Nancy-Université
Université Nancy 2

UE 503

L3 MIAGE

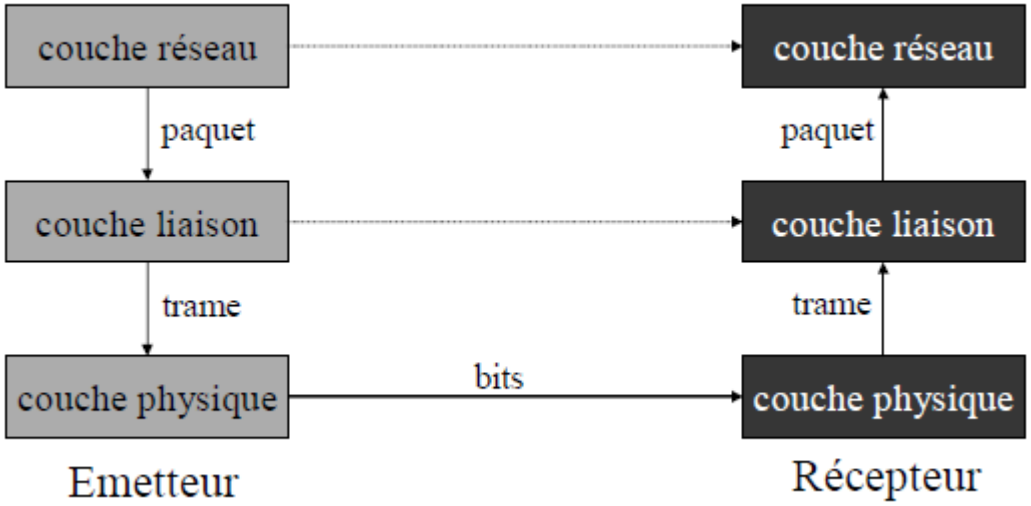
Initiation Réseau et Programmation Web
La couche liaison

A. Belaïd
abelaid@loria.fr
http://www.loria.fr/~abelaid/

Année Universitaire 2011/2012

Protocoles de la couche liaison

- Description
  - La couche liaison récupère des paquets de la couche réseau
  - Pour chaque paquet, elle construit une (ou plusieurs) trame(s)
  - Elle envoie chaque trame à la couche physique



Protocoles de la couche liaison

- Description
  - Lors de la transmission des trames des erreurs peuvent se produire :
    - le circuit de données n'est pas sûr en général
  - Le protocole de liaison de données supervise et définit un ensemble de règles pour assurer la fiabilité des échanges de ces trames sur un circuit de données

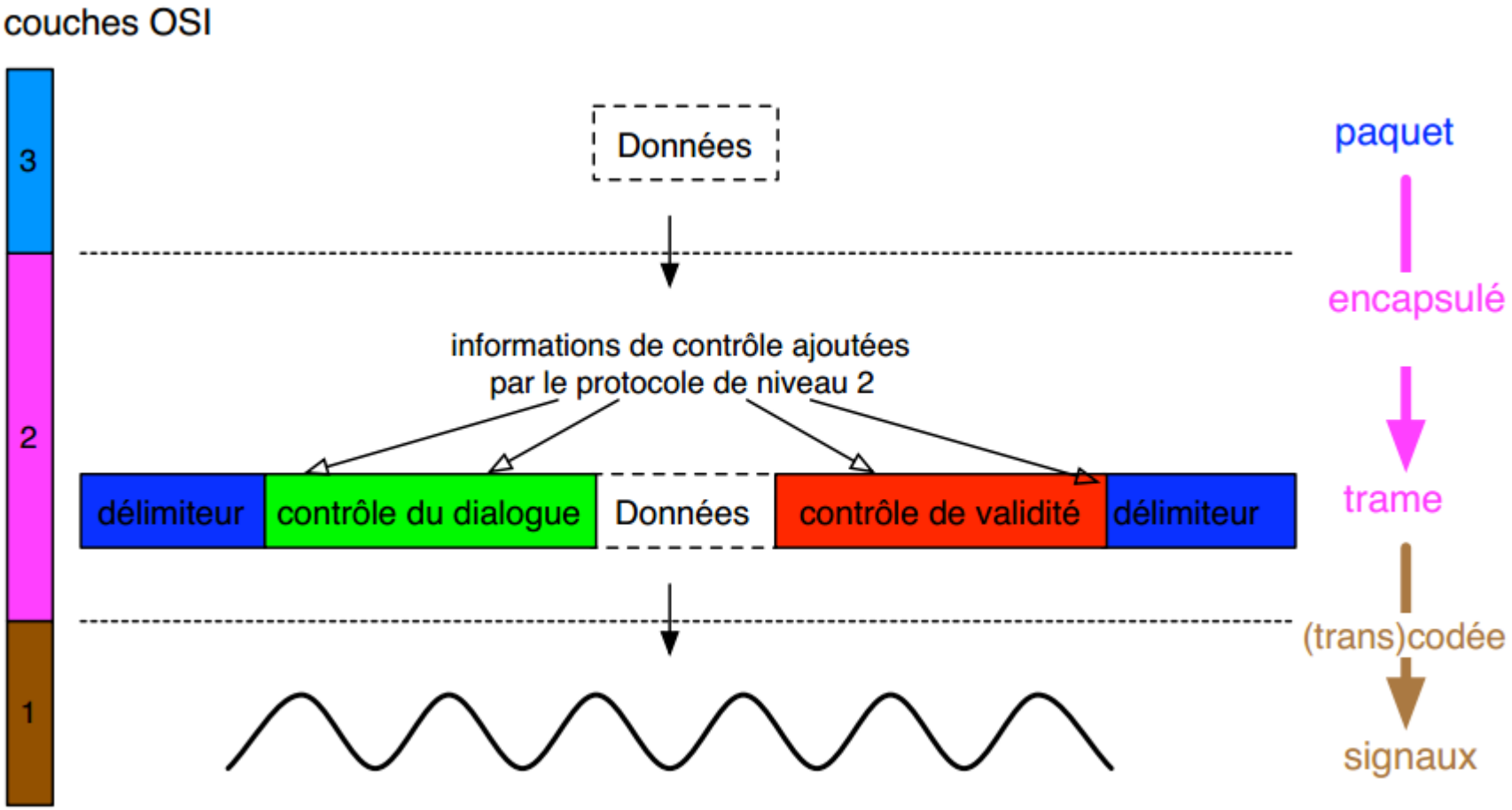
Protocoles de la couche liaison

- Règles ou services offerts
  - Gestion (délimitation) de trames
  - Contrôle d'erreur : détection/correction d'erreurs
  - Contrôle de flux entre extrémités
  - Contrôle d'accès à un canal partagé (MAC)

Délimitation de trames

- Il existe trois méthodes :
  - Compter les caractères
  - Utiliser des champs délimiteurs de trames
  - Utiliser le codage de la couche physique

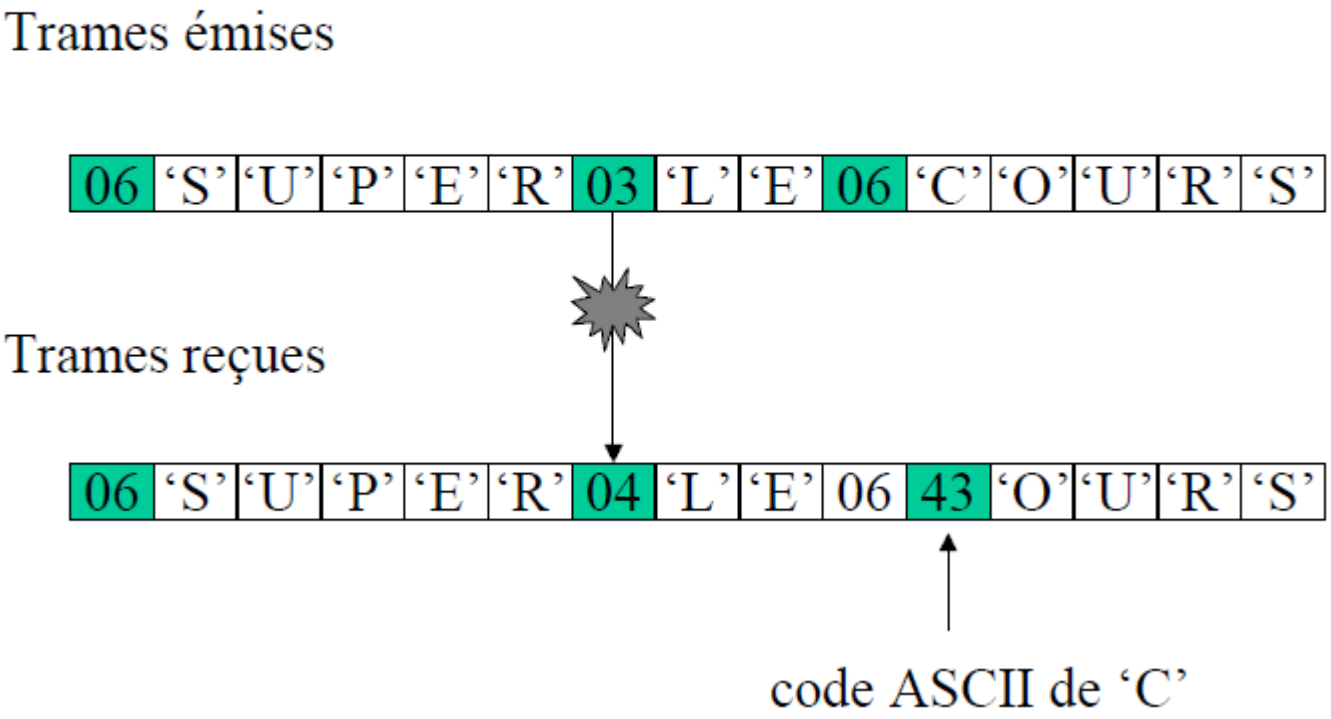
- Trame =
  - Paquet (données provenant de la couche 3) + infos de contrôle
  - On dit que le paquet est encapsulé dans une trame



Délimitation de trames

- Compter les caractères
  - On utilise un champ dans l'en-tête de la trame pour indiquer le nombre de caractères de la trame
  - Problème :
    - Si la valeur du champ est modifiée au cours de la transmission
  - Méthode rarement utilisée seule

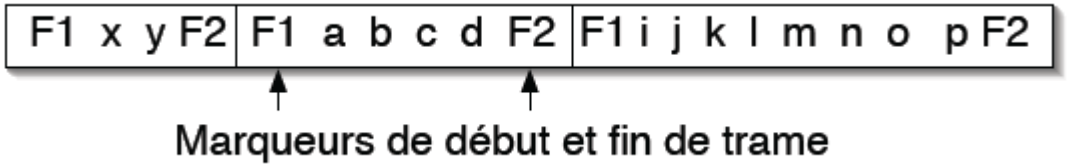
Exemple



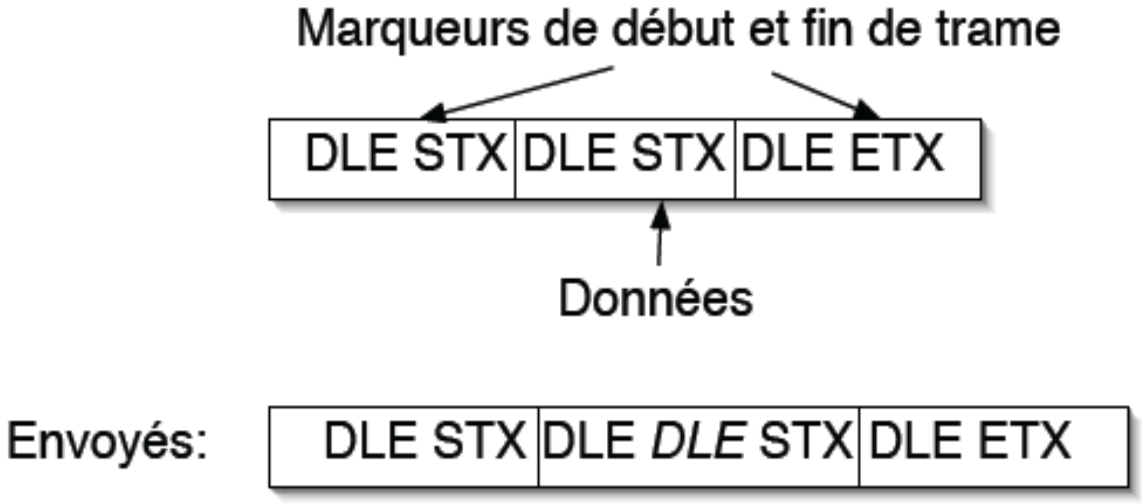
- Les chiffres en vert indiquent les longueurs des zones qui suivent y compris le chiffre

