13 Data Communication

Week 13 Data Link Layer (Error Correction)

Susmini I. Lestariningati, M.T

Error Correction

- Error correction may generally be realized in two different ways:
  - **Forward error correction (FEC):**
    - The sender encodes the data using an error-correcting code (ECC) prior to transmission. The additional information (redundancy) added by the code is used by the receiver to recover the original data. In general, the reconstructed data is what is deemed the "most likely" original data.
  - **Automatic repeat request (ARQ)**
    - (sometimes also referred to as **Backward Error Correction**): This is an error control technique whereby an error detection scheme is combined with requests for retransmission of erroneous data. Every block of data received is checked using the error detection code used, and if the check fails, retransmission of the data is requested – this may be done repeatedly, until the data can be verified.
  - ARQ and FEC may be combined, such that minor errors are corrected without retransmission, and major errors are corrected via a request for retransmission: this is called hybrid automatic repeat-request (HARQ).

Error Control

- Berfungsi untuk mendeteksi dan memperbaiki eror-eror yang terjadi dalam transmisi frame-frame.
- Ada 2 tipe eror yang mungkin:
  - Frame hilang : suatu frame gagal mencapai sisi yang lain
  - Frame rusak : suatu frame tiba tetapi beberapa bit-bitnya eror

- Metode Backward Error Control menyebabkan delay pengiriman yang cukup besar, karena memerlukan waktu untuk mengirimkan feedback dan kemudian mengirimkan kembali paket jika tidak diterima dengan benar. Untuk trasmisi jarak jauh di mana delay propagasi sangat besar, metode BEC ini tidak dapat digunakan. Metode Forward Error Control (FEC) memecahkan masalah tersebut. Pada FEC, error yang terjadi saat pengiriman akan diperbaiki sendiri oleh penerima. Agar hal ini dapat dilakukan, penerima harus dapat mendeteksi error yang terjadi dan letaknya. Setelah penerima tahu letak bit yang error, maka penerima dapat membetulkannya tanpa harus meminta pengirim untuk mengirim ulang paket tersebut.
- Beberapa metode FEC antara lain Block Parity, Hamming Code, Turbo Code, RS Code, BCH Code, dan lain-lain. Yang akan dibahas pada buku ini adalah metode Block Parity dan Hamming Code.

Hamming Code

- Kode Hamming merupakan kode non-trivial untuk koreksi kesalahan yang pertama kali diperkenalkan.
- Kode ini dan variansinya telah lama digunakan untuk kontrol kesalahan pada sistem komunikasi digital.
- Kode Hamming biner dapat direpresentasikan dalam bentuk persamaan:

$$\mathbf{(n,k) = (2^m-1, 2^m-1-m)}$$

- Contoh:

