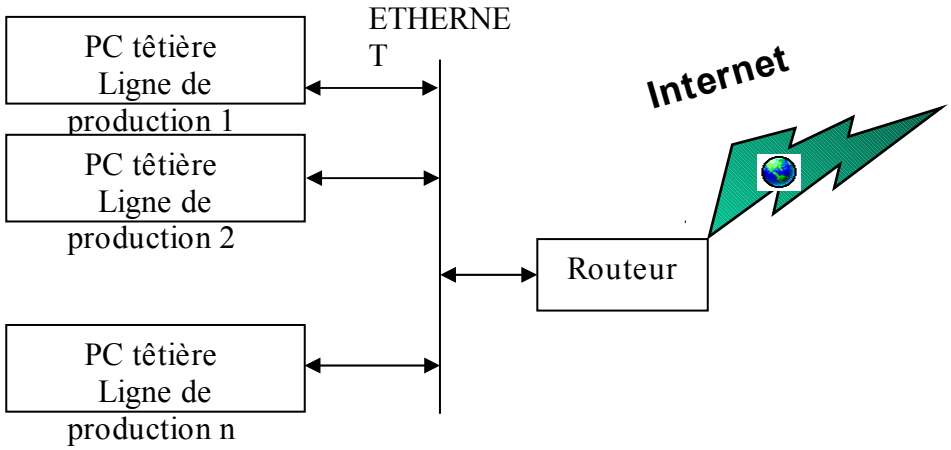
Sommaire

Modbus : Chaîne de bobinage textile

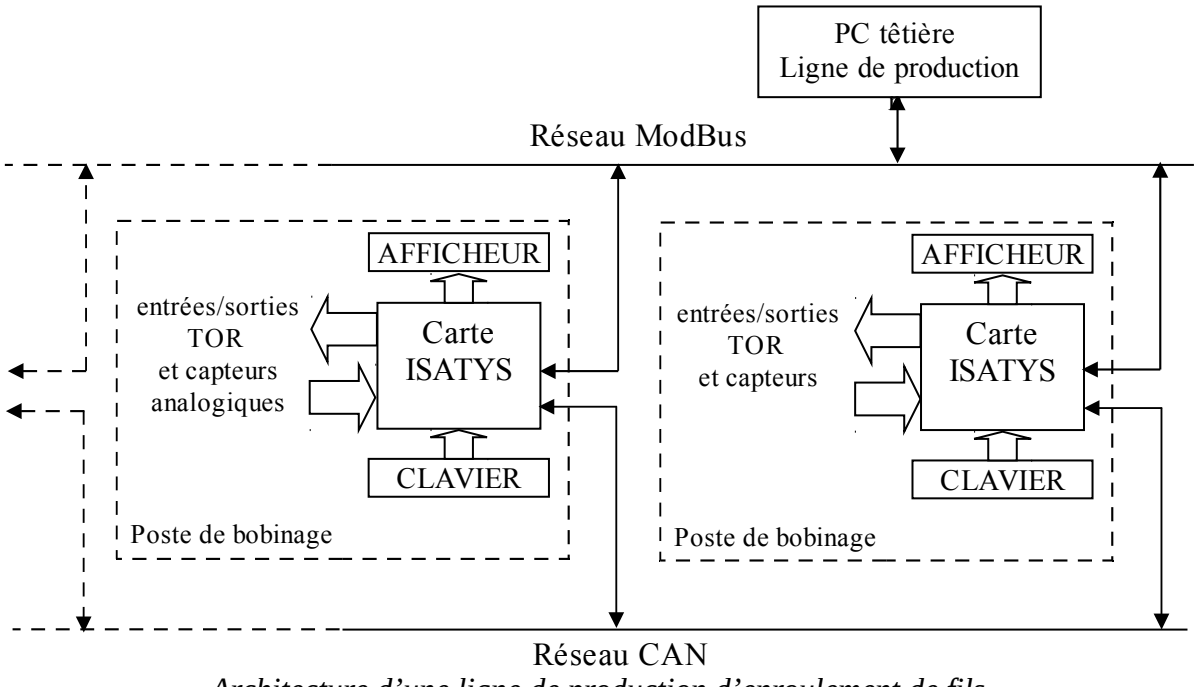
Depuis 1984, une société française fabrique des machines de production pour l'industrie du textile. L'utilisation de fibres aux propriétés variées a rendu les procédés de fabrication de plus en plus complexes. Les principaux débouchés sont l'habillement, l'ameublement, l'automobile, les pneumatiques, l'électronique, l'hygiène, l'aéronautique, la chimie, etc.

Dans les années 2000, sa restructuration industrielle a été menée après le rachat par un groupe européen, ce qui permit à l'entreprise d'aborder des marchés en Asie. Constituée de 70% cadres et techniciens, cette société a recruté de nombreux techniciens supérieurs pour constituer son bureau d'étude.

Une machine est organisée en **lignes de production** constituées de **postes de bobinage**.
Le PC (appelé têtière) qui pilote chaque ligne de production est un PC industriel type Pentium avec système d'exploitation **LINUX**. Il permet de gérer la ligne de production composée de plusieurs postes de bobinage, de remonter les informations pour la supervision et de communiquer par Internet avec le fournisseur pour la télémaintenance.



Chaque poste de bobinage est piloté par une carte à microcontrôleur (appelée carte ISATYS) reliée par **réseau** ModBus au PC têtière. Par ailleurs, les postes s'échangent des informations issues des capteurs grâce à un **réseau CAN**.



Architecture d'une ligne de production d'enroulement de fils

Étude des trames échangées entre le poste pilote et les cartes ISATYS

Lors de la remise à zéro des compteurs, on relève sur l'analyseur les trames suivantes :

Question du contrôleur : **01 10 07 E6 00 03 06 00 00 00 00 00 00 18 BD**

Réponse de l'esclave : **01 10 07 E6 00 03 60 8B**

Q1 . Interpréter ces trames relevées sur le ModBus ? Détailler les champs des trames en utilisant l'Annexe 1.

Réponses :

- Question du contrôleur : 01 10 07 E6 00 03 06 00 00 00 00 00 00 18 BD

01	
18 BD	CRC

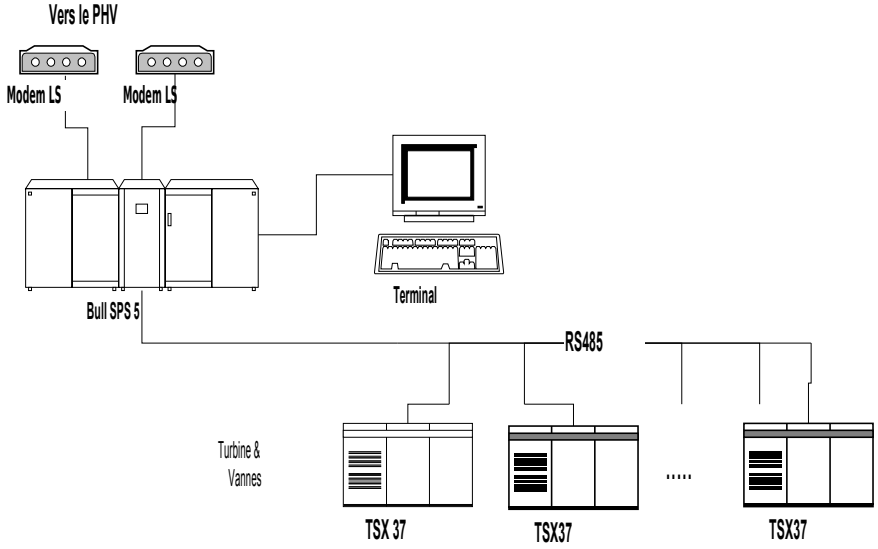
- Réponse de l'esclave : 01 10 07 E6 00 03 60 8B

01	

- Indiquer le nombre de compteurs 16 bits remis à zéro ? ___ compteurs

Modbus TCP

Un réseau Modbus relie le calculateur BULL et les API. La liaison est de type série RS485.
Le BULL est maître du bus, les API en sont les esclaves.



