer 4 Protocol
Transport
4
Layer 3/4 Interface
Layer 3/4 Interface
3
Network
Layer 3 Protocol
Network
3
Layer 2/3 Interface
Layer 2/3 Interface
2
Data Link
Layer 2 Protocol
Data Link
2
Layer 1/2 Interface
Layer 1/2 Interface
1
Physical
Layer 1 Protocol
Physical
1

Физички слој



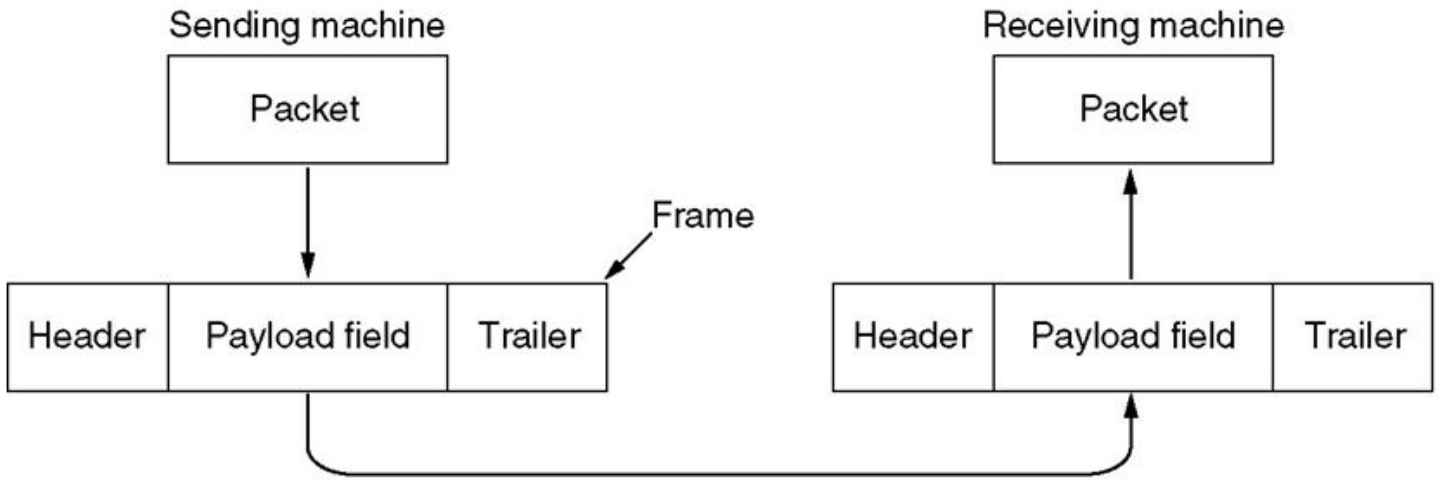
Слој везе података (Data Link Layer)

- Намена слоја везе података, основна јединица преноса
- Уоквиравање
  - Пребројавање бајтова
  - Индикаторски бајтови уз уметање знакова
  - Почетни и завршни индикатори уз уметање битова

Слој везе података (Data Link Layer)

- Сврха слоја везе података је постизање поузданости и ефикасности комуникација између два суседна рачунара, тј. обезбеђивање поузданих и ефикасних услуга преноса битова на мрежни слој.
- Функције слоја везе података су следеће:
  - Добро дефинисан услужни интерфејс према мрежном слоју
  - Обрада грешака при преносу података
  - Управљање током података
- Основна јединица преноса је оквир (Frame), који садржи заглавље (Header), поље за корисничке податке и завршни блок (Trailer).

