Table des matières

Le bus CAN

Introduction

Le **bus CAN** (*Controller Area Network*) est né du besoin de trouver une solution de communication série dans les véhicules automobiles. Il a été initialement développé par la société **Bosch**, au milieu des années 80, puis a fait l'objet d'une normalisation **ISO 11898 et 11519**.

Le bus CAN est un **réseau** à part entière respectant le modèle d'interconnexion des systèmes ouverts (modèle OSI) de l'ISO et il est classé dans la catégorie des **réseaux de terrain** utilisés dans l'industrie pour remplacer la boucle analogique 4/20mA.

Remarque : Le GIE Renault-PSA avec les partenaires comme Sagem, Valeo et autres ont développé le **bus VAN** (*Vehicule Area Network*).

Protocole CAN

Le protocole CAN (*Control Area Network*) est un protocole de **communication série** qui supporte des systèmes temps réel avec un haut niveau de fiabilité.

La norme ISO 11898 spécifie un débit maximum de **1Mbits/s**. La longueur maximum du bus est déterminée par la charge capacitive et le débit (de 20 kbps sur 1 km à 1Mbps sur 40 m).

La structure du protocole du bus CAN possède implicitement les principales propriétés suivantes :

- souplesse de configuration
- fonctionnement multi maître
- détections et signalisations d'erreurs
- retransmission automatique des messages altérés
- déconnexion automatique des nœuds défectueux

Le protocole CAN ne couvre que 2 (ou 3) des 7 couches du modèle OSI : les couches PHYSIQUE et LIAISON et éventuellement la couche APPLICATION.

Le protocole est basé sur le principe de **diffusion générale** : lors de transmission, aucune station n'est adressée en particulier, mais le contenu de chaque message est explicité par une identification reçue par tous les abonnés. Grâce à cet identificateur, les noeuds, qui sont en permanence à l'écoute du réseau, reconnaissent et traitent les messages qui les concernent. Elles ignorent simplement les autres.

Applications

La première utilisation en série du bus CAN date de 1992 avec une Mercedes Classe S. Bien entendu, il y a plusieurs types de réseaux communicants utilisés dans l'industrie automobile (GM et FORD avec le J1850, Renault et Peugeot avec le VAN). Mais la tendance se dirige vers le bus CAN (Peugeot l'utilise dans ses modèles récents 607, 307 et 206 par exemple).

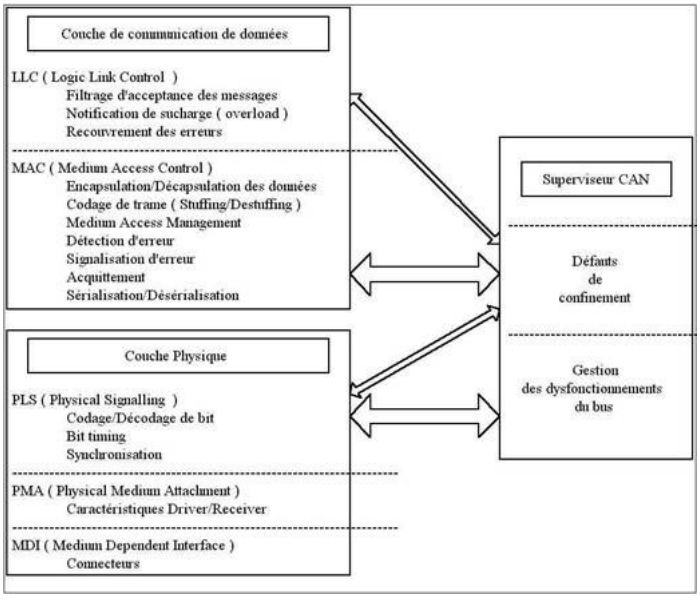
Dans un véhicule, plusieurs réseaux « multiplexés » cohabitent :

- **réseau carroserie** (optiques, rétroviseurs, vitres, ...) à faible débit (32,5 à 62,5 kbps) avec peu d'informations et un temps de réponse demandé de 100 ms
- **réseau inter-systèmes** (ABS, ...) à débit moyen (125 à 500 kbps) avec peu d'informations et un temps de réponse demandé de 10 ms
- **réseau confort** (auto-radio, climatisation, navigation, ...) à débit moyen (125 kbps à 1 Mbps) avec un grande quantité d'informations et un temps de réponse faible de 500 ms
- **réseau diagnostic** (appelé aussi liaison K) à débit faible (10 kbps) avec un grande quantité d'informations et un temps de réponse lent de l'ordre de la seconde)

Autres applications :
Du fait de son efficacité et de sa robustesse éprouvée, le bus CAN est également utilisé dans de nombreuses applications industrielles (systèmes de navigation maritime, ascenseurs, machines agricoles, photocopieurs, systèmes médicaux, industrie textile, machines outils, système de production industrielle, ...)

Modèle OSI

Le bus CAN couvre la couche 2 (LIAISON) et la couche 1 (PHYSIQUE) du modèle OSI :



La sous-couche MAC représente le noyau du protocole CAN. Elle a pour fonction de présenter les messages reçus en provenance de la sous-couche LLC et d'accepter les messages devant être transmis vers la sous-couche LLC. Elle est responsable des fonctions suivantes :

- la mise en trame du message.
- l'arbitrage.
- l'acquittement.
- la détection des erreurs.
- la signalisation des erreurs.

La sous-couche LLC s'occupe quant à elle :

- du filtrage des messages.
- de la notification de surcharge (*overload*).
- de la procédure de recouvrement des erreurs.

La couche Physique définit le signal transmis et a pour pour rôle d'assurer le transfert physique des bits entre les différents nœuds en accord avec toutes les propriétés (électriques et électroniques) du système. Cette couche s'occupe :

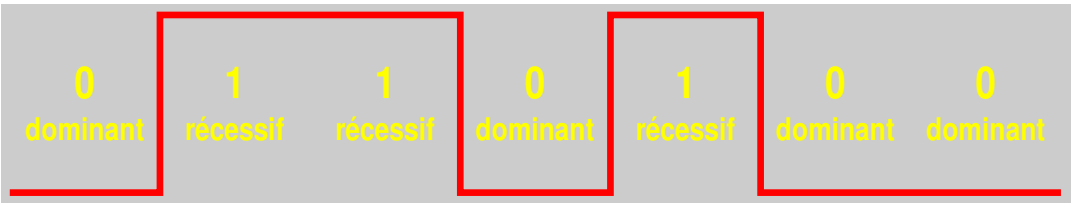
- de gérer la représentation du bit (codage, timing...).
- de gérer la synchronisation bit.
- de définir les niveaux électriques des signaux.
- de définir le support de transmission.

Remarque : la couche 7 (APPLICATION) peut être implémentée sous forme d'un protocole supplémentaire (CANOpen par exemple).

Définitions

Nœud : sous-ensemble relié à un réseau de communication et capable de communiquer sur le réseau selon un protocole de communication (ici le protocole CAN). Chaque nœud peut détecter des erreurs sur un message qui ne lui est pas destiné et en informer les autres nœuds.

Valeurs du bus : le bus peut avoir l'une des deux valeurs logiques complémentaires définies, non pas en 0 et 1 comme d'habitude, mais sous la forme de bit nommé **dominant** ou **récessif**.



Message : chaque information est véhiculée sur le bus à l'aide d'un **message** (trame de bits) de format défini mais de longueur variable et limitée. Dès que le bus est libre, n'importe quel nœud relié au réseau peut émettre un nouveau message.

Routage des informations : des nœuds peuvent être ajoutés au réseau sans qu'il n'y ait rien à modifier tant au niveau logiciel que matériel. **Chaque message possède un identificateur (*identifier*) qui n'indique pas la destination du message mais la signification des données du message.** Ainsi tous les nœuds reçoivent le message, et chacun est capable de savoir grâce au système de filtrage de message si ce dernier lui est destiné ou non.

Trame de données et de requête : une trame de données transporte, comme son nom l'indique, des données. Une trame de requête est émise par un nœud désirant recevoir une trame de données (dans ce cas l'identificateur est le même pour les deux trames).

Débit (bit/s) : le débit peut varier entre différents systèmes, mais il doit être fixe et uniforme au sein d'un même système.

Priorités : en cas de demandes de prise du bus simultanées les identificateurs de chaque message permettent aussi de définir quel message est prioritaire sur tel autre.

Fonctionnement multi-maître : lorsque le bus est libre, chaque nœud peut décider d'envoyer un message. Seul le message de plus haute priorité prend possession du bus.

