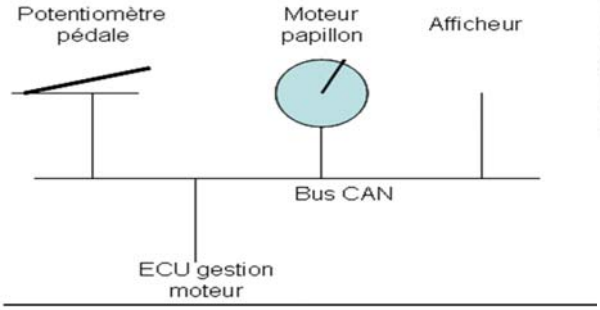
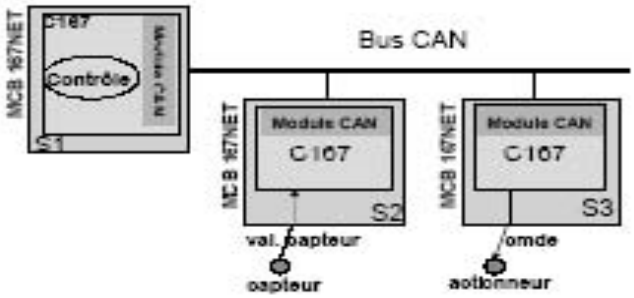
TP CAN - Objectifs

1. Objectifs

⇨ Utiliser un bus can comme support à la mise en œuvre d'applications distribuées
- comprendre et programmer des échanges CAN
- « découper/répartir » l'application

⇨ Exemple d'application = commande de position d'une vanne
- Ex. concret (automobile) : commande d'un boitier d'admission d'air dans moteur thermique, en fonction de l'appui sur la pédale d'accélération
- En TP : simulation via une maquette
  - Pédale (**capteur**) : un potentiomètre
  - Moteur de vanne (**actionneur**) : servo-moteur commandé en PWM
  - Micro contrôleur C167 **calculateur**)





Démarche de mise en Réseau

2. Etapes de la démarche pour la mise en place d'un réseau

1) Raisonner au Niveau applicatif : quels sont les besoins en termes d'échanges ?

2) Raisonner au niveau « choix du réseau » : quel(s) réseau(x) répond(ent) aux besoins ?

3) Raisonner au niveau « matériel/logiciel » du réseau : quels choix techniques/technologiques faire ou quels choix sont imposés pour mettre en œuvre le réseau ?

Démarche de mise en Réseau

■ 3. Niveau applicatif : quels besoin d'échanges pour l'application à distribuer ?

Capteur

Calculateur

Actionneur

Démarche de mise en Réseau

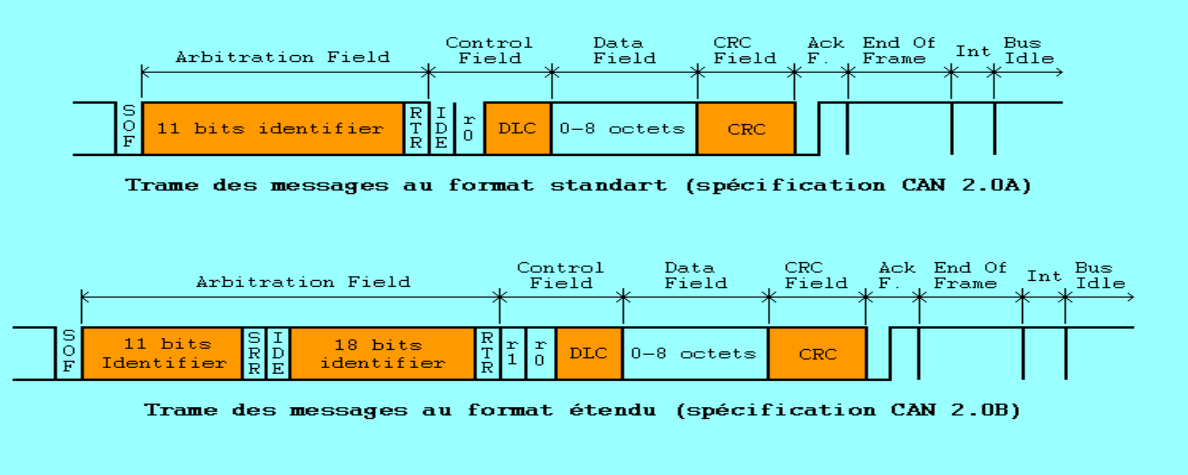
■ 4. Niveau « choix du réseau » : quel(s) réseau(x) proposent les mécanismes pour répondre aux besoin des échanges ?