L'étude porte sur un échange de trames Ethernet (Modbus TCP) entre un maître MODBUS (BULL) et un esclave MODBUS (API). Les trames Ethernet suivantes ont été relevées à l'aide d'un logiciel de capture de trames :

LECTURE DU MOT %MW25 : QUESTION PC → API

```
Packet #5, Direction: Out, Time:11:27:45,065
Ethernet II
      Destination MAC: 00:80:F4:01:69:76
      Source MAC: 00:40:D0:4D:C2:89
      Ethertype: 0x0800 (2048) - IP
IP
      IP version: 0x04 (4)
      Header length: 0x05 (5) - 20 bytes
      Type of service: 0x00 (0)
      Total length: 0x002F (47)
      ID: 0x00F6 (246)
      Flags
            Don't fragment bit: 1 - Don't fragment
            More fragments bit: 0 - Last fragment
      Fragment offset: 0x0000 (0)
      Time to live: 0x80 (128)
      Protocol: 0x06 (6) - TCP
      Checksum: 0x9C17 (39959) - correct
      Source IP: 192.168.1.129
      Destination IP: 192.168.1.1
      IP Options: None
TCP
      Source port: 1062
      Destination port: 502
      Sequence: 0x3E544F4D (1045712717)
      Acknowledgement: 0x7EBC69D5 (2126277077)
      Header length: 0x05 (5) - 20 bytes
      Flags: PSH ACK
      Window: 0xFFE8 (65512)
      Checksum: 0xCCD1 (52433) - correct
      Urgent Pointer: 0x0000 (0)
      TCP Options: None
```

```
      Data length: 0xC (12)
Raw Data:
0x0000   00 80 F4 01 69 76 00 40 D0 4D C2 89 08 00 45 00
0x0010   00 2F 00 F6 40 00 80 06 9C 17 C0 A8 01 81 C0 A8
0x0020   01 01 04 26 01 F6 3E 54 4F 4D 7E BC 69 D5 50 18
0x0030   FF E8 CC D1 00 00 00 00 00 00 00 06 00 03 00 19
0x0040   00 01
```

LECTURE DU MOT %MW25 : RÉPONSE API → PC

```
Packet #11, Direction: In, Time:11:27:45,095
Ethernet II
      Destination MAC: 00:40:D0:4D:C2:89
      Source MAC: 00:80:F4:01:69:76
      Ethertype: 0x0800 (2048) - IP
IP
      IP version: 0x04 (4)
      Header length: 0x05 (5) - 20 bytes
      Type of service: 0x00 (0)
      Total length: 0x0033 (51)
      ID: 0x0040 (64)
      Flags
            Don't fragment bit: 0 - May fragment
            More fragments bit: 0 - Last fragment
      Fragment offset: 0x0000 (0)
      Time to live: 0x40 (64)
      Protocol: 0x06 (6) - TCP
      Checksum: 0x1CCA (7370) - correct
      Source IP: 192.168.1.1
      Destination IP: 192.168.1.129
      IP Options: None
TCP
      Source port: 502
      Destination port: 1062
      Sequence: 0x7EBC69D5 (2126277077)
      Acknowledgement: 0x3E544F58 (1045712728)
      Header length: 0x05 (5) - 20 bytes
      Flags: PSH ACK
      Window: 0x1000 (4096)
      Checksum: 0xC9A2 (51618) - correct
      Urgent Pointer: 0x0000 (0)
      TCP Options: None
      Data length: 0xB (11)
Raw Data:
0x0000   00 40 D0 4D C2 89 00 80 F4 01 69 76 08 00 45 00
0x0010   00 33 00 40 00 00 40 06 1C C0 A8 01 01 C0 A8
0x0020   01 81 01 F6 04 26 7E BC 69 D5 3E 54 4F 58 50 18
0x0030   10 00 C9 A2 00 00 00 00 00 00 00 05 00 03 01 00
0x0040   FA
```

Q2 . Remplir les champs contenus dans le tableau ci-dessous en vous aidant de l'échange « Réponse API → PC ».

Réponses :

Adresse IP source :		**Port Source :**	
Adresse IP destination :		**Port Destination :**	

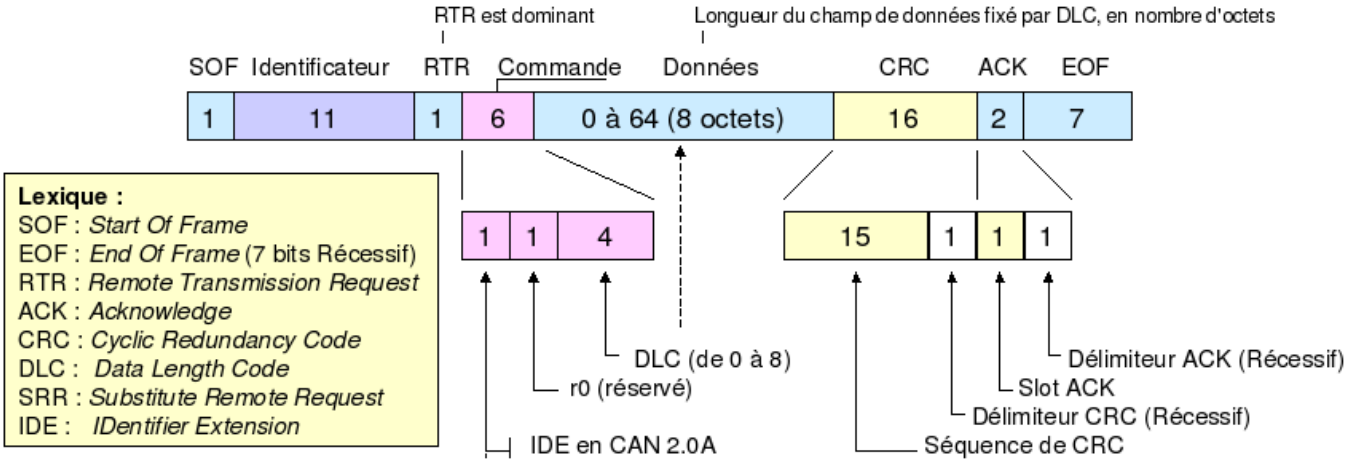
Q3 . Remplir les champs contenus dans le tableau ci-dessous en vous aidant de l'échange PC → API et du protocole Modbus sur TCP/IP décrit dans l'Annexe 2.