Arbitrage : le problème de l'arbitrage résulte du fonctionnement multi maître. Si deux nœuds ou plus tentent d'émettre un message sur un bus libre il faut régler les conflits d'accès. On effectue alors un arbitrage bit à bit non destructif tout au long du contenu de l'identificateur. Ce mécanisme garantit qu'il n'y aura ni perte de temps, ni perte d'informations. Dans le cas de deux identificateurs des trames de requête et de données identiques, la trame de données gagne le bus. Lorsqu'un bit récessif est envoyé et qu'un bit dominant est observé sur le bus, l'unité considérée perd l'arbitrage, doit se taire et ne plus envoyer aucun bit.

Canal de liaison simple : le bus consiste en un simple canal bidirectionnel qui transporte les bits. A partir des données transportées, il est possible de récupérer des informations de resynchronisation.

Acquittement : tous les récepteurs vérifient la validité d'un message reçu, et dans le cas d'un message correct ils doivent acquitter en émettant un flag.

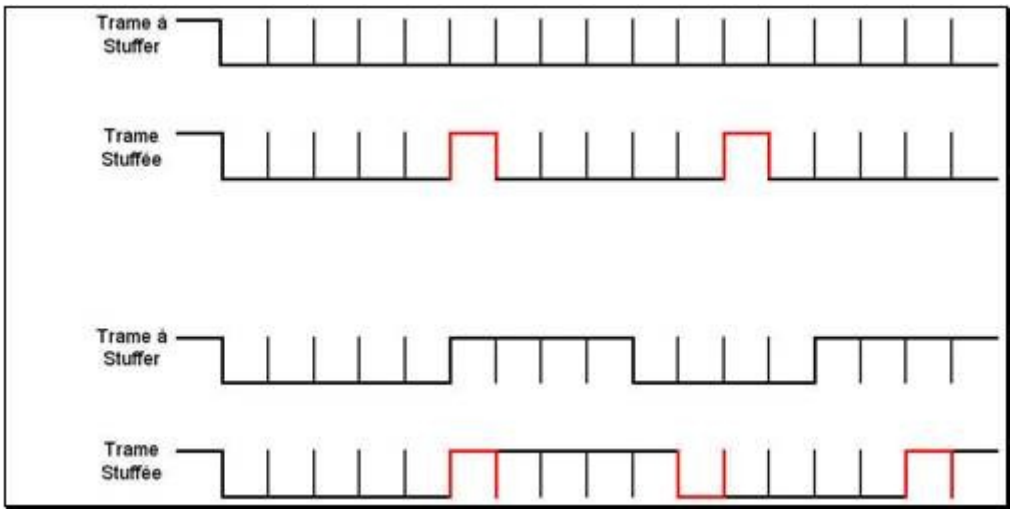
Sécurité de transmission : dans le but d'obtenir la plus grande sécurité lors de transferts sur le bus, des dispositifs de signalisation, de détection d'erreurs, et d'autotests ont été implémentés sur chaque nœud d'un réseau CAN. On dispose ainsi d'un ***monitoring bus*** (vérification du bit émis sur le bus), d'un **CRC** (*Cyclic Redundancy Check*), d'une procédure de contrôle de l'architecture du message et d'une méthode de *Bit-Stuffing*.

Signalement des erreurs et temps de recouvrement des erreurs : tous les messages entachés d'erreur(s) sont signalés au niveau de chaque nœud par un *flag*. Les messages erronés ne sont pas pris en compte, et sont retransmis automatiquement.

Erreurs de confinement : un nœud CAN doit être capable de faire les distinctions entre des perturbations de courtes durées et des dysfonctionnements permanents. Les nœuds considérés comme défectueux doivent passer en mode *switched off* en se déconnectant (électriquement) du réseau.

Mode *Sleep* (sommeil) et Mode *Wake-up* (réveil) : afin de réduire la consommation d'énergie, chaque élément CAN peut se mettre en *Sleep mode*. Dans ce mode il n'y a aucune activité interne au nœud CAN considéré et ses drivers sont déconnectés du bus. La reprise de fonctionnement (mode *Wake-up*) s'effectue lorsqu'il y a une activité sur le bus ou par décision interne à l'élément CAN.

Méthode de *Bit-Stuffing* : Cette méthode consiste, dès que l'on a émis 5 bits de même polarité sur le bus, à insérer un bit de polarité contraire pour casser des chaînes trop importantes de bits identiques. On obtient ainsi dans le message un plus grand nombre de transitions ce qui permet de faciliter la synchronisation en réception par les nœuds. Cette technique est uniquement active sur les champs de SOF, d'arbitrage, de contrôle, de CRC (délimiteur exclu). Pour un fonctionnement correct de tout le réseau, cette technique doit être implémentée aussi bien à la réception qu'à l'émission.



Exercice n°1

Compléter le diagramme suivant en respectant la méthode de *Bit-Stuffing*.
Réponse :

Trame avant *stuffing*

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20				

Trame avec *stuffing*

Couche 2 – Niveau Trame

Le transfert des messages se réalise à l'aide de quatre types de trames spécifiques et d'un intervalle de temps les séparant :

- les **trames de données** : ces trames transportent des données d'un producteur vers des consommateurs.
- les **trames de requête** : ces trames de *polling* sont émises par un maître vers des esclaves pour requérir la transmission d'une trame de données.
- les **trames d'erreurs** : ces trames sont transmises lorsqu'une station détecte une erreur de transmission sur le bus.
- les **trames de surcharge (*overload*)** : ces trames sont émises pour demander un laps de temps supplémentaire entre des trames (de données ou de requête) successives.

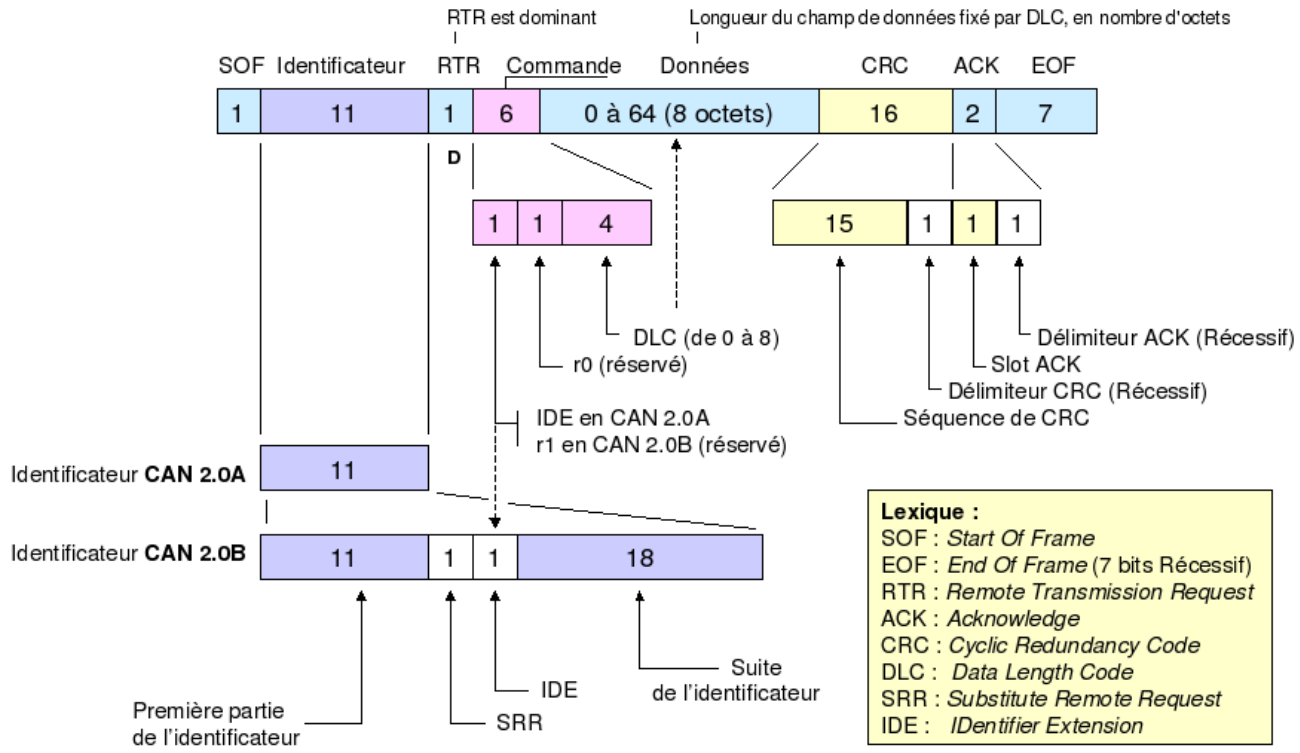
Il existe un **espace inter trame de 3 bits récessifs** entre les trames de données et de requête.
Par ailleurs, il existe deux types de format de trames :

- la **trame standard** (format standard - **CAN 2.0A**) possède un **identificateur de 11 bits**
- la **trame étendue** (format étendu - **CAN 2.0B**) possède un **identificateur 29 bits**

Ce format étendu permet d'obtenir un plus grand nombre de messages et/ou de priorités.
Remarque : Le bus CAN2.0B est compatible avec le CAN2.0A.