Délimitation de trames

- Utiliser des délimiteurs
  - Un marqueur est placé :
    - au début de chaque trame
    - à la fin de chaque trame (en fait, au début de la suivante)



- Problème de transparence
  - Confusion des délimiteurs de trame et des données
  - Les données peuvent contenir les délimiteurs de trames
  - Exemple 1 : caractères de délimitation
    - Les caractères DLE STX et DLE ETX délimitent le début et la fin des trames. Pour assurer la transparence des données l'émetteur rajoute un DLE devant tout DLE des données

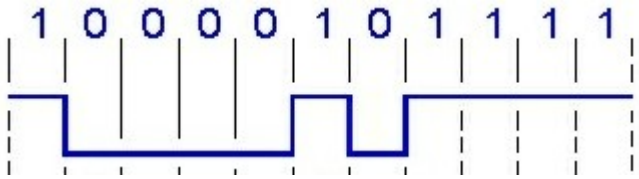


Délimiter des trames

- Avantages
  - permet toujours de retrouver la synchronisation
  - permet l'envoi de trames de tailles quelconques
  - technique la plus simple

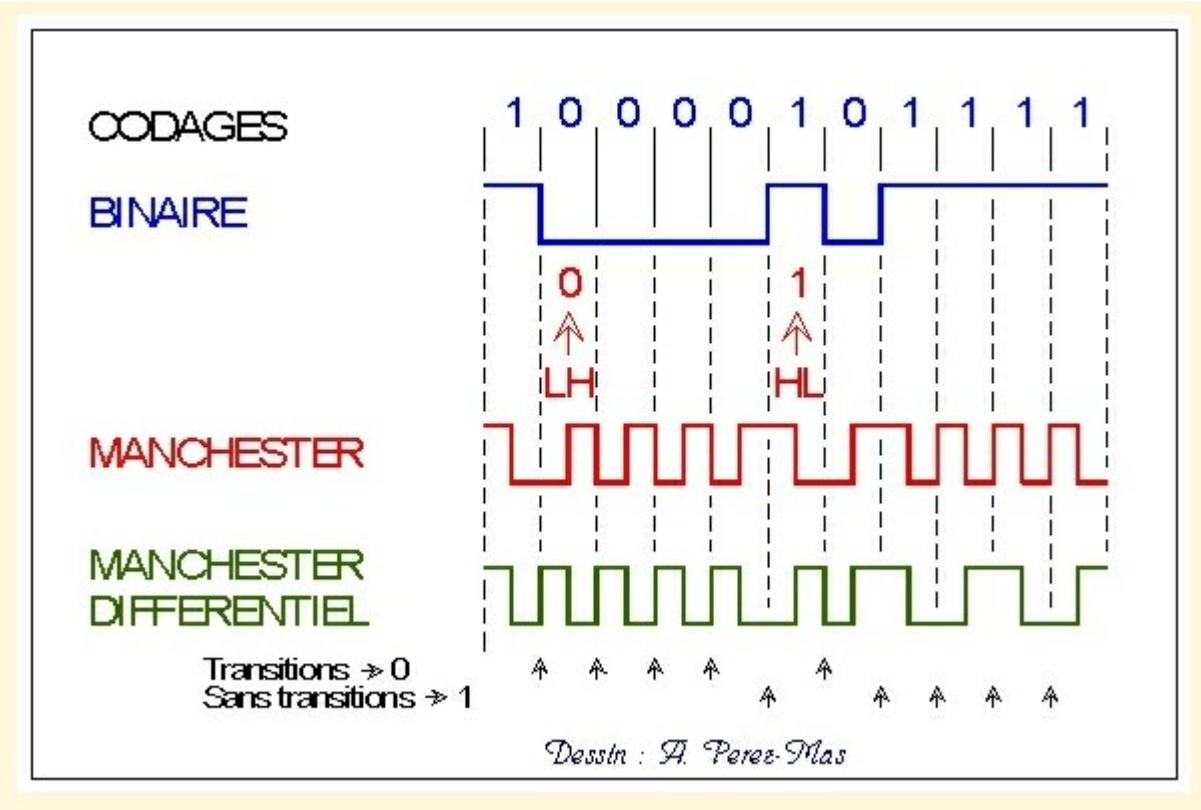
Délimiter des trames

- Utilisation du codage de la couche physique
  - On regarde la suite des impulsions dans le signal



    - 0 = impulsion positive puis négative
    - 1 = impulsion négative puis positive
  - puis on utilise les combinaisons positive-positive et négative-négative pour délimiter les trames (on appelle cela violation du codage)

– Ethernet utilise un marqueur de début de trame et effectue une violation du codage Manchester pour détecter la fin de trame



Contrôle d'erreur

- Les données peuvent être modifiées (ou perdues) pendant le transport

  ➔Deux problèmes à résoudre

  – Au niveau de la **détection** d'erreur
    - Comment se rendre compte de la modification/perte des données à l'arrivée des trames ?
  – Au niveau de la **correction** d'erreur
    - Comment corriger à l'arrivée les données erronées ? : la *correction*
    - Faire en sorte que l'émetteur, renvoie les trames erronées/perdues: la *récupération* d'erreurs

Modèle de correction

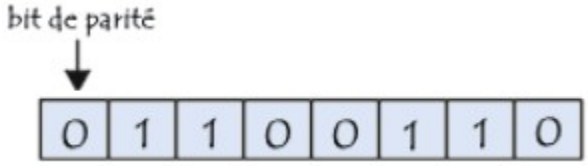
- Principe général :
  - Chaque suite de bits (une trame) à transmettre est **augmentée** par une autre suite de bits dite de redondance ou de contrôle
  - Pour chaque suite de k bits transmise, on ajoute r bits
    - On dit alors que l'on utilise un code C(n,k) avec n = k +r
  - À la réception, on effectue l'opération inverse et les bits ajoutés permettent d'effectuer des contrôles à l'arrivée

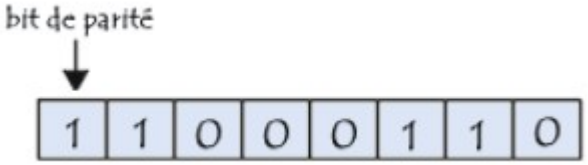
Codes

- Il existe deux catégories de codes
  - les codes détecteurs d'erreurs
  - les codes correcteurs d'erreurs
  - Cependant
    - Le code de Hamming : est à la fois un code détecteur et correcteur d'erreurs

Détection des erreurs

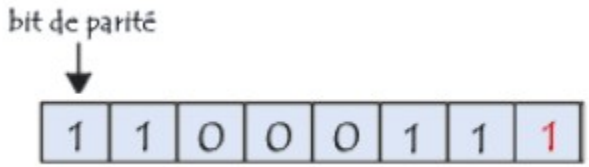
- Contrôle de parité
  - Appelé parfois VRC (Vertical Redundancy Checking)
    - Un des systèmes de contrôle le plus simple
  - Consiste à ajouter un bit supplémentaire (appelé bit de parité) à un certain nombre de bits de données appelé mot de code (généralement 7 bits) pour former un octet avec le bit de parité :
    - 1 si le nombre de bits à 1 du mot de code est impair, 0 sinon

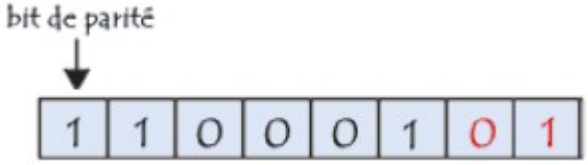




Détection des erreurs

- Contrôle de parité (suite)
  - Imaginons désormais qu'après transmission le bit de poids faible (le bit situé à droite) de l'octet précédent soit victime d'une interférence
    - Le bit de parité ne correspond alors plus à la parité de l'octet : une erreur est détectée





    - Ce système est fragile car si deux bits (ou un nombre pair de bits) venaient à se modifier simultanément lors du transport de données, aucune erreur ne serait détectée

Détection des erreurs

- Contrôle de parité croisée
  - appelé contrôle de redondance longitudinale ou Longitudinal (Redundancy Check, noté LRC) consiste non pas à contrôler l'intégrité des données d'un caractère, mais à contrôler l'intégrité des bits de parité d'un bloc de caractères
  - Soit « HELLO » le message à transmettre, en utilisant le code ASCII standard
  - Voici les données telles qu'elles seront transmises avec les codes de contrôle de parité croisé :

Détection des erreurs

- Contrôle de parité croisée
  - Ici, on met un contrôle horizontal ou longitudinal (=LRC) pour chaque lettre
  - Et un contrôle vertical (=VRC), avec un bit de parité par colonne de bit dans chaque caractère

Lettre	Code ASCII (sur 7 bits)	Bit de parité (LRC)
H	1001000	0
E	1000101	1
L	1001100	1
L	1001100	1
0	1001111	1
VRC	1000010	0

- Bilan sur le bit de parité
  - Le bit de parité est un mécanisme de détection d'erreurs peu coûteux (1 seul bit rajouté), qui est très simple mais son efficacité n'est pas extraordinaire, du fait qu'il peut avoir des messages contenant un nombre d'erreurs pair mais qui ne peut être détecté par la technique de contrôle de parité
  - Exemple :

message original	10101010	**1**
message altéré	01010101	**1**

  - Ce petit exemple nous montre que bien que le message soit altéré, le contrôle de parité ne le détecte pas, ce qui constitue la limite la plus reprochée de cette technique
- Exemple d'utilisation
  - Dans les années 90, certaines barrettes de mémoire RAM rajoutaient un bit de parité tous les 8 bits (il y avait donc 9 bits stockés), ce qui permettait de diminuer leur taux d'erreurs

Le code de Hamming

- Particularités
  - Il permet de détecter une erreur et seulement une erreur ce qui fait sa particularité
  - En cas de détection d'erreur, ce code va proposer une correction qui est la plus probable sous certaines hypothèses statistiques
    - Sur une tranche de 7 bits, il est très probable qu'il n'y ait pas plus de **1 erreur**

http://fr.wikibooks.org/wiki/Architecture_des_ordinateurs/Codes_d%C3%A9tecteurs_et_correcteurs_d'erreurs

Le code de Hamming

- Principe de base
  - On va découper l'information par tranche de N bits et on va rajouter t bits : t sera le plus petit entier tel que $2^t-1 \geq N+t$. On aura donc au total N+t bits
    - Cette formule s'explique par le fait que ces bits devant être rajoutés aux positions égales aux puissances de 2, cela nous amènerait à ce que 2^t-1 couvrirait la donnée complétée
  - Ainsi les N+t bits de la donnée finale seront notés $f_{N+t}\ f_{N+t-1}\ f_{N+t-2} \ldots f_2 f_1$