Слој везе података (Data Link Layer)



Уоквиравање

- За пружање услуга мрежном слоју, слој везе користи услуге физичког слоја.
- Физички слој прихвата ток необрађених података и покушава да их испоручи, али без гаранције поузданости. Број примљених битова може бити мањи, једнак или већи од броја послатих битова и њихове вредности могу бити промењене.
- **Слој везе треба да открије грешке у преносу и да их исправи ако је потребно.**
- Слој везе обично дели ток података у оквире (frames) коначне величине. Подела података у оквире није тако једноставна. Једна од идеја је да се убаци празан временски интервал између оквира, али то је лоше због недостатка синхронизације, као и губљења драгоценог времена.

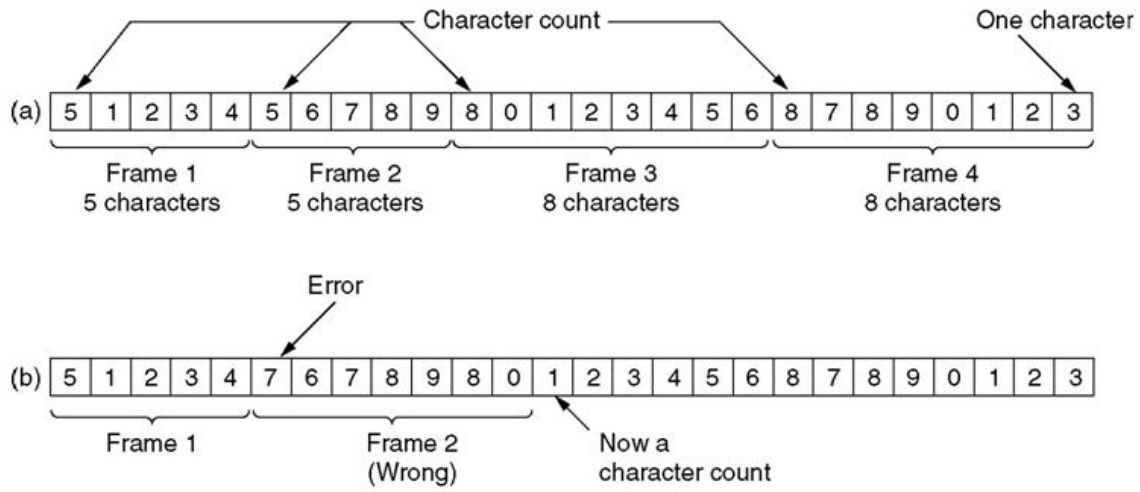
Методе уоквиравања

- Пребројавање знакова
- Употреба почетних и завршних индикатора уз уметање бајтова
- Употреба почетних и завршних индикатора уз уметање битова

Пребројавање знакова

- Пребројавање знакова је једна од најстаријих метода уоквиравања. У заглављу оквира постоји поље са бројем знакова унутар датог оквира. На следећој слици приказана су 4 оквира дужине 5, 5, 8 и 8 знакова респективно.
- Проблем може да настане услед погрешног очитавања броја знакова оквира што доводи до потпуног губитка синхронизације.

Пребројавање знакова



A character stream. (a) Without errors. (b) With one error.

Задаци

- Задатак 1.
  - Дата је секвенца бајтова добијене после уоквиравања методом пребројавања знакова:

    4 9 8 2 5 2 3 8 19 7 2 4 11 13 2 18 3 2 20
  - Како изгледа оригинална секвенца (када се одстране маркери оквира)?