Trame de données

Le début de trame (SOF) n'est effectif que si le bus était précédemment au repos (*bus idle*).



Tous les nœuds du réseau se resynchronisent sur le bit de SOF.

Une trame de données se décompose en 7 champs différents :

- le début de trame SOF (*Start Of Frame*) : 1 bit dominant
- le champ d'arbitrage : 11 bits (ID) + 1 bit (RTR) en standard et, 29 bits (ID) + 2 bits (SRR/IDE) + 1 bit (RTR) en étendu
- le champ de commande : 6 bits
- le champ de données : 0 à 64 bits
- le champ de CRC (*Cyclic Redundancy Code*) : 15 bits + 1 bit
- le champ d'acquittement (*Acknowledge*) : 2 bits
- le champ de fin de trame EOF (*End Of Frame*) : 7 bits récessifs

La signification de certains bits :

- RTR : *Remote Transmission Request* bit,
- SRR : *Subsitute Remote Request* bit,
- IDE : *Identifier Extension* bit.

Le bit RTR (*Remote Transmission Request*) permet de déterminer le type de trame :

- RTR = état dominant : trame de données
- RTR = état récessif : trame de requête

Exercice n°2

Déterminer, pour les formats standard et étendu, les longueurs minimum et maximum en bits d'une trame de données circulant sur le bus CAN en ne tenant pas compte du *bit-stuffing*.

Réponses :

Exercice n°3

Déterminer le nombre de capteurs/actionneurs TOR (Tout Ou Rien) différents qu'un noeud peut gérer dans une seule trame de données.

Réponse :

Exercice n°4

Calculer alors le rendement du protocole CAN lorsqu'il émet une trame de données complète.

Réponse :

Exercice n°5

Pour comparaison, déterminer le rendement du protocole *Ethernet* pour la même situation. Quel est alors le protocole le plus efficace pour ce type de situation (notion de bus de terrain ou bus industriel) ?
Réponse :

Exercice n°6

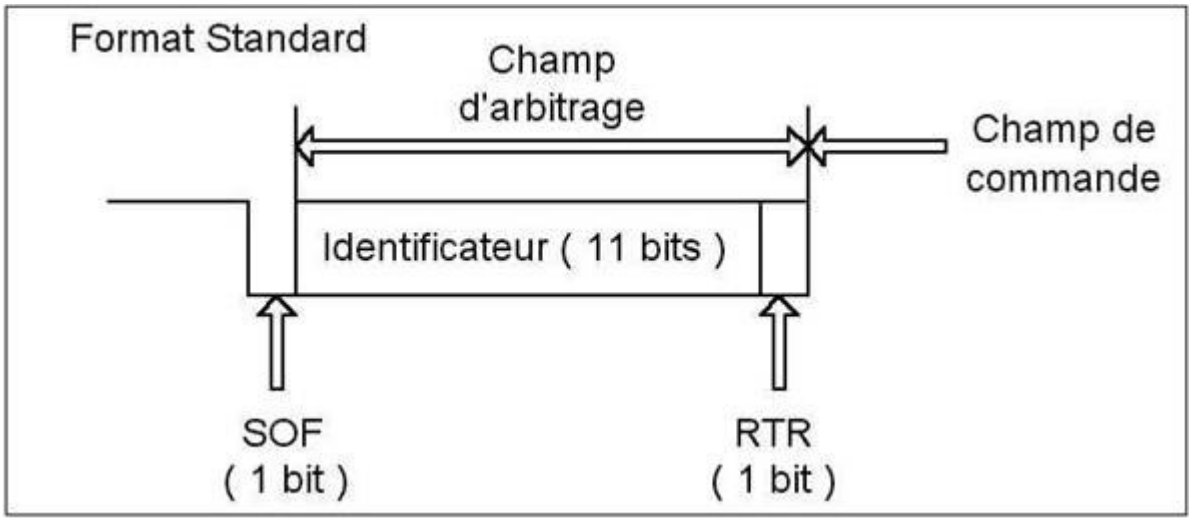
Avec des trames de 60 bits en moyenne, déduire le nombre maximum de trames pouvant circuler sur le bus en une seconde et ce pour le débit maximum (1 Mbits/s).
Réponse :

Exercice n°7

Dans le cas où il y a 4 noeuds identiques qui émettent et reçoivent 16 trames (de 60 bits en moyenne) par secondes , déterminer le taux de charge maximum pour ce bus.
Réponse :

Arbitrage

Dans une trame standard, le champ d'arbitrage est composé des 11 bits de l'identificateur plus un bit de **RTR** (*Remote Transmission Request*) qui est **dominant pour une trame de données** et **récessif pour une trame de requête**.



Comme nous l'avons vu précédemment, l'identificateur permet de router le message vers le bon nœud mais il indique aussi la priorité du message, qui détermine l'assignation du bus lorsque plusieurs stations émettrices sont en concurrences. Dans sa version standard, l'identificateur est codé sur 11 bits.

Les priorités sont attribuées statiquement lors de l'analyse conceptuelle du réseau, au moyen de valeur binaire, et ne peuvent donner lieu à aucune modification dynamique.

Exercice n°8

Déterminer en théorie le nombre de messages qu'il est possible de définir pour une trame standard.
Réponse :

Les bits de l'identificateur sont numérotés de ID_10 à ID_0 (du MSB vers LSB). Par ailleurs les 7 bits les plus significatifs (de ID_10 à ID_4) ne doivent pas tous être récessifs, ce qui réduit le nombre de combinaisons possibles.

Exercice n°9

En tenant compte de ces particularités, déterminer réellement le nombre de messages qu'il est possible de définir pour une trame standard.
Réponse :

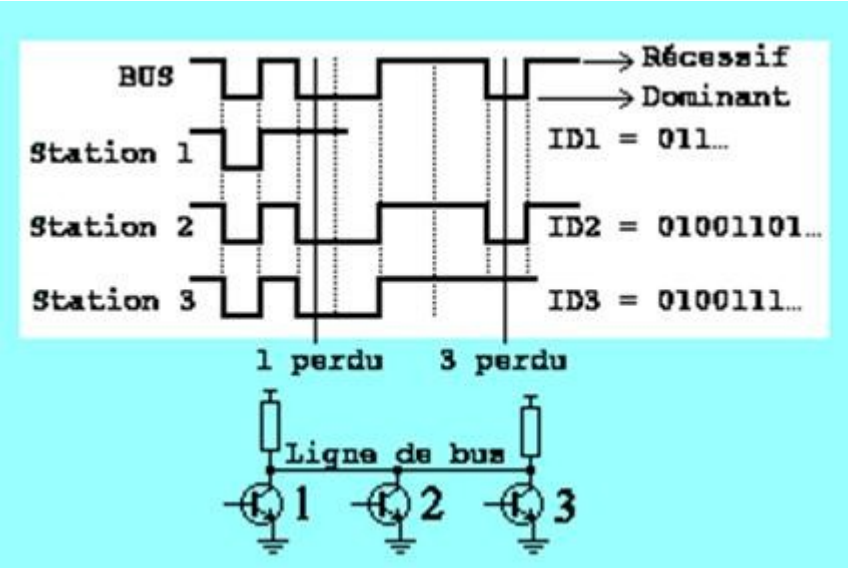
Le procédé d'attribution du bus est basé sur le principe de l'arbitrage bit à bit, selon lequel les noeuds en compétition, émettant simultanément sur le bus, comparent bit à bit l'identificateur de leur message avec celui des messages concurrents. Les stations de priorité moins élevées perdront la compétition face à celle qui a la priorité la plus élevée.

Les stations sont câblées sur le bus par le principe du "ET câblé", en cas de conflit c'est à dire émission simultanée, **la valeur 0 écrase la valeur 1**.

On appelle donc **l'état dominant l'état logique 0, et l'état récessif l'état logique 1**.

Les bits de l'identificateur sont transmis dans l'ordre, de ID_10 à ID_0 (du MSB vers LSB).

Exemple :



Dans l'exemple ci-dessus, les stations 1, 2 et 3 demandent le bus en même temps. Pour les départager, on applique la méthode d'arbitrage et les premières stations a émettre un bit récessif sont exclues et devront attendre que la station qui a pris le bus (la station avec la priorité la plus haute) libère la ligne.

Dans l'exemple la station 1 est exclue en premier puis la station 3. La station 2 à donc la plus haute priorité.