Réponses :

CHAMP (MODBUS)	VALEUR	CHAMP (MODBUS)	VALEUR	CHAMP (MODBUS)	VALEUR
Identificateur de transaction		**Unit Identifier**		**Nombre de mots lus**	
Identificateur de protocole		**Code requête**		**Numéro du mot lu**	
Longueur					

Bus CAN

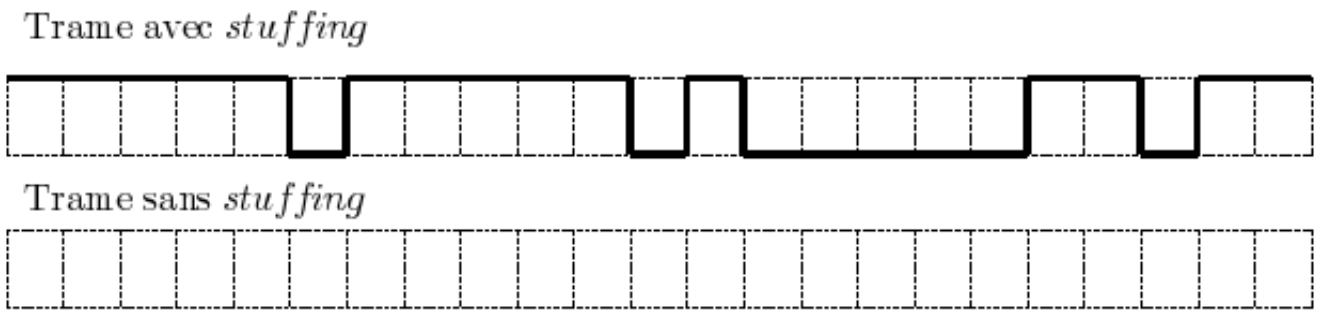
La **trame standard** (format standard - **CAN 2.0A**) possède un **identificateur de 11 bits.** La structure d'un message CAN est la suivante :



Q4 . Calculer la durée maximale de transmission d'une trame CAN standard sur un réseau à 125 kbits/s.
Réponse :

La méthode du ***Bit-Stuffing*** 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.

Q5 . Compléter le diagramme suivant en respectant la méthode de *Bit-Stuffing*.
Réponse :



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**.

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

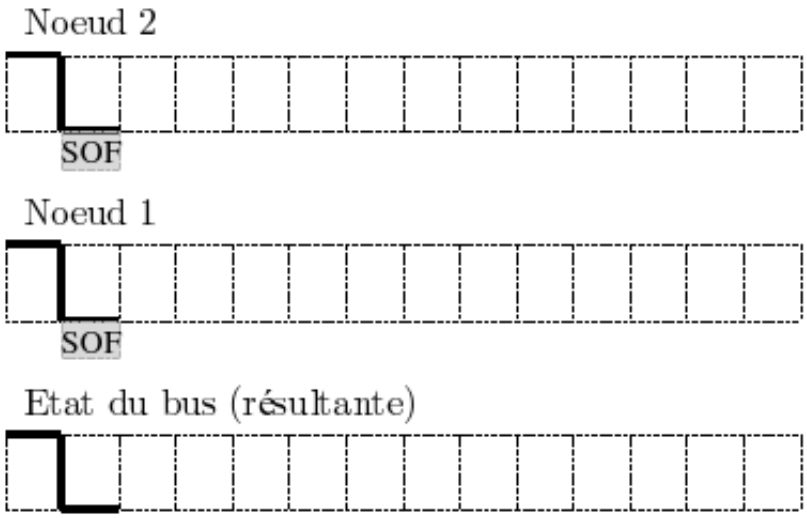
celui des messages concurrents. Les stations de priorité moins élevée perdront la compétition face à celle qui a la priorité la plus élevée.

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 (état dominant) écrase la valeur 1 (état récessif)**.

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

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, libère la ligne et ne doit plus envoyer aucun bit.

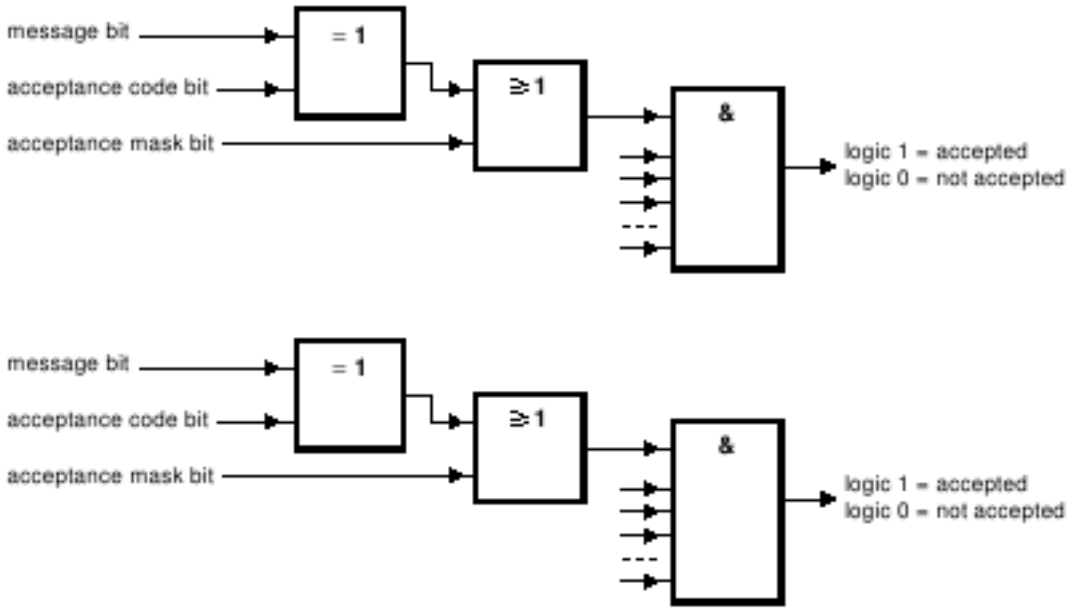
Q6 . A un instant donné, le bus devient libre et 2 trames d'identificateur 0x01F (trame de données) et 0x01D (trame de requête), émises respectivement par les noeuds 1 et 2, sont en concurrence.
a . Représenter les bits émis par les noeuds 1 et 2 et le niveau résultant sur le bus.
b . Indiquer le noeud qui remporte l'arbitrage et le moment où il le gagne.
Réponses :



Le noeud ____ gagne le bus.

Le **filtre d'acceptation** (*Acceptance Filter*) a pour rôle de contrôler les identificateurs (*identifiers*) des messages présents sur le bus avant de les laisser entrer dans le tampon (buffer) de réception. Une bonne utilisation de ce filtre permet d'éviter une saturation du buffer de réception. Le filtre d'acceptation est constitué de deux registres : *Acceptance Code Register* et *Acceptance Mask Register*.