  jika m = jumlah paritas = 3

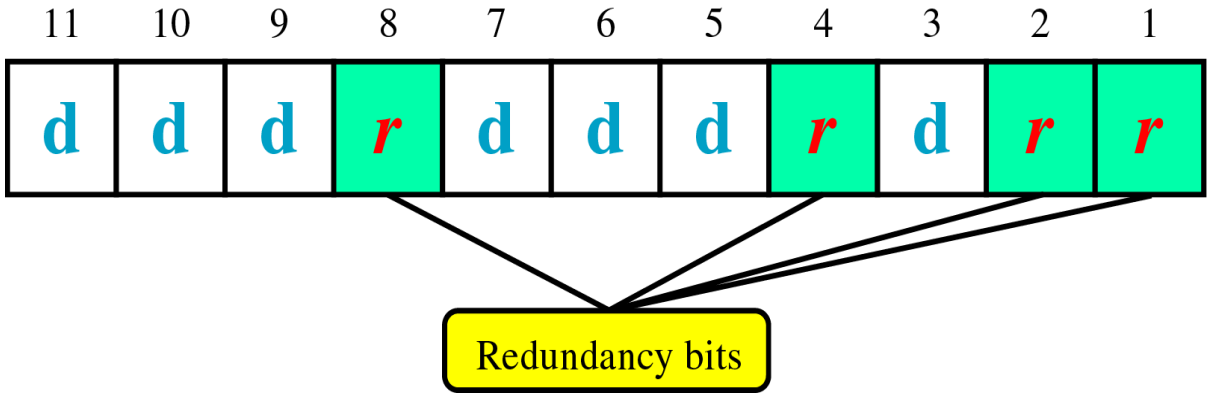
  k = jumlah data = 4

  n = jumlah bit informasi yang membentuk n sandi

  = 7

  maka kode Hamming nya adalah C (7,4) dengan dmin =3

- Error Correcting codes dinyatakan sebagai penerusan koreksi kesalahan untuk mengindikasikan bahwa pesawat penerima sedang mengoreksi kesalahan.
- Kode pendeteksi yang paling banyak digunakan merupakan kode Hamming.
- Posisi bit-bit Hamming dinyatakan dalam 2^n dengan n bilangan bulat sehingga bit-bit Hamming akan berada dalam posisi 1, 2, 4, 8, 16, dst..



- Hamming Code diciptakan oleh Richard Wesley Hamming, seorang ahli matematika Amerika. Hamming Code ini menggunakan bit- bit tambahan (disebut juga bit parity) yang disisipkan pada posisi tertentu. Secara detail, langkah-langkah metode ini di sisi pengirim adalah sebagai berikut:

  1. Tentukan posisi bit-bit tambahan (bit-bit parity) dengan aturan peletakan pada posisi $2n$, di mana n adalah bilangan bulat. Jadi, letak bit-bit parity adalah pada posisi ke 1, 2, 4, 8, 16, 32, dan seterusnya. Bit-bit parity yang dibutuhkan tergantung jumlah data.
  2. Bit-bit selain di posisi 1, 2, 4, 8 dan seterusnya adalah bit-bit informasi. Jika informasi terdiri dari 8 bit, maka setelah disisipkan bit-bit parity akan diperoleh urutan sebagai berikut p1, p2, d1, p3, d2, d3, d4, p4, d5, d6, d7, d8. Di sini p adalah bit parity dan d adalah bit data.
  3. Lakukan parity check dengan memperhatikan letak bit-bit yang diperiksa sesuai Tabel pada halaman berikut.

Posisi Bit Hamming

Bit position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Encoded data bits		p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	d10	d11	p5	d12	d13	d14	d15	
Parity bit coverage	p1	X		X		X		X		X		X		X		X		X		X		
	p2		X	X			X	X			X	X			X	X			X	X		...
	p3				X	X	X	X					X	X	X	X					X	
	p4								X	X	X	X	X	X	X	X						
	p5																X	X	X	X	X	

Langkah-langkah Kode Hamming

- Tandai semua posisi bit 2^n sebagai bit redundancy (yaitu posisi 1, 2, 4, 8, 16, 32, 64, …)
- Posisi bit sisanya selain no 1 diatas adalah posisi bit yang akan dipakai (yaitu posisi 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, …)
- Setiap bit paritas: (no.1)
  - Posisi 1 (r1): periksa tiap satu bit dari bit pertama, lalu lompati tiap satu bit, periksa 1 bit berikutnya, lompati 1 bit, dan seterusnya. (1, 3, 5, 7, 9, 11, 13, 15, 17, …)
  - Posisi 2 (r2): periksa tiap dua bit dari bit kedua, lompati 2 bit, periksa lagi bit berikutnya dan seterusnya (2, 3, 6, 7, 10, 11, 14, 15, …)
  - Posisi 4 (r4): periksa tiap 4 bit dari bit keempat, lompati 4 bit berikutnya, cek 4 bit, dan seterusnya (4,5,6,7,12,13,14,15,20,21,22,23, …)
  - Posisi 8 (r8): periksa setiap 8 bit dari bit kedelapan, lompati 8 bit berikutnya, periksa 8 bit berikutnya dan seterusnya (8-15, 24-31, 40-47, …)
  - Posisi 16 (r16): periksa setiap 16 bit dari bit ke-enambelas, lompati 16 bit berikutnya, periksa lagi 16 bit, dan seterusnya (16-31, 48-63, 80-95, …)
  - Posisi 32 (r32): periksa setiap 32 bit dari bit keenambelas, lompati 32 bit berikutnya, periksa lagi 32 bit, dan seterusnya (32-63, 96-127, 160-191, …)
  - Set bit paritas 1 jika total bit 1 ganjil, set bit paritas 0 jika jumlah bit 1 nya adalah genap.