- ⇨ Ici, le choix est imposé : protocole CAN sur bus (filaire)
- ⇨ Quels mécanismes offrent le réseau imposé ? Répondent-ils directement aux besoins ou faudra-t-il faire des adaptations ?
- ⇨ CAN : Combien de types de trames ?

Démarche de mise en Réseau

■ 4. Niveau « choix du réseau » : quel(s) réseau(x) proposent les mécanismes pour répondre aux besoin des échanges ?

⇨ Rappel :



Démarche de mise en Réseau

■ 5. Niveau « matériel/logiciel » du réseau : quels choix faire ou quels choix sont imposés pour mettre en œuvre le réseau choisi ?

⇨ Ici, choix imposé : contrôleur de bus CAN sur microcontrôleur C167 Siemens

⇨ Solution imposée → obligation d'utiliser ce qui est imposé … par le fournisseur, en termes de performances, fonctions, dénominations, etc.

■ 6. TP en 2 étapes

1) Appréhender la programmation d'échanges CAN (caractères, entiers)
2) Essayer de répartir un programme de commande déjà écrit et fourni, … répartir sur 2 à 3, ou plus, microcontrôleurs

Détails pour le TP

7. Ce qui est fourni

Un ensemble de fichiers mis à disposition dans /home/MEEA/TPRLI … est à recopier tel quel sur votre compte :

- ⇨ Un répertoire Doc avec 4 fichiers : 2 documentations techniques sur le CAN du C167, une aide pour utiliser la liaison série du C167 et faire de l'affichage sur l'écran du pc : **Info_commc167.txt**.
- ⇨ **servo.c** : programme de l'application de commande en position du servo-moteur. Programme complet écrit pour ne tourner sur un seul micro-contrôleur.
- ⇨ **LA librairie « BAS niveau » des routines CAN fournie par SIEMENS**
- ⇨ un squelette, nommé « squelet….can.c » : il sera **à renommer en « can.c »** et à compléter : **ce sera LA BIBLIOTHÈQUE de HAUT niveau à réaliser**
- ⇨ un makefile à modifier/compléter : pas pour de la vrai compilation séparée, juste pour ne pas se tromper et ne pas perdre du temps avec la commande de compilation

Détails pour le TP

8. Ce que vous avez à faire

- ⇨ Compléter le fichier de « bibliothèque » de haut niveau
- ⇨ Créer un fichier avec le main du programme pour chaque utilisation sur chaque microcontrôleur : AUTANT de programmes que nécessaire !!!
- ⇨ Adapter le makefile : PAS DE COMPILATION SEPARÉE (pas le but du TP), seulement « gagner » du temps à la compilation, spécifique au microcontrôleur + librairie CAN

```
ALL : ABON1
ABON1:
    $(CC) -O2 -I "./include" -I "." -Wall -m7 -g abon1.c -o Xabon1 -ltpc167 -lcan_16x_l
clean:
    rm -f    Xabon1
```

9. Points d'attention

- ⇨ Dans le squelette fourni : X
  - – Beaucoup d'explications via des commentaires …pas toujours bien « affichés » (décalages et autres), et pas toujours à jour
  - – Tout ce qui commence par ??? n'est pas à faire … mais il en manque : en résumé, tout ce qui touche au DLC n'est pas à tester