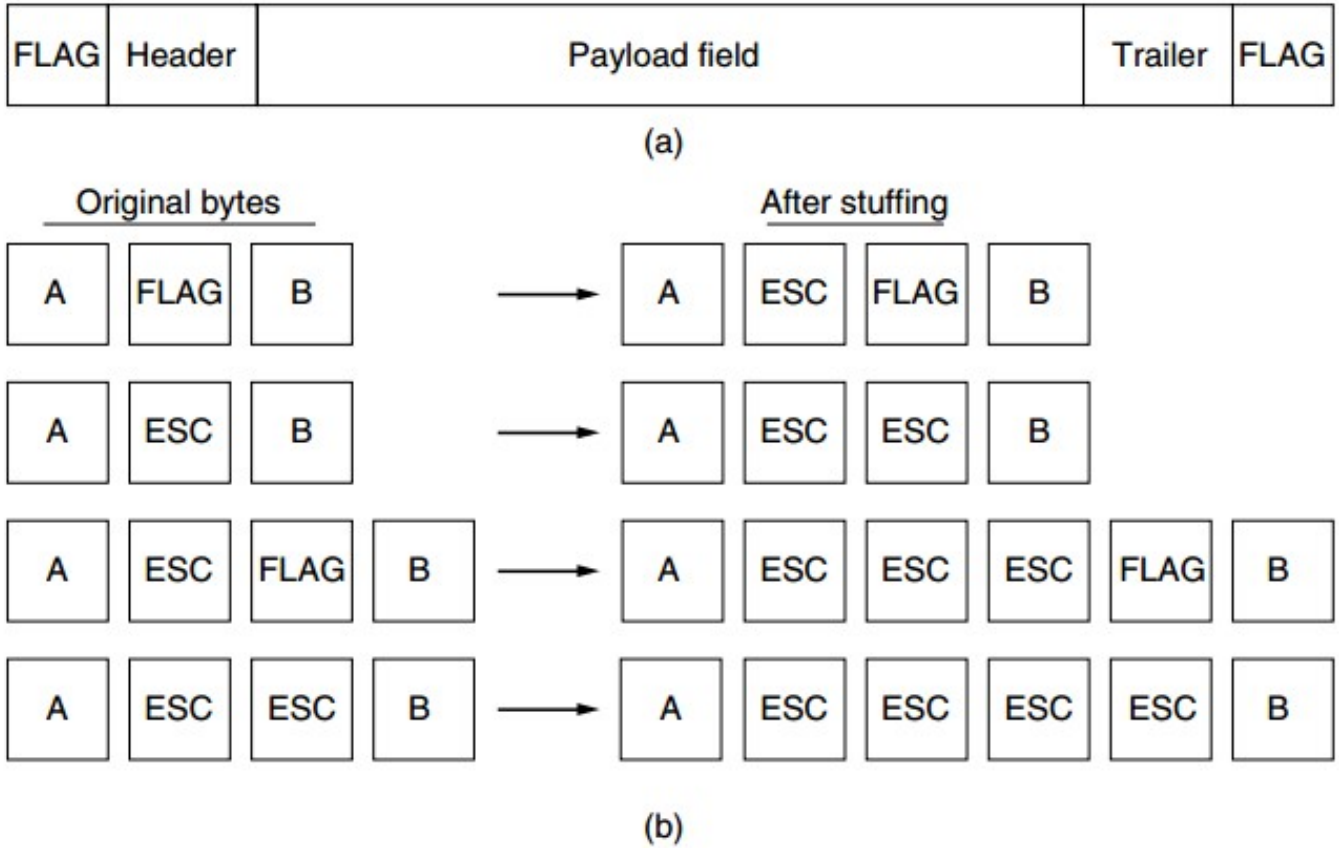
- Задатак 2
  - Дата је секвенца бајтова добијене после уоквиравања методом пребројавања знакова:

    8 2 5 4 1 3 1 1 8 4 5 2 2 7 3 9 4 3 5 1
  - Како изгледа оригинална секвенца (када се одстране маркери оквира)?

Почетни и завршни индикатори уз уметање бајтова

- Код ове методе уоквиравања, почетак и крај сваког оквира обележава се тзв. индикаторским бајтом (flag byte)
- У случају да одредишни рачунар изгуби корак, све што треба да уради је да пронађе наредни FLAG бајт, којим је означен крај актуелног оквира. Два FLAG бајта означавају крај једног и почетак другог оквира
- Проблем може да настане при преносу бинарних података, јер се лако догађа да се FLAG бајт нађе унутар самих података које треба пренети. Ово се решава уметањем контролног карактера ESC испред таквог бајта
- Међутим, шта се дешава када се унутар податка случајно нађе контролни ESC бајт?

