

Kanalinis lygis

Kanalinis lygis

Taikomasis lygis

Transporto lygis

Tinklo lygis

Kanalinis lygis

Fizinis lygis

Taikomasis lygis

Transporto lygis

Tinklo lygis

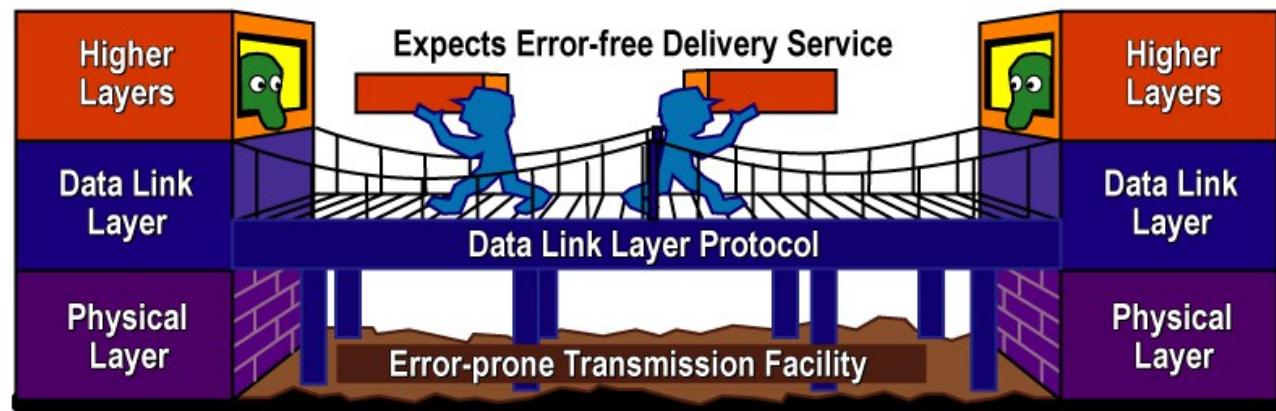
Kanalinis lygis

Fizinis lygis



Kanalinis lygis

- Samprata
- Tikslai:
 - Teikti servisą tinklo lygiui;
 - Suskaidyti paketus į kadrus;
 - Atstatyti gautų kadrų tvarką;
 - Užtikrinti patikimą duomenų pristatymą;
 - Valdyti srautą

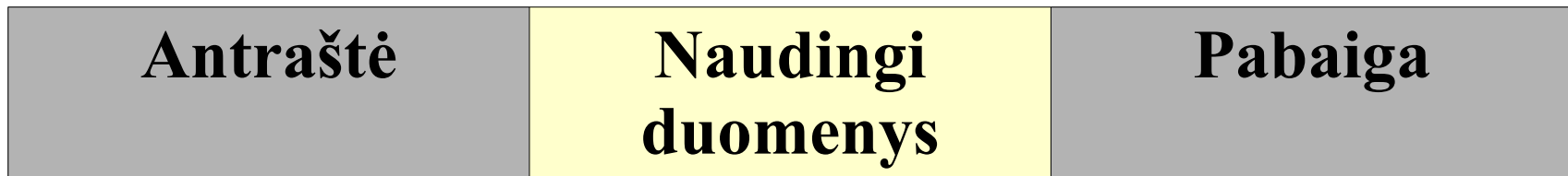


Kanalinis lygis

- Kanalinio lygio protokolų prototipai:
 - Paprastieji lygio protokolai;
 - Slenkančio lango protokolas;
- Standartinių protokolų pavyzdžiai

Kanalinio lygio samprata

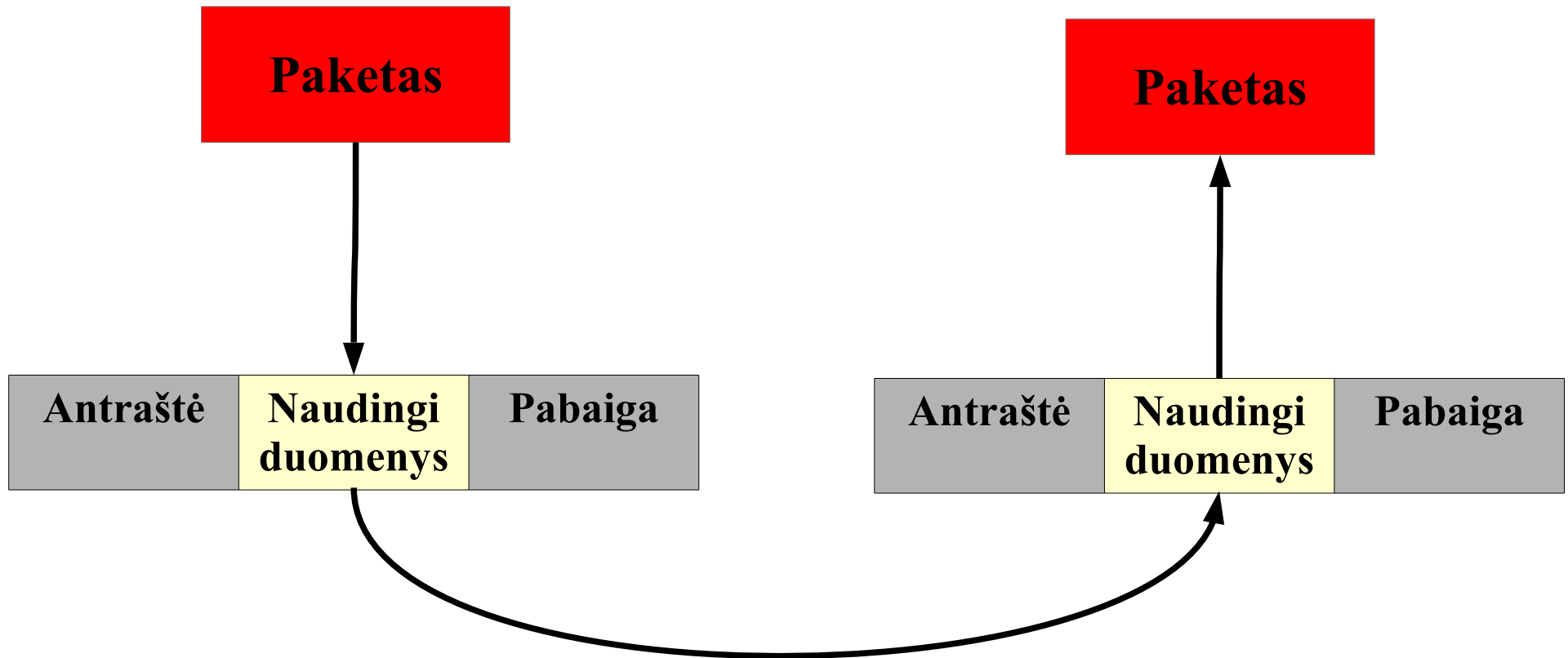
- Kanalinis lygis jungia techninę ir programinę dalis;
- Perduodamų duomenų vienetas – kadras (angl. Frame);
- Tipinė kadro sandara:
 - Antraštė;
 - Naudingi duomenys;
 - Pabaiga



