UTBM **Département EE**

Printemps 2013 **Durée : 2 heures. Sans documents ni calculatrice**

NOM : **Signature :**

Prénom :

Examen SM57 : Systèmes embarqués

Véhicule à projecteurs directionnels

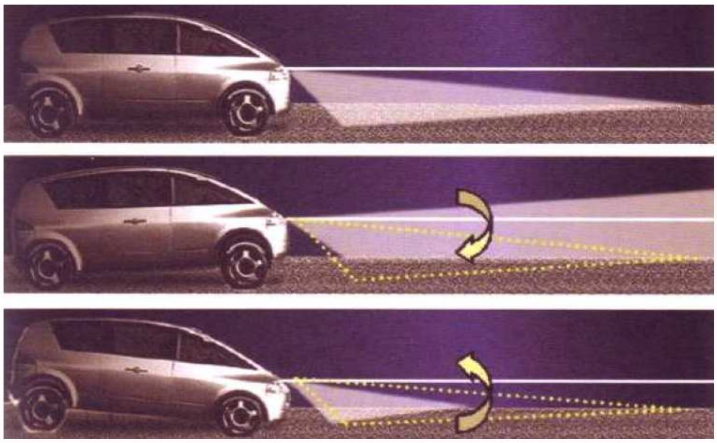
I Présentation

I.1 Introduction

La fonction « projecteurs directionnels » ou encore AFS (Adaptative Frontlighting System) doit apporter au conducteur de la lumière dans la zone où il porte son regard et vers ses proches alentours compte tenu des caractéristiques du véhicule (hauteur de caisse) et du virage qu'il aborde (sens, courbure, vitesse). Elle doit en outre se conformer à la réglementation concernant les projecteurs équipés de lampe à décharge. Par conséquent, elle adopte un dispositif de correction de site automatique.

I.2 Correction de site

La régulation de site a pour objet de garder un angle constant des faisceaux lumineux par rapport à l'horizontale malgré les variations d'assiette statiques (chargement) et dynamiques (freinage, accélération, irrégularités de la route) du véhicule. Le principe de correction de site est illustré sur la figure ci-dessous.



Faisceaux bien réglés
→Valeur de consigne

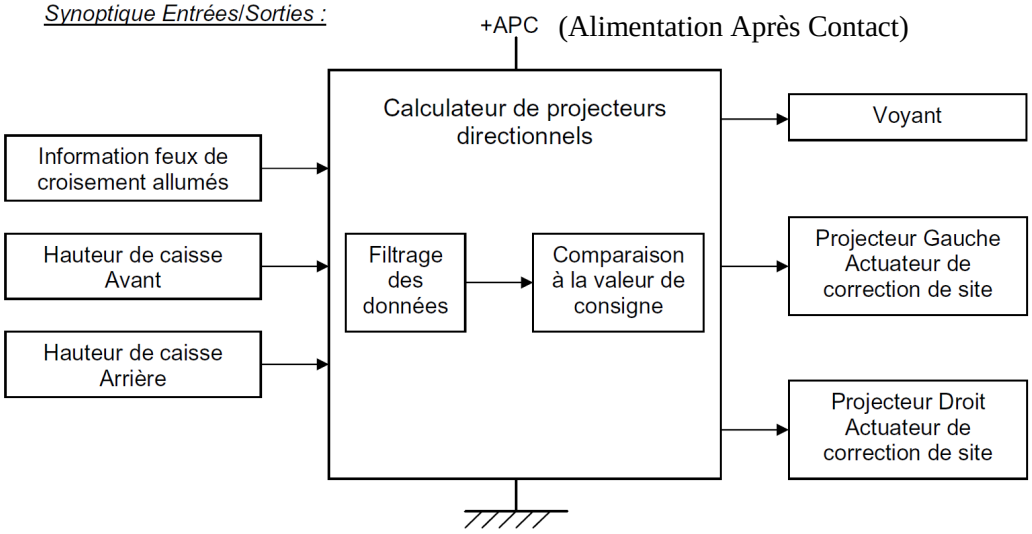
Faisceaux trop hauts
- Accélération
- Charge arrière

→Baisse du faisceau

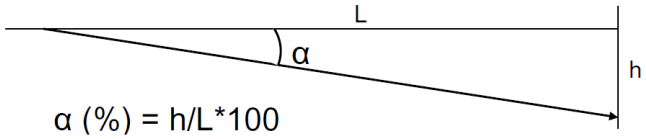
Faisceaux trop bas
- Freinage

→Montée du faisceau

Synoptique Entrées/Sorties :