Le code de Hamming

- Exemple :
  - La donnée initiale est 0110 1110
  - N vaut donc 8
  - On cherche la plus petite valeur de t vérifiant $2^t-1 \geq 8+t$
  - On trouve facilement t=4. Il y a au total 12 bits
  - Les bits de contrôle sont f_1, f_2, f_4 et f_8
  - Exemple

f_{12}	f_{11}	f_{10}	f_9	f_8	f_7	f_6	f_5	f_4	f_3	f_2	f_1
0	1	1	0		1	1	1		0		

Le code de Hamming

- Quoi mettre maintenant comme bits de contrôle ?
  - Idée
    - Former des ensembles de bits
    - Pour trouver les ensembles de bits il faut au préalable écrire les entiers de 1 à N+t en base 2 sur t bits
    - Sur l'exemple, N vaut 8 donc t vaut 4 : il faut donc écrire les entiers de 1 à 12 en base 2 sur 4 bits

0001 ===> 1

0010 ===> 2

0011 ===> 3

0100 ===> 4

0101 ===> 5

0110 ===> 6

0111 ===> 7

1000 ===> 8

1001 ===> 9

1010 ===> 10

1011 ===> 11

1100 ===> 12

Que nous plaçons dans un tableau

```
 i--    3  2 1 0
 j      |  | | |
 1 --   0 0 0 1
 2 --   0 0 1 0
 3 --   0 0 1 1
 4 --   0 1 0 0
 5 --   0 1 0 1
 6 --   0 1 1 0
 7 --   0 1 1 1
 8 --   1 0 0 0
 9 --   1 0 0 1
10 -- 1 0 1 0
11 -- 1 0 1 1
12 -- 1 1 0 0
```

i = rang du bit, j = numéro de ligne

- Suite 1
  - On construit t ensembles de bits notés E1, E2 ... Et
  - Pour construire Ei, on regarde la colonne i à partir de la droite. S'il y a un 1 dans colonne i sur la ligne j alors le bit fj appartiendra à l'ensemble Ei.
  - On trouve donc :
    - E1 = { f1 , f3, f5, f7 , f9, f11 }
    - E2 = { f2 , f3, f6, f7 , f10, f11 }
    - E3 = { f4 , f5, f6, f7 , f12 }
    - E4 = { f8 , f9, f10, f11 , f12}

- Suite 2
  - On reporte dans chaque ensemble les valeurs trouvées :
    - f12=0
    - f11=1
    - f10=1
    - f9=0
    - f7=1
    - f6=1
    - f5=1
    - f3=0
  - On obtient :
    - E1 = { f1 , 0, 1, 1, 0, 1}
    - E2 = { f2 ,0 , 1, 1, 1, 1 }
    - E3 = { f4 ,1 , 1, 1 , 0 }
    - E4 = { f8 ,0 , 1, 1, 0}

- Suite 3
  - On s'aperçoit que dans chaque ensemble il y a un et un seul bit inconnu
  - Nous allons déterminer ce bit de manière à ce que dans chaque ensemble de bits le nombre de bits à 1 soit pair
  - On trouve donc :
    - f1=1
    - f2=0
    - f4=1
    - F8=0
  - Nous connaissons maintenant tous les bits de notre donnée finale qui est donc : 0110 0111 1001

f_{12}	f_{11}	f_{10}	f_9	f_8	f_7	f_6	f_5	f_4	f_3	f_2	f_1
0	1	1	0	0	1	1	1	1	0	0	1

- Vérification et correction du code de Hamming
  - Construire les t ensembles de bits Ei.
  - Calculer t bits selon la règle suivante ei=0 si le nombre de bits à 1 dans Ei est pair sinon ei=1
  - Si aucune erreur ne s'est produite tous les bits ei doivent être nuls
  - calculer E=(et ...e1)$_2$ (en base 2)
  - Si E est nul alors on en déduit qu'il n'y a pas eu d'erreur
  - Si E n'est pas nul, une erreur s'est produite et le code de Hamming propose la correction suivante : inverser la valeur du bit fE
  - On récupère ensuite aisément la valeur de la donnée initiale en supprimant les bits de contrôle

- Exemple :
  - On récupère la donnée 0111 0111 1001, le bit en rouge signalant l'erreur

f_{12}	f_{11}	f_{10}	f_9	f_8	f_7	f_6	f_5	f_4	f_3	f_2	f_1
0	1	1	1	0	1	1	1	1	0	0	1

  - On construit les ensembles de bits :
    - E1 = { f1 , f3, f5, f7 , f9, f11 } ={ 1 , 0, 1, 1, 1, 1} ==> le nombre de 1 (5) est impair ==> e1=1
    - E2 = { f2 , f3, f6, f7 , f10, f11 } = { 0 ,0 , 1, 1, 1, 1 } ==> le nombre de 1 (4) est pair ==> e2=0
    - E3 = { f4 , f5, f6, f7 , f12 } = { 1 ,1 , 1, 1 , 0 } ==> le nombre de 1 (4) est pair ==> e3=0
    - E4 = { f8 , f9, f10, f11 , f12} ={ 0 ,1 , 1, 1, 0} ==> le nombre de 1 (5) est impair ==> e4=1
    - E s'écrit donc en base 2 (e4e3e2e1) soit (1001). E vaut donc 9. Il y a donc eu une erreur : la correction propose d'inverser la valeur de bit f9. f9 valait 1 : nous allons donc changer sa valeur en 0

- Suite :
  - La donnée corrigée est donc :

f_{12}	f_{11}	f_{10}	f_9	f_8	f_7	f_6	f_5	f_4	f_3	f_2	f_1
0	1	1	0	0	1	1	1	1	0	0	1

  - En enlevant les bits f8, f4, f2 et f1, on obtient donc la donnée initiale après correction soit 0110 1110

Codage de Hamming et de parité

- Inconvénients
  - Ces deux codages sont intéressants mais ne sont pas adaptés à tous les types de situation :
    - lorsqu'il n'y a que peu d'erreurs et que ces erreurs sont statistiquement isolées, ces codes peuvent être intéressants
    - Mais dès qu'il y a un grand nombre d'erreurs et que ces erreurs arrivent groupées, alors ces codages deviennent de moins en moins performants

Détection des erreurs

- Contrôle de redondance cyclique (CRC)
  - Consiste à protéger n'importe quelle donnée : paquet, trame
  - A chaque trame est associé un code de contrôle (parfois appelé CRC par abus de langage ou FCS pour Frame Check Sequence dans le cas d'un code de 32 bits ou cheksum)
  - Le code CRC contient des éléments redondants vis-à-vis de la trame, permettant de détecter les erreurs, mais aussi de les réparer

Détection des erreurs

- Principe du CRC
  - Il consiste à traiter les séquences binaires comme des polynômes binaires, c'est-à-dire des polynômes dont les coefficients correspondent à la séquence binaire
  - Ainsi la séquence binaire 0110101001 peut être représentée sous la forme polynomiale suivante :
    - $0*X^9 + 1*X^8 + 1*X^7 + 0*X^6 + 1*X^5 + 0*X^4 + 1*X^3 + 0*X^2 + 0*X^1 + 1*X^0$
  - soit
    - $X^8 + X^7 + X^5 + X^3 + X^0$
  - ou encore
    - $X^8 + X^7 + X^5 + X^3 + 1$

Détection des erreurs

- Autre représentation des nombres binaires
  - 0110101001 : $X^8 + X^7 + X^5 + X^3 + 1$
  - De cette façon
    - le bit de poids faible de la séquence (le bit le plus à droite) représente le degré 0 du polynôme ($X^0 = 1$),
    - le 4ème bit en partant de la droite représente le degré 3 du polynôme (X3)...
    - Une séquence de n bits constitue donc un polynôme de degré maximal n-1
    - Toutes les expressions polynomiales sont manipulées par la suite avec une arithmétique modulo 2 (i.e. X est représentée par 0 ou 1)

Détection des erreurs

- Utilisation
  - On définit un diviseur du polynôme précédent (appelé polynôme générateur et noté G(X))
    - Il sera prédéfini par le protocole de liaison
    - et connu de l'émetteur et du récepteur
  - Le reste de la division polynomiale de M/G servira au contrôle
  - La détection d'erreur consiste :
    - Pour l'émetteur : à envoyer : M' = M + reste(M/G)
    - Pour le récepteur : Il suffit d'effectuer le même calcul afin de vérifier que le CRC est valide : M'/G = 0

http://www.commentcamarche.net/contents/base/control.php3

Détection des erreurs

- Intérêt du G(X)
  - Une division polynomiale est similaire à une division entière, sauf que l'on essaie à chaque fois de supprimer le terme de degré supérieur
  - Exemple :
    - $M = X^2-1 = (X+1) * (X-1)$
    - $G = X+1$
    - $M/G = X-1$

- Exemple
  - Soit maintenant M un message binaire de 16 bits :

    1011 0001 0010 1010
  - On convertit ce message en polynôme correspondant
    - $X^{15} + X^{13} + X^{12} + X^8 + X^5 + X^3 + X$
  - Prenons $G(X) = X^3 + 1$ (représenté en binaire par 1001)
  - Étant donné que G(X) est de degré 3, il s'agit d'ajouter 4 bits nuls à M : 10110001001010100000 de manière à pouvoir diviser et prendre tous les chiffres en compte
  - Le CRC est égal au reste de la division de M par G