Kanalinio lygio samprata

- Tipiniai kadrų tipai:
 - Duomenų kadrai;
 - Kontroliniai kadrai:
 - Patvirtinimo kadras (angl. *Acknowledgment frame*) – ACK
 - Atmetimo kadras (angl. *Negative acknowledgment frame*) – NAK

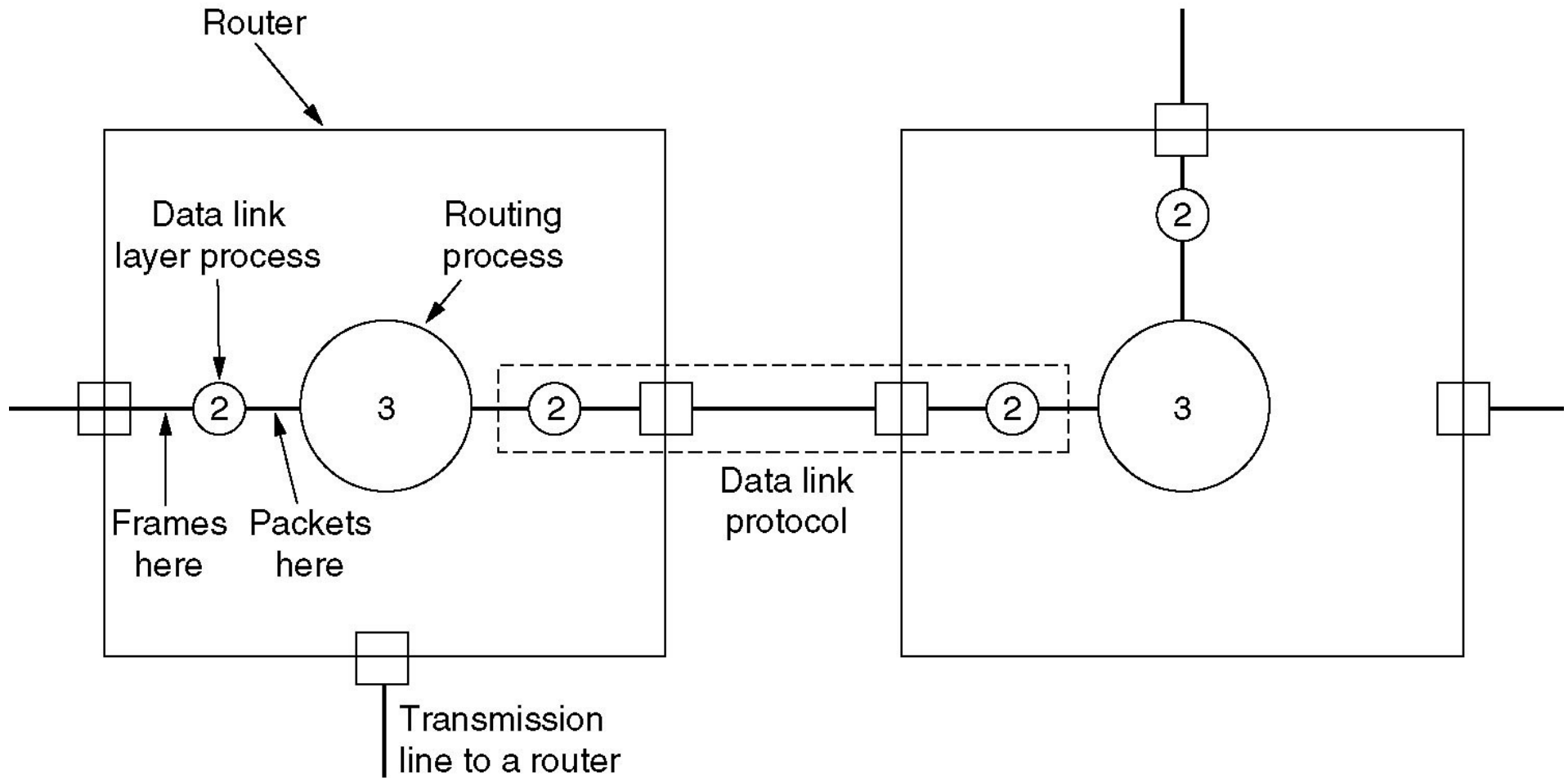
Paketų ir kadru sąsaja



Kanalinis lygis

- Servisas, pateikiamas tinklo lygiui
 - Be sujungimo ir patvirtinimų:
 - Nėra loginių susijungimų;
 - Siunčiami nepriklausomi kadrai
 - Be sujungimo ir su patvirtinimais:
 - Nėra loginių susijungimų;
 - Siunčiamas ACK kadras sėkmės atveju
 - Reikalingi taimeriai
 - Su sujungimu ir patvirtinimais:
 - Reikalingas kadrų numeravimas
 - Buferis