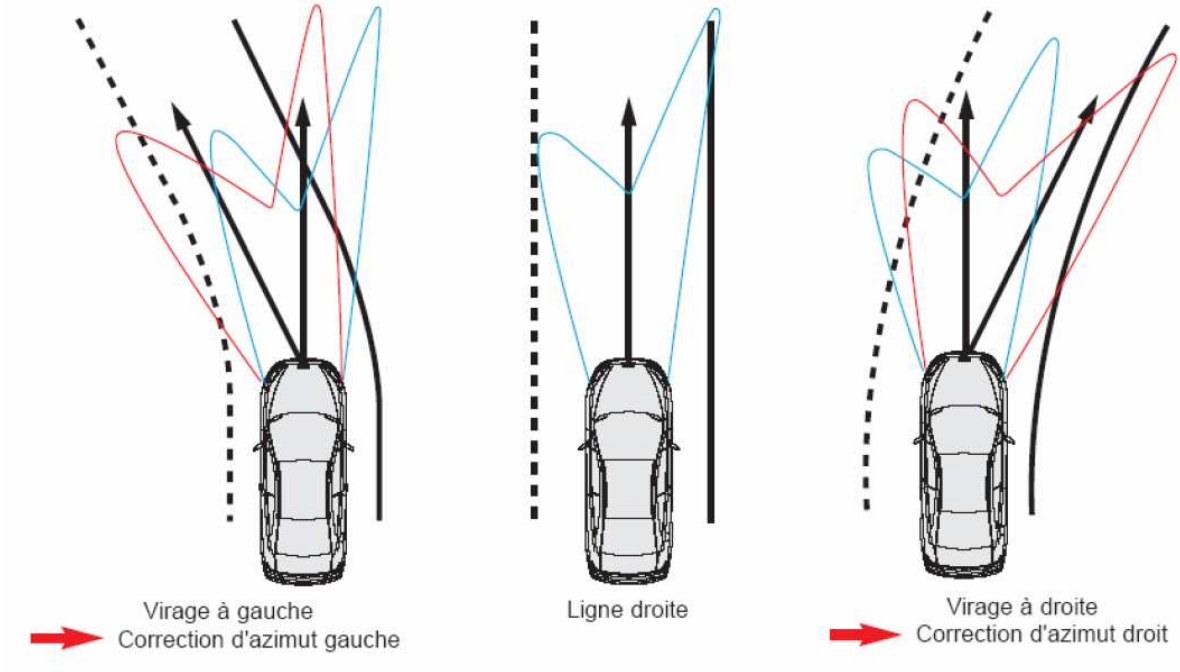


L'angle du faisceau lumineux par rapport à l'horizontale est exprimé en % à partir de la relation suivante :



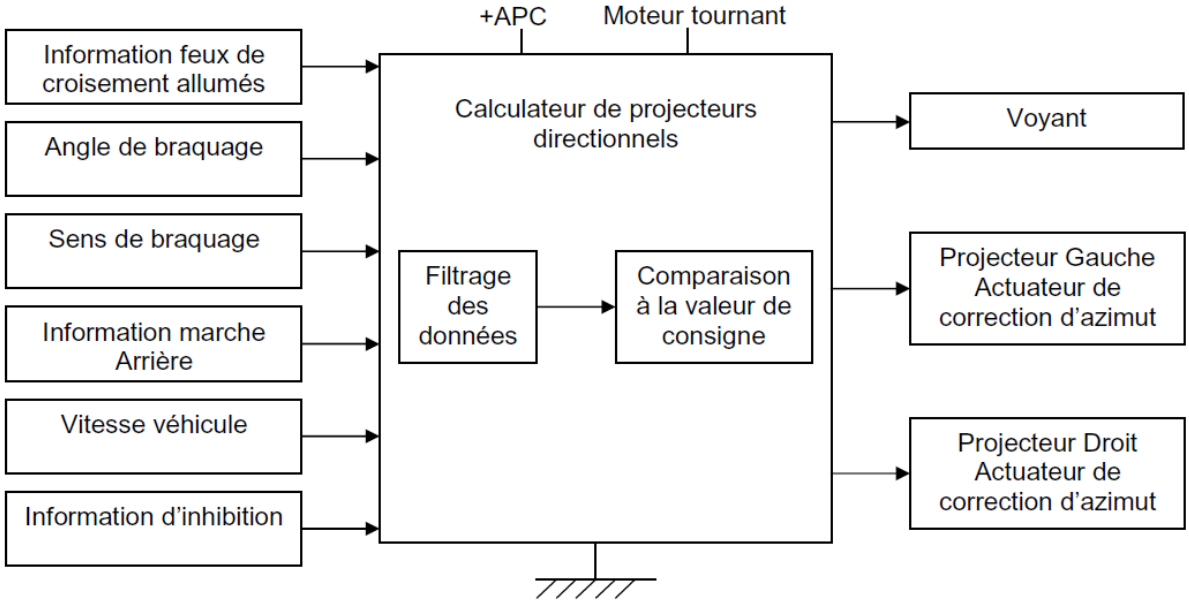
I.3 Correction d'azimut

La régulation d'azimut a pour objet de modifier l'angle du faisceau lumineux par rapport à l'axe longitudinal du véhicule, compte tenu du virage abordé.

