

Important : Un rapport électronique doit être envoyé au plus tard **15 jours après le TP** à l'adresse : **rachedi@univ-mlv.fr**.

Objectif :

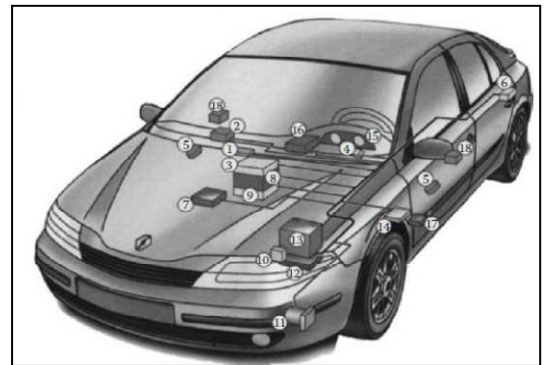
- Comprendre l'intérêt et l'organisation d'un bus de terrain.
- Analyser les données et les formats des données échangées sur le bus.
- Décoder une trame CAN.
- Comprendre comment les différents équipements de voiture échangent des informations via le bus CAN

Pré-requis :

- Principes de base du réseau CAN
- Logiciel CANOE version 5.1

A) **Mise en situation**

A1) **Généralités sur le réseau Bus CAN**



En 1983, le bus CAN (Controllor Area Network) a été conçu par la société allemande Robert BOSCH GmbH pour répondre aux besoins de communication interne dans les automobiles : multiplexage de commandes électriques, fiabilité, diagnostic, compatibilité électromagnétique, commandes d'organes (suspension, frein, contrôle moteur). Son exploitation ne commence qu'à partir de 1985 où une convention entre BOSCH et Intel a permis d'implanter le protocole CAN dans des circuits Intel. En 1986, la première standardisation du bus par l'ISO (International Standard Organization) a vu le jour. En 1987, Intel produit le premier contrôleur CAN, le 82526. **La première voiture multiplexée à utiliser le bus CAN comme support de transmission a été réalisée en 1991 (avec un débit de 500 kbit/s).**

La première spécification du bus CAN était propre aux applications dans l'automobile. Une **classification** simple et pragmatique a été établie par ISO en retenant deux classes d'applications :

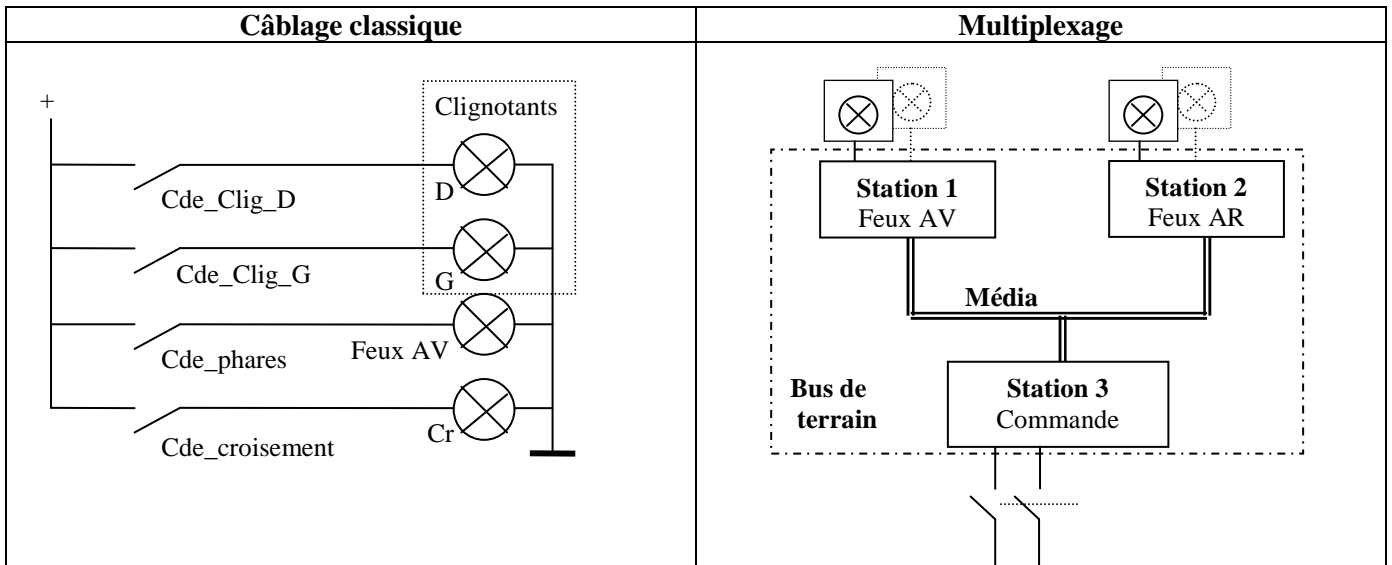
- vitesse de transmission lente définie comme étant inférieure à **125 kbit/s** ;
- vitesse de transmission élevée considérée comme étant supérieure à **125 kbit/s**.