Kanalinio lygio veikimo vieta



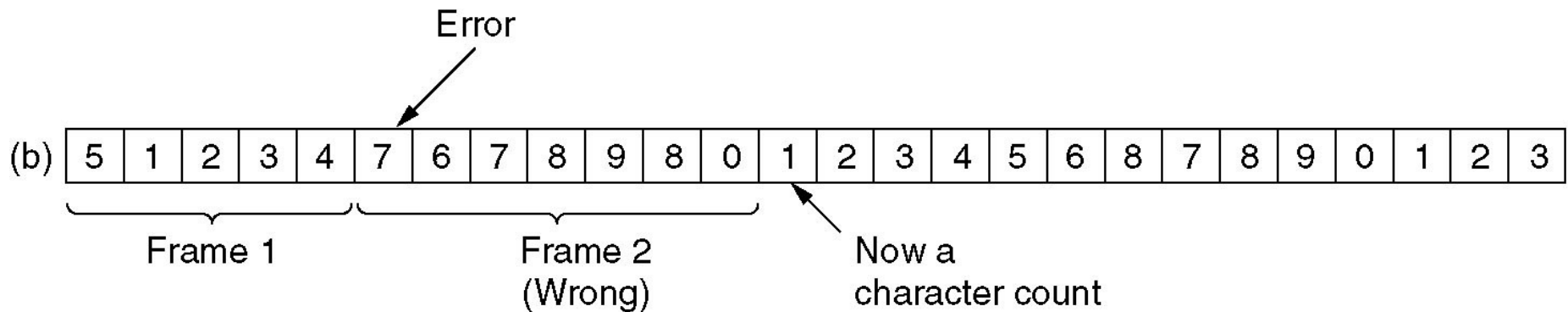
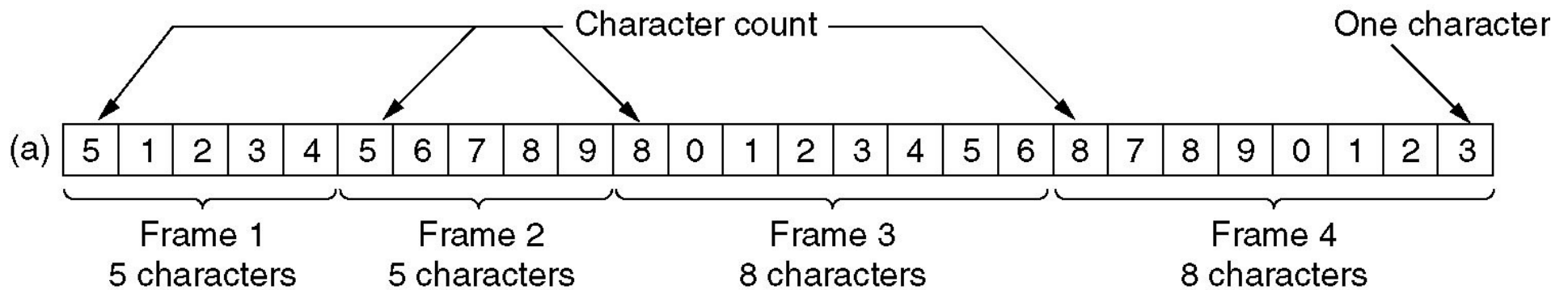
Kanalinio lygio tikslai

- Paketų skaidymas į kadrus
- Klaidų aptikimas ir taisymas
- Srauto kontrolė

Paketų skaidymas į kadrus

- Paketų skaidymo į kadrus metodai:
 - Baitų kiekio nurodymas
 - Specialių starto ir pabaigos simbolių įterpimas
 - Starto ir pabaigos bitų įterpimas
 - Fizinio lygio kadrų pabaigos ir pradžios aptikimas

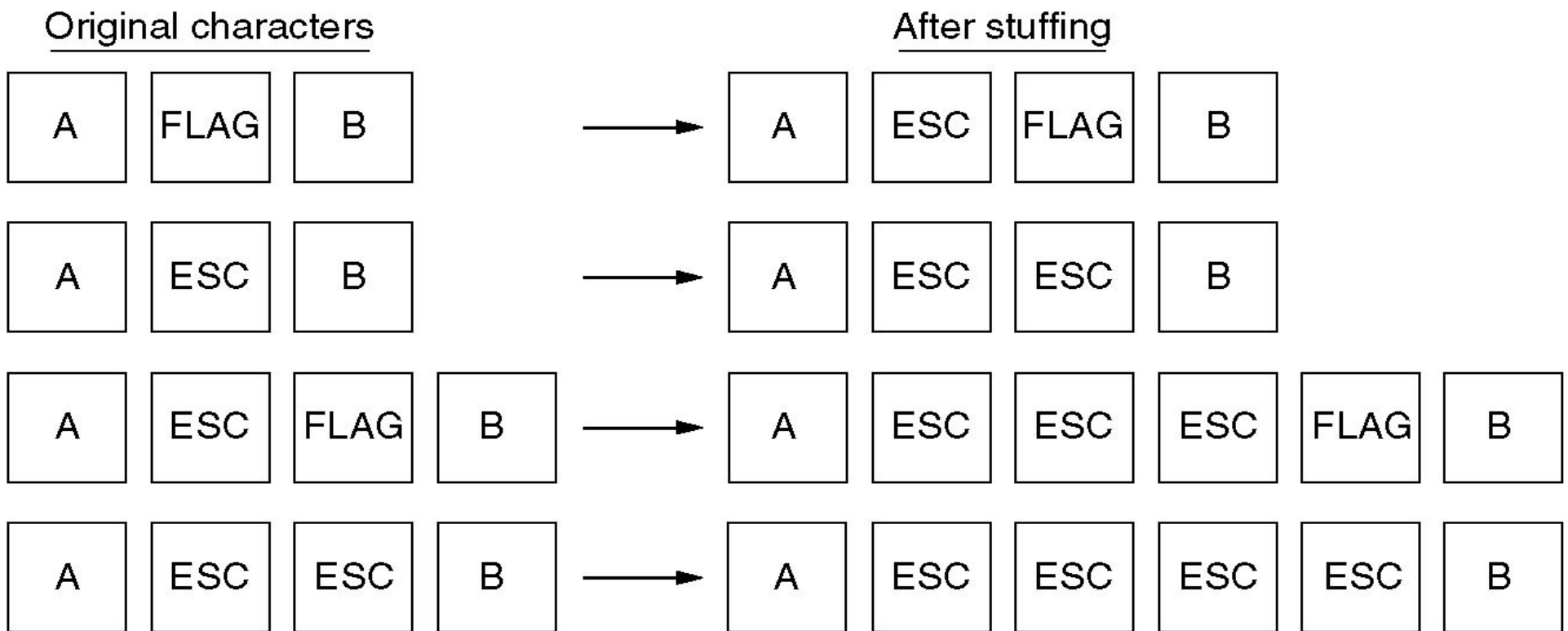
Skaidymas į kadrus



Skaidymas į kadrus



(a)




(b)

Skaidymas į kadrus

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0



Stuffed bits

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Kadrų gavimo klaidų aptikimas ir taisymas

- Klaidas aptinkantys ir taisantys kodai
- Atbulinis ryšys patvirtinimui
- Taimeriai
- Kadrų numeracija