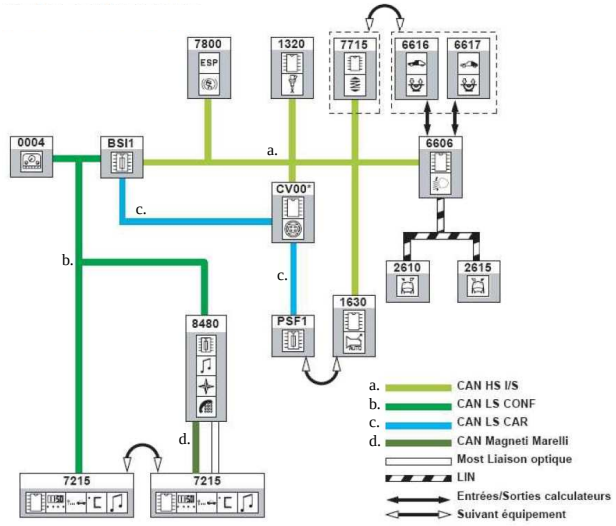


Ce système augmente le confort et la sécurité de conduite par une meilleure anticipation de la trajectoire du véhicule par le conducteur.

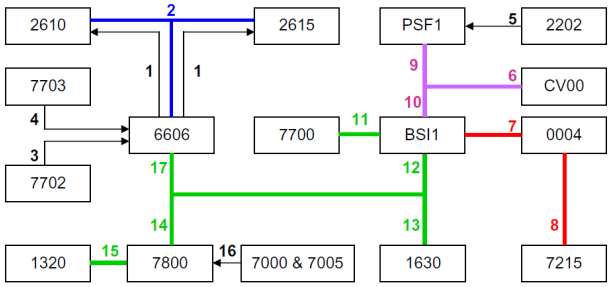
Le synoptique d'entrées sorties de la correction d'azimut est illustré ci-dessous :



I.4 Schémas synoptiques



Organes	
BSI1	Boîtier Servitude Intelligent
BSM	Boîtier Servitude Moteur
CV00	Module de commutation sous volant
PSF1	Platine servitude fusible compartiment moteur
0004	Combiné
1320	Calculateur moteur
1630	Calculateur boîte de vitesses automatique
2202	Contacteur de Marche AR (BVM)
2610	Projecteur gauche
2615	Projecteur droit
6606	Boîtier de correction dynamique des projecteurs
6616	Capteur hauteur de caisse avant
6617	Capteur hauteur de caisse arrière
7000 & 7005	Capteur antiblocage de roue AVG et AVD
7215	Ecran multifonctions
7700	Capteur d'angle volant
7702	Capteur hauteur de caisse avant
7703	Capteur hauteur de caisse arrière
7715	Calculateur suspension
7800	Calculateur ESP
8410	Autoradio
8480	Emetteur récepteur télématique



N°	Signal	Liaison
1	Commande de correction de site	Filaire
2	Commande de correction d'azimut	LIN
3	Information de hauteur de caisse avant	Filaire
4	Information de hauteur de caisse arrière	Filaire
5	Information Marche AR (BVM)	Filaire
6	Etat des commandes d'éclairage	CAN Car
7	Commande voyants Demande correction d'azimut	CAN Conf
8	Demande activation/inhibition correction d'azimut	CAN Conf
9	Information Marche AR (BVM)	CAN Car
10	Information Marche AR (BVM) / Etat des commandes d'éclairage	CAN Car
11	Information capteur d'angle volant	CAN I/S
12	Information capteur d'angle volant / Information Marche AR (BVM) Etat des commandes d'éclairage / Commande voyants Demande activation/inhibition correction d'azimut	CAN I/S
13	Information Marche AR (BVA)	CAN I/S
14	Info moteur tournant / Vitesse véhicule	CAN I/S
15	Information moteur tournant	CAN I/S
16	Information vitesse de rotation des roues avants	Filaire
17	Information capteur d'angle volant / Information Marche AR Etat des commandes d'éclairage / Commande voyants / Demande correction d'azimut / Info moteur tournant / Vitesse véhicule	CAN I/S

CAN Car : CAN Carrosserie
CAN I/S: CAN Intersystèmes
CAN Conf: CAN Confort

CV00 : Module de commutation sous volant (COM2003)
Il acquiert et informe le BSI sur les volontés du conducteur, au travers du réseau CAN LS CAR : allumage ou extinction des feux de croisement / route.

7700 : Capteur angle volant
Il acquiert et diffuse sur le réseau CAN HS I/S l'angle et le sens de rotation du volant à destination du boîtier AFS (6606).

