

E3020 Systèmes embarqués

CM5-BUS SPI/USART/CAN

Objectifs

1. Analyse du bus SPI
2. Manipulation du bus série via le module USART
3. Particularités du bus CAN

Bus de communication

SPI – SERIAL PERIPHERAL INTERFACE

SPI

Quelles sont ses caractéristiques ?



- Synchrone/asynchrone ?
- Half duplex / Full duplex ?
- Relation entre les puces ?
- Type de communication ?
- Nombre de fils et leurs fonctions ?

Quels sont les nouveautés par rapport à l'I2C ?


Serial Peripheral Interface

- Caractéristiques :
 - Communication Série
 - Synchrone
 - Maître-esclave
 - Full-duplex

- Le **Full-duplex** consiste à pouvoir envoyer des information tout en pouvant en recevoir.
- Il y a par contre toujours au minimum une trame de décalage entre la demande de lecture d'une information et sa réception.

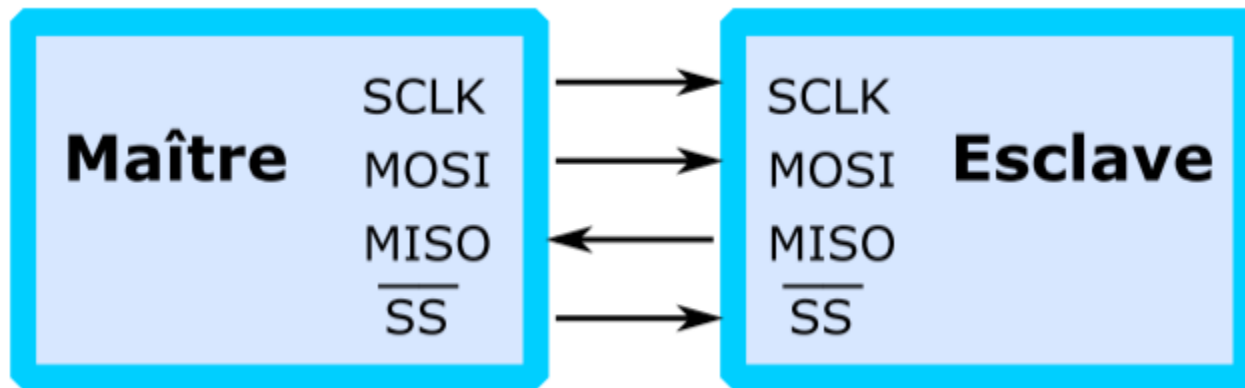


Serial Peripheral Interface

- Le SPI est composé de 4 fils :
 - 1. **SCLK** : il s'agit de la clock générée par le maître. 
 - 2. **MISO** : Il s'agit de l'acronyme pour **Master Input / Slave Output**. Sur ce fil les données vont de l'esclave vers le maître.
 - 3. **MOSI** : Il s'agit de l'acronyme pour **Master Output / Slave Input**. Sur ce fil, les données vont du maître vers l'esclave.
 - 4. **SS** : Le slave select, géré par le maître permet d'activer un (voire plusieurs) esclave. Son entrée est généralement négative. (0 actif – 1 inactif)

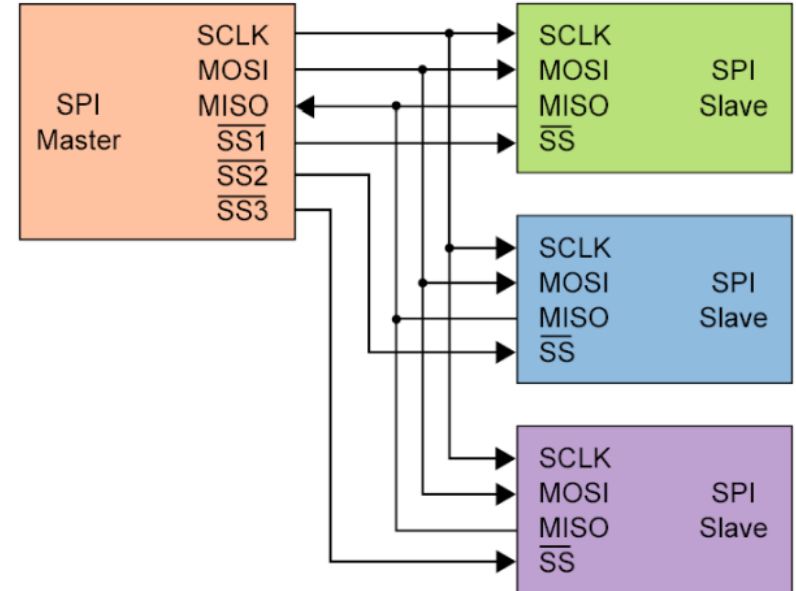
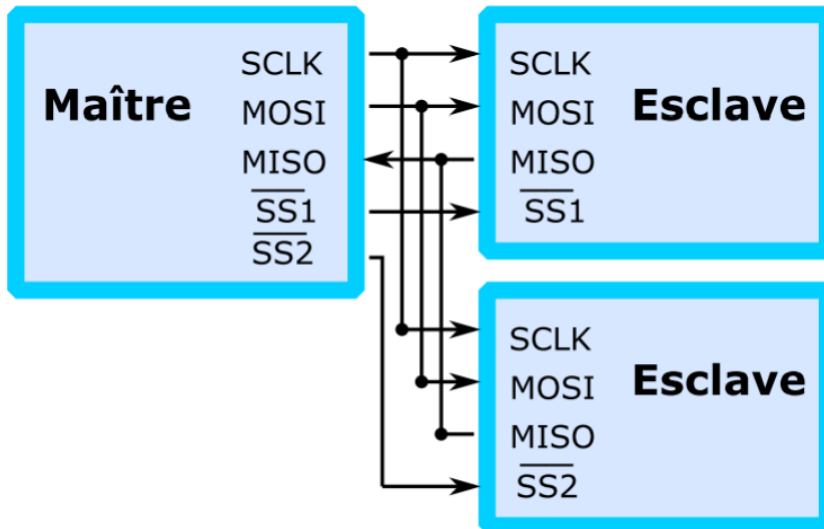
BUS : SPI

- Schéma et sens des connexions entre un maître et un esclave.



BUS : SPI

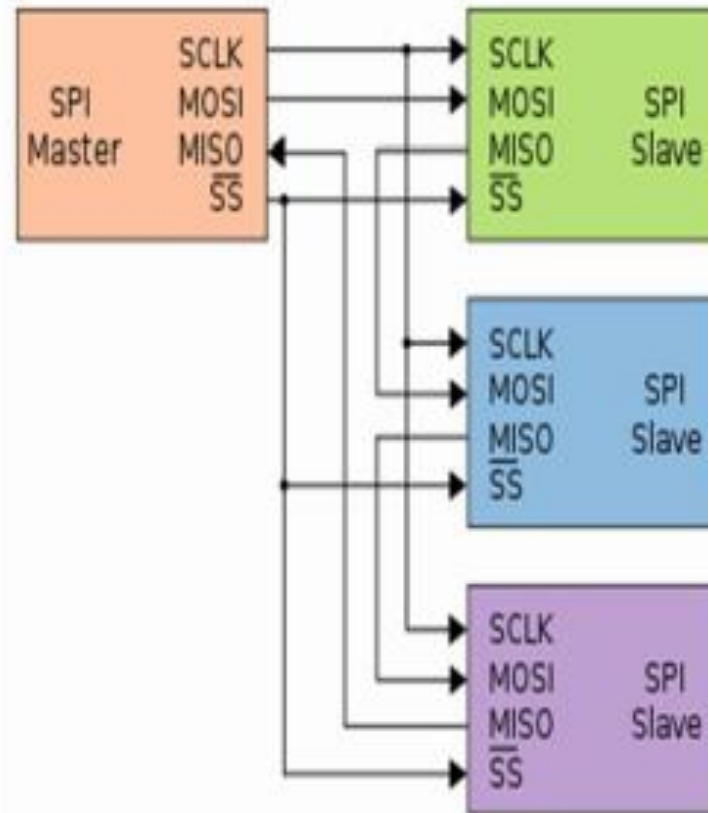
- Exemple de connexion entre un maître et plusieurs esclaves.