Le contrôleur de bus CAN Philips **SJA1000** fonctionne de la manière suivante :



Acceptance Mask Bit	*Acceptance Code Bit*	*ID Bit*	*Accepté/Rejeté*
1	X	X	**Accepté**
0	**0**	**0**	**Accepté**
0	0	1	Rejeté
0	1	0	Rejeté
0	**1**	**1**	**Accepté**

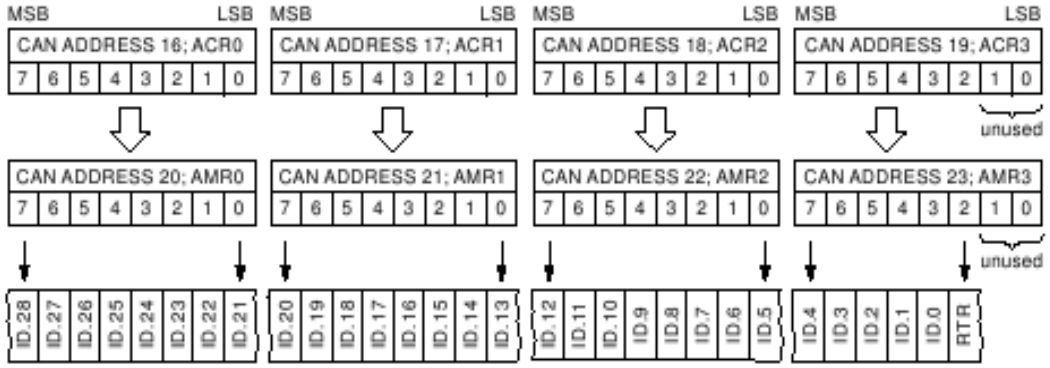
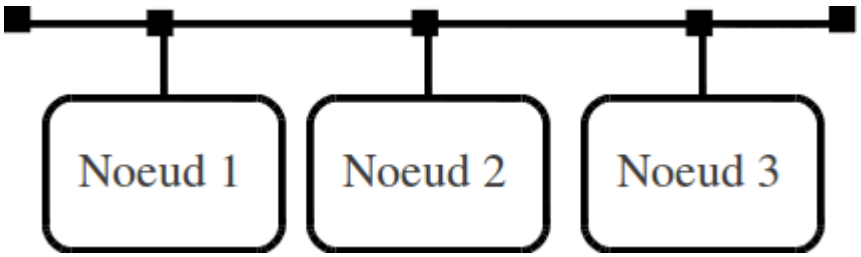


Fig.10 Single filter configuration, receiving extended frame messages.

ACR = Acceptance Code Register
AMR = Acceptance Mask Register

Pour la messagerie CAN suivante, les nœuds ne traiteront en réception que les identifiants indiqués dans ce tableau :

Noeuds		ID	DATAS							
Noeud 1	←	0x101	41	55	42	45	52	54		
Noeud 1	←	0x102	42	4F	4E	4E	45	54		
Noeud 1	←	0x105	45	43	48	45	56	49	4E	
Noeud 1	←	0x106	46	4F	55	52	4E	49	45	52
Noeud 1	←	0x107	47	41	4D	42	4F	41	4E	
Noeud 2	←	0x110	47	52	4F	53	59			
Noeud 2	←	0x114	4C	45	50	41	47	45		
Noeud 2	←	0x115	4D	45	45	52	54			
Noeud 3	←	0x120	50	41	4C	4D	41	53		
Noeud 3	←	0x122	52	49	47	41	55	44		
Noeud 3	←	0x124	54	52	4F	43	4D	45		



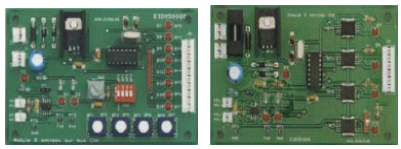
Q7 . Calculer le code et le masque d'acceptation pour les nœuds 1, 2 et 3 en complétant le tableau ci-dessous.

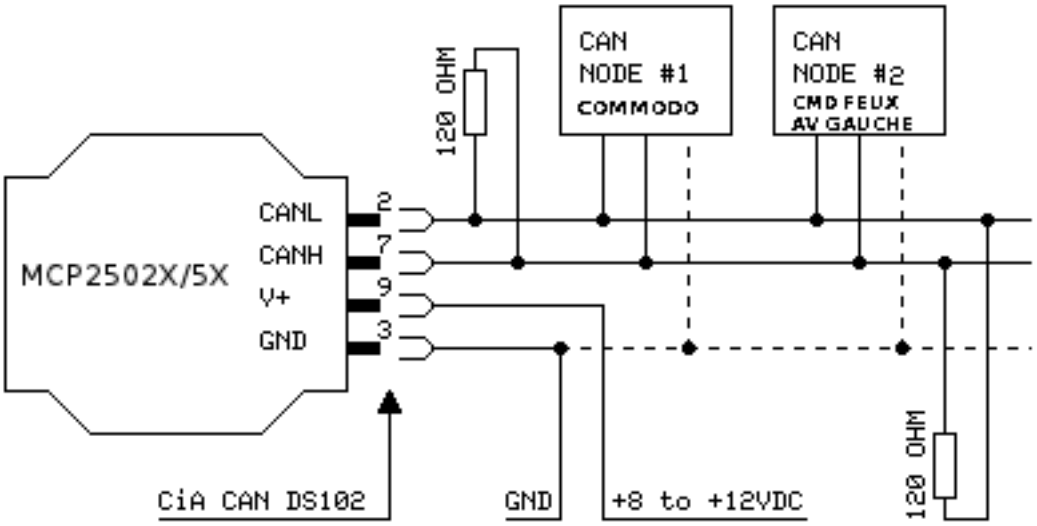
Réponses :

Noeuds	*Acceptance Mask Register*		*Acceptance Code Register*	
	AMR2	AMR3	ACR2	ACR3
Noeud 1				
Noeud 2				
Noeud 3				

Pilotage d'un bus CAN automobile

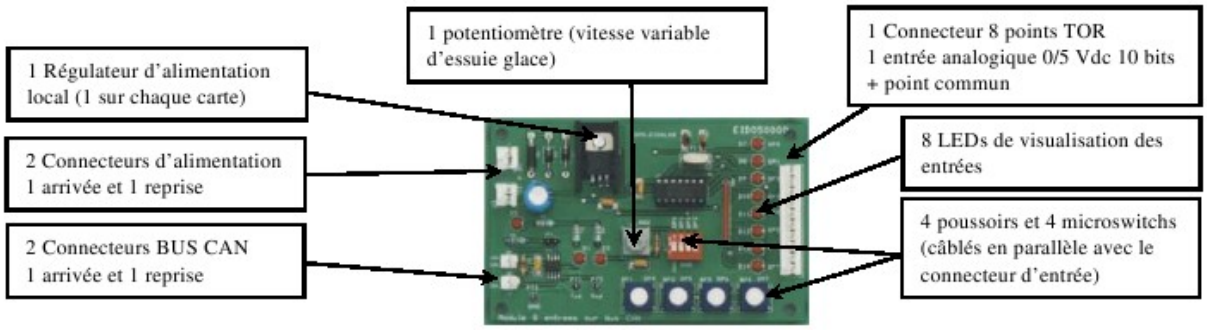
L'étude ne porte que sur les noeuds « Commodo » et « Commande des feux avant gauche » :





