



FISA S3E

BUS DE TERRAIN A4



Etablissement d'enseignement supérieur technique privé

HESAM
UNIVERSITÉ

FORMATION
D'INGÉNIEURS
SPÉCIALITÉ

**SYSTÈMES
ELECTRIQUES ET
ELECTRONIQUES
—
EMBARQUÉS**

Stéphane DUVAL
Nicolas ANTINI

Sommaire

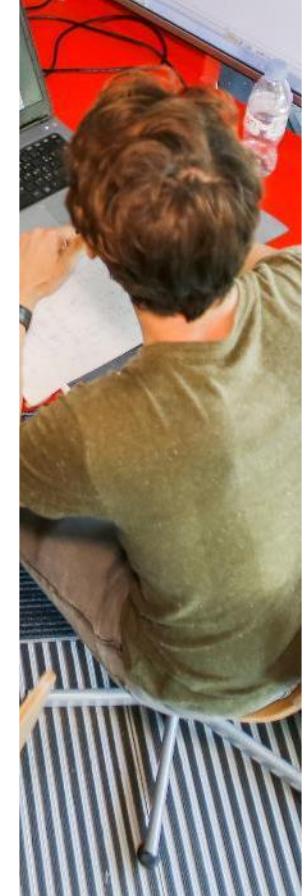
1. Présentation Générale

3

2. FISA S3E A4 Réseaux de TERRAINS/BUS SPI

12

• 2.1 Généralités	13
• 2.2 Origines	15
• 2.3 Philosophie	16
• 2.4 Synoptique d'une liaison SPI Maître-Esclave	17
• 2.5 Propriétés physiques du bus SPI	18
• 2.6 Synoptique d'une liaison SPI Maître-Multi-Esclaves	24
• 2.7 Bus SPI et niveaux électriques	25
• 2.8 Les différents autres noms utilisés avec un bus SPI	27
• 2.9 Avantages et Inconvénients du bus SPI	28
• 2.10 Constitution pour un système à microcontrôleur PIC	29
• 2.11 Compléments SD carda	33
• 2.12 Travaux pratiques	35





1

FISA S3E Réseau de terrain

Présentation générale

Stéphane DUVAL

Nicolas ANTINI

2020

Mallettes pédagogiques



Résumé :

- Microcontrôleur
18f87K22
- CPU à 33Khz et 11
MHz
- 8 Ko de RAM
- Ecran graphique
- Clavier tactile
- Passerelles :
 - I2C
 - 1 wire,
 - RS232/485
 - CAN
 - Arinc 429