Example 1

A byte of data: 10011010

Create the data word, leaving spaces for the parity bits: _ _ 1 _ 0 0 1 _ 1 0 1 0

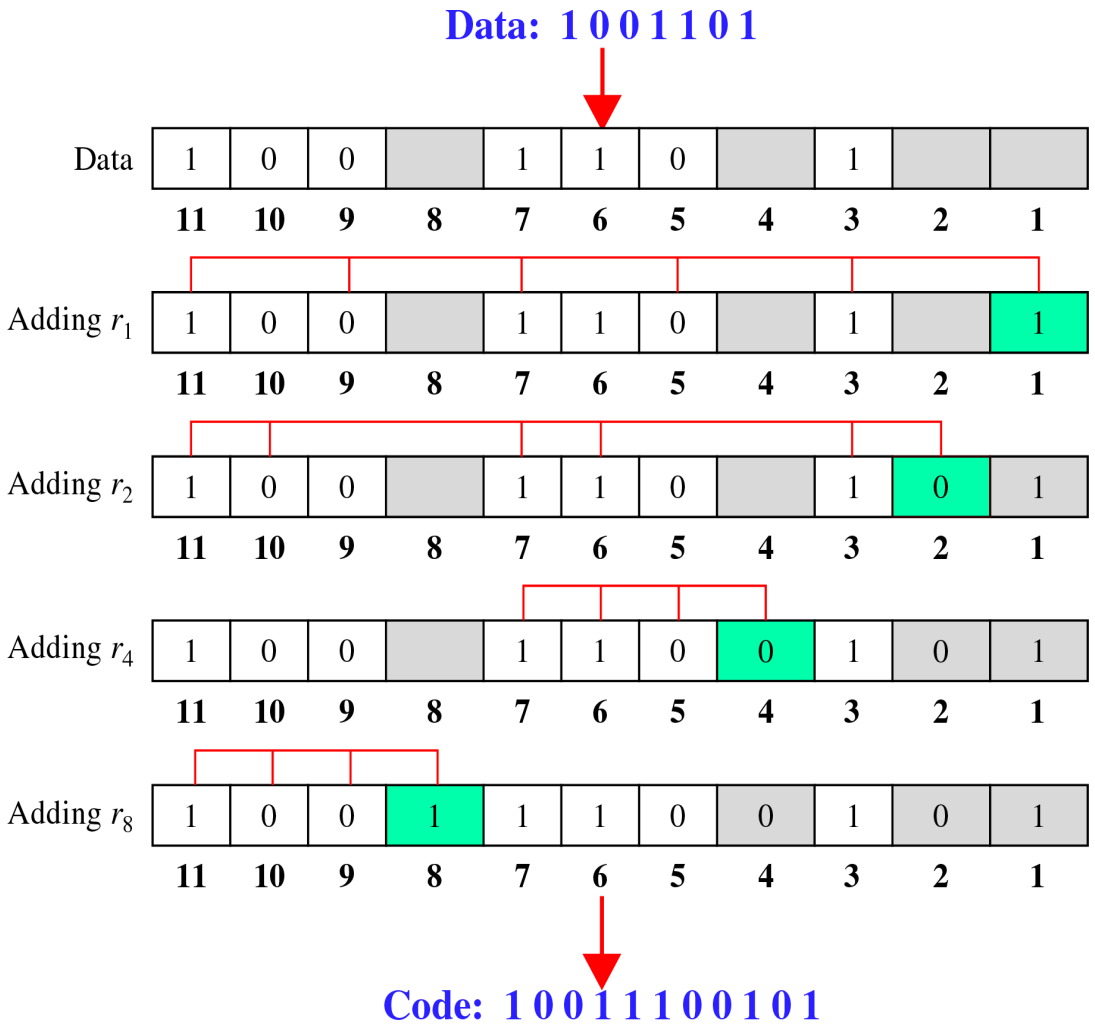
Calculate the parity for each parity bit (a ? represents the bit position being set):