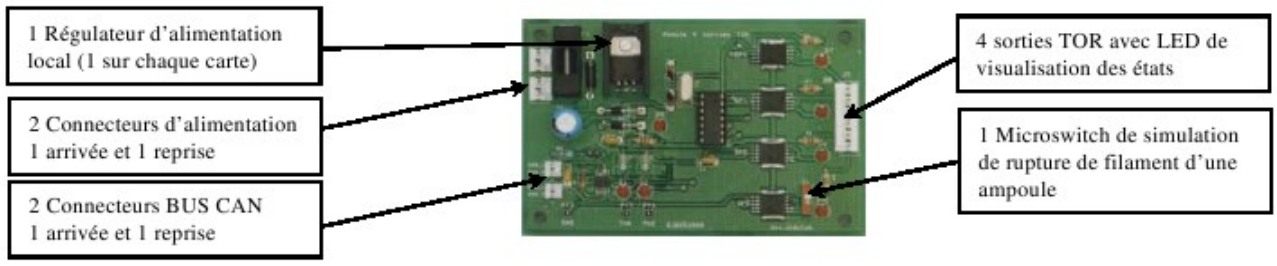
L'affectation des 8 entrées de la carte ***Commodo Lumière*** est la suivante:

GP7	*GP6*	*GP5*	*GP4*	*GP3*	*GP2*	*GP1*	*GP0*
KLAXON	**STOP**	**CLIGNOTANT DROIT**	**CLIGNOTANT GAUCHE**	**CODE**	**PHARE**	**WARNING**	**VEILLEUSE**



L'affectation des 8 E/S de la carte ***Feux Avant Gauche*** est la suivante:

GP7	*GP6*	*GP5*	*GP4*	*GP3*	*GP2*	*GP1*	*GP0*
ETAT CLIGNOTANT	**ETAT PHARE**	**ETAT CODE**	**ETAT VEILLEUSE**	**CLIGNOTANT**	**PHARE**	**CODE**	**VEILLEUSE**



Les noeuds « Commodo » et « Commande des feux avant gauche » utilisent aussi le contrôleur CAN MCP2502X/5X.

Ce contrôleur possède deux buffers de réception qui lui permettent de recevoir deux types de messages :

- **Information Request Messages** (**IRM**) dans le buffer RXF0 : messages de demande
- **Input Messages** (**IM**) dans le buffer RXF1 : utilisés pour écrire dans des registres du contrôleur

Le contrôleur CAN enverra des messages **Output Messages** (**OM**) en réponse à des messages du type IRM en utilisant le même identifiant.

Le contrôleur CAN MCP2502X/5X est capable d'émettre trois types d'identifiants différents :

- Transmit Message ID0 (TXID0) : contient l'identifiant utilisé pour envoyer des messages "**On Bus**". Ces messages sont envoyés a des intervalles prédéfinis et contiennent les états du contrôleur.
- Transmit Message ID1 (TXID1) : cet identifiant est utilisé pour envoyer des messages "**Command Acknowledge**" (acquittement en réponse à un message IM), "Receive Overflow" et "Error Condition".
- Transmit Message ID2 (TXID2) : contient l'identifiant utilisé pour envoyer des messages de conversion cyclique Analogique-Numérique, changement d'état sur une entrée numérique ou dépassement de seuil sur une entrée analogique.

Remarque : les messages TXID0 et TXID2 peuvent être envoyés automatiquement par le contrôleur CAN.

En résumé, pour communiquer avec un noeud CAN, on distinguera 3 types de messages :
- messages de demande d'information (Information Request Messages) qui permettent de demander au circuit de renvoyer la valeur d'un ou plusieurs de ses registres internes (par exemple pour lire l'état de ses entrées),
- messages d'entrée (Input Messages) qui permettent de modifier la valeur d'un ou plusieurs registres internes (par exemple pour modifier l'état de ses sorties),
- messages de sortie (Output messages) qui permettent de répondre à un message d'interrogation.

Les identifiants étendus sont définis de la manière suivante :

28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	P	P	N	N	N	M	M	M	M	M	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Les 3 bits (PPP) définissent la Priorité :
000 : "Asservissement" (Priorité la plus haute)
001 : "Commodo" (lumière ou essuie glace)
010 : "Afficheur clavier"
011 : "Feux"

Les 3 bits (NNN) définissent le type de Noeud :
001 : Noeud "Asservissement"
010 : Noeud "Commodo lumière"
011 : Noeud "Commodo essuie glace"
100 : Noeud "Feux avant gauche"
101 : Noeud "Feux avant droit"
110 : Noeud "Feux arrière gauche"
111 : Noeud "Feux arrière droit"

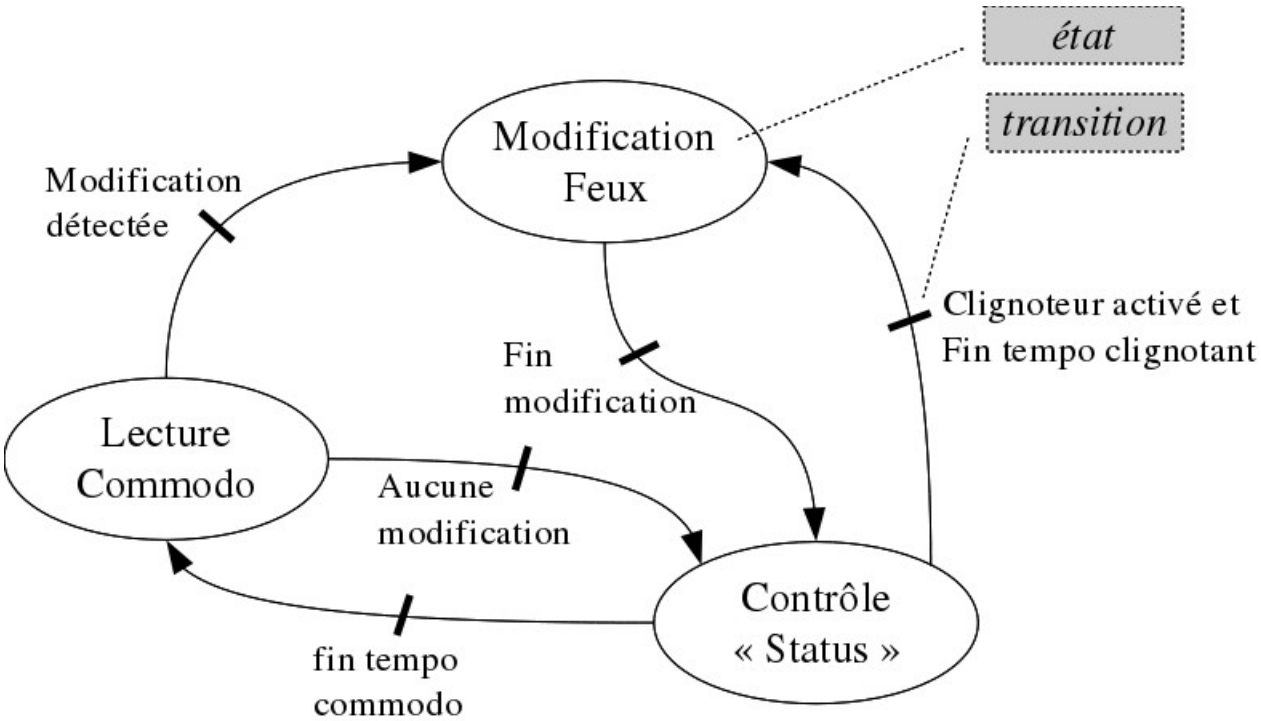
Les 5 bits (MMMMM) définissent le type de Message:
00001 : repère RXF0 -> Réception d'une IRM (Information Request Message) et réponse OM (Output Message).
00010 : repère RXF1 -> Pour écrire dans un registre (Input Message)
00100 : repère TXID0 -> Pour On bus message (au démarrage pour informer présence noeud)
01000 : repère TXID1 -> Pour acquitement suite à un IM (Input Message)
10000 : repère TXID2 -> Pour messages automatiques (ex: conversions cyclique Analogique-Numérique)