1-Wire®

RS 485
RS 232



CAN

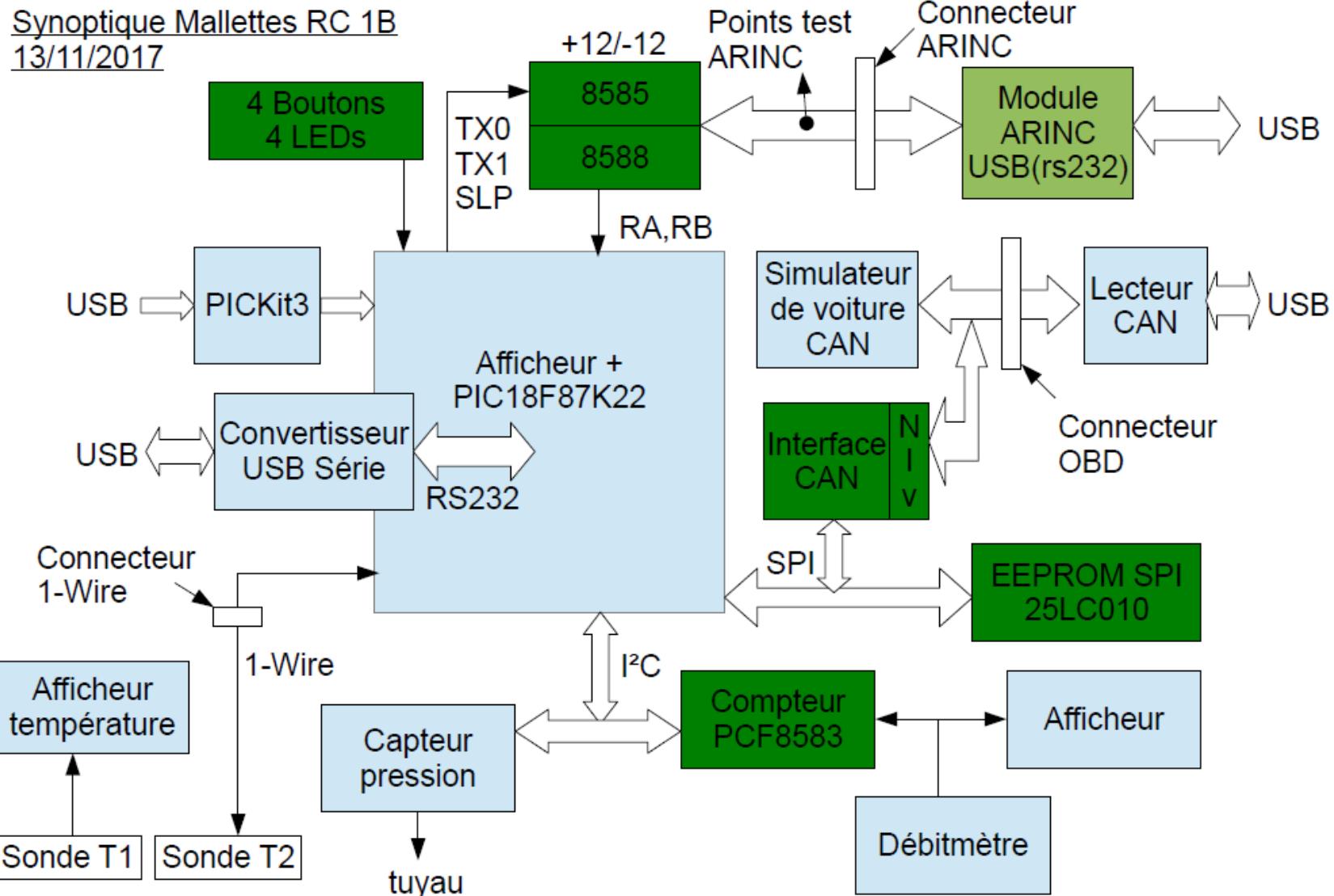


FISA S3E A4 Réseaux de TERRAINS

1.1 Synoptique

Implantation
générale :

