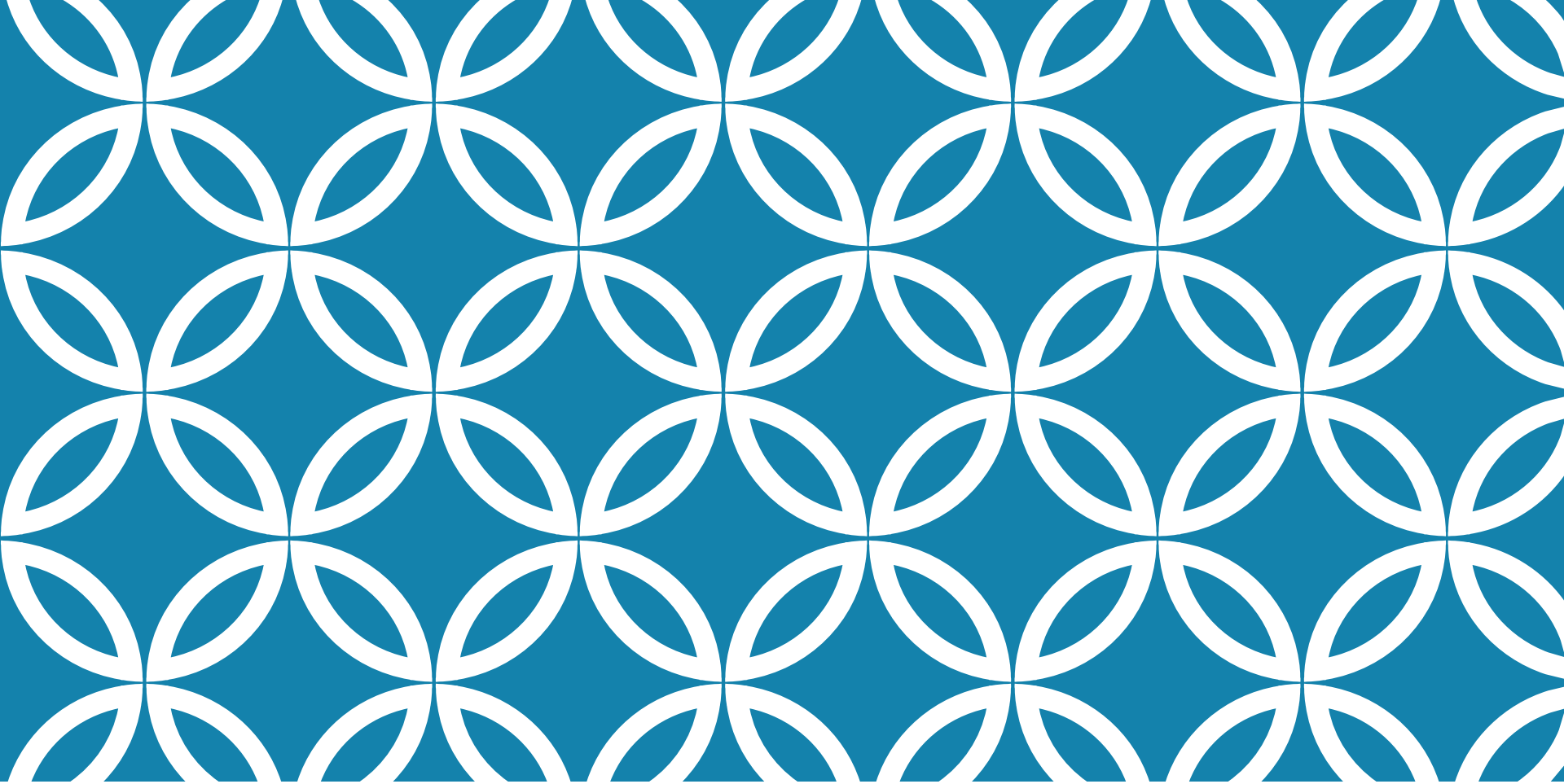
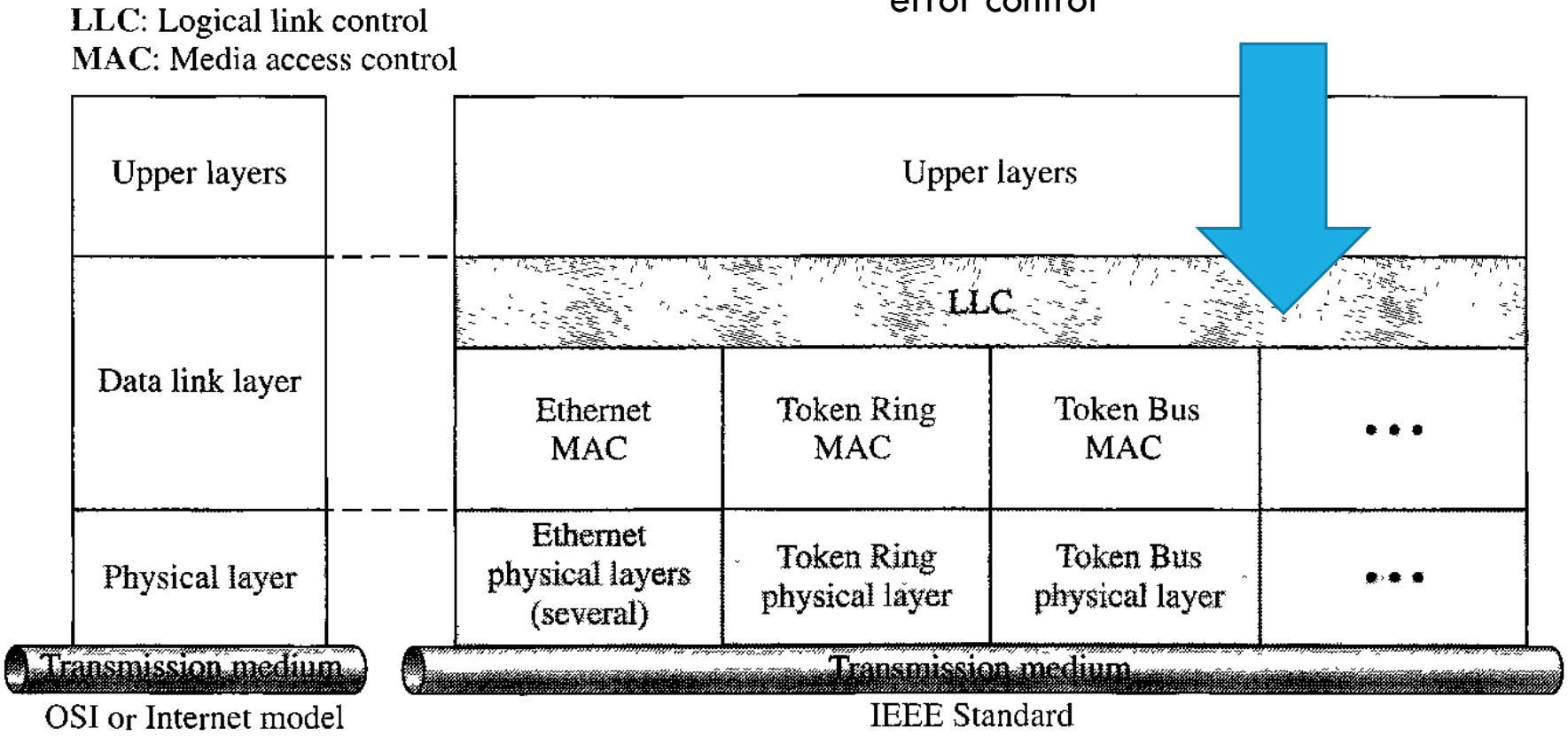
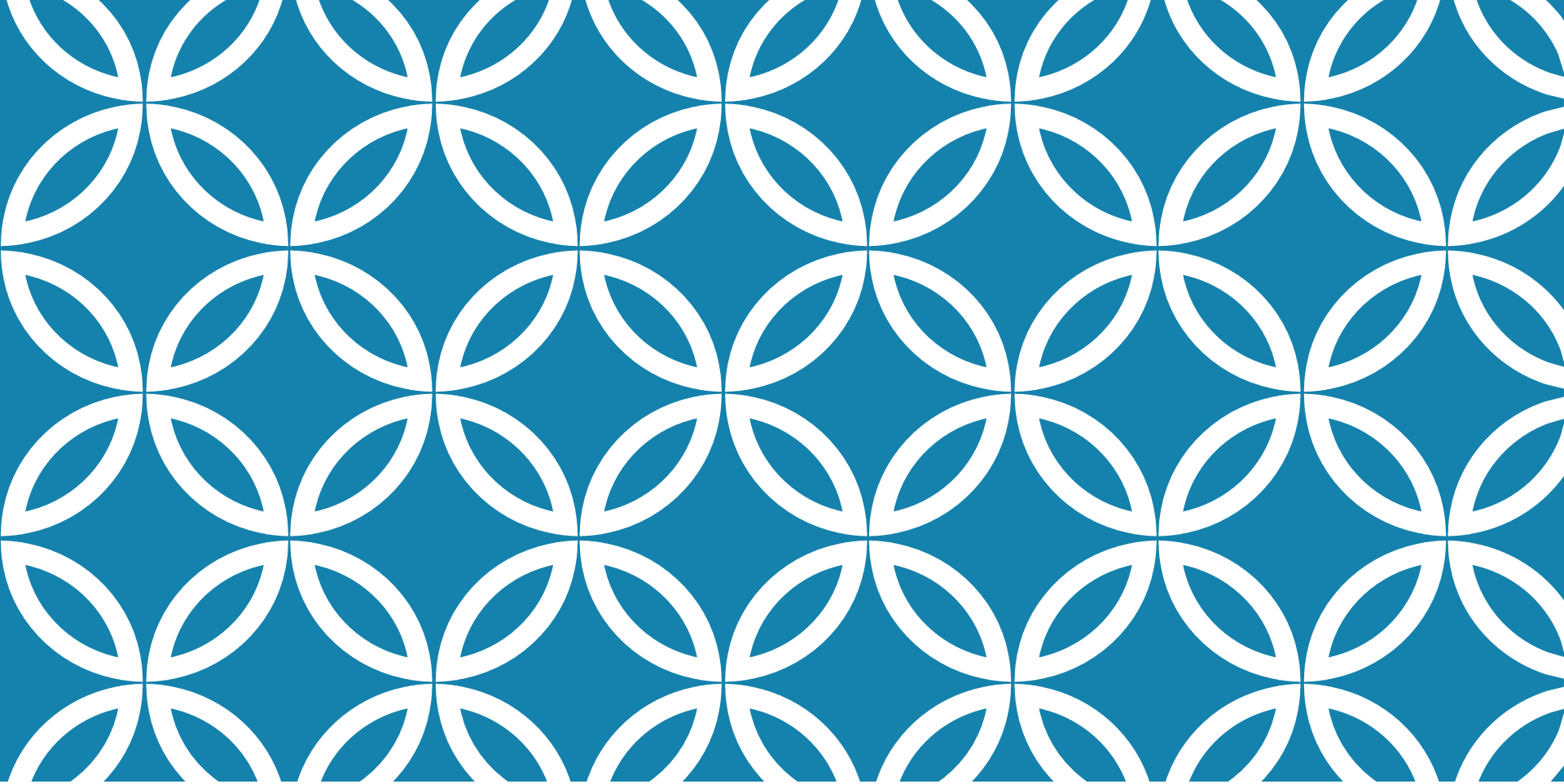