Le bus CAN utilise la méthode d'accès **CSMA/CR** (*Carrier Sense Multiple Access / Collision Resolution*) dont les principes généraux sont les suivants :

- Toutes les stations sont égales
- Chaque station émet quand elle veut (bus libre)
- Les collisions sont acceptées
- Les collisions sont détectées par les stations
- Les collisions sont gérées par le protocole
- Il y a une stratégie d'arbitrage des collisions
- La station qui remporte l'arbitrage continue sa transmission

Exercice n°10

A partir de l'exemple précédent, attribuer les priorités pour les 3 noeuds en fonction de leur rôle.

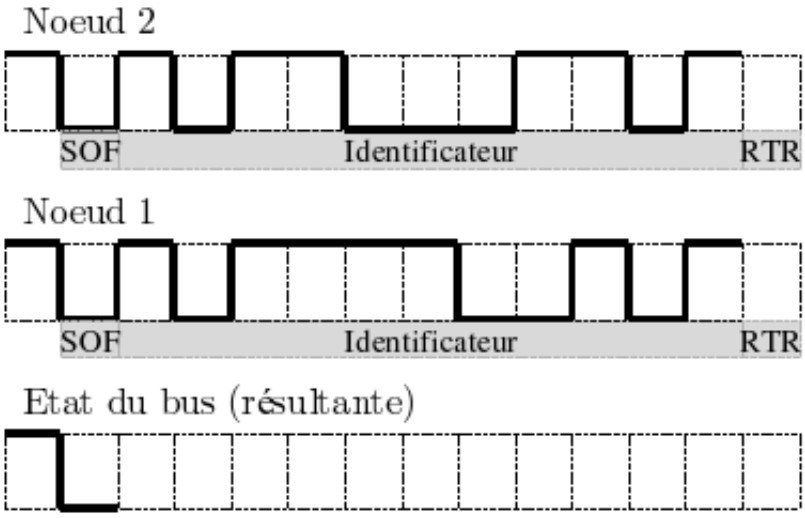
Réponses :

- un capteur de température relié sur le noeud ___
- le capteur d'anti-patinage relié sur le noeud ___
- l'ABS relié sur le noeud ___

Exercice n°11

Compléter le troisième chronogramme (résultante sur le bus) et indiquer le noeud qui a réussi à émettre sa trame.

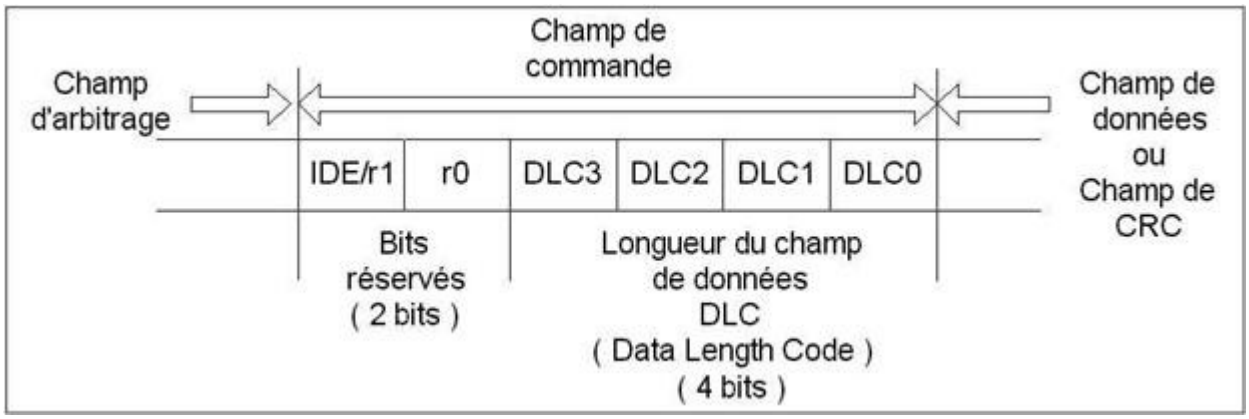
Réponses :



Le noeud ____ gagne le bus.

Champ de contrôle

Le champ de contrôle est composé de 6 bits. Les deux premiers sont des bits de réserve et leur rôle est d'assurer des compatibilités futures ascendantes.



Les quatre derniers bits permettent de déterminer le nombre d'octets de données contenus dans le champ de données. Le nombre d'octets de données ne peut pas excéder la valeur de 8 (soit 64 bits).

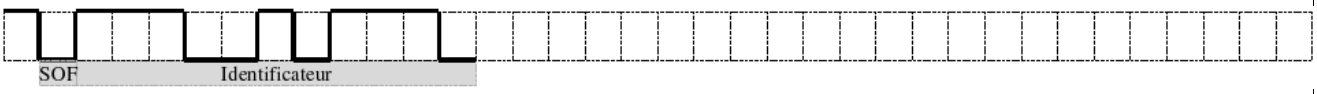
Codage des bits DLC suivant la taille des données en octets (D : bit Dominant et R : bit Récessif) :

Taille des données en octets	DLC (*Data Length Code*)			
	DLC3	DLC2	DLC1	DLC0
0	D	D	D	D
1	D	D	D	R
2	D	D	R	D
3	D	D	R	R
4	D	R	D	D
5	D	R	D	R
6	D	R	R	D
7	D	R	R	R
8	R	D	D	D

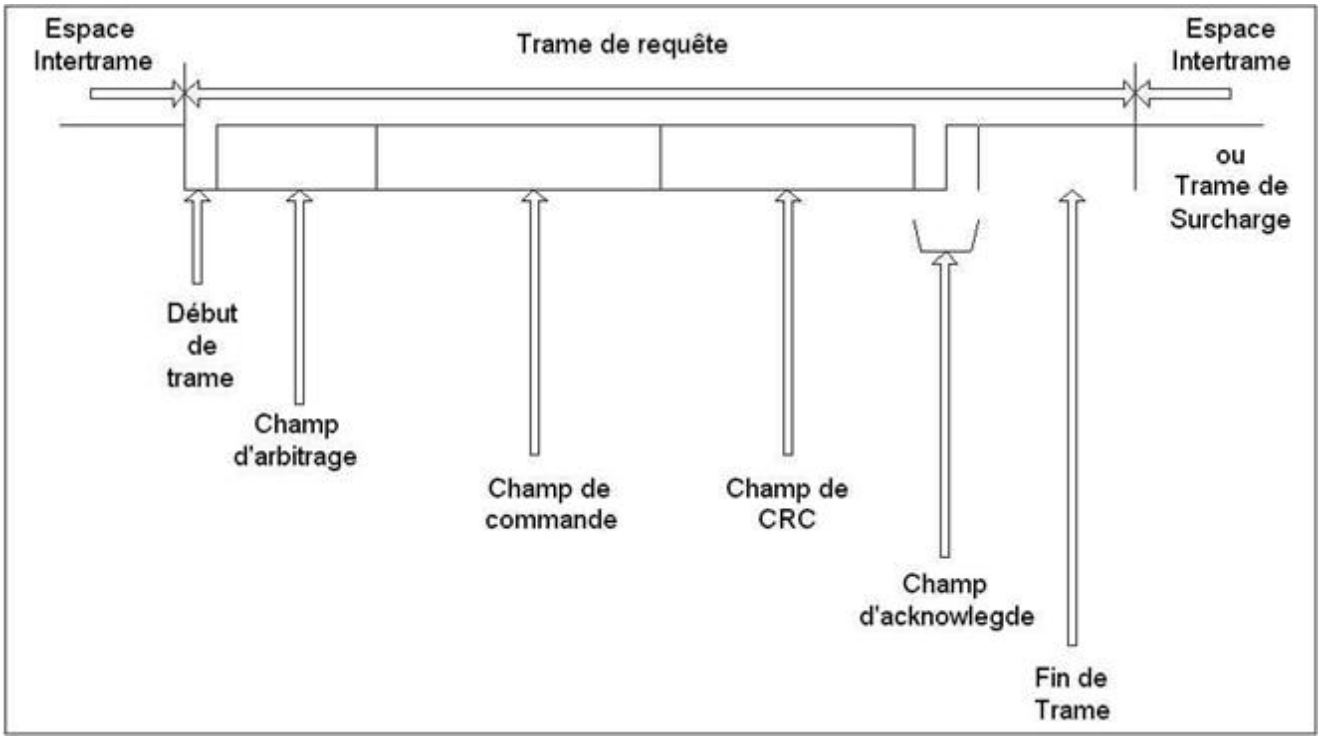
Exercice n°12

Compléter la trame (jusqu'au champ CRC exclu) dans le cas où un noeud émet les données 'B' (0x42) suivi de la valeur 6.

Réponse :



Trame de requêtes



Une trame de requête est constituée de la même manière qu'une trame à deux différences près :

- Le champ de données est **vide**.
- Dans le champ d'arbitrage, **le bit de RTR est récessif**.