PSF1 : Boitier de servitude moteur (BSM)
Il alimente sous les commandes du BSI1 les feux avant (positions, projecteurs bi-fonction, clignotants). Il acquiert puis communique au BSI1 l'information feux de recul (2200, BVM).

0004 : Combiné
Il affiche les témoins des feux.

1320 : Calculateur de contrôle moteur
Il diffuse l'information moteur tournant sur le réseau CAN HS I/S à destination du boîtier AFS (6606).

1630 : Calculateur BVA
Il diffuse l'information marche arrière enclenchée sur le réseau CAN HS I/S à destination du boîtier AFS (6606).

2610/2615 : Projecteur directionnel gauche/droit (POWER MODULE)
Ils communiquent avec le boîtier AFS (6606) et exécutent les corrections de site et d'azimut.

6606 : Boîtier de correction dynamique des projecteurs (AFS)
Il gère la fonction « projecteurs directionnels » (AFS) et également la correction de site
Il acquiert l'information hauteur de caisse soit :
- en filaire, donnée par les deux capteurs de hauteur de caisse (6616 / 6617)
- par le réseau CAN HS I/S, donnée par le calculateur CSS (suspension pilotée).

Il communique par une liaison LIN avec les « POWER MODULE » intégrés aux projecteurs.
Ceux-ci commandent les moteurs pas à pas de correction d'azimut

6616 / 6617 : Capteur de hauteur de caisse avant/arrière
Sur les véhicules sans calculateur de suspension piloté (7715), ils mesurent la hauteur de caisse à destination du boîtier AFS (6606) en filaire.

7215 : Ecran Multifonctions (EMF)
Affichage des menus pour l'activation ou l'inhibition de la fonction « projecteurs directionnels ».
Affichage des défauts.

7715 : Calculateur de suspension pilotée (CSS)
Il reçoit les informations de hauteur de caisse pour la suspension pilotée puis les diffuse sur le réseau CAN HS I/S à destination du boîtier AFS

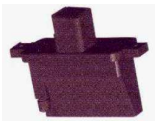
7800 : Calculateur ESP
Il diffuse l'information vitesse du véhicule sur le réseau CAN HS I/S à destination du boitier AFS (6606).

8480 : Emetteur récepteur radio téléphone / autoradio
Il acquiert et informe l'écran multifonctions des consignes du conducteur par le menu « Personnalisation - Configuration »

I.5 Description des composants

Le calculateur :

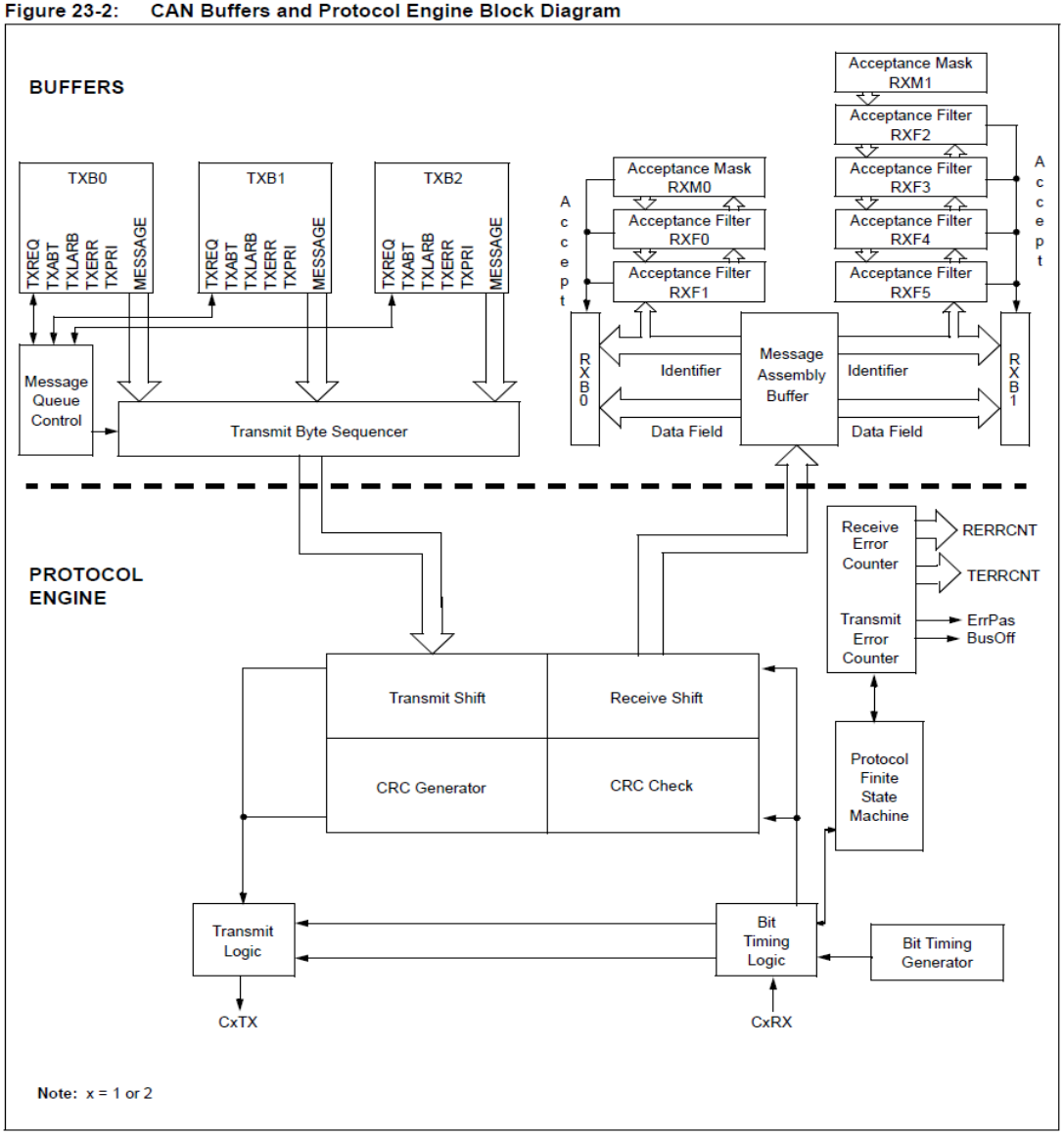
Le calculateur de correction dynamique des projecteurs (6606) traite les informations provenant du capteur d'angle volant (7700), des capteurs de hauteur de caisse (6616 et 6617), et des autres calculateurs et transmet les consignes de commandes aux modules de puissance pour les corrections de site et d'azimut (2610 et 2615).