$X^{19}+X^{17}+X^{16}+X^{12}+X^9+X^7+X^5$ | X^3+1

$-X^{19}-X^{16}$ | $X^{16}+X^{14}-X^{11}+X^9+X^8-X^5+X^4+X^3-X-1$

$X^{17}+X^{12}+X^9+X^7+X^5$

$-X^{17}-X^{14}$

$-X^{14}+X^{12}+X^9+X^7+X^5$

$+X^{14}+X^{11}$

$X^{12}+X^{11}+X^9+X^7+X^5$

$-X^{12}-X^9$

$X^{11}+X^7+X^5$

$-X^{11}-X^8$

$-X^8+X^7+X^5$

$+X^8+X^5$

$X^7+2X^5\ (=X^6)$

$-X^7-X^4$

X^6-X^4

$-X^6-X^3$

$-X^4-X^3$

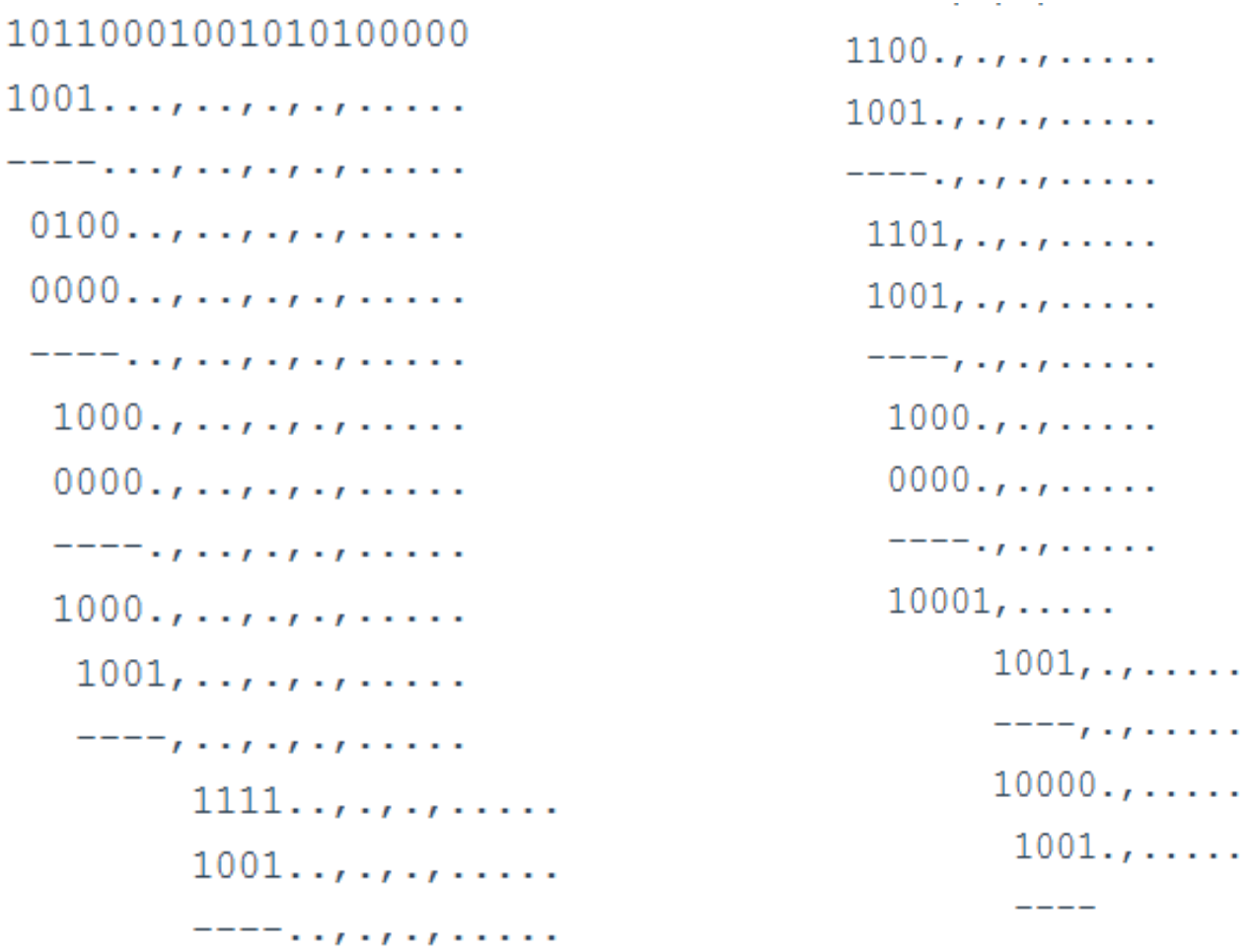
$+X^4+X$

$-X^3+X$

$+X^3+1$

X+1 ► le reste est X+1, soit 0011

La même avec des nombres binaires



La même avec des nombres binaires

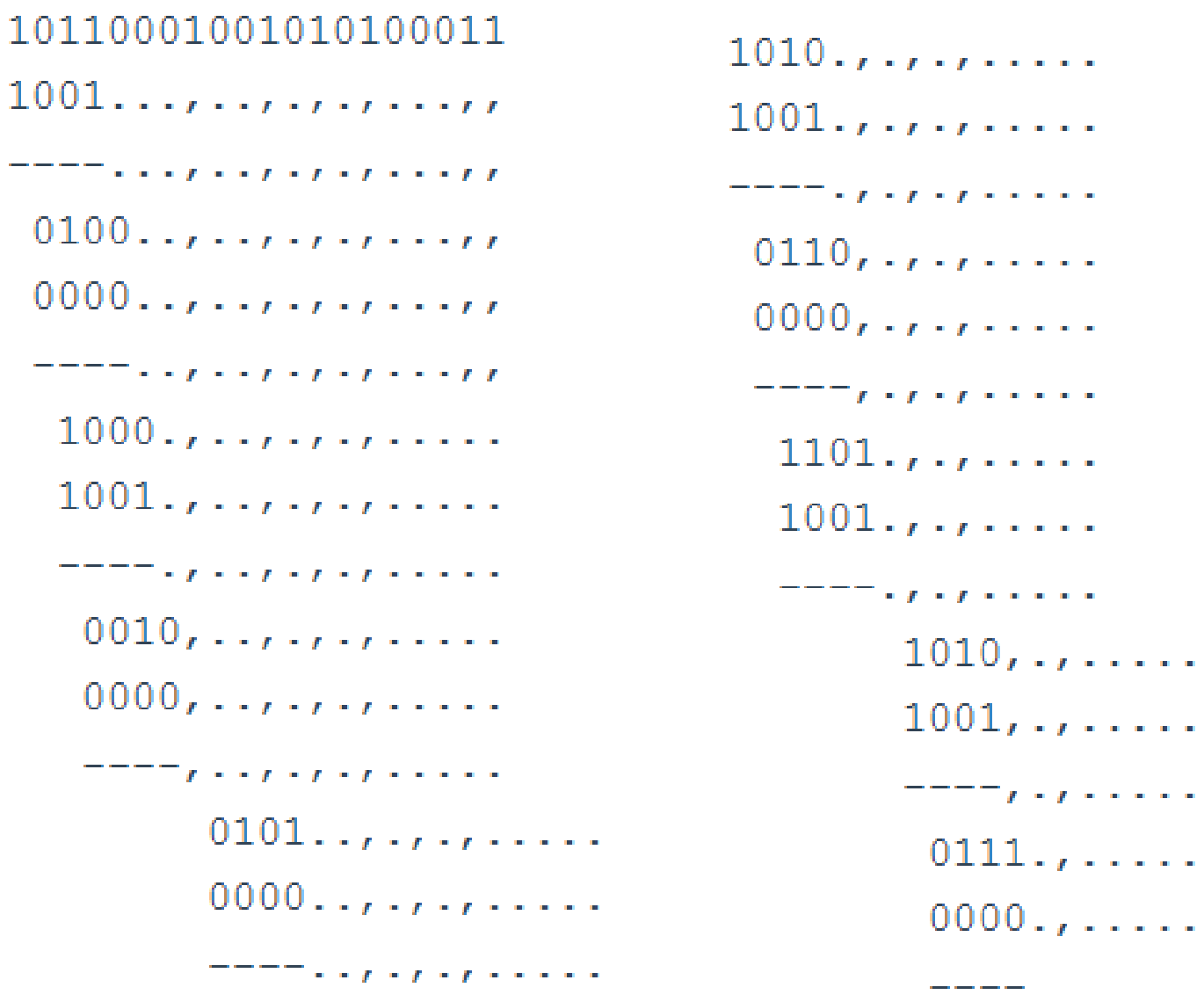
```
1111,.....
1001,.....
----,.....
 1100.....
 1001.....
 ----.....
      1100....
      1001....
      ----....
       1010...
       1001...
       ----...
        0110..
        0000..
        ----..
```

```
1100.
1001.
----.
     1010
     1001
     ----
     0011
```

Détection des erreurs

Restitution

- Le récepteur
  - L'émetteur envoie le message M accompagné du reste obtenu par CRC
    - M' = 1011000100101010 + 0011
    - Au lieu de rajouter 4 (0) comme précédemment, on ajoute le reste
    - M' = 10110001001010100011
  - Le récepteur du message effectue la division de M' par G, et obtient un reste nul si la transmission s'est effectuée sans erreur :


10110001001010100011
1001

0100
0000

1000
1001

0010
0000

0101
0000

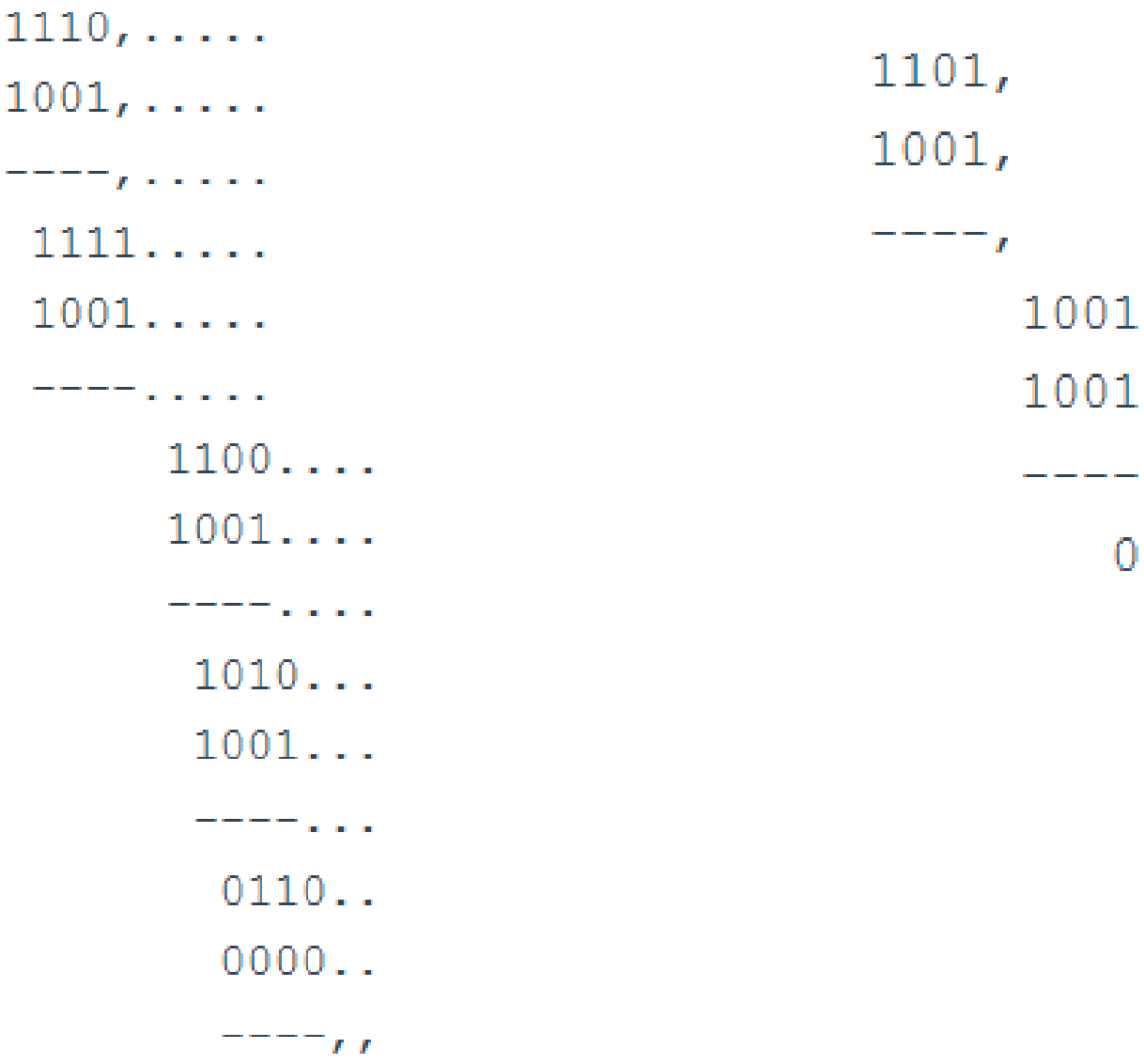
1010
1001

0110
0000

1101
1001

1010
1001

0111
0000


1110,.....
1001,.....
----,.....
1111.....
1001.....
----.....
1100....
1001....
----....
1010...
1001...
----...
0110..
0000..
----,,
1101,
1001,
----,
1001
1001

0

Détection des erreurs

Polynômes générateurs

- Les polynômes générateurs les plus couramment employés sont :

- **CRC-12** : $X^{12} + X^{11} + X^3 + X^2 + X + 1$
- **CRC-16** : $X^{16} + X^{15} + X^2 + 1$
- **CRC CCITT V41** : $X^{16} + X^{12} + X^5 + 1$
  (Ce code est notamment utilisé dans la procédure *HDLC*.)
- **CRC-32** (**Ethernet**) : = $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- **CRC ARPA** : $X^{24} + X^{23} + X^{17} + X^{16} + X^{15} + X^{13} + X^{11} + X^{10} + X^9 + X^8 + X^5 + X^3 + 1$

Contrôle de flux

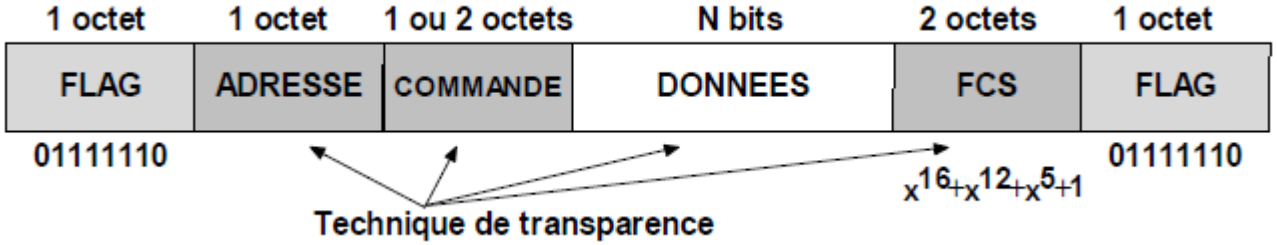
- Objectif
  - Réguler le flux de données entre un émetteur et un récepteur, en termes de
    - Circulation
    - Acquittement
    - Stockage temporaire
    - etc.