Connection de
l'afficheur graphique

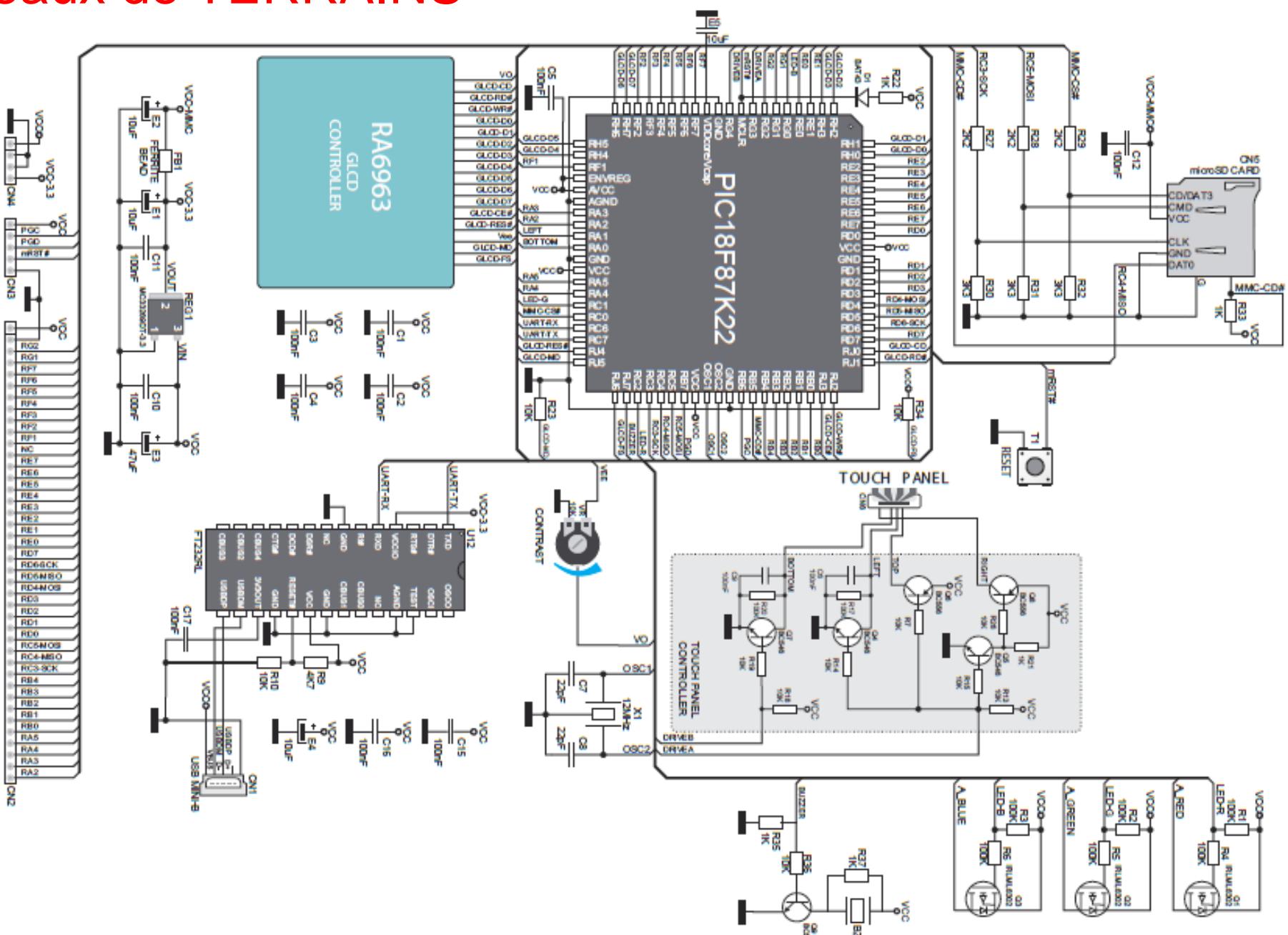


FISA S3E A4 Réseaux