FLOW CONTROL & ERROR CONTROL

Fungsi SUBLAYER LLC pada datalink

bertanggung jawab terhadap kontrol data link, termasuk flow control dan error control

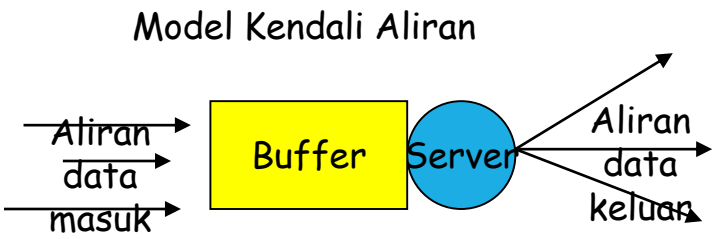




FLOW CONTROL

KENDALI ALIRAN (FLOW CONTROL)

- Fungsi lain yang diperlukan dalam mentransmisikan data di suatu link adalah kendali aliran
- Dibutuhkan terutama jika aliran data dari yang cepat ke yang lambat, dimana aliran data harus diatur agar penerima tidak overflow
- Mengatur aliran dengan cara:
  - Start – stop
  - Besarnya aliran



DUA JENIS KENDALI ALIRAN

- Start-stop
  - Aliran data diatur sesuai dengan permintaan pihak penerima, jika penerima merasa buffer penerimaannya penuh, maka ia akan mengirim sinyal stop ke pengirim, dan jika buffer penerimaannya kosong, ia akan mengirim sinyal start.
  - Teknik ini sederhana, relatif mudah di implementasikan
  - Teknik start-stop umum:
    - RTS,CTS
- Mengatur aliran
  - Aliran data diatur berdasarkan besar bandwitdh saluran saat itu, teknik ini bekerja berdasarkan feedback dari penerima yang 'mengukur' laju data yang mampu dia terima.
  - Relatif lebih rumit dari teknik start-stop
  - Contoh : (sliding) window

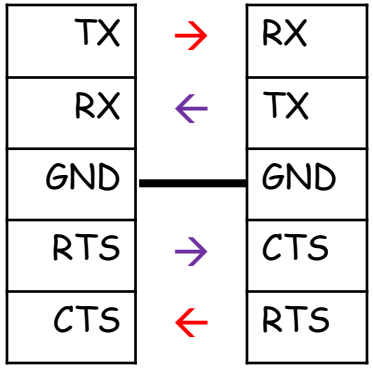
PENGGUNA KENDALI ALIRAN

- Pengguna utama adalah protokol lapis datalink (RS-232, RS-.., HDLC,…)
- Untuk teknik kendali aliran yang lebih canggih diterapkan di lapis atas seperti TCP (lapis transport)

KENDALI ALIRAN DI RS-232

- RTS – CTS (hardware), digunakan saluran tambahan untuk mengkomunikasikan informasi kendali aliran, dirancang untuk berkomunikasi dengan modem yang lebih lambat dari interface RS-232.

Koneksi fisik



Pertukaran sinyal

- RTS
- Jika dijawab CTS maka TX jika tidak tunggu

SLIDING WINDOW

- Teknik kendali aliran start-stop mempunyai kelemahan trafik yang terjadi menjadi diskrit (bisa juga bursty), menyebabkan naiknya peluang kongesti di jaringan, tidak cocok untuk komunikasi jarak jauh (melalui banyak link).
- Dikembangkan teknik pengendalian aliran yang lebih adaptif sesuai dengan kondisi jalur transmisi yang dilewati, sehingga data dapat ditransmisikan dengan jumlah yang 'cukup' tidak berlebih dan tidak kurang. Teknik ini meningkatkan efisiensi bandwidth yang pada ujungnya akan mengurangi terjadinya kongesti jaringan.
- Salah satu teknik yang ada sejak awal dibuatnya protokol internet adalah teknik sliding windows

SLIDING WINDOW

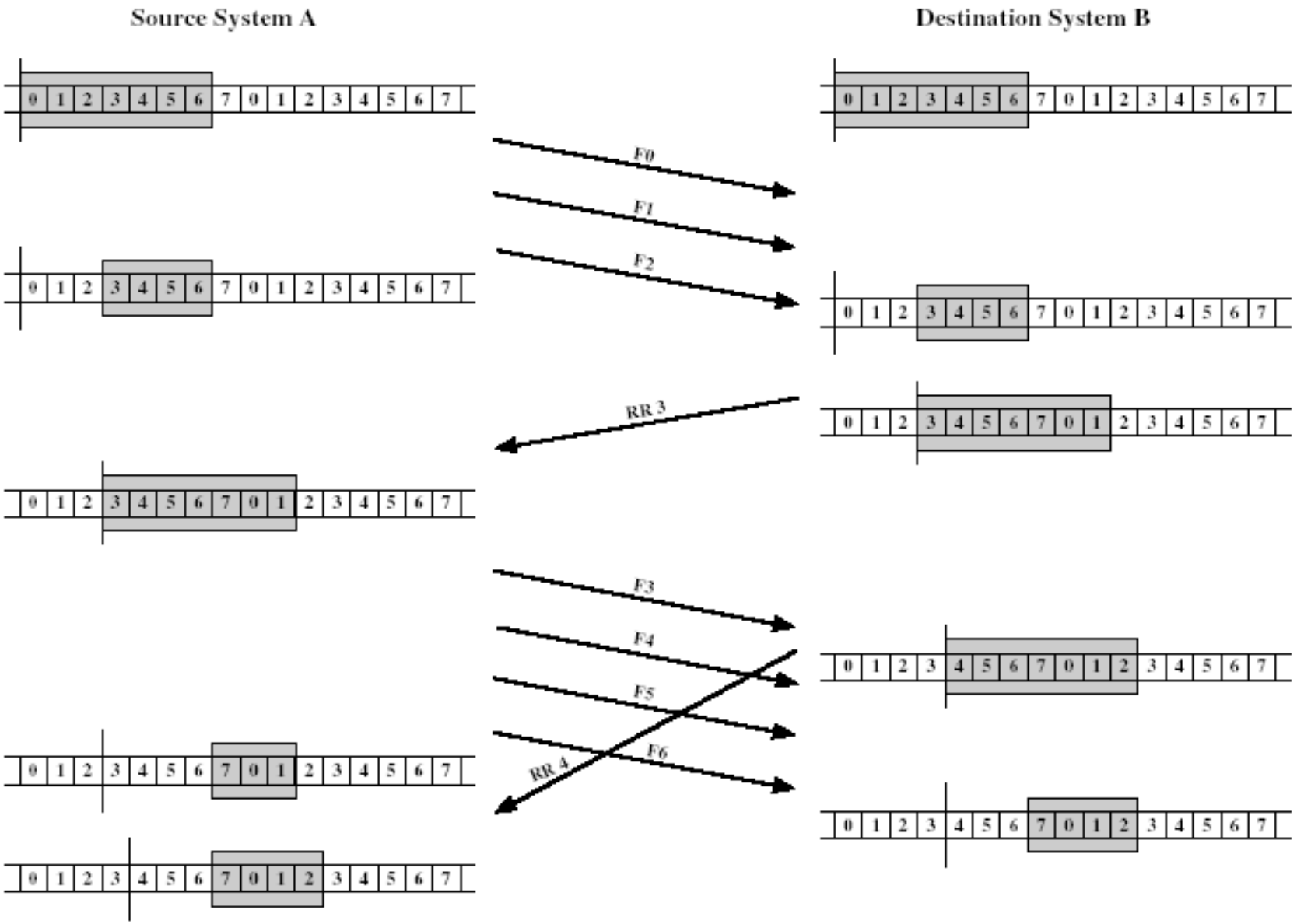
- Window = angka jumlah pengiriman paket saat ini
- Window = 3 → satu kali kirim maksimum 3 paket
- Cara kerja:
  - Penerima akan menetapkan jumlah window terimanya berdasarkan tingkat keberhasilan penerimaan paket, kebijakan yang ditetapkan oleh lapis aplikasi, dll
  - Pengirim kemudian akan mengirim paket sesuai dengan jumlah window yang ditetapkan penerima
- Pada TCP besarnya windows di'ikutkan' ke paket arah pengirim dari pihak penerima → tidak perlu paket khusus, meningkatkan efesiensi transmisi