Cahier des charges : A intervalles de temps réguliers, on interroge le module sur lequel est connecté le commodo lumière afin de connaître son état. En fonction de l'état du commodo lumière, on active les différentes lampes des blocs optiques avant et arrière.

Après analyse du cahier des charges, les tâches à réaliser sont :

- Tâche principale : On envisage un fonctionnement cyclique où l'état du commodo est demandé à intervalles de temps réguliers, imposés par une base de temps. L'état recueilli du commodo est alors comparé à l'état précédent. Si un changement de position a été détecté, une phase de changement des états feux débute alors et on commande successivement les 4 blocs optiques avec les nouvelles valeurs.
- Tâches secondaires : On interroge successivement les 4 blocs optiques et on vérifie si les valeurs des "Status" corroborent les commandes envoyées. Si une différence est détectée, un message d'alerte sera affiché. Suite à l'envoi d'une trame, on vérifiera que la trame reçue en réponse est correcte : acquittement du module ayant reçu une trame de commande, ou réponse cohérente du module ayant reçu une trame de interrogative.

On peut synthétiser le fonctionnement global par le diagramme d'état-transition (DET) suivant :



Etat "Modification feux" : il s'agit d'envoyer une trame de commande (trame de type IM) à chacun des blocs optiques. On décide d'envoyer les trames de commande dans l'ordre suivant : Feux aVant Gauche (FVG), puis Feux aVant Droit (FVD), Feux aRrière Gauche (FRG), et enfin Feux aRrière Droit (FRD).

Etat "Lecture commodo" : il s'agit d'envoyer une trame interrogative (trame de type IRM) au module 8 entrées sur lequel est connecté le commodo. L'analyse de la trame réponse et la comparaison avec l'état mis en mémoire permet de détecter un changement éventuel.

Etat "Contrôle status" : il s'agit d'envoyer des trames interrogatives (trame de type IRM) aux différents modules 4 sorties de puissance sur lesquels sont connecter les blocs optiques.

Pour commander le bloc optique « Feux avant gauche », il faut envoyer message IM (fonction Write Register) vers le noeud « Feux Avant Gauche » contenant les 3 octets (DLC=3) suivants :

Registre (GPLAT)	***Masque***	***Valeur (4 sorties TOR)***
0x1E	**0x0x**	**0x0X**

Le ***Masque* x** permet d'indiquer les sorties concernées par la valeur de la commande. On positionne les bits à 1 si on désire modifier la sortie correspondante et, à 0 si on ne veut pas modifier la valeur courante. La ***Valeur* X** contient la commande des n sorties.

Q8 . Compléter le tableau ci-dessous pour chaque trame CAN permettant de commander les sorties de la carte Feux Avant Gauche.

Remarque : les bits ID_0 à ID_17 sont positionnés à 0 pour former un identifiant étendu.

Réponses :

VEILLEUSE	CODE	PHARE	CLIGNOTANT	ID	DLC	Registre	Masque	Valeur
0	0	0	1		3	0x1E		
0	0	1	x		3	0x1E		
1	0	0	0		3	0x1E		
0	0	0	x		3	0x1E		

Pour obtenir les états de la carte Commodo Lumière, il y a 3 situations possibles qui sont décrites en Annexe 3.

Q9 . Établir la trame CAN nécessaire pour tester l'acquisition des entrées de la carte *Commodo Lumière* pour le mode suivant : on désire un envoi cyclique des états toutes les 5 secondes. Compléter le tableau ci-dessous en utilisant l'Annexe 3.

Réponses :

ID	***DLC***	***Data 0***	***Data 1***	***Data 3***

Q10 . Indiquer, dans le tableau ci-dessous, l'état des 8 entrées de la carte Commodo Lumière pour la trame reçue suivante :

ID	*DLC*	*1 IOINTFL*	*2 GPIO*	*3 AN0H*	*4 AN1H*	*5 AN0L*	*6 AN2H*	*7 AN3H*	*8 AN32L*
0x05100000	8	0x00	0x08	0x00	0x00	0x00	0x00	0x00	0x00

Réponses :

GP7	*GP6*	*GP5*	*GP4*	*GP3*	*GP2*	*GP1*	*GP0*
KLAXON	**STOP**	**CLIGNOTANT DROIT**	**CLIGNOTANT GAUCHE**	**CODE**	**PHARE**	**WARNING**	**VEILLEUSE**

Q11 . Vérifier que l'identifiant reçue provient bien d'un message envoyé par le noeud « Commodo Lumière » en complétant le tableau ci-dessous.

Réponses :

Id = 0x05100000	*Valeur en bits*	*Signification*
Priorité (PPP)		
Noeud (NNN)		
Type de message (MMMMM)		

Annexe 1 : Protocole Modbus

Le schéma qui suit montre l'ordre définissant la syntaxe de la trame du message utilisée par JBUS et MODBUS :

Début de la trame	Adresse du périphérique	Code fonction	Données accompagnant la fonction	Données de détection d'erreur	Fin de la trame
	1 octet	1 octet	n octets	2 octets	

Le début de la trame est une période d'inactivité égale à au moins 3,5 fois la durée de transmission d'un caractère unique. Par exemple, à 9600 bauds, un caractère comportant un bit de départ, un bit d'arrêt et 8 bits de données a besoin d'un début de trame de 3,5 millisecondes. Cette période est la fin de transmission implicite d'une transmission antérieure.