Klaidų aptikimas

- Klaidas taisantys kodai
 - Artimiausias Hemingo (angl. Hamming) atstumas
- Klaidų aptikimas
 - CRC (Cyclic redundancy check)
 - Pavyzdys:
 - Kadras: 1101011011
 - Generatoriaus polinomas: $G(x) = x^4 + x + 1$
 - CRC Pavyzdžiai:
 - CRC-1, 0x1 arba $G(x) = 1$
 - CRC-32, 0x04C11DB7 arba

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

Srauto kontrolė

- Reikalinga kai kadrai yra gaunami greičiau nei juos spėjama apdoroti;
- Ribotas kadru buferis;
- Dvi strategijos ribojant srautą:
 - Atgalinio ryšio – gavėjas išsiunčia leidimą siuntėjui siųsti duomenis
 - Kiekio – iš anksto nustatomas leistinas duomenų kiekis, nenaudojant atgal siunčiamų leidimų

Paprastieji kanalinio lygio protokolai

```
#define MAX_PKT 1024    // Paketo dydis baitais

typedef enum {false,true} boolean; // Sveikas tipas
typedef unsigned int seq_nr    // Tipas sekos numeriui arba patvirtinimo
numeriui
typedef struct {unsigned char data[MAX_PKT];} packet; // Paketas
typedef enum {data, ack,nak} frame_kind // Kadro tipas
typedef struct{ // Kadro struktūra
    frame_kind kind; // Kadro tipas
    seq_nr seq;    // Sekos numeris
    seq_nr ack;    // Atsakymo numeris
    packet info;   // Tinklo sluoksnio paketas
} frame;

void waif_for_event(event_type *event);
void from_network_layer(packet *p);
void to_network_layer(packet *p);
void from_physical_layer(frame *r);
void to_physical_layer(frame *s);
void start_timer(seq_nr k);
void stop_timer(seq_nr k);
void start_ack_timer(void);
void stop_ack_timer(void);
void enable_network_layer(void);
void disable_network_layer(void);
#define inc(k) k=k++%MAX_SEQ
```

Paprastieji kanalinio lygio protokolai

- Simplex – protokolas be apribojimų
 - Duomenys perduodami tik viena kryptimi
 - Siuntėjas ir gavėjas visada pasiruošę išsiųsti ir priimti duomenis
 - Duomenų apdorojimo laikas ignoruojamas
 - Daroma prielaida, kad buferis yra neapribotas
 - Duomenys kanale neprarandami ir neiškraipomi

Neapribotas simplex protokolas

```
typedef enum{frame_arrival} event_type;  
#include "protocol.h"
```

```
void sender1(void){  
    frame s;  
    packet buffer;  
    while (true) {  
        from_network_layer(&buffer);  
        s.info = buffer;  
        to_physical_layer(&s);  
    }  
}
```

```
void receiver1(void){  
    frame r;  
    event_type event;  
    while (true) {  
        wait_for_event(&event);  
        from_physical_layer(&r);  
        To_network_layer(&r.info);  
    }  
}
```

Paprastieji kanalinio lygio protokolai

- Simplex protokolas su sustojimu ir laukimu
 - Laiko skaičiavimas duomenų apdorojimui kanaliname lygmenyje
 - Siuntėjas negali perduoti duomenų be galo greitai
 - Įvedamas atbulinis ryšys, kuriam panaudojamas specialus tarnybinis kadras, kuris leidžia siuntėjui išsiųsti kitą kadrą

Simplex protokolas su sustojimu ir laukimu

```
typedef enum{frame_arrival} event_type;  
#include "protocol.h"
```

```
void sender2(void){  
    frame s;  
    packet buffer;  
    event_type event;  
    while (true) {  
        from_network_layer(&buffer);  
        s.info = buffer;  
        to_physical_layer(&s);  
        wait_for_event(&event);  
    }  
}
```

```
void receiver2(void){  
    frame r, s;  
    event_type event;  
    while (true) {  
        wait_for_event(&event);  
        from_physical_layer(&r);  
        to_network_layer(&r.info);  
        to_physical_layer(&s);  
    }  
}
```

Paprastieji kanalinio lygio protokolai

- Simplex protokolas kanalui su triukšmu
 - Pagrindinė problema – patvirtinimo kadras gali būti prarastas
 - Kadru numeracija – kiek vietos palikti antraštėje

Simplex protokolas kanalui su triukšmu

```
#define MAX_SEQ 1
typedef enum{frame_arrival, cksum_err, timeout } event_type;
#include "protocol.h"
```

```
void sender3(void){
    seq_nr next_frame_to_send;
    frame s;
    packet buffer;
    event_type event;
    next_frame_to_send = 0;
    from_network_layer(&buffer);
    while (true) {
        s.info = buffer;
        s.seq = next_frame_to_send;
        to_physical_layer(&s);
        start_timer( s.seq);
        wait_for_event(&event);
        if ( event == frame_arrival) {
            from_physical_layers(&s);
            if ( s.ack == next_frame_to_send ) {
                from_network_layer( &buffer );
                inc( next_frame_to_send );
            }
        }
    }
}
```


Simplex protokolas kanalui su triukšmu

```
void receiver3(void){
    seq_nr frame_expected;
    frame r, s;
    event_type event;
    while (true) {
        wait_for_event(&event);
        if ( frame == event_arrival ) {
            from_physical_layer(&r);
            if ( r.seq == frame_expected ) {
                to_network_layer(&r.info);
                inc(frame_expected);
            }
            s.ack = 1 -frame_expected;
            to_physical_layer(&s);
        }
    }
}
```

Kanalo efektyvumas

- Kanalo efektyvumas:

$$efektyvumas = \frac{l}{l + bR}$$

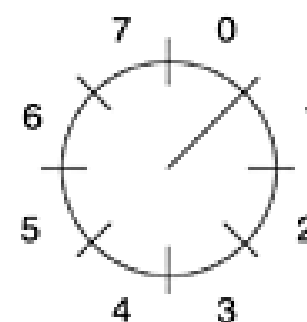
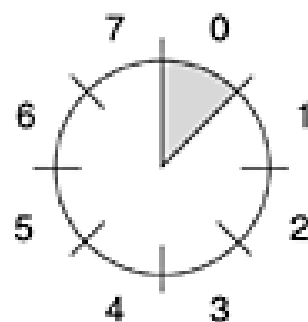
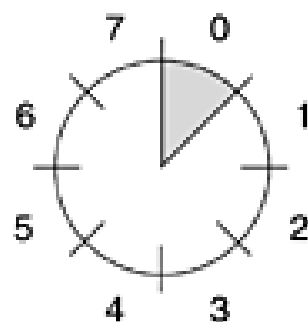
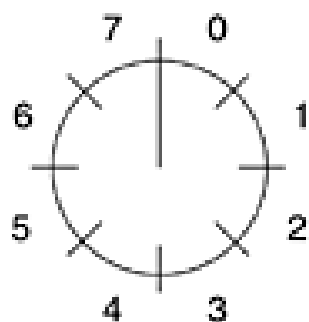
- l – kadro dydis bitais
- b – kanalo pralaidumas b/s
- R – bendras perdavimo laikas (išsiuntimas + gavimas)