Le protocole HDLC
High-Level Data Link Control

- Objectif
  - Nous allons voir ce protocole qui montre comment tous ces contrôles sont pris en compte au niveau de la trame
  - Son but est de définir un mécanisme pour délimiter des trames de différents types, en ajoutant un contrôle d'erreur
  - Protocole de référence normalisé par l'ISO
  - Utilisé dans de nombreux réseaux : Transpac, Numéris, LAN, Internet, GSM
- Caractéristiques
  - Transmission synchrone
  - Orienté bit
  - Liaisons point-à-points ou multi-points

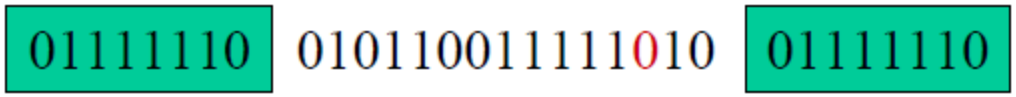
HDLC

- Format de la trame
  - L'unité utilisée est la trame (Frame)
  - Chaque trame est délimitée par deux fanions identiques



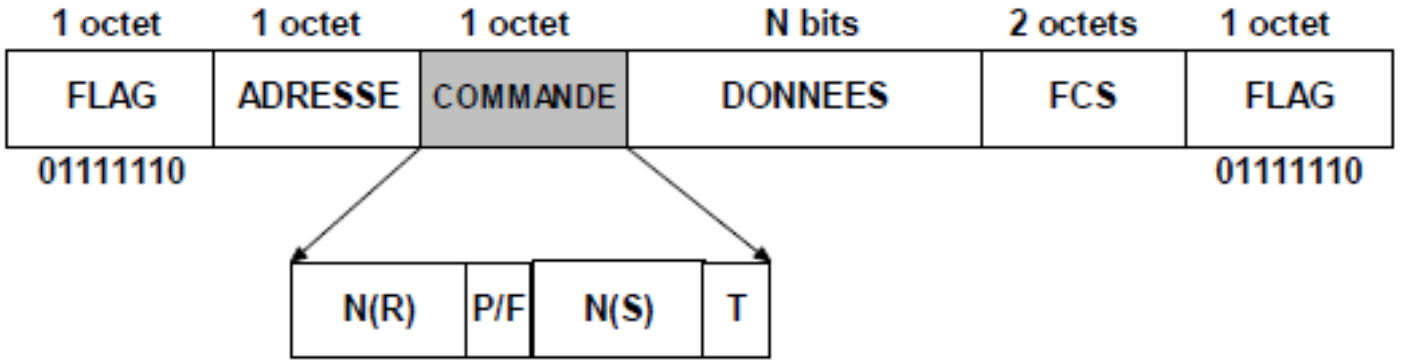
  - Fanion
    - Le fanion est un délimiteur de trame pour la synchronisation. Sa valeur est : 01111110
    - Les trames HDLC peuvent être envoyées les unes derrière les autres : dans ce cas, le fanion de fin de la première trame peut être mis en commun et servir de fanion de début pour la trame suivante

- Utilisation des fanions
  - Pour assurer la transparence, l'émetteur ajoute systématiquement un 0 après toute séquence de 5 bits à 1 rencontrée dans la trame
  - Le récepteur effectue l'opération inverse, c'est à dire qu'il retire le 0 qui suit chaque séquence de 5 bits à 1
- Exemple
  - Fanion : 01111110
  - Bit de transparence : 0 inséré après toute séquence de cinq 1 successifs dans la trame
  - Données : 010110011111110
  - Trame :



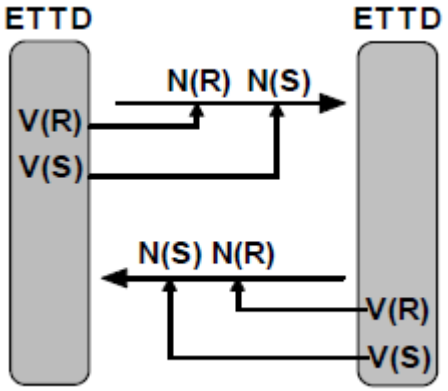
- Adresse (Address)[modifier]
  - L'adresse est celle du destinataire à qui est envoyée la trame
  - Cette adresse était utilisée lorsque la communication était de type maître-esclave, l'adresse étant celle de l'esclave
  - En communication point-à-point, elle n'est pas utilisée

- Commande (Control)
  - Ce champ permet de distinguer 3 types de trames :



- T (1 bit) : Indique le type de trame
  - ➢ 0 : information, 1 : supervision
- N(S) et N(R) (6 bits) : Indique le numéro des trames émises et reçues
- P/F (1 bit) : Demande de réponse immédiate à la suite de l'envoi d'une trame de commande

- Numérotation des trames
  - V(S) : numéro de la prochaine trame à envoyer (0 à 7)
  - V(R) : numéro de la prochaine trame attendue en réception (0 à 7)
  - N(S) : numéro de la trame
  - N(R) : acquittement des trames reçues de numéro strictement inférieur à N(S)



  - Un équipement terminal de traitement de données (ETTD est un élément susceptible d'échanger des données avec un réseau, qui ne se connecte pas directement à la ligne de transmission. Par exemple : un ordinateur, un terminal, une imprimante…

- Données
  - Ce champ optionnel, de longueur variable, contient les données à envoyer
  - Le nombre de bits à expédier n'a pas à être un multiple de 8 :
    - comme ce champ n'a pas besoin d'être aligné du point de vue octet, il n'est pas nécessaire d'ajouter de bits de bourrage à la fin
- FCS
  - Frame Check Sequence
    - le FCS est un code ajouté après les données pour détecter d'éventuelles erreurs de transmission
    - Il est codé habituellement sur 16 bits, mais après négociation entre les deux interlocuteurs, il peut être sur 32 bits
    - Cette séquence correspond au CRC calculé sur les champs adresse + commande + données

- Transparence
  - Pour que le fanion serve de délimiteur, il est indispensable que la valeur de celui-ci ne se trouve pas dans les données transportées entre le début et la fin
  - Pour cela, les données seront modifiées pour éliminer les séquences de bits 01111110
  - Il y a deux méthodes :
    - la méthode par bit et la méthode par octet
  - La première méthode (appelée bourrage de bit, Bit stuffing en anglais) est la plus courante :
    - il s'agit d'éviter de rencontrer six bits consécutifs de valeur 1
    - Lors de l'écriture de la trame, si les données contiennent 5 bits successifs à la valeur 1, un 0 est automatiquement ajouté après

- La deuxième méthode (appelée bourrage d'octet)
  - utilise un caractère d'échappement, de valeur hexadécimal CE
  - Si parmi les octets à envoyer, on rencontre la valeur du fanion (01111110 ou 7E en hexa), alors cet octet est remplacé par les deux octets suivants : CE puis 5E
  - Du coup, il s'agit de s'assurer que la valeur de l'octet d'échappement ne se trouve pas dans les données, si on le rencontre, l'octet CE est alors remplacé par les octets CE et 9E.
  - Ainsi il n'y a pas, avant transmission, de confusion possible entre données et fanions de début/fin

HDLC

- 3 types de trames circulent sur la liaison
  - Trames I (Information)
    - Transportent les données utilisateurs
    - Acquittement – Retransmision (Piggybacking)
  - Trames S (Supervision)
    - Acquittement RR - RNR
    - Retransmission REJ - SREJ
    - Contrôle de flux RR - RNR
  - Trames U (Unnumbered) SABM – UA - DISC
    - Gestion de la liaison (initialisation, libération, …)

HDLC

- Trames de supervision : S
  - Ces trames transportent des commandes ou des réponses liées au contrôle d'erreurs, et au contrôle de flux
    - RR = Receive Ready [1 0 0 0 P/F Nr] :
      - le récepteur est prêt à recevoir
    - RNR = Receive Not Ready [1 0 1 0 P/F Nr] :
      - le récepteur ou la couche réseau est débordé
    - REJ = Reject [1 0 0 1 P/F Nr] :
      - demande de retransmission des trames de numéro supérieur ou égal à Nr
    - SREJ = Selective Reject [1 0 1 1 P/F Nr] :
      - demande de retransmission de la trame numéro Nr

HDLC

- Trames non numérotées : U
  - Ces trames transportent des commandes ou des réponses de la gestion de la liaison (établissement, rupture, choix d'un mode de réponse…)

HDLC

- Commandes
  - SABM = Set Asynchronous Balanced Mode [1 1 1 1 P/F 1 1 0] :
    - demande de connexion
  - SABME = Identique à SABM, mais mode étendu (numéroté en modulo 128)
  - DISC = Disconnect [1 1 1 1 P/F 0 1 0] :
    - libération de connexion

HDLC

- Réponses
  - UA = Unnumbered Acknowledgement [1 1 0 0 P/F 1 1 0] :
    - acquittement de trame non-numérotée
  - FRMR = FRaMe Reject [1 1 1 1 P/F 0 1 1] :
    - rejet de trame
  - DM = Disconnect Mode [1 1 1 1 P/F 0 0 0] :
    - le terminal est déconnecté

HDLC

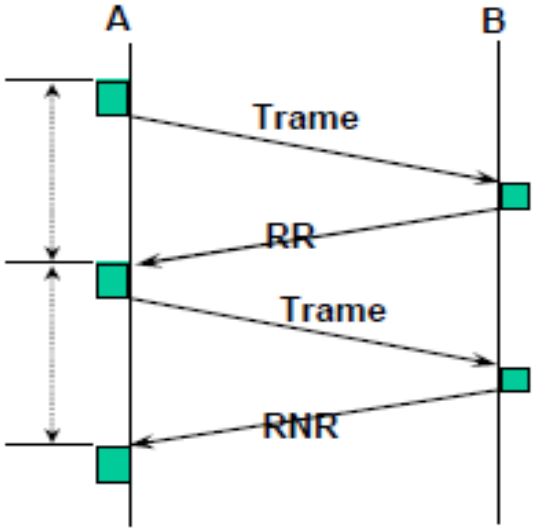
- Échanges
  - Il existe 2 modes de fonctionnement dans HDLC :
    - Le mode Best-Effort : dans ce mode, on ne garantit pas la livraison de toutes les trames. Cela est pris en charge par la couche réseau du protocole ISO.
    - Le mode Balanced : dans ce mode, on utilise des mécanismes hardware pour assurer la fiabilité des transmissions
  - Le protocole HDLC est la couche liaison utilisée pour de nombreux protocoles : H.323, V.120, TCN ou X.25.
  - N.B. : il existe une variante spécifique de HDLC développée par Cisco, qui modifie l'usage de champ Adresse et ajoute 2 octets de protocole