SLIDING WINDOW

- Karena frame yang berada dalam window pengirim bisa hilang atau rusak, pengirim harus tetap menyimpan frame tersebut dalam memorinya sebagai antisipasi kemungkinan retransmisi.
- Piggybacking → teknik penumpangan balasan pada frame data untuk komunikasi 2 arah (menghemat kapasitas komunikasi).
- Sending window: jumlah deretan frame maksimum yang dapat dikirim pada suatu saat
- Receiving window: jumlah frame maksimum yang dapat diterima

SLIDING WINDOW

Contoh: ukuran window=7



SLIDING WINDOW

- Asumsi: field nomor urut 3-bit dan ukuran window maksimum 7 frame.
- Mula-mula A dan B mengindikasi bahwa A akan mengirim 7 frame, dimulai dengan frame 0 (F0)
- Setelah transmit 3 frame (F0, F1, F2) tanpa ack, A telah mengurangi window-nya menjadi 4 frame dan tetap menyimpan kopi dari ketiga frame yang baru dikirim.
- Window ini berarti A masih boleh mengirim 4 frame lagi, dimulai dari frame 3.
- Kemudian B mengirim RR3 (receive ready), yang berarti "saya telah menerima sampai frame 2 dan siap menerima 7 frame berikutnya yang dimulai dari nomor 3"
- Dengan ack ini, A mendapat ijin untuk mengirim 7 frame, serta A dapat menghapus/menghilangkan frame 0, 1, dan 2 dari buffer
- A melanjutkan pengiriman frame 3, 4, 5, dan 6.
- dst.

METODA DETEKSI KESALAHAN

- Agar bisa melakukan kendali kesalahan → mekanisme deteksi kesalahan
- Beberapa metoda yang umum digunakan:
  - Pariti → paling sederhana
  - CRC → lebih sulit, meminta kemampuan komputasi
  - Checksum → operasi word

PARITI

- Penambahan 1 bit sebagai bit deteksi kesalahan
- Terdapat 2 jenis pariti : genap dan ganjil
  - Pariti genap = jumlah bit 1 dalam kode adalah genap
  - Pariti genap = d1 xor d2 xor ….. Dn
  - Pariti ganjil = jumlah bit 1 dalam kode adalah ganjil
  - Pariti ganjil = (d1 xor d2 xor ….. Dn) xor 1
- Sistem sederhana dan mudah dibuat hardwarenya (di PC digunakan IC 74LS280)

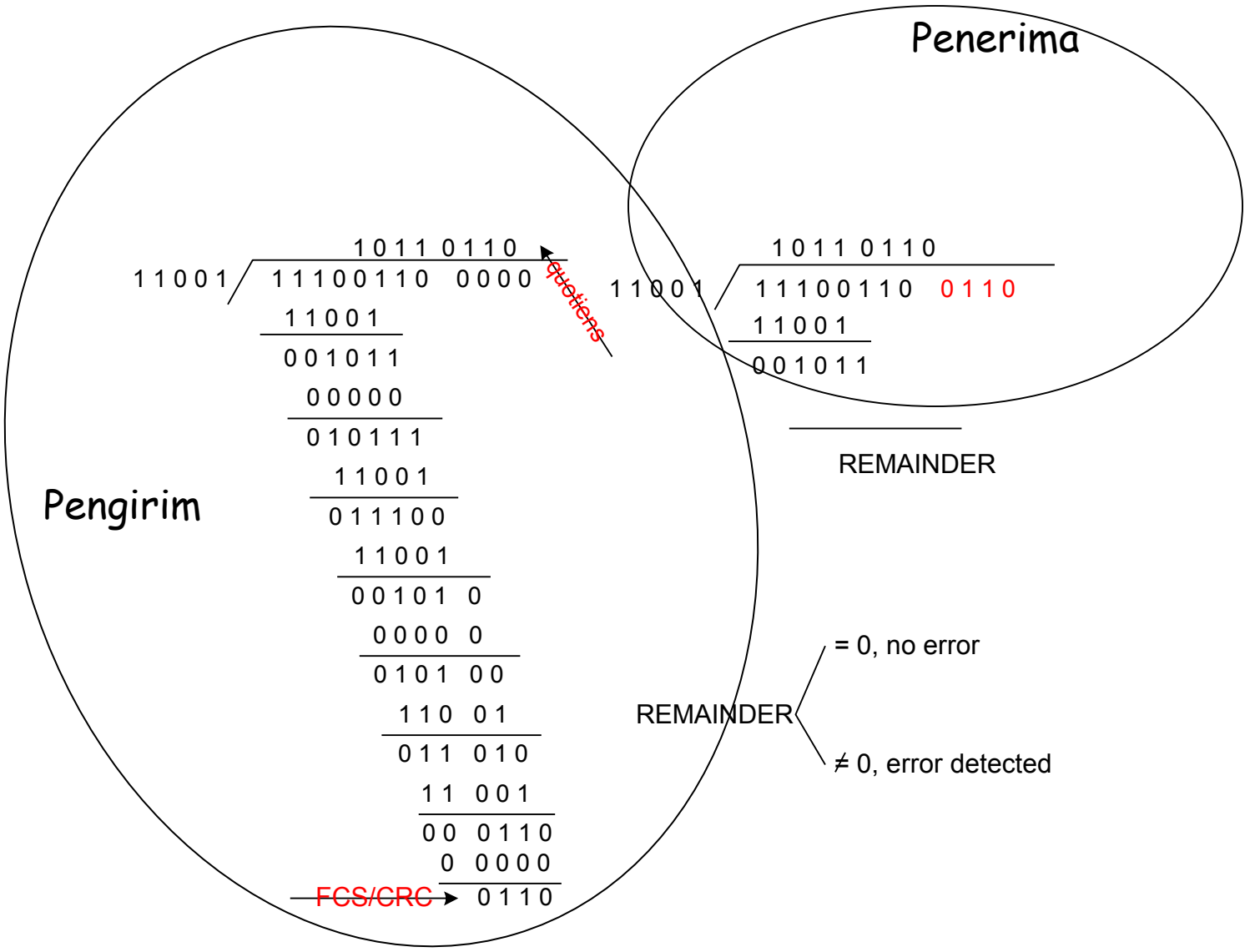
CYCLIC REDUNDANCY CHECK: SISI PENGIRIM

- Pada metode CRC ini di sisi pengirim akan dilakukan proses pembagian data dengan suatu pembagi tertentu yang disebut dengan generator *polynomial*. Sisa pembagian disebut dengan reminder Bit-bit sisa pembagian inilah yang ikut dikirimkan bersama data asli.
- Merupakan hasil operasi pembagian biner dengan suatu pembagi tertentu (generator polinomial)
  - Pembagi : Dn Dn-1 …D1
  - Deretan bit : b1 b2 b3 …. bm
  - Operasi :
    - (b1 b2 b3…bm)n-1 / Dn…D1 → sisa (Rn-1…R1)
  - Dikirim b1 b2 b3…bm Rn-1…R1

CYCLIC REDUNDANCY CHECK: SISI PENERIMA

- Oleh penerima dilakukan operasi yang sama
  - b1 b2 b3…bm Rn-1…R1 / Dn…D1 → sisa (rn-1…r1)
  - Data benar jika rn-1…r1 = 0
  - Data salah jika rn-1…r1 ≠ 0
- Pembagi standar internasional
  - CRC-16 → 11000000000000101
  - CRC-ITU → 10001000000100001
  - CRC-32 → 100000100100000010001110110110111
- Jika diperlukan pembagi boleh tidak menggunakan standar ini asal memenuhi:
  - Diawali dan diakhiri dengan bit 1 (1xxxxxx1)
  - Jumlah minimum bit "1" : 3 bit
  - Agar bisa mendeteksi jumlah bit kesalahan ganjil :harus habis dibagi oleh (11 = X + 1)

