Sécurité des Systèmes Industriels

Les bus CAN

Maxime Puys

22 mars 2025

CLERMONT AUVERGNE INP Isima

Historique et utilisations

Contexte 1/2

Le besoin de modernisation dans l'automobile :

- **Depuis les années 1960 :**
  - Croissance constante de la longueur des câbles dans les véhicules.
  - **1995 :** Plus de **2000 mètres** de câbles et environ **1800 connexions** dans une voiture moyenne.
  - Menaces sur la **fiabilité** et la **sécurité** des systèmes.
- **Normes strictes :**
  - Réduction de la pollution.
  - Consommation énergétique optimisée.
  - Multiplication des capteurs et actionneurs intelligents pour répondre aux exigences des normes.

Contexte 2/2

L'explosion des systèmes embarqués :

- **Besoins de sécurité accrue :**
  - Systèmes ABS, ESP, Airbags.
- **Demande croissante de confort :**
  - Mémorisation des réglages.
  - Climatisation régulée par passager.
  - Systèmes de navigation embarqués.
- **Conséquence :**
  - Multiplication des câbles et connexions, rendant les systèmes complexes et coûteux.

Pourquoi le CAN ?

Le besoin d'un système de communication fiable :

- **Contexte initial :** Complexité croissante dans les véhicules avec de multiples capteurs et actionneurs.
- **Avant CAN :** Utilisation de câblages point à point.
  - Réduction de la fiabilité.
  - Augmentation des coûts et du poids.
- **Solution :** Bus CAN pour centraliser et structurer les communications.

L'Invention du Bus CAN

Une réponse signée Bosch :

- **Développement initial :**
  - Années **1980**, par la société **Bosch**.
  - Objectif : Réduire la complexité grâce à une solution de **multiplexage des informations**.
- **Lancement :**
  - Première démonstration officielle du **Bus CAN** en **1983**.
  - Normalisation dans les années suivantes (ISO 11898).

Avec CAN vs. Sans CAN

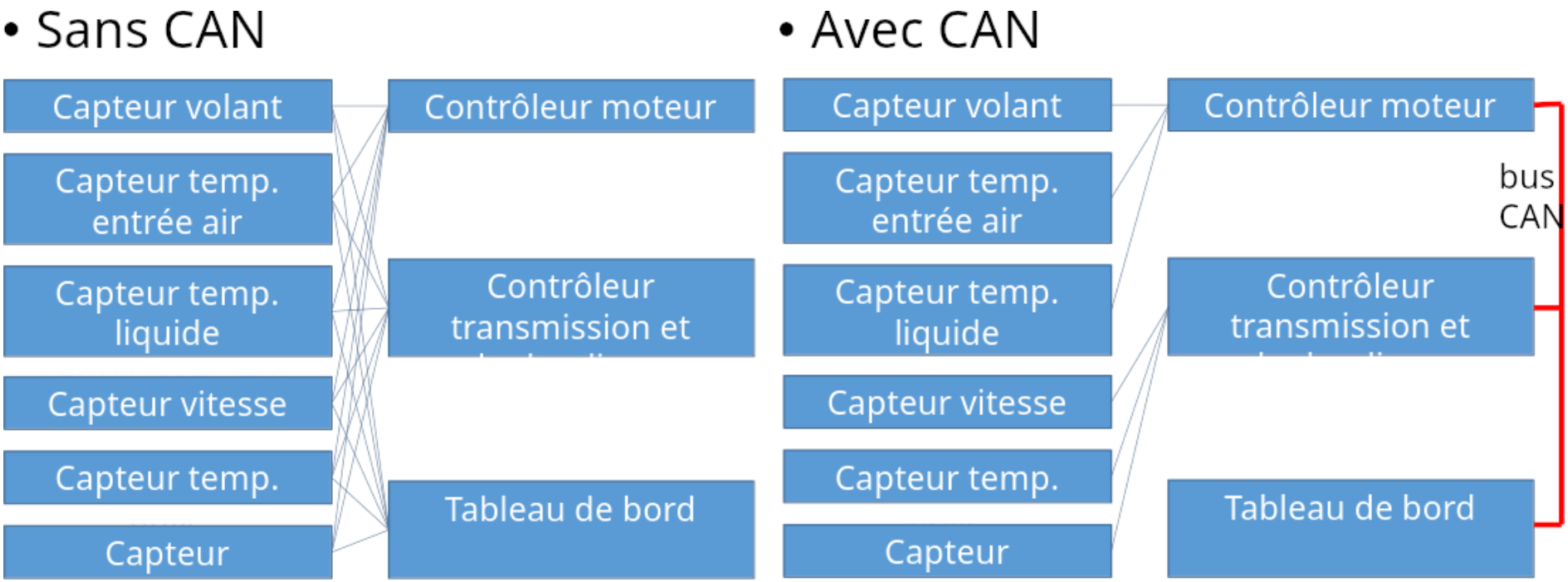

Figure – Avec ou sans CAN ?

Sans CAN – Une Renault 4L (1961)

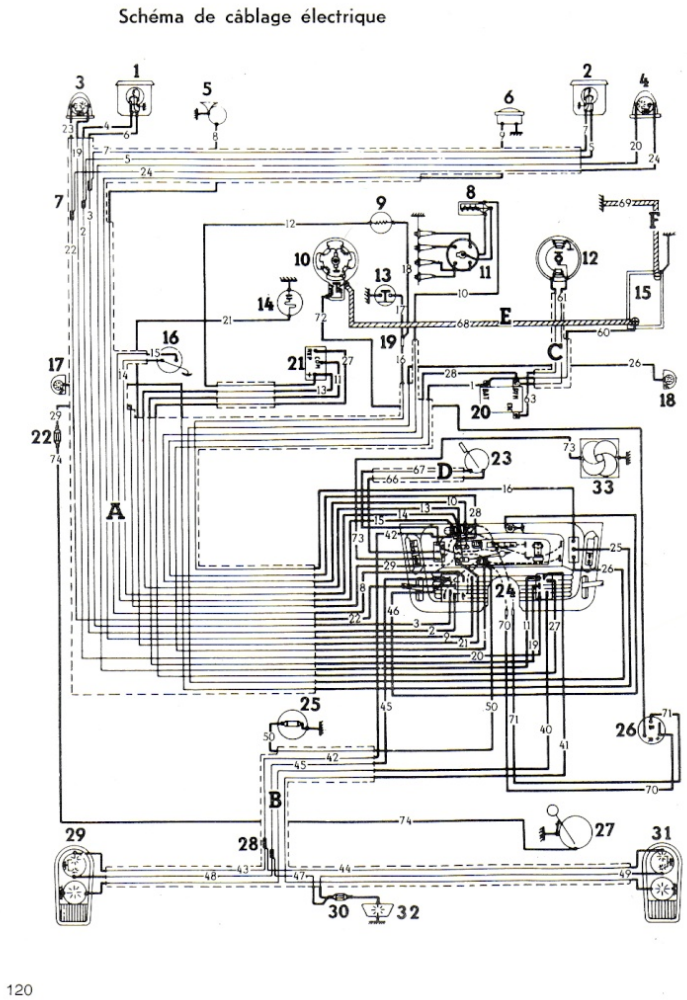

Figure – https://www.r4-4l.com/forum/viewtopic.php?id=12923

Avec CAN – Une Citroen XM (1989)