Retour sur le contrôle de flux

- Plusieurs protocoles
  - Protocole de type « envoyer et attendre » (Send and Wait)
    - Les données ne circulent que dans un sens
    - Une seule trame est envoyée à la fois
    - Le récepteur informe l'émetteur de son état par un *acquittement*
  - Protocoles avec fenêtre d'anticipation (Sliding Window)
    - Les données circulent dans les deux sens
    - Plusieurs trames sont envoyées à la fois
    - Liste des numéros de séquence de trames = *fenêtre d'anticipation*

Mécanisme « SEND & WAIT » simple et utopique

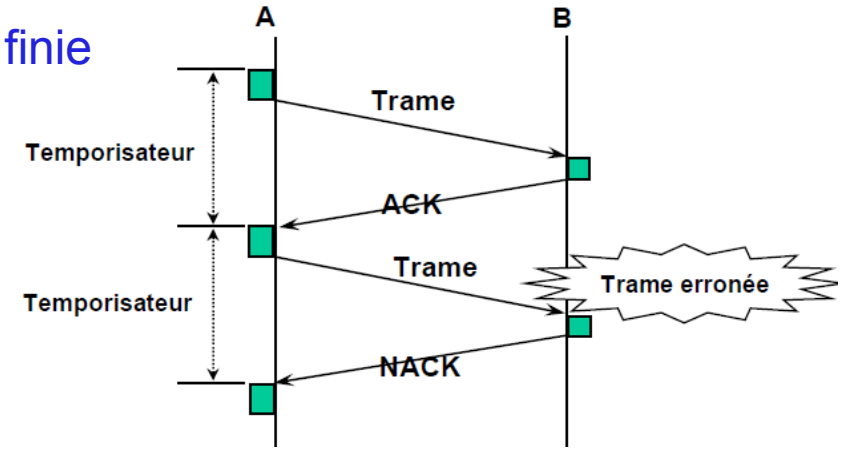
- Hypothèses
  - Transmission de trames de données (I) dans un seul sens
  - Canal de communication parfait (pas d'erreur ni perte)
  - Taille des mémoires de tampon finie
- Solution
  - Introduction de 2 trames de supervision (S), qui ne transportent aucune information utile et qui sont invisibles aux utilisateurs :
    - RR (Receiver Ready)
    - RNR (Receiver Not Ready)
- 2 variantes
  - Envoi d'une trame de supervision après chaque trame
  - Envoi d'une trame RNR ssi tampon plein, suivi d'une trame RR pour reprendre les envois



CONTRÔLE DE FLUX

Mécanisme « SEND & WAIT » avec acquittement

- Hypothèses :
  - Transmission de données dans un sens
  - Canal de communication bruité
  - Taille de mémoires de tampon finie
- Problèmes:
  - Trames perdues
  - Trames erronées
  - Duplication de trame
- Solution :
  - Ajouter un processus d'acquittement positif ou négatif
  - Utiliser un temporisateur ou Timer pour borner le délai de réception des ACK

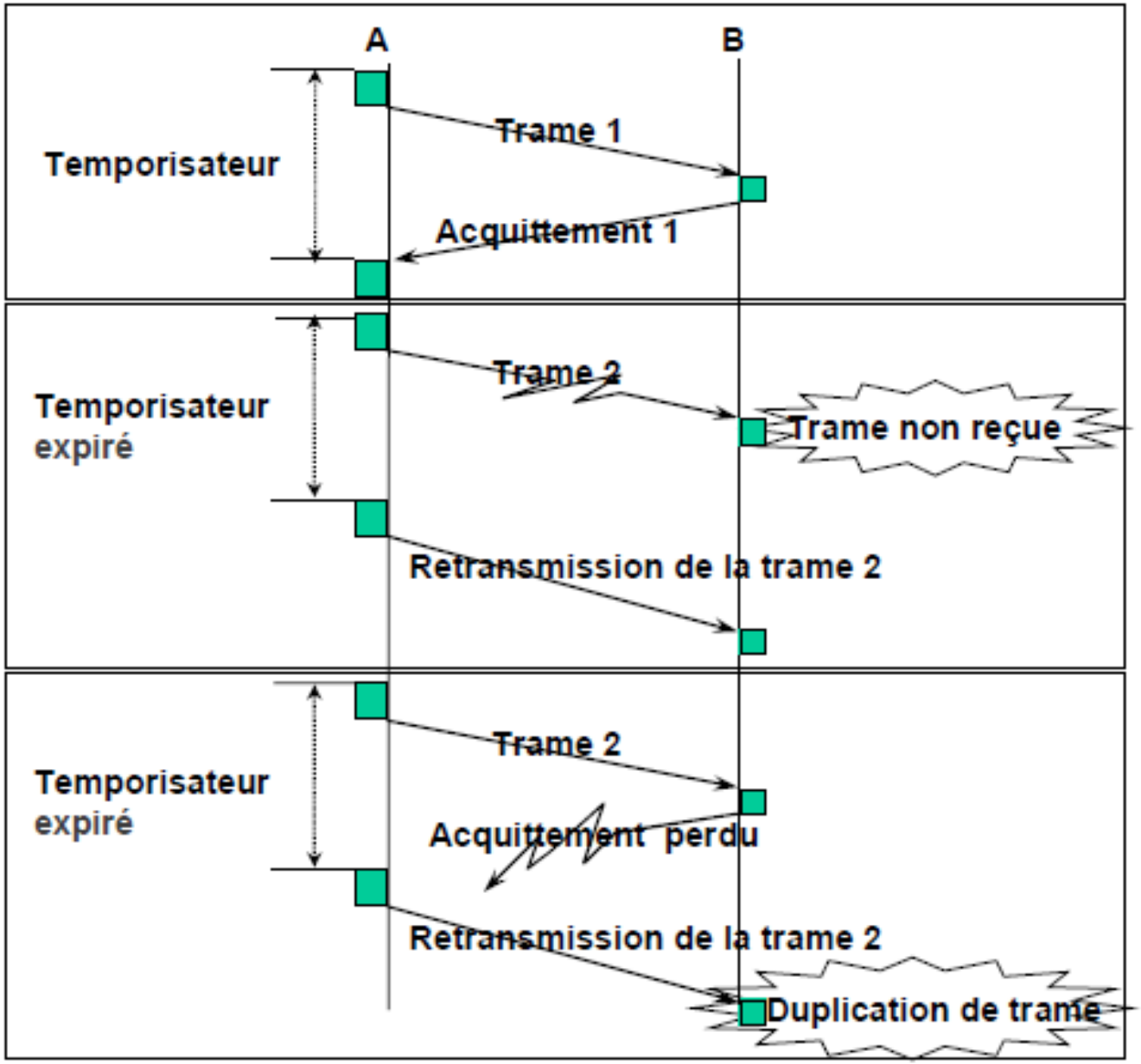


CONTRÔLE DE FLUX

Mécanisme « SEND & WAIT » avec acquittement

- Remarque1
  - La fonction de Contrôle de Flux et de contrôle d'erreurs peuvent utiliser la même trame de supervision (par exemple RR et RNR)
- Remarque 2
  - Les trames de supervision peuvent aussi servir d'acquittement afin de gérer les erreurs de transmission
  - Cela nécessite l'utilisation d'un temporisateur

- Illustration



CONTRÔLE DE FLUX

Mécanisme « SEND & WAIT » avec numérotation des acquittements

- Solution
  - Pour éviter qu'une même trame soit reçue deux fois, le protocole de communication numérote chaque trame d'information

  - Deux trames avec des numéros différents seront considérées différentes !

CONTRÔLE DE FLUX

Mécanisme « SEND & WAIT » avec numérotation des acquittements

- Numérotation
  - Ajout d'un champ N(S) dans l'en-tête des trames de données et de supervision
    - N(S) = Numéro de la prochaine trame attendue à la réception
  - Ajout de compteurs V(S) et V(R) dans les terminaux émetteurs et récepteurs
    - V(S) contient le numéro de la prochaine trame à envoyer, et V(R) celui de la trame à recevoir
  - Requiert une initialisation de l'échange pour la négociation de la valeur du compteur (protocole en mode connecté)

CONTRÔLE DE FLUX

Mécanisme « SEND & WAIT » avec numérotation des acquittements

- Fonctionnement
  - Émetteur
    - Émission d'une trame *n*
    - Attendre l'acquittement de la trame émise
    - Si acquittement de la trame est reçu alors émission de la prochaine trame *n+1*
  - Récepteur
    - Réception d'une d'une trame *n*
    - Vérification de l'intégrité et de la non duplication de la trame
    - Si OK alors envoi d'un acquittement pour la trame *n*

Le Piggybacking

- Objectif
  - Réduire le trafic de trames de supervision (S)
- Principe
  - **Lors d'un dialogue bidirectionnel, les trames d'informations utiles (I)** peuvent être utilisées pour faire des acquittements positifs et donc se substituer aux trames de supervision
  - Chaque trame I doit alors posséder 2 champs de numérotation N(S) et N(R) pour assurer les acquittements

Le Piggybacking

- REMARQUE 1
  - Les trames RNR, REJ (reject) et SREJ sont toujours transportées explicitement
- REMARQUE 2
  - Si une station n'a pas de trame I à transmettre, elle peut toujours utiliser explicitement des trames RR pour acquitter le trafic qu'elle reçoit

CONTRÔLE DE FLUX

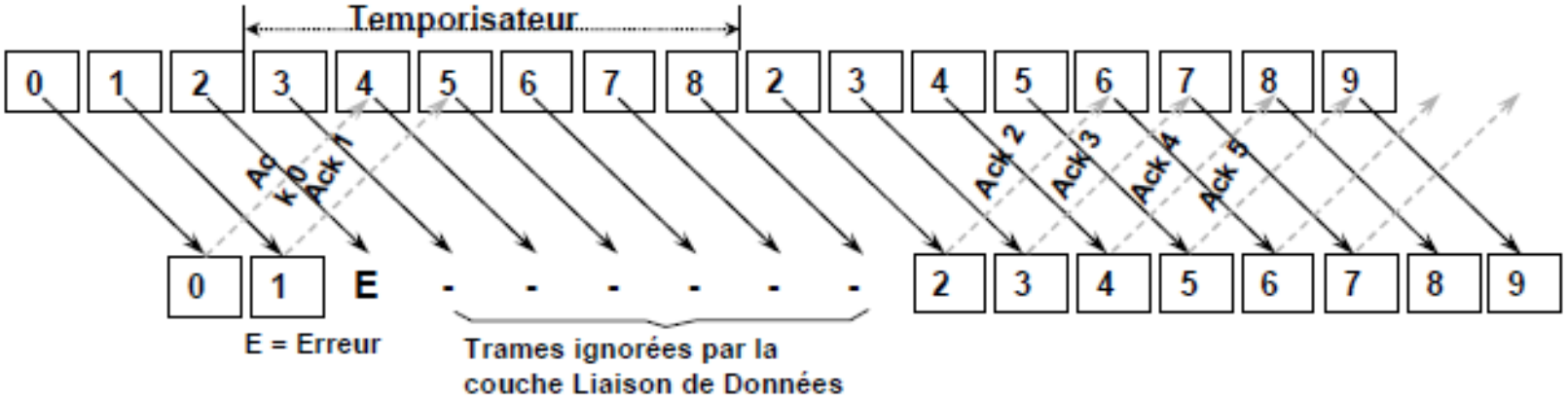
Mécanisme avec Fenêtre d'anticipation de taille N