CONTOH PERHITUNGAN CRC



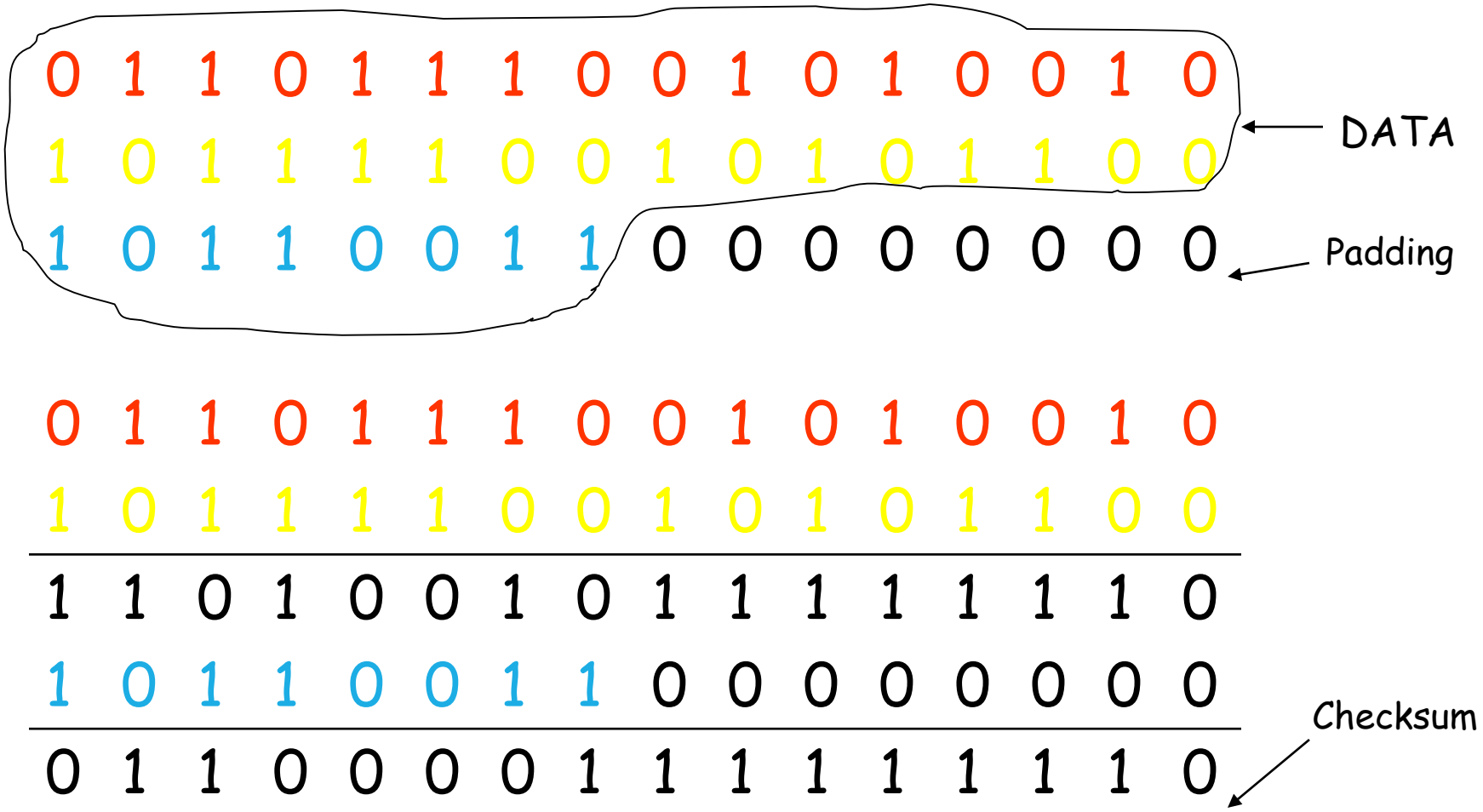
PENGGUNAAN : PADA PAKET LAN (MAC)

Destination MAC Address
Source MAC Address
Protocol/Length
Data (46 – 1500 B)
CRC-32

CHECKSUM

- CRC memerlukan perhitungan xor sebanyak jumlah bit data → memerlukan kemampuan komputasi yang cukup besar
- Diciptakan metoda checksum (untuk mengurangi perhitungan) pada beberapa jenis transmisi tidak perlu kecanggihan CRC atau sudah melakukan CRC di lapis lain
- Cara perhitungan checksum:
  - Data dibagi menjadi kelompok-kelompok 16 bit (word)
  - Word pertama di xor dengan word kedua
  - Hasil di xor dengan word ketiga, keempat, …sampai word terakhir (jika bit-bit terakhir tidak cukup untuk menjadi word, ditambahkan padding bit '0' sampai membentuk word)
  - Hasil akhir (16 bit) = checksum