Slystančio lango protokolai

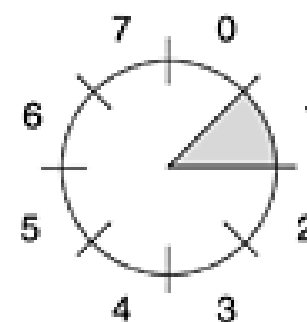
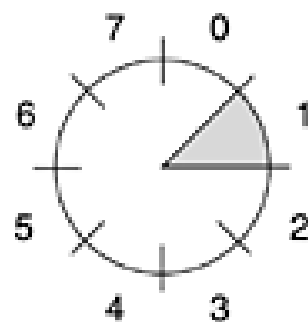
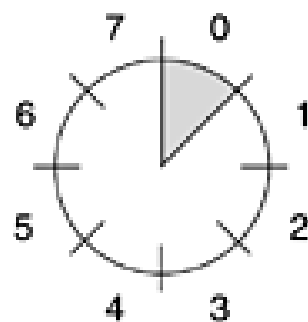
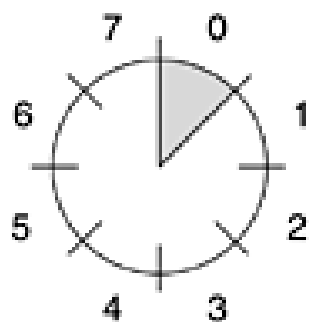
- Patvirtinimo kadrai – neleistinas resurso (kanalo pralaidumo) apkrovimas
- Slystančio lango protokolų klasė
 - Siuntėjo langas
 - Gavėjo langas
 - Kadru numeriai išsiuntimo lange – išsiųsti kadrai, bet nepatvirtinti.
 - Kai tik iš tinklo lygio ateina dar vienas paketas, jam priskiriamas pirmas laisvas didžiausias numeris ir viršutinis lango režis pakeliamas
 - Kai tik ateina patvirtinimas – apatinis režis pakeliamas.

Slystančio lango protokolai

Sender



Receiver



(a)

(b)

(c)

(d)

Slystančio lango lango protokolai su vienu langu ir trijų bitų sekos numeriu

Slystančio lango protokolai

```
#define MAX-SEQ 1
typedef enum{frame_arrival,cksum_err, timeout} event_type;
#include "protocol.h"

void protocol4 (void) {
    seq_nr next_frame_to_send;
    seq_nr frame_expected;
    frame r, s;
    packet buffer;
    event_type event;
    next_frame_to_send = 0;
    frame_expected = 0;
    from_network_layer(&buffer);
    s.info = buffer;
    s.seq = next_frame_to_send;
    s.ack = 1 -frame_expected;
    to_physical_layer(&s);
    start_timer(s.seq);
}
```

