Алдаа илрүүлэлт ба засварлалт

Error Detection and Correction

Лекц 7

Алдаа илрүүлэлт ба засварлалт

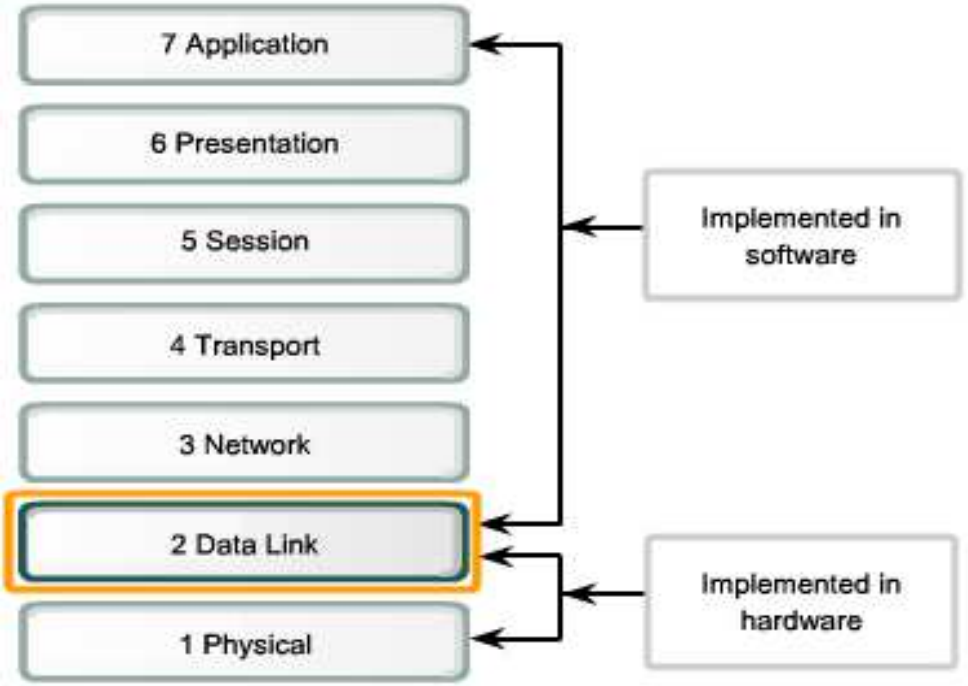
Агуулга:

- Алдаа илрүүлэлт ба засварлалт
- Алдааны төрөл
- Алдаа илрүүлэх аргууд:
  - Parity check
  - Two dimensional parity check
  - Checksum
  - Cyclic redundancy check
- Алдаа засварлах арга (Hamming code)

Data Link Layer

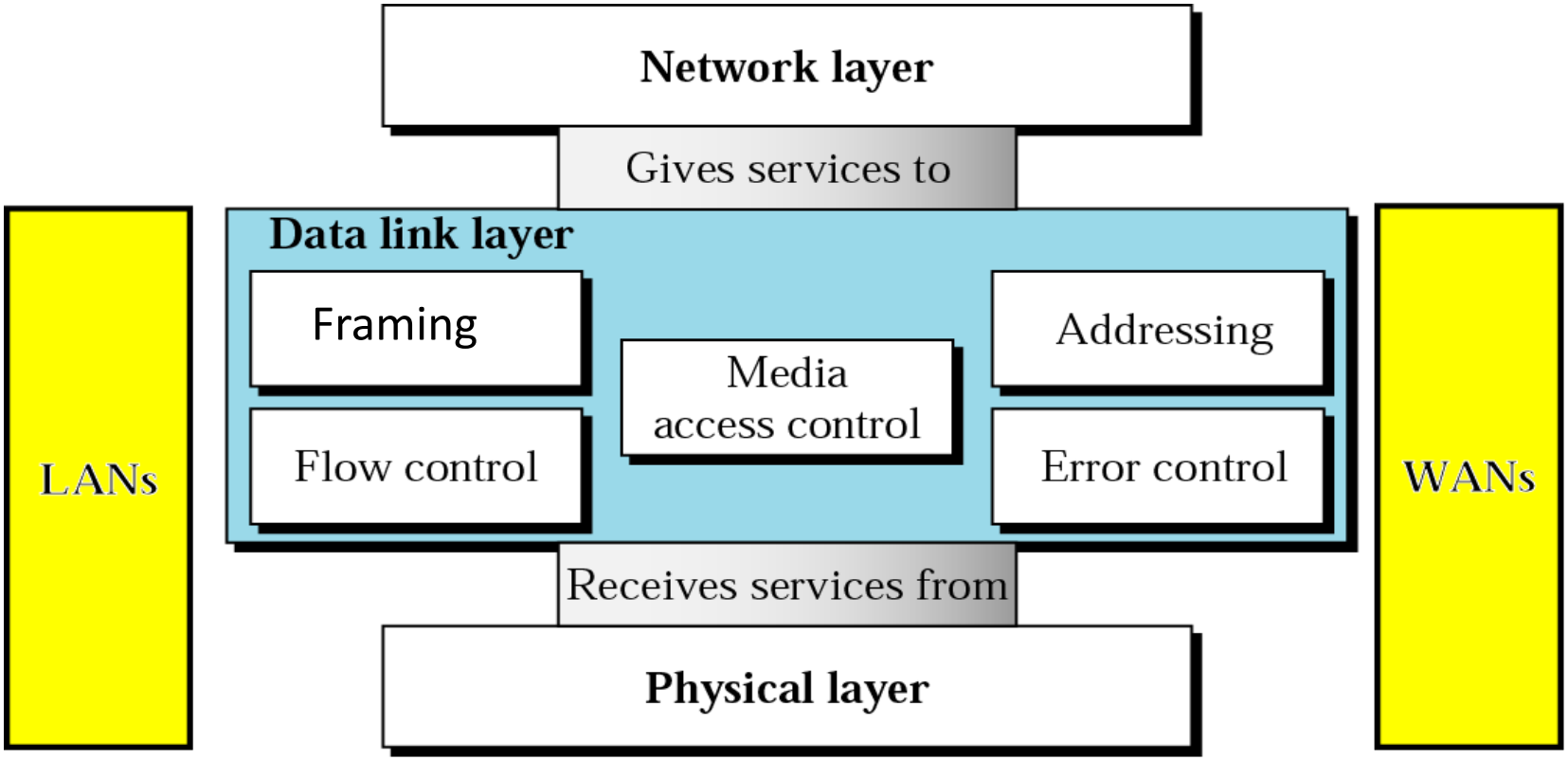
Өгөгдөл орчинд солилцох үйлчилгээг үзүүлнэ

- *PDU - Frame*
- *Hop-to-Hop*



PC NIC

Data Link Layer



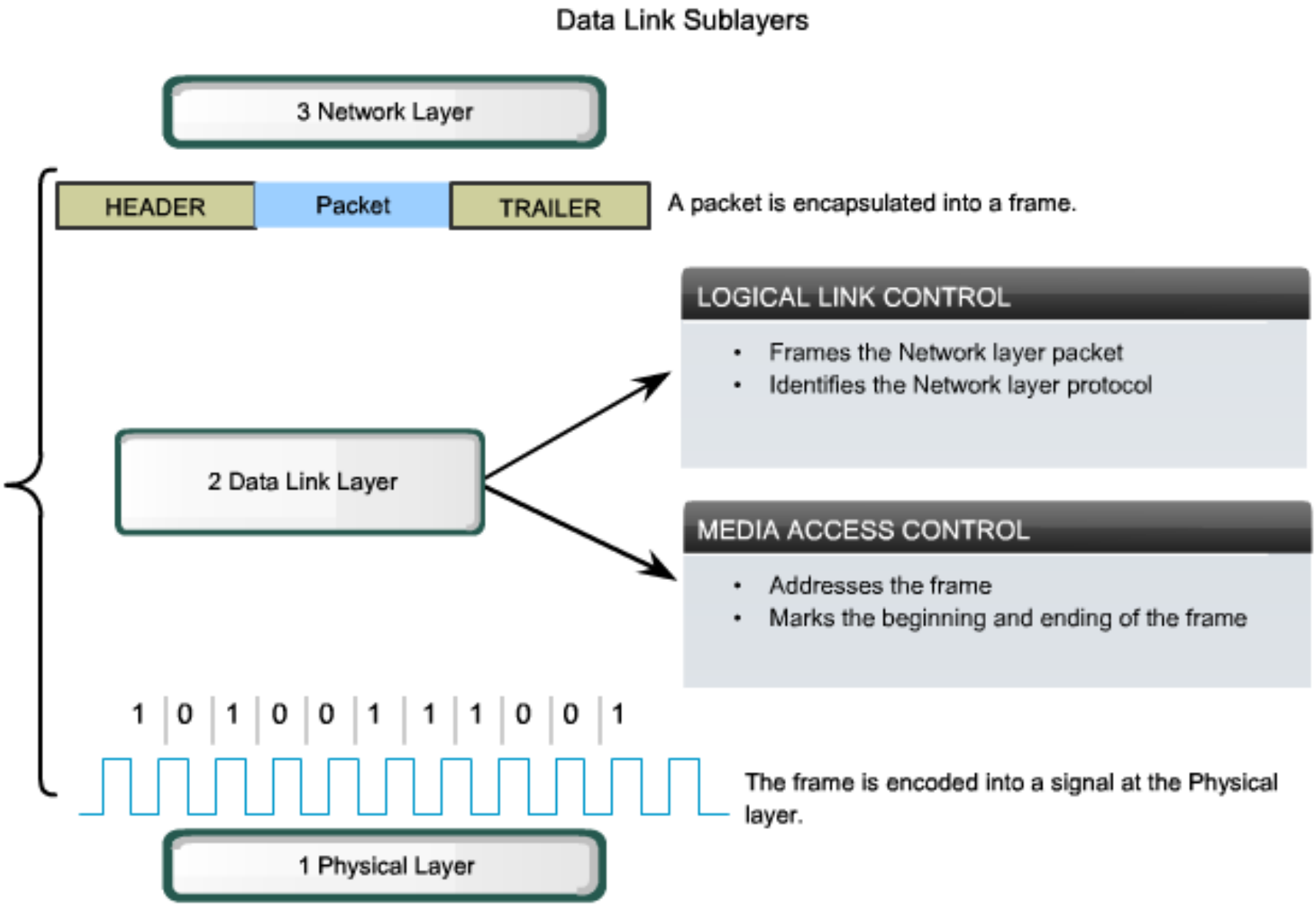
Data Link layer

Энэ түвшинд физик түвшнээс ирсэн битийн цувааг алдаагүй болгоод дээд түвшин (Network)-д дамжуулна. Өөрөөр хэлбэл ирсэн битүүдийг багцалж, физик хаягаар нь дамжуулахын тулд алдааг нь илрүүлэх, засварлах үйлчилгээг үзүүлнэ.