CONTOH PERHITUNGAN

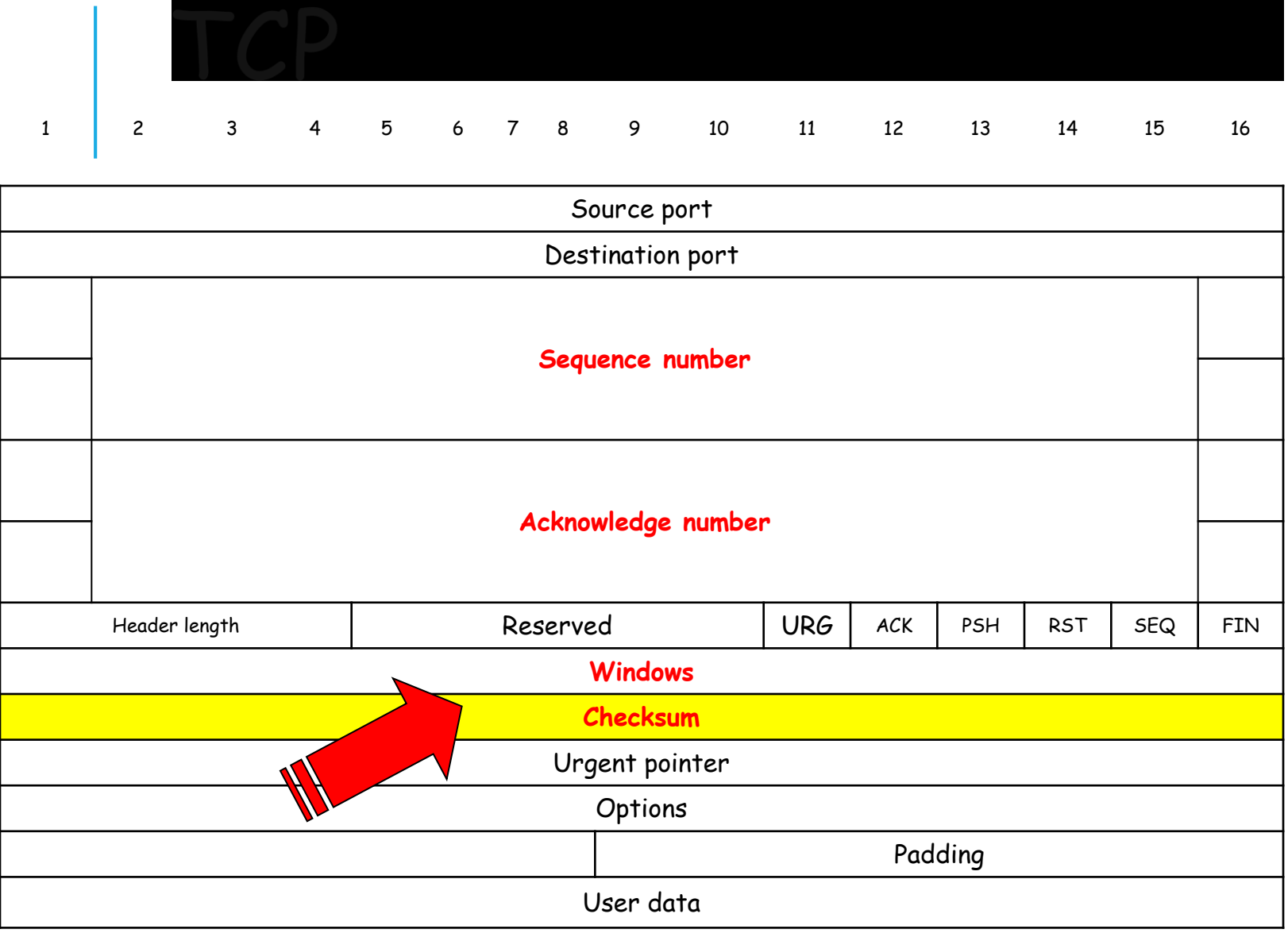


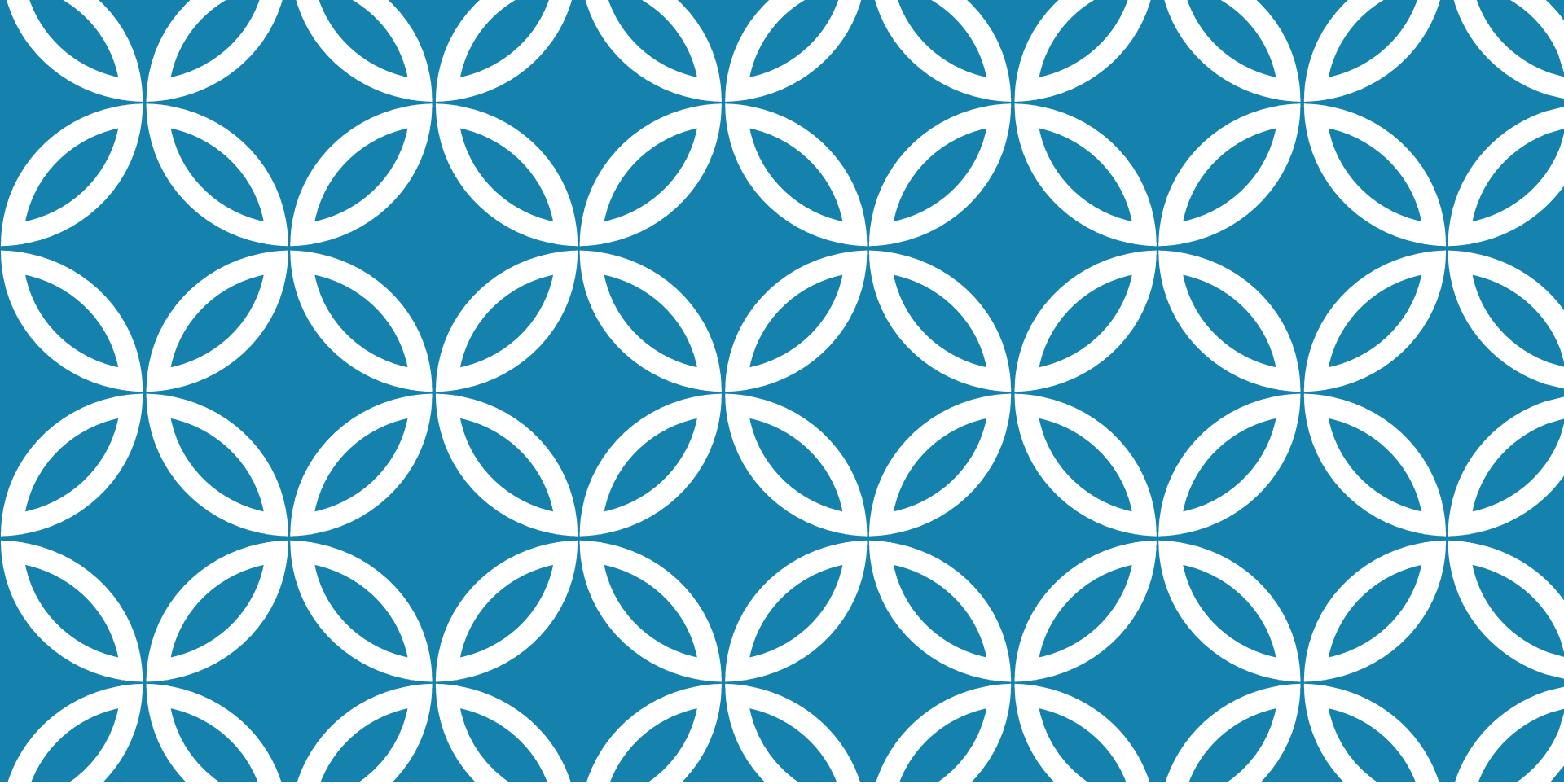
PENGGUNA CHECKSUM: IP

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
									Priority (0-7)		*low*	*high*	*high*	← "1"	

Version	Header length (dword)	Precedence	D	T	R	unused
Total length						
Identification						
D	M		*Fragment offset*			
Time to live (seconds)		Protocol				
Header checksum						
Source IP address						
Destination IP address						
Option (0 word atau lebih)						
Data ≤ 64 kB						

PENGGUNA CHECKSUM: TCP





ERROR CONTROL

TEKNIK ERROR CONTROL

BACKWARD ERROR CONTROL (BEC)

FORWARD ERROR CONTROL (FEC)

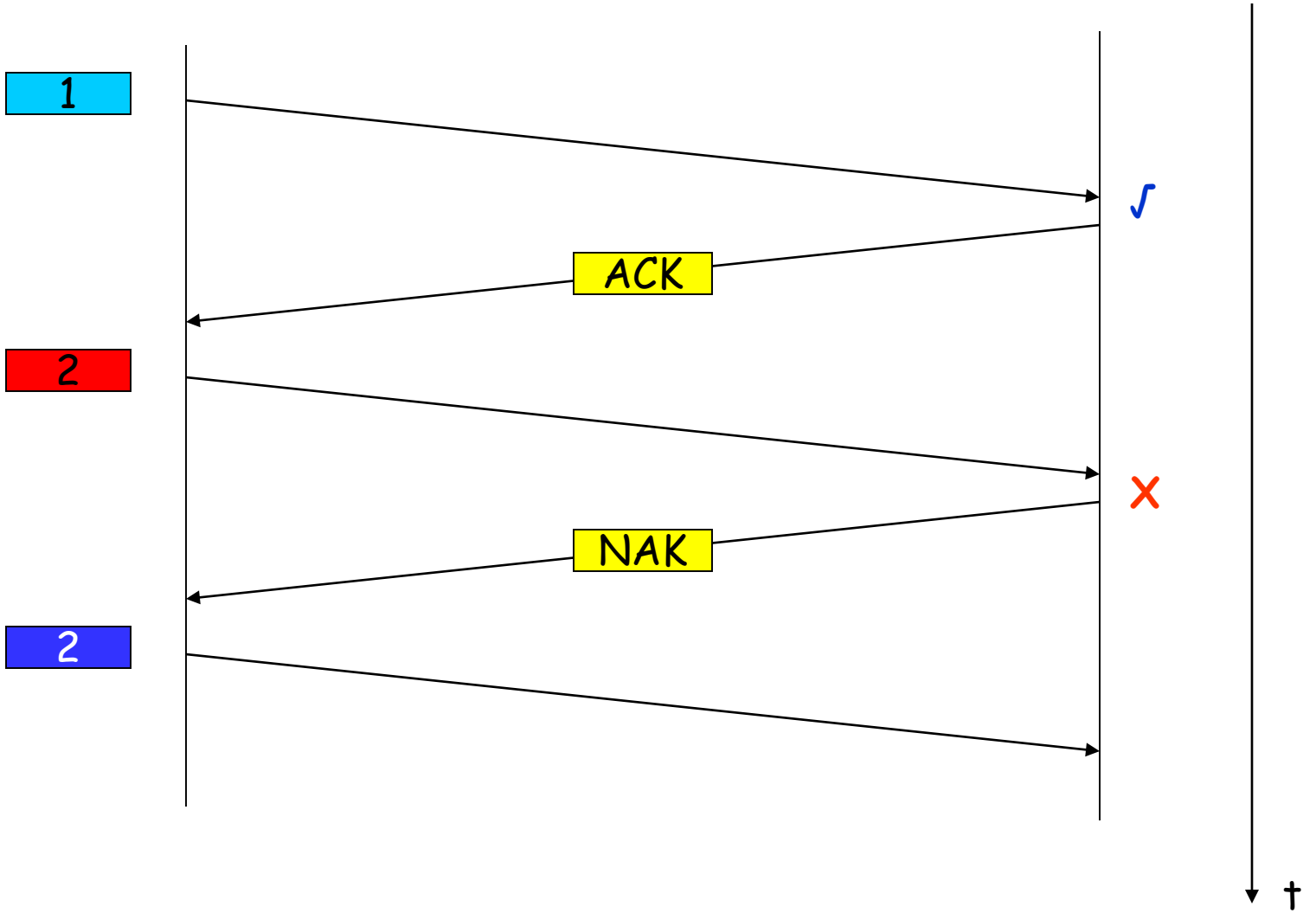
BACKWARD ERROR CONTROL (BEC)

- Kemampuan deteksi kesalahan digunakan untuk melakukan perbaikan kesalahan (error control) dengan cara meminta pengiriman ulang jika paket yang diterima salah

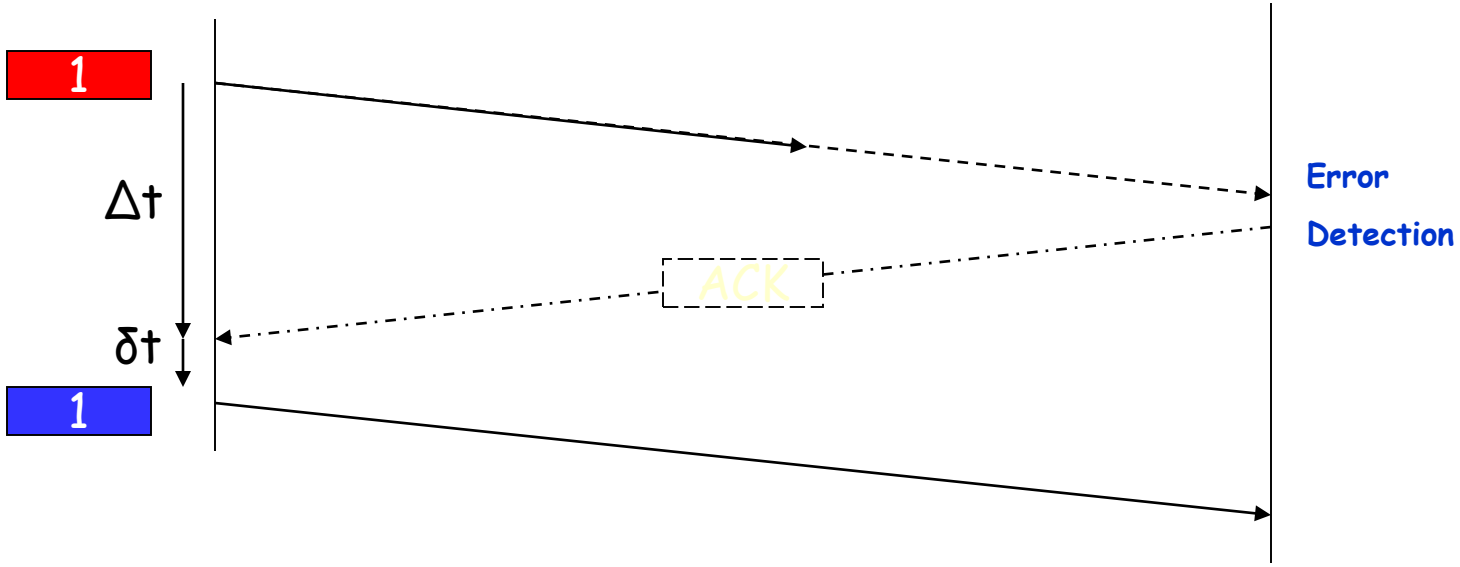
BACKWARD ERROR CONTROL: ARQ

- ARQ = Automatic Repeat reQuest
- ARQ akan mengulang / tidak mengulang pengiriman data sesuai dengan feedback dari penerima
- Feedback dari penerima
  - ACK = acknowledge → data diterima benar
  - NAK = not acknowledge → data diterima salah