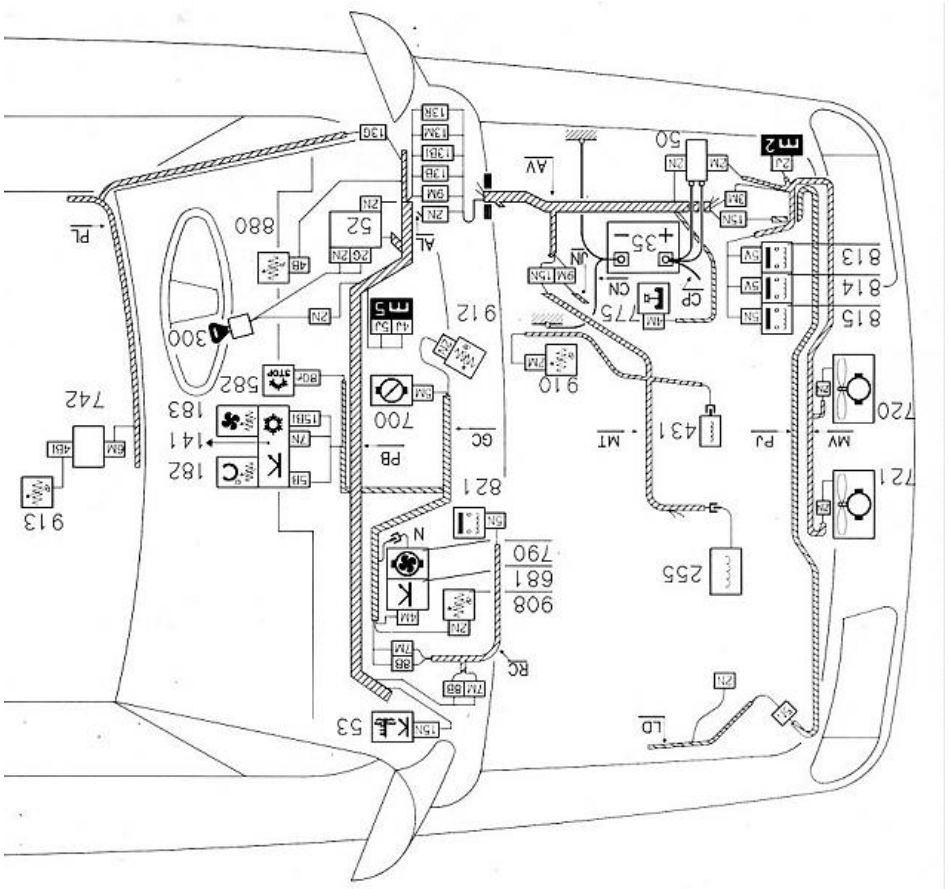

Figure – https://www.planete-citroen.com/topic/22676-sch%C3%A9ma-%C3%A9lectrique/

Historique du CAN

L'Historique du bus CAN :

- **1980** : Développement initial par Bosch.
- **1983** : Première démonstration officielle au SAE Congress.
- **1993** : Standardisation par l'ISO (ISO 11898).
- **2003** : Introduction de CAN FD (Flexible Data-Rate) pour des performances améliorées.
- **Aujourd'hui** : Utilisation mondiale dans l'automobile, l'aéronautique, les industries et bien plus.

Domaines d'Utilisation

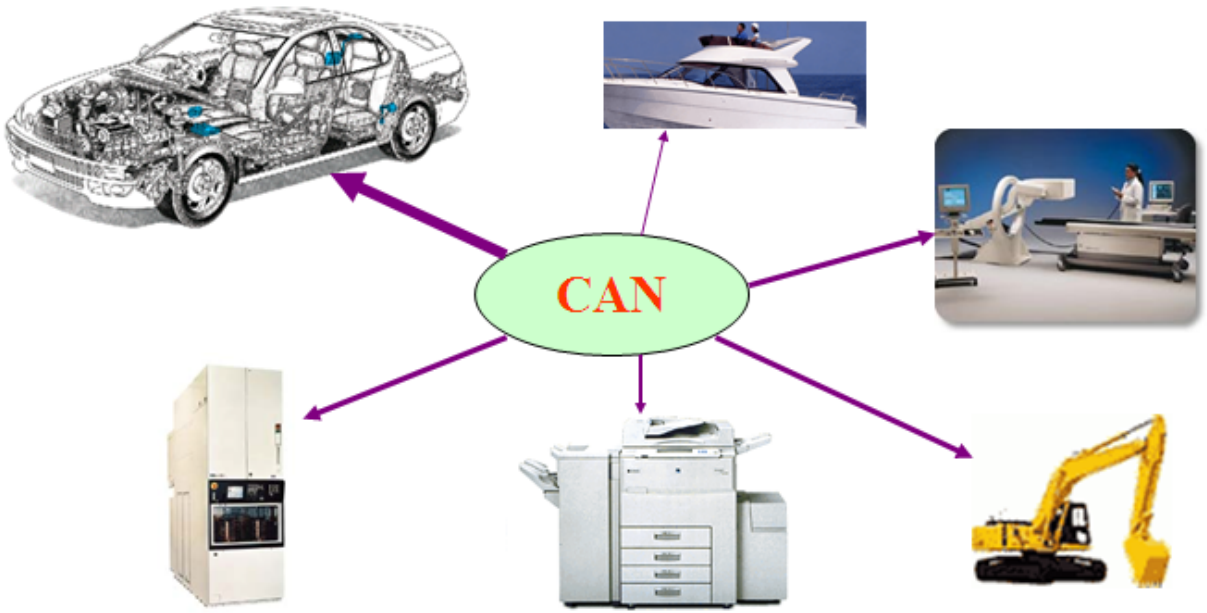

Figure – Utilisation de CAN

- **Industrie Automobile :** Gestion des moteurs, systèmes ABS, airbags
- **Industrie :** Automatisation, contrôle de machines, robots industriels
- **Médical :** Communication dans les appareils de diagnostic
- **Aéronautique et Spatial :** Systèmes embarqués dans les avions

Fonctionnement physique d'un bus CAN

Introduction au Bus CAN

- Le bus CAN (**Control Area Network**) est un protocole de communication série conçu pour :
  - Supporter les **systèmes embarqués temps réel**.
  - Garantir un **haut niveau de fiabilité**.
- **Domaines d'application :**
  - Réseaux à **moyen débit**.
  - Réseaux de **multiplexage à faible coût**.
- **Classification :**
  - Catégorie des **réseaux de terrain** (utilisés principalement dans l'industrie).

Propriétés du Bus CAN

- **Principales caractéristiques :**
  - **Hiérarchisation des messages** pour prioriser les informations.
  - **Garantie des temps de latence**, essentielle pour les systèmes critiques.
  - **Souplesse de configuration**, adaptée à des systèmes variés.
- **Fonctionnalités avancées :**
  - **Réception synchronisée** de multiples sources.
  - Mode **multimaître** pour une gestion collaborative des données.
  - **Détection et signalisation des erreurs**, avec retransmission automatique.
  - Identification des erreurs temporaires vs permanentes.
- **Sécurité renforcée :**
  - **Déconnexion automatique** des nœuds défectueux pour préserver le réseau.