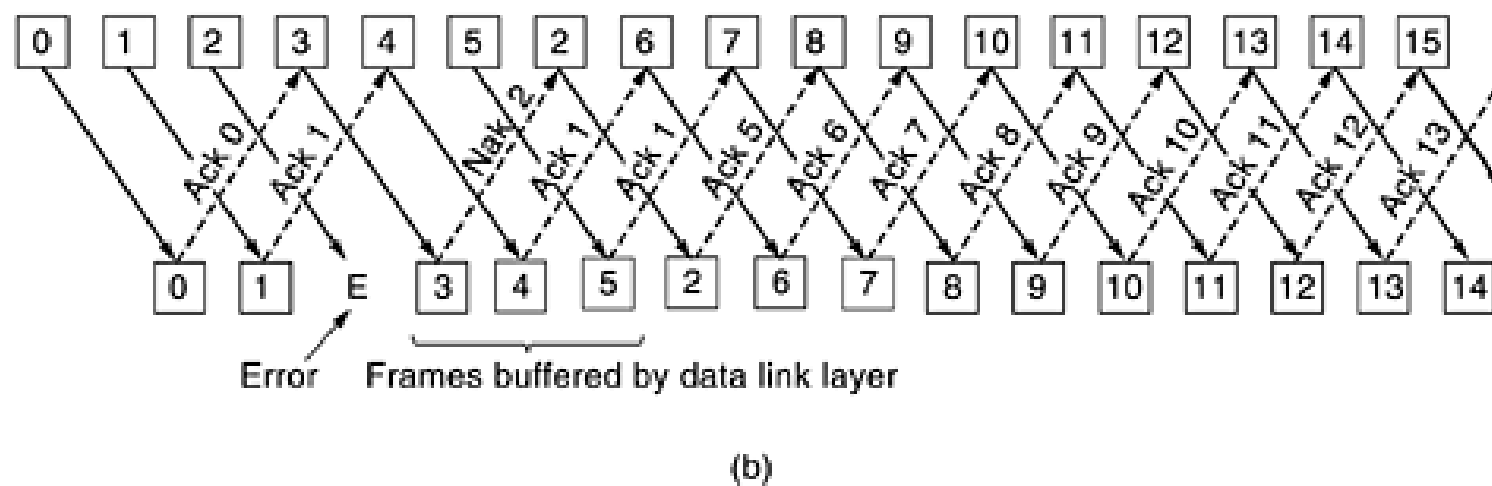
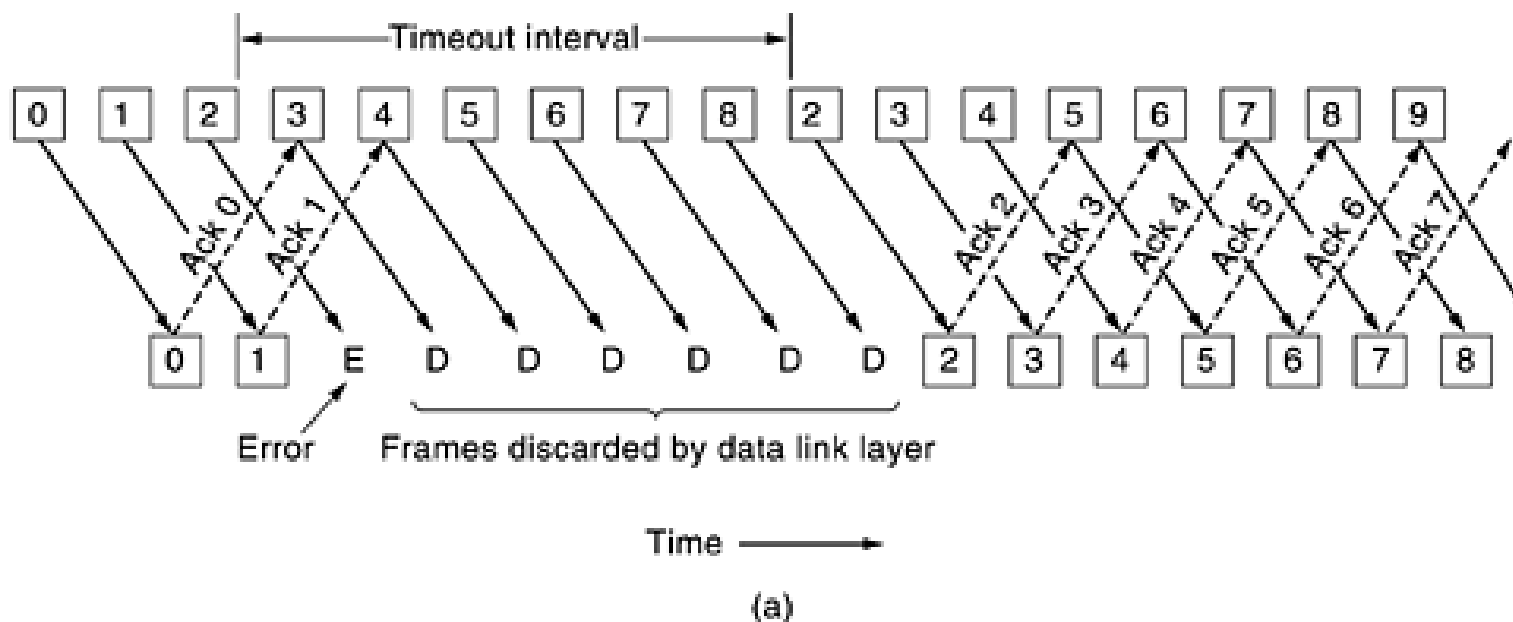
Slystančio lango protokolai

```
while (true) {
    wait_for_event(&event);
    if (event == frame_arrival) {
        from_physical_layer(&r);
        if (r.seq== frame_expected) {
            to_network_layer(&r.info);
            inc(frame_expected);
        }
        if (r.ack== next_frame_to_send) {
            from_network_layer(&buffer);
            inc(next_frame_to_send);
        }
    }
    s.info = buffer;
    s.seq= next_frame_to_send;
    s.ack= 1 -frame_expected;
    to_physical_layer(&s);
    start_timer(s.seq);
}
}
```

Slystančio lango protokolai

- Pavyzdys:
 - Palydovinis 50 Kbps kanalas su bendru užlaikymu 500 ms. Norime panaudoti 4 protokolą 1000 bitų kadro perdavimui. Laiko momentu $t=0$ ms siuntėjas išsiunčia pirmą kadrą. Laiko momentu $t=20$ ms kadras pilnai išsiųstas, laiko momentu $t=270$ ms jis priimtas ir momentu $t=520$ ms siuntėjas gauna patvirtinimą. Siuntėjas blokuojamas $500/520$, t.y. 96% laiko.
 - Reikalingas langas 26 kadrams

Slystančio lango protokolai



Kanalinio lygio protokolų pavyzdžiai

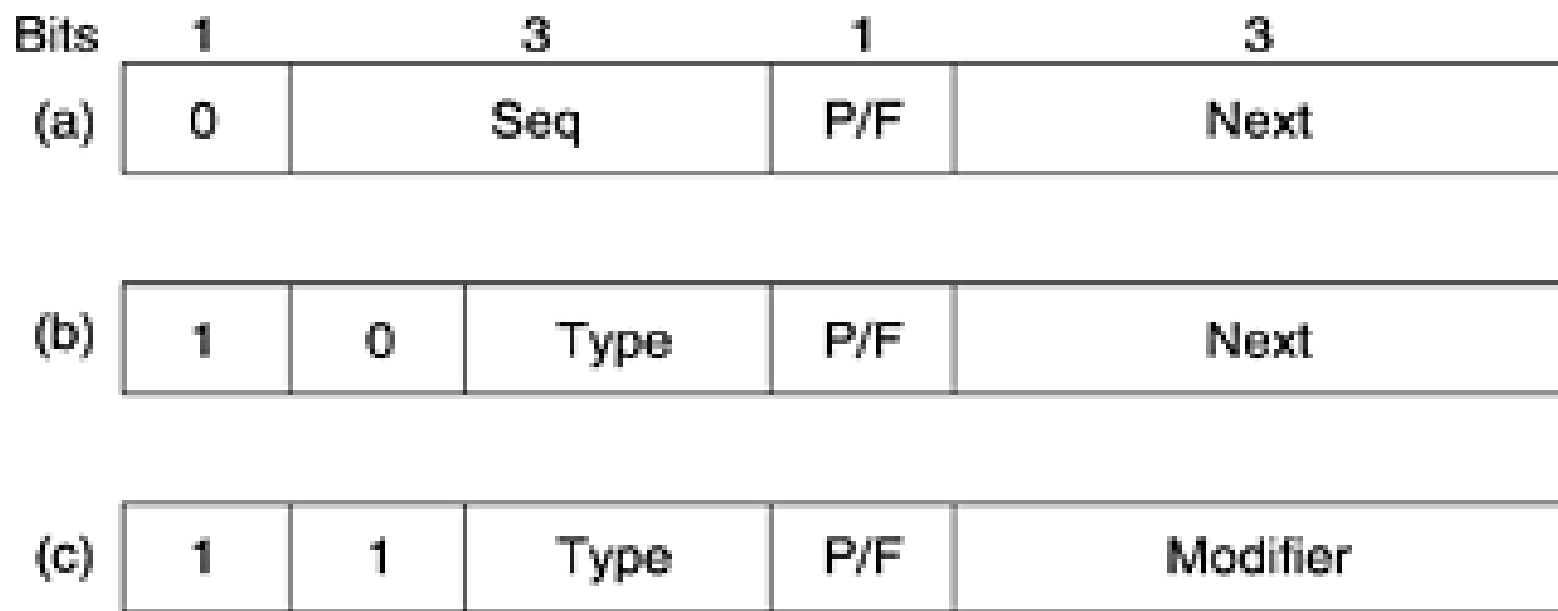
- HDLC
- SLIP
- PPP

HDLC protokolas

- SDLC – Synchronous Data Link Control – sinchroninio kanalo valdymo protokolas, pasiūlytas IBM SNA tinklui
- ISO modifikavo protokolą ir išleido HDLC – High level Data Link Control
- Protokolas orientuotas į bitus