Почетни и завршни индикатори уз уметање бајтова



Задаци

- Задатак 1
  - Рачунар А шаље оквире рачунару Б и ради уоквиравање са индикаторским бајтом Y. Контролни карактер је представљен симболом H. Низ бајтова који добија рачунар Б је:

    YGOHYEIDYJDKKEDYJJLLHHEJMYDOHWPEYKFJHDKEDY

  - Написати како изгледају оквири после обраде на рачунару Б?

- Задатак 2
  - Рачунар А шаље оквире рачунару Б и ради уоквиравање са индикаторским бајтом уз уметање знакова. Као Flag бајт користи се карактер F, а као контролни бајт карактер E. Два пакета на рачунару А су пре слања представљени следећим низом бајтова:

    FFVEERTC и PHYOFEQW

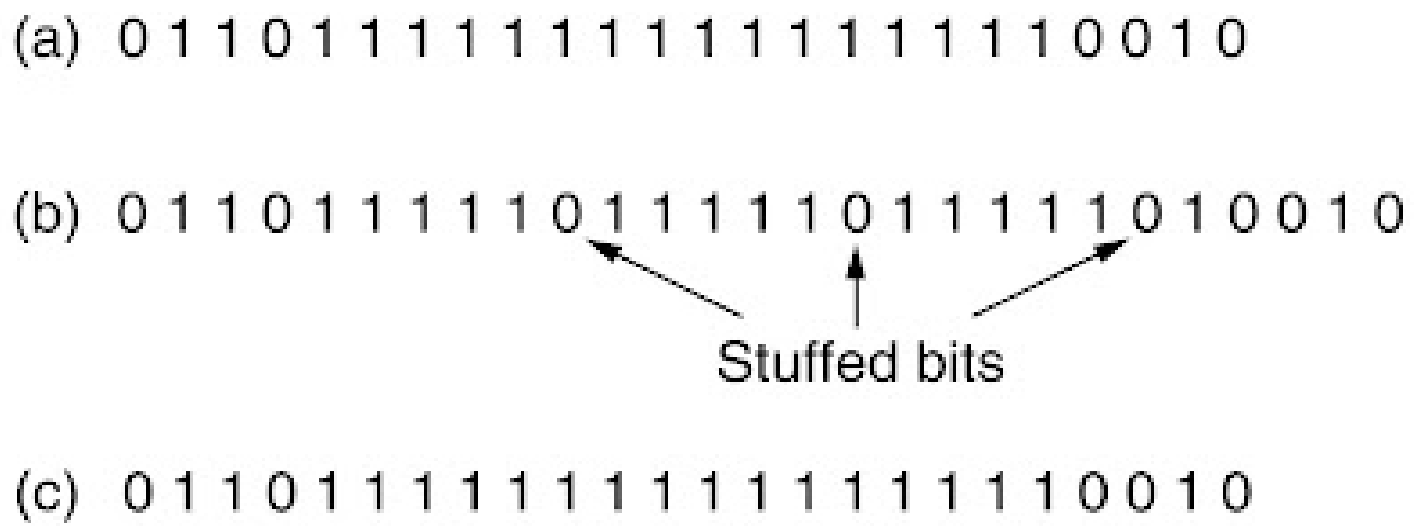
  - Како изгледа пристигли низ бајтова на рачунару Б?

Почетни и завршни индикатори из уметање битова

- Сваки оквир почиње и завршава се специјалном секвенцом битова (6 узастопних јединица)
- Када слој везе рачунара који шаље оквир наиђе на 5 узастопних јединица у подацима који се шаљу, аутоматски се умеће нула у излазни ток што је аналогно уметању ESC бајта у претходно показаном методу.
- Када прималац у долазном току пронађе датих 5 узастопних јединица иза којих следи нула, аутоматски избацује ту нулу.