ARQ : IDLE RQ



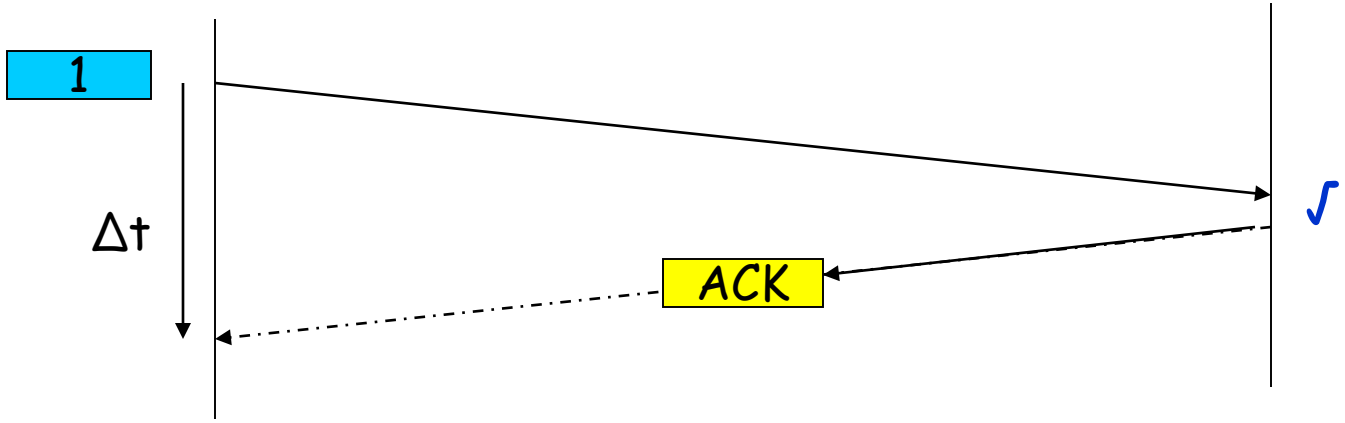
KASUS 1: JIKA PAKET TIDAK SAMPAI



Pengirim menunggu feedback sampai Δt +δt, jika tidak ada respon maka pengirim harus mengirimkan kembali paket tersebut.

Waktu tersebut disebut dengan waktu *timeout*

KASUS 2: FEEDBACK TIDAK SAMPAI



Diperlakukan sama dengan kondisi kasus 1 (time-out)

KAPANKAH PENGIRIM MENGIRIM ULANG PAKET ???

- Jika mendapat feedback NAK
- Jika timeout
- Jika mendapat feedback yang tidak dimengerti

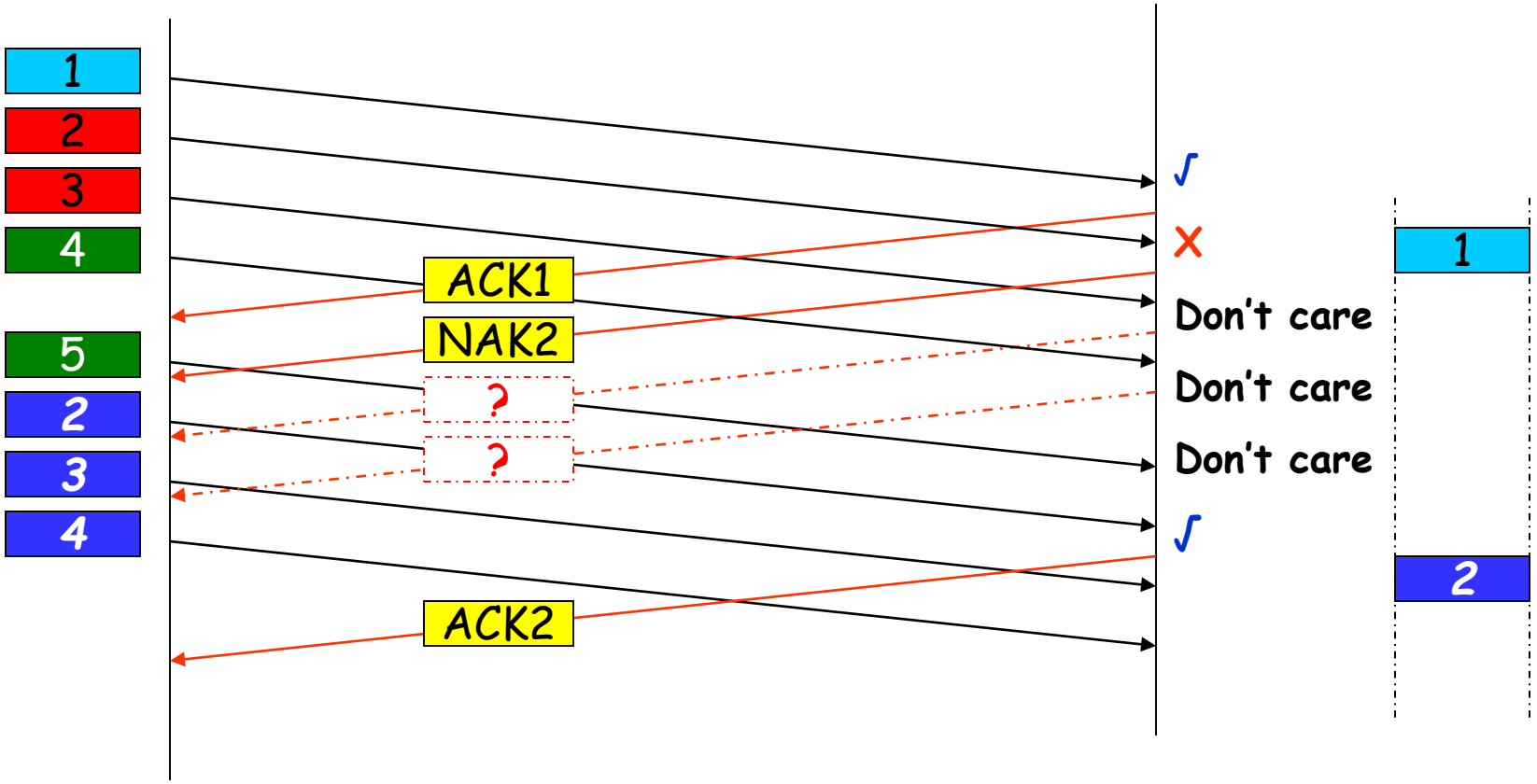
- Kesimpulan : pengirim mengirim ulang paket → Jika tidak mendapat ACK

ARQ : IDLE RQ

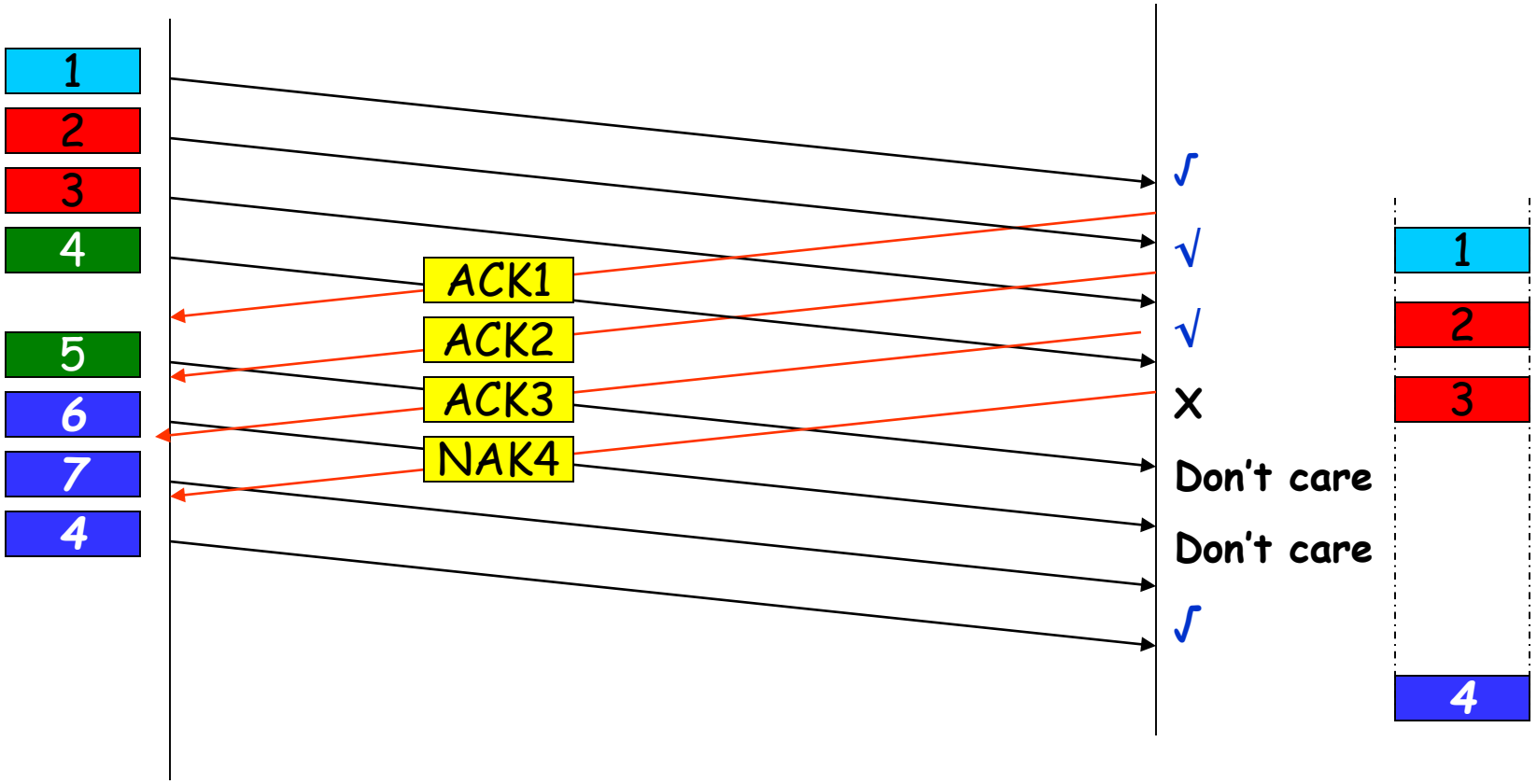
- Paket akan diterima terjaga urutannya
- Efisiensi saluran paling rendah
- Cocok digunakan untuk saluran transmisi yang sangat jelek kualitasnya (banyak error)