Topologie

Chaque équipement connecté, appelé « noeud », peut communiquer avec tous les autres.

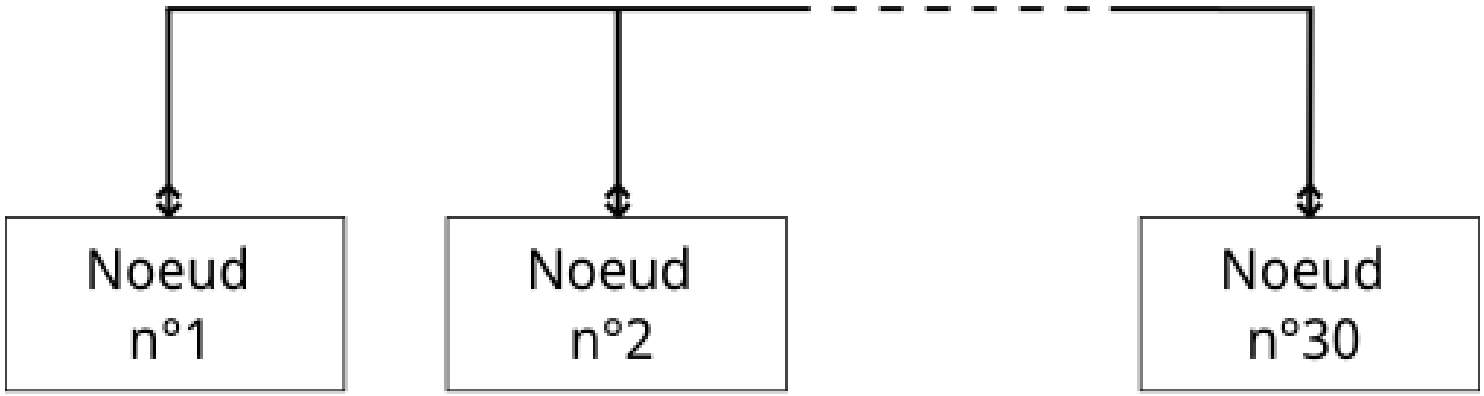

Figure – Architecture CAN avec plusieurs nœuds

Le nombre maximal de noeuds sur un bus CAN est limité par la capacité de pilotage d'un émetteur-récepteur CAN.

Support Physique

- Noeuds connectés au bus par l'intermédiaire d'une paire torsadée.
- Les deux extrémités du bus doivent être rebouclées par des résistances de 120 ohms +/-10%

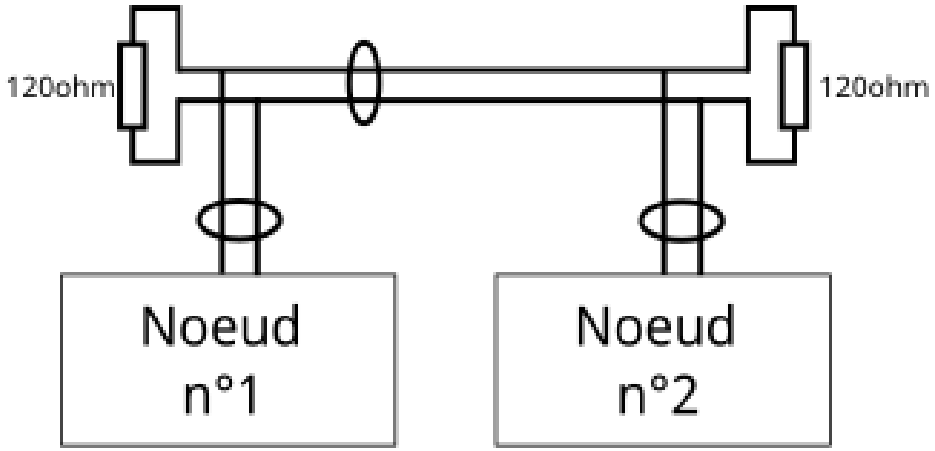

Figure – Support d'un bus CAN

Accès au bus de données en CSMA/CR :

- Écoute de chaque station avant de parler mais pas de tour de parole
- Résolution des collisions par priorité

Support Physique

- Bus de terrain soumis à des parasites importants.
- La transmission en paire différentielle limite les interférences.
- Les deux lignes du bus sont toutes les deux affectées de la même manière par un signal perturbateur.