Почетни и завршни индикатори из уметање битова

- a) оригинални подаци
- b) подаци током преноса
- c) подаци на страни примаоца



Задаци

- Задатак 1
  - Потребно је пренети следећи низ битова:

    01111011111101111110

  - Како изгледа оквир података након уметања специјалне секвенце битова?
- Задатак 2
  - Два рачунара А и Б приликом комуникације користе уоквиравање са почетним и завршним индикатором уз уметање битова. Рачунар Б је од рачунара А примио следећи низ битова (битови су представљени у хексадекадном систему):

    0x7E6FB57E7EF8F97E7EB7ED7E

  - Како изгледају распаковани пакети на рачунару Б?

Контрола грешака

- Поставља се питање како бити сигуран да су оквири заиста испоручени мрежном слоју примаоца и то исправним редоследом?
- Поузданост испоруке се обично обезбеђује тако што пошиљалац добија неку повратну информацију о томе шта се догађа на другом крају линије.
- Постоји могућност да због проблема са хардвером оквир потпуно нестане. У том случају прималац уопште неће реаговати, јер за то нема разлога. У овом случају, пошиљалац може заувек чекати на одговор.
- Ако се оквир или потврда о пријему изгубе, тајмер ће се после унапред дефинисаног времена аутоматски искључити и указати пошиљаоцу на могући проблем.
- Међутим, увек постоји ризик од примања истог оквира више пута.
- Да би се то спречило, оквирима који се шаљу додељују се редни бројеви.

Откривање и исправљање грешака

- Кодови за исправљање грешака
  - Уз податке, пошиљалац шаље и одређени “вишак” битова који је довољан да прималац закључи шта су стварни подаци чак и ако је настала грешка у преносу.
- Кодови за откривање грешака
  - Укључени вишак се користи да се открије да грешка постоји, након чега се захтева поновно слање оквира.

Хамингов код

- Омогућава откривање једноструке грешке и места на коме се грешка догодила, а самим тим и њено исправљање.
- Хамингов код се састоји од битова података и заштитних битова.
- Број заштитних битова зависи од броја битова података.
- Заштитни битови постављају се на позицијама у кодној речи које су степен броја 2 (прва, друга, четврта, осма..)

Методологија развоја Хаминговог кода

	2^0	2^1		2^2			
положај	1	2	3	4	5	6	7
улога	Z_1	Z_2	P_1	Z_3	P_2	P_3	P_4
			↓		↓	↓	↓
Код	?	?	**1**	?	**0**	**1**	**0**

- Улазна секвенца је: 1010
- На позицијама 1,2,4,8 поставити заштитне битове које обележавамо знаком Z. Остала поља попуњавамо редом битовима улазне секвенце.
- У реду “Код”, на пољима P_1, P_2, P_3 и P_4 преписујемо улазну секвенцу а заштитне битове обележавамо знаком “?”.

Методологија развоја Хамингового кода

	2^0	2^1		2^2			
положај	1	2	3	4	5	6	7
улога	Z_1	Z_2	P_1	Z_3	P_2	P_3	P_4
			↓		↓	↓	↓
Код	?	?	**1**	?	**0**	**1**	**0**

- Заштитни битови се рачунају на следећи начин:
  - Z_1 - формира низ од сваког друго бита, па је сходно томе потребно одредити вредност знака “?” како би се задовољила парност низа:
    - Z_1= ?100 ? = 1 да би се обезбедила парност секвенце
  - Z_2 - формира низ од сваког свака два узастопна бита, па размак за простор узастопних битова (положај: 2,3,6,7):
    - Z_2= ?110 ? = 0, парност је већ испуњена бројем јединица
  - Z_3 - формира низ од свака 4 узастопна бита, па размак за простор узастопних битова (положај: 4,5,6,7, па се прескачу 8, 9, 10, 11):
    - Z_3= ?010 ? = 1, да би се обезбедила парност секвенце.
  - Коначан Хамингов код је: 1 0 1 1 0 1 0