- Position 1 checks bits 1, 3, 5, 7, 9, 11:
  ? _ **1** _ **0** 0 **1** _ **1** 0 **1** 0. Even parity so set position 1 to a 0: 0 _ 1 _ 0 0 1 _ 1 0 1 0
- Position 2 checks bits 2, 3, 6, 7, 10, 11:
  0 **? 1** _ 0 **0 1** _ 1 **0 1** 0. Odd parity so set position 2 to a 1: 0 1 1 _ 0 0 1 _ 1 0 1 0
- Position 4 checks bits 4, 5, 6, 7, 12:
  0 1 1 **? 0 0 1** _ 1 0 1 **0**. Odd parity so set position 4 to a 1: 0 1 1 1 0 0 1 _ 1 0 1 0
- Position 8 checks bits 8, 9, 10, 11, 12:
  0 1 1 1 0 0 1 **? 1 0 1 0**. Even parity so set position 8 to a 0: 0 1 1 1 0 0 1 0 1 0 1 0

- Code word: **0 1** 1 **1** 0 0 1 **0** 1 0 1 0.

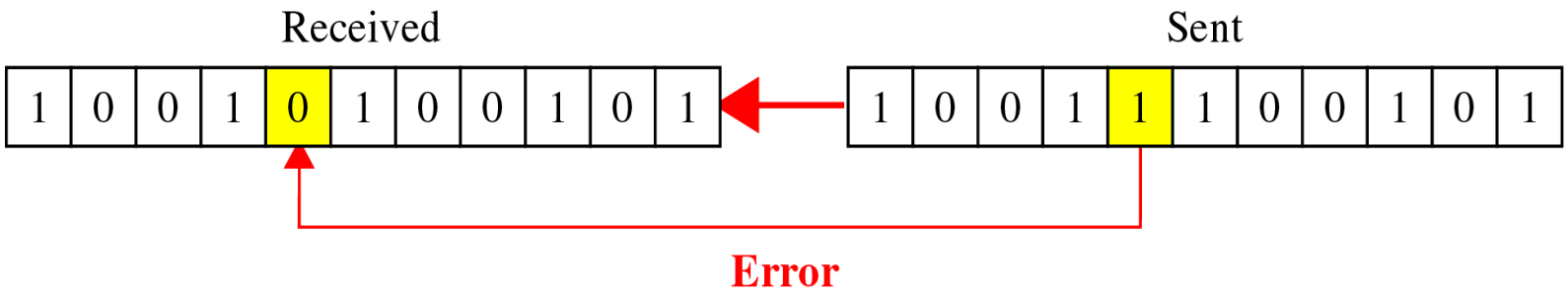
Example 2

- Example of Hamming Code

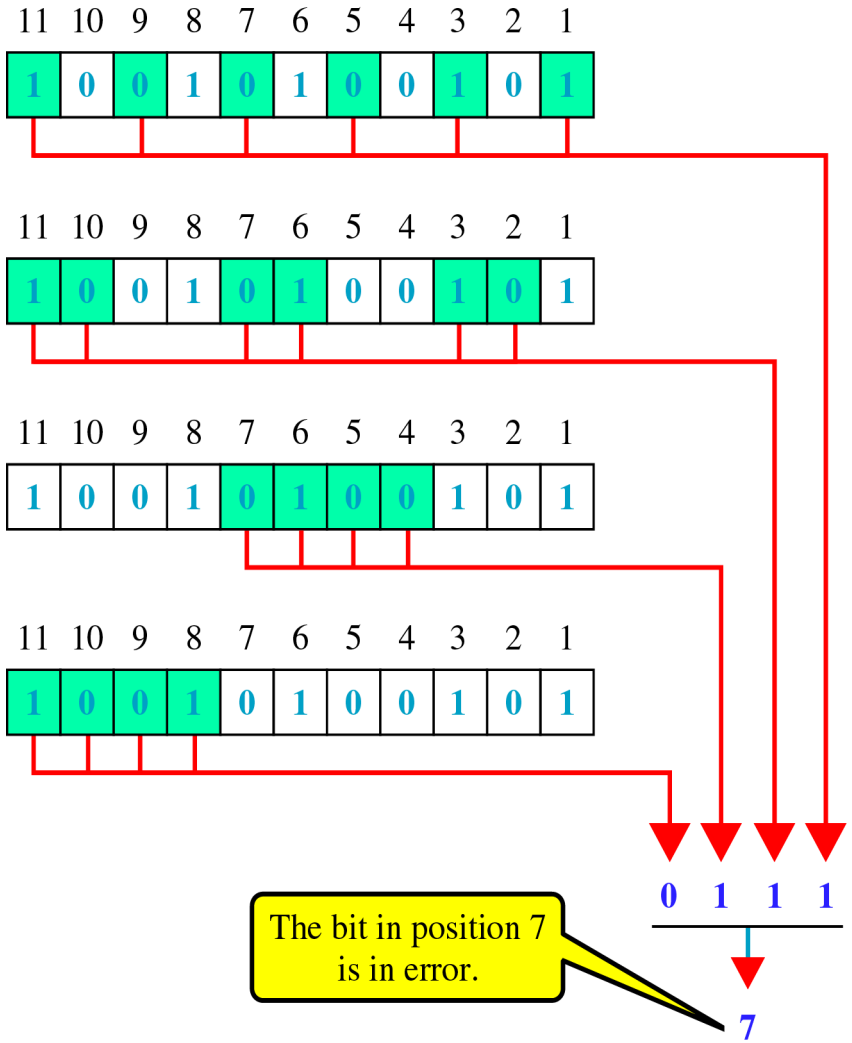


Example 3

- Jika bit yang dikirim dan sampai di penerima mengalami error pada bit tertentu, hitung dengan menggunakan kode Hamming dimana posisi bit yang mengalami kerusakan!