Le CAN est un système de communication, en temps réel, par **liaison série** conçu pour relier des composants intelligents ainsi que des capteurs et des actionneurs dans une machine ou un procédé. Il possède des propriétés multimaîtres, c'est-à-dire que plusieurs noeuds peuvent simultanément demander l'accès au bus. Le CAN ne possède pas de système d'adressage mais plutôt un système d'allocation de priorités aux messages basé sur l'identificateur attribué à chaque message. Un émetteur transmet un message sans indication de destinataire ; sur la base de l'identificateur associé à ce message, chaque noeud décide de traiter ou d'ignorer ce message.

A2) **L'automobile n'échappe pas aux réseaux**

En diminuant l'encombrement et le poids du câblage dans les automobiles, les **bus de communication** ont permis de réduire les coûts de fabrication, d'améliorer la fiabilité des systèmes électriques et d'assurer leur évolutivité.

Le remplacement du câblage classique (un capteur, un fil, un récepteur) par un **bus (comportant peu de fils)** auquel se connectent les équipements du véhicule nécessite la mise en place d'une technique de transmission des informations appelée **multiplexage**.



Si le schéma de gauche est relativement simple (des connaissances élémentaires sur les circuits électriques et leur représentation schématique suffisent pour comprendre que l'action sur le commutateur « Cde_phares » établira un courant électrique dans l'ampoule « Feux AV » et donc son éclairage), le schéma de droite n'est pas aussi explicite.

En effet, il n'y a plus une liaison directe entre le capteur et le récepteur mais des interfaces, appelées **stations** (ou nœuds) qui se chargent de collecter les **informations** délivrées par les capteurs (commutateur, etc...) et de les **transmettre** sur un **média** à d'autres stations chargées de commander les récepteurs (ampoule de phare, etc...).

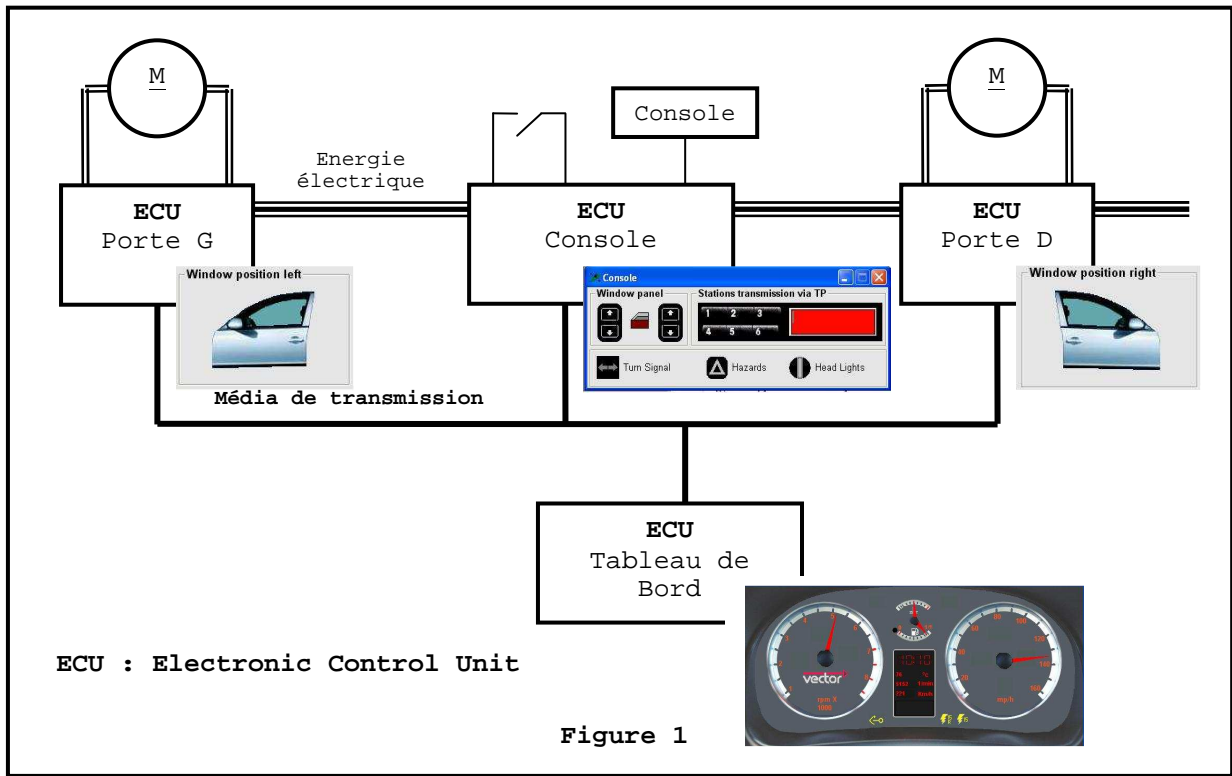
Lorsque toutes les stations sont connectées à une même voie de communication comme ci-dessus, on parle de **topologie** de type **bus**.