BUS : SPI – daisy chain

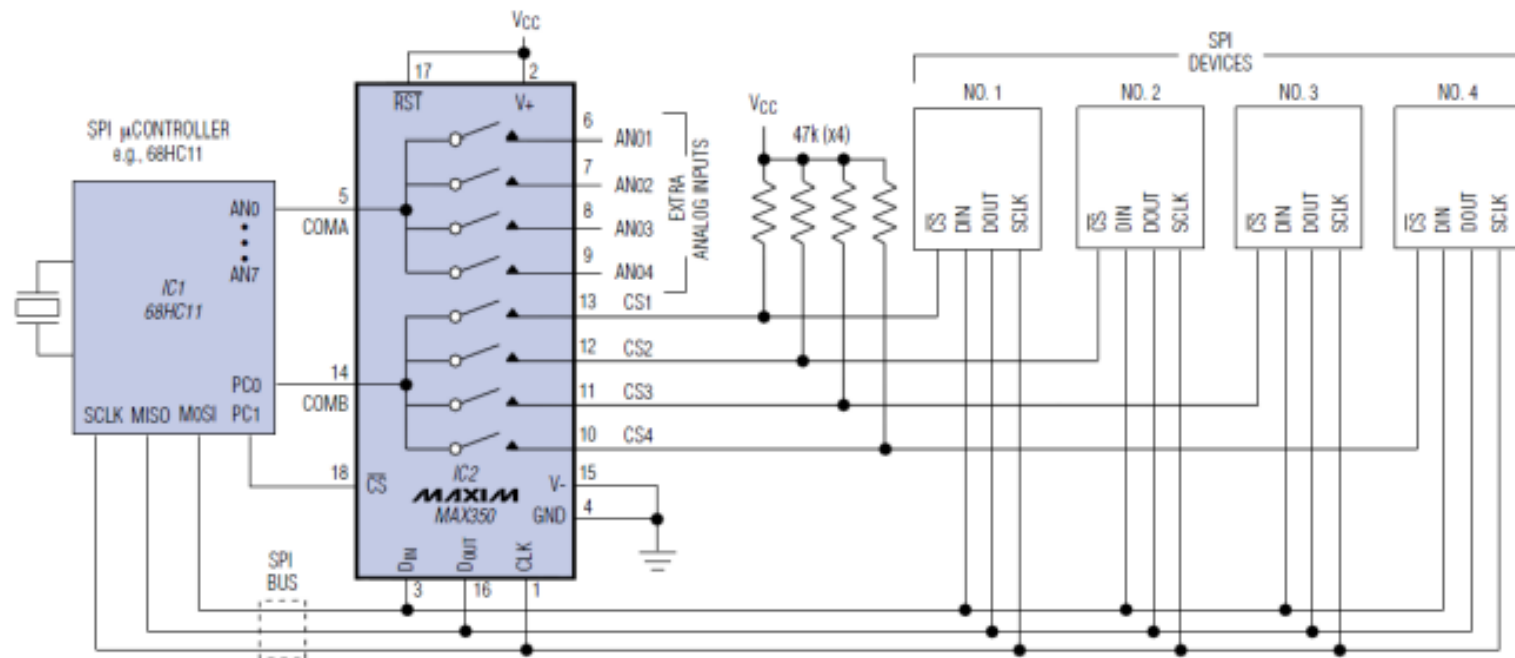


- Astuce pour contrôler plusieurs esclaves



BUS : SPI – multiplexeur

- Astuce pour contrôler plusieurs esclaves



BUS : SPI

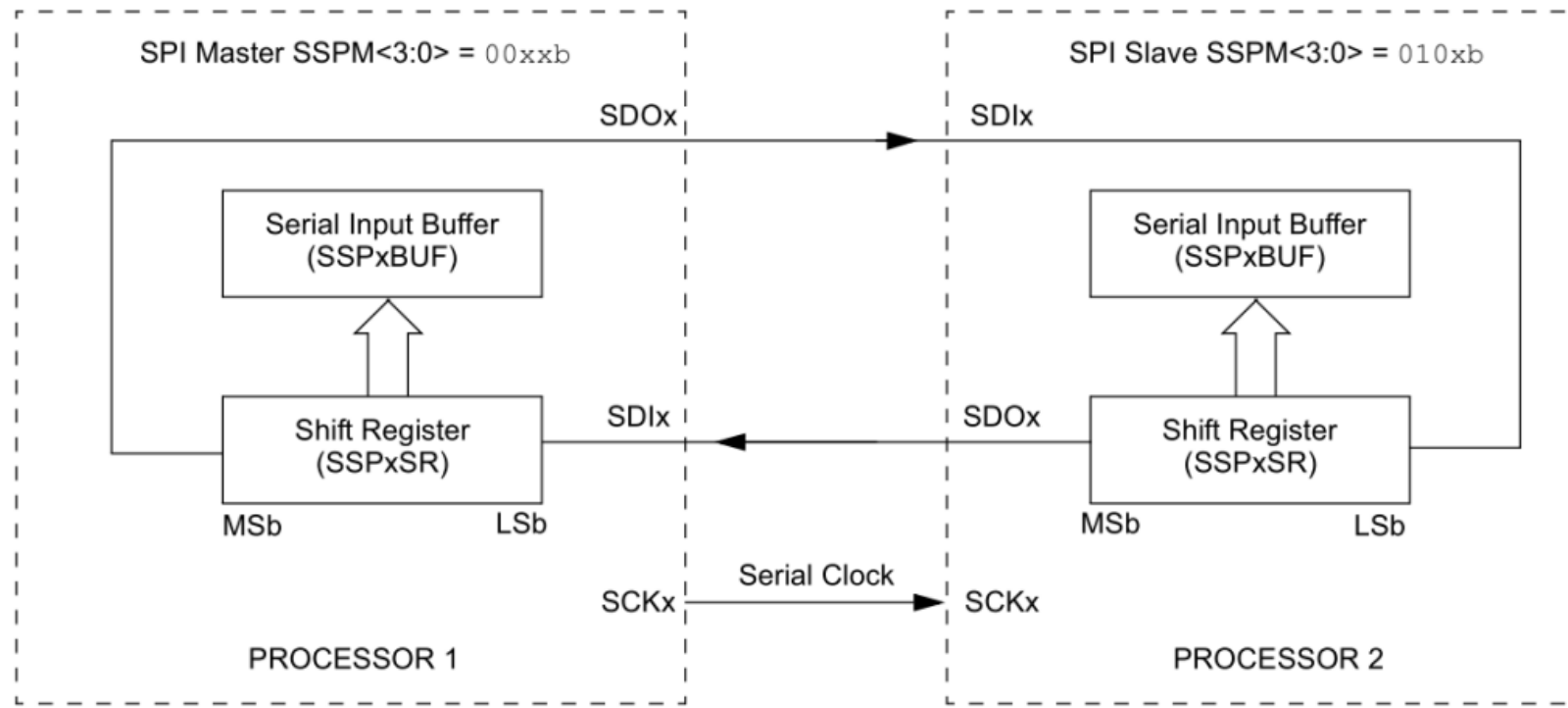
- La séquence de communication :
 - 1. Le maître commence la communication en mettant une de ses sorties SS à "0". Celle qui atteint l'esclave souhaité.
 - 2. A chaque coup de clock, le byte préchargé dans le registre à décalage du maître est envoyé bit à bit sur le MOSI.
 - 3. Pendant ces mêmes coups de clock, le byte préchargé dans le registre à décalage du maître est envoyé bit à bit sur le MISO.
 - 4. Au bout de 8 coups de clock, les bytes sont échangés.

BUS : SPI - dans le Pic

- Qu'auriez-vous besoin pour implémenter une telle communication dans un microcontrôleur ?
- Combien de registres auriez-vous besoin ?

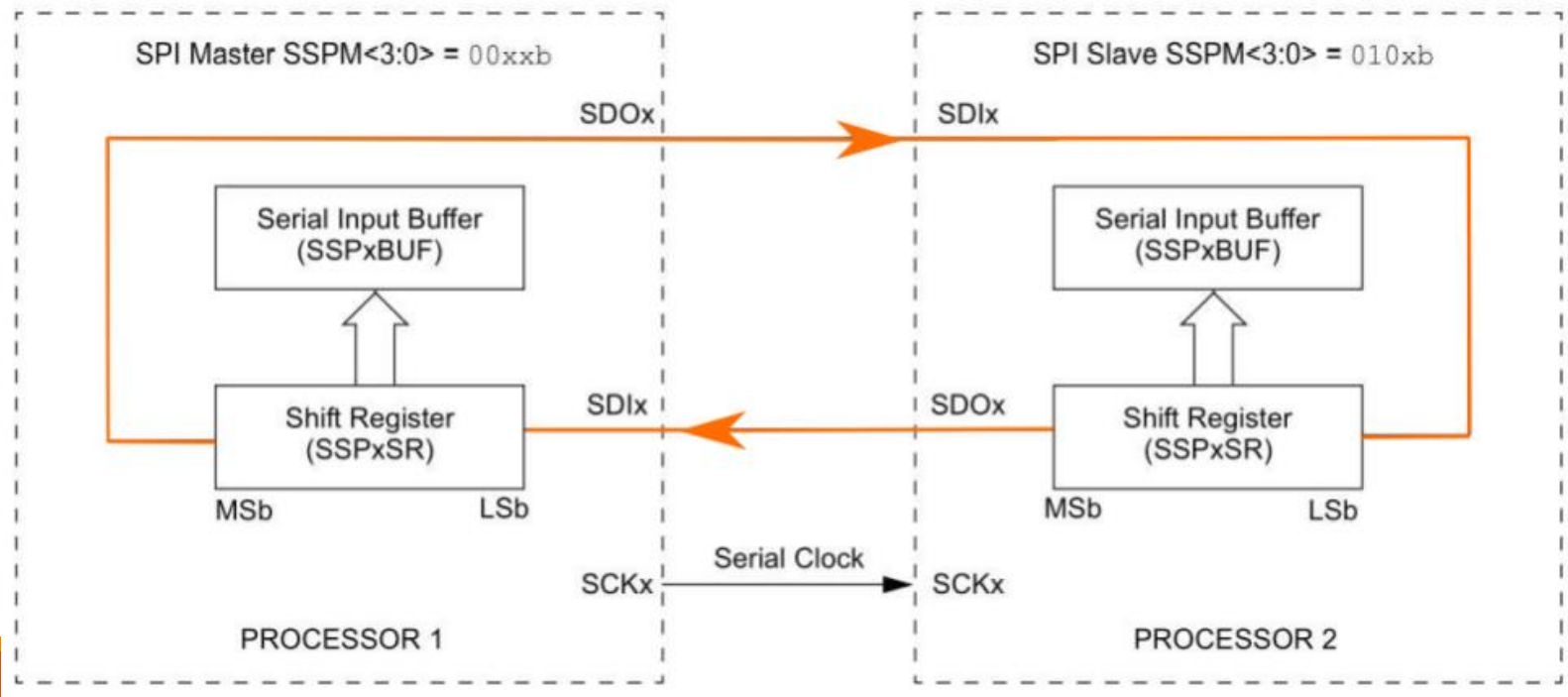
Bus : SPI - dans le Pic

- Pour l'implémentation, il doit y avoir :
 - 1 registre à décalage
 - 1 registre buffer



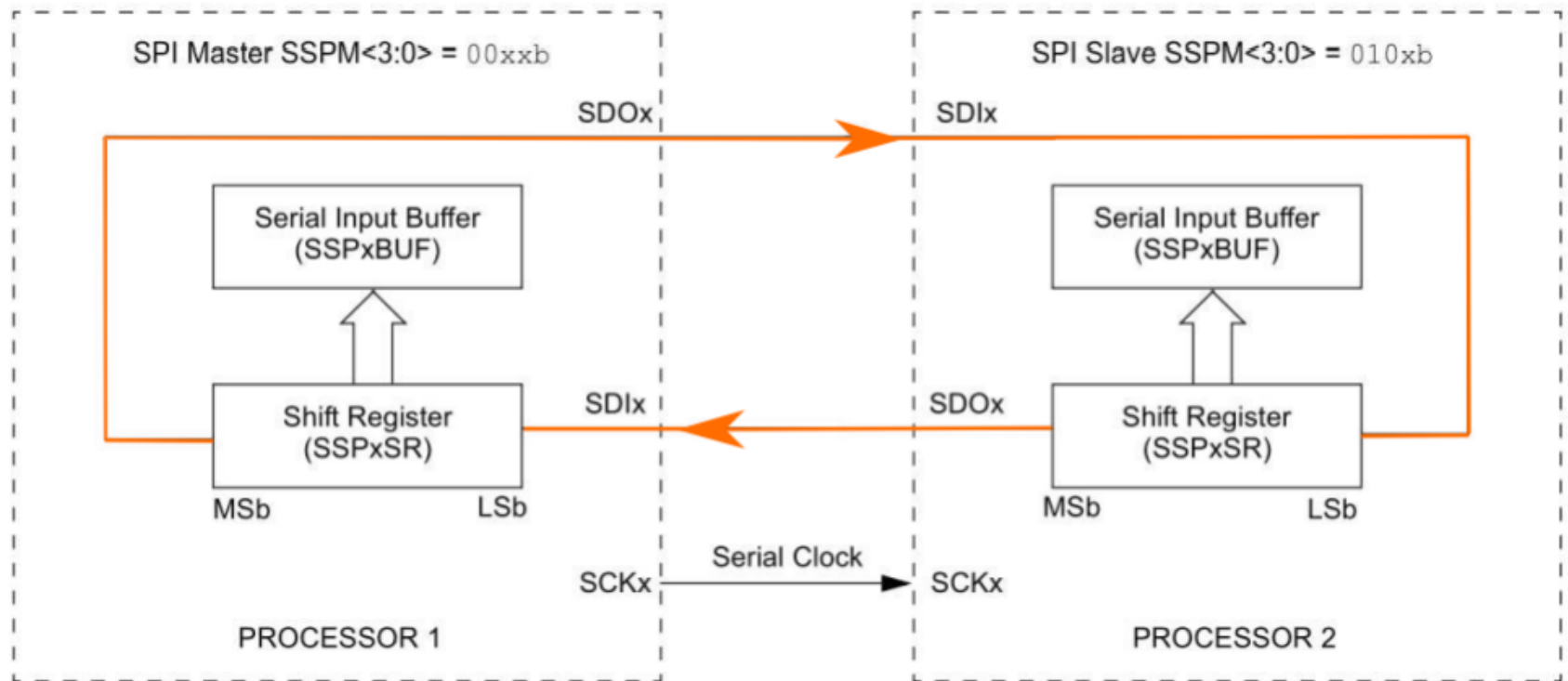
Bus : SPI - dans le Pic

- A chaque coup de clock, les "shift registers" décalent leurs données de 1 bit vers la gauche.
- Ils reçoivent donc en même temps que l'envoi un bit. Ce dernier est placé à la place du LSb.



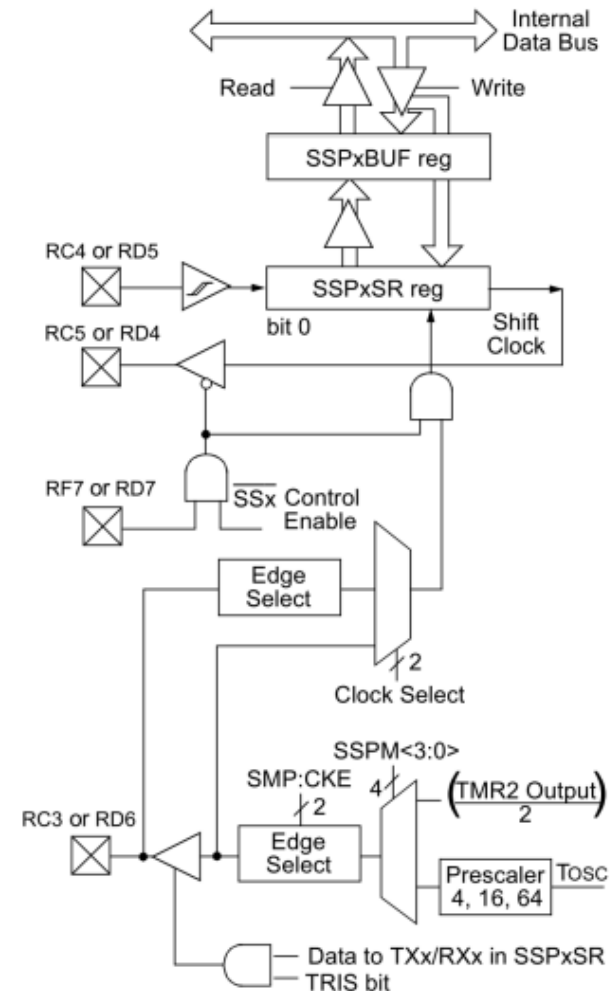
Bus : SPI - dans le Pic

- Au bout des 8 coups de clock, tous les bits ont été envoyés et sont stockés dans un buffer interne. Le flag SSPIF passe à un pour informée que la lecture du buffer peut se faire.