Le contrôleur CAN intégré au calculateur 6606 est d'architecture Microchip :

The CAN bus module consists of a Protocol Engine and message buffering and control. The Protocol Engine can best be understood by defining the types of data frames to be transmitted and received by the module. These blocks are shown in Figure 23-2.

Figure 23-2: CAN Buffers and Protocol Engine Block Diagram



Filtres des messages et masques de filtrage (standard) :

Register 23-13: CiRXFnSID: Acceptance Filter n Standard Identifier

Upper Byte:							
U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	SID<10:6>				
bit 15							bit 8

Lower Byte:							
R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	U-0	R/W-x
SID<5:0>						—	EXIDE
bit 7							bit 0

bit 15-13 **Unimplemented:** Read as '0'

bit 12-2 **SID<10:0>:** Standard Identifier bits

bit 1 **Unimplemented:** Read as '0'

bit 0 **EXIDE:** Extended Identifier Enable bits
If MIDE = 1, then
1 = Enable filter for extended identifier
0 = Enable filter for standard identifier
If MIDE = 0, then EXIDE is don't care

Register 23-16: CiRXMnSID: Acceptance Filter Mask n Standard Identifier

Upper Byte:							
U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	SID<10:6>				
bit 15							bit 8

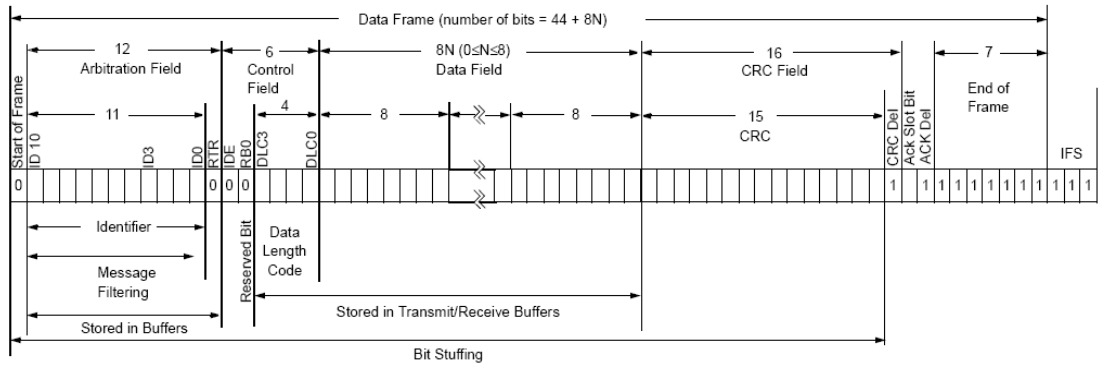
Lower Byte:							
R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	U-0	R/W-x
SID<5:0>						—	MIDE
bit 7							bit 0