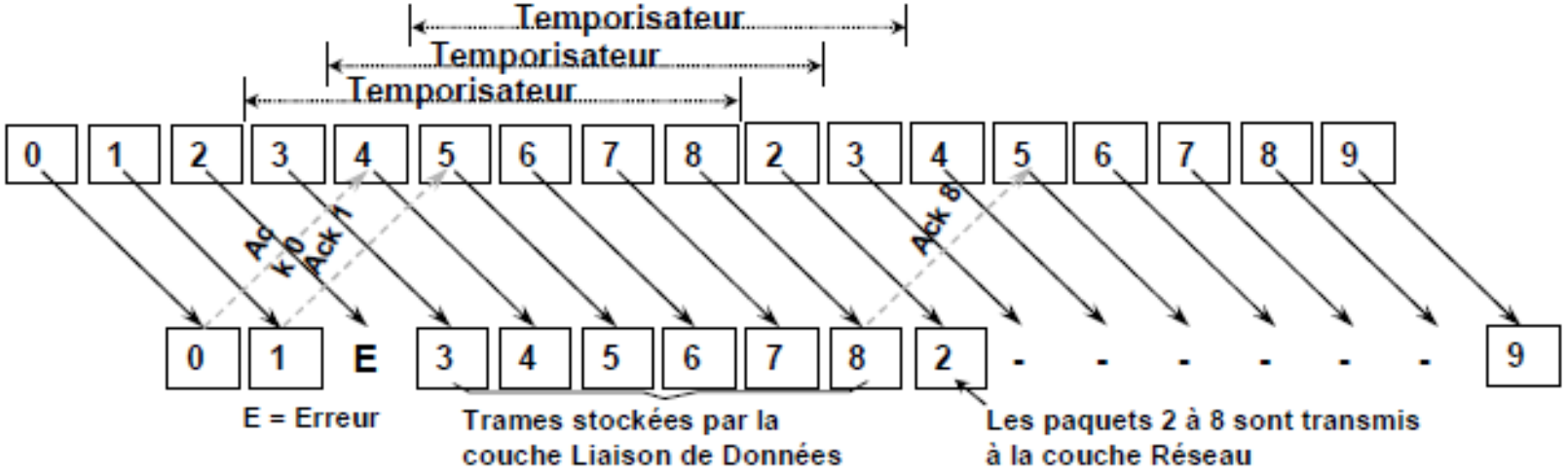
- Objectif
  - Augmenter l'efficacité du dialogue
  - Efficacité = Nb de bits envoyés / Temps total de l'échange
- Principe
  - Émission de plusieurs trames à la suite sans attendre la réception d'un ACK
  - Une trame de supervision peut acquitter un groupe de trames de données
  - Nombre de trames émises avant ACK = *N-1*

CONTRÔLE DE FLUX

Mécanisme avec Fenêtre d'anticipation de taille N

- Principe (suite)
  1. RETRANSMISSION
     - (GO-Back-N) de toutes les trames à partir de la trame erronée ou perdue au moyen d'une trame de supervision REJ
  2. REJET SELECTIF
     - (Selective Reject) au moyen de la trame de supervision SREJ

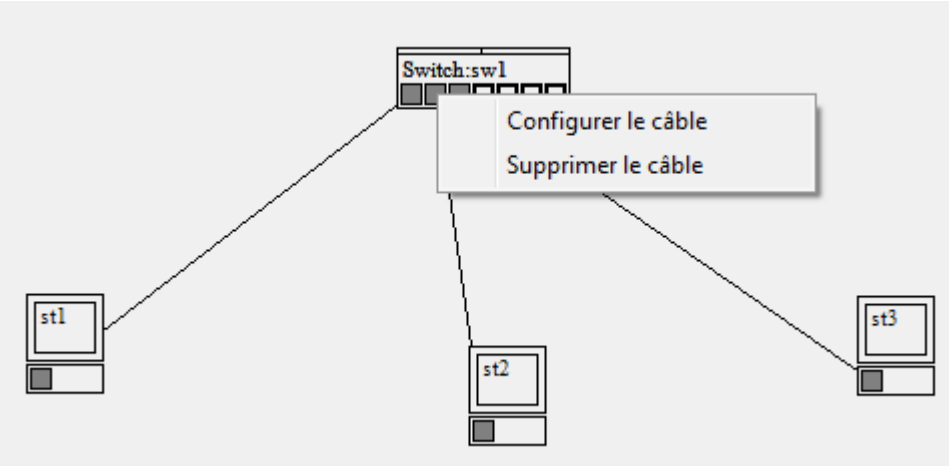

Temporisateur
0 1 2 3 4 5 6 7 8 2 3 4 5 6 7 8 9
Ack 0
Ack 1
Ack 2
Ack 3
Ack 4
Ack 5
0 1 E - - - - - - 2 3 4 5 6 7 8 9
E = Erreur
Trames ignorées par la couche Liaison de Données

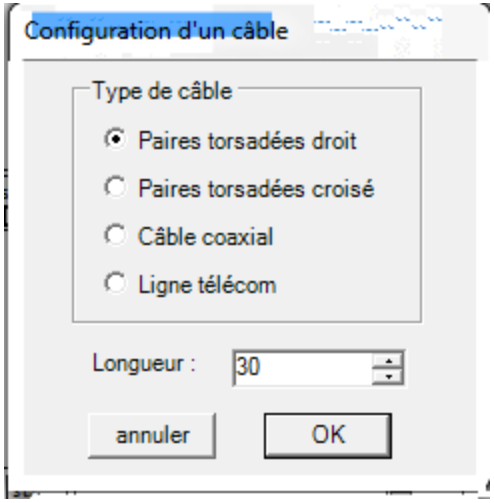

Temporisateur
Temporisateur
Temporisateur
0 1 2 3 4 5 6 7 8 2 3 4 5 6 7 8 9
Ack 0
Ack 1
Ack 8
0 1 E 3 4 5 6 7 8 2 - - - - - - 9
E = Erreur
Trames stockées par la couche Liaison de Données
Les paquets 2 à 8 sont transmis à la couche Réseau

Contrôle de la diffusion de la trame (sur Ethernet)

- Le concentrateur
  - Les cartes réseaux des postes sont reliées aux concentrateurs par l'intermédiaire de câbles en paires torsadées ou en fibre optique
  - Il y a systématiquement deux voies de communication,
    - une voie pour l'émission
    - l'autre pour la réception
  - Si on veut interconnecter deux cartes réseaux sans concentrateur, il faut croiser ces voies

Contrôle de la diffusion de la trame

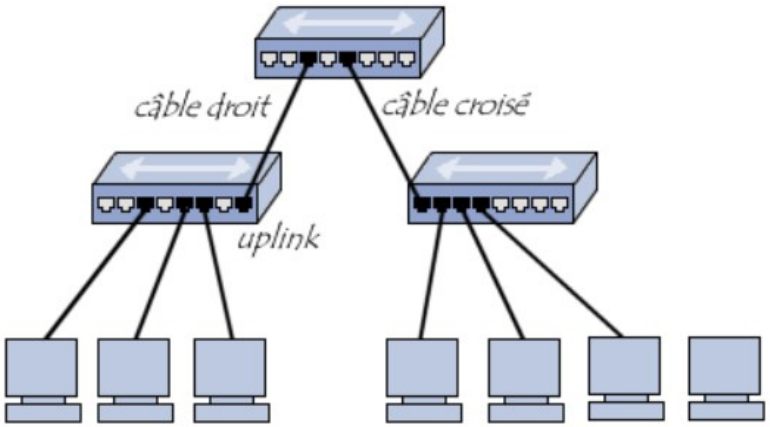




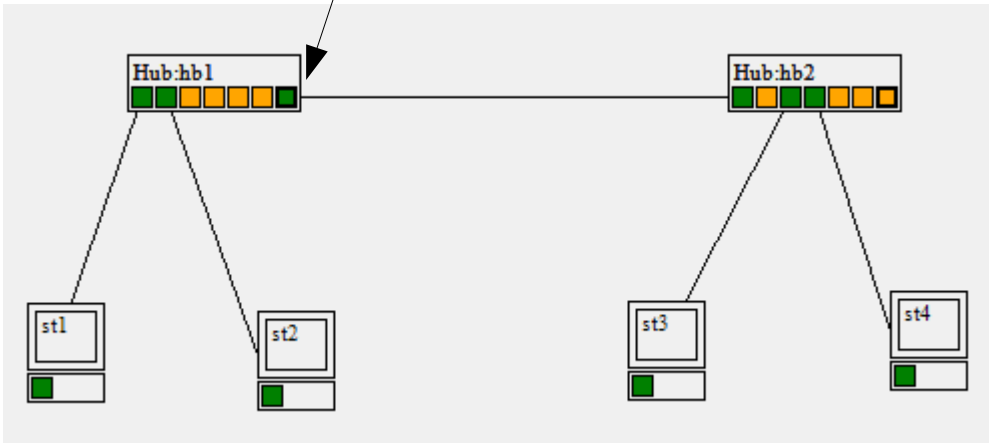
Contrôle de la diffusion de la trame

- Un port de concentrateur ou de commutateur
  - est appelé un port MDI (Medium Dependant Interface)
  - Un port MDI a le comportement suivant :
    - ce qu'il reçoit sur sa paire d'émission il l'émet sur les paires de réception des autres ports
  - Un concentrateur répète ce qu'il reçoit sur un câble d'émission, reçoit le message sur tous les câbles de réception, il **diffuse**
  - Mais seule la carte destinataire du message le lit
    - ceci implique que la carte destinataire doit savoir que le message lui est adressé

- Les concentrateurs sont dotés d'un port spécial appelé "uplink" permettant d'utiliser un câble droit pour connecter deux hubs entre eux
- Il existe également des hubs capables de croiser ou de décroiser automatiquement leurs ports selon qu'il est relié à un hôte ou à un hub



Port uplink sur le simulateur



- *pourquoi faut-il croiser les paires entre deux hubs ?*
  - *car un concentrateur répète sur les paires de réception ce qu'il reçoit sur une paire d'émission*
  - *donc entre deux hubs la paire de réception sur lequel doit répéter un concentrateur doit être la paire d'émission pour l'autre*

Contrôle de la diffusion de la trame

- La trame et l'adresse du destinataire dans la trame
  - Chaque poste sur un réseau a une adresse (appelé adresse MAC)
    - Cette adresse est associée à la carte réseau
  - Le message qui circule sur le réseau (trame) contient l'adresse de l'émetteur et l'adresse du destinataire
  - Mais comme le concentrateur ne connaît pas l'emplacement d'un poste, pour être sûr que le message lui arrive, il l'envoie à tous les postes (principe de la diffusion)
  - Les messages transmis sont découpés pour
    - pouvoir être transportés plus facilement et
    - pour mieux répartir entre les postes le support de communication