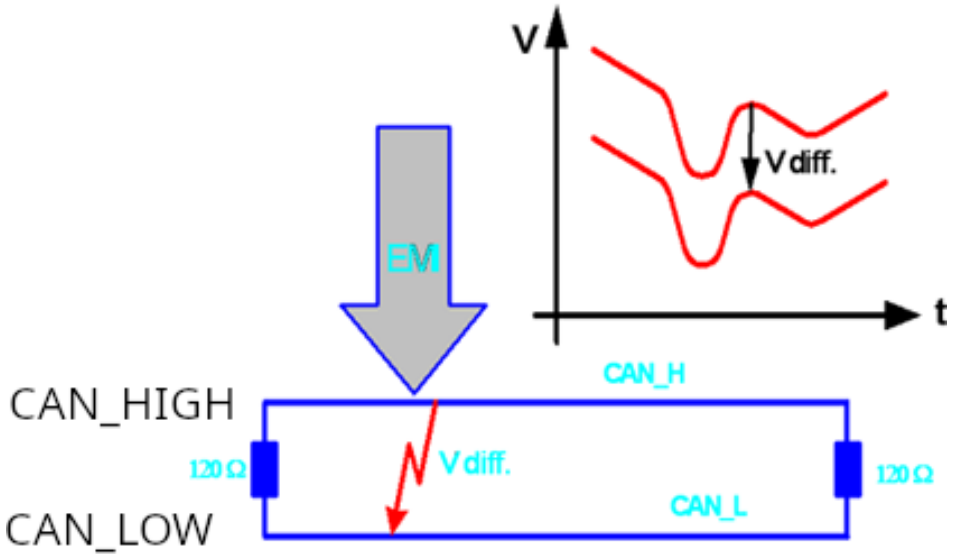

Figure – Support d'un bus CAN

Bus CAN High Speed

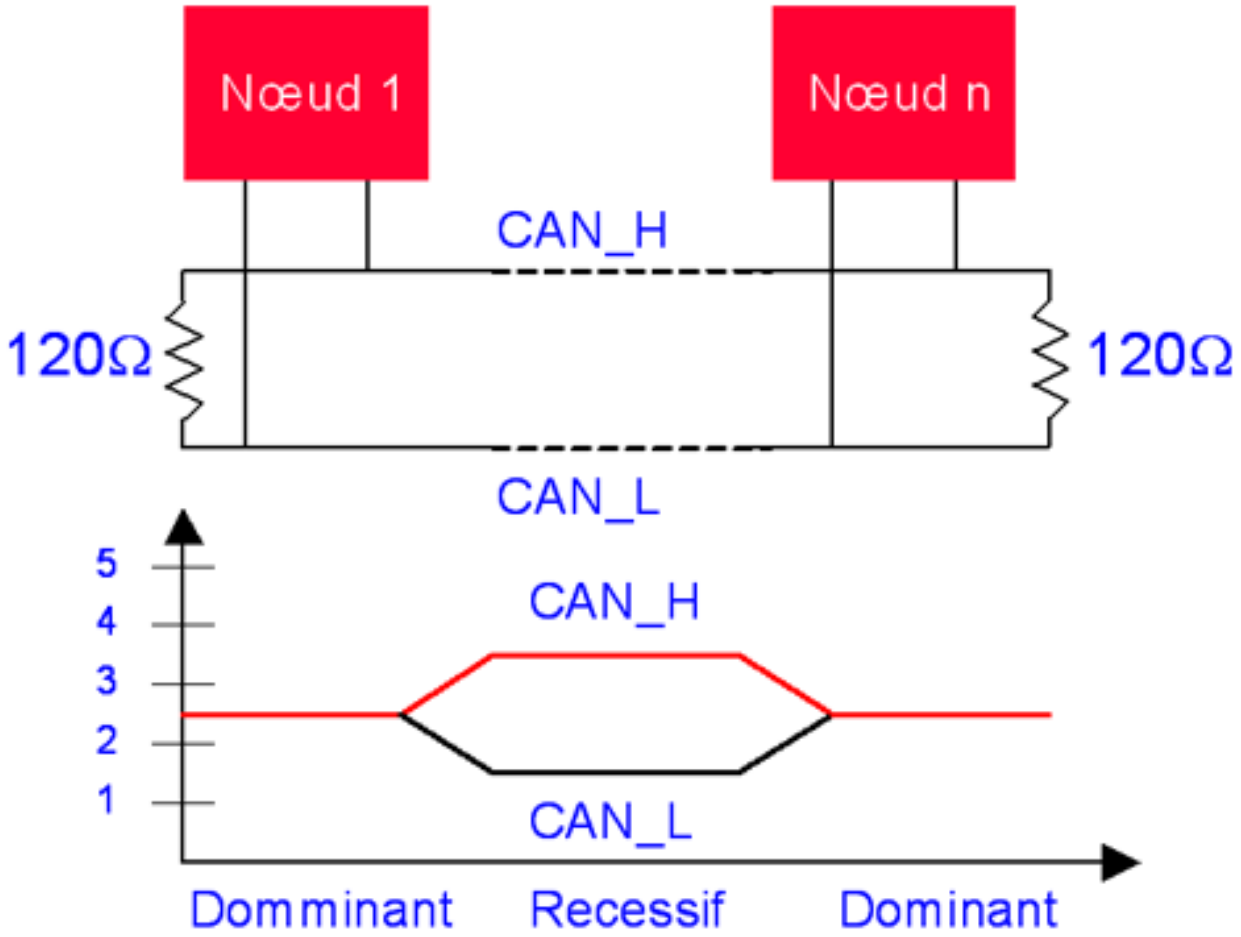

Figure – Bus CAN High Speed

Bus CAN High Speed

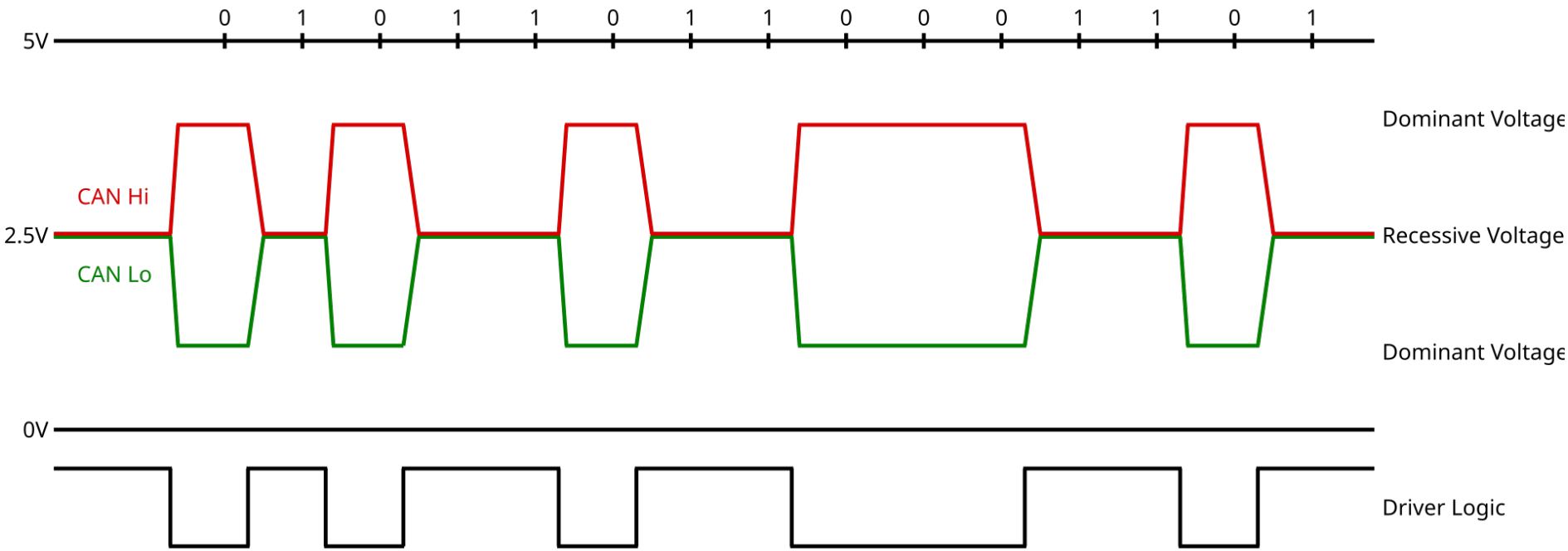

Figure – Bus CAN High Speed

Bus CAN Low Speed

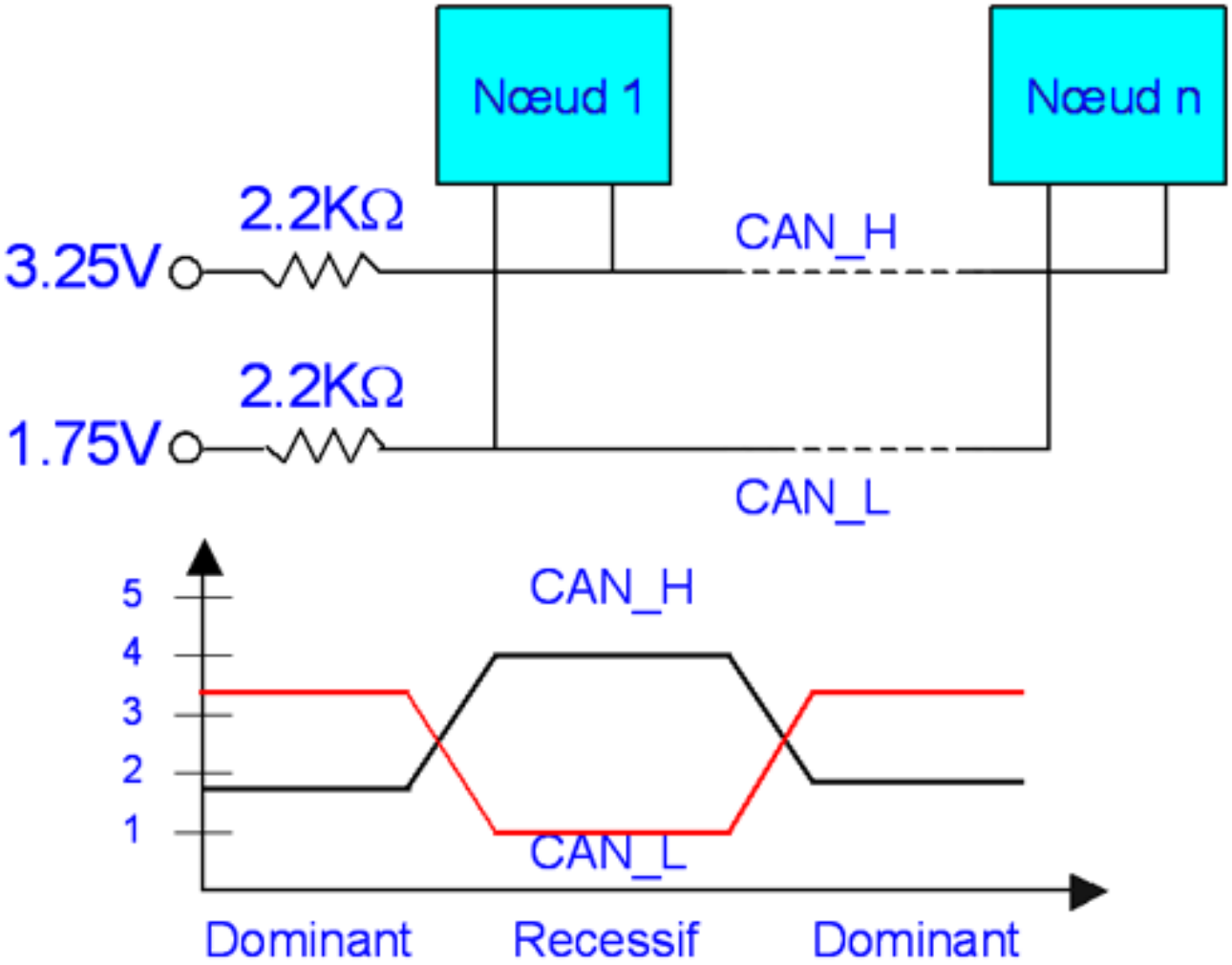

Figure – Bus CAN Low Speed

Bus CAN Low Speed

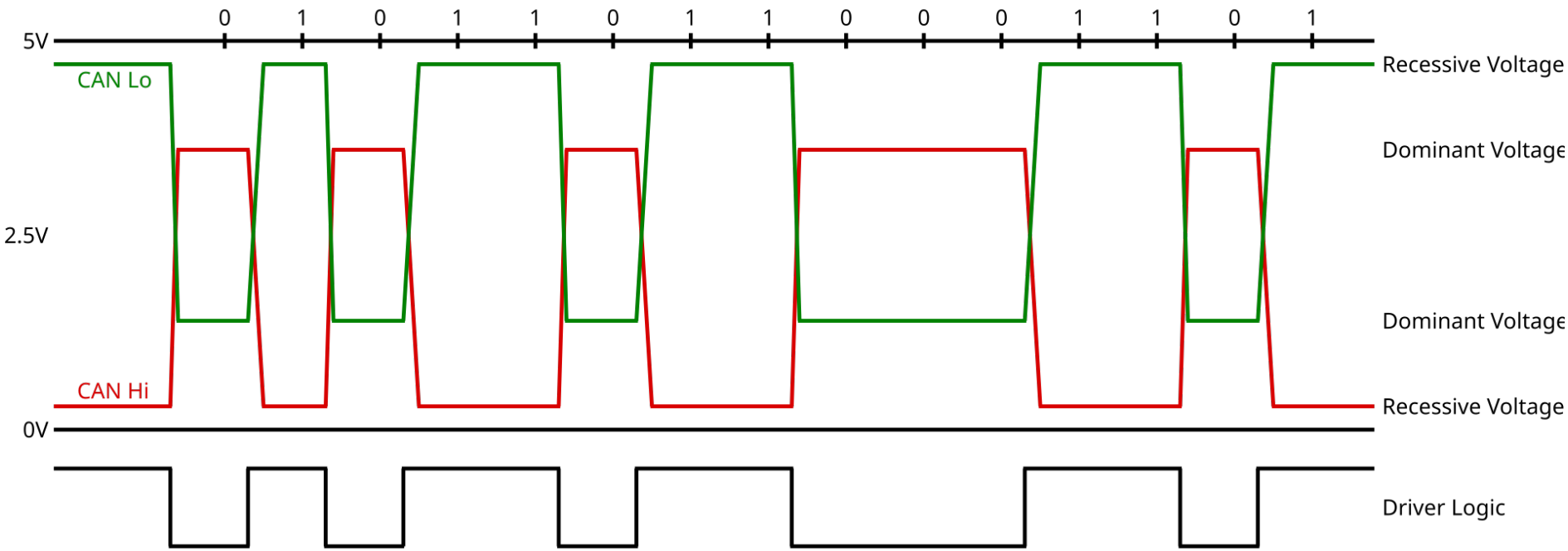

Figure – Bus CAN Low Speed

High Speed vs. Low Speed

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*120Ω
Tensions d'alimentation	5V	5V

Figure – Comparaison

Comparaison

Pourquoi ces différneces ?

- Dans **CAN High-Speed**, l'écart de tension **différentiel (2 V)** avec niveau dominant égal :
  - Permet des transitions rapides entre états dominant et récessif.
  - Moindre résistance aux erreurs.

- Dans **CAN Low-Speed**, un écart différentiel plus large (3,3 V) :
  - Aide à éviter la confusion entre les signaux.
  - Cependant, l'électronique est plus lente pour garantir la robustesse face aux interférences et pannes.

Implications sur la longeur du câble

Vitesse (kbit/s)	Longueur (m)
1 000	30
800	50
500	100
250	250
125	500
62,5	1 000
20	2 500
10	5 000

Garantir la synchronisation des récepteurs

- **Protocole asynchrone :** pas de signal d'horloge dédié pour synchroniser l'émetteur et les récepteurs.

- Repose sur les transitions de signal (de 0 à 1 ou de 1 à 0) pour maintenir la synchronisation entre les nœuds.
  - Lorsqu'une transition est détectée, les récepteurs réajustent leurs horloges pour rester alignés avec l'émetteur.

- Longue série de bits identiques $\rightarrow$ risque de désynchroniser les récepteurs.

Détection et gestion des erreurs

- Mauvaise synchronisation → erreurs de réception ou des décalages dans les bits interprétés.