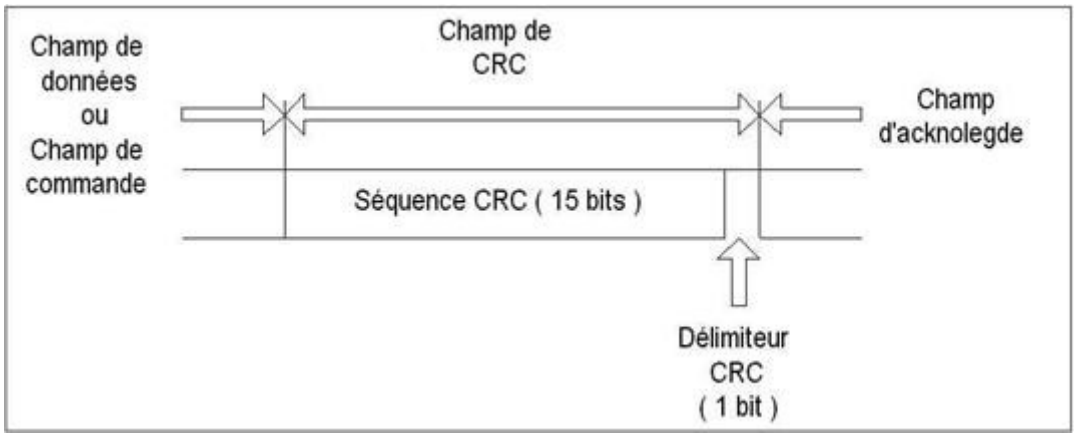
Si un nœud a besoin d'un certain nombre de données, il va émettre une trame de requête dès que le bus sera libre en prenant soin d'indiquer dans le champ de contrôle (DLC) le nombre d'octets de données dont il a besoin.

Exercice n°13

Que se passe-t-il si deux noeuds émettent chacun une trame possédant le même identificateur mais une des deux trames est une trame de données et l'autre une trame de requête.
Conclure sur la priorité entre une trame de données et une trame de requête.
Réponse :

Champ CRC

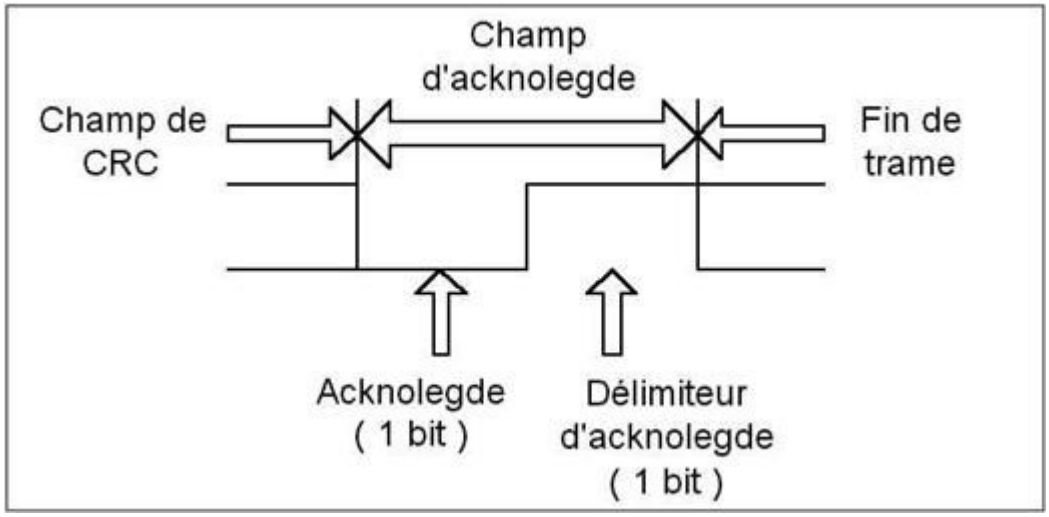
le champ de CRC est composé de 16 bits. La séquence CRC calculée est contenue dans les 15 premiers bits tandis que le dernier bit est un délimiteur de fin de champ de CRC (bit toujours récessif).



Ce champ de CRC permet de s'assurer de la validité du message transmis, et tous les récepteurs doivent s'astreindre à ce procédé de vérification. Seuls les champs de SOF, d'arbitrage, de contrôle et de données sont utilisés pour le calcul de la séquence de CRC.

Champ d'acquittement

Le champ d'acquittement possède 2 bits.



Si aucune erreur n'a été détectée par un nœud, ce dernier émet un bit dominant sinon il émet une trame d'erreur. La station émettrice du message originel doit alors être capable de réagir en fonction de l'émission d'un bit dominant ou non par les autres stations sur le premier bit du champ d'acquittement.

Le second bit est un bit délimiteur d'acquittement qui doit toujours être récessif.

Champ de fin de trame

Ce champ de fin de trame est constitué de 7 bits récessifs, ce qui déroge à la règle de *Bit-Stuffing*.

Traitement des erreurs

Lors de l'émission d'une trame sur le bus, des erreurs de transmission peuvent venir perturber le bon fonctionnement des différents utilisateurs du bus. L'erreur peut venir d'un nœud, et empêcher le réseau de fonctionner correctement. Pour cela, des méthodes de détection d'erreurs de transmissions sont prévues par le protocole CAN.

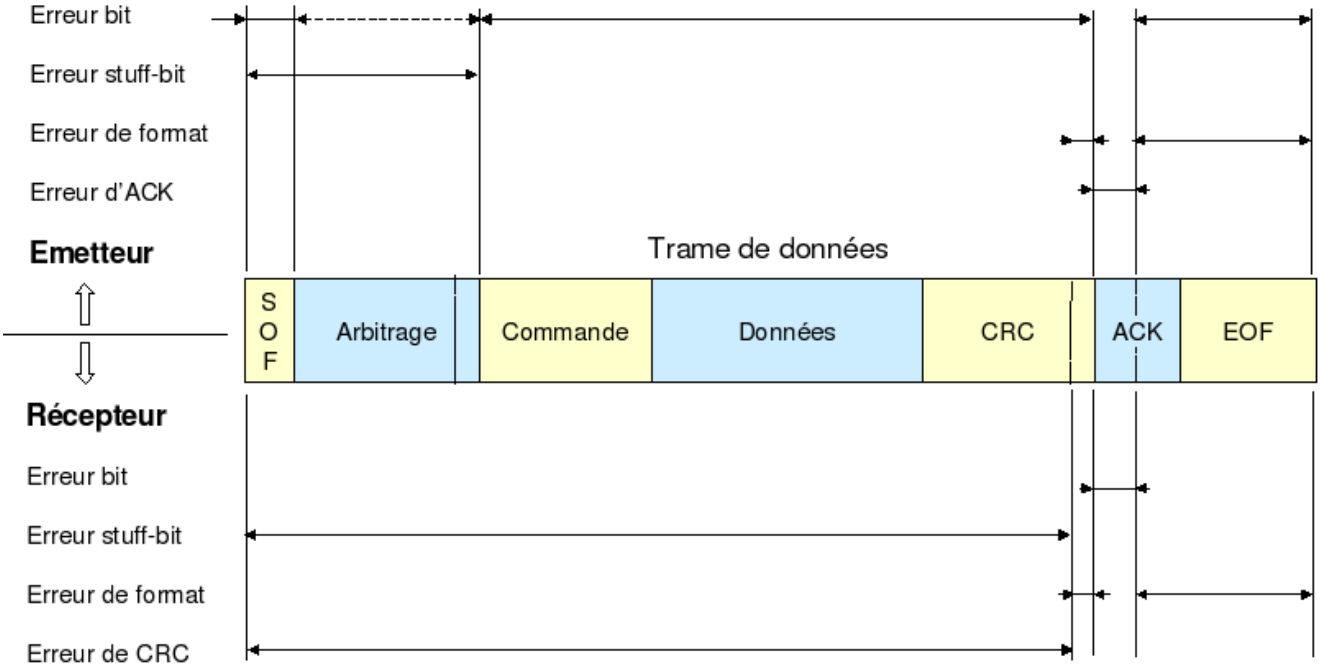
Différents types d'erreur

Le *Bit Error* : Chaque fois qu'un émetteur envoie un bit sur le bus, il vérifie en même temps si le niveau émis sur le bus correspond à celui qu'il désire envoyer en faisant une surveillance du bus. Si le niveau ne correspond pas, il le signale par un *Bit Error*. Sauf si :

- Un bit dominant est envoyé dans le champ d'arbitrage à la place d'un bit récessif. Dans ce cas, le bit dominant signifie simplement une perte d'arbitrage.
- Un bit dominant est envoyé lors de l'*acknowledge slot*, à la place d'un bit récessif.
- Un émetteur envoyant un flag d'erreur passive (bit récessif) et recevant un bit dominant, ne doit pas signaler un *Bit Error*.