de TERRAINS

1.1 Connexions

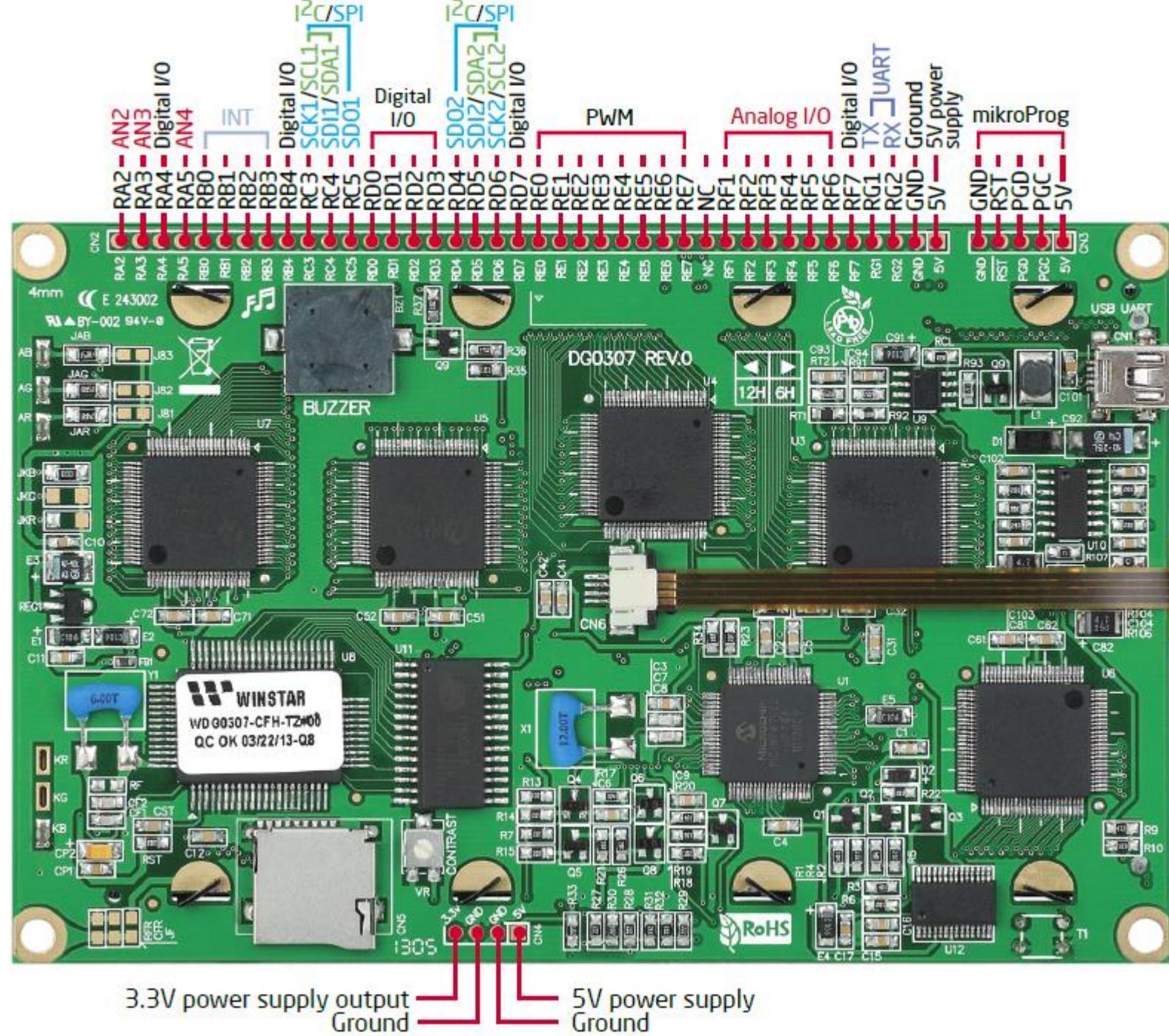
Implantation générale :