A3) Le bus CAN : un standard pour relier les équipements de confort d'un véhicule

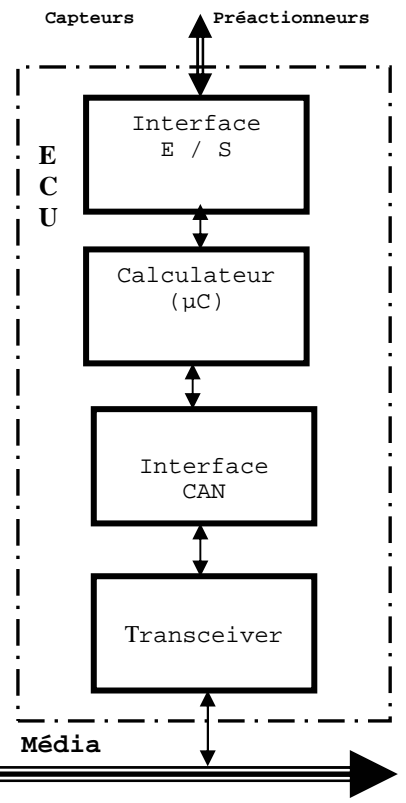
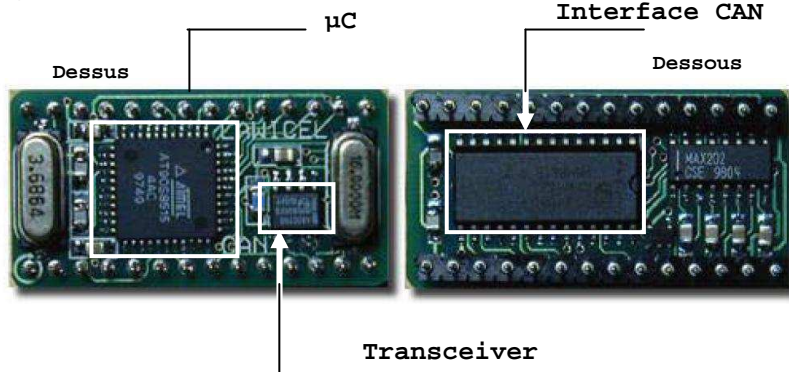
Dans sa version **basse vitesse** (125kbts/s), le **bus CAN** (**C**ontroler **A**rea **N**etwork) est utilisé dans l'automobile pour relier les **équipements de confort** (éclairage, lève-vitre, rétroviseur etc...)

La figure 1 ci-dessous représente **quatre équipements** : deux moteurs de lève-vitre, une console de commande et un tableau de bord.

Ces éléments communiquent par l'intermédiaire d'un bus CAN composé d'un **média de transmission** (fils électriques) et d'unités de contrôle électronique (**ECU**). Les ECU sont les stations du réseau CAN (nœud CAN).



Organisation d'une E.C.U.



Remarque : L'interface CAN est souvent intégrée au μC

Rôle des éléments

L'interface E/S adapte les signaux échangés entre le μC et les capteurs ou les préactionneurs.

Le μC se charge du transfert des données (utiles), identifie les messages, analyse l'état des capteurs et commande les préactionneurs.

L'interface CAN : se charge de transmettre les données en respectant le **protocole CAN**.