Bus : SPI - dans le Pic

- Si le module est en mode esclave, le slave select contrôle un buffer pour empêcher la donnée d'aller sur le MISO.
- Si le module est en mode maître, il doit générer la clock.



Bus : SPI - dans le Pic

- 4 pins sont nécessaires :
 - Data out via RC5 ou RD4
 - Data in via RC4 ou RD5
 - La clock via RC3 ou RD6
 - Slave select RF7 ou RD7
- Il y a quatre registres :
 - MSSP control register (SSPxCON)
 - MSSP status register (SSPxSTAT)
 - Serial buffer register (SSPxBUF)
 - MSSP shift register (SSPxSR)

SPI : Modes

- Mode Maître :
 - Il initialise le transfert de donnée en imposant la clock. (plusieurs choix possible, cfr SSPxCON)
 - La donnée est transmise dès que SSPxBUF est écrit.
- Mode Esclave :
 - On peut activer le Slave Select.
 - S'il passe à '1' pendant la transmission cette dernière est interrompue et doit attendre pour reprendre la transmission.

SPI : horloge

Il n'y a pas de spécification précise dans le protocole SPI précisant les fronts d'horloges pour la validation des données...

En pratique, il y a 4 modes de fonctionnements...

Bus : SPI – polarité et phase de l'horloge

Recherchez la signification...

Clock polarity (valeur de la clock au repos)

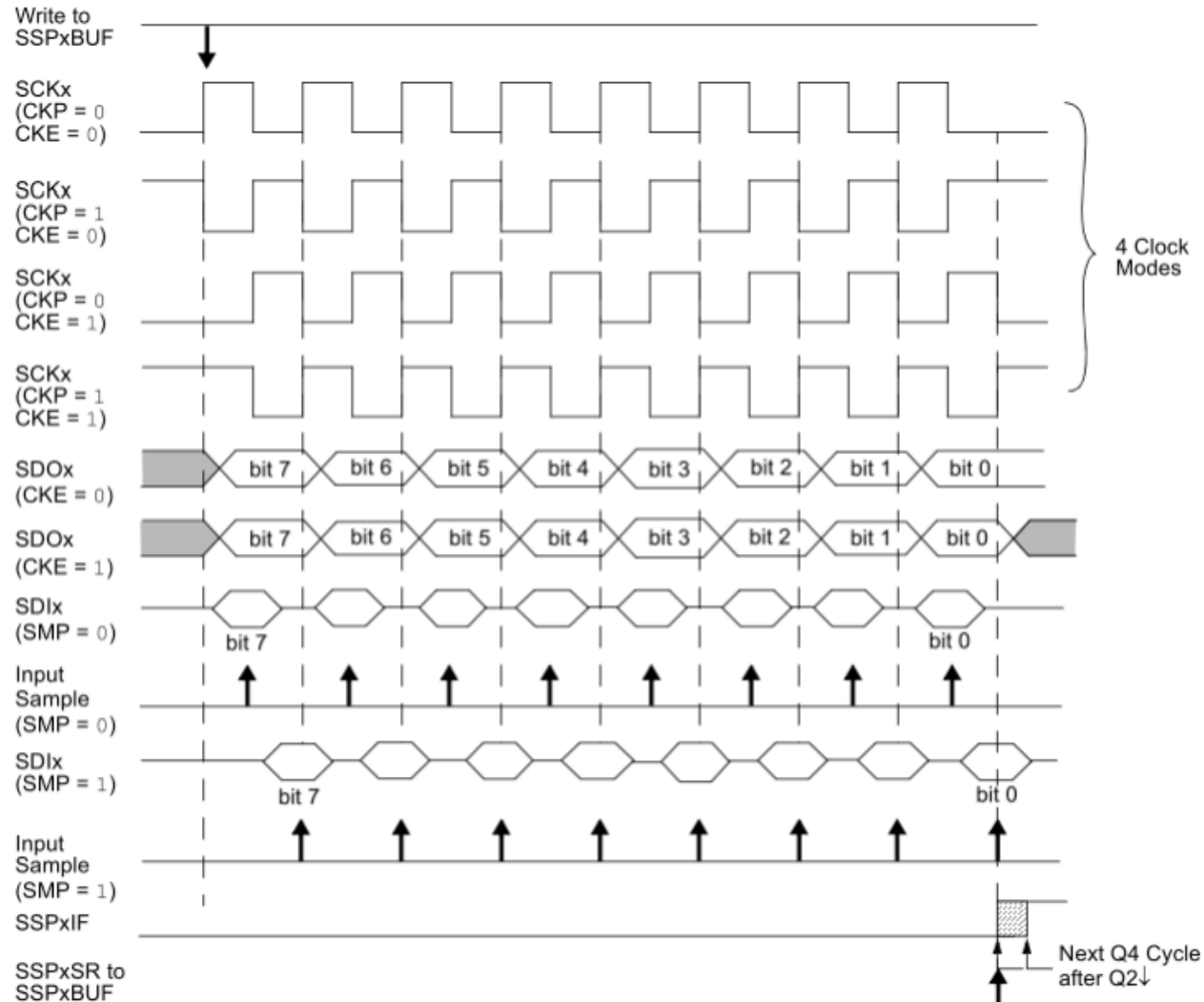
and

Phase (permet de spécifier quand se fait l'acquisition, premier ou deuxième front)

Bus : SPI – polarité et phase de l'horloge

- Configuration :
 - Mode maître ou esclave
 - Clock polarity choix entre (CKP)
 - Choix de l'état par défaut de la clokc en absence de transmission. (Idle state)
 - Clock select bit (CKE)
 - 1. Shift sur flanc montant et echantillonnage sur flanc descendant
 - 2. Inverse de 1.
 - *Il faut la même polarité à gauche et à droite.*
 - *Définit par rapport à l'idle state.*

SPI : Transmissions en maître



SPI

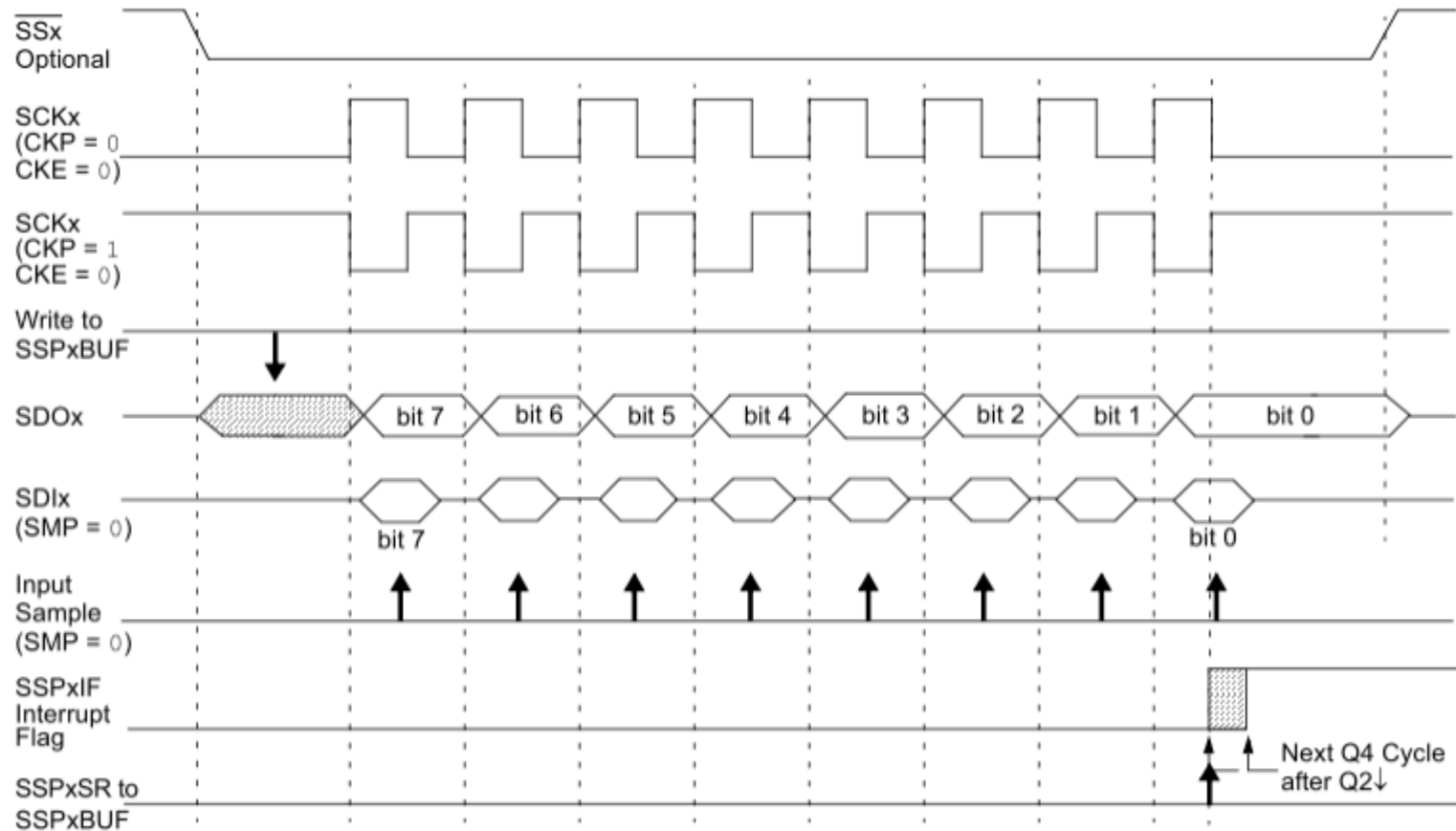
Le maître place toujours sa donnée en début de cycle