ARQ : GO BACK N

- Mengirim ulang mulai dari paket yang salah
- Paket akan diterima terjaga urutannya
- Efisiensi saluran lebih rendah dari Selective Repeat

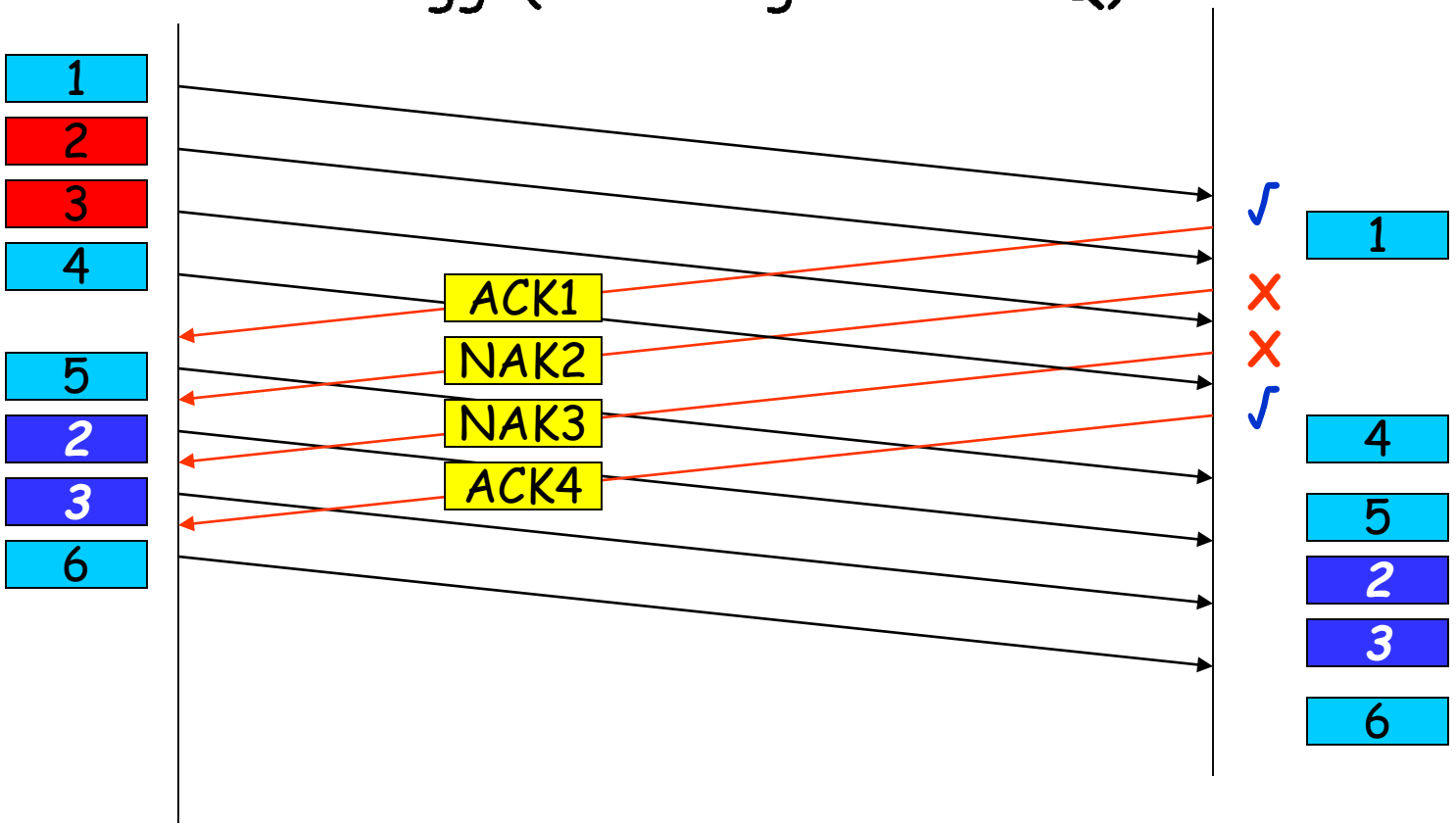


Kasus Lain Go Back N



ARQ : SELECTIVE REPEAT

- Hanya mengirim ulang untuk paket yang salah
- Paket diterima tidak berurutan
- Efisiensi saluran tinggi (dibandingkan idle RQ)



KESIMPULAN:

Backward Error control = error detection + ARQ

Kelemahan : waktu yang diperlukan untuk mengirim dengan benar adalah minimal 2 x waktu propagasi

LATIHAN

1. Diketahui urutan bit informasi sebagai berikut:

0 0 1 0 1 1 1 0 1 0 1 1 1 1 0 0

1 1 1 1 1 0 0 0 1 0 1 0 1 0 1 1

1 1 0 .

A. Lakukan metode *Check*sum pada informasi tersebut!

B. Untuk apakah metode ini dilakukan?

2. Diketahui urutan bit informasi adalah sebagai berikut

1 1 0 1 0 1 1 1 0 1 1.

Generator polynomial yang digunakan adalah 1 1 0 0 1 1.

A. Tentukanlah Data yang dikirimkan (data informasi ditambah bit-bit *redundancy*) menggunakan metode CRC!

B. Untuk apakah metode ini dilakukan?

3. Urutan data yang diterima oleh PC Penerima adalah :

1 1 1 1 1 0 0 0 0 1 1 1 0 . Digunakan generator polinomial seperti pada nomor soal nomor 2! Periksalah apakah data yang sampai mengalami *error* atau tidak!

FORWARD ERROR CONTROL

FORWARD ERROR CONTROL

- Backward EC menyebabkan delay pengiriman paket yang cukup besar tergantung dari berapa kali paket tersebut harus dikirim
- Untuk sistem transmisi jarak jauh dimana delay propagasi sangat besar (kelas detik, menit atau jam) BEC tidak bisa menjadi pilihan
- Juga untuk aplikasi multimedia, dimana ketepatan waktu kedatangan lebih utama dibandingkan dengan 'kebenaran' data, BEC menyebabkan delay yang lewat batas toleransi waktu
- Dipergunakan Forward Error Correction (FEC) untuk memecahkan masalah ini
- FEC berprinsip dasar: penerima mampu membetulkan sendiri kesalahan data yang sudah diterima, karena selain menerima data juga menerima bit-bit redundansi yang diperlukan

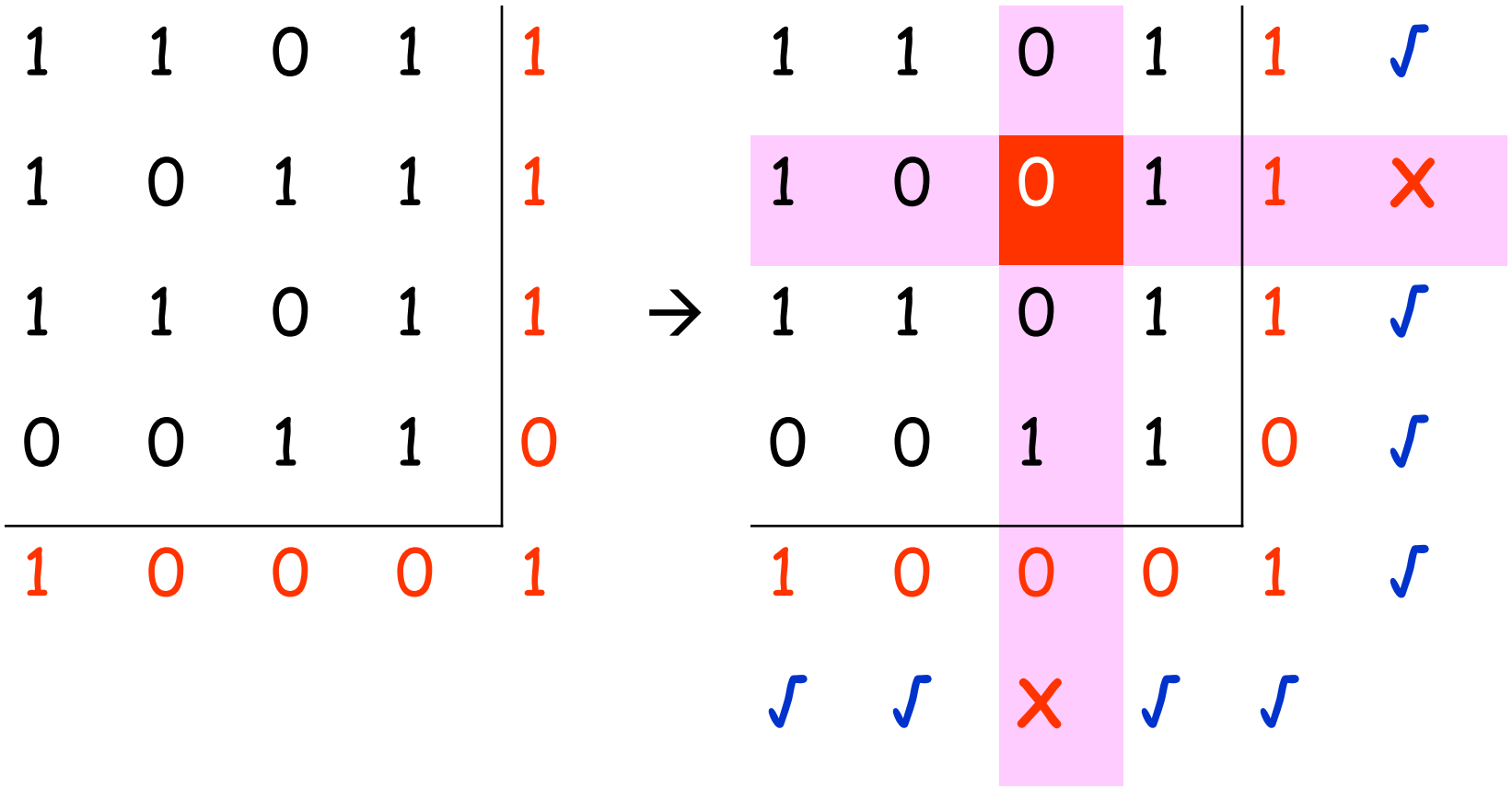
JENIS-JENIS FEC

- Metoda FEC yang umum dikenal :
  - Block Parity
  - Hamming Code
  - Turbo Code, RS Code, BCH Code

