La liaison PS2 clavier

Scan codes

Le processeur du clavier passe la plupart de son temps à scanner la matrice de touches. S'il trouve qu'une touche a été enfoncée ou relâchée ou tenue appuyée, le clavier enverra un paquet d'information, le scan code, à l'odinateur.
Il y a 2 types différents de scan codes Les "Make Codes" (MC) et les "Break Codes" (BC).
Un MC est envoyé quand une touche est appuyée ou maintenue appuyée.
Un BC est envoyé quand une touche est relâchée.
Un seul MC et un seul BC sont assignés à chaque touche, ce qui fait que le PC peut déterminer exactement ce qui est arrivé à chaque touche en regardant un simple scan code.
L'ensemble des MC et BC pour chaque touche forme un "scan code set".Il n'y a pas de formule simple pour représenter les scan codes pour chaque touche. Si vous voulez savoir quels sont les MC et BC pour une touche donnée, vous devez consulter une table. Il existe 3 tables de scan codes les "scan codes set" 1 , 2 et 3. Tous les claviers modernes ont par défaut l'ensemble "scan codes set 2".

Make Codes, Break Codes et répétition automatique

Lorsqu'on appuie sur une touche, son MC est envoyé à l'ordinateur. Gardez à l'esprit qu'un MC représente seulement une touche sur le clavier, il ne représente pas le caractère imprimé sur cette touche. Celà signifie qu'il n'y a pas de relation définie entre un MC et un code ASCII. C'est à l'ordinateur de transcrire les scan codes en caractères.

Bien que la plupart des scan codes de l'ensemble 2 soit de 1 octet, il y a une poignée de "touches étendues" dont les scan codes sont définis sur 2 ou 4 octets. Leurs MC peuvent être identifiés par le fait que leur premier octet est E0h.

De la même façon qu'un MC est envoyé à l'ordinateur lorsqu'une touche est enfoncée, un BC est envoyé quand une touche est relâchée. Chaque touche a son propre BC. Heureusement, on a pas toujours à utiliser les tables pour représenter le BC d'une touche. Des relations existent entre les MC et les BC. La plupart des BC ont 2 octets, le premier étant F0h et le second étant le MC de la touche. Les BC pour les touches étendues font habituellement 3 octets dont les 2 premiers sont E0h, F0h et le dernier octet est le dernier octet du MC de la touche considérée.

Exemple (scan code set 2)

Key	(Set 2) Make Code	(Set 2) Break Code
"A"	1C	F0,1C
"5"	2E	F0,2E
"F10"	09	F0,09
Right Arrow	E0, 74	E0, F0, 74
Right "Ctrl"	E0, 14	E0, F0, 14

Exemples: quelle séquence de MC et BC doit-elle être envoyée à l'ordinateur pour que le caractère "G" apparaisse dans un traitement de texte?

Etant donné qu'il s'agit d'une lettre majuscule, la séquence d'événements à mettre en place est: appuyer sur "shift", appuyer sur "G", relâcher "G", relâcher "shift". Les scan codes associées à ces événements sont les suivants:

MC pour la touche "shift" (12h), MC pour la touche "G" (34h), BC pour la touche "G" (F0h, 34h), BC pour la touche "shift"(F0h, 12h)

De là les données envoyées à l'ordinateur:

12h, 34h, F0h, 34h, F0h, 12h

Si vous appuyez sur une touche, son MC est envoyé à l'ordinateur. Quand vous appuyez et tenez une touche enfoncée, cette touche devient "typematic" (TM), ce qui signifie que le clavier va continuer à envoyer le MC de cette touche jusqu'à ce que cette touche soit relâchée ou qu'une autre touche soit appuyée. Pour vérifier ceci, vous ouvrez un traitement de texte et vous tenez la touche "A" appuyée. Lorsque vous appuyez, le caractère "a" apparaît immédiatement sur l'écran. Après un court délai, un autre "a" apparaît suivi par une série de "a" jusqu'à ce que vous relachiez le "A" du clavier. Il y a ici deux paramètres importants: le "typematic delay" (TMD) qui est le court délai entre le premier et le second "a" et le "typematic rate" (TMR) qui indique combien de caractères par seconde vont apparaître sur votre écran après le "typematic delay" - le TDM peut varier de 0,25 à 1 sec et le TMR de 2 cps à 30 cps (caractères par seconde). Vous pouvez changer les TDM et TDR en utilisant la commande "Set Typematic Rate/Delay" (0xF3).