Le transceiver met en forme le signal électrique transmis sur le média.

B) Etude de cas

Pour vous aider à comprendre le principe de fonctionnement d'un **réseau à diffusion** tel que le bus CAN ainsi que les notions plus générales de **multiplexage**, de **message** et de **protocole**, vous allez utiliser un outil de conception, de test et d'analyse pour les systèmes embarqués (logiciel **CANOË** distribué par la société VECTOR*).

Phase 1 : Mise en situation, mise en œuvre de l'environnement de simulation (logiciel CANOË)

- (1) Lancez le logiciel CANOË à partir du menu démarrer.
- (2) Chargez le fichier de démonstration (**automot.cfg**) situé dans le répertoire AUTOMOT du TP.
→ File → Load configuration → automot.cfg

→ clic sur « **Both Busses** » dans le panneau « Control » pour faire apparaître les fenêtres « Door », « Console » et « Engine »

Organisez votre écran pour qu'il ressemble à celui de la figure ci-dessous.

→ **Fermez** les fenêtres non utilisées (**Conservez uniquement « Trace Ibus »**)

Les différentes fenêtres présentes à l'écran (Console, Door, Dashboard etc.) vont vous permettre de générer des **trames CAN** sur le bus et de les **visualiser** dans la fenêtre « **Trace Ibus** ».

Lorsque vous allez lancer la démonstration, le logiciel va simuler la mise en route du véhicule :

- Les cinq vitesses vont passer automatiquement dans le panneau « Engine »,
- Les paramètres du véhicule (vitesse, tours moteur, température etc...) vont s'afficher dans le panneau « Dashboard ».

Une fois le démarrage automatique réalisé, vous pourrez « conduire le véhicule »

- (3) Pour lancez la démo :



Présentation de l'environnement de simulation Ibus

Vous pouvez agir sur :

- l'accélérateur (accelerator position) dans le panneau « Engine »
- les clignotants (turn signal), les feux de détresse (hazards), les feux (head light), monter et descendre les vitres droite et gauche et choisir une station de radio dans le panneau « Console ».

Les lève-vitres sont simulés dans le panneau « Doors ».

Modifiez les différents paramètres cités ci-dessus et observez ce qui se passe dans la fenêtre « Trace Ibus ».

La configuration actuelle de cette fenêtre sera « confortable » dans la suite du TP mais n'est pas très représentative de ce qui se passe « **dans la réalité** ».

Configuration de la fenêtre « trace ibus » en mode chronologique (annexe 2 dossier ressource)

Positionnez la souris sur la fenêtre « Trace ibus » puis clic droit

- Configuration
- Display mode (cochez chronological) → ok

Remarque : Pour arrêter le défilement : → clic droit sur la fenêtre → pause

Le moins que l'on puisse dire est qu'il se passe toujours quelque chose sur les bus !

En effet, le fichier de démonstration « automot » met en œuvre **deux** bus CAN: un bus est affecté au moteur (Motbus), l'autre (**Ibus**) est affecté aux équipements de confort. Indépendamment de vos actions sur les lèves-vitres ou sur les autres commandes, des messages issus du bus moteur (Motbus) sont transmis au tableau de bord (Dashboard) par l'intermédiaire d'une passerelle (**Gateway**) qui assure la communication entre les bus.

Des messages circulent donc régulièrement sur Ibus.

La fenêtre « Trace Ibus » est votre outil de mesure

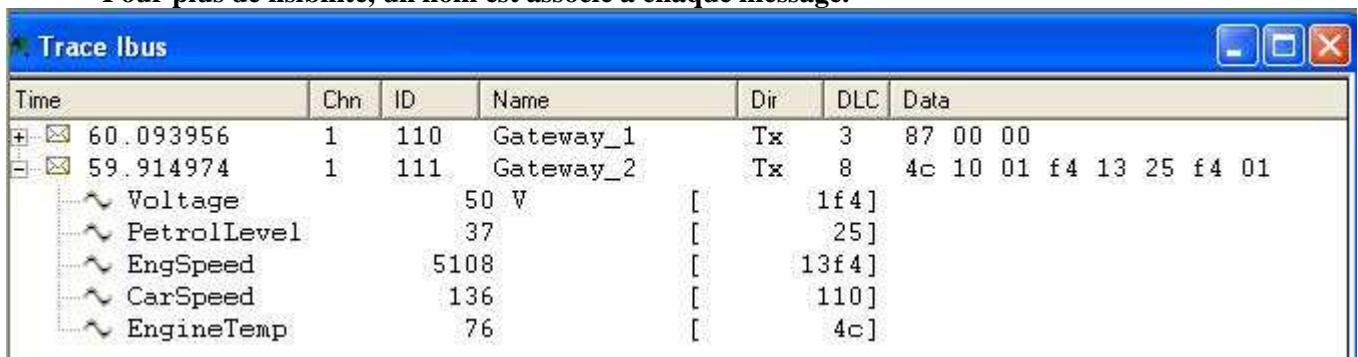
Organisation de la fenêtre « Trace Ibus »