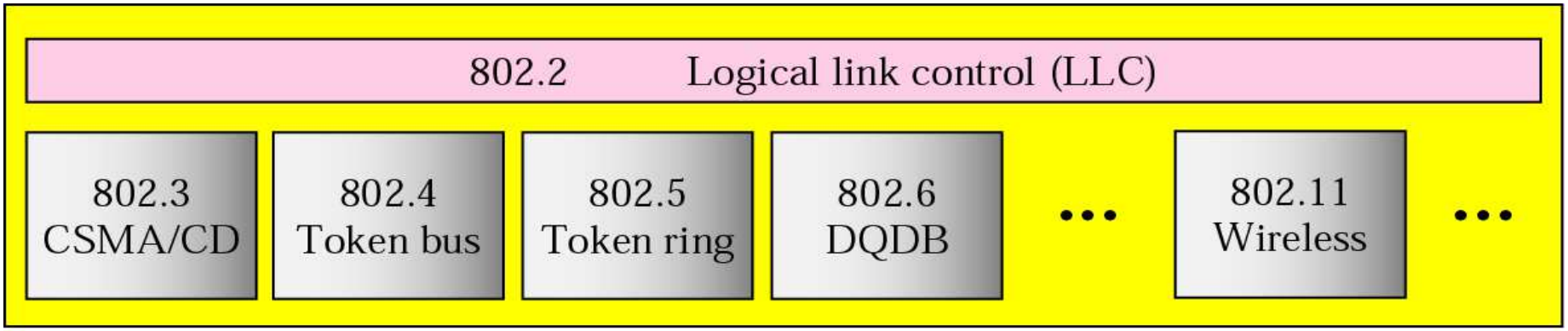
Data Link түвшний үдсэн үүрэг:

- ***Fraiming*** *Data Link layer нь битийн цувааг удирдах боломж бүхий байдлаар хувааж багцлана. Үүнийг фрейм гэнэ.*
- ***Flow Control*** *Илгээгч болон хүлээн авагчийн өгөгдөл боловсруулах хэмжээ нь ялгаатай байвал энэ түвшинд удирдана.*
- ***Addressing*** *Фрейм нь сүлжээгээр дамжихын тулд илгээгч болон хүлээн авагчийн хаягийг агуулсан header мэдээлэл агуулсан байна. Хэрэв фрейм нь илгээгчийн сүлжээнээс гадуур сүлжээн дэх төхөөрөмжид хүргэгдэх бол хүлээн авагчийн хаяг нь дараагийн төхөөрөмжийн хаяг байна.*
- ***Error Control*** *Data Link Layer нь алдагдсан болон эвдэрсэн фреймийг илрүүлэх болон дахин илгээнэ. Алдааны мэдээлэл нь фреймийн төгсгөлд буюу trailer хэсэгт нэмэгдэнэ.*
- ***Media Access Control*** *Сүлжээнд 2 ба түүнээс дээш төхөөрөмж холбогдсон байвал энэ түвшинд яг аль төхөөрөмжид хүргэх вэ гэдгийг шийдэж өгнө.*

LLC and MAC sublayers



IEEE standards for LANs



Project 802

Яагаад алдаа илрүүлэх шаардлагатай вэ?

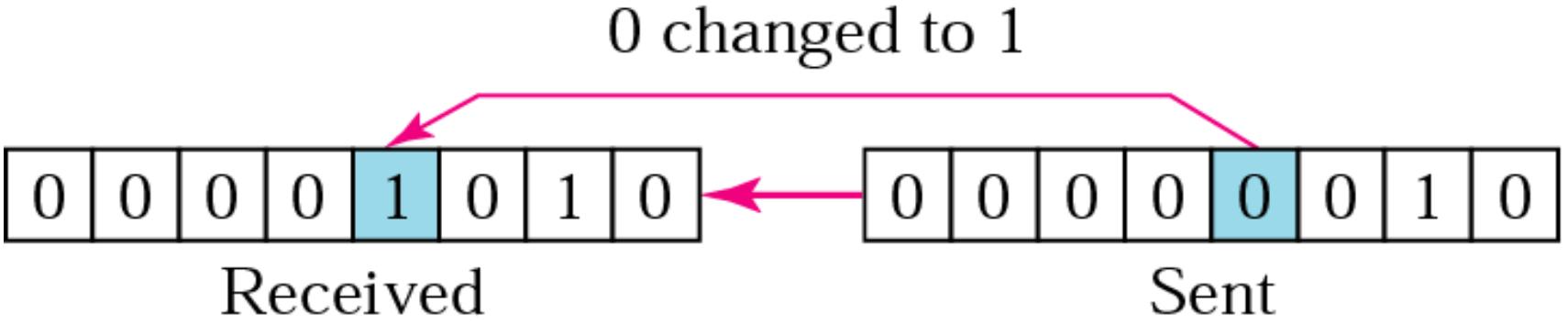
- *Унтралт, гажуудал, шуугиан, гадны соронзон орны нөлөөлөл зэрэг нь өгөгдөл дамжуулалтанд сөргөөр нөлөөлөх ба зарим битүүдийг гэмтээдэг.*

- *Фреймийн хэмжээ их бол битийн алдах магадлал өндөр, бага бол алдаагүй дамжих магадлал өндөр байдаг.*

- *Өгөгдөл дамжигдах үед гэмтэж алдаа үүснэ. Өгөгдлийн холбооны найдваржилт бол алдааг илрүүлэх, засах явдал юм.*

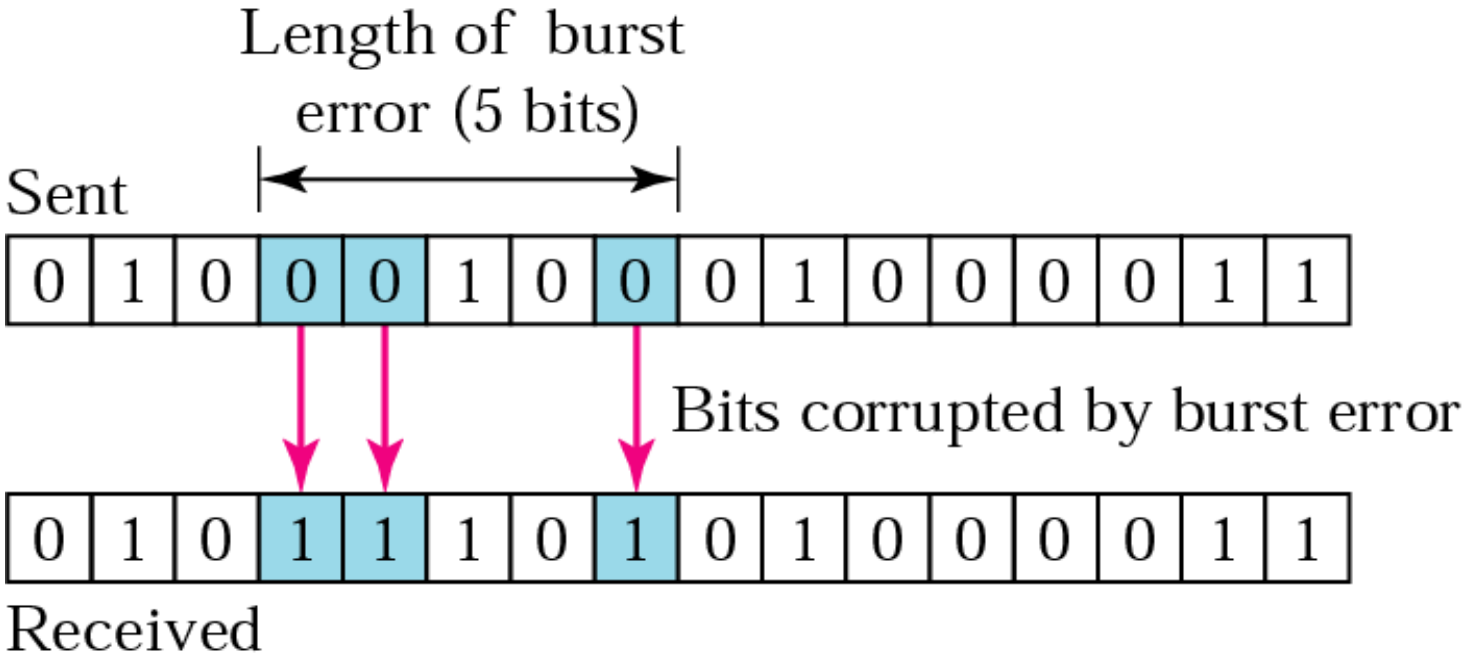
Алдааны төрөл

- *Нэг битийн алдаа: Өгөгдлийн зөвхөн нэг бит солигдож дамжигдана. Ихэвчлэн параллель дамжуулалтанд тохиолдоно.*