Les données TM ne sont pas bufférisées dans le clavier. Dans le cas ou plus d'une touche est enfoncée, seule la dernière touche devient TM. TM se répète et stoppe quand cette touche est relâchée, même si d'autres touches sont appuyées.

Reset

A la mise sous tension ou au reset logiciel, le clavier effectue un test de diagnostic appelé BAT (test d'assurance de base) et charge les valeurs suivantes par défaut:

TDM 500ms
TMR 10,9 cps
Scan Code set 2
Toutes les touches typematic/make/break

Lorsque le clavier entre dans le BAT, il allume ses 3 leds et les éteint lorsque le BAT est fini. A ce moment, un code de fin de BAT soit AA (succès), soit FC (erreur) est envoyé à l'ordinateur. Ce code doit être envoyé de 500 à 750 ms après la mise sous tension. Beaucoup de claviers ignorent les lignes Clock et Data tant que le code de BAT n'a pas été envoyé. De là une condition d'inhibition (Clock à 0) peut ne pas prévenir l'envoi du code de fin de BAT par le clavier.

Ensemble de commandes

Quelques notes concernant les commandes que l'ordinateur peut émettre vers le clavier.

- Le clavier efface son buffer de sortie quand il reçoit une commande
- Si le clavier reçoit une commande invalide, il doit répondre avec "renvoyer" (FE)
- Le clavier ne doit pas envoyer de scan code quand le process d'une commande est en cours.
- Si le clavier est en attente d'un octet argument et qu'il reçoit à la place une commande, il doit supprimer la commande précédente et prendre en compte la nouvelle commande.

Voici quelques exemples de commandes que l'hôte peut envoyer au clavier:

- FF (reset) le clavier répond avec "ack" (FA) et entre dans le mode reset
- FE (resend) le clavier répond en renvoyant le dernier octet envoyé. L'exception à cela est: si le dernier octet envoyé était "resend" (0xFE) Dans ce cas le clavier envoie le dernier non-FE byte. Cette commande est utilisée pour indiquer une erreur dans la réception.
- EE Echo A l'envoi d'une commande d'écho, le clavier devrait répondre avec Echo (EE)
- F0 Set Scan Code Set. A l'envoi de F0, le clavier doit répondre avec ACK (FA) et attendre une autre octet, 01-03 qui détermine le scan code utilisé. L'envoi de 00 renvoie le scan code actuellemnt utilisé.
- F3 Set Typematic Repeat Rate. Le clavier accuse réception avec FA et attend le second octet qui détermine le TMR
- F4 Keyboard Enable - Efface le buffer de sortie du clavier, autorise le scan du clavier, le clavier retourne un ACK (FA)
- F5 Keyboard Disable - reset le clavier, désactive le scan clavier, le clavier retourne un ACK (FA)

Initialisation

Ce qui suit est un exemple de communication entre un PC et son clavier quand il démarre.

```
Keyboard: AA  Self-test passed               ;Keyboard controller init
Host:     ED  Set/Reset Status Indicators
Keyboard: FA  Acknowledge
Host:     00  Turn off all LEDs
Keyboard: FA  Acknowledge
Host:     F2  Read ID
Keyboard: FA  Acknowledge
Keyboard: AB  First byte of ID
Host:     ED  Set/Reset Status Indicators    ;BIOS init
Keyboard: FA  Acknowledge
Host:     02  Turn on Num Lock LED
Keyboard: FA  Acknowledge
Host:     F3  Set Typematic Rate/Delay       ;Windows init
Keyboard: FA  Acknowledge
Host:     20  500 ms / 30.0 reports/sec
Keyboard: FA  Acknowledge
Host:     F4  Enable
Keyboard: FA  Acknowledge
Host:     F3  Set Typematic Rate/delay
Keyboard: FA  Acknowledge
Host:     00  250 ms / 30.0 reports/sec
Keyboard: FA  Acknowledge
```

Le Protocole PS/2 Souris/Clavier

Source : http://www.computer-engineering.org/
Auteur : Adam Chapweske
Dernière Mise à jour : 05/09/03

Information Légale :

Toutes les informations dans cet article sont fournies "telles quelles" et sans garantie.

l'Interface Électrique :