L'erreur de *Stuffing* (*Stuff Error*) : Une erreur de *Stuffing* est détectée à chaque fois qu'il y a 6 bits ou plus consécutifs de même signe sur le bus.
L'erreur de CRC (*CRC Error*) : Si la valeur du CRC calculée par le récepteur est différente de celle envoyée par l'émetteur, il y a erreur de CRC.
L'erreur d'*Acknowledge Delimiter* : Une erreur d'*Acknowledge Delimiter* est signalée lorsque le récepteur n'observe pas un bit récessif lors du champ de *Acknowledge Delimiter*. Il en est de même pour le *CRC Delimiter*.
L'erreur de *Slot Acknowledge* (*Acknowledgment Error*) : Une erreur de *Slot Acknowledge* est signalée par l'émetteur s'il ne lit pas un bit dominant lors du champ de *slot acknowledge*.

La gestion des erreurs s'applique sur certains champs, selon les cas :

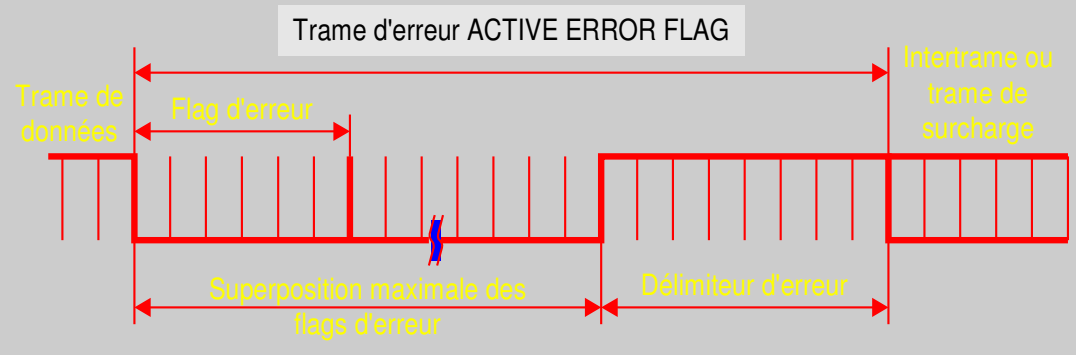


Trame d'erreur (Error Frame)

Si la station détecte une erreur, elle renvoie une trame d'erreur constituée de

- **6 bits à l'état dominant** (*Error Flag*) **suivi de**
- **8 bits délimiteurs à l'état récessif** (*Error Delimiter*).

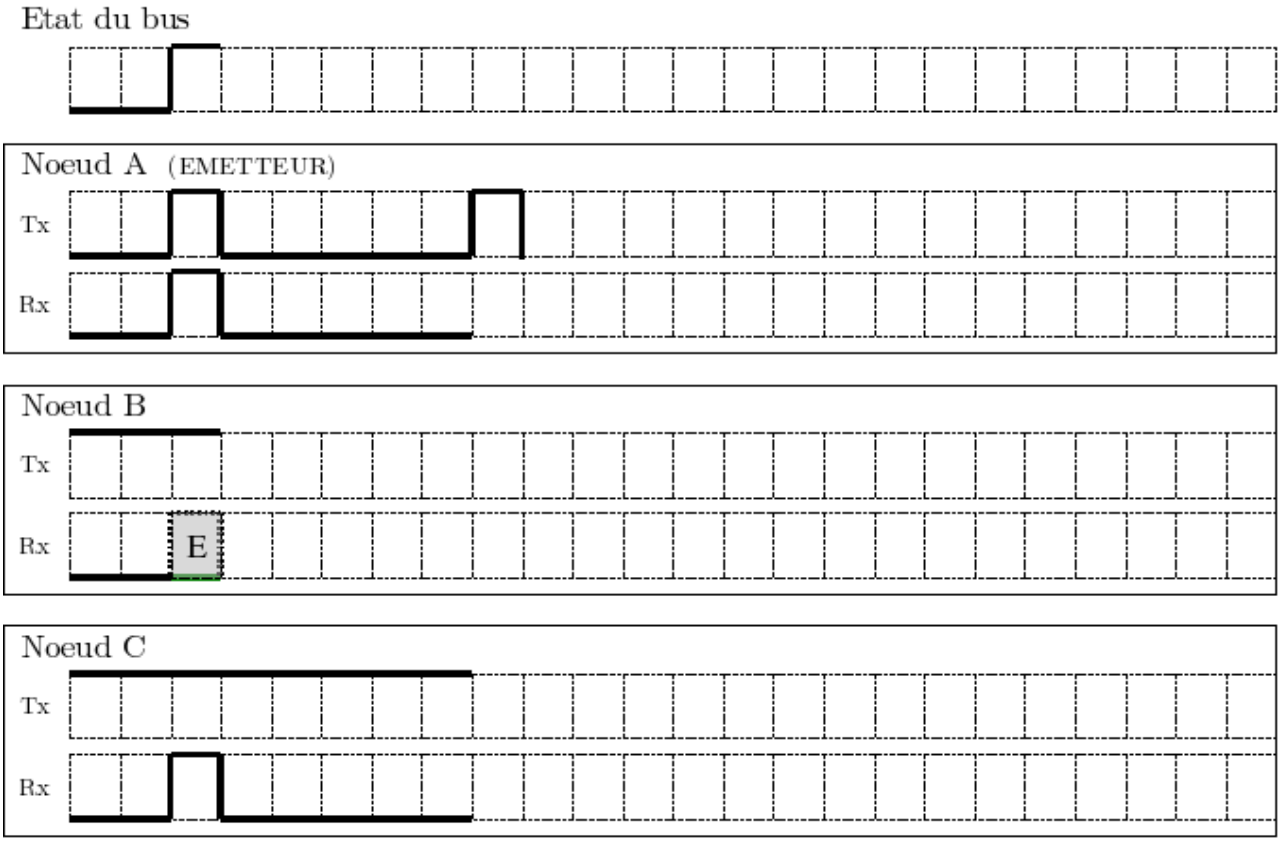
Plusieurs stations peuvent détecter l'erreur et envoyer elles aussi une trame d'erreur : il y a aura superposition de bits d'erreur (un maximum de 6 bits supplémentaires soit 12 bits max pour l'*Error Flag* afin de ne pas bloquer indéfiniment le bus). Les bits formant l'*Error Flag* sont dominants et écrasent donc les données contenues dans la trame de données. Ils provoqueront donc la retransmission de cette dernière.



A l'issue de 8 bits délimiteurs, le bus est libéré (3 bits inter-trame) et les communications peuvent reprendre. Le nœud essaiera à nouveau de transmettre le message.

Exercice n°14

Compléter le diagramme suivant dans le cas où le noeud B détecte un Bit Error (noté E).
Réponse :



Exercice n°15

Lors d'une erreur, le nœud essaiera à nouveau de transmettre le message (retransmission automatique en cas d'erreur). Déterminer le délai en bits qu'il faudra attendre au mieux et au plus (si aucun message de priorité supérieure ne prend la main sur le réseau).
Réponse :

Le confinement d'erreurs

Etant donné qu'une station défectueuse peut perturber le fonctionnement de tout le système (envoi ininterrompu de trames d'erreurs), les noeuds «défectueux» se déconnectent automatiquement (ou limitent leur fonctionnement comme le signalement d'erreurs). Pour mettre en oeuvre le confinement, chaque noeud comporte obligatoirement deux compteurs :

- sur les trames émises (TEC - *Transmit Error Counter*)
- sur les trames reçues (REC - *Receive Error Counter*)