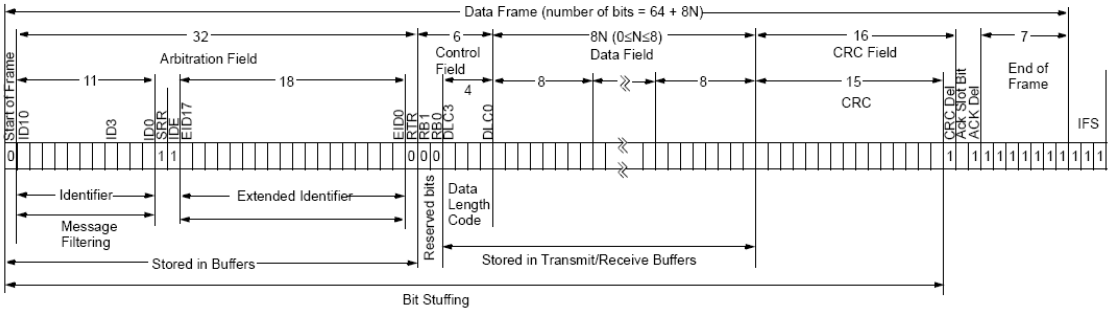
bit 15-13 **Unimplemented:** Read as '0'

bit 12-2 **SID<10:0>:** Standard Identifier Mask bits
1 = Include bit in the filter comparison
0 = Don't include bit in the filter comparison

bit 1 **Unimplemented:** Read as '0'

bit 0 **MIDE:** Identifier Mode Selection bit
1 = Match only message types (standard or extended address) as determined by EXIDE bit in filter
0 = Match either standard or extended address message if the filters match





Modules de puissance :

Ils sont fixés sous les projecteurs (2610/2615), Ils pilotent les moteurs pas à pas de site et d'azimut.

Capteurs de hauteur de caisse :

Ces capteurs sont reliés directement au calculateur de projecteurs directionnels. Ils sont identiques à ceux montés sur C8. Un télécodage permet d'indiquer quel type de capteur est monté. Les capteurs se détaillent en pièce de rechange.

Les capteurs angulaires sont fixés sur un élément de caisse du véhicule et sont reliés au bras de suspension à l'arrière et à l'avant sur la barre anti-devers par le biais d'un système de bielles. Ils sont destinés à mesurer l'assiette du véhicule, leur course est de +/- 45 degrés.

Ils sont de type numérique, et ils délivrent une tension de sortie de type PWM (Pulse Width Modulation), c'est à dire un signal carré de **fréquence 200Hz** à rapport cyclique variable. Le rapport cyclique est proportionnel à l'angle du levier. La tension d'alimentation nominale du capteur est égale à la tension batterie. Sa consommation est inférieure à 15 mA.

Platines mobiles et actionneurs d'azimut :

Les platines et les actionneurs d'azimut ne se détaillent pas en pièce de rechange. En cas de dysfonctionnement, le remplacement du projecteur est nécessaire.

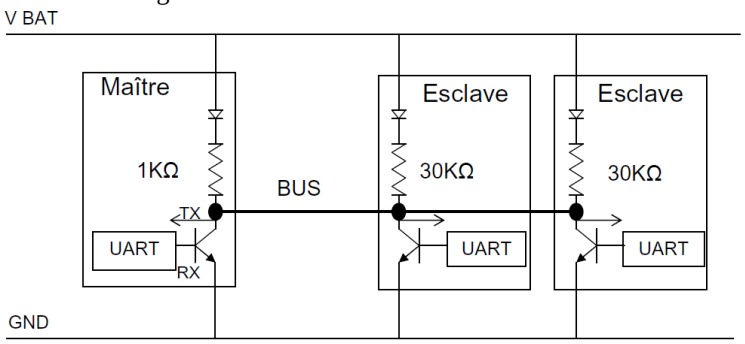
I.6 Principe du bus LIN

Le bus LIN est un bus série monofilaire de type maitre esclave qui comprend 1 maitre et jusqu'à 32 esclaves. Les transmissions entre nœuds LIN sont assurées par des contrôleurs de type SCI ou UART (contrôleurs compatible

STANDARD DATA FRAME

EXTENDED DATA FRAME

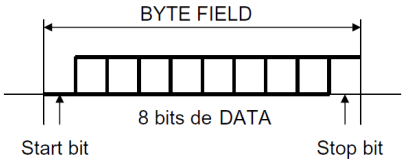
RS232). Le principe d'interconnexion des nœuds LIN permet une transmission des trames sur un seul fil comme illustré sur la figure ci-dessous.