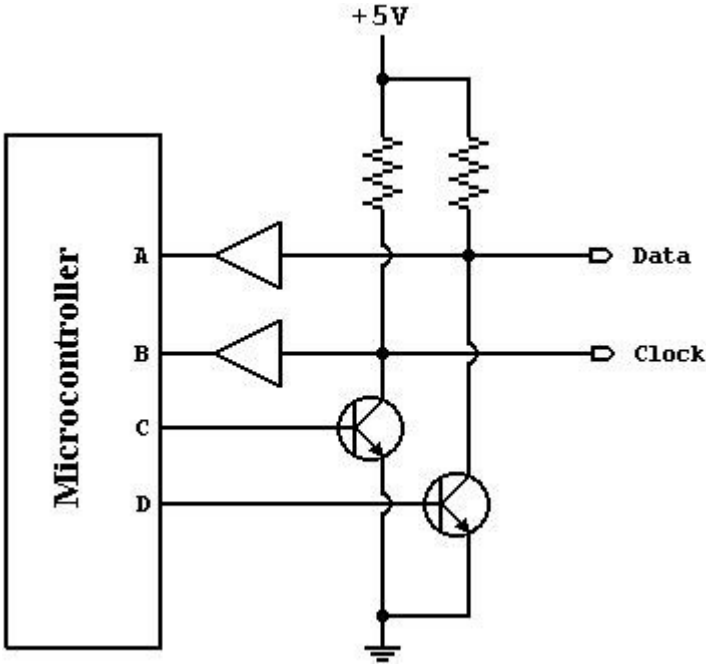
Note : Dans tout ce document, j'emploierai le terme plus général "centre serveur" pour me référer à l'ordinateur -- ou à celui auquel le clavier ou la souris sont reliés -- et le terme "périphérique" se rapportera au clavier ou à la souris.

Vcc/Ground fournissent la puissance au clavier ou à la souris. Le clavier ou la souris ne doit pas tirer plus de 275 mA du centre serveur et on doit veiller aux surtensions transitoires. De telles surtensions peuvent être provoquées "en branchant à chaud" le clavier ou la souris.

En résumé: Caractéristiques de Puissance
Vcc = +4.5V à +5.5V.
Courant maximum = 275 mA.

Les lignes de données et d'horloge sont toutes deux open-collector avec des résistances de pullup à Vcc. Une interface "open-collector" a deux états possibles : impédance basse ou élevée . Dans l'état bas, un transistor met la ligne à la masse. Dans l'état "d'impédance élevée", l'interface agit en tant que circuit ouvert et ne met pas la ligne en état bas ou haut. En outre, une résistance de "pullup" à reliée à Vcc ce qui met la ligne en état haut si le transistor n'est pas actif. La valeur exacte de cette résistance n'est pas très importante (1 ~ 10 kOhms) ; des valeurs élevées ont comme conséquence moins de demande d'énergie et de plus petites un temps de montée plus rapide. Une interface open-collector générale est montrée ci-dessous :

Le schéma 1 : Interface open-collector générale. Des données et l'horloge sont lues sur les broches A et B du microcontrôleur, respectivement. Les deux lignes sont normalement maintenues à +5V, mais peuvent mises à la masse en appliquant "1" sur les broches C et D. En conséquence, A égale D inversée, B égal C inversée.



Note : En regardant les exemples de ce site Web, vous verrez quelques trucs utilisant une interface open-collector avec les microcontrôleurs PIC. J'utilise la même broche pour entrée et sortie, et j'utilise les résistances internes de pullup du PIC plutôt que des résistances externes. Une ligne est mise à la masse en plaçant la broche correspondante en sortie, et en écrivant un "zéro" sur ce port. La ligne est placée à l'état "d'impédance élevée" en plaçant la broche en entrée. Tenant compte des diodes intégrées de protection du PIC et du courant de sortie suffisant, je pense que c'est une configuration valide.

Communication : Description Générale

La souris PS/2 et le clavier mettent en application un protocole série synchrone bidirectionnel. Le bus est "à vide" quand les deux lignes sont hautes (open-collector). C'est le seul état où on permet que le clavier ou la souris commencent à transmettre des données. Le centre serveur a le contrôle final du bus et peut empêcher la communication à tout moment en mettant la ligne d'horloge à l'état bas.

Le périphérique produit toujours un signal d'horloge. Si le centre serveur veut envoyer des données, il doit d'abord empêcher la communication du périphérique par la mise à l'état bas de la ligne d'horloge. Le centre serveur met alors la ligne de données à l'état bas et libère l'horloge. C'est la "Demande d'Envoi" et le signal au périphérique de générer des impulsions d'horloge.