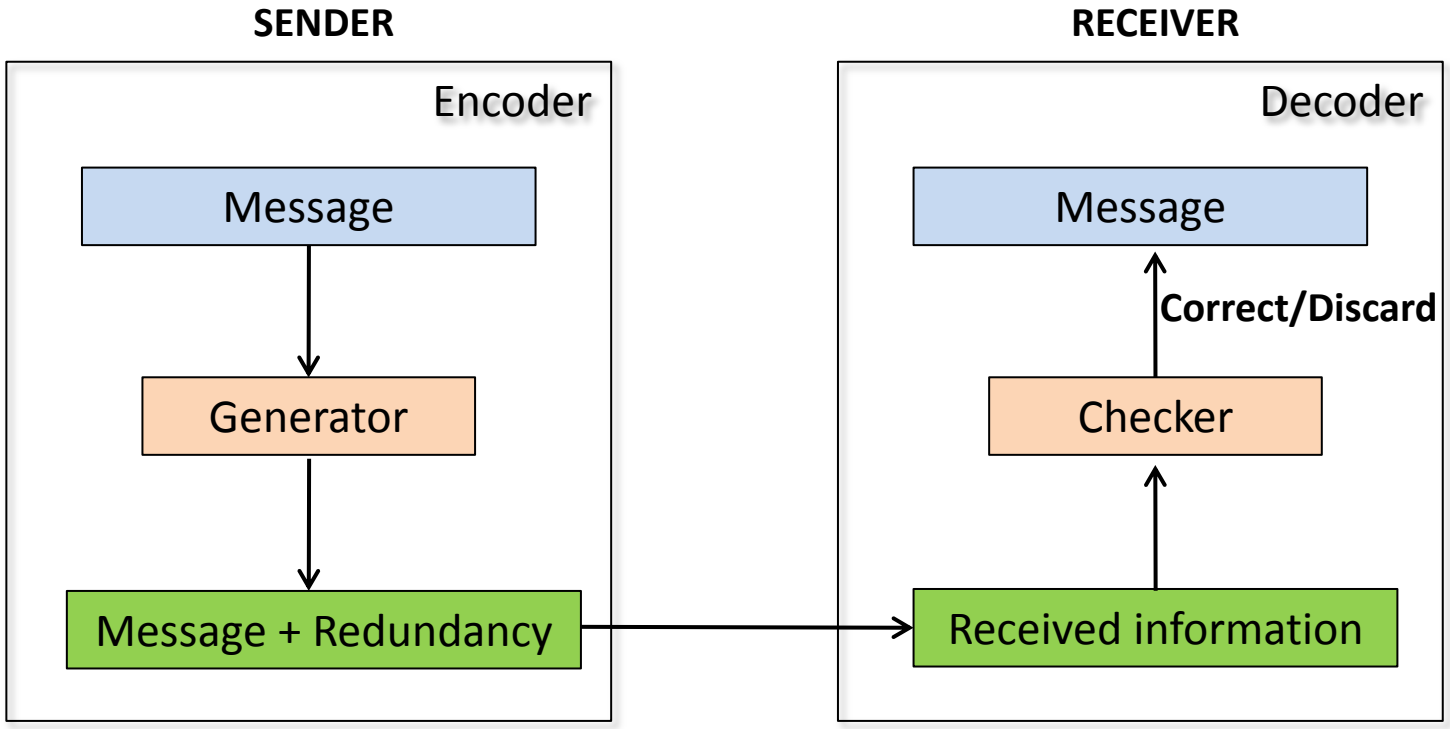
Алдааны төрөл

- *Багц битийн алдаа : Өгөгдлийн хоёр ба түүнээс олон бит алдагдаж дамжигдсан алдааг хэлнэ. Сериал дамжуулалтанд их тохиолдоно.*



Өгөгдөл ба алдааны илрүүлэх кодын хамаарал

- ***Redundancy – Нэмэлт бит***
  Өгөгдлийн холбоонд алдааг илрүүлэхдээ нэмэлт бит ашиглах ба хүлээн авагч төхөөрөмж дээр шалгаж алдааг илрүүлнэ.

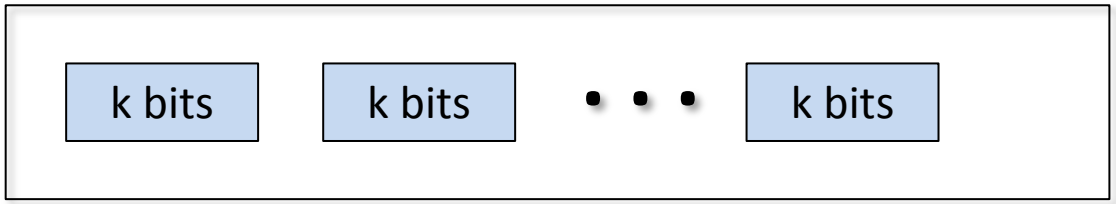


Алдааны илрүүлэх кодын схем

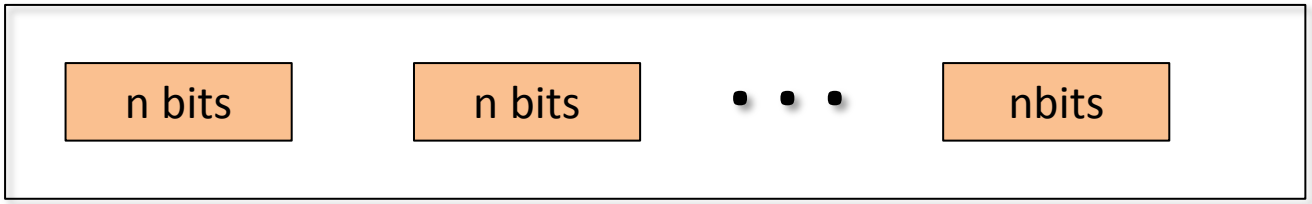
- ***Block Coding - Блок кодлол***

  ‘k’ bits – datawords

  ‘n = k + r’ bits – codewords



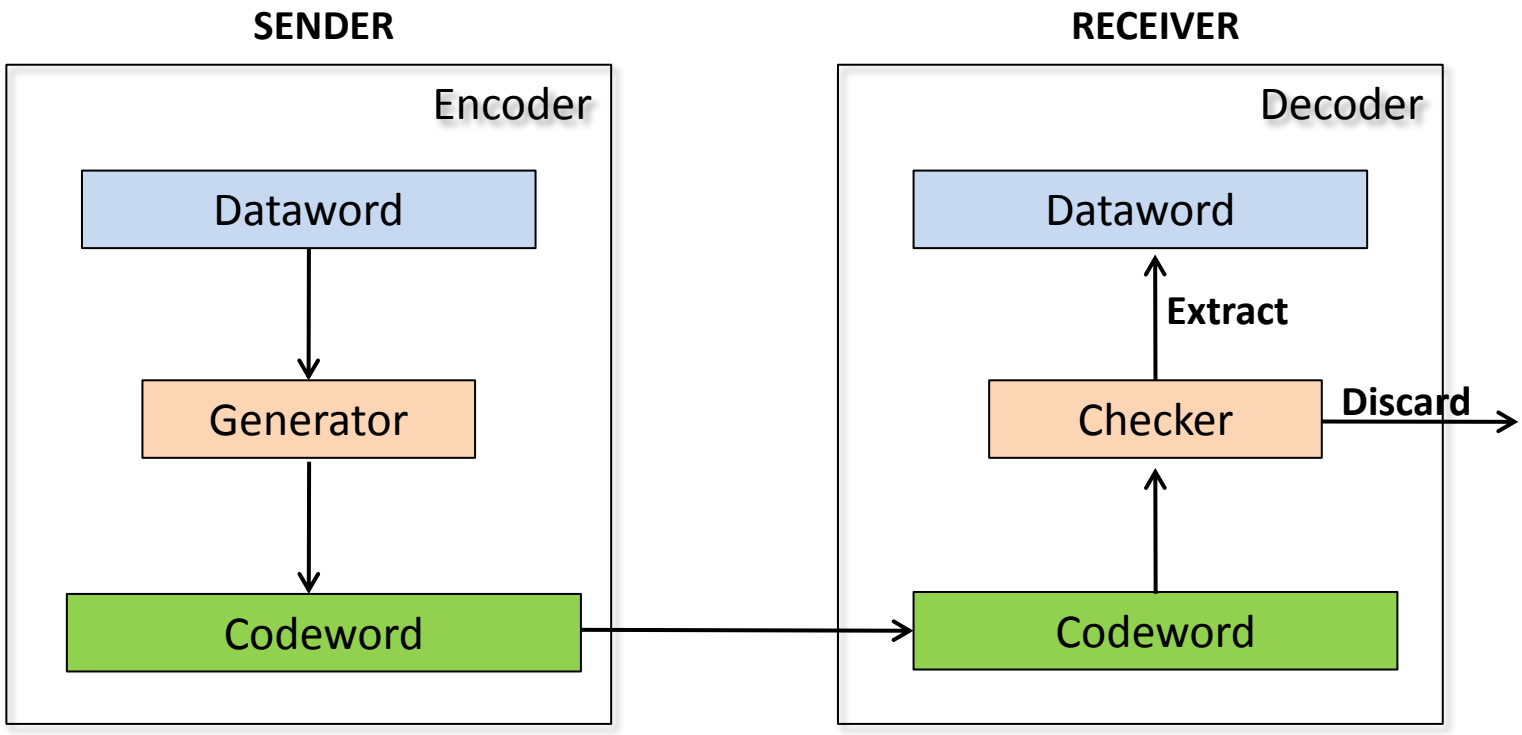
2^k Datawords



2^n Codewords (2^k хувилбар нь ашиглагдана)

Алдаа илрүүлэлт

- *Хүлээн авагч зөвшөөрөгдөх codeword-үүдийн жагсаалттай байна.*
- *Хүлээн авсан мессежээ зөвшөөрөгдөх codeword-ийн жагсаалттай тулгаж шалгана.*



Алдаа илрүүлэлт

Жишээ:

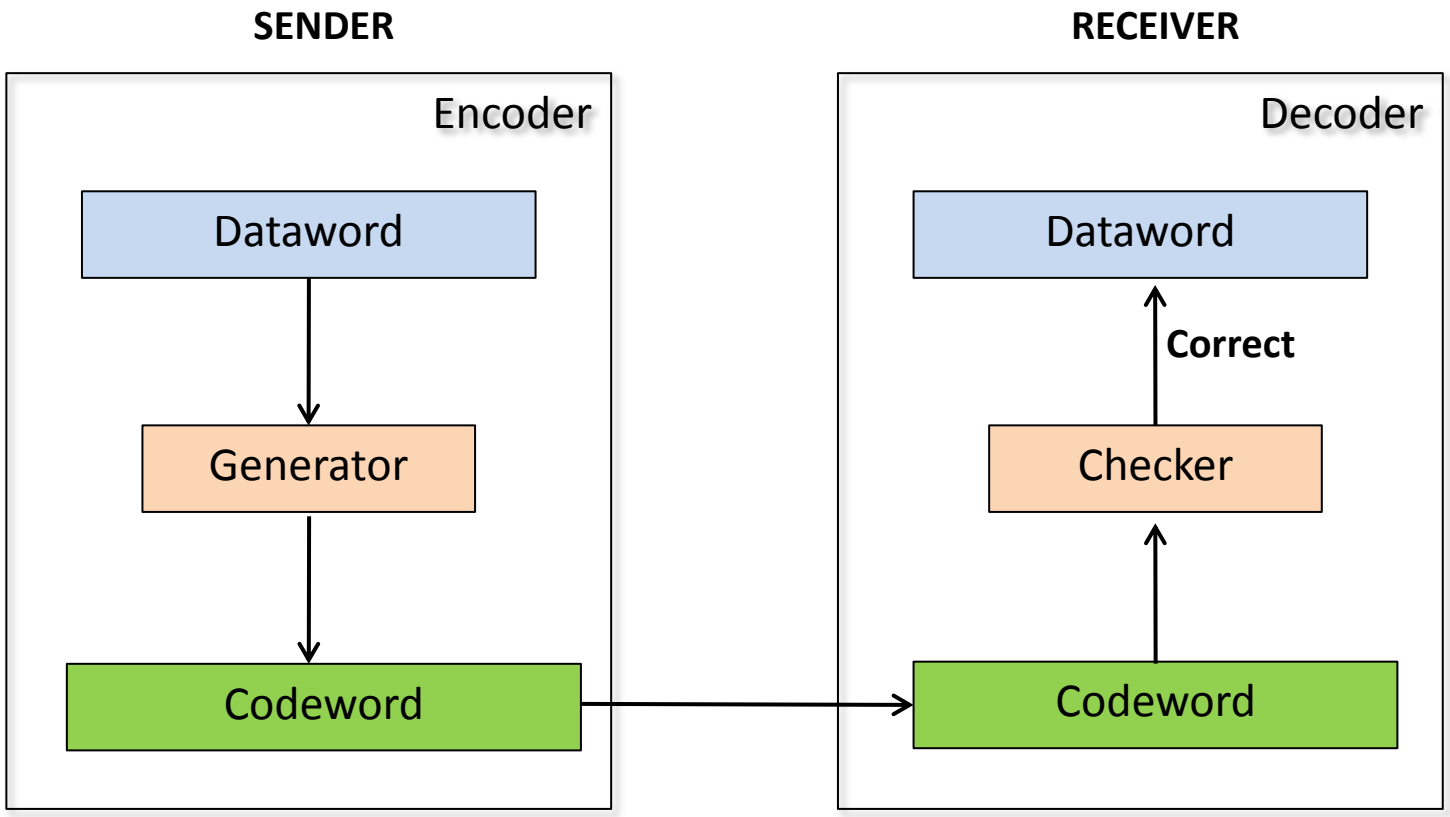
Datawords	*Codewords*
00	*000*
01	*011*
10	*101*
11	*110*

1. *Илгээгч 01 гэсэн өгөгдөл илгээсэн гэж үзье*
2. *Хүлээн авагч 011 –ыг хүлээж авбал алдаагүй.*
3. *Хэрэв codeword = 111 –ыг хүлээж авбал алдаатай.*
4. *Хэрэв codeword = 000 – ыг хүлээж авбал ...*

Алдаа илрүүлэх код нь зөвхөн блок битийн дүрслэл алдагдсан тохиолдолд илрүүлнэ.

Алдаа засалт

- *Хүлээн авагч хүлээн авсан мессежээ илгээгдсэн codeword мөн эсэхийг шалгана.*
- *Алдаа илрүүлэлтээс илүү олон нэмэлт битийг ашиглана.*



Алдаа илрүүлэх арга

- ***Simple Parity check***
  (Нэг паритигаар шалгах арга)

- ***Two dimensional Parity check***
  (Хоёр хэмжээст паритигаар шалгах арга)

- ***Checksum***
  (Нийлбэрээр шалгах арга)

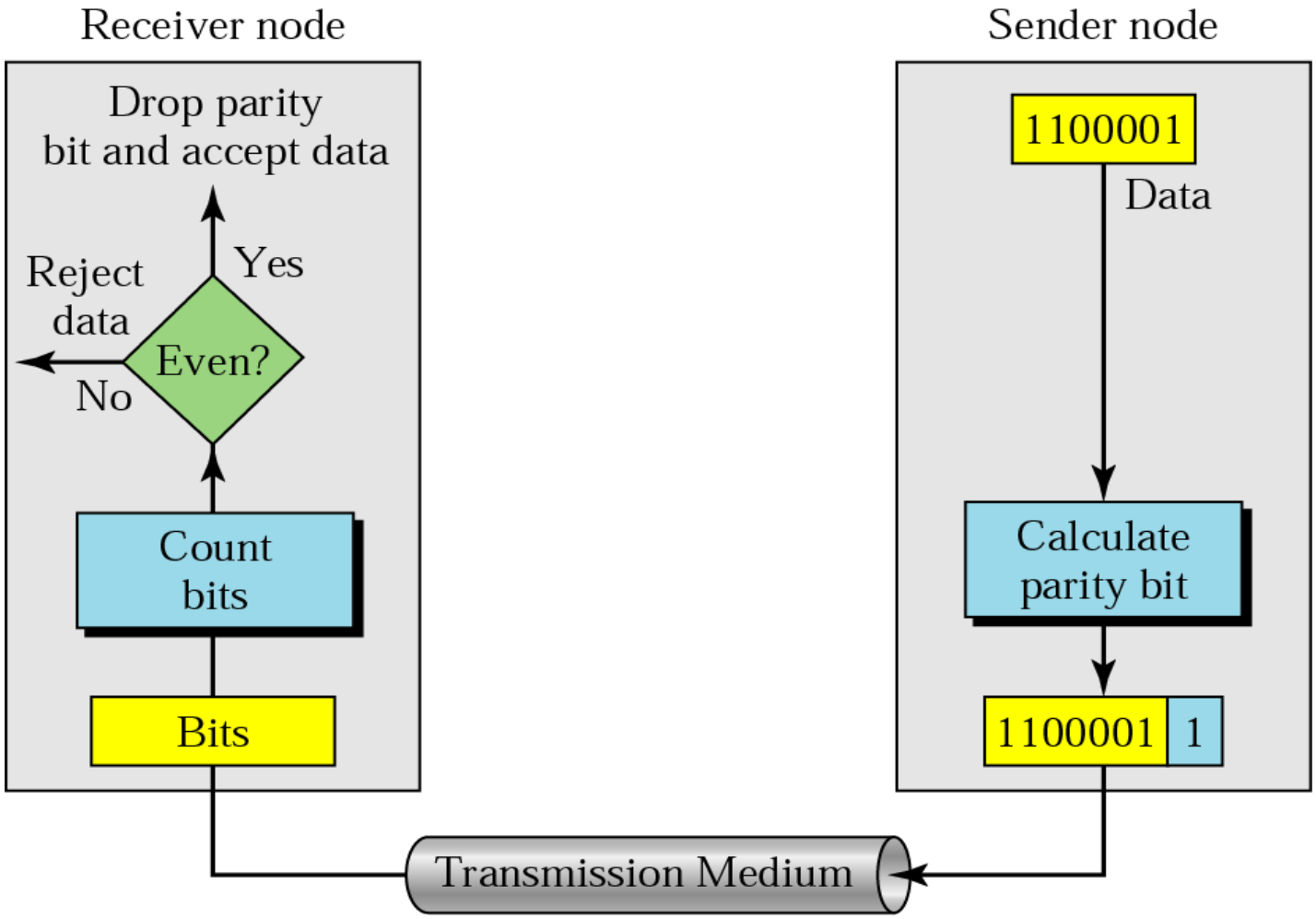
- ***Cyclic redundancy check***
  (Цикл битүүтээр шалгах арга)

Нэг паритигаар шалгах арга (VRC – Vertical Redundancy Check)

- *Энэ тохиолдолд парити бит нь өгөгдлийн битүүдээс 1-үүдийг тоолно. (тэгш ба сондгой)*
  - *Синхрон дамжууллын горим – Тэгш парити*
  - *Асинхрон дамжууллын горим – Сондгой парити*

- *Парити бит нь өгөгдлийн төгсгөлд нэмэгдэнэ.*

Нэг паритигаар шалгах арга



Жишээ Илгээгч *world* гэдэг үг илгээхэд таван тэмдэгтийн ASCII код нь:

w	*o*	*r*	*l*	*d*
1110111	1101111	1110010	1101100	1100100

Тэгш паритигаар илгээсэн гэвэл:

11101110 11011110 11100100 11011000 11001001

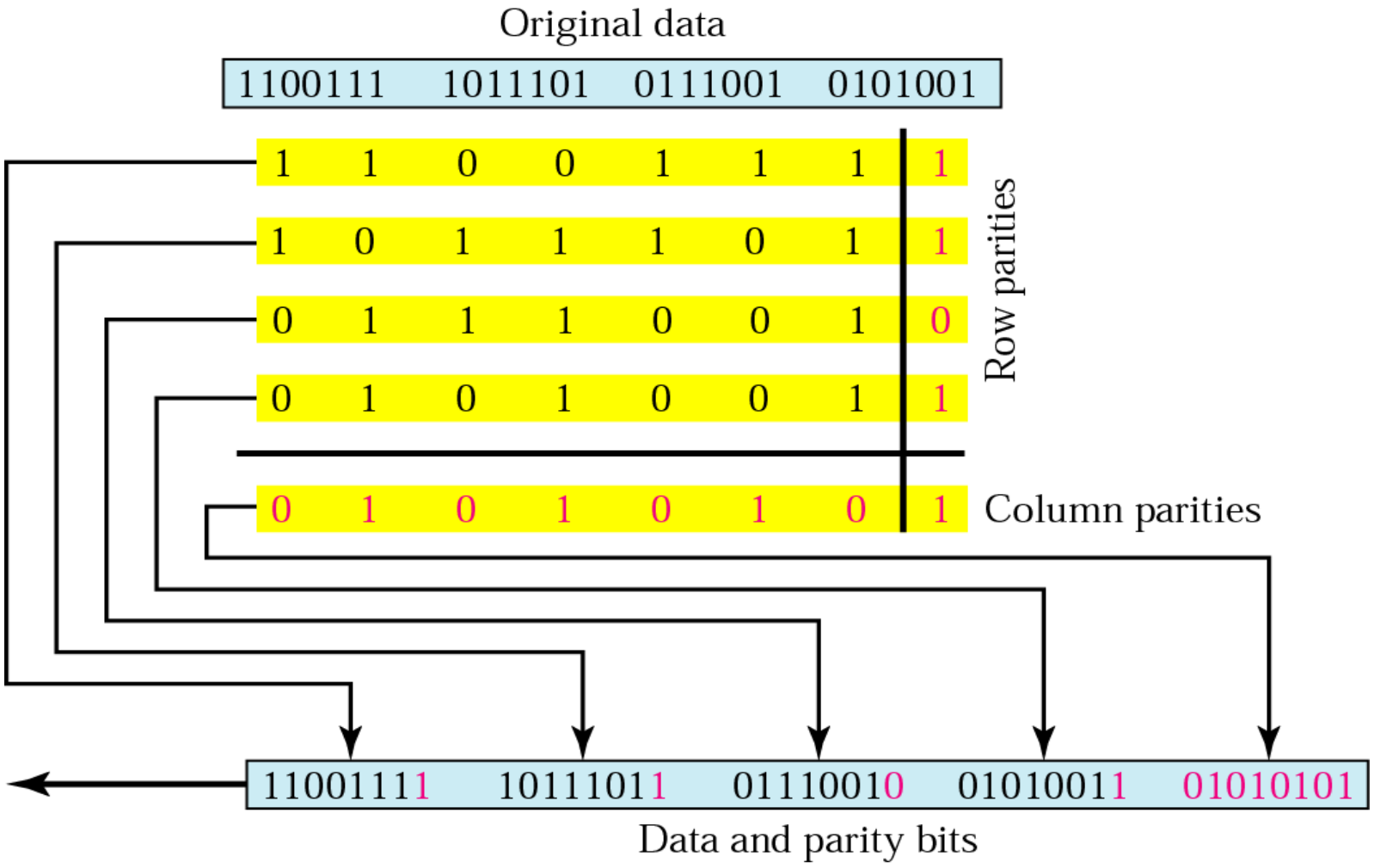
Нэг паритигаар шалгах арга (VRC – Vertical Redundancy Check)

- *Бүх нэг-бит алдаа илрүүлэгдэнэ.*
- *Хэрэв 1-үүдийн тоо тэгш/сондгой тоологдвл багц битийн алдааг мөн илрүүлнэ.*
- *Нэгээс олон бит солигдоод ирсэн үед тийм ч сайн арга биш, хэрэв битүүд солигдоод ирсэн үед тэгш/сондгой нэгүүдийн нийлбэр өөрчлөгдөөгүй тохиолдолд илрүүлж чадахгүй.*

Хоёр хэмжээст паритигаар шалгах арга (LRC – Longitudinal Redundancy Check)

- *Өгөгдлийн битүүдийг блок байдлаар зохион байгуулж баганы паритиг тооцох арга юм.*
- *Өгөгдлийн битүүдийг мөрүүдэд хуваах ба парити бит нь бүх мөр ба баганад бодогдож өгөгдөлтэй хамт дамжигдана.*
- *Хүлээн авагч нь паритигаар нь харьцуулж өгөгдлийг тооцож гаргана.*

Хоёр хэмжээст паритигаар шалгах арга (LRC – Longitudinal Redundancy Check)



Хоёр хэмжээст паритигаар шалгах арга (LRC – Longitudinal Redundancy Check)

- *Нэмэлт битийн тоо олон болох тусам алдаа илрүүлэлт нь сайжирна.*
- *Энэ арга нь олон багц алдааг илрүүлэх боловч бүгдийг нь илрүүлж чадахгүй.*

Жишээ:

Илгээгч төхөөрөмжөөс дараах блокыг илгээхэд:

10101001 00111001 11011101 11100111 10101010

8 битийн урттайгаар шуугианд өртөж зарим битүүд гэмтэж дараах байдлаар хүлээн авагдсан гэвэл:

10100011 10001001 11011101 11100111 10101010

Хүлээн авагч парити шалгах аргаар шалгаад тэгш паритигийн дүрэм биелэхгүй байгаа тул блокыг хаяж дахин дамжуулахыг хүснэ.

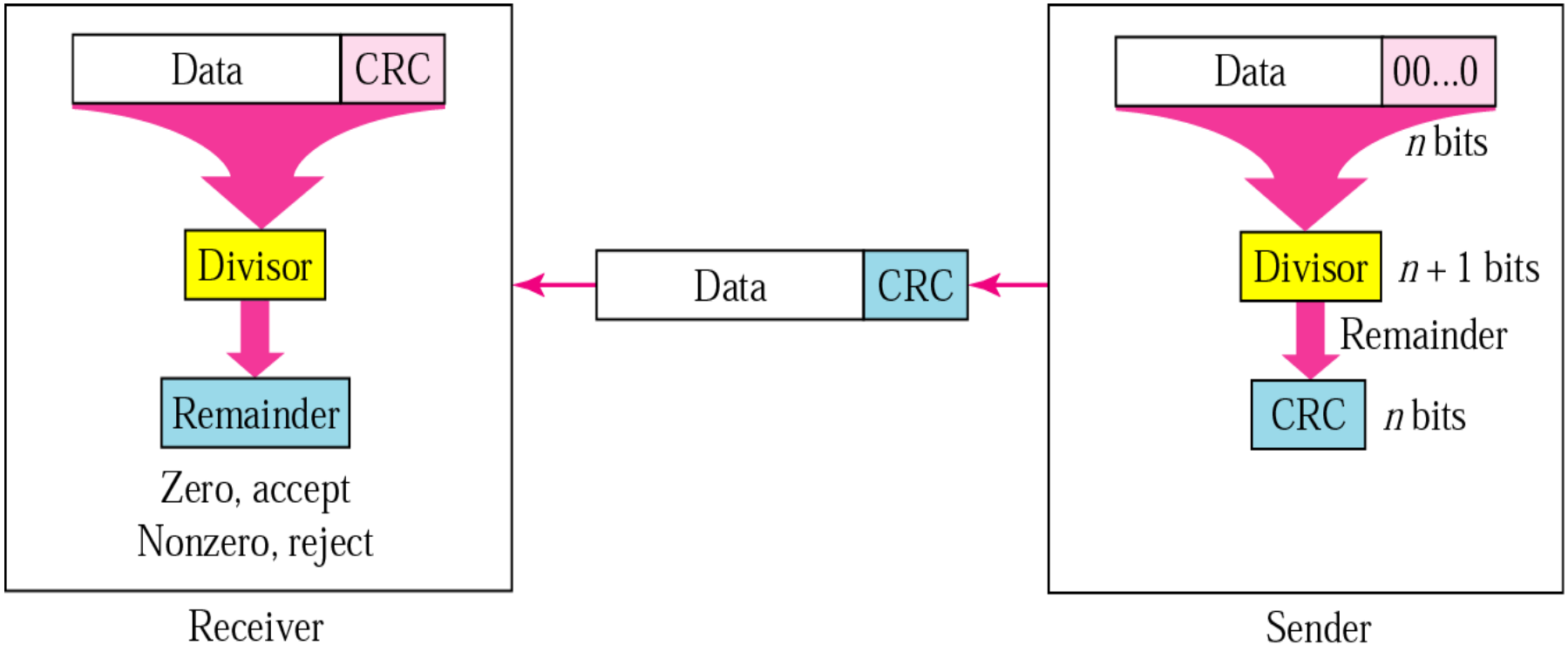
10100011 10001001 11011101 11100111 10101010

Цикл битүүдээр шалгах арга (CRC – Cyclic Redundancy Check)