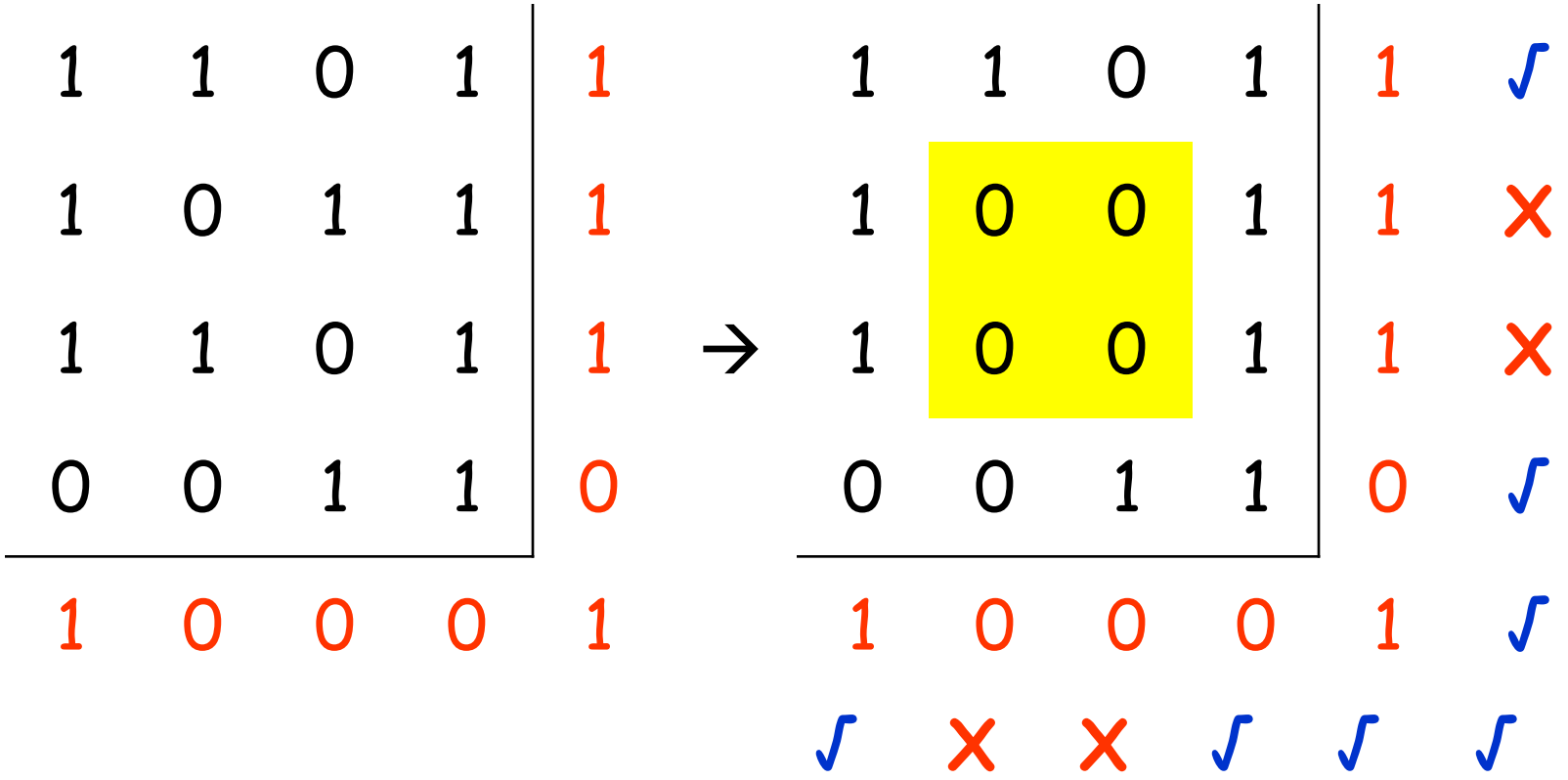
BLOCK PARITY

- Sederhana, menggunakan perhitungan pariti dasar
- Menggunakan pariti baris dan kolom sebagai sarana koreksi kesalahan
- Hanya mampu mengkoreksi kesalahan 1 bit, mampu mendeteksi kesalahan lebih dari 1 bit
- Efisiensi tergantung dari ukuran baris dan kolom yang digunakan, semakin banyak baris dan kolom akan semakin banyak bit pariti

CONTOH BLOCK PARITY 1



CONTOH BLOCK PARITY 2



CONTOH BLOCK PARITY 3

HAMMING CODE

Hamming Code diciptakan oleh Richard Wesley Hamming, seorang ahli matematika Amerika

HAMMING CODE: SISI PENGIRIM(1)

- Menggunakan metoda matematik modulo 2
- Langkah-langkah Hamming code di sisi pengirim :

1. Disisipkan bit-bit pariti di posisi bit 2^n : bit ke 1,2,4,8,16,32 dst

Sehingga deretan bit →

P_1 P_2 d_1 P_3 d_2 d_3 d_4 P_4 d_5 d_6 d_7 d_8 d_9 dst

HAMMING CODE: SISI PENGIRIM(2)

2. Lakukan *parity check* dengan memperhatikan letak bit-bit yang diperiksa.

Ketentuan bit yang diperiksa: skip (n-1) bit, check n bit, skip n bit, check n bit, dst.. [n = posisi bit parity]

Posisi bit	1	2	3	4	5	6	7	8	9	10	11	12	13
Kategori	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9
p1	√		√		√		√		√		√		√
p2		√	√			√	√			√	√		
p3				√	√	√	√					√	√
p4								√	√	√	√	√	√

HAMMING CODE: SISI PENGIRIM(3)

3. Lakukan langkah XOR untuk semua bit yang posisinya telah ditandai. Bit hasil XOR ini adalah bit paritynya.

4. Data dikirimkan dengan bit-bit parity yang telah disisipkan.

HAMMING CODE SISI PENERIMA (1)

1. Untuk menentukan posisi bit informasi dan *parity*, gunakan ketentuan seperti pada langkah 1 dan 2 metode Hamming di sisi pengirim.

2. kemudian lakukan proses xor untuk bit-bit sesuai ketentuan pada langkah ke-3 metode Hamming seperti di sisi pengirim.

HAMMING CODE SISI PENERIMA (2)

Posisi bit	1	2	3	4	5	6	7	8	9	10	11	12	13	Proses Xor
Kategori	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	
Bit informasi	0	1	1	1	0	0	1	1	0	1	1	1	1	
p1	√		√		√		√		√		√		√	p1=0 xor 1 xor 0 xor 1 xor 0 xor 1 xor 1 = 0
p2		√	√			√	√			√	√			p2= 1 xor 1 xor 0 xor 1 xor 1 xor 1 = 1
p3				√	√	√	√					√	√	p3= 1 xor0 xor 0 xor 1 xor 1 xor 1 = 0
p4								√	√	√	√	√	√	P4= 1 xor 0 xor 1 xor 1 xor 1 xor 1 = 1

Hasil xor jika dilihat dari mulai urutan pertama sampai keempat adalah 0 1 0 1. Urutan bit ini dibaca terbalik, yaitu 1010 sama dengan nilai 10 dalam desimal. Artinya, ada yang salah yaitu bit ke-10

METODA FEC LAIN

- Semua metoda FEC pada dasarnya menggunakan metoda matematik modulo 2
- Metoda ini terus dikembangkan dengan tujuan:
  - Mendapatkan kemampuan koreksi bit yang semakin banyak
  - Dengan mengurangi jumlah bit pariti yang dibutuhkan
  - Mampu melanjutkan komunikasi walaupun sempat terputus.
- Metoda yang umum digunakan:
  - BCH Code
  - Reed Solomon Code
  - Convolutional Code
  - Trellis Code
  - Turbo Code

LATIHAN

1. Diketahui urutan data informasi:

1 0 0 0 1 1 1 1 0 1 1 1 1 1 0 0 0 0

A. Tentukan data yang dikirim (data informasi ditambah dengan bit-bit *parity*) jika digunakan *Hamming Code*!

B. Untuk apakah metode ini digunakan?

2. Diketahui urutan data yang sampai di penerima adalah:

1 0 1 0 0 1 1 1 0 1 1 1 1 1 1 0 0. Jika digunakan *Hamming Code* pada jaringan,

A. tentukanlah apakah data yang diterima tersebut mengalami *error* atau tidak?

B. Jika iya, perbaiki agar data menjadi benar!

HAPPY LEARNING!!