La fenêtre « Trace Ibus » vous permet de connaître :

- le moment (**Time**) auquel passe la trame par rapport à une date zéro (début de la simulation),
- le bus CAN (**Chn**) sur lequel circule la trame (Ibus = 1, Motbus = 2),
- l'identificateur du message (**ID**),
- la direction (**Dir**) du message (Transmis **Tx** ou reçu **Rx**),
- le nombre d'octets de données dans la trame (**DLC**),
- les données transmises dans la trame (**Data**).

Vous comprendrez mieux tout ceci après la phase 2 de l'étude de la problématique.

Pour plus de lisibilité, un nom est associé à chaque message.



Time	Chn	ID	Name	Dir	DLC	Data
60.093956	1	110	Gateway_1	Tx	3	87 00 00
59.914974	1	111	Gateway_2	Tx	8	4c 10 01 f4 13 25 f4 01
			Voltage			50 V [1f4]
			PetrolLevel			37 [25]
			EngSpeed			5108 [13f4]
			CarSpeed			136 [110]
			EngineTemp			76 [4c]

Exemple : Le message Gateway_2 dont l'identificateur est 111 a été transmis 59,9...s après le début de la simulation. Il contient 8 octets représentatifs de l'état du véhicule (tension générateur (Voltage), niveau de carburant (PetrolLevel) etc...).

Comme vous pouvez le constater, il suffit de cliquer sur l'onglet +/- pour connaître le détail du message. Voltage, PetrolLevel, EngSpeed, CarSpeed, EngineTemp sont les signaux contenus dans le message Gateway_2.

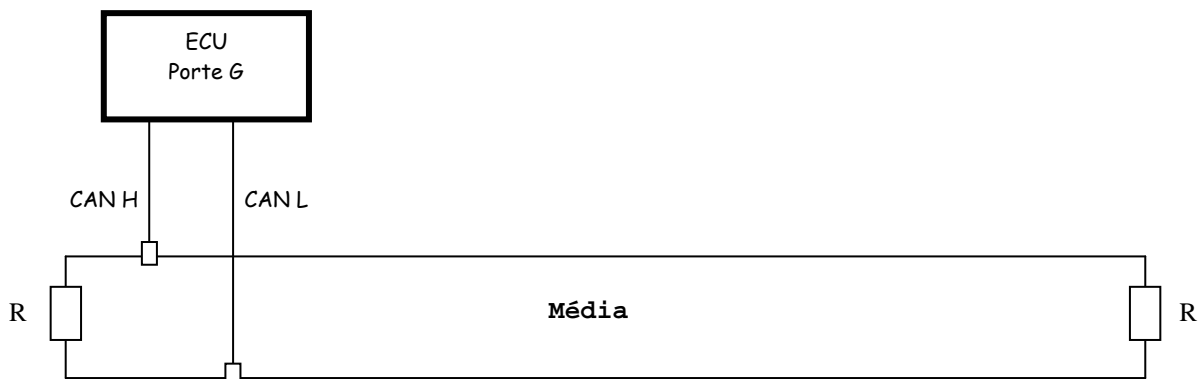
Phase 2 : Généralités sur le bus CAN

Avant d'analyser les trames qui « passent » sur Ibus, il est nécessaire d'acquérir un minimum de connaissances sur le protocole CAN et notamment sur l'organisation d'une **trame CAN**.

Connexion des stations (nœuds CAN) au média

Lorsque le média est constitué de fils électriques, les stations (nœuds CAN) sont câblées comme sur le schéma du DR1. La valeur des résistances R de terminaison dépend de la longueur du média.

Q1) Complétez ce schéma ci-dessous pour qu'il corresponde à l'organisation de Ibus.