L'esclave lira donc toujours la donnée en début de cycle

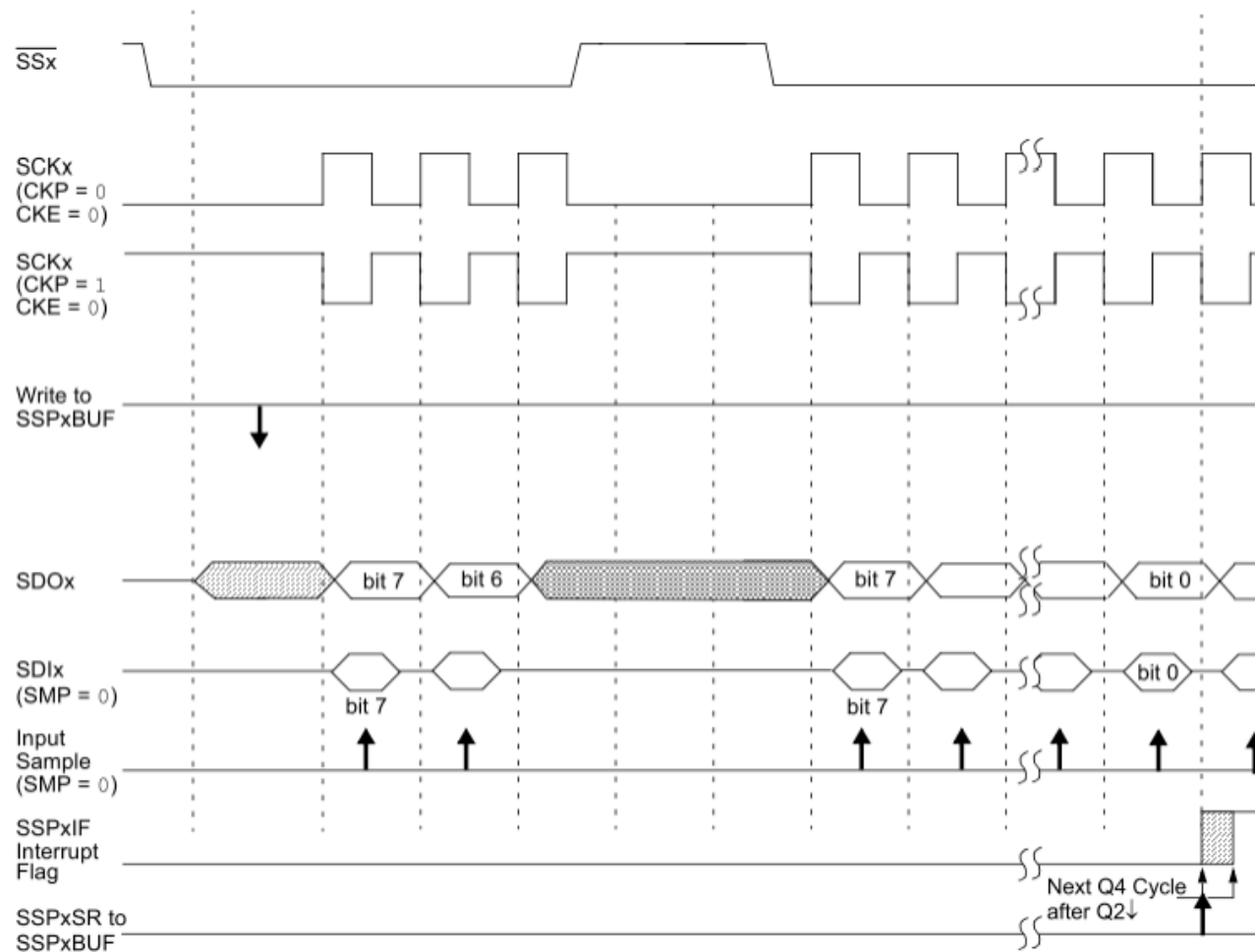
L'esclave peut placer sa donnée soit en début soit en milieu de cycle

Le maître lira donc soit en milieu ou en fin de cycle

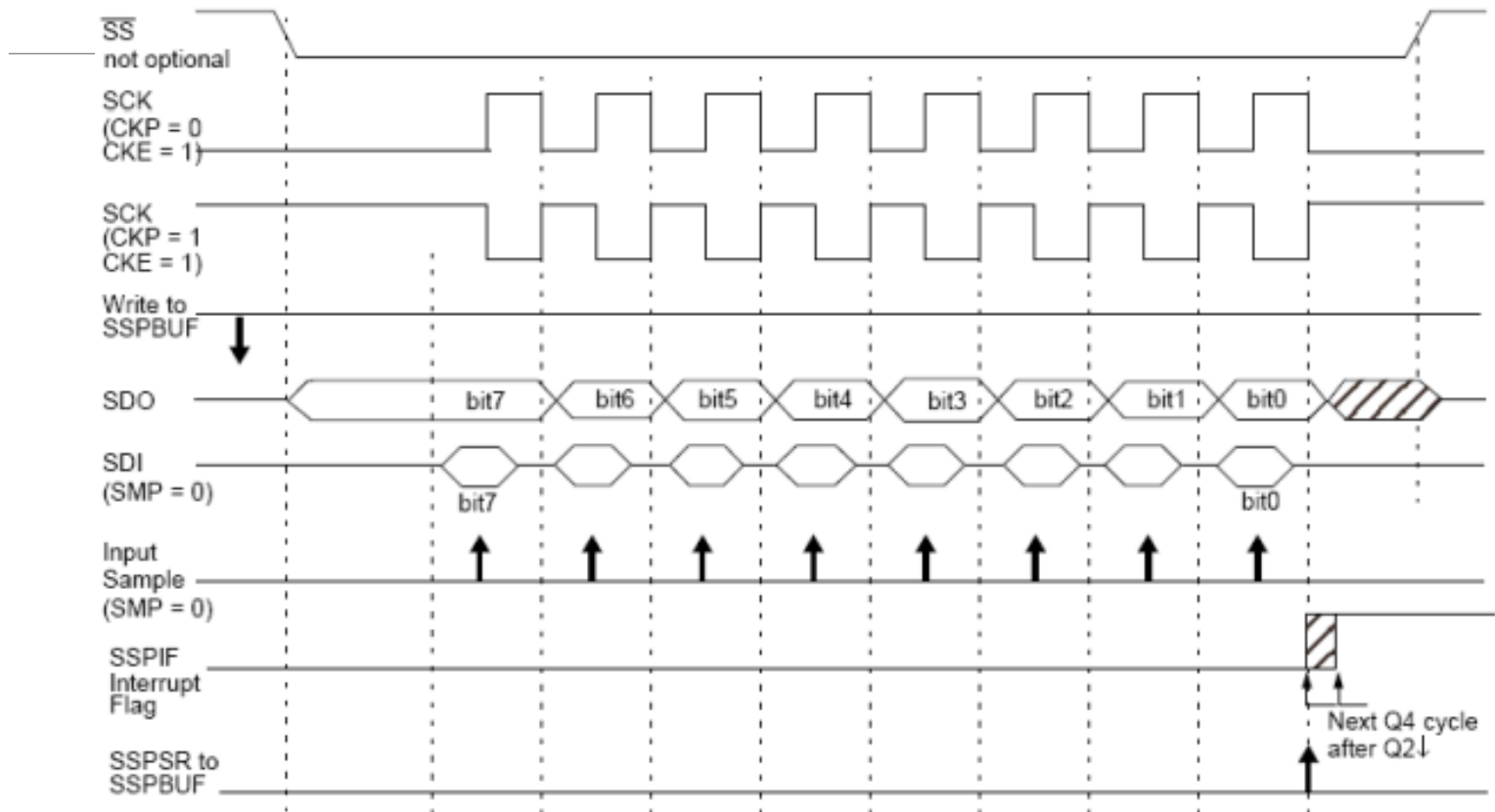
SPI : Transmissions en esclave



SPI : Transmissions en esclave



SPI : Transmissions en esclave



SPI - esclave

Si CKE = 0, plus besoin de SS car il devient l'esclave et il a juste besoin de détecter le premier flanc actif sur SCK

Si CKE=1, alors SS est nécessaire car il faut que la donnée soit là avant la première transition.

SPI : capteur en mode SPI

Comment passer dans ce mode ?

Qu'est-ce qui change ?

Bus de communication

USART – UNIVERSAL SYNCHRONE/ASYNCHRONE
RECEIVER TRANSMITTER

USART :

- Universal Synchronous & Asynchronous Receiver Transmitter
 - L'usart est un composant permet de transférer des données en série :
 - soit en mode asynchrone et Full-Duplex.
 - Soit en mode synchrone et half-duplex.
 - Le terme universel prend son sens dans le fait que l'on peut implémenter divers protocoles avec ce module.

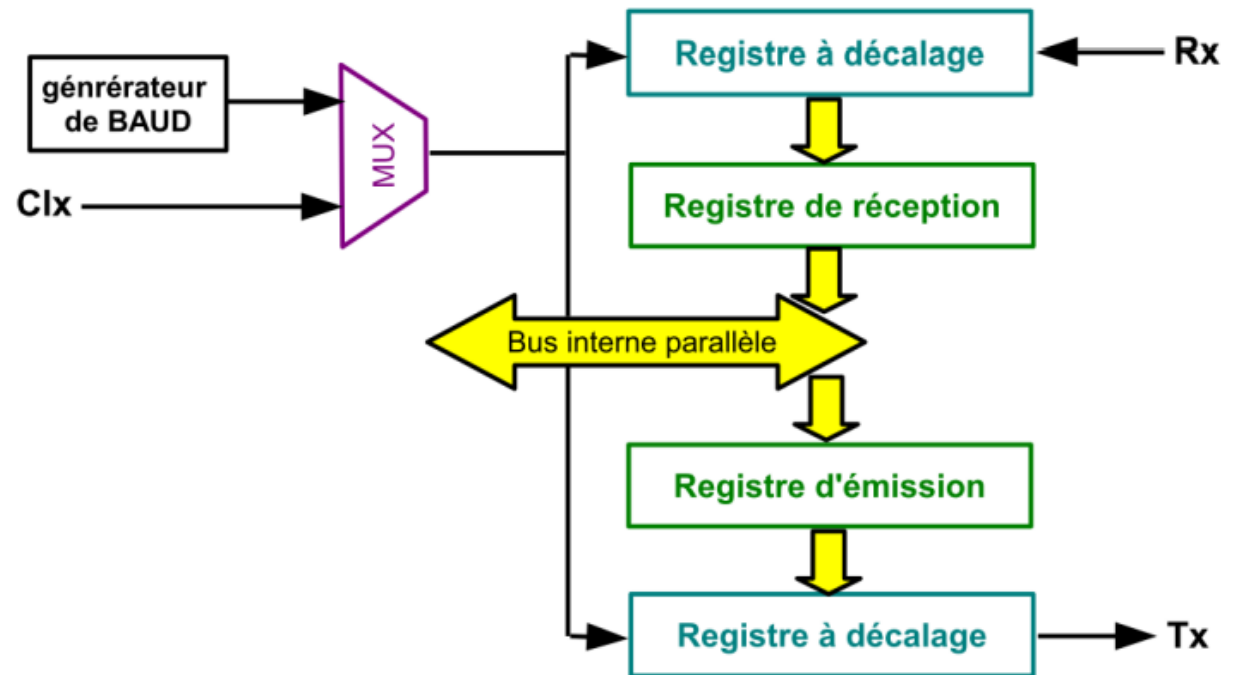


USART :

- Mode asynchrone et full-duplex
 - 1 pin transmet la donnée. (TXx)
 - 1 pin reçoit l'information (RXx)
 - Il faut alors définir la fréquence à l'avance.
 - On parle de Baud (bit/sec)
- Mode synchrone et half-duplex.
 - 1 pin reçoit ou envoie l'information (DTx)
 - 1 pin génère ou reçoit la clokc (CKx)

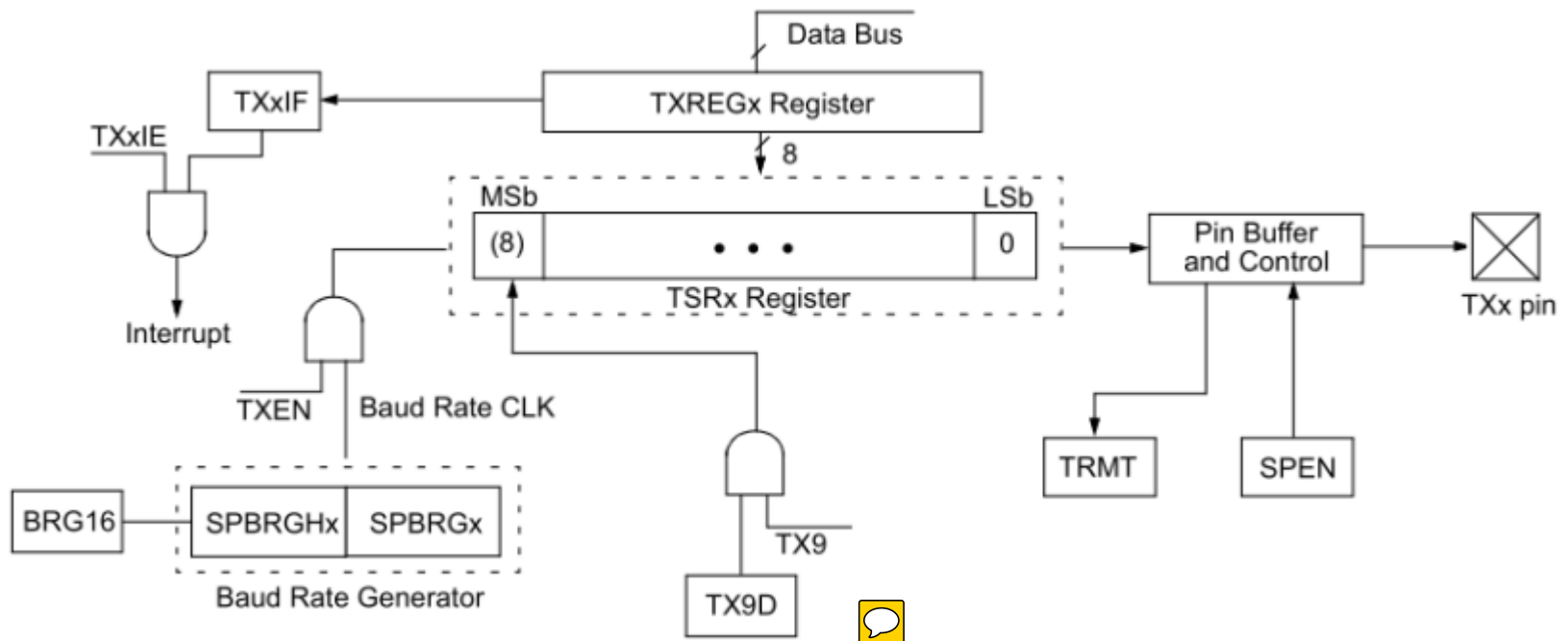
USART :