Détails pour le TP

9. Points d'attention

⇨ Le sujet vous indique les éléments à cibler dans la doc du Siemens

– Attention, certain éléments sont des TABLEAUX, et non des FONCTIONS

VARIABLES ET PRIMITIVES DE BASE DU CONTRÔLEUR CAN À UTILISER DANS CE TP

a) VARIABLES utiles (2 tableaux) :

- **char dir_bit_16x[16] : valeur des champs (bits) de direction**
- **char dlc_16x[16] : longueur du message émis/reçu**

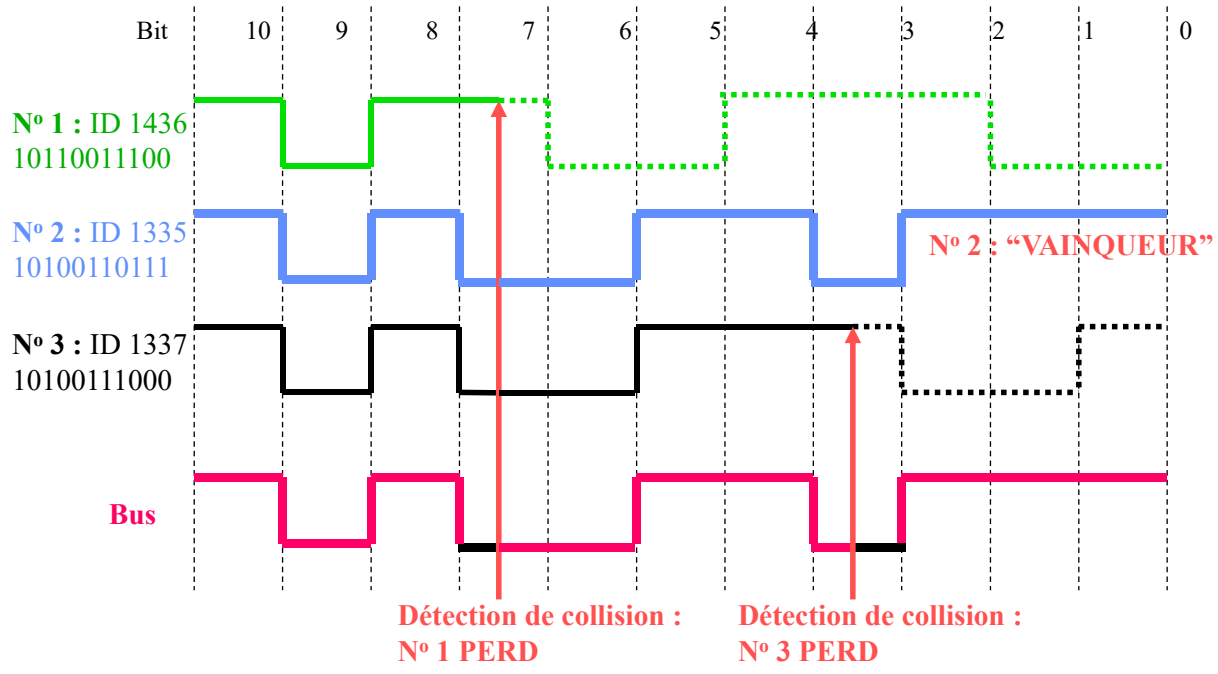
b) FONCTIONS utiles :

- **init_can_16x(...) : initialisation du module CAN**
- **def_mo_16x (...) : définition/initialisation d'un message objet**
- **ld_modata_16x(...) : charge une donnée dans le registre**
- **send_mo_16x(...) : émet le message**
- **check_mo_16x(...) : vérifie la présence d'un message**
- **rd_modata_16x(...) : copie la donnée d'un message reçu**

Détails pour le TP

10. Complément : Bit dominant

⇨ Principe fondamental de résolution de collision : toujours un vainqueur



10. Complément : Contrôleur CAN dans le microcôntroleur C167