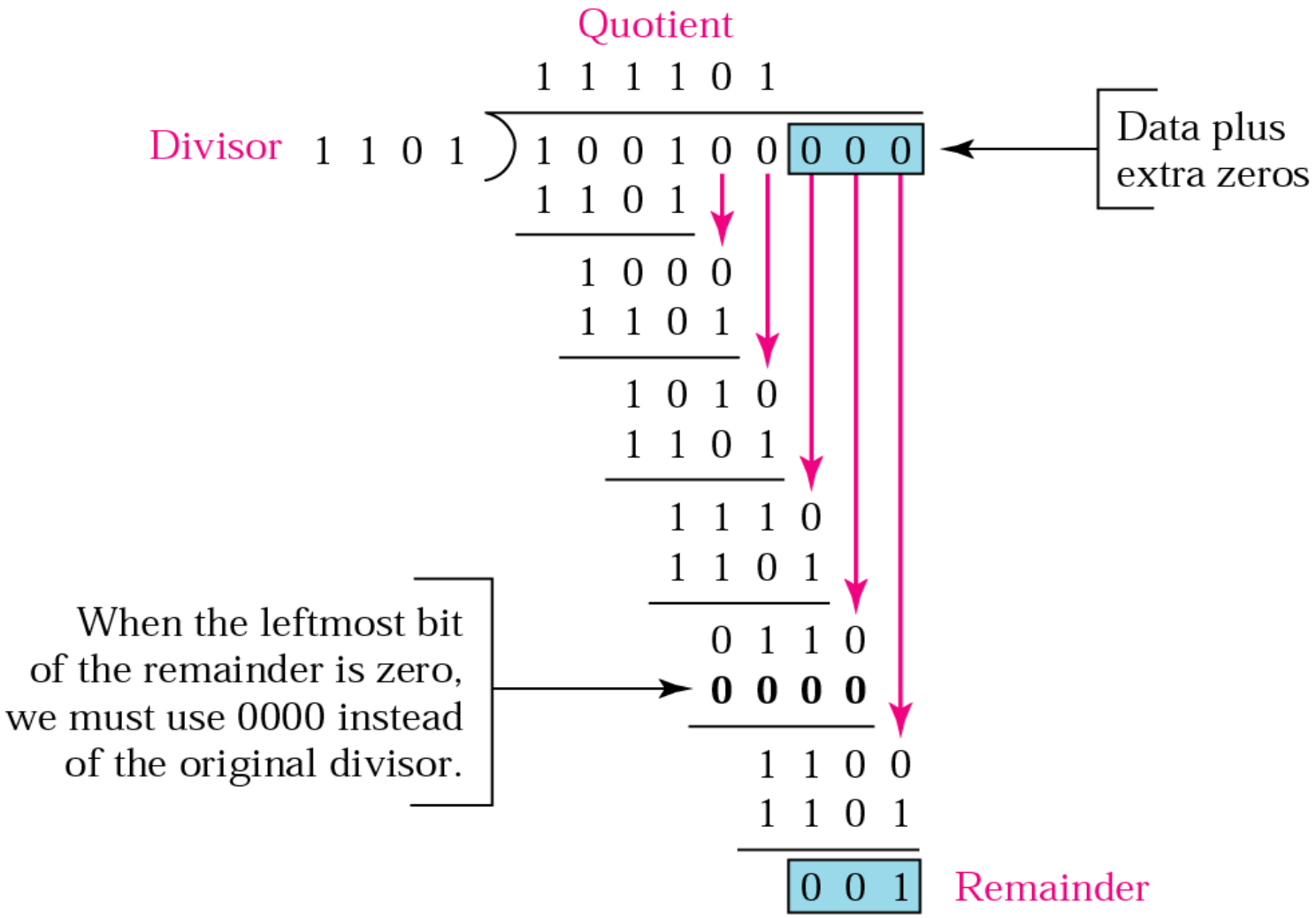
- *Хамгийн өргөн ашиглагддаг алдаа илрүүлэх аргын нэг.*

- *Үндсэн процесс:*
  - *Өгөгдлийг илэрхийлсэн m-бит дарааллыг илгээгч төхөөрөмжөөр үүсгэгдсэн Frame check sequence (FCS) гэх n-битийн дарааалалтай хамт илгээнэ.*
  - *Хүлээн авагч нь фреймийг хуваагч загвар битэд үлдэгдэлгүйгээр хуваавал алдаагүй гэж үзнэ.*

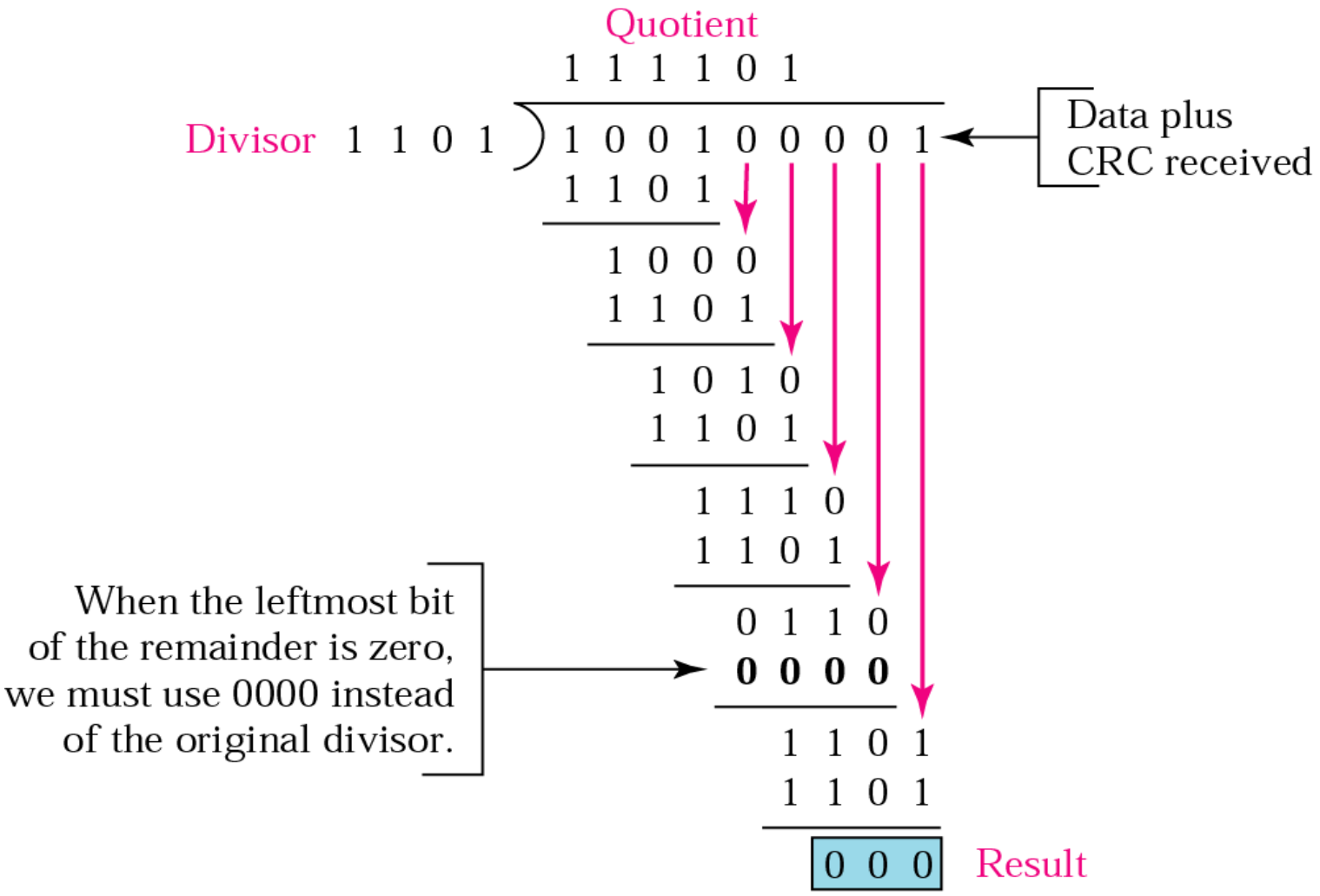
CRC генератор ба шалгагч



CRC генераторын хоёртын хуваалт

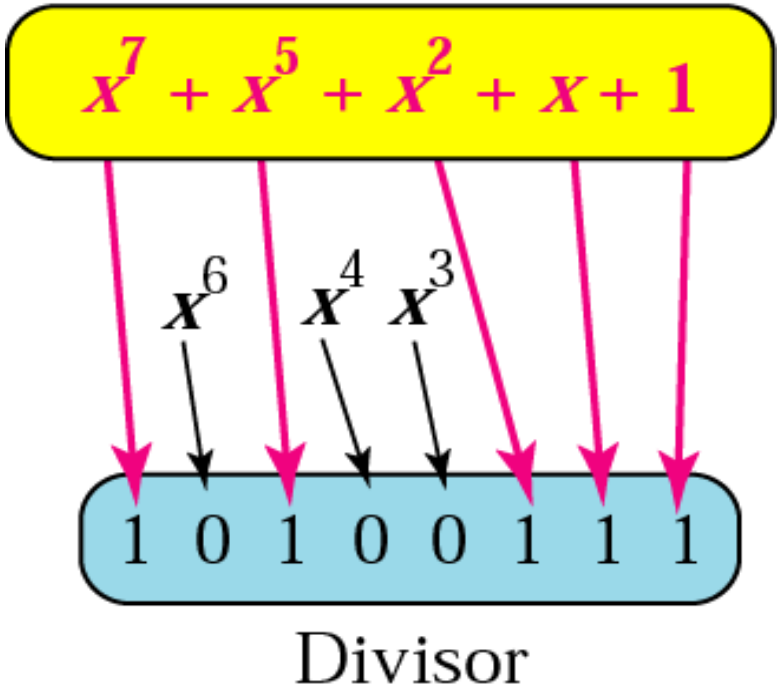


CRC шалгагч


Quotient
1 1 1 1 0 1
Divisor 1 1 0 1
1 0 0 1 0 0 0 0 1
Data plus
CRC received
1 1 0 1
1 0 0 0
1 1 0 1
1 0 1 0
1 1 0 1
1 1 1 0
1 1 0 1
0 1 1 0
0 0 0 0
1 1 0 1
1 1 0 1
0 0 0
Result
When the leftmost bit
of the remainder is zero,
we must use 0000 instead
of the original divisor.