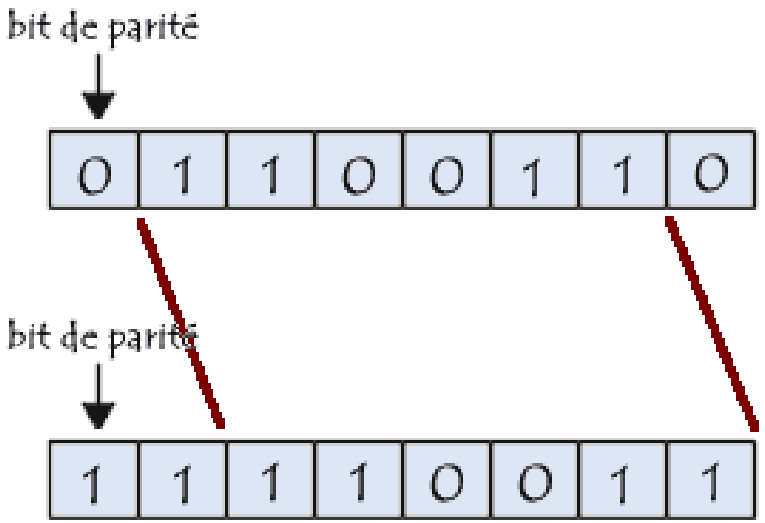

Figure – Exemple : Bits de parité et décallage vers la droite

Détection et gestion des erreurs

- **En pratique :** Utilisation de **vérifications des erreurs cycliques** (CRC) pour assurer de l'intégrité des données transmises → Même problème

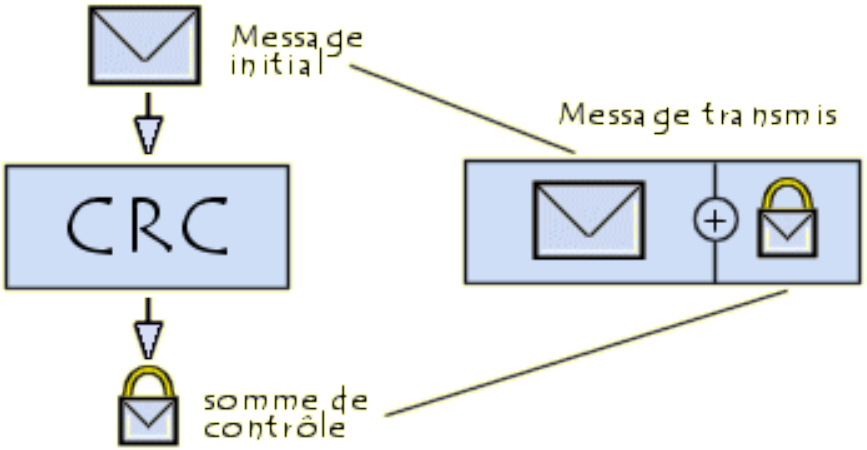

Figure – https://web.maths.unsw.edu.au/~lafaye/CCM/base/control.htm

Bit stuffing

Bit stuffing : Ajouter un bit de polarité opposée après cinq bits consécutifs identiques :

- Garantit une transition régulière pour maintenir la synchronisation.

- **Trame originale** : `111111` (6 bits identiques).
- Avec bit stuffing : un “0” est inséré après cinq “1” consécutifs :
  - **Trame après bit stuffing** : `1111101`.

- Les récepteurs détecteront cette transition, maintenant leur synchronisation.
- Les bits ajoutés sont retirés à la réception grace au CRC (calculé sans stuffing).

Différences entre CAN High-Speed et Low-Speed

- **CAN High-Speed** : Le bit stuffing est essentiel car les débits élevés (jusqu'à 1 Mbps) augmentent le risque de désynchronisation sur le bus. Les transitions fréquentes garantissent des communications stables et fiables.

- **CAN Low-Speed** : Bien que le débit soit plus faible (125 kbps typiquement), le bit stuffing est également utilisé pour éviter la désynchronisation.

Bits dominants et récessifs

- **Dominant :** 0
- **Récessif :** 1
- Comme en génétique, en cas de conflit, le dominant écrase le récessif.

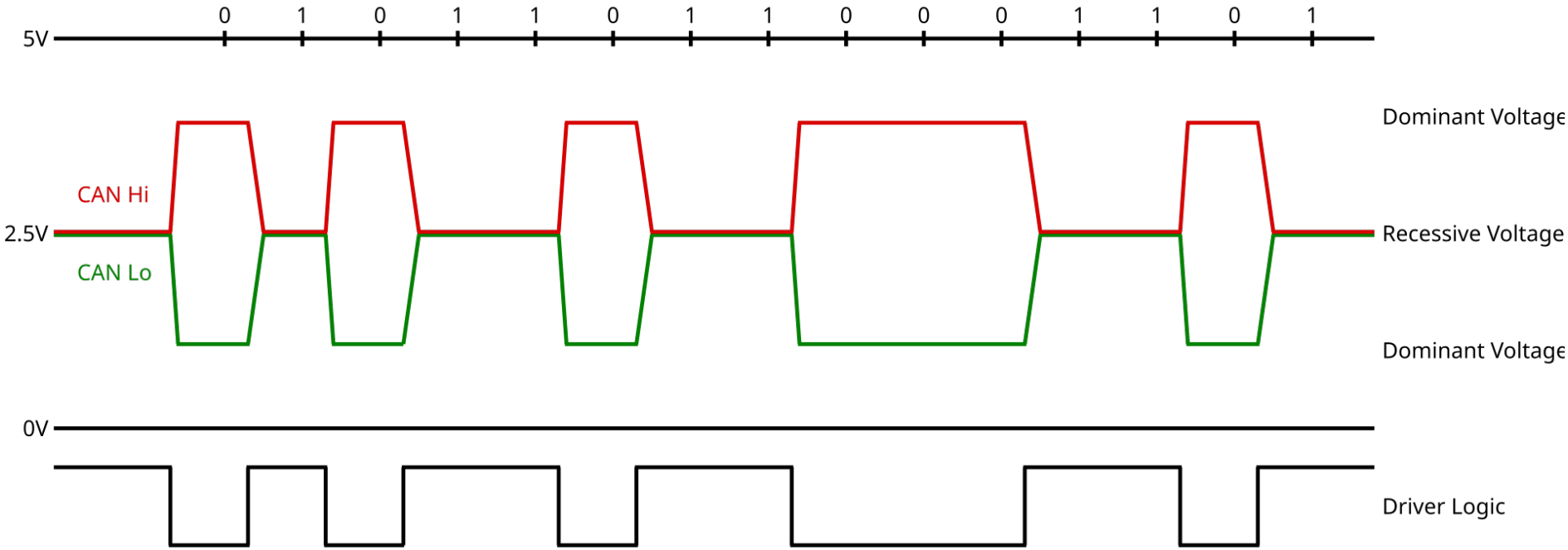

Figure – Bus CAN High Speed

Couche applicative d'un bus CAN

Trame d'un message CAN

- **Types de trames :** trame de données, trame de requête, trame d'erreurs, trame de surcharge

- Format des trames de données (CAN standard, version 2.0A) :

SOF	ID	Commande	Données	Contrôle	ACK	FIN	Intertrame
1 bit	12 bits	6 bits	0 à 64 bits	16 bits	2 bits	7 bits	3 bits

Figure – Trame d'un message CAN

Trame d'un message CAN

SOF	ID	Commande	Données	Contrôle	ACK	FIN	Intertrame
1 bit	12 bits	6 bits	0 à 64 bits	16 bits	2 bits	7 bits	3 bits

Figure – Trame d'un message CAN

- **SOF (Start of Frame)** : bit 0 (dominant) => permet la détection de début d'émission
- **ACK (acknowledgement)** : l'émetteur envoie 11 ; le récepteur acquitte un message correct en envoyant 0 sur le premier bit (qui écrase le 1)
- **FIN (finalisation)** : tous les bits sont à l'état 1
- **Intertrame** : tous les bits sont à l'état 1