Baitai	1	1	1	>0	2	1
	01111110	Adresas	Valdymas	Duomenys	CRC	01111110

HDLC protokolas. Control lauko struktūra



- a – Informacinis kadras
- b – Supervizorinis kadras
- c – Nenumerojamas kadras

HDLC protokolas. Control lauko struktūra

- Supervizorinis kadras būna keturių tipų
 - Tipas 0 – patvirtinimas apie sekančio kadro laukimą (RECEIVE READY). Naudojamas kuomet nėra trafiko tam, kad perduoti patvirtinimą su duomenim
 - Tipas 1 – negatyvus patvirtinimas (REJECT) – nurodo, kad įvyko klaida duomenų perdavime. Kadro laukas NEXT nurodo kadro numerį nuo kurio reikia pakartotinai siųsti kadrus

HDLC protokolas. Control lauko struktūra

- Supervizorinis kadras būna keturių tipų
 - Tipas 2 – (RECEIVE NOT READY). Patvirtina visus kadrus, išskyrus tą, kurio numeris nurodytas NEXT lauke. Naudojamas kai reikia pranešti siuntėjui, kad reikia laikinai sustabdyti perdavimą. Po problemų išsprendimo siunčia kadrą RECEIVE READY, REJECT arba kita tarnybinį kadrą.
 - Tipas 3 – (SELECTIVE REJECT) – nurodo, kad reikia persiųsti kadrą, kurio numeris yra lauke NEXT. SDLC nenaudoja šio kadrų tipo

HDLC protokolas. Control lauko struktūra

- Nenumerojamos kadras naudoja:
 - Valdymo tikslams
 - Duomenims perduoti nepatikimu perdavimu be sujungimo

SLIP protokolas

- SLIP – Serial Line IP
 - Nėra klaidų kontrolės ir taisymo
 - Dirba tik su IP paketais
 - Bendraujančių pusių IP adresai turi būti žinomi iš anksto
 - Protokolas nepateikia autentiškumo priemonių
 - Protokolui nėra standarto ir yra labai daug jo versijų

SLIP protokolas

- Kadras neturi antraštės
- Naudojamas pabaigos (END) simbolis **0xbd**
- Naudojamas escape (ESC) simbolis **0xc0**

Baitai

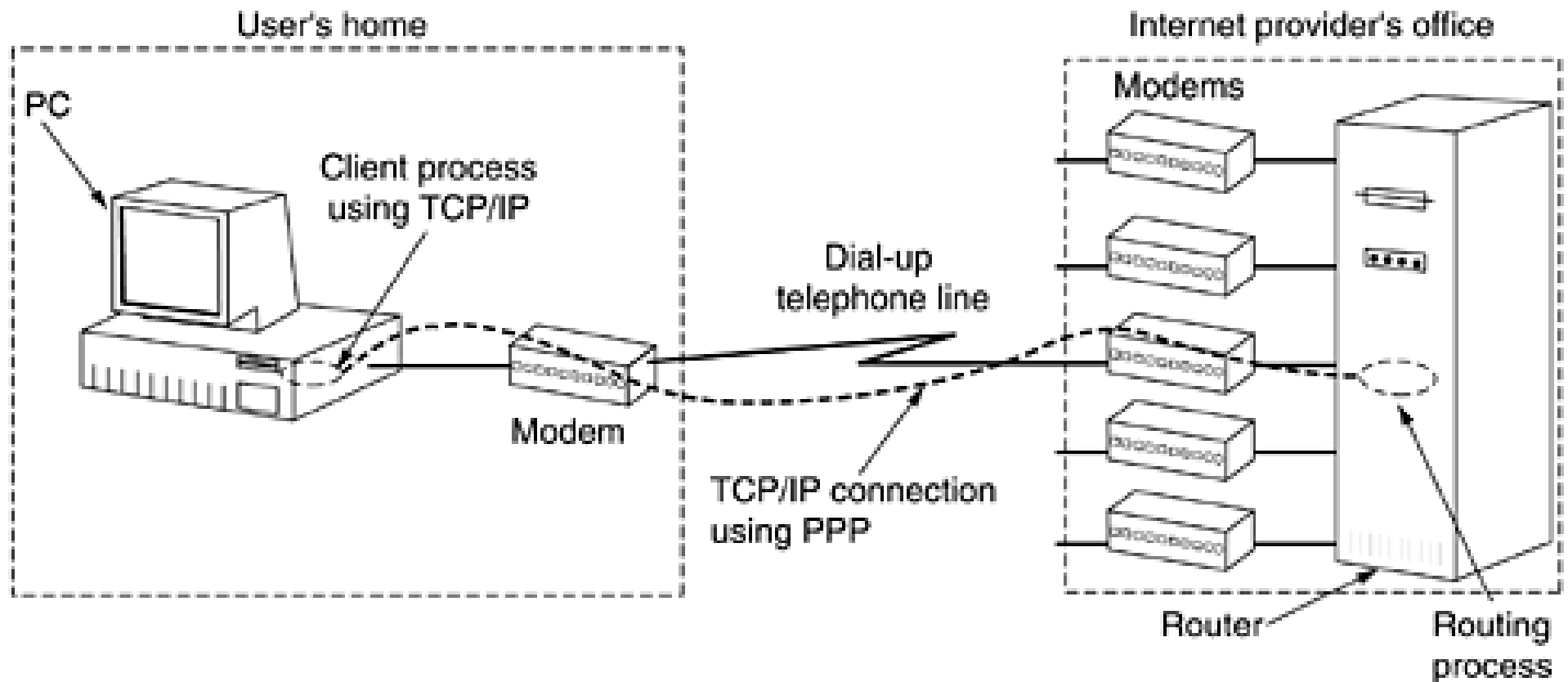
≤ 1006

1

Duomenys	Pabaiga 11011011
-----------------	-----------------------------------

Kanalinis lygis internete

- Prisijungimas prie interneto



PPP protokolas

- PPP – (Point-to-Point Protocol) sukurtas IETF:
 - Apibrėžtas RFC 1661, 1662, 1663
 - Su klaidų tikrinimu
 - Palaiko skirtingus protokolus
 - Leidžia dinamiškai priskirti IP adresą sujungimo laikui
 - Tikrina abonentų autentiškumą
 - Kiti privalumai palyginti su SLIP