- Dengan menggunakan kode Hamming



Metoda Forward Error Control Lainnya

- Beberapa metode FEC lainnya adalah Turbo Code, Low Density Parity Check (LDPC) Code, dan Reed Solomon Code. Turbo Code dikembangkan pada tahun 1993, kebanyakan digunakan pada teknologi komunikasi menggunakan satelit dan juga CDMA 2000 1x dan EVDO.
- Low Density Parity Check (LDPC) Code mula-mula diperkenalkan oleh Robbert G. Gallager pada thesisnya tahun 1960. Saat ini LDPC code digunakan pada standar komunikasi kecepatan tinggi seperti DVB-S2 dan WiMAX.
- Reed-Solomon ditemukan oleh Irving S. Reed dan Gustave Solomon. Reed-Solomon Code dapat mendeteksi mengoreksi multiple random symbol error. Code ini banyak digunakan pada peralatan elektronik seperti CD, DVD, Blu-Ray Disc dan transmisi data pada DSL dan WiMAX.

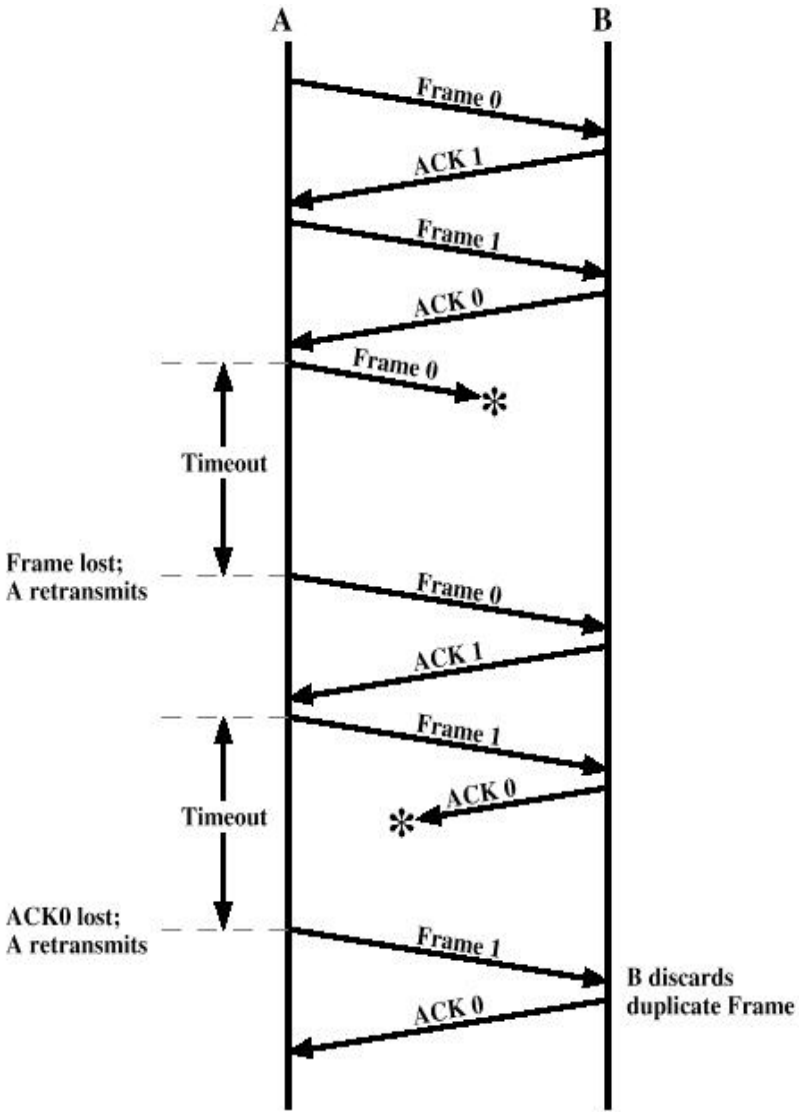
Automatic Repeat Request (ARQ)