Connection de l'afficheur graphique



Connection des bus

Implantation générale :



- UART Lines
- Interrupt Lines
- SPI Lines
- I2C Lines
- Analog Lines

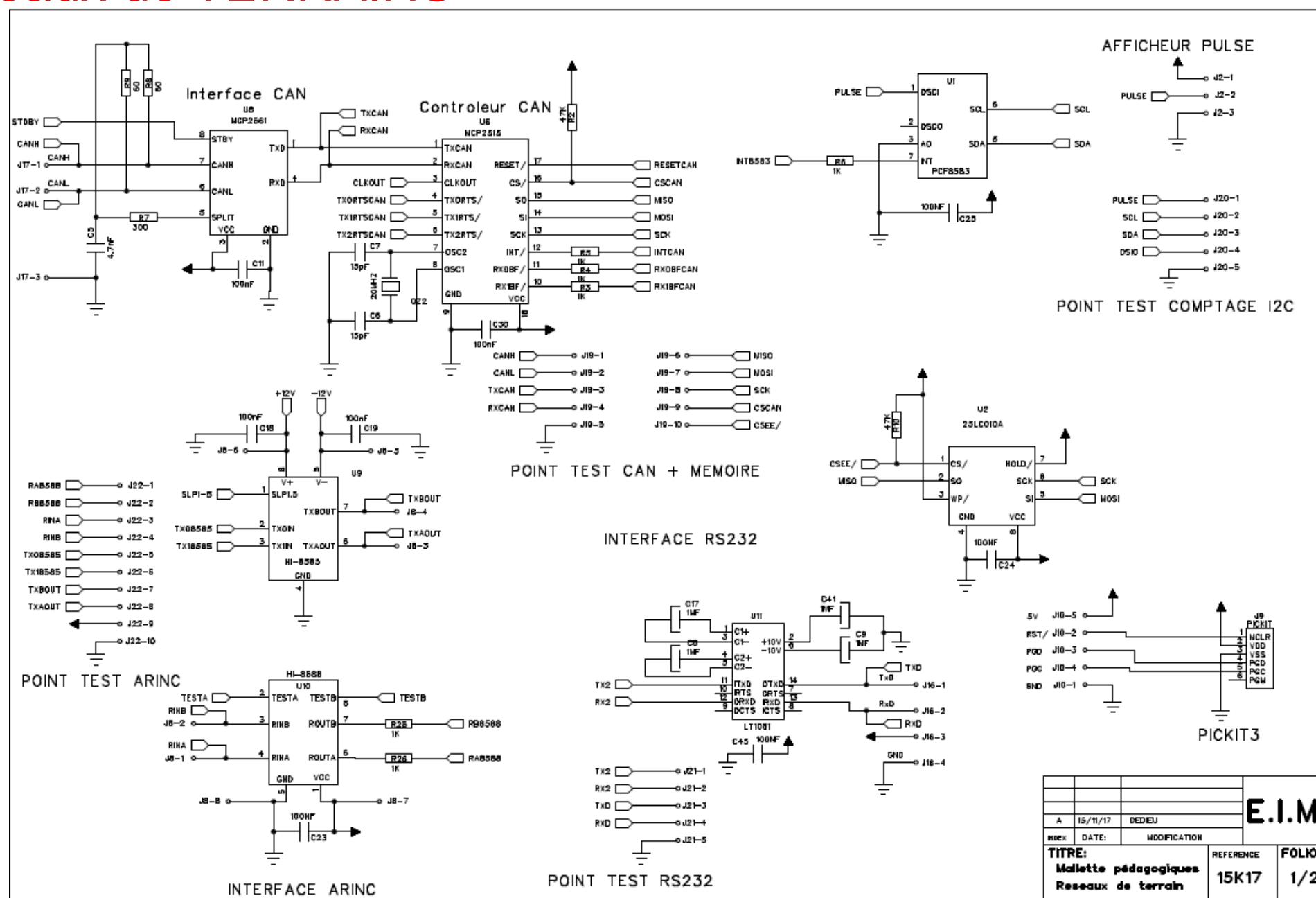
FISA S3E A4 Réseaux de TERRAINS

1.1 Connexions

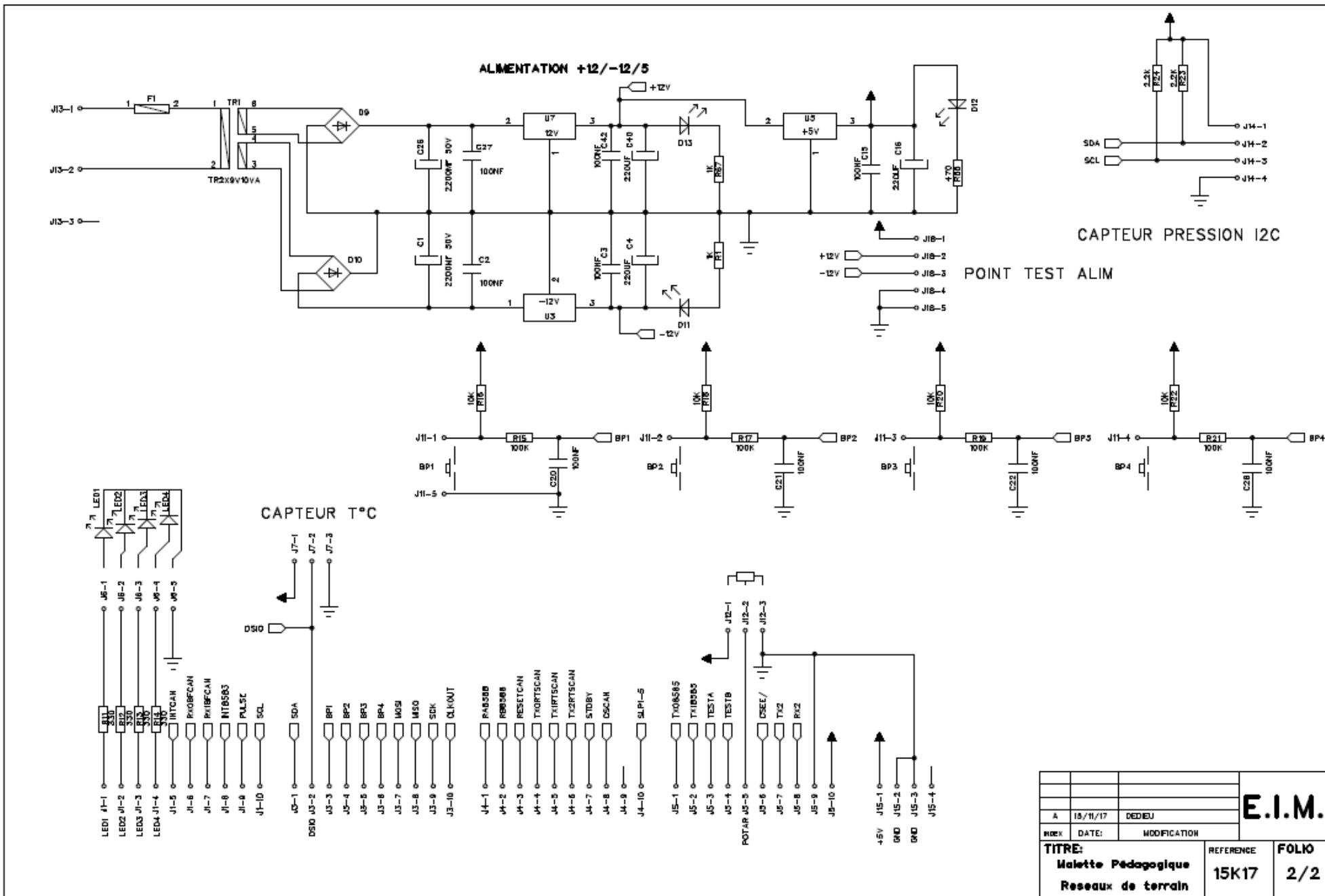
1 - RA2	LED 1	10 PT 1
2 - RA3	LED 2	10 PT 2
3 - RA4	LED 3	10 PT 3
4 - RA5	LED 4	10 PT 4
5 - RB0	INT Contrôleur CAN	10 PT 5
6 - RB1	RX0BF Contrôleur CAN	10 PT 6
7 - RB2	RX1BF Contrôleur CAN	10 PT 7
8 - RB3	INT8583	10 PT 8
9 - RB4	PULSE	10 PT 9
10 - RC3	SCL Compteur I2C + Capteur pression I2C	10 PT 10
11 - RC4	SDA Compteur I2C + Capteur pression I2C	10 PT 1
12 - RC5	1-Wire DSIO	10 PT 2
13 - RD0	BP 1	10 PT 3
14 - RD1	BP 2	10 PT 4
15 - RD2	BP 3	10 PT 5
16 - RD3	BP 4	10 PT 6
17 - RD4	SO Contrôleur CAN/EE	10 PT 7
18 - RD5	SI Contrôleur CAN/EE	10 PT 8
19 - RD6	SCK Contrôleur CAN/EE	10 PT 9
20 - RD7	CLKOUT	10 PT 10