- Schéma de principe :
 - Implémente des fonctionnalités pour
 - RS-232
 - RS-485
 - LIN/J2602



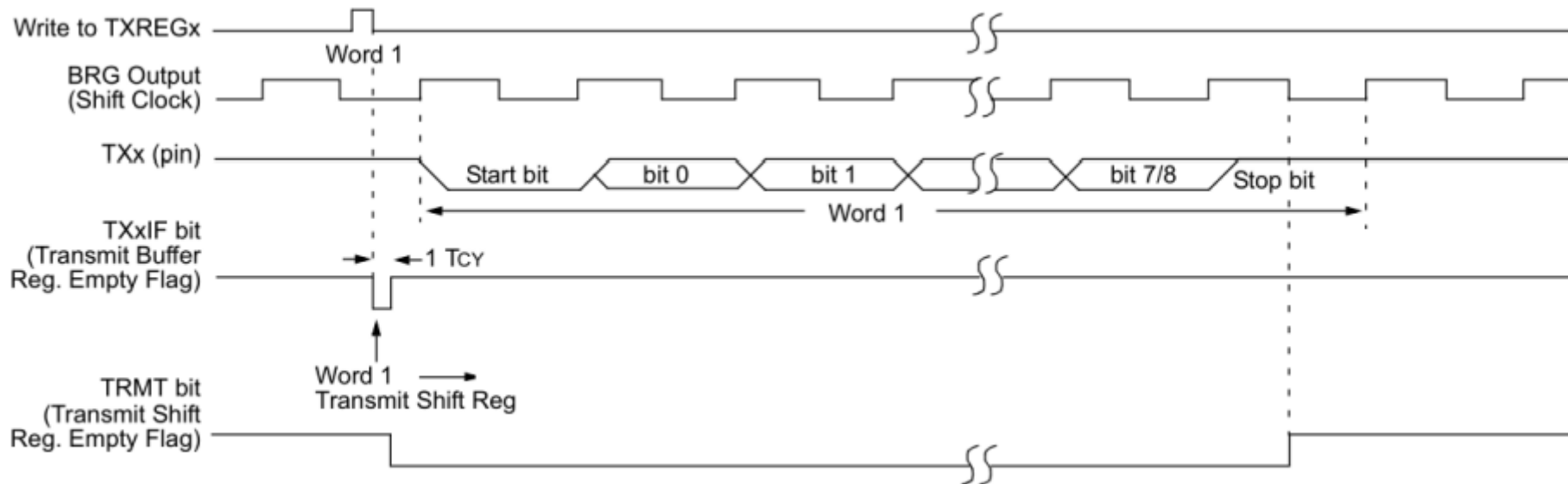
USART : Mode Asynchrone

- Une trame de base est composée de 8 bits stockés dans un registre tampon. (TXREGx)
- Une fois la donnée transférée, le flag associé est mis à 1.



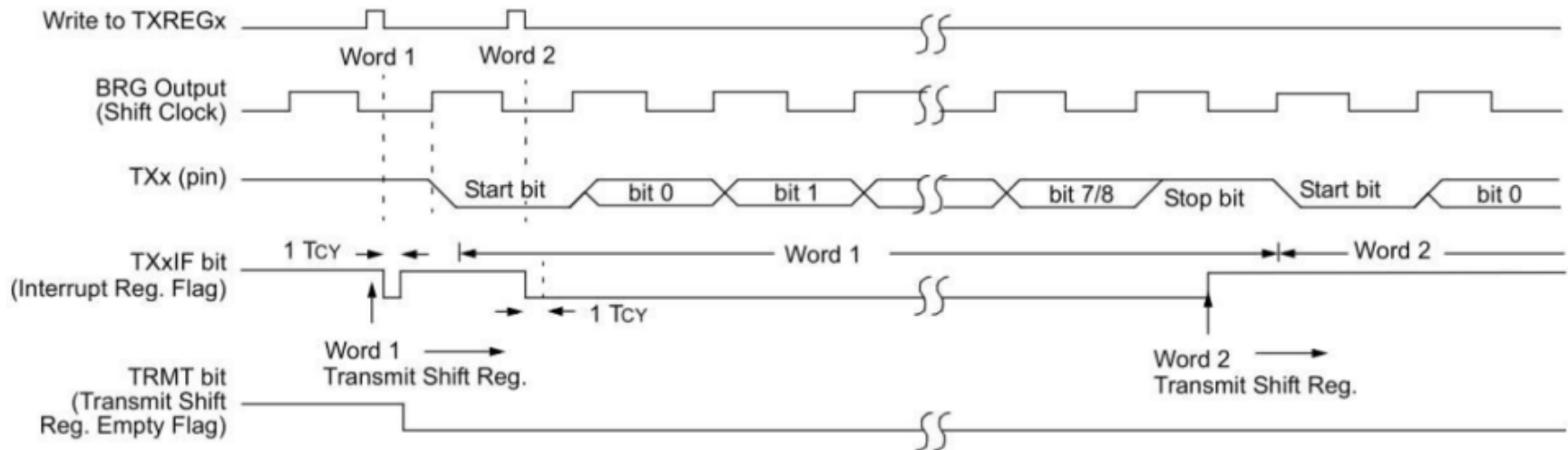
USART : Transmission asynchrone

- Dès que l'on écrit dans TXREGx, le flag TXxIF passe à '0'. La fin de la transmission est donnée par le bit TRMT qui passe à '1'.



USART

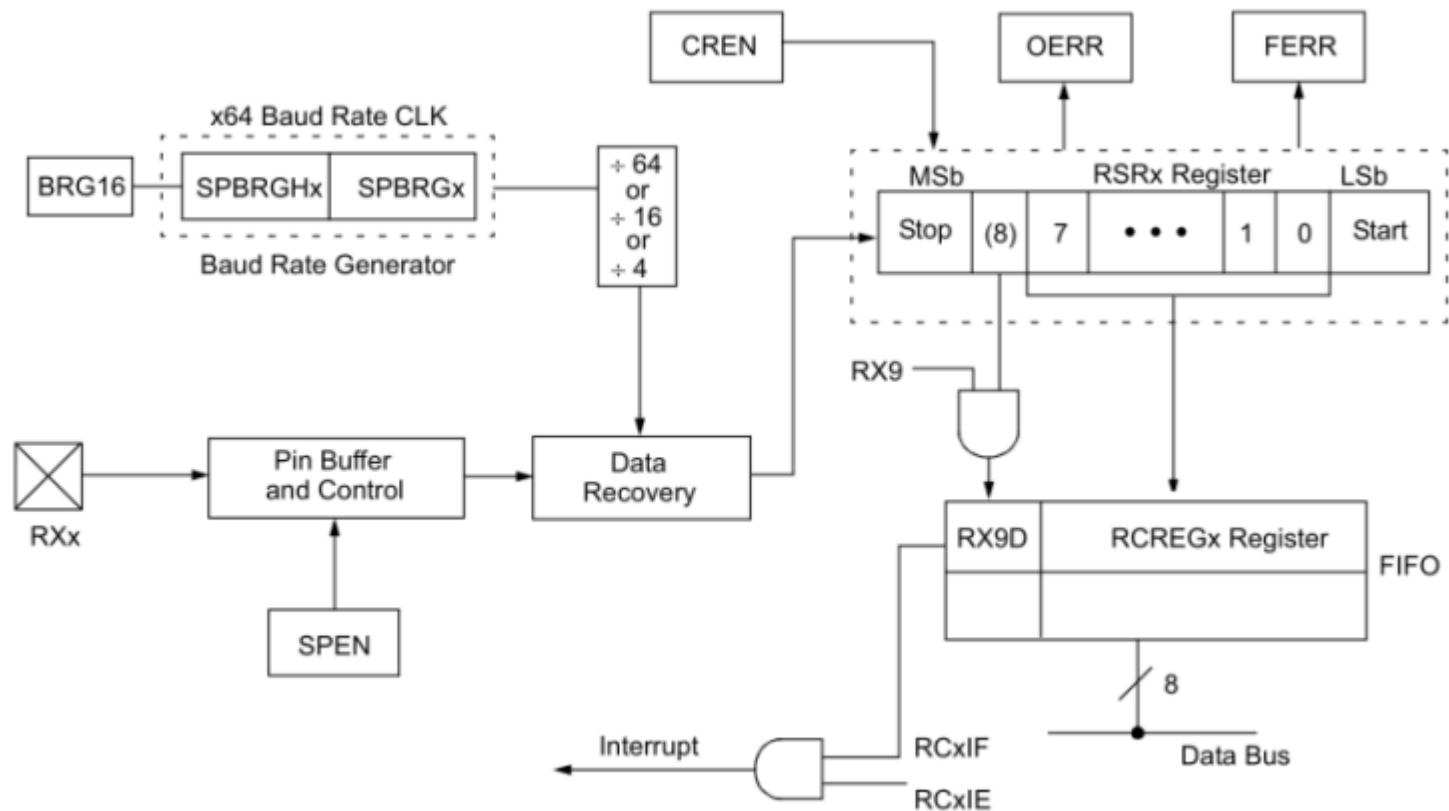
- Cas d'une transmission de deux mots consécutifs.
- TXxIF reste à '0' tant que le deuxième mot n'est pas transféré.



USART : Réception asynchrone

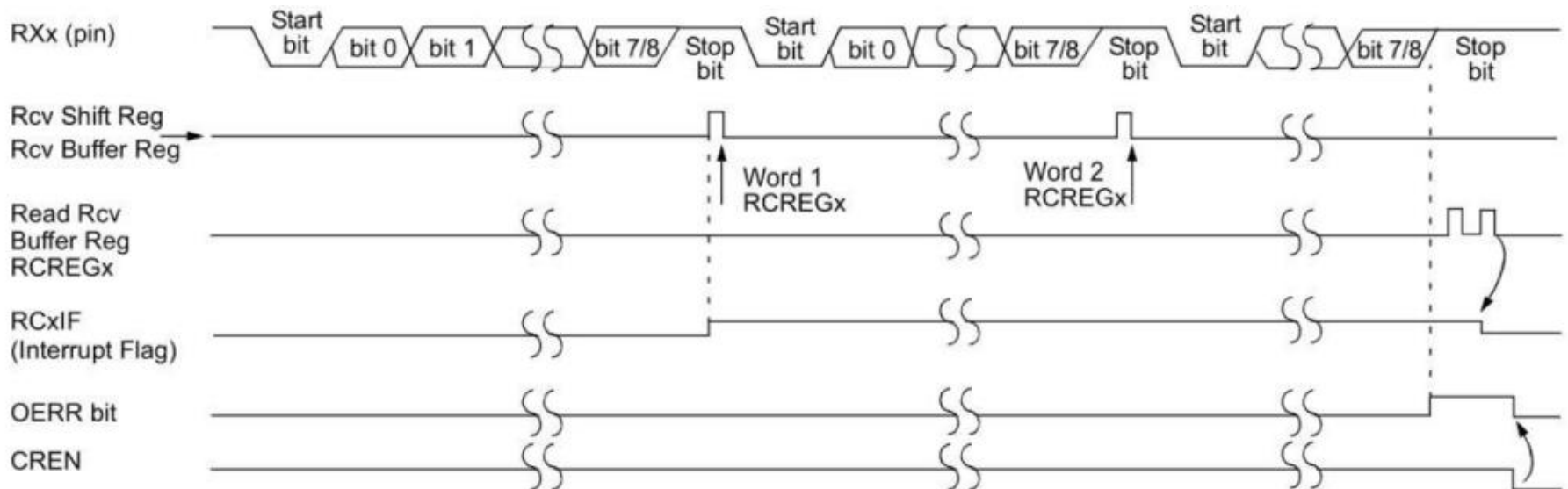
- La fréquence d'acquisition est définie par le Baud rate.
- Une fois terminée, le bit est transféré dans une FIFO à 2 étages.
 - Si la FIFO n'est pas vidée assez rapidement, une erreur d'overflow apparaît.
 - Si la FIFO n'est pas vide, le flag reste à 1