Résumé : États bus
Données = haut, horloge = haut : *État à vide.*
Données = haut, horloge = bas : *Communication Empêchée.*
Données = bas, horloge = haut : *Le Centre serveur Demande-à-Envoyer*

Toutes les données sont transmises un byte à la fois et chaque byte est envoyé sous forme d'une trame se composant de 11-12 bits. Ces bits sont:

- 1 bit de départ. C'est toujours 0.
- 8 bits d'information, bit le moins significatif d'abord.
- 1 bit de parité* (parité impaire = bit de parité à 0 si le nombre de bits est impair).
- 1 bit d'arrêt. C'est toujours 1.
- 1 bit d'acquiescement (communication de centre serveur-à-périphérique seulement)

Le bit de parité est mis à 1 s'il y a un chiffre pair de 1 dans les bits d'information et à 0 s'il y a un nombre impair de 1 dans les bits d'information. Ceci est employé pour la détection des erreurs. Le périphérique doit vérifier ce bit et s'il est incorrect répondre comme s'il avait reçu une commande inadmissible.

Les données envoyées du périphérique au centre serveur sont lues sur le front descendant du signal d'horloge ; les données envoyées du centre serveur au périphérique sont lues sur le front montant . La fréquence de base doit être dans la gamme 10 - 16.7 kilohertz. Ceci signifie que le signal d'horloge doit être haut pendant 30 - 50 micro-secondes et bas pendant 30 - 50 micro-secondes. Si vous concevez un clavier, une souris, ou un émulateur de centre serveur, vous devez modifier ou échantillonner la ligne de données au milieu de chaque créneau. C.-à-d.. 15 - 25 micro-secondes après la transition appropriée d'horloge. Encore une fois, le périphérique génère toujours du signal d'horloge, mais le centre serveur a toujours le contrôle final de la communication.

La synchronisation est absolument cruciale. Chaque fois, le timing que je donne dans cet article doit être suivi exactement.

Communication : périphérique-à-Centre serveur

Les lignes de données et d'horloge sont deux collecteurs ouverts. Une résistance est reliée entre chaque ligne et +5V, ainsi l'état "à vide" du bus est haut. Quand le clavier ou la souris veut envoyer l'information, elle doit d'abord s'assurer que la ligne d'horloge est à l'état haut. Si elle ne l'est pas, le centre serveur empêche la communication et le périphérique doit "bufférisér" les données à envoyer jusqu'à ce que le centre serveur libère l'horloge. La ligne d'horloge doit être à l'état haut de façon continue pendant au moins pendant 50 micro-secondes avant que le périphérique puisse commencer à transmettre ses données.

Comme je l'ai mentionné dans la section précédente, le clavier et la souris emploient un protocole périodique avec des trames de 11 bits. Ces bits sont:

- 1 bit de départ. C'est toujours 0.
- 8 bits d'information, le moins significatif d'abord.
- 1 bit de parité (parité impaire).
- 1 bit d'arrêt. C'est toujours 1.

Le clavier ou la souris écrit un bit sur la ligne de données quand l'horloge est haute et il est lu par l'hôte quand l'horloge est basse. Les schémas 2 et 3 illustrent ceci.

Schéma 2 : communication de périphérique-à-centre serveur. La ligne de données change d'état lorsque la ligne d'horloge est à l'état haut et la donnée est valide lorsque l'horloge est à l'état bas.

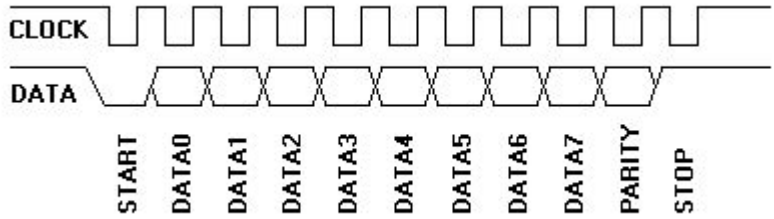


Schéma 3 : Scan Code pour la touche "Q" (15h) envoyée par un clavier à l'ordinateur. La voie A est le signal d'horloge ; La voie B est le signal de données.