Trame d'un message CAN

SOF	ID	Commande	Données	Contrôle	ACK	FIN	Intertrame
1 bit	12 bits	6 bits	0 à 64 bits	16 bits	2 bits	7 bits	3 bits

Figure – Trame d'un message CAN

- **Commande** : 2 bits réservés + 4 bits pour la taille des données
- **Données** : les données applicatives
- **Contrôle** : 15 bits de CRC (polynôme générateur 0xC599) + 1 bit toujours à 1

Trame d'un message CAN

SOF	ID	Commande	Données	Contrôle	ACK	FIN	Intertrame
1 bit	12 bits	6 bits	0 à 64 bits	16 bits	2 bits	7 bits	3 bits

Figure – Trame d'un message CAN

- Dans un réseau CAN, chaque message possède un **identifiant (ID)** qui a deux fonctions principales :
  1. **Identifier de manière unique les données** transmises (par ex., vitesse moteur, température).
  2. **Déterminer la priorité** du message pendant l'arbitrage.
- **ID (Identification)** : 11 bits d'identifiant/priorité (avec les 7 premiers qui ne doivent pas être tous à 1) et 1 bit données/requête

Comment fonctionne la priorité (arbitrage) ?

- CAN utilise le protocole **CSMA/AMP** (Carrier Sense Multiple Access avec Arbitrage basé sur la Priorité des Messages) :
  - Plusieurs nœuds peuvent transmettre des messages en même temps.
  - Le champ ID du message détermine quel nœud a accès au bus.
  - Les valeurs **les plus basses** de l'ID ont une **priorité plus élevée**.

- Lorsqu'au moins deux nœuds transmettent en même temps :
  - Chaque nœud surveille le bus pendant qu'il transmet.
  - Si un nœud transmet un bit récessif (`1`) mais détecte un bit dominant (`0`), il arrête sa transmission car il a perdu l'arbitrage.
  - Le nœud avec l'**ID le plus faible (priorité la plus élevée)** continue de transmettre.

Exemple d'arbitrage

	Start bit	ID bits										The rest of the frame
		10	9	8	7	6	5	4	3	2	1	0	
Node 15	0	0	0	0	0	0	0	0	1	1	1	1	
Node 16	0	0	0	0	0	0	0	1	Stopped Transmitting				
CAN data	0	0	0	0	0	0	0	0	1	1	1	1	

Figure – Exemple d'arbitrage

Structure de l'ID des messages

Le CAN prend en charge deux formats d'identifiants :

1. **CAN Standard (CAN 2.0A)** :
   - Identifiant de 11 bits.
   - Permet **2 048 ID uniques** (de `0x000` à `0x7FF`).
2. **CAN Étendu (CAN 2.0B)** :
   - Identifiant de 29 bits.
   - Permet **536 870 912 ID uniques** (de `0x00000000` à `0x1FFFFFFF`).

Pourquoi deux formats ?

- Les **11 bits** suffisaient pour les premières applications automobiles.
- Avec l'extension des applications industrielles, aérospatiales, et IoT, les **29 bits** offrent une meilleure évolutivité pour les réseaux plus grands.

Implications de la priorité des IDs

Pourquoi la priorité est-elle importante ?

- Les messages critiques (par ex., état du système de freinage) doivent être transmis plus rapidement, même si le réseau est occupé.
- Les messages de faible priorité (par ex., données de divertissement) peuvent attendre en cas de congestion.

1. **Communication déterministe** :
   - Les messages critiques remportent toujours l'arbitrage et sont transmis sans délai.
2. **Utilisation efficace du bus** :
   - Pas de temps perdu à retransmettre les messages de faible priorité lorsque le bus est occupé.
3. **Pas de collisions** :
   - Le CAN évite les collisions grâce à son mécanisme d'arbitrage.

Attaques sur CAN

Bus Flood Attack

- **Objectif :** Bloquer le bus en envoyant des trames CAN à une vitesse maximale, empêchant les trames légitimes d'être transmises à temps.
- **Mécanisme :**
  - Utiliser un ID de trame de haute priorité (ex. : ID = 0) pour monopoliser la bande passante.
  - Retarder les trames de basse priorité et provoquer des erreurs dans le système.
- **Conséquence :**
  - Peut entraîner des défaillances dans les systèmes critiques si les trames ne sont pas reçues à temps.

Simple Frame Spoofing

- **Objectif :** Faire croire qu'une trame provient d'un émetteur légitime.
- **Mécanisme :**
  - Envoyer une trame avec un ID et des données falsifiés.
  - Risque de **boucle fatale d'arbitrage** : deux trames avec le même ID provoquent une désynchronisation car chaque nœud détecte une erreur et retransmet.
- **Conséquence :**
  - Les nœuds passent en mode dégradé ou fail-safe.

Adaptive Spoofing

- **Objectif :** Maximiser l'impact de l'usurpation en remplaçant une trame légitime dans la mémoire tampon d'un récepteur.
- **Mécanisme :**
  - Observer le bus pour détecter une trame légitime.
  - Envoyer rapidement une trame falsifiée juste après pour écraser la trame légitime dans le buffer.

Figure – Adaptive Spoofing

- **Exemple :** Modifier les données d'un capteur critique comme la vitesse ou la pression des pneus.

Wire-Cutting Spoofing

- **Objectif :** Partitionner le bus pour simuler des données falsifiées.
- **Mécanisme :**
  - Couper physiquement le bus CAN et modifier les trames à la volée.
  - Plus besoin de jouer sur les priorités pour injecter.
  - Possiblilité de d'envoyer des trames différentes des deux côtés.