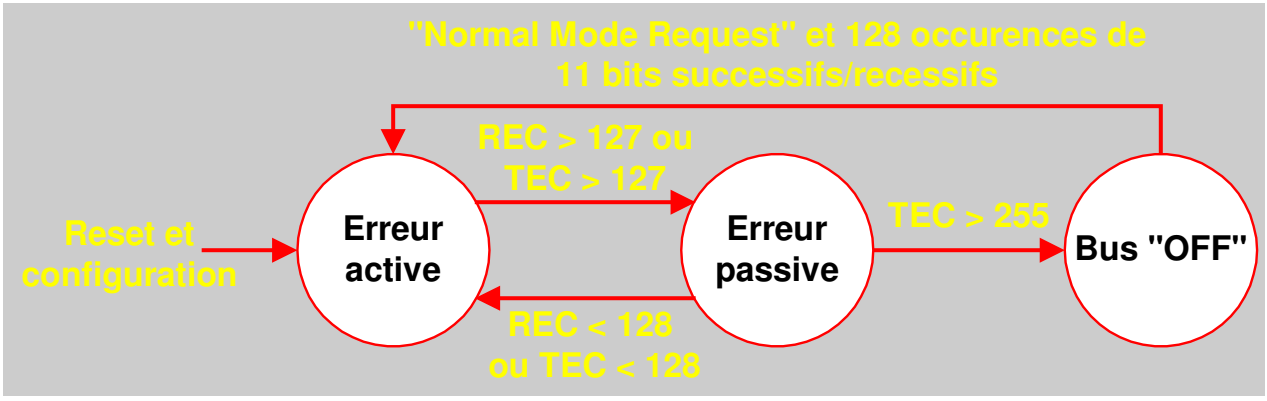
Ces compteurs s'incrémentent et se décrémentent en utilisant un mécanisme de pondération sophistiqué. Pour simplifier, on considérera le fonctionnement suivant (Il existe notamment quelques exceptions à ces règles) :

REC :
- Réception d'une trame corrompue : +1 (jusque 128)
- Réception d'une trame correcte : -1 (si >0)

TEC :
- Emission d'une trame corrompue : +8 (jusque 256)
- Emission d'une trame correcte : -1 (si >0)

Suivant la valeur de ces compteurs, le noeud se trouve dans un des 3 états suivant :

- Etat Erreur-active : fonctionnement normal
- Etat Erreur-passive : émission possible mais 8 bits après que le bus soit libre (temps de réponse !), plus de signalement d'erreurs (le noeud transmet uniquement des *passive error flag* constitués de bits recessifs)
- Etat Bus-off : la station se déconnecte du bus (plus d'émission ni de réception)



Remarque : une condition de "WARNING" est signalée à 96.

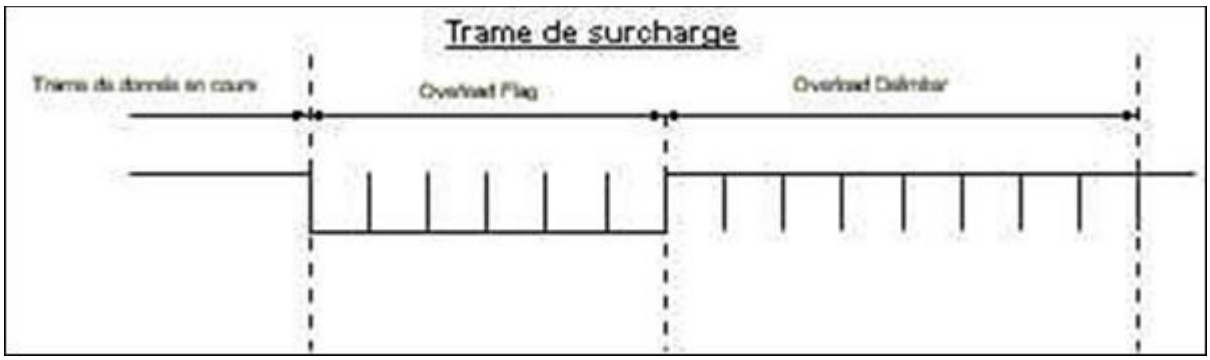
Exercice n°16

Ceci peut durer indéfiniment, si par hasard les erreurs persévèrent, les compteurs d'erreurs augmentent, le circuit produisant l'erreur (ou s'en rendant compte) passe le premier en mode *bus off*. A ce moment là il ne devrait plus y avoir d'erreur et, le message devrait alors bien passer. Et si tous les noeud sont récalcitrants sauf votre noeud, que se passe-t-il ?
Réponse :

Trame de surcharge

La trame de surcharge indique aux autres nœuds qu'une station est surchargée. Elle est formée de deux champs :

- le drapeau de surcharge (*Overload Flag*) avec six bits dominants,
- le délimiteur de surcharge (*Overload Delimiter*) avec huit bits récessifs.



Une trame de surcharge est émise sur le bus si :

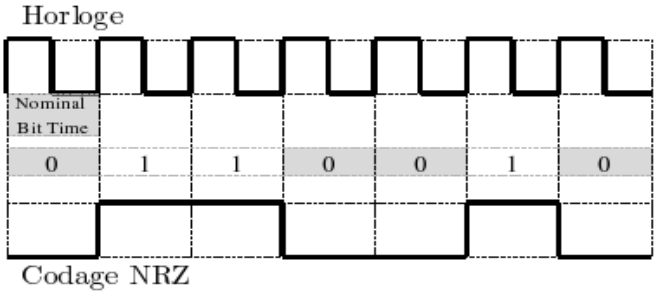
- un bit dominant est détecté durant la période d'inter trame.
- un récepteur n'est pas prêt pour la réception d'une nouvelle trame de donnée ou de requête (retard sur le traitement des informations circulant sur le bus).

Dès qu'une trame de surcharge est émise, les autres nœuds voient sur le bus une suite de six bits dominants qui ne respectent pas la règle du *Bit-Stuffing*. Ils émettent à leur tour une trame de surcharge. Seulement deux trames de surcharges consécutives sont autorisées sur le bus.

Couche 1 – Niveau Bit

Codage

La transmission des bits sur le bus CAN se fait en mode Bande de Base suivant un **codage NRZ** (*Non Return to Zero*). Les transitions des bits s'effectuent sur chaque front montant de l'horloge.

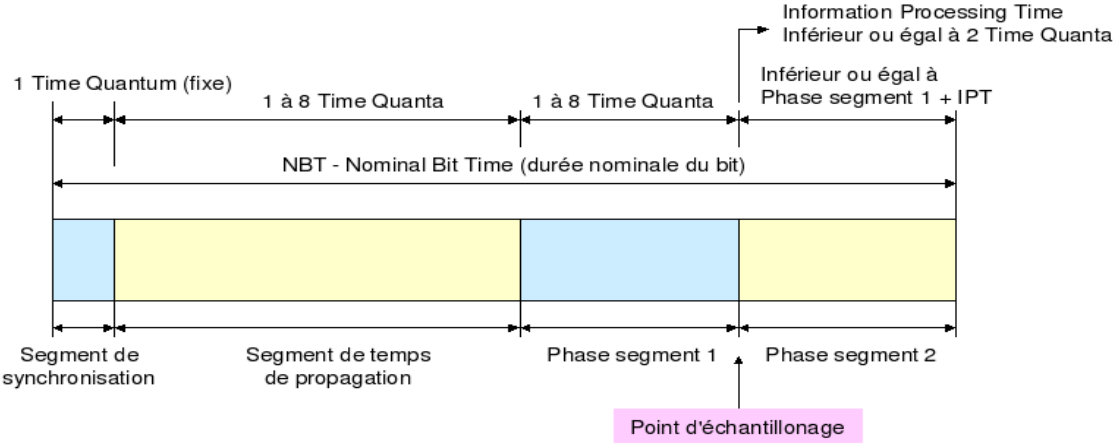


Une période d'horloge correspond à ce que l'on appelle le *Nominal Bit Time*. Le *Nominal Bit Time* représente donc la durée d'un bit sur le bus. Le niveau de tension est maintenu constant pendant la durée d'un bit (NRZ). Chaque station reliée sur le bus doit être cadencée avec le même *Nominal Bit Time* pour pouvoir émettre et recevoir correctement les données circulant sur le bus.

La norme décrit avec précision la composition de ce *Nominal Bit Time* qui est divisé en plusieurs segments :