Методологија уочавања грешке помоћу Хамингового кода

	2^0	2^1		2^2			
положај	1	2	3	4	5	6	7
улога	Z_1	Z_2	P_1	Z_3	P_2	P_3	P_4
	?	**?**	**?**	**?**	**?**	**?**	**?**
Код	**1**	**0**	**1**	**1**	**1**	**1**	**0**

- Улазна секвенца је: 1011110
- У реду “Код”, уписати пристиглу улазну секвенцу
- Потребно је одредити исправност заштитних битова као на претходно описан начин:
  - Z_1 = ?110 ? = 0
  - Z_2 = ?101 ? = 0
  - Z_3 = ?101 ? = 0

Методологија уочавања грешке помоћу Хамминговог кода

	2^0	2^1		2^2			
положај	1	2	3	4	5	6	7
улога	Z_1	Z_2	P_1	Z_3	P_2	P_3	P_4
	?	**?**	**?**	**?**	**?**	**?**	**?**
Код	**1**	**0**	**1**	**1**	**1**	**1**	**0**

- Z_1 = ?110 ? = 0
- Z_2 = ?101 ? = 0
- Z_3 = ?101 ? = 0
- Провером уочавамо да Z_1 има вредност 1, а требало би да има вредност 0, Z_2 има вредност 0 што је и јесте случај и вредност Z_3 има вредност 1 а требало би да има вредност 0. Ово указује да је пристигла Хамингова секвенца неисправна, тј. десила се грешка у преносу.

Методологија уочавања грешке помоћу Хаминговог кода

	2^0	2^1		2^2			
положај	1	2	3	4	5	6	7
улога	Z_1	Z_2	P_1	Z_3	P_2	P_3	P_4
	?	**?**	**?**	**?**	**?**	**?**	**?**
Код	**1**	**0**	**1**	**1**	**1**	**1**	**0**

- Заштитни битови Z1 и Z3 се разликују и то нам указује да помоћу њих можемо установити на којој позицији се десила грешка. Сабирајући положаје заштитних битова 1 + 4 = 5 долазимо до податка да је грешка настала на петој позицији.
- Дакле исправљајући грешку на петој позицији са 1 на 0 и отклањајући заштитне битове, добијамо послату секвенцу: 1010

Задаци

- Пошиљалац треба да пошаље низ битова вредности 0xCCAA. Користећи Хамингов код, написати ток битова који се стварно шаље. Уколико је 7. бит са десне стране инвертован, доказати да прималац детектује грешку.
- Пошиљалац треба да пошаље низ битова вредности 0x0F0F. Користећи Хамингов код, написати ток битова који се стварно шаље. Ако је 4. бит са десне стране инвертован, доказати да прималац детектује грешку.
- Пошиљалац треба да пошаље низ битова вредности 1011101010111010 користећи Хамингов код. Написати ток битова који се стварно шаље. Ако је 4. бит са десне стране грешком инвертован доказати да прималац детектује ову грешку.

CRC (Cyclic Redundant Check)

- Ова метода служи за откривање грешака у преносу, а користи се најешће у бежичним технологијама.
- Уколико желимо да пренесемо K битова неке информације (пакет са мрежног слоја), креираћемо оквир који садржи N битова, тако да је N>K. Разлику N - K називамо FCS (Frame check sequence).
- Како би смо изводили рачун везан за добијање FCS, потребно је познавање XOR функције.
- Такође, код CRC-а, пошиљалац и прималац морају се сложити око генераторског полинома. Да бисте израчунали контролни збир за оквир од м битова, оквир мора бити дужи од полинома генератора.
- Циљ алгоритма је да се на крај оквира дода контролни збир тако да полином који представља оквир са контролним збиром дељивим са генераторским полиномом. Ако то није случај, дошло је до грешке и потребно је поновно слање оквира.