La fréquence d'horloge est de 10-16.7 kilohertz. Le temps à partir du front de montée d'une impulsion d'horloge à une transition de données doit être au moins de 5 micro-secondes. Le temps d'une transition de données au front de descente d'une impulsion d'horloge doit être au moins de 5 micro-secondes et pas plus grande que 25 micro-secondes.

Le centre serveur peut empêcher la communication à tout moment en mettant à l'état bas le signal d'horloge pendant au moins 100 micro-secondes. Si une transmission est inhibée avant la 11ème impulsion d'horloge, le périphérique doit interrompre la transmission en cours et se préparer à retransmettre le "gros morceau" des données quand le centre serveur libère l'horloge. Un "gros morceau" des données a pu être un Make Code, Break Code, une identification de périphérique (ID), un paquet de mouvement de souris, etc... Par exemple, si un clavier est interrompu lorsqu'il envoie le deuxième byte d'un code à deux octets de Break Code, il devra retransmettre les deux bytes de ce Break Code et pas simplement celui qui été interrompu.

Si le centre serveur met à 0 le signal d'horloge avant la première transition haut vers bas d'horloge, ou après le front descendant de la dernière impulsion d'horloge, le périphérique n'a pas besoin de retransmettre la donnée. Cependant, si de nouvelles données sont à transmettre, il faudra les "bufférisér" jusqu'à ce que le centre serveur libère l'horloge. Les claviers ont pour cela un buffer de 16 bytes. Si plus de 16 bytes de frappes se produisent, elles seront ignorées jusqu'à ce qu'il y ait place dans le buffer. Les souris stockent seulement le paquet de mouvement le plus récent pour la transmission.

Communication de Centre-à-périphérique :

Le paquet est envoyé un peu différemment dans la communication de centre-à-périphérique...

Tout d'abord, le périphérique PS/2 génère toujours du signal d'horloge. Si le centre serveur veut envoyer des données, il doit d'abord mettre l'horloge et les lignes de données dans l'état "Demande-à-envoyer" comme suit :

- Inhiber la communication en mettant le signal d'horloge à l'état bas pendant au moins 100 micro-secondes.
- Appliquer "Demande-à-envoyer" en mettant à l'état bas la ligne de données, puis libèrer l'horloge.

Le périphérique devrait vérifier cet état à des intervalles n'excédant pas 10 millisecondes. Quand le périphérique détecte cet état, il commence à produire des signaux d'horloge et recueille huit bits de données et un bit d'arrêt sur la ligne de données. Le centre serveur change la ligne de données seulement quand la ligne d'horloge est à l'état bas, et des données sont lues par le périphérique quand l'horloge est à l'état haut. C'est l'opposé de ce qui se produit dans la communication de périphérique-à-centre serveur.

Après que le bit d'arrêt soit reçu, le périphérique donnera accusé réception du byte reçu en mettant la ligne de données à l'état bas et en générant une dernière impulsion d'horloge. Si le centre serveur ne libère pas la ligne de données après la 11ème impulsion d'horloge, le périphérique continuera à produire des impulsions d'horloge jusqu'à ce que la la ligne de données soit libérée (le périphérique produira alors une erreur.)

Le centre serveur peut avorter la transmission avant la 11ème impulsion d'horloge (Bit d'acquiescement) en maintenant l'horloge à l'état bas pendant au moins 100 micro-secondes.

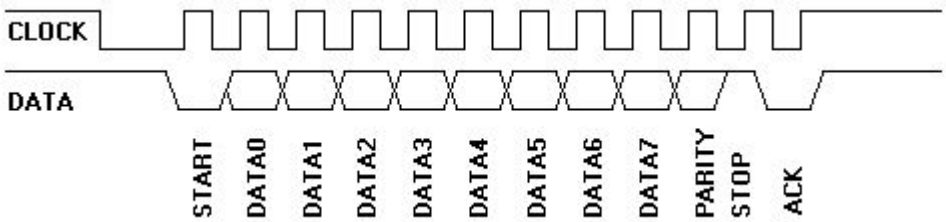
Pour rendre ce processus un peu plus facile à comprendre, voici les étapes que le centre serveur doit suivre pour envoyer des données à un périphérique PS/2 :