Хуваагч загварыг олон хувьсагчаар илэрхийлэх

$$x^7 + x^5 + x^2 + x + 1$$



Стандарт олон хувьсагч илэрхийлэл

Name	Polynomial	Application
CRC-8	$x^8 + x^2 + x + 1$	**ATM header**
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	**ATM AAL**
ITU-16	$x^{16} + x^{12} + x^5 + 1$	**HDLC**
ITU-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	**LANs**

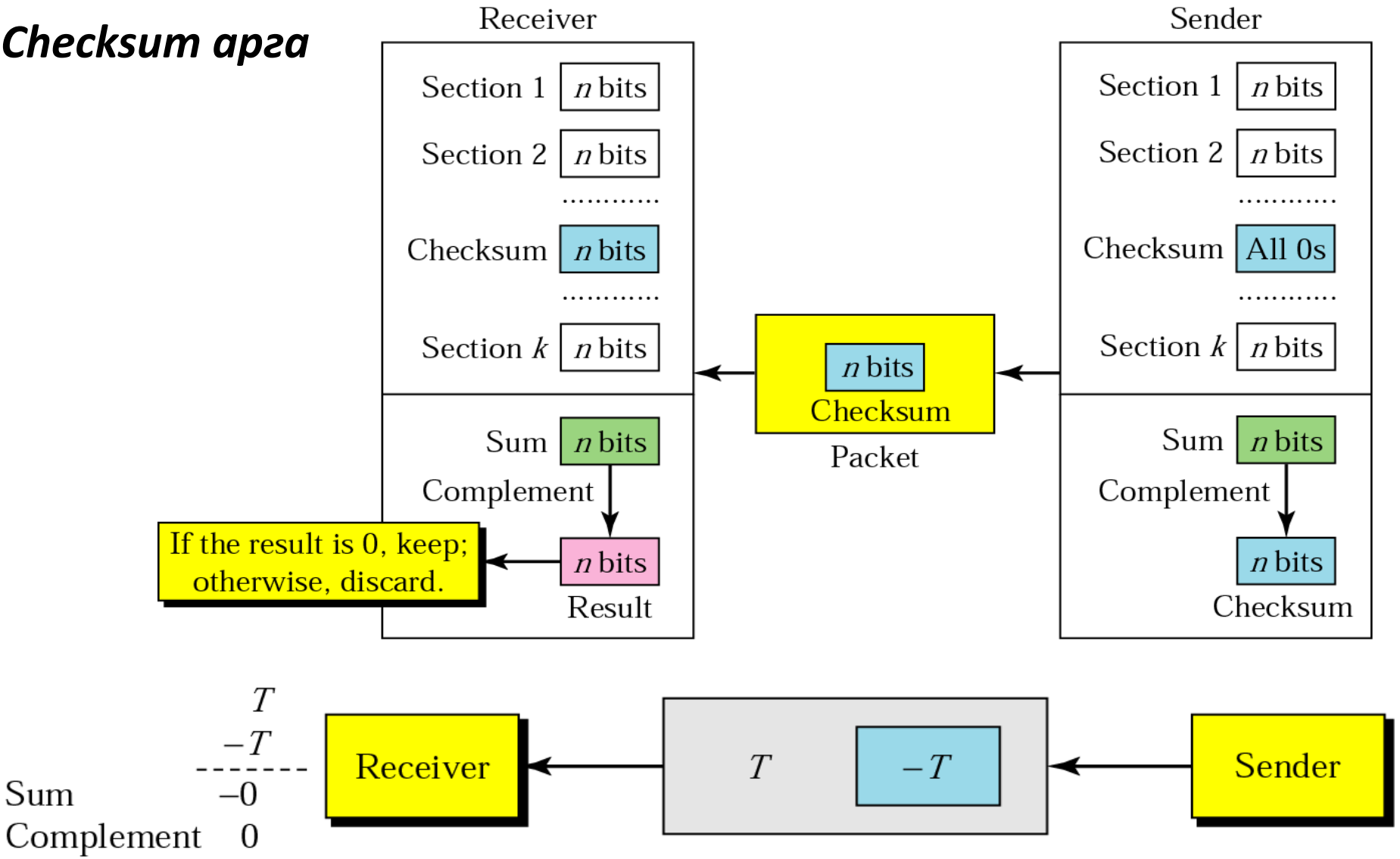
Цикл битүүдээр шалгах арга (CRC – Cyclic Redundancy Check)

- *CRC арга нь бүх нэг битийн алдааг илрүүлнэ.*
- *CRC арга нь бүх хоёр битийн алдааг илрүүлнэ.*
- *CRC арга нь дурын сондгой битийн алдааг илрүүлнэ.*
- *CRC арга нь олон хувьсагчийн зэрэг хүртэлх бүх багц битийн алдааг илрүүлнэ.*
- *Олон хувьсагчийн зэргээс илүү багц битийн алдааг бараг бүгдийг нь илрүүлнэ.*

Нийлбэрийг шалгах арга (Checksum)

- *Өгөгдлийг сегментүүдэд хуваана.*
- *Бүх сегментүүдийг нэмж нэг бүхэл болгоно.*
- *Нийлбэрийн аргаар алдааг шалгана.*

Checksum арға



T = Data segment sum

(-T) = Checksum

Нийлбэрийг шалгах арга (Checksum)

Илгээгч талд хийгдэх үйлдэл:

- *Өгөгдлийг n битийн урттайгаар k сегментэд хуваана*
- *Бүх сегментүүдийг нэмж нэг бүхэл болгоно.*
- *Сегментүүдийн нийлбэрийн 1-ийн гүйцээлтээр checksum-ийг тодорхойлно.*
- *Checksum-ийг өгөгдлийн хамт илгээнэ.*

Нийлбэрийг шалгах арга (Checksum)

Хүлээн авагч талд хийгдэх үйлдэл:

- *Өгөгдлийг n битийн урттайгаар k сегментэд хуваана*
- *Бүх сегментүүдийг нэмж нэг бүхэл болгоно.*
- *Нийлбэрийн 1-ийн гүйцээлтийг тооцоолно.*
- *Үр дүн 0 бол алдаагүй, бусад тохиолдолд алдаатай гэж үзнэ.*

Өгөгдсөн 16 бит өгөгдлийг 8 битийн урттайгаар сегментүүдэд хувааж илгээвэл:

10101001 00111001

Өгөгдлийн нийлбэрийг олбол:

```
                10101001
                00111001
                ----------
Sum             11100010
Checksum        00011101
```

Илгээгдэх өгөгдөл 10101001 00111001 00011101

Илгээгдсэн өгөгдөл хүлээн авагчид алдаагүйгээр очсон гэвэл:

10101001 00111001 00011101

Хүлээн авагч 8 битийн урттайгаар 3 сегментэд хуваагаад нэмэхэд нийлбэр нь бүгд 1 гарна. Инверс нь бүгд 0 гарах тул алдаагүй ирсэн гэсэн үг.

	10101001
	00111001
	00011101
Sum	11111111
Complement	00000000 OK. ☺

5 битийн урттай 4 бит алдагдсан гэж үзвэл:

10101111 11111001 00011101

Хүлээн авагчид 3 сегментэд хуваагаад нэмнэ.

	10101111
	11111001
	00011101
Partial Sum	1 11000101
Carry	1
Sum	11000110
Complement	00111001 өгөгдөл алдаатай ирсэн ☹.

Алдаа засалт

Алдаа засалтын үндсэн хоёр арга:

- *Backward Error Correction: Фрейм алдаатай хүлээн авагдсан тохиолдолд илгээгч дахин дамжуулна. Ө.х* ***Retransmission*** *буюу үүсгүүр алдаатай өгөгдлийг дахин дамжуулах замаар алдааг засварлана. Үүнийг* ***Automatic Repeat Request (ARQ)*** *арга гэж нэрлэнэ.*
- *Forward Error Correction : Хүлээн авуур Алдаа шалгалтын нэмэлт битүүдийг ашиглан өгөгдлийн нэгжийн алдааг автоматаар засварлаж зөв хүлээн авна. Энэ аргыг Хаммингийн кодын арга гэнэ.*

Алдаа засалтын арга – Хаммингийн код

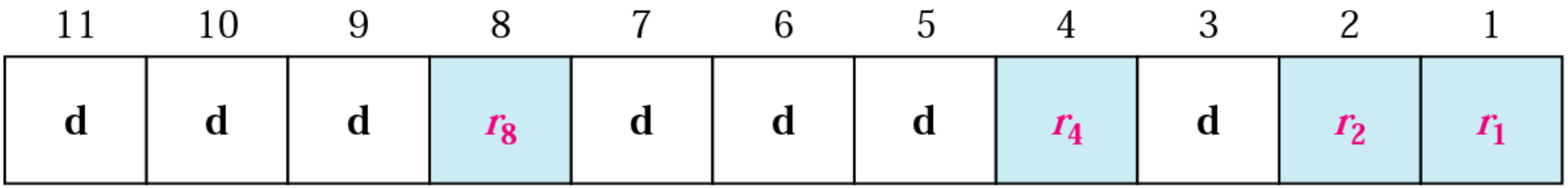
- Мэдээг илэрхийлсэн m багц бит нь k парити битийн хамт (m+k) хэлбэрт шилжинэ.
- Битүүдийн байрлал (m+k) тоогоор дугаарлагдана.
- k ширхэг парити битүүд нь өгөгдлийн алдааг илрүүлэх 1, 2 …, 2^k байрлалд байрлана.
- k ширхэг парити бит нь сонгогдсон тоогоор шалгах биттэй байна.
- Хүлээн авагч нь парити битээр шалгана.
- Хэрэв бит алдагдаж дамжигдвал k парити битийн аравтын тооллын утга нь алдаатай битийн байрлалыг илэрхийлнэ.
- Өгөгдлийн битийн тооноос хамааран нэмэлт битийн тоо : $2^k > m+k+1$

k – нэмэлт бит, m – өгөгдлийн бит

Өгөгдөл ба нэмэлт бит

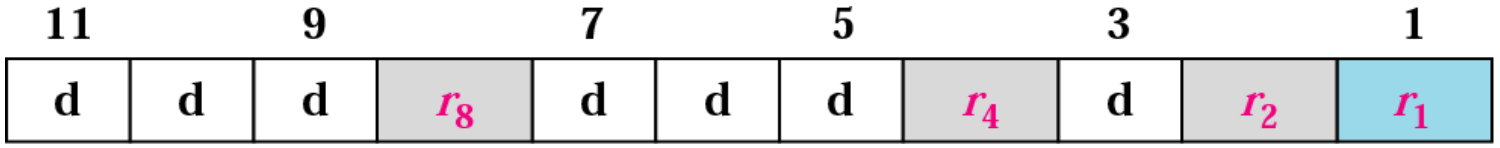
Number of data bits m	Number of redundancy bits r	Total bits m + r
1	2	3
2	3	5
3	3	6
4	3	7
5	4	9
6	4	10
7	4	11

Хаммингийн кодлолын нэмэлт битүүдийн байрлал

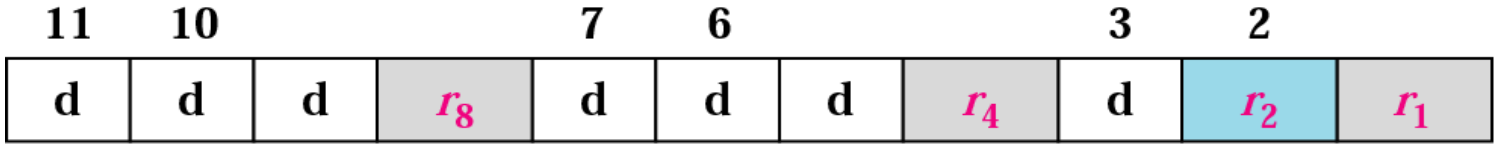


Нэмэлт битийн тооцоолол

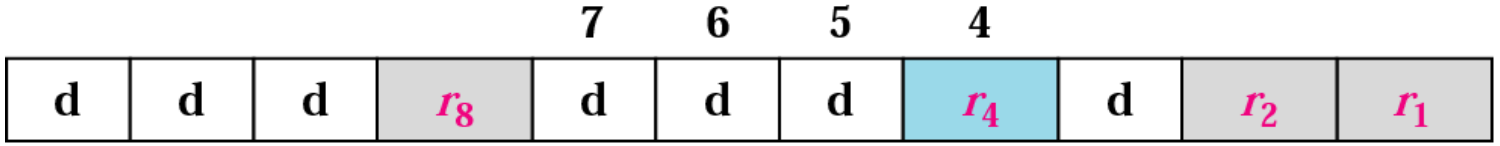
r_1 will take care of these bits.