USART : Réception asynchrone



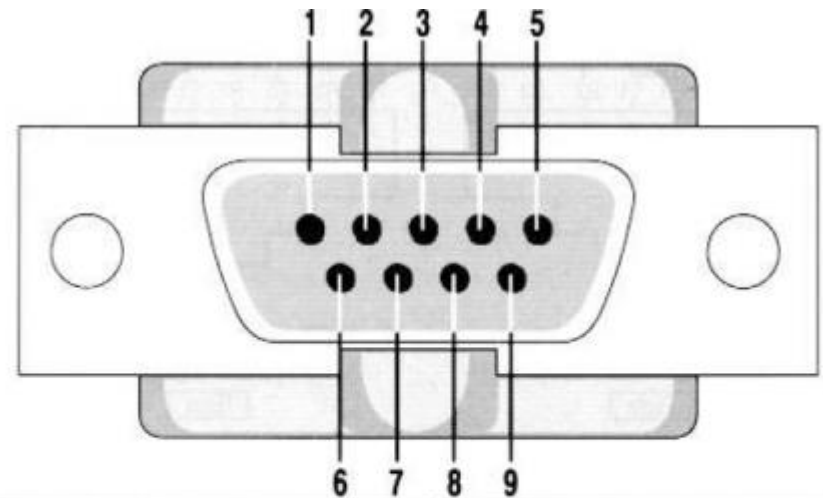
USART : Reception Asynchrone

- Si la FIFO est pleine, OERR passe à 1.
- Pour le remettre à zéro, il faut déconnecter le module USART via CREN.



USART

- Implémentation pour le RS-232 :
 - Dans le modèle 9 pins illustrés, les signaux asynchrones "Rece



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Bus de communication

CAN – CONTROL AREA NETWORK

A solid orange horizontal bar at the bottom of the slide.

CAN (Control Area Network)

- Créé par BOSCH en 1983
- Motivation :
 - Consommation de câble important
 - Fiabilité
 - Environnement perturbé
 - Vitesse de débit
 - Occupation du bus
 - ...

CAN (Control Area Network)

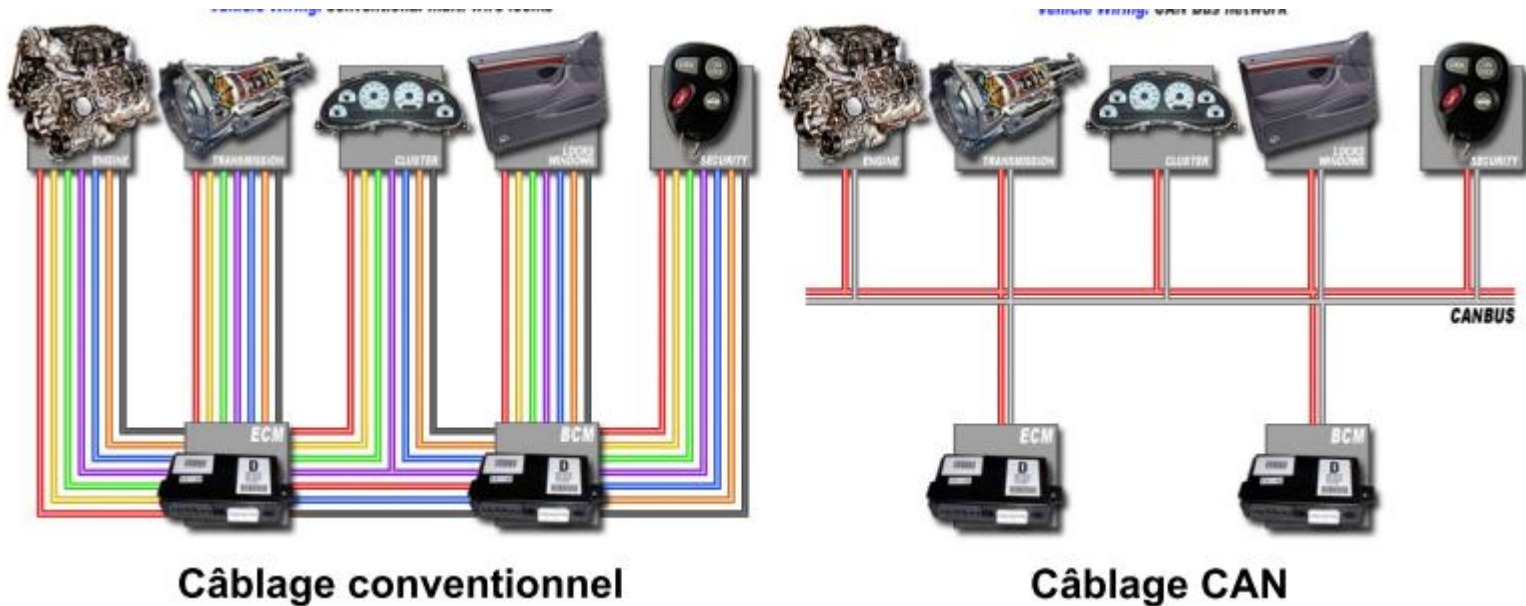
- Caractéristiques :
 - Série
 - Asynchrone
 - Maître/Esclave (multi-maîtres)
 - Half duplex
- Vitesse : 1Mbit/s (<40m) et 125kb/s (<500m)
- occupation monte à 70% -80%



origines

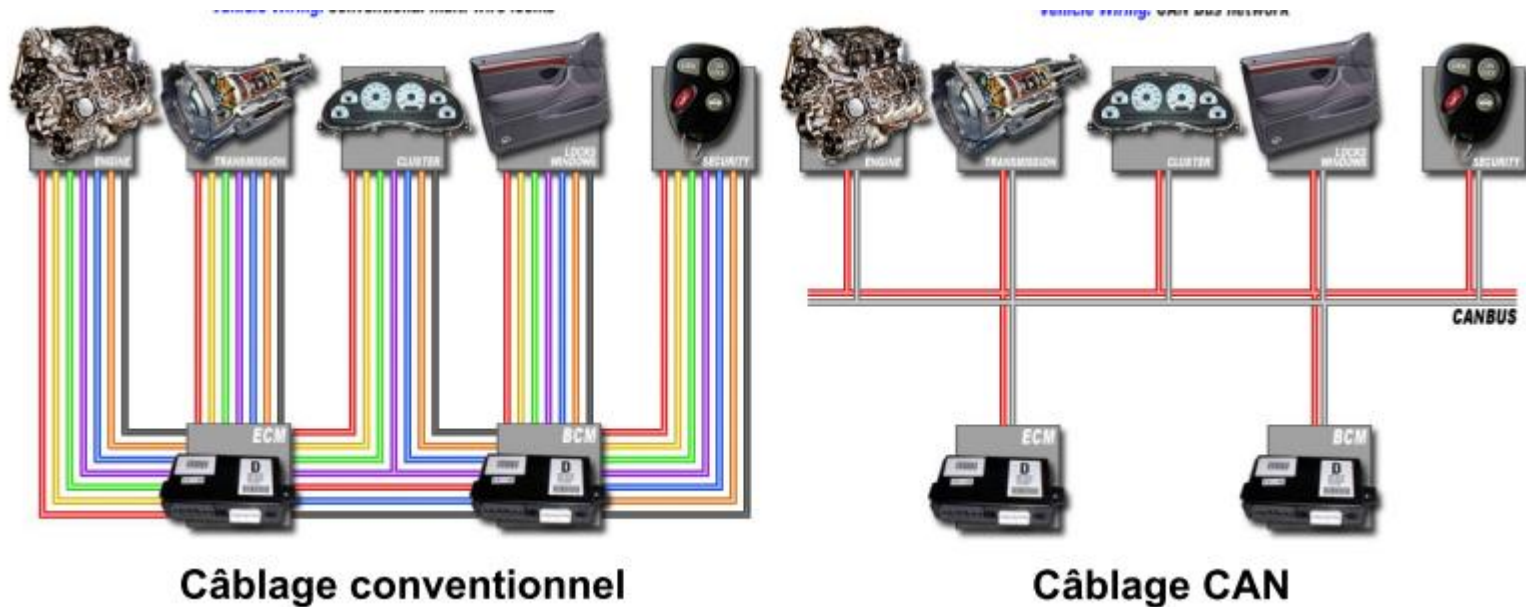
Constat de l'automobile :

- 2000m de câbles et 1800 connexions (95) => la fiabilité et la sécurité ne sont plus assurés
- De plus en plus de capteurs et d'actionneurs...
- Besoin de sécurité (ABS, ESP,AIR-BAG,..)



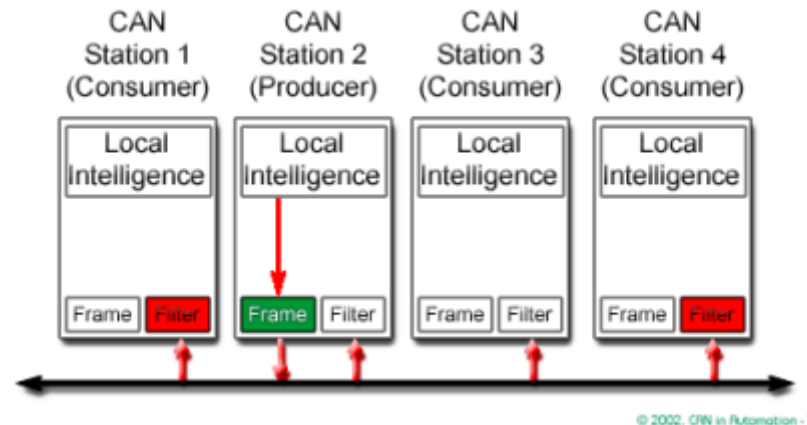
principe

Tous les nœuds reçoivent le message CAN. Chaque nœud récepteur décide de l'intérêt du message émit. Il est possible d'interroger un nœud en particulier.