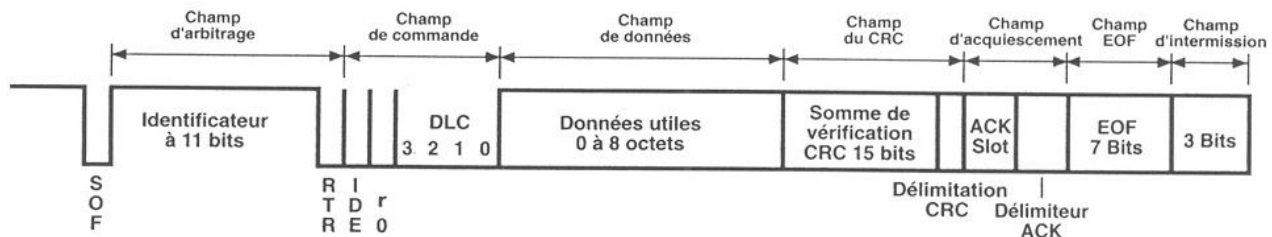


Identification des champs (En-tête, Données, Terminateur) dans la trame CAN

On rappelle ci-dessous l'organisation schématique d'une trame proposée dans le cours « Introduction aux transmissions de données ».

En tête	Données applicatives	Terminateur
---------	----------------------	-------------

Q2) Identifiez, en les entourant, les champs en-tête, données applicatives et terminateur de la trame CAN donnée par le schéma ci-dessous.



Q3) A quoi correspondent les termes « Identificateur » et « DLC » (voir § Data-frame, le paquet de données) ?

Phase 3 : Analyse de trames CAN

- Configuration de la fenêtre « Trace Ibus » en mode fixe
Positionnez la souris sur la fenêtre « trace Ibus » puis clic droit
→ Configuration → Display mode (cochez fixed position)

Arrêtez puis relancez la simulation.

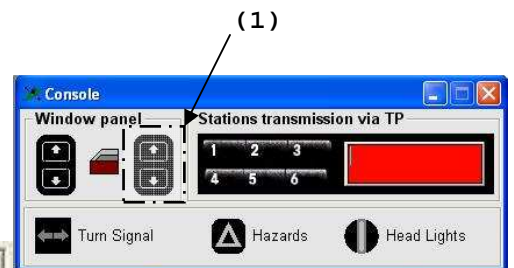
Clic pour arrêter



Chacune de vos actions dans l'environnement de simulation va se traduire par la génération d'un message dans la fenêtre Trace Ibus.

Rappel : La fenêtre « Trace Ibus » affiche les messages contenus dans les trames CAN « qui passent » sur Ibus.

- Commande de la vitre droite
En agissant sur les boutons-poussoirs de la console (repère 1 ci-contre) l'utilisateur déclenche, à son insu, une séquence de transmission de données.
Dans ce paragraphe, vous allez établir cette séquence en identifiant les messages générés et leur contenu.
Appuyez sur les boutons (1) et observez le comportement de la vitre et le contenu des messages dans la fenêtre « Trace Ibus »



Média

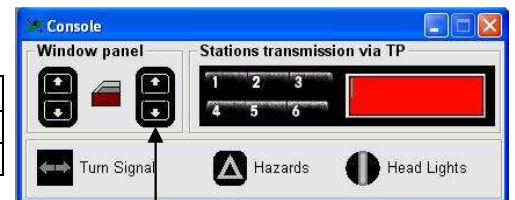


Q4) Quel est le nom des messages associés à la porte droite et aux boutons (1) de la console.

- Détermination du contenu des messages associés à la vitre droite et aux boutons (1) de la console

Q5) Ouvrez complètement la vitre de la porte droite et complétez le tableau ci-dessous.