- L'adresse du périphérique est un seul octet (8 bits) propre à chaque périphérique du réseau.
- Les codes fonction sont une instruction à un seul octet destinée à l'esclave et décrivant l'action à exécuter.
- Le segment de données d'un message dépend du code fonction et le nombre d'octets varie en conséquence.
- En règle générale, le segment de données contient une adresse de paramètres et le nombre de paramètres à lire ou écrire.
- Le contrôle de redondance cyclique est un code de détection d'erreur qui a une longueur de 2 octets.

Le segment Fin de la transmission est une période d'inactivité égale à 3,5 fois la durée de transmission d'un seul caractère. Le segment Fin de la transmission à la fin d'un message indique à l'appareil récepteur que la transmission suivante sera un nouveau message et par conséquent un caractère d'adresse de périphérique.

Extrait des fonctions de ModBus : ECRITURE DE N MOTS

Code fonction : **16, (10h)**

Commande :

Adresse du périphérique	Code fonction 10	Adresse du premier mot		Nombre de mots à écrire		Nombre d'octets de données (n)	Données	CRC	
1 octet	1 octet	octet MSB	octet LSB	octet MSB	octet LSB	1 octet	n octets	octet MSB	octet LSB

Le nombre maximal de mots qui peuvent être transmis est de 125 mots, ce qui correspond à 250 octets de données.

Les deux premiers octets sont des données qui comportent la valeur exigée du premier paramètre, le bit de poids fort étant le premier. Les paires suivantes d'octets sont des données pour les adresses de paramètres consécutives.

Réponse :

Adresse du périphérique	Code fonction 10	Adresse du premier mot		Nombre de mots écrits		CRC	
1 octet	1 octet	octet MSB	octet LSB	octet MSB	octet LSB	octet MSB	octet LSB

Exemple :

Commande : Ecrire dans l'esclave situé à l'adresse 2.

Consigne 1 = 379 à l'adresse 164
Consigne 2 = 918 à l'adresse 165
Consigne 3 = 250 à l'adresse 166

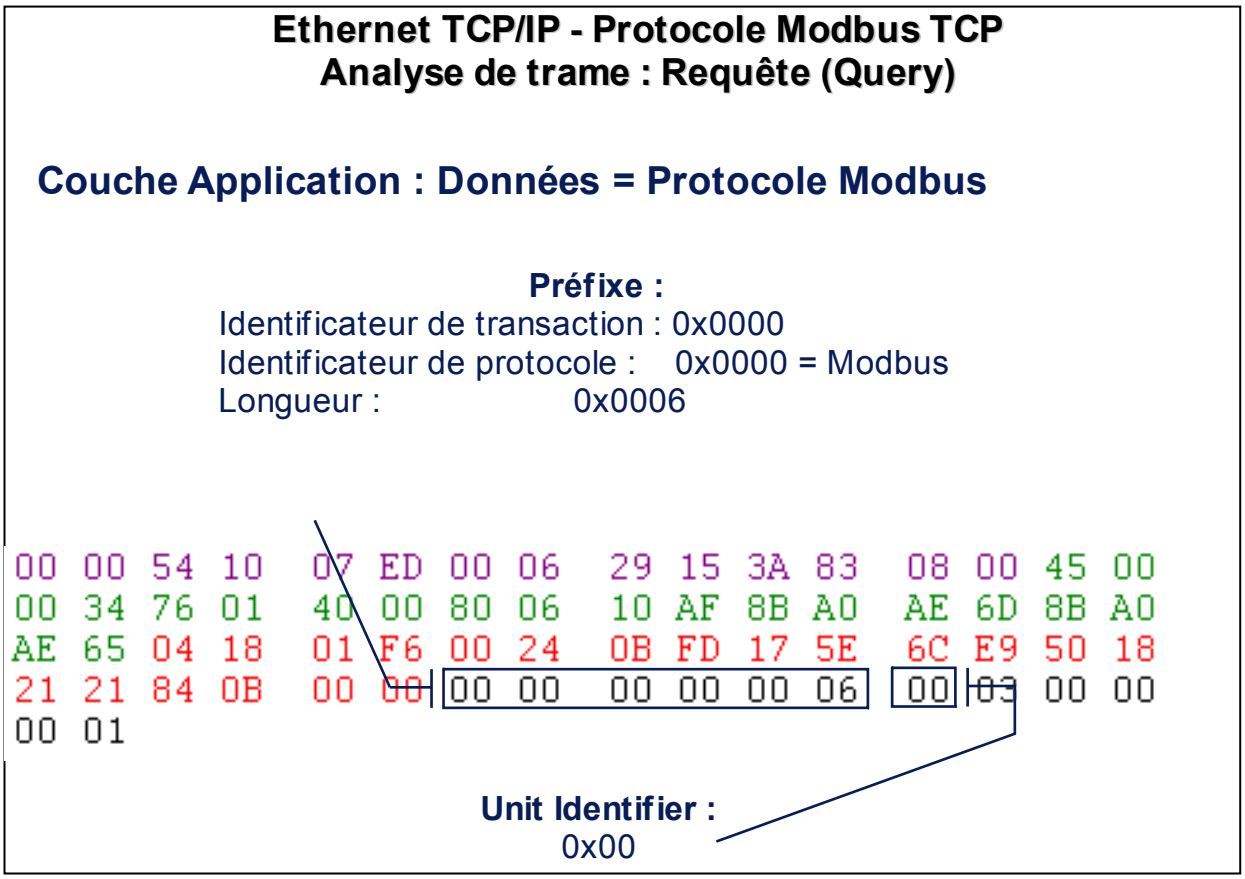
Adresse du périphérique	Code fonction	Adresse du premier mot		Nombre de mots à écrire		Nombre d'octets de données	Données	CRC	
02	10	00	A4	00	03	06	Cf. ci-dessous	20	71

Données (379) pour l'adresse 164		Données (918) pour l'adresse 165		Données (250) pour l'adresse166	
01	7B	03	96	00	FA