Les bits '1' (niveau Vbat) sont récessifs et les bits '0' (GND) sont dominants. La norme LIN prévoit trois débits binaires :

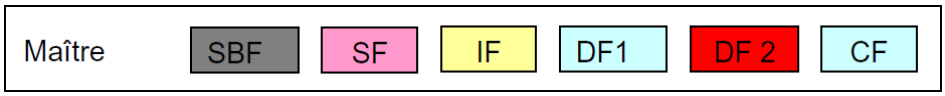
Slow : 2400 bit/s
Médium : 9600 bit/s
High : 19200 bit/s

Protocole, structure des trames:
Les trames du bus LIN sont composées de paquets de 10 bits (propriété des UART) structurés de la manière suivante :

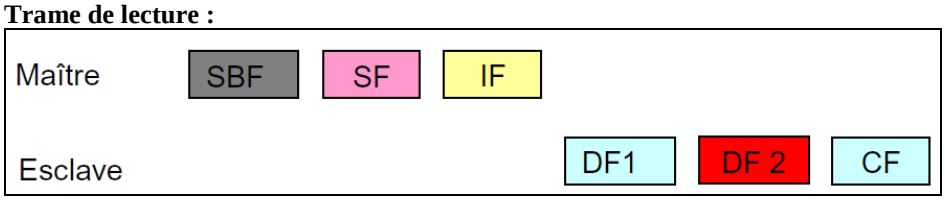


La trame complète de communication entre un maitre et un esclave LIN comporte plusieurs paquets de 10 bits séquencés comme l'illustre la figure ci-dessous dans les cas d'écriture ou de lecture par le maitre.

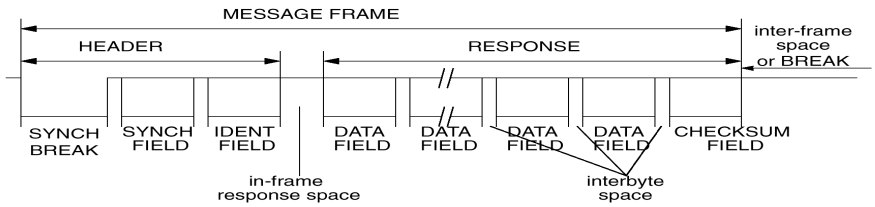
Trame d'écriture :



Trame de lecture :



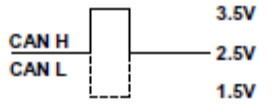
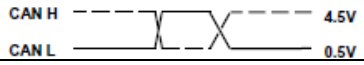
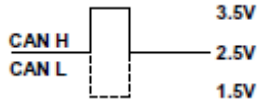
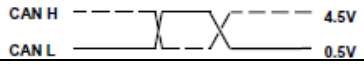
Description des paquets :



Documentation partiellement issue de la notice d'utilisation EXXOTEST MT-CAN-LIN-BSI.

II Questions

II.1 Compléter le tableau ci-dessous ou rayer les cas impossibles (les identificateurs sont exprimés en hexadécimal sur 3 digits pour les ID de 11 bits et sur 8 digits pour les ID de 29 bits)

ID	Niveaux mesurés	Débit binaire	Type (Standard/Etendu)	Norme (LS/HS)
0x100		500 kbit/s		
0x100		100 kbit/s		
		250 kbit/s	Standard	
0x12340000		100 kbit/s		
0x132		250 kbit/s		
0x8765000		125 kbit/s		
0x100				LS
0x10000000		1 Mbit/s		
0x12340000		250 kbit/s		
		150 kbit/s	Etendu	

II.2 Déterminer la fréquence maximale des trames pour CAN et LIN et des données pour PWM :

Liaison	Fréquence maximale (Hz)
CAN I/S (250 kbit/s, standard, 2 octets)	
CAN Car et CAN Conf (125 kbit/s, standard, 2 octets)	
CAN Magneti Marelli (100 kbit/s, standard, 2 octets)	
LIN (High speed, 2 octets, interbyte et interframe =0)	
Filaire PWM sorties des capteurs de hauteur de caisse	

Explications :

II.3 Les capteurs de hauteur de caisse avant 6616 (ou 7702) et arrière 6617 (ou 7703) délivrent des signaux PWM de rapport cyclique variant de 0 à 1 pour la course totale du capteur. Exprimer le rapport cyclique en fonction de l'angle du levier relié au capteur exprimé en degrés :

Réponse : Alpha =

II.4 Le calculateur de correction dynamique des projecteurs 6606 fait l'acquisition des signaux provenant des capteurs de hauteur de caisse. Citer un périphérique microcontrôleur usuel capable d'acquérir avec précision le rapport cyclique **Alpha** à chaque impulsion du signal PWM.

Réponse :

II.5 Indiquer dans quel domaine de l'angle α entre le faisceau lumineux et l'horizontale la formule $\alpha(\%)$ donnée dans l'énoncé peut-elle être adoptée.