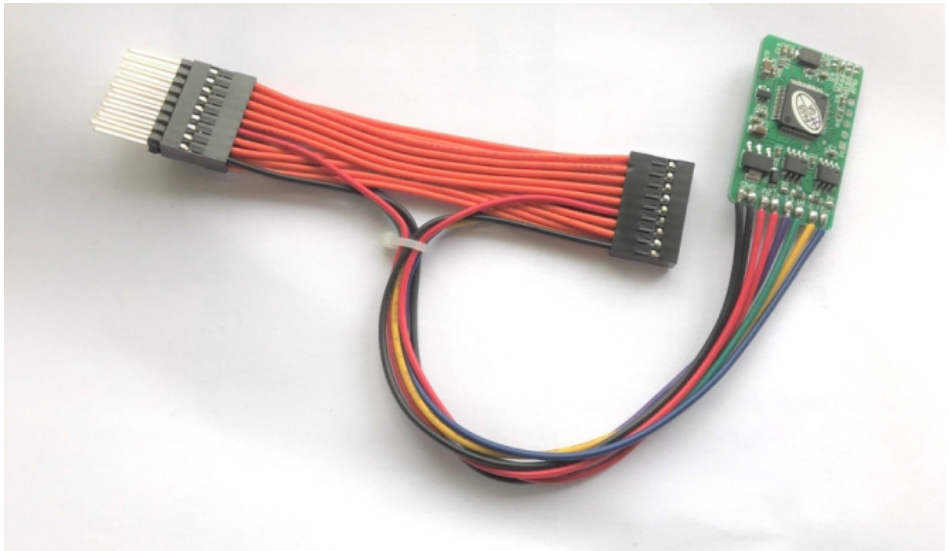

Figure – Wire-cutting Attack

Double Receive Attack

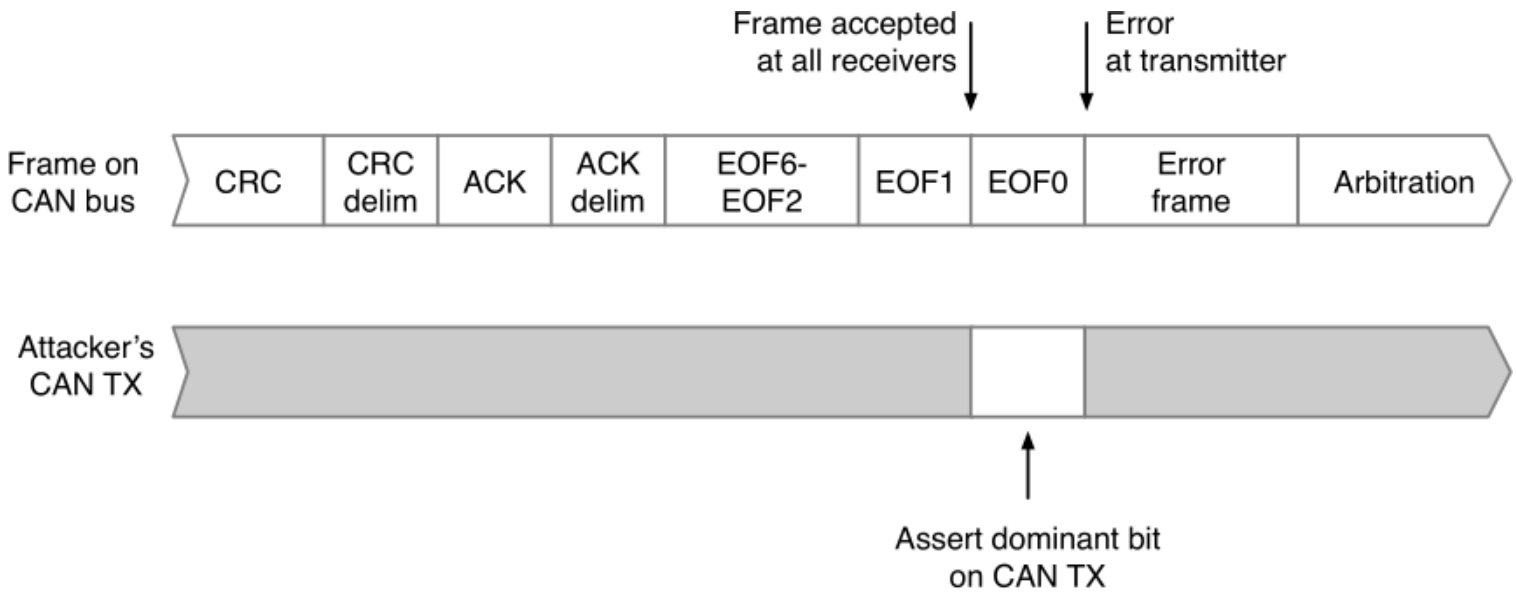

Figure – Double receive attack

- **Mécanisme :**
  - Insérer un bit dominant à la fin d'une trame pour déclencher une retransmission.
  - Les récepteurs acceptent la trame deux fois, causant des incohérences dans le système.

Bus-off Attack

- **Objectif :** Bloquer les trames d'un ECU ciblé pour interrompre ses communications.
- **Mécanisme :**
  - Provoquer des erreurs répétées sur toutes les trames émises par l'ECU.
  - Augmenter le compteur d'erreurs jusqu'à ce que l'ECU soit en mode **bus-off**.
- **Conséquence :**
  - Le véhicule passe en mode dégradé ("Limp Mode"), ou des fonctionnalités essentielles cessent de fonctionner.

Freeze Doom Loop Attack

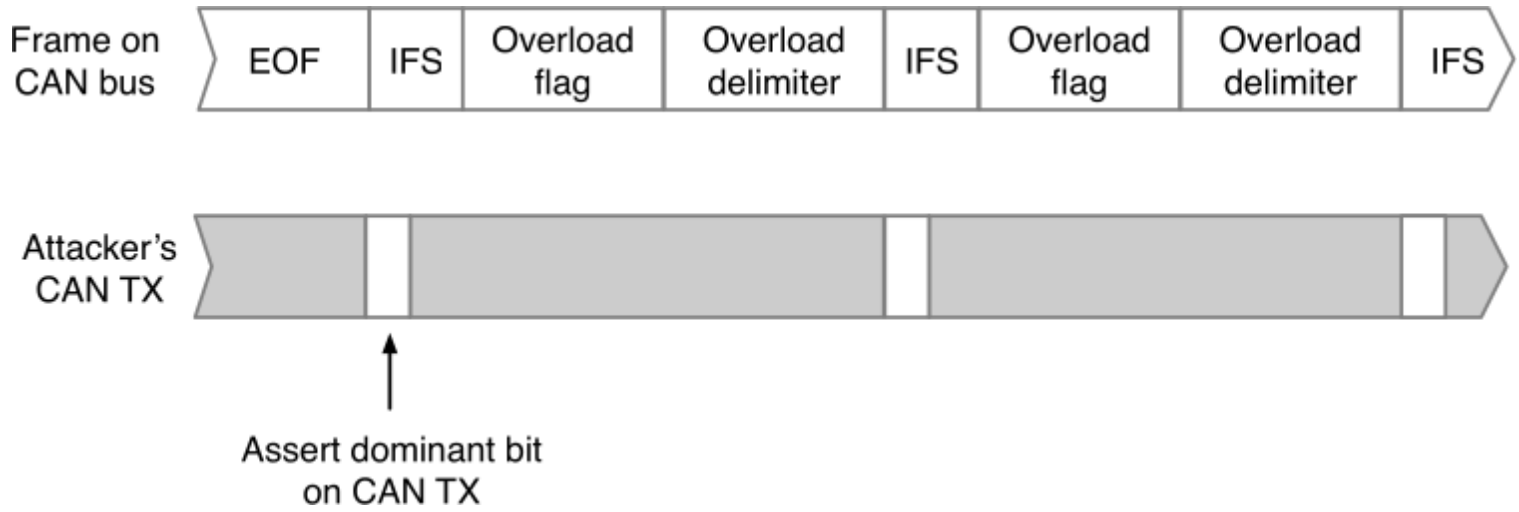

Figure – Freeze loop attack

- **Objectif :** Geler le trafic CAN pour retarder ou bloquer des trames critiques.
- **Mécanisme :**
  - Exploiter un ancien comportement du protocole où un bit dominant dans l'espace inter-trame (IFS) provoque une boucle de récupération d'erreur.
  - Répéter cette action pour maintenir le bus dans un état bloqué.

Références

Références utilisées dans le cours

- https://www.technologuepro.com/cours-systemes-embarques/cours-systemes-embarques-Bus-CAN.htm
- https://en.wikipedia.org/wiki/CAN_bus
- https://en.wikipedia.org/wiki/OBD-II_PIDs