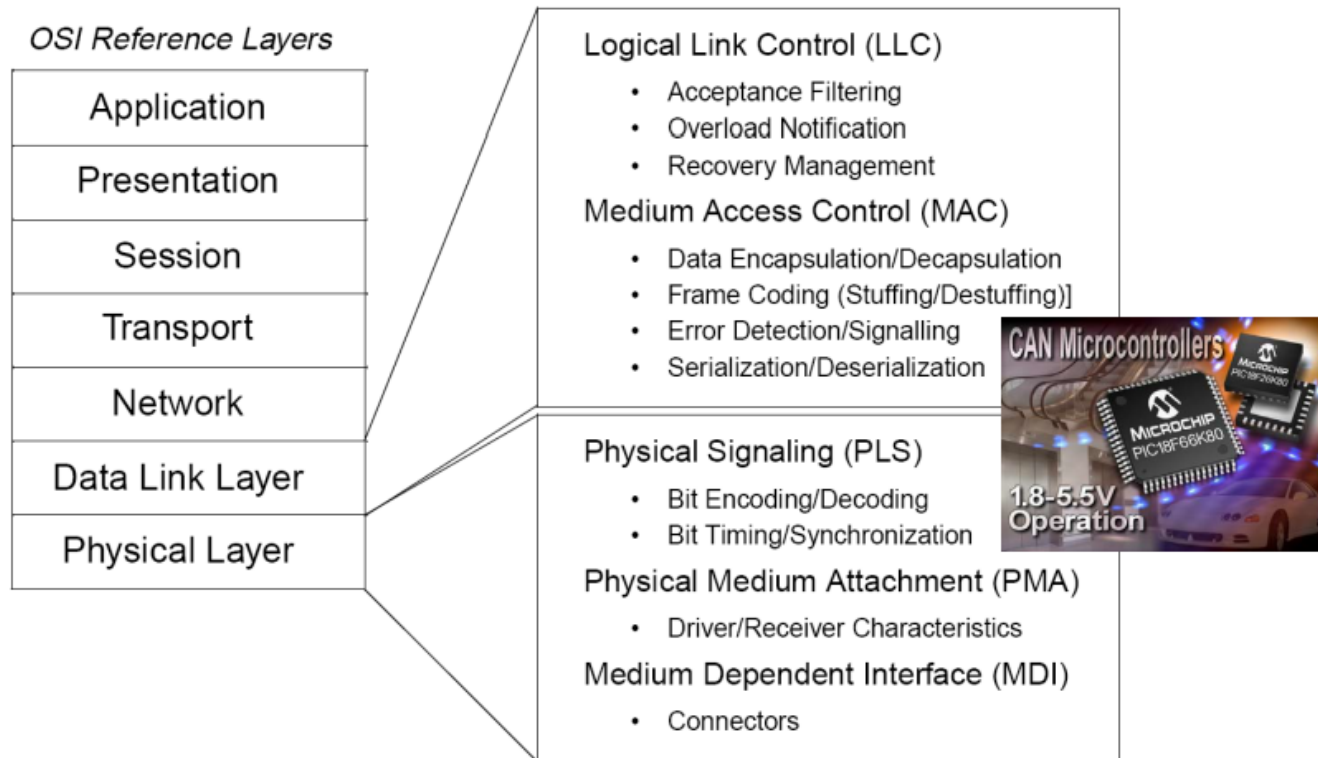


ISO 11898

Mécanisme de communication par diffusion (broadcast)



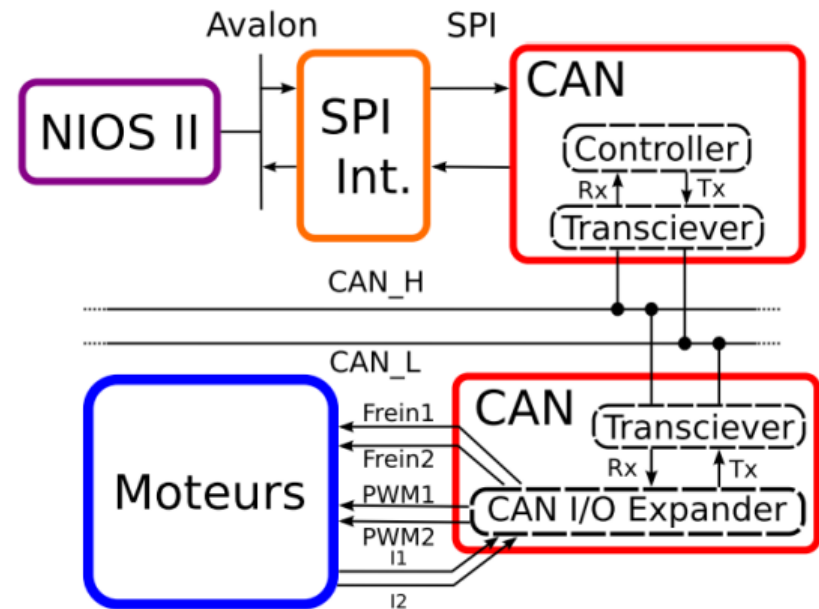
CAN (Control Area Network)



Les sous-couches LLC, MAC et PLS sont traités par les circuits contrôleur de bus CAN (µC, circuits spécialisés)

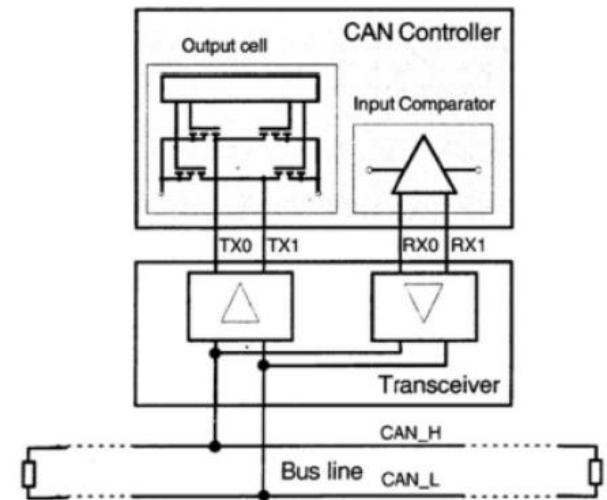
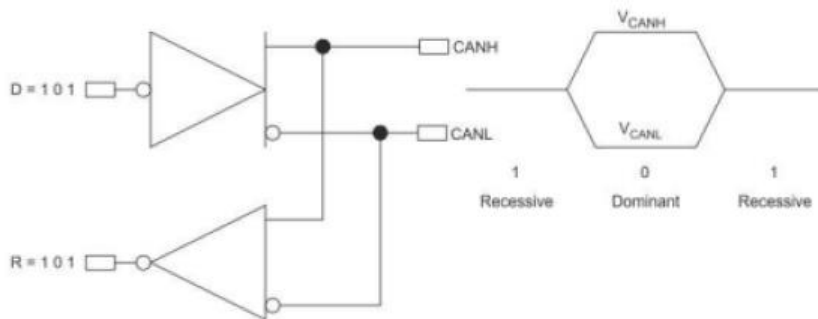
CAN (Control Area Network)

- Ensemble CAN:
 - Référence de Controller : MCP2515
 - Référence de Transceiver : MCP2551
 - Référence d'I/O expander : MCP25050



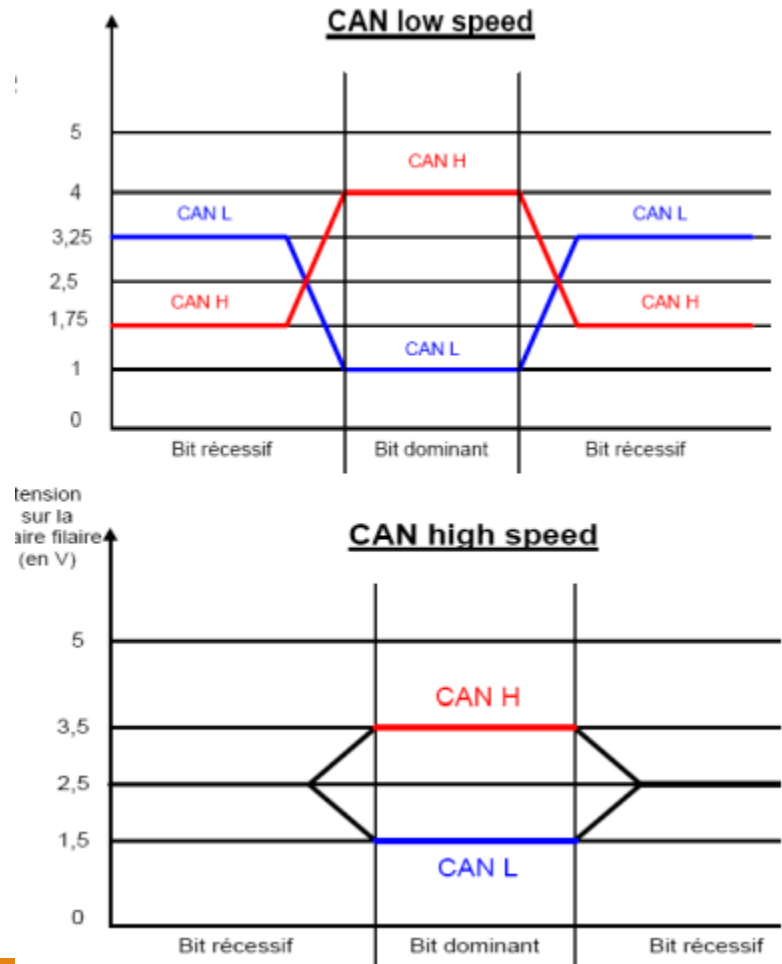
CAN (Control Area Network)

- Intérêt annuler les perturbations électromagnétiques.
- Le '0' est dominant, le '1' est récessif.
- Permet de faire varier certains paramètres plus vite.
- Permet un arbitrage non-destructif.



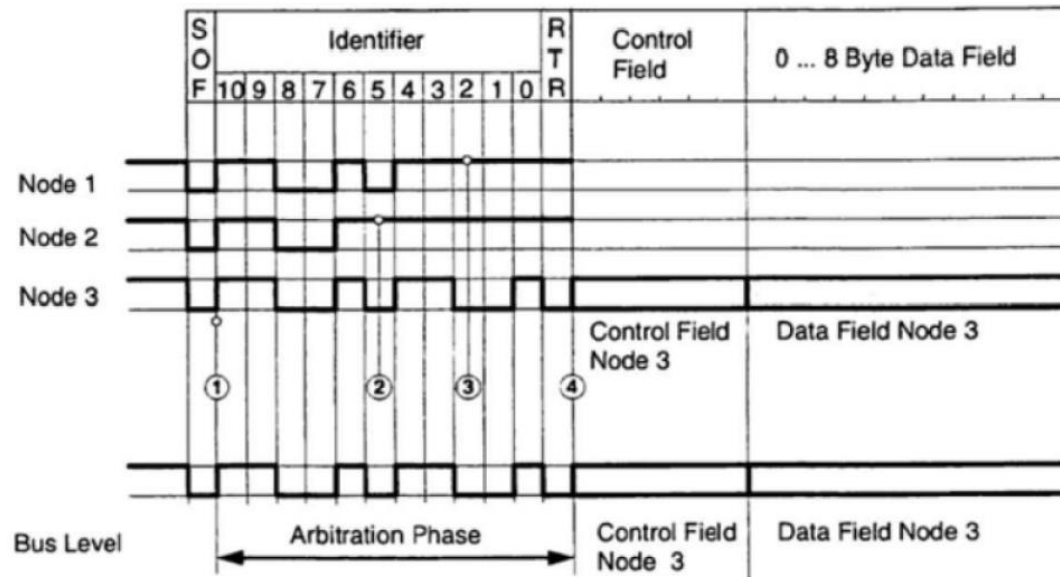
CAN (Control Area Network)

- Intérêt annuler les perturbations électromagnétiques.
- Le '0' est dominant, le '1' est récessif.
- Permet de faire varier certains paramètres plus vite.
- Permet un arbitrage non-destructif.



CAN (Control Area Network)

- Comme pour l'I2C, la dominance du "0" permet d'arbitrer les maîtres!