21 - RE0	RA8588	10 PT 1
22 - RE1	RB8588	10 PT 2
23 - RE2	RESET Contrôleur CAN	10 PT 3
24 - RE3	TX0RTS Contrôleur CAN	10 PT 4
25 - RE4	TX1RTS Contrôleur CAN	10 PT 5
26 - RE5	TX2RTS Contrôleur CAN	10 PT 6
27 - RE6	STBY MCP2561FD CAN	10 PT 7
28 - RE7	CS Contrôleur CAN pull up 47K	10 PT 8
29 - NC	-	10 PT 9
30 - RF1	SLP1,5	10 PT 10
31 - RF2	TX08585	10 PT 1
32 - RF3	TX18585	10 PT 2
33 - RF4	TESTA	10 PT 3
34 - RF5	TESTB	10 PT 4
35 - RF6	V_POT 10K	10 PT 5
36 - RF7	CS/ EEPROM SPI 25LC010 pull up 47K	10 PT 6
37 - RG1	TX2 Liaison Série secondaire	10 PT 7
38 - RG2	RX2 Liaison Série secondaire	10 PT 8
39 - GND	0 V	10 PT 9
40 - 5V	+5 V	10 PT 10

FISA S3E A4 Réseaux de TERRAINS

1.2 Schémas



FISA S3E A4 Réseaux de TERRAINS



Nom	Modifié le	Type
afficheur	29/01/2018 10:22	Fichier C
afficheur	29/01/2018 10:22	Fichier H
main	29/01/2018 10:22	Fichier H
RC_vierge	29/01/2018 10:32	Fichier C