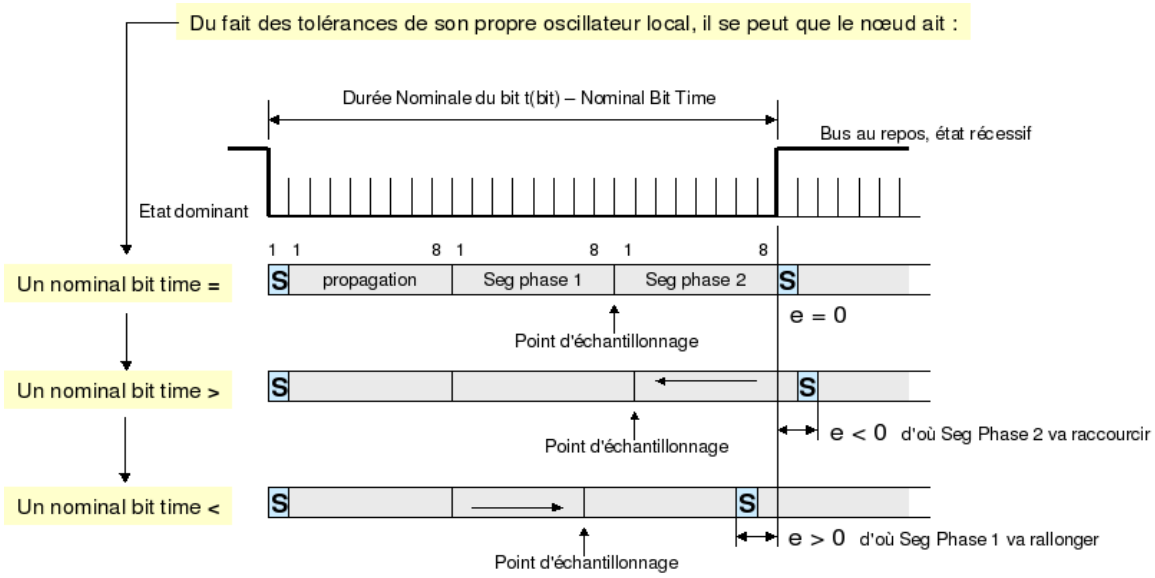
- le segment de synchronisation (SYNC_SEG);
- le segment de propagation (PROP_SEG) sert à compenser les retards temporels physiques dans le réseau. Sa valeur est fixée à deux fois la somme du temps de propagation des signaux dans les lignes du bus et des retards apportés par les entrées-sorties des drivers de lignes ;
- le segment de phase buffer n°1 (PHASE_SEG1),
- le segment de phase buffer n°2 (PHASE_SEG2) ces deux segments sont utilisés pour compenser les erreurs de synchronisation. Ils peuvent être allongés ou raccourcis par la resynchronisation. .

On définit la plus petite base de temps reconnue sur un bus CAN comme étant le *Time Quantum*. Cette base de temps est une fraction de l'horloge de l'oscillateur.



8 < Nombre total de Time Quanta (contenu dans le Bit Time) < **25**

Le point d'échantillonnage (*sample point*) est l'instant où l'état du bus est lu et où la valeur du bit est interprétée. Il se place à la fin de PHASE_SEG1. Sur des bus CAN lents ou très perturbés, on pourra prendre trois points d'échantillonnage.



Pour corriger les dérives du *Nominal Bit Time*, il faut placer des butées que la dérive de la période ne pourra pas dépasser. La butée en question s'appelle le **RJW** : *Resynchronisation Jump Width*. Le RJW est une variable entière programmée à une valeur comprise entre 1 et le minimum de (4, segment de phase1). La valeur est mise dans le registre du circuit lors de l'initialisation et ne change pas en cours de fonctionnement.

Remarque : il faudra donc consulter la documentation du circuit mise en oeuvre pour assurer sa configuration. On devra :

- définir une base pour la vitesse de transmission et un temps de synchronisation (ces temps doivent être égaux sur tous les systèmes d'un même réseau)
- définir la durée de chaque bit, le point d'échantillonnage et le nombre d'échantillonnages.

Exemple : les registres du SJA1000 (*nominal bit time* = $t_{SYNCSEG} + t_{TSEG1} + t_{TSEG2}$)

Registre Bit Timing 0 (BTR0)

SJW1	SJW0	BRP (6 bits)

SJW *Resynchronisation Jump Width* (de 0 à 3) $t_{SJW} = t_{SCLK} \times (SJW + 1)$
C'est le nombre de TQ que le bit time sera rallongé ou raccourci en resynchronisation

BRP *Baud Rate Prescaler* (0 à 63) $tq = 2 \times t_{SCLK} \times (BRP + 1)$

Registre Bit Timing 1 (BTR 1)

SPL (1 bit)	TSEG2 (3 bits)	TSEG1 (4 bits)

SPL *Sampling Mode* 0 = 1 point d'échantillonnage (rapide)
1 = 3 points d'échantillonnage (lent ou parasité)

TSEG1 2 à 15 TQ $t_{TSEG1} = t_{SCLK} \times (TSEG1 + 1)$

TSEG2 1 à 7 TQ $t_{TSEG2} = t_{SCLK} \times (TSEG2 + 1)$

$t_{SYNCSEG} = 1 \times t_{SCLK}$

Exercice n°17

Le circuit contrôleur CAN SJA1000 à 16 Mhz a été configuré avec les valeurs suivantes : BTR0 = 0x43 et BTR1 = 0x2F. Déterminer le débit de ce bus CAN.
Réponse :

Longueur et débit

La longueur du bus dépend des paramètres suivants :

- Le délai de propagation sur les lignes physiques du bus.
- La différence du quantum de temps défini précédemment, du aux différences de cadencement des oscillations des nœuds.
- L'amplitude du signal qui varie en fonction de la résistance du câble et de l'impédance d'entrée des nœuds.

Remarques : la longueur du bus diminue lorsque le débit augmente. Au delà d'une certaine distance (supérieure au km), il faudra mettre en place des répéteurs.

Débit	Longueur	Longueur d'un bit
1 Mbit/s	30 m	1 µs
800 kbit/s	50 m	1,25 µs
500 kbit/s	100 m	2 µs
250 kbit/s	250 m	4 µs
125 kbit/s	500 m	8 µs
62,5 kbit/s	1000 m	16 µs
20 kbit/s	2500 m	50 µs
10 kbit/s	5000 m	100 µs

Caractéristiques électriques

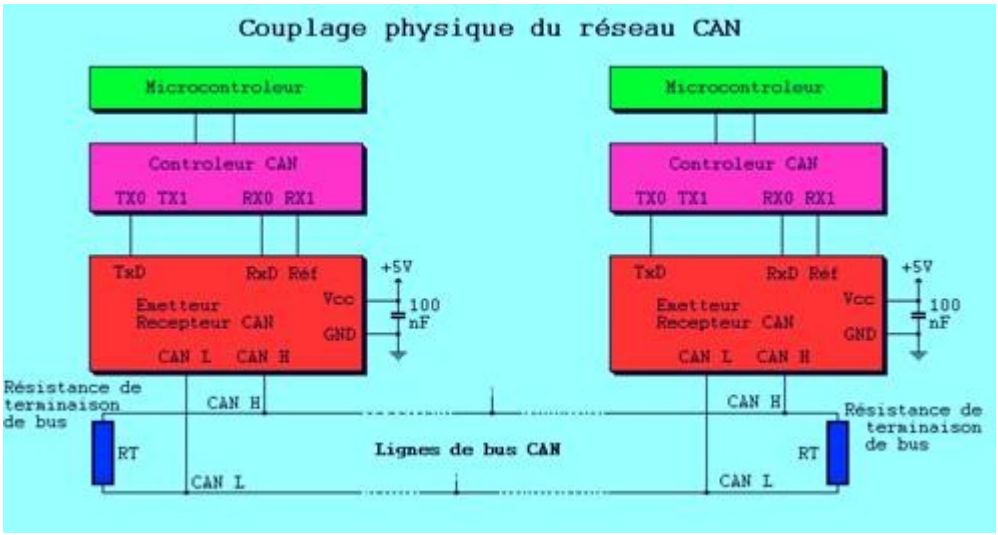
La transmission des données est effectuée sur **une paire filaire différentielle**. La ligne est donc constituée de deux fils :

- CANL (CAN LOW) -> broche 2 d'un connecteur DB9
- CANH (CAN HIGH) -> broche 6 d'un connecteur DB9

Le CAN est un bus de terrain, soumis à des parasites importants. La transmission en paire différentielle permet de s'affranchir de ces problèmes : l'immunité électromagnétique (EMI) est assurée car les deux lignes sont toutes deux affectées de la même manière par la perturbation.

Le tableau ci-dessous résume les principales différences entre les deux types de bus:

Paramètres	*CAN low speed*	*CAN high speed*
Débit	125 kb/s	125 kb/s à 1 Mb/s
Nombre de nœuds sur le bus	2 à 20	2 à 30
Courant de sortie (mode émission)	> 1 mA sur 2,2 kΩ	25 à 50 mA sur 60Ω
Niveau dominant	CAN H = 4V CAN L = 1V	$V_{CAN\ H} - V_{CAN\ L} = 2V$
Niveau récessif	CAN H = 1,75V CAN L = 3,25V	$V_{CAN\ H} - V_{CAN\ L} = 2,5V$
Caractéristique du câble	30 pF entre les câbles de ligne	2*120 Ω
Tensions d'alimentation	5V	5V



***Transceiver* CAN** : 80C250 de Philips ou MCP2551 de MicroChip

Bibliographie

Le bus de terrain CAN de Patrice Kadionik :
http://www.enseirb.fr/~kadionik/formation/canbus/canbus.html
Le réseau CAN et le protocole CANOpen : http://www.a2v.fr/program/canopen.htm
Gestion du bus CAN : http://www.oberle.org/can-index.html
Introduction au bus CAN : http://edelaunay.chez.tiscali.fr/buscan.htm
etc ...