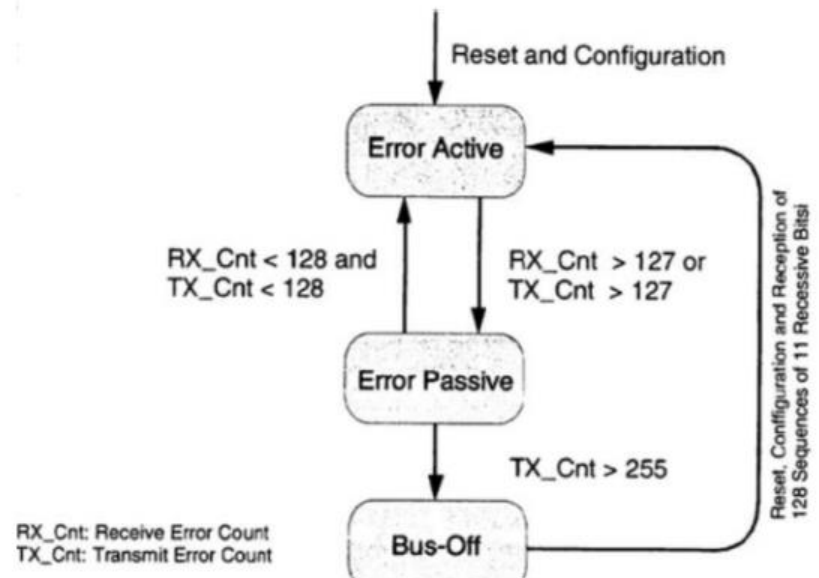


CAN : Compteur d'erreur

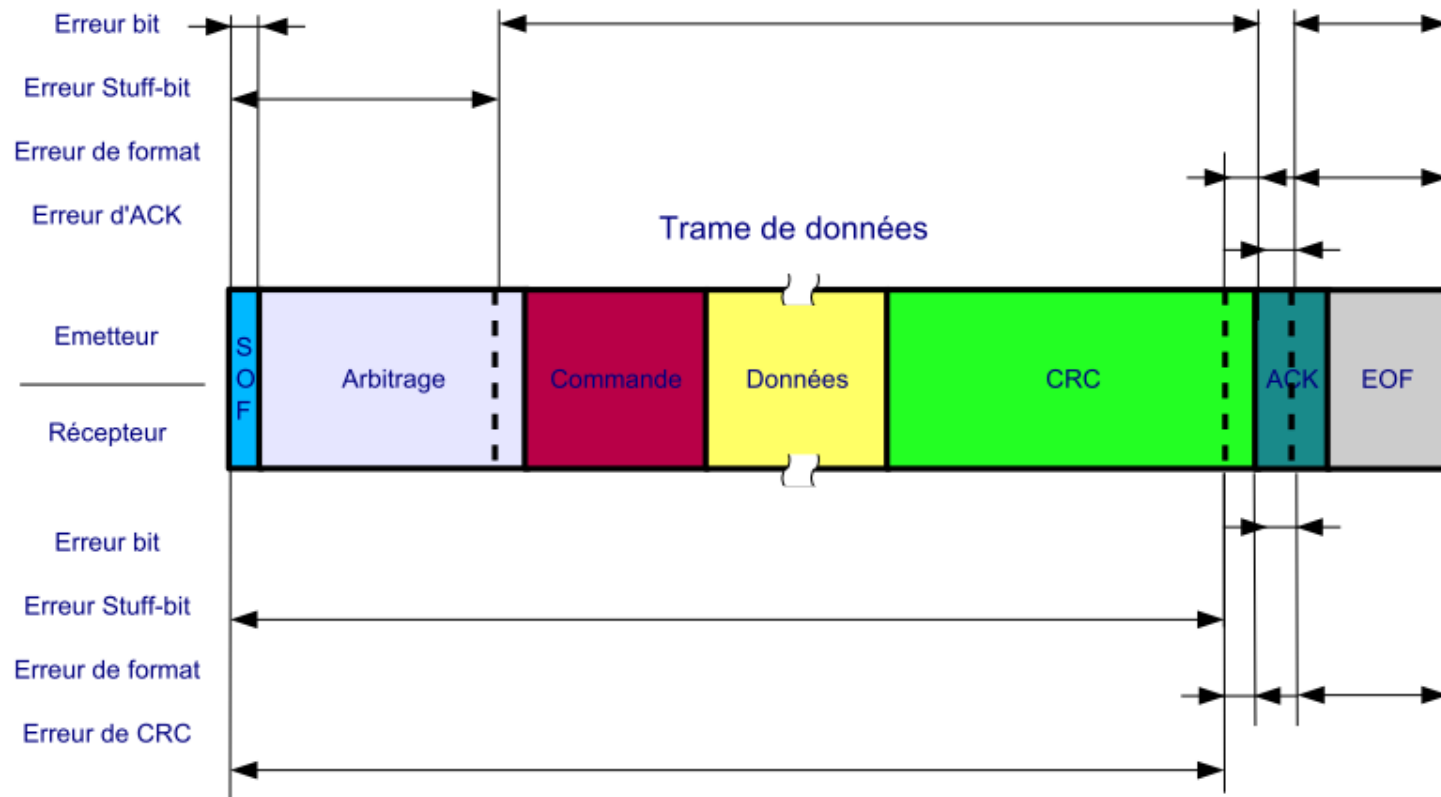
- On parle de confinement d'erreurs qui peuvent être temporaires (glitch) ou permanentes (composant défaillant)
- Un noeud défaillant pourra être éliminé
- Il y a trois états possibles :
 - 1. error-active : peut prendre normalement part à la communication sur le bus. S'il détecte une erreur, il la transmet en mode Actif.
 - 2. error-passive : peut prendre part à la communication sur le bus. S'il détecte une erreur, il la transmet en mode Passif (= noeud à problème).
 - 3. bus-off ne peut influencer le bus.

CAN : Compteur d'erreurs

- Deux compteurs d'erreurs sont implémentés dans chaque noeud.
 - Erreur de transmission
 - Erreur de réception
- Ils varient différemment en fonctions des erreurs et des réussites.
- Il est en mode
 - erreur active si $RX_Cnt < 128$ et $TX_Cnt < 128$
 - Erreur passive |e| 128 et 256
 - Bus-off si erreur de transmission > 256



CAN : origines des erreurs



CAN (Control Area Network)

- On ne précise plus la puce destinataire mais la signification du message via un identifiant (ID). La puce regarde si cela l'intéresse.
- Les ID doivent être unique
- Il permet de donner une priorité des messages!

Input Messages (to MCP25050)																											
	Standard ID													Data Bytes													
	1	9	8	7	6	5	4	3	2	1	0	R	I	DLC													
	0											T	D														
												R	E														
Write Register	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	1	1	3	addr	mask	value	n/a	n/a	n/a	n/a	n/a

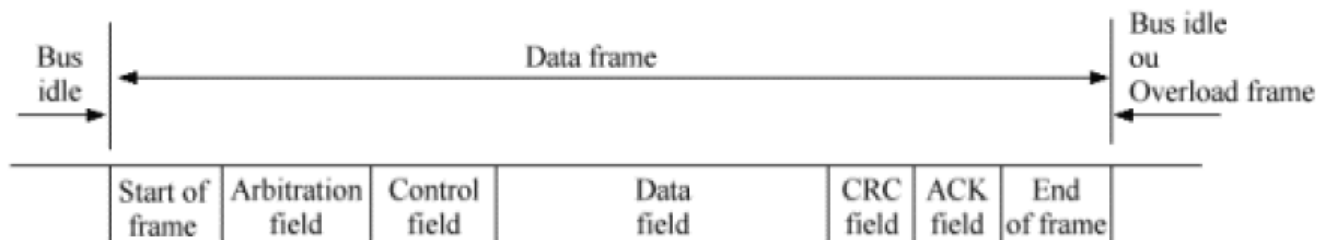
CAN (Control Area Network)

- Il y a quatre types de messages :
 - 1. Les data frames – transporte des données
 - 2. Les remote frames – demande d'une info et réponse via le même identifiant.
 - 3. Error Frames sont transmises par un noeud qui détecte une erreur.
 - 4. Overload Frames – produit un délai entre des données

Input Messages (to MCP25050)																											
	Standard ID													Data Bytes													
	1	9	8	7	6	5	4	3	2	1	0	R	I	DLC													
	0											T	D														
												R	E														
Write Register	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	1	1	3	addr	mask	value	n/a	n/a	n/a	n/a	n/a

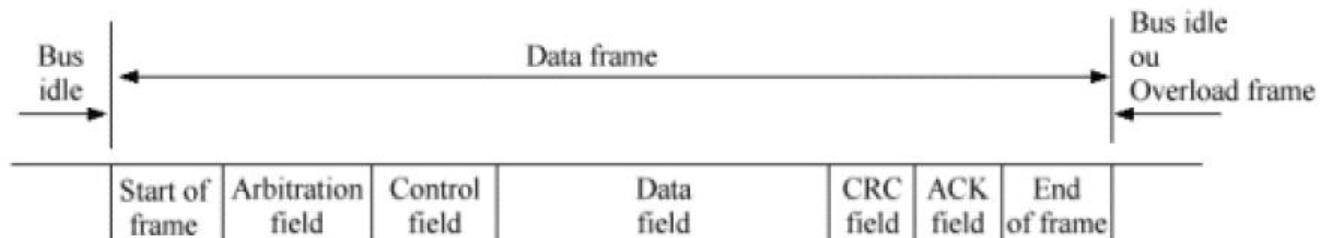
CAN (Control Area Network)

- Séquence :
 - Start of frame : un '0' unique
 - Identifiant
 - Control field : bits DLC permettent de donner le nombre de byte transmis
 - Données transmises.



CAN (Control Area Network)

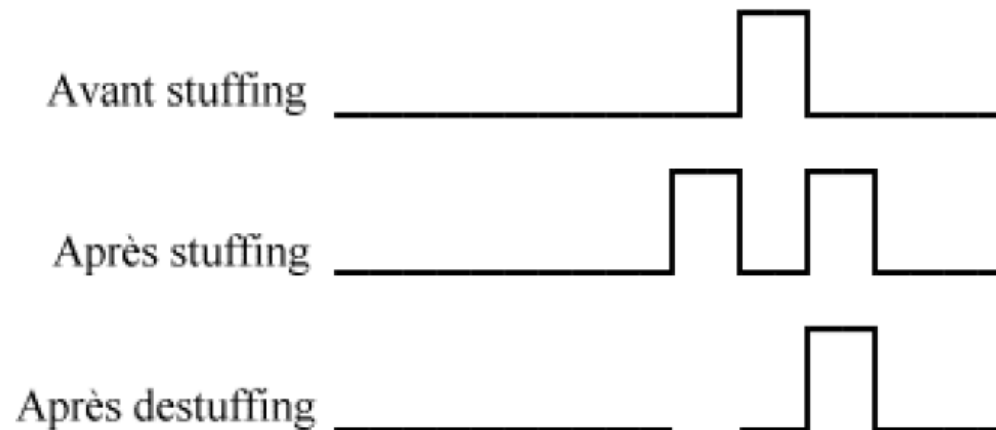
- Séquence :
 - CRC : sorte de checksum pour assurer de la bonne transmission des valeurs basée sur la parité des bits.
 - Acknowledge par le répondant
 - End of frame : 7 bits récessifs



CAN (Control Area Network)

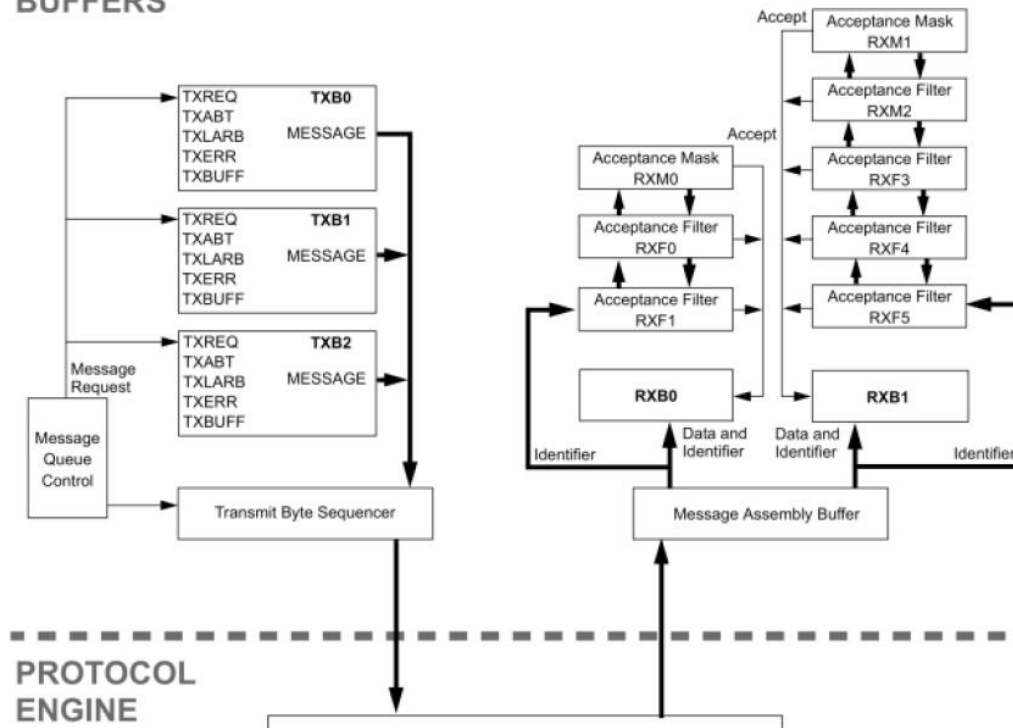
- Bit-stuffing :

- Quand un transmetteur détecte 5 bits consécutifs de même valeur dans les bits à transmettre. Il ajoute automatiquement un bit de valeur opposée. Cela permet d'éviter une désynchronisation des clocks.



CAN : Dans un PIC

BUFFERS



CAN (Control Area Network)