PPP protokolas

- PPP susideda iš trijų protokolų:
 - Kadrių atpažinimas, t.y. vienareikšmiškai apibrėžia kadro pradžia ir pabaigą. Taip pat čia vyksta klaidų aptikimas.
 - Linijos valdymo protokolas, t.y. linijos aktyvacija, patikrinimas, pagrindinių perdavimo parametrų nustatymas, korektiškas perdavimo nutraukimas. Protokolas vadinasi LCP (angl. Link Control Protocol)
 - Pagrindinių tinklinio lygio parametrų nustatymo protokolas, kuris užtikrina nepriklausomybę nuo tinklo lygio realizacijos. Toks metodas gali veikti su skirtingais NCP (angl. Network Control Protocol) kiekviename tinklo lygyje

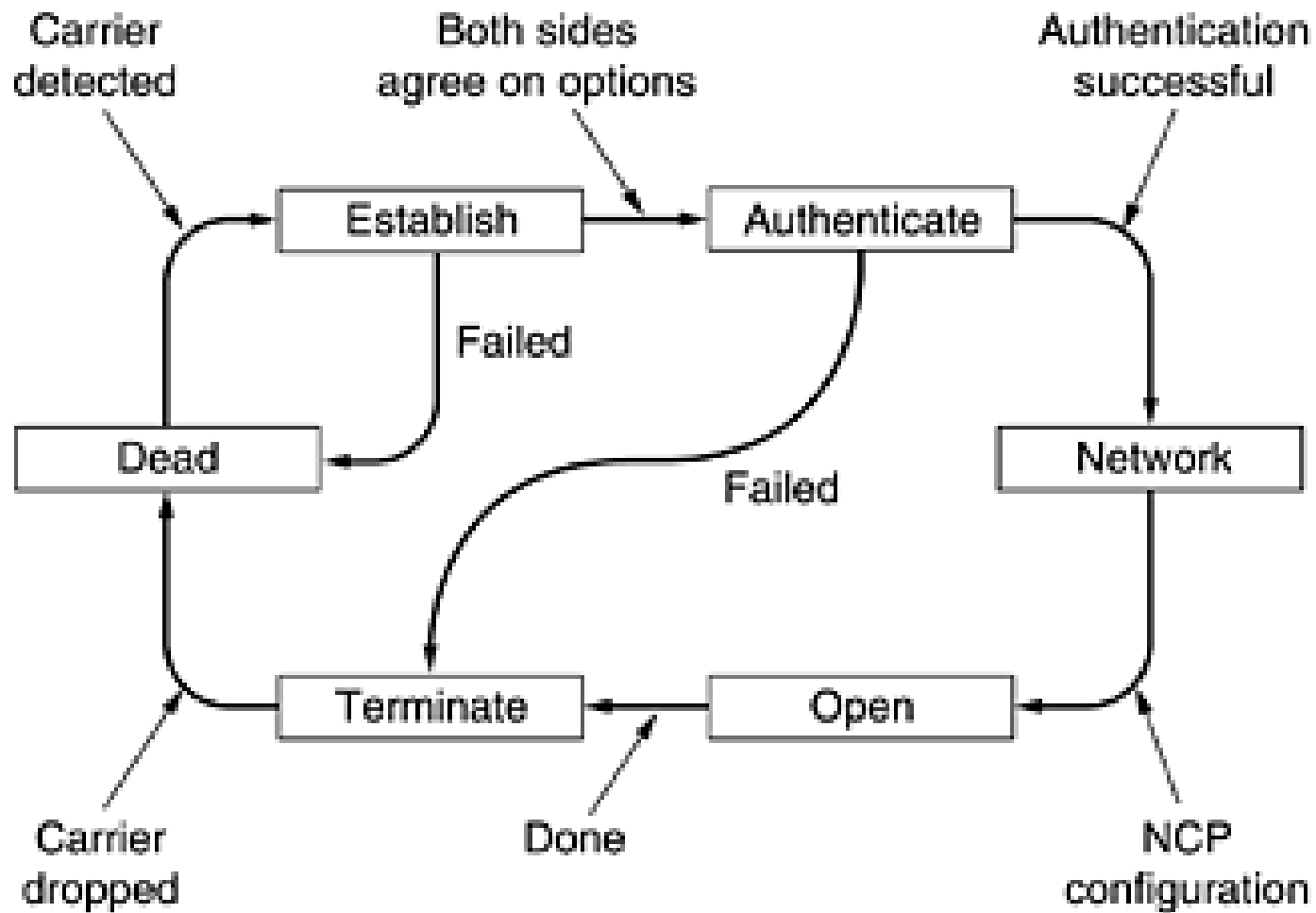
PPP protokolas. Struktūra

Baitai	1	1	1	1 arba 2	-	2 arba 4	1
	Žyma 01111110	Adresas 11111111	Valdymas 00000011	Protokolas	Duomenys	CRC	Žyma 01111110

LCP Kadrų tipai

Pavadinimas	Kryptis	Aprašymas
Configure-request	$I > R$	Pateikiami susijungimo sąlygas
Configure-ack	$I < R$	Sutinkama su sąlygomis
Configure-nak	$I < R$	Nesutinkama su sąlygomis
Configure-reject	$I < R$	Kai kurios sąlygos nėra derinamos
Terminate-request	$I > R$	Prašoma nutraukti susijungimą
Terminate-ack	$I < R$	Sutinkama su nutraukimo prašymu
Code-reject	$I < R$	Nežinomas prašymas gautas
Protocol-reject	$I < R$	Nežinomas prašomas protokolas
Echo-request	$I > R$	Prašymas atsiųsti šį kadrą atgal
Echo-reply	$I < R$	Siunčiamas kadras atgal
Discard-request	$I > R$	Prašoma ignoruoti kadrą (testavimo tikslais)

PPP Schema



Klausimai?