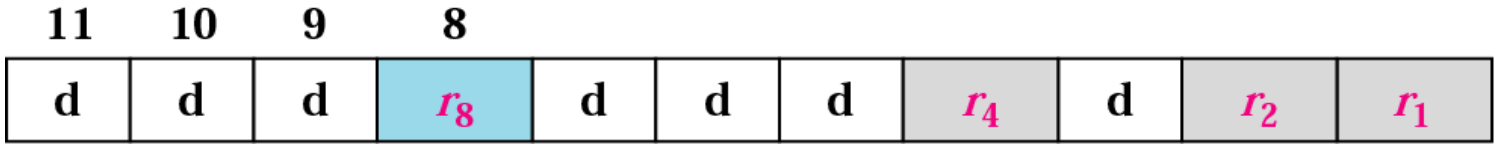
r_2 will take care of these bits.



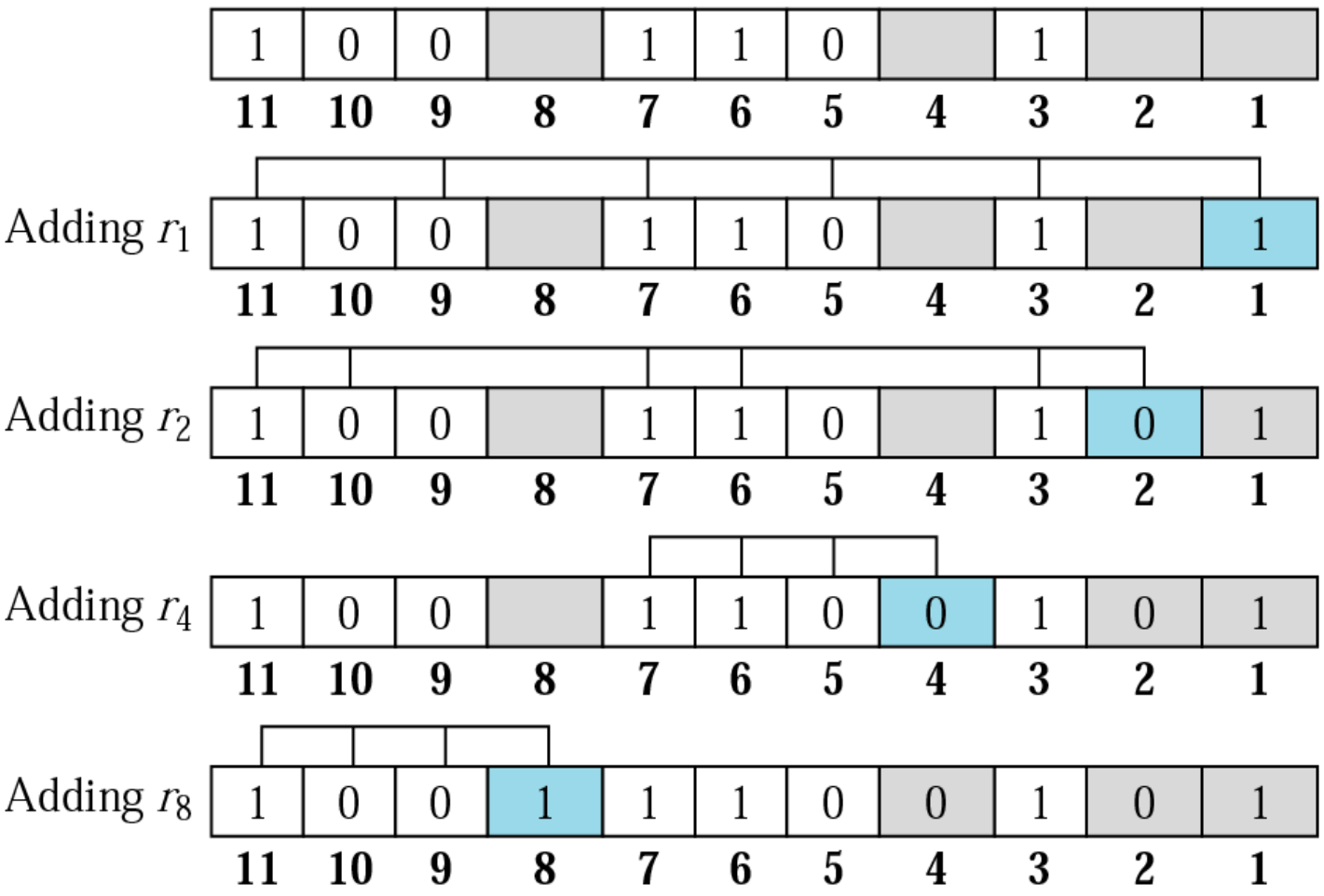
r_4 will take care of these bits.



r_8 will take care of these bits.



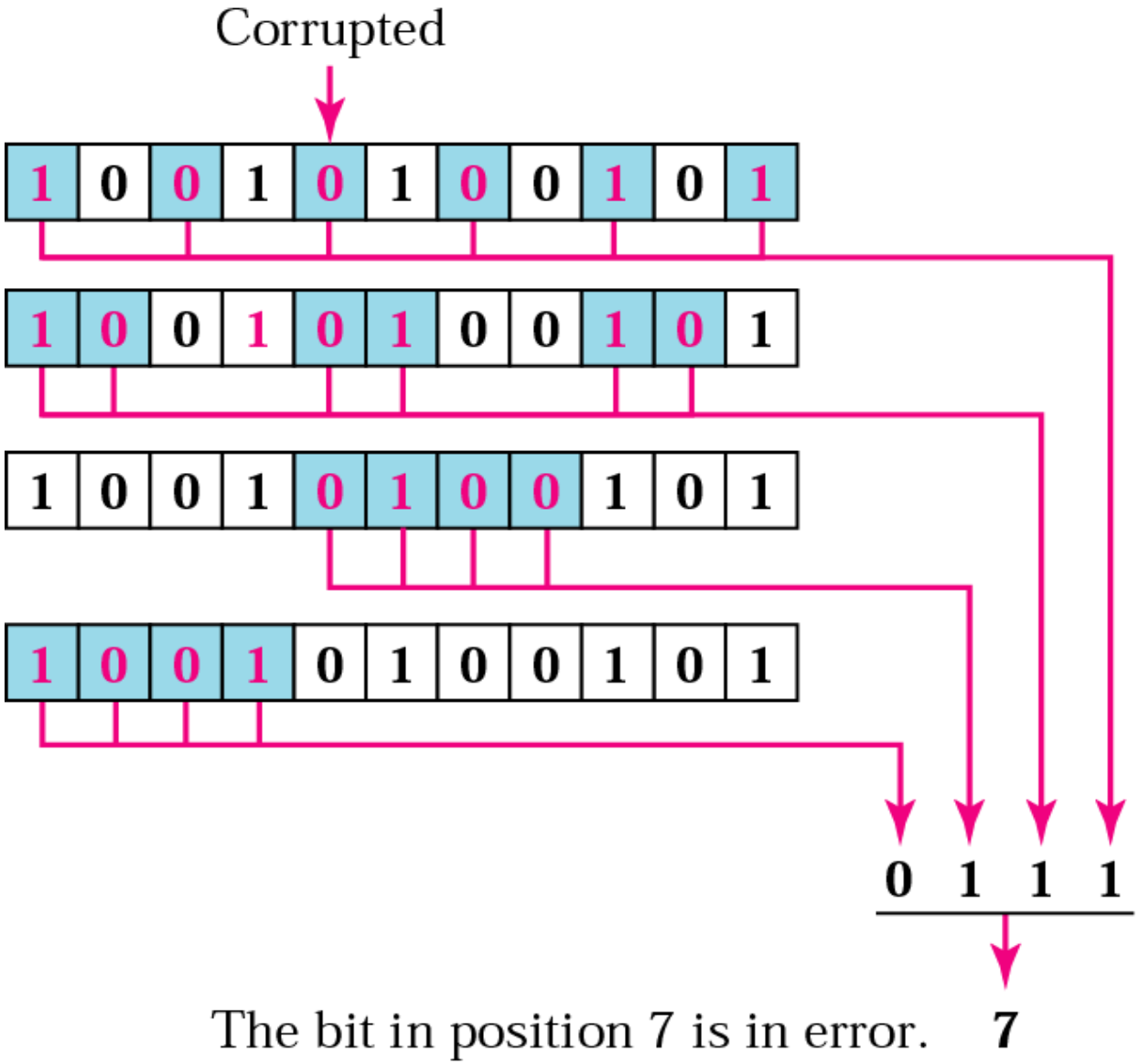
Нэмэлт битийн тооцоолол



Data:
1 0 0 1 1 0 1

Code:
1 0 0 1 1 1 0 0 1 0 1

Хаммингийн кодоор алдааг илрүүлэх



Анхаарал тавьсанд баярлалаа.