1) mettre la ligne d'horloge à l'état bas pendant au moins 100 micro-secondes.
2) mettre la ligne de données à l'état bas.
3) libérer la ligne d'horloge.
4) attendre que le périphérique mette la ligne d'horloge à l'état bas.
5) Set/reset la ligne de données pour envoyer le premier bit d'information
6) attendre que le périphérique mette la ligne d'horloge à l'état haut.
7) attendre que le périphérique mette la ligne d'horloge à l'état bas.
8) répéter les étapes 5-7 pour les sept autres bits d'information et le bit de parité
9) libérer la ligne de données.
10) attendre que le périphérique mette la ligne de données à l'état bas.
11) attendre que le périphérique mette la ligne d'horloge à l'état bas.
12) attendre que le périphérique libère les lignes de données et d'horloge

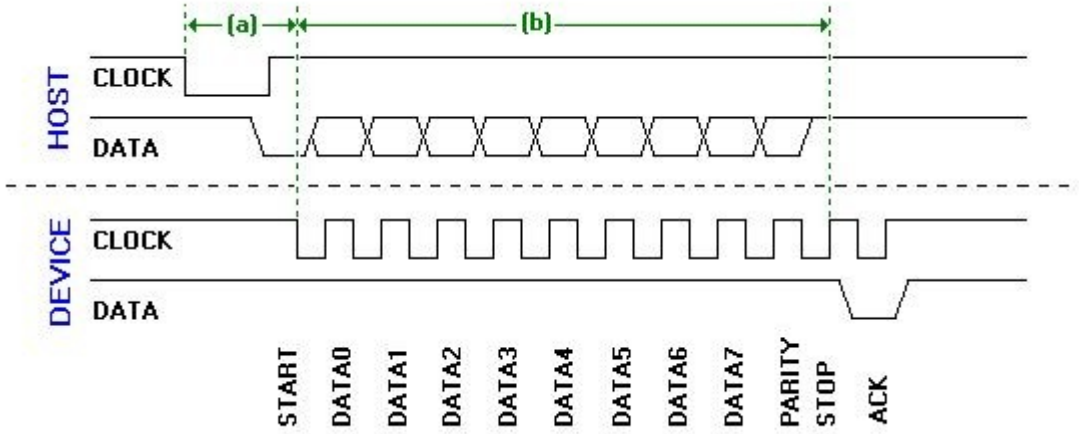
Le schéma 3 montre cela graphiquement et le schéma 4 sépare la synchronisation pour montrer quels signaux sont produits par le centre serveur, quels signaux sont produits par le périphérique PS/2. Notez le changement de la synchronisation pour le bit "ACK" -- les

transitions de données surviennent quand la ligne d'horloge est haute (plutôt que quand elle est basse comme c'est le cas pour les 11 autres bits.)

Schéma 3 : Communication de Centre-à-périphérique.



Le schéma 4 : Communication détaillée de centre-à-périphérique.



En référence au schéma 4, il y a deux quantités de temps que le centre serveur recherche.

(a) est le temps que prend le périphérique pour commencer à produire des impulsions d'horloge après que le centre serveur ait mis initialement la ligne d'horloge à l'état bas, qui ne doit pas être supérieure à 15 ms

(b) est le temps qu'il faut pour que le paquet soit envoyé, qui ne doit pas être supérieur à 2 ms. Si l'un ou l'autre de ces délais n'est pas respecté, le centre serveur doit produire d'une erreur. Juste après que le "ACK" est reçu, le centre serveur peut mettre la ligne d'horloge à l'état bas pour empêcher la communication pendant qu'il traite des données. Si la commande envoyée par le centre serveur exige une réponse, cette réponse doit être reçue pas plus tard que 20 ms après que le centre serveur a libéré la ligne d'horloge. Si ceci ne se produit pas, le centre serveur produit d'une erreur.

*Parité : le mot transmis peut être suivi d'un bit de parité qui sert à détecter les erreurs éventuelles de transmission. Il existe deux parités : la parité paire et la parité impaire. Dans le cas de la parité paire, et pour le mot 10110101 contenant 5 états à 1, le bit de parité sera 1 amenant ainsi le nombre total de 1 à un nombre pair (6). Dans le cas de la parité impaire, le bit de parité aurait été 0 car le nombre total de 1 est déjà impair. L'intérêt de ce rajout est le suivant : si jamais lors de la transmission un état 1 est transformé en état 0 (perturbation du canal par des parasites par exemple) le nombre total de 1 change et donc le bit de parité recalculé par le récepteur ne correspond plus à celui reçu. L'erreur est donc détectée. Evidemment, si deux états à 1 passent à 0, l'erreur ne sera pas détectée mais la probabilité pour que cela arrive est très faible.