ID	Name	DLC	Data

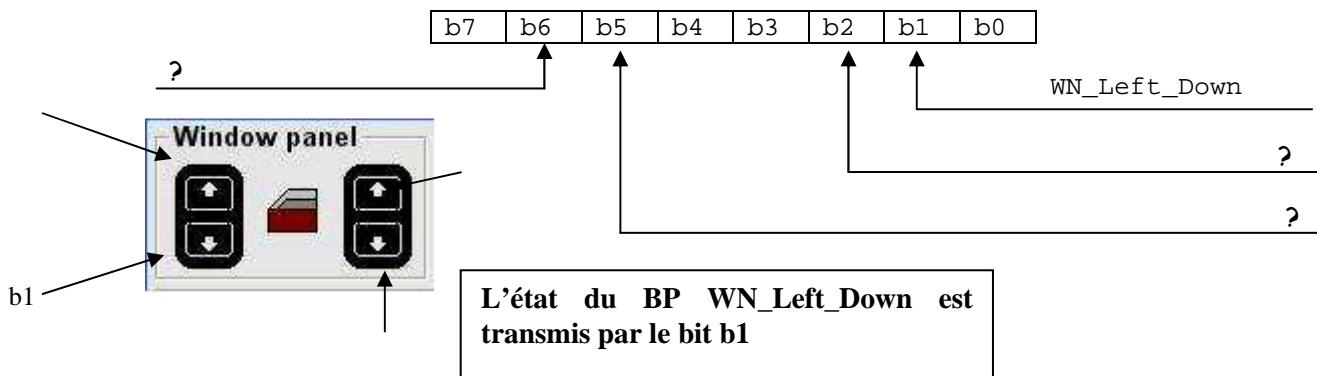


Appui sur

- Identification des bits associés aux boutons-poussoirs dans le message Console_1

Q6) Quel octet {premier (le plus à gauche dans la trame), deuxième, troisième ou quatrième} contient l'état des boutons-poussoirs dans le message Console_1.

Q7) Affichez le détail du message Console_1 et précisez la position des quatre signaux associés aux boutons-poussoirs dans l'octet et sur le dessin « Window panel » montré ci-dessous.



Q8) Quelle est la nature (monter, descendre, position etc..) de l'information renvoyée par l'ECU de la porte droite ?

- Chronologie des messages lors de l'action sur les BP de commande des vitres

Q9) Configurez la fenêtre « trace Ibus » en mode chronologique. Appuyez brèvement sur un des quatre BP. Mettez en pause par clic droit → pause
Retrouvez les trames Console_1 et DOOR_r dans la fenêtre «Trace Ibus».
Que remarquez-vous ?

1a0	Console_1	Tx	4	00 00 00 40
1f1	DOOR_r	Tx	1	06
1a0	Console_1	Tx	4	00 00 00 40
1f1	DOOR_r	Tx	1	05
110	Gateway_1	Tx	3	87 00 00
1a0	Console_1	Tx	4	00 00 00 40
1a0	Console_1	Tx	4	00 00 00 40
1f1	DOOR_r	Tx	1	05
1a0	Console_1	Tx	4	00 00 00 40
1f1	DOOR_r	Tx	1	04
1a0	Console_1	Tx	4	00 00 00 40
1a0	Console_1	Tx	4	00 00 00 00
1f1	DOOR_r	Tx	1	03
110	Gateway_1	Tx	3	87 00 00
1f1	DOOR_r	Tx	1	03
1f1	DOOR_r	Tx	1	03
1f1	DOOR_r	Tx	1	03
110	Gateway_1	Tx	3	87 00 00
1f1	DOOR_r	Tx	1	03

C) Synthèse

On donne les messages ci-dessous

```

227.214972  1  111  Gateway_2      Tx  8  4c 13 01 20 14 06 f4 01
├── Voltage      50 V      [ 1f4]
├── PetrolLevel   6          [    6]
├── EngSpeed      5152     [ 1420]
├── CarSpeed      137.5    [ 113]
├── EngineTemp    76       [ 4c]
+ 116.930236  1  604  TP_ConsoleMessage Tx  8  05 05 53 57 52 20 33 00
+ 116.930236  1  <OTP>      Tx  5  53 57 52 20 33
+ 228.300132  1  1a1  Console_2      Tx  2  02 01

375.014970  1  111  Gateway_2      Tx  8  4c c8 00 4c 1d eb f4 01
├── Voltage      50 V      [ 1f4]
├── PetrolLevel   235      [ eb]
├── EngSpeed      7500     [ 1d4c]
├── CarSpeed      100      [ c8]
├── EngineTemp    76       [ 4c]
+ 366.740944  1  604  TP_ConsoleMessage Tx  8  05 22 72 56 65 63 74 6f
    
```

La vitesse du véhicule (CarSpeed) peut être déterminée à partir de la relation : $CarSpeed = k.Tx$

Q10) Déterminez le coefficient k.

A la date **116.930236** le conducteur a choisi la station SWR 3.