- Metode Backward Error Control (BEC) merupakan suatu metode kendali kesalahan di mana penerima akan melakukan pengecekan terhadap data yang datang. Jika data yang diterima diketahui telah mengalami error, maka penerima akan meminta pengirim untuk mengirimkan kembali data tersebut sehingga data yang diterima benar. Jadi, pada metode ini, keputusan untuk mengirimkan ulang atau tidak tergantung feedback dari penerima. Saat pengirim mengirimkan data, maka pengirim akan menyimpan salinan data tersebut sampai ada feedback dari penerima bahwa data telah tiba dengan benar ke penerima. Ada beberapa metode BEC yang umum digunakan, yaitu

  1. Idle Automatic Repeat Request (idle ARQ) atau Stop and Wait ARQ,
  2. Go Back N ARQ
  3. Selective Repeat.

Idle ARQ / Stop and Wait ARQ

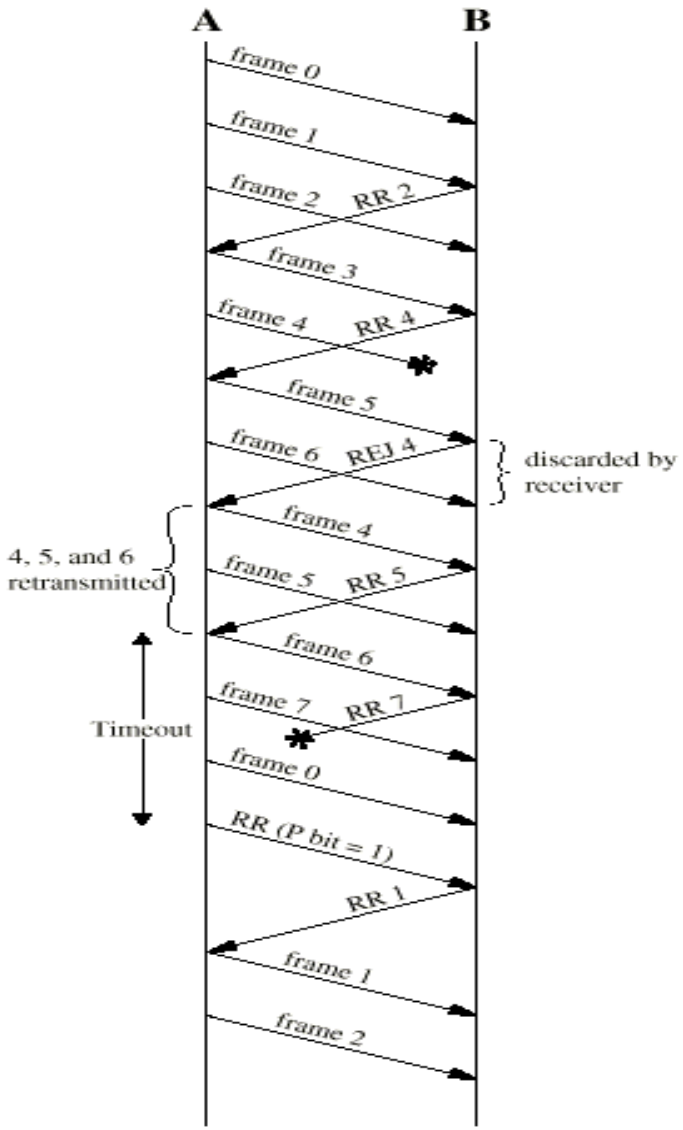
- Stasiun sumber mentransmisi suatu frame tunggal dan kemudian harus menunggu suatu acknowledgment (ACK) dalam periode tertentu. Tidak ada data lain dapat dikirim sampai balasan dari stasiun tujuan tiba pada stasiun sumber. Bila tidak ada balasan maka frame ditransmisi ulang. Bila error dideteksi oleh tujuan, maka frame tersebut dibuang dan mengirim suatu Negative Acknowledgment (NAK), yang menyebabkan sumber mentransmisi ulang frame yang rusak tersebut.
- Pada metode Idle ARQ paket yang sampai pada penerima akan terurut, namun efisiensi salurannya rendah karena paket hanya bisa dikirimkan jika pengirim telah menerima feedback ACK dari penerima untuk paket sebelumnya. Metode ini cocok digunakan untuk saluran yang tingkat error-nya tinggi.