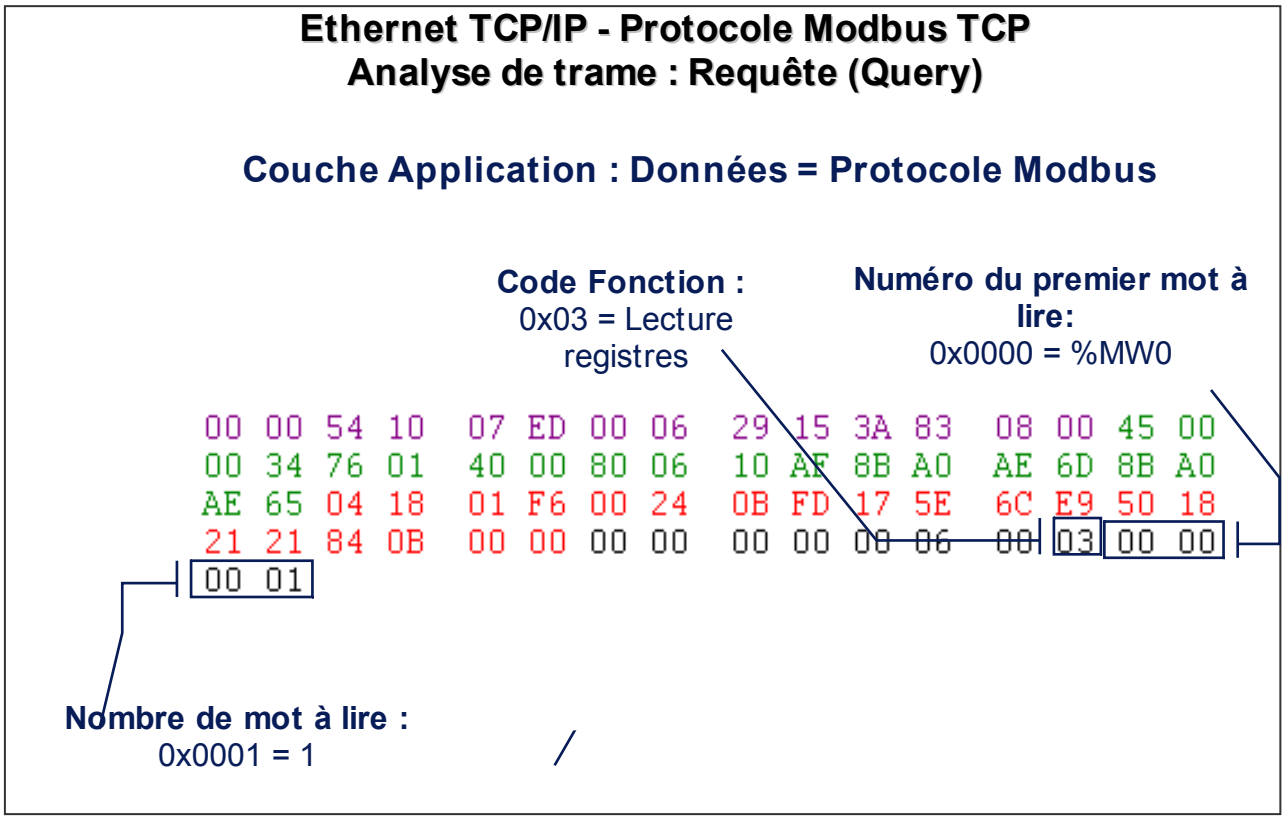
Réponse :

Adresse du périphérique	Code fonction	Adresse du premier mot		Nombre de mots écrits		CRC	
02	10	00	A4	00	03	C1	D8

Annexe 2 : Modbus TCP



Pour la réponse le champ indiquant« numéro du premier mot à lire »(sur deux octets) devient « nombre d'octets lus »(sur un octet) et celui « nombre de mot à lire »(sur deux octets) devient « valeur du mot lu »(sur deux octets).



Annexe 3 : Commodo Lumière

Pour obtenir les états de la carte Commodo Lumière, il y a plusieurs situations possibles:

- envoi des états sur demande (trame RTR ou non)
- envoi cyclique des états
- envoi des états déclenché par un changement d'état

- **Lire l'état des entrées de la carte *Commodo Lumière***

a . Il faut tout d'abord envoyer une trame CAN pour configurer ce mode avec l'Id étendu 0x05080000 (T_Ident_IM_Commodo_Feux) contenant les 3 octets (DLC=3) suivants :

Registre (OPTREG2)	*Masque (bit 3 = MTYPE)*	*Valeur*
0x2D	**0x08**	**0x08**

La valeur (page 35 du MCP2502X/5X) permettra de déterminer si on utilise le bit RTR (0x00) ou non (0x08).

b . Pour lire l'état des entrées, il faut ensuite envoyer une trame CAN avec l'Id étendu 0x05041E0F et avec 0 octet (DLC=0) de données. Le calcul de l'Id se fait de la manière suivante:

ID = ID IRM | (Registre << 8) | Fonction | NON-RTR
ID = T_Ident_IRM_Commodo_Feux | (GPLAT << 8) | Read_Register | BIT3_ON
ID = 0x05040000 | (0x1E << 8) | 0x07 | 0x08
ID = 0x05041E0F

- **Envoi cyclique des états de la carte *Commodo Lumière* toutes les x secondes**

a . Il faut tout d'abord envoyer une trame CAN pour configurer ce mode avec l'Id étendu 0x05080000 (T_Ident_IM_Commodo_Feux) contenant les 3 octets (DLC=3) suivants :

Registre (STCON)	*Masque*	*Valeur*
0x2C	**0xFF**	**0xXX**

Le registre **STCON** (*Scheduled Transmission CONtrol Register*) permet de déterminer la valeur 0xXX de la manière suivante (page 28 du MCP2502X/5X):

7	*6*	*5*	*4*	*3*	*2*	*1*	*0*
STEN	STMS	STBF1	STBF0	STM3	STM2	STM1	STM0
Enable bit 1 : Enabled 0 : Disabled	*Message Select* 1 : dlc=8 0 : no data	*Base Transmission Frequency bits* 00 : 4096xTOSC 01 : 16x(4096xTOSC) 10 : 256x(4096xTOSC) 11 : 4096x(4096xTOSC)		*Scheduled Transmission Multiplier bits* 0000 = 1 0001 = 2 ... 1111 = 16			

Pour obtenir un envoi cyclique toutes les 2 secondes, on configurera le registre STCON de la manière suivante:

7	*6*	*5*	*4*	*3*	*2*	*1*	*0*
STEN	STMS	STBF1	STBF0	STM3	STM2	STM1	STM0
1	1	1	1	0	0	0	1
1 : Enabled	1 : dlc=8	11 : 4096x(4096xTOSC) = 1,048 s		0001 = 2 x 1,048 = 2,097 s			

Pour désactiver l'envoi cyclique, on on configurera le registre STCON de la manière suivante:

7	*6*	*5*	*4*	*3*	*2*	*1*	*0*
STEN	STMS	STBF1	STBF0	STM3	STM2	STM1	STM0
0	0	1	1	0	0	0	1
0 : Disabled	0 : no data	11 : 4096x(4096xTOSC) = 1,048 s		0001 = 2 x 1,048 = 2,097 s			

b . La carte ***Commodo Lumière*** enverra un message T_Ident_OnBus_Commodo_Feux (0x05100000) avec l'état des entrées dans le deuxième octet reçu (GPIO).

- **Envoi déclenché par un changement d'état sur la carte *Commodo Lumière***

a . Il faut tout d'abord envoyer une trame CAN pour configurer ce mode avec l'Id étendu 0x05080000 (T_Ident_IM_Commodo_Feux) contenant les 3 octets (DLC=3) suivants :

Registre (IOINTEN)	***Masque***	***Valeur***
0x1C	**0xF0**	**0xXX**

Par exemple, pour les 4 BP, on mettra la valeur **0xF0** (GP7, GP6, GP5 et GP4 à 1 pour valider le *Transmit-On-Change*). ***Remarque:*** le type de *Transmit-On-Change* peut être configuré par les registres IOINTPO et IOINTFL (page 33 du MCP2502X/5X). Remettre les valeurs à 0 pour désactiver ce mode.

b . Réception des messages envoyés par la carte ***Commodo Lumière***.