Contrôle de la diffusion de la trame

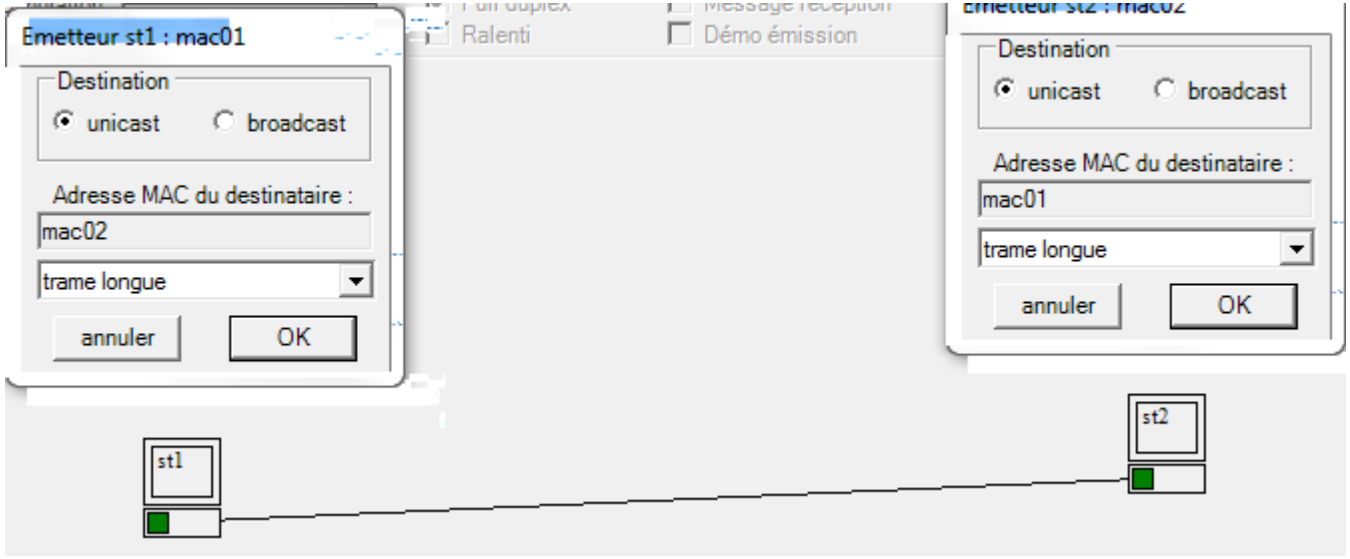
- La méthode d'accès au support : CSMA/CD
  - Le partage d'un même support de communication par tous les postes implique une méthode d'accès à celui-ci
  - En effet, comme dans une communication classique si tout le monde parle en même temps on ne se comprend pas, il faut donc régler les temps de parole
  - Différentes techniques sont possibles
    - La technologie Ethernet se caractérise par la **méthode d'accès CSMA/CD**
    - Si plusieurs cartes émettent en même temps cela provoque une collision

Contrôle de l'accès partagé

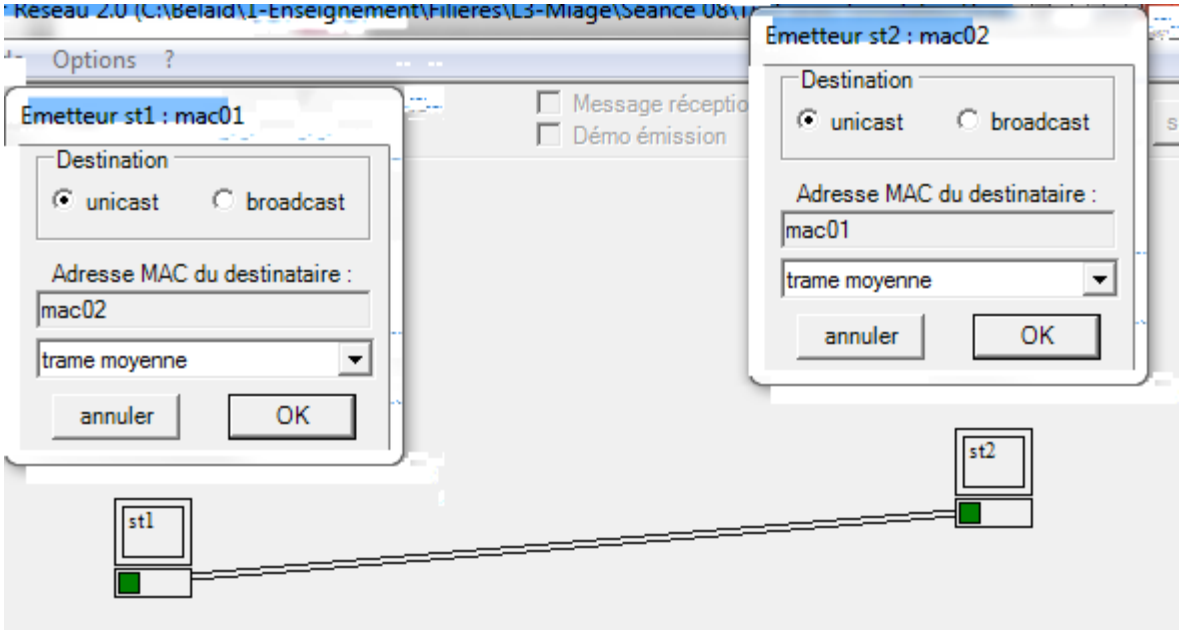
- La détection de collision
  - La carte détecte la collision sur la paire de réception
  - C'est pourquoi la carte ne peut émettre et recevoir en même temps
    - car sa voie de réception est monopolisée par la détection de collision, elle travaille à l'**alternat** (**half-duplex**)
  - Pour pouvoir détecter la collision
    - la carte doit rester à l'écoute jusqu'à ce que la trame ait parcouru l'ensemble du réseau et qu'une collision qui a pu se produire à l'extrémité du réseau se soit propagée en retour jusqu'à elle
    - Cela correspond en fait au délai d'aller / retour d'une trame (**le round trip delay**)

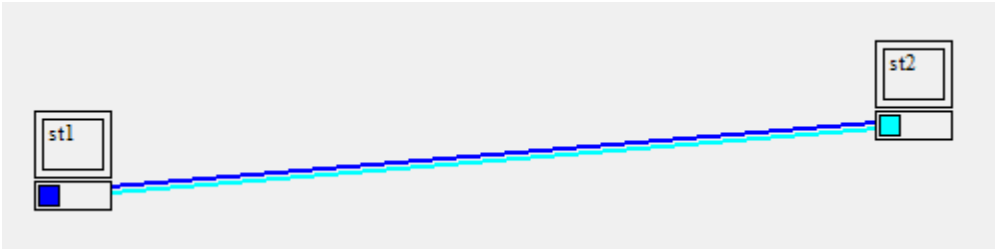
Exemple de collision

- En câble coaxial, se mettre en trame réelle



- Exemple de diffusion en paire torsadée croisée
  - Pour obtenir la paire, configurer le câble en paire torsadée croisée, lancer une trame de chaque poste, alors la paire se double
  - On voit qu'il n'y a pas de collision





Contrôle de l'accès partagé

- Le délai d'aller/retour d'une trame est déterminé par une norme Ethernet
  - Elle est fonction de
    - d'une taille de trame minimale
    - d'un délai de répétition par les répéteurs
    - et d'une longueur maximum de câble
  - Quand les normes ne sont pas respectées,
    - une carte peut ne pas détecter une collision et des trames peuvent ainsi être perdues
  - *Cette norme s'appelle 10BASE-T*
    - *Elle détermine le débit en fonction de la longueur et la nature du câble*

- *10BaseT*
  - *En 10 mbps, la période d'émission d'un bit (bit time) est de 0,1 microsecondes et en 100 de 0,01 ms*
  - *Le standard Ethernet définit un "round trip delay" maximum au sein de chaque domaine Ethernet de 576 périodes bit, ce qui correspond à 57,6 microsecondes pour 10mbps et 5,76 pour 100mbps*
  - *576 correspond à une taille de trame minimum de 64 octets (512 bits) plus 64 bits (7 octets de préambule et un octet de délimiteur de trame)*
  - *Si le temps du signal (émission sur câble plus retardement au sein des répéteurs) dépasse cette valeur, l'équipement émetteur ne peut reconnaître la collision*
  - *Lorsqu'une carte émet une trame en 10 mbps elle écoute pendant 57,6 µs et pendant 5,76 µs en 100mbps, il faut que la collision soit détectée dans ce temps*
  - *Entre deux trames on attend 96 bits time soit 9,6 µs*

- *10BaseT*
  - *A 200 000 km par seconde qui est la vitesse approximative de propagation, on fait 200 m par microseconde soit 0,5 µs pour 100m*
  - *Soit 200 * 57,6 soit 11520 mètres en 10 et 200 * 5,76 soit 1152m en 100*
  - *Donc théoriquement si les câbles le supportent deux stations peuvent être séparées par ces distances*

  - *Mais l'affaiblissement sur les câbles à transmission électrique implique des distances maximums de 500m pour le coaxial épais, 185 m pour le coaxial fin et 100 m pour la paire torsadée*
  - *Les répéteurs sont donc nécessaires mais ils engendrent des délais de transmission supplémentaires*

- *10BaseT*
  - *En 10baseT la règle est la suivante :*
    - *On ne peut pas avoir plus de 4 concentrateurs séparant deux stations en 10 mbps et 500m de câble entre les stations.*
  - *En 100base T on distingue :*
    - *Les répéteurs de classe 1 qui ont un temps de retardement de 0,7 microsecondes.*
    - *Les répéteurs de classe 2 qui ont un temps de retardement de 0,46 microsecondes*
    - *En 100mbps entre deux stations on peut avoir au plus un seul répéteur de classe 1 et 200 m de câble en paire torsadée*
  - *En 100mbps entre deux stations*
    - *on peut avoir au plus deux répéteurs de classe 2 ces répéteurs étant séparés de 5 m au maximum et chaque station étant séparée au plus de 100m du concentrateur*

Contrôle de l'accès partagé

- Le commutateur
  - Avant de réémettre les trames, le commutateur vérifie que le support de communication est libre
  - Un commutateur évite donc les collisions au contraire d'un concentrateur

Contrôle de l'accès partagé

- Le commutateur
  - deux modes de fonctionnement :
    - *Store and forward* : il stocke les trames entièrement avant de les ré emettre
      - Il ne réémet donc pas les trames erronées (CRC "Control Redundancy Check" faux) ou en collision
      - Par contre ces commutateurs sont plus lents et nécessitent des mémoires tampons importantes
    - *On the fly* (appelé aussi *cut through* chez CISCO) : à la volée
      - Notamment les trames en collision et celles dont le CRC est faux. c'est plus rapide mais on propage les trames erronées - notamment les trames en collision et celles dont le CRC est faux

Contrôle de l'accès partagé

- Le commutateur
  - Le commutateur permet de répartir la bande passante d'un réseau
    - La bande passante est le débit d'un réseau
  - En ne diffusant pas à tous les postes mais aux seuls postes concernés par l'échange, le commutateur optimise l'utilisation de la bande passante
  - Ainsi un commutateur 100 mb/s de 12 ports garantira 100 mb/s par port alors qu'un concentrateur 100 mb/s de 12 ports divisera cette bande passante entre tous ses ports

Contrôle de l'accès partagé

- Ethernet commuté
  - L'actualité des architectures réseau est **l'Ethernet entièrement commuté** et donc la disparition progressive des concentrateurs
    - Si on n'a que des commutateurs, il n'y a plus de collision possible
  - Car chaque port forme un mini-segment composé du commutateur et d'une carte ou aucune collision ne peut se produire
  - Dans ce cas pendant l'émission d'une trame, la paire de réception n'est plus monopolisée par la détection de collision et on peut recevoir en même temps, c'est à dire travailler en mode **bidirectionnel** (***full duplex***)

Contrôle de l'accès partagé

- Dans une architecture entièrement commutée
  - On met généralement en œuvre des interconnexions redondantes entre commutateurs pour une plus grande tolérance aux pannes
  - Les liaisons redondantes doivent être invalidées quand elles ne sont pas utiles et validées en cas de rupture d'une liaison
  - Cette gestion de la redondance est prise en charge par le **protocole 802.1d (arbre de recouvrement, en anglais *spanning tree*)**