- Bila sinyal acknowledgment rusak pada waktu transmisi, kemudian sumber akan habis waktu dan mentransmisi ulang frame tersebut.
- Untuk mencegah hal ini, maka frame diberi label 0 atau 1 dan positive acknowledgment dengan bentuk ACK0 atau ACK1 : ACK0 mengakui menerima frame 1 dan mengindikasi bahwa receiver siap untuk frame 0. Sedangkan ACK1 mengakui menerima frame 0 dan mengindikasi bahwa receiver siap untuk frame 1.

Go Back N ARQ

- Pada metode Go Back N, pengirim akan mengirimkan beberapa paket secara berurutan tanpa menunggu paket pertama mendapat feedback dari penerima. Pengecekan paket yang sampai dilakukan oleh penerima. Segera setelah penerima berhasil mengidentifikasi paket tersebut penerima akan mengirimkan feeback kepada pengirim. Jika mendapat feedback ACK dari penerima, maka pengirim akan mengirimkan paket selanjutnya. Jika mendapat feedback NACK dari penerima, maka pengirim akan mengirimkan paket mulai dari yang salah tersebut. Karena mekanisme ini, keterurutan data di penerima akan tetap terjaga.
- Teknik Go-back-N ARQ yang terjadi dalam beberapa kejadian :
- Frame yang rusak. Ada 3 kasus :
  - A mentransmisi frame i. B mendeteksi suatu error dan telah menerima frame (i-1) secara sukses. B mengirim A NAKi, mengindikasi bahwa frame i ditolak. Ketika A menerima NAK ini, maka harus mentransmisi ulang frame i dan semua frame berikutnya yang sudah ditransmisi.
  - Frame i hilang dalam transmisi. A kemudian mengirim frame (i+1). B menerima frame (i+1) diluar permintaan, dan mengirim suatu NAKi.
  - Frame i hilang dalam transmisi dan A tidak segera mengirim frame -frame tambahan. B tidak menerima apapun dan mengembalikan baik ACK atau NAK. A akan kehabisan waktu dan mentransmisi ulang frame i.



- ACK rusak. Ada 2 kasus :
  - B menerima frame i dan mengirim ACK (i+1), yang hilang dalam transmisi. Karena ACK dikomulatif (contoh, ACK6 berarti semua frame sampai 5 diakui), hal ini mungkin karena A akan menerima sebuah ACK yang berikutnya untuk sebuah frame berikutnya yang akan melaksanakan tugas dari ACK yang hilang sebelum waktunya habis.
  - Jika waktu A habis, A mentransmisi ulang frame I dan semua frame -frame berikutnya.
- NAK rusak. Jika sebuah NAK hilang, A akan kehabisan waktu (time out) pada serangkaian frame dan mentransmisi ulang frame tersebut berikut frame-frame selanjutnya.

Selective Repeat

- Metode ini mirip dengan Go Back N, hanya saja pengiriman dan pengecekan paket tetap dilakukan walaupun ada paket yang error saat diterima. Jika pada Go Back N pengiriman ulang paket dilakukan mulai dari paket yang salah, pada Selective Repeat pengiriman ulang paket hanya dilakukan untuk paket yang salah saja. Karena itu, paket bisa saja diterima tidak berurutan. Efisiensi saluran lebih tinggi daripada idle RQ maupun Go Back N.