Réponse :

II.6 Le calculateur 6606 reçoit plusieurs types d'informations depuis le bus CAN I/S afin de déterminer et envoyer les commandes des actionneurs de correction de site et d'azimut. Déterminer **en binaire et en hexadécimal** les valeurs des filtres et masques de filtrage pour permettre la réception des ID ci-dessous uniquement :

Signal	**ID**
Etat commande d'éclairage	0x100
Information capteur d'angle volant	0x110
Information sens de braquage	0x120
Information d'activation/inhibition cor. d'azimut	0x130
Information marche arrière	0x207
Information moteur tournant	0x3D5
Vitesse véhicule	0x401

Réponse :

C1RXF0SID =
C1RXF1SID =
C1RXF2SID =
C1RXF3SID =
C1RXF4SID =
C1RXF5SID =

C1RXM0SID =
C1RXM1SID =

II.7 La gestion des différentes tâches par le calculateur 6606 est réalisée par le noyau temps réel PICOS. La transmission de trames CAN toutes les 50ms est réalisée par synchronisation des envois sur une alarme. Compléter le code en langage C ci-dessous correspondant (initialisation de l'alarme et envoie synchronisé des trames).

Dans taskdesc.c :

```
volatile AlarmObject Alarm_list[] =
  {
   /*************************************************************************
    * ------------------------- First alarm -------------------------
    *************************************************************************/
   {
     OFF,                          /* State                */
     0,                            /* AlarmValue           */
     0,                            /* Cycle                */
     (RefCounter)&Counter_kernel,  /* ptrCounter           */
     TASK0_ID,                     /* TaskID2Activate      */
     ALARM_EVENT,                  /* EventToPost          */
     NULL                          /* CallBack             */
   },
};
```

Dans define.h : (A compléter)

```
/*************************************************************************
 * ------------------------------ Events ------------------------------
 *************************************************************************/
#define IT_EVENT       0x80
```

Dans tsk_task0.c : (A compléter)

```
/*************************************************************************
 * ------------------------------ TASK0 ------------------------------
*************************************************************************/
TASK(TASK0)
{
  CAN_config();


  while(1)
  {



  envoi_trame_CAN();

  }
}

/* End of File : tsk_task0.c */
```

II.8 Expliquer l'intérêt de réceptionner les trames CAN par interruption et stockage en buffer circulaire.

Réponse :

II.9 L'évènement BUFFER_NON_VIDE est activé dans la fonction d'interruption liée au contrôleur CAN. Cet évènement est exploité par la tâche 1. Compléter cette fonction d'interruption pour qu'elle soit complète et intégrée à l'environnement PICOS.

Dans tsk_task1.c : (A compléter)

```
void __attribute__((__interrupt__)) _C1Interrupt(void)
{


        IFS1bits.C1IF = 0;          // RAZ flag interruption CAN1
        if (C1INTFbits.RX0IF) stock_trame_recu();



}
```

Dans define.h : (A compléter)

```
/*****************************************************************************
 * ---------------------------- Events ------------------------------
 *****************************************************************************/
#define IT_EVENT      0x80
```

II.10 Compléter la tâche 1 qui lit et traite la plus ancienne des trames CAN reçues et stockées en buffer circulaire (nbval_CAN est une variable égale au nombre de valeurs stockées dans le buffer circulaire et non lues).

Dans tsk_task1.c : (A compléter)

```
/*****************************************************************************
 * ------------------------------ TASK1 -----------------------------
*****************************************************************************/
TASK(TASK1)
{

  while(1)
  {



  traite_trame_CAN();



  }
}
```

Annexe: Fonctions de l'API PICOS

DeclareTask(TaskIdentifier)	AppModeType **GetActiveApplicationMode**(void)
StatusType **ActivateTask**(TaskType TaskID)	void **StartOS**(AppModeType Mode)
StatusType **TerminateTask**(void)	void **ShutdownOS**(StatusType Error)
StatusType **ChainTask**(TaskType TaskID)	StatusType **ResumeAllInterrupts**(void)
StatusType **Schedule**(void)	StatusType **SuspendAllInterrupts**(void)
StatusType **GetTaskID**(TaskRefType TaskID)	void **EnableAllInterrupts**(void)
StatusType **GetTaskState**(TaskType TaskID,TaskStateRefType State)	void **DisableAllInterrupts**(void)
StatusType **SetEvent**(TaskType TaskID,EventMaskType Mask)	void **ResumeOSInterrupts**(void)
StatusType **ClearEvent**(EventMaskType Mask)	void **SuspendOSInterrupts**(void)
StatusType **GetEvent**(TaskType TaskID,EventMaskRefType Event)	void **EnterISR**(void)
StatusType **WaitEvent**(EventMaskType Mask)	void **LeaveISR**(void)
StatusType **GetAlarmBase**(AlarmType AlarmID,AlarmBaseRefType Info)	StatusType **GetResource**(ResourceType ResID)
StatusType **GetAlarm**(AlarmType AlarmID,TickRefType Tick)	StatusType **ReleaseResource**(ResourceType ResID)
StatusType **SetRelAlarm**(AlarmType AlarmID, TickType inc, TickType cycle)	
StatusType **SetAbsAlarm**(AlarmType AlarmID,TickType start,TickType cycle)	
StatusType **CancelAlarm**(AlarmType AlarmID)	