Алгоритам CRC-а

- Нека је r степен генераторског полинома . На крај оквира додје се r нула тако да он сада садржи n + r битова и одговара му полином:

- се дели по модулу 2 са полиномом .
- Одузима се остатак претходног дељења од и тиме се добија оквир са контролним збиром који треба послати .

Алгоритам CRC-а

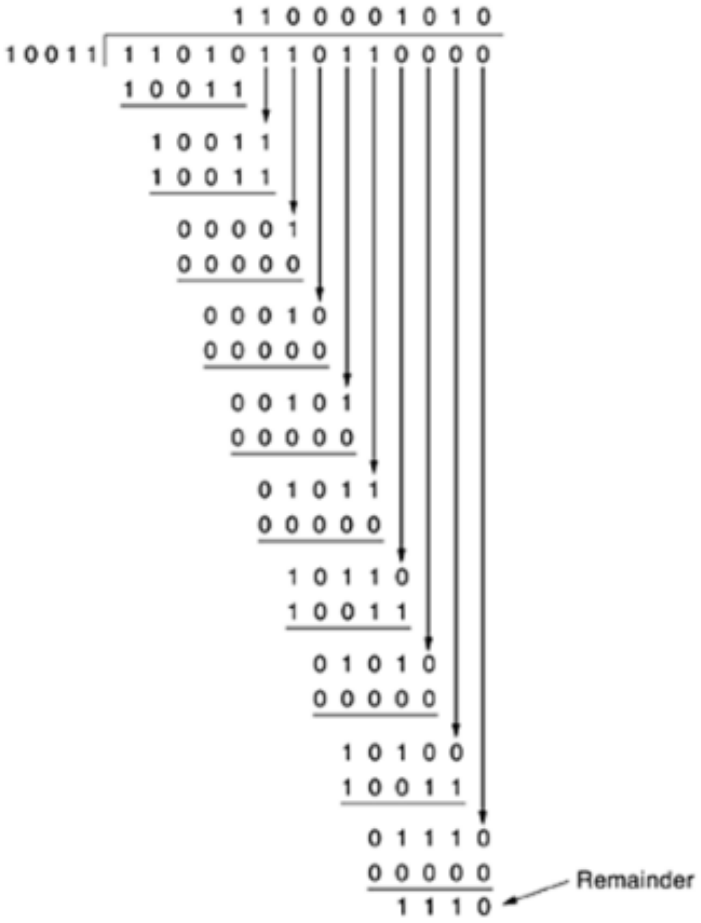
- Оквир (порука): 1101011011
- Генератор 10011
- Корак-1: С обзиром да се ради о генератору 4.степена, то значи да је потребно оквиру додати 4 нуле, самим тим оквир сада изгледа овако:

  11010110110000

- Корак-2: Оквир након што су му додате 4 нуле треба поделити са генератором по модулу 2:
- Корак-3: Остатак добијен при дељењу одузимамо од оквира на који смо претходно додали 4 нуле. Одузимање се ради по модулу 2 и одговара логичкој операцији XOR.
- Остатак: 1110
- Оквир: 11010110110000
- Резултат: =11010110111110
- Проверу да ли је дошло до грешке на пријемној страни вршимо одузимањем генераторског полинома од пристигле секвенце, уколико је остатак 0, тада је секвенца примљена без грешака.

Алгоритам CRC-а

Frame : 1 1 0 1 0 1 1 0 1 1
Generator: 1 0 0 1 1
Message after 4 zero bits are appended: 1 1 0 1 0 1 1 0 1 1 0 0 0 0



Transmitted frame: 1 1 0 1 0 1 1 0 1 1 1 1 0

Задатак

- Пошиљалац треба да пошаље низ битова вредности 0xCCAA. Написати ток битова који се стварно шаље, ако се за пренос користи стандардна CRC метода са генераторским полиномом $x^5+x^4+x^3$.

Питања?