On donne ci-dessous la table des caractères ASCII. (Ex de lecture : 'A' = 41₍₁₆₎)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	DLE	SP	0	@	P	'	p				°	À	Ð	à	ð
1	SOH	DC1	!	1	A	Q	a	q			ı	±	Á	Ñ	á	ñ
2	STX	REP	"	2	B	R	b	r	,		ı	²	Â	Ò	â	ò
3	ETX	SEP	#	3	C	S	c	s	, f		ı	³	Ã	Ó	ã	ó
4	EOT	DC4	\$	4	D	T	d	t	"		ı	'	Ä	Ô	ä	ô
5	ENQ	NAK	%	5	E	U	e	u	...	•	ı	μ	Å	Õ	å	õ
6	ACK	SYN	&	6	F	V	f	v	†	—	ı	¶	Æ	Ö	æ	ö
7	BEL	ETB	'	7	G	W	g	w	‡	—	ı	·	Ç	×	ç	÷
8	BS	CAN	(8	H	X	h	x			ı	˙	È	Ø	è	ø
9	HT	SS2)	9	I	Y	i	y	%	™	ı	°	É	Ù	é	ù
A	LF	SUB	*	:	J	Z	j	z	‰	š	ı	°	Ê	Ú	ê	ú
B	VT	ESC	+	;	K	[k	{	÷	÷			Ë	Û	ë	û
C	FF	FS	,	<	L	\	l		œ	œ	ı	¼	Ì	Ü	ì	ü
D	CR	GS	-	=	M]	m	}	ı	ı	ı	½	Í	Ý	í	ý
E	SO	RS	.	>	N	^	n	~			ı	¾	Î	Þ	î	þ
F	SI	US	/	?	O	_	o	DEL	ÿ	ÿ	ı	¿	Ï	ß	ï	ÿ

Q11) Quelle est le nom de la station choisie à la date 366.740944 ?

L'arbitrage sur le bus CAN (bus arbitration)

Comme toutes les **stations** (nœuds CAN) sont connectées sur 2 fils (CAN H, CAN L), toutes les trames passent sur ces deux seuls fils. Or, à **tout moment**, chaque station est susceptible de transmettre un message. Aussi, il est très probable qu'à un moment donné plusieurs stations essaient de « parler » en même temps. Il est donc nécessaire de disposer d'un **mécanisme d'arbitrage**.

L'extrait ci-dessous correspond à une communication entre la console et la porte droite (Door_r). La première ligne encadrée correspond au début de la communication. La dernière ligne encadrée correspond à la fin. On s'aperçoit que le message Gateway_1 vient s'intercaler dans cette communication. Ce message est émis toutes les 100ms (**fonctionnement cyclique**). Il est donc normal qu'il essaie de passer à ce moment là. Cependant le bus étant occupé par Console_1 et Door_r il ne devrait pas « être là » à moins que l'arbitrage lui soit **favorable** !

+	✉	6741.593956	1	110	Gateway_1	Tx	3	87 00 00
+	✉	6741.693956	1	110	Gateway_1	Tx	3	87 00 00
+	✉	6741.793956	1	110	Gateway_1	Tx	3	87 00 00
+	✉	6741.893956	1	110	Gateway_1	Tx	3	87 00 00
+	✉	6741.900176	1	1a0	Console_1	Tx	4	00 00 00 40
+	✉	6741.901292	1	1f1	DOOR_r	Tx	1	0a
+	✉	6741.920176	1	1a0	Console_1	Tx	4	00 00 00 40
+	✉	6741.931290	1	1f1	DOOR_r	Tx	1	09
+	✉	6741.940176	1	1a0	Console_1	Tx	4	00 00 00 40
+	✉	6741.960176	1	1a0	Console_1	Tx	4	00 00 00 40
+	✉	6741.961290	1	1f1	DOOR_r	Tx	1	08
+	✉	6741.980176	1	1a0	Console_1	Tx	4	00 00 00 40
+	✉	6741.991292	1	1f1	DOOR_r	Tx	1	07
+	✉	6741.993956	1	110	Gateway_1	Tx	3	87 00 00
+	✉	6742.000176	1	1a0	Console_1	Tx	4	00 00 00 40
+	✉	6742.020180	1	1a0	Console_1	Tx	4	00 00 00 00
+	✉	6742.021292	1	1f1	DOOR_r	Tx	1	07

Q12) Expliquez pourquoi l'ID du message Gateway_1 lui permet de s'intercaler dans la communication établie entre la console et la porte droite.

Q13) Pourquoi les concepteurs des réseaux CAN du véhicule ont-ils rendu le message Gateway_1 plus prioritaire que les messages Console_1 et DOOR_r ?