IHM fournie :

Prototypes de fonctions dans afficheur.h

Pour aller plus loin :



Répertoire/
COMPOSANTS

```

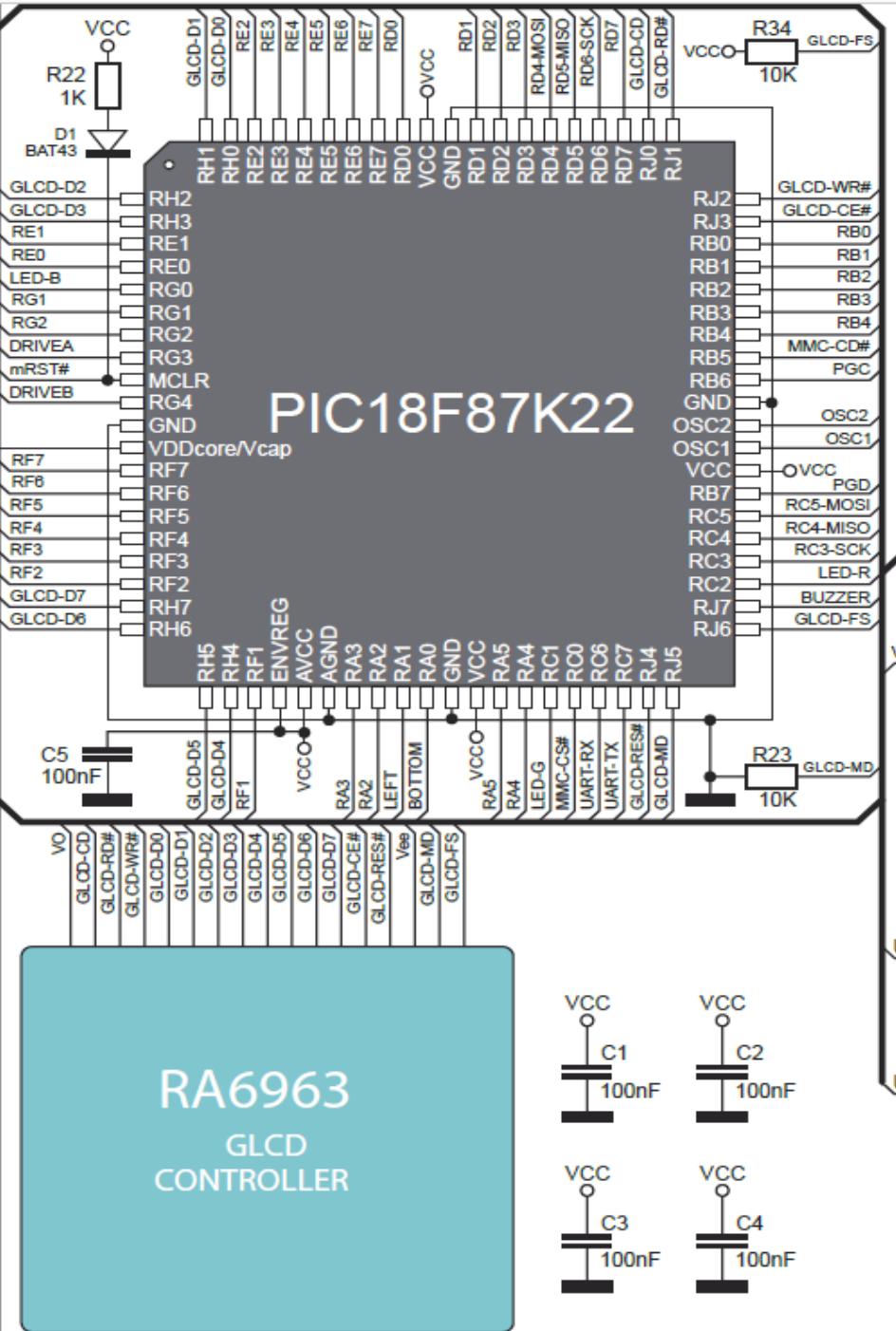
        // Commandes afficheur
#define SET_CURSOR_POINTER 0x21
#define SET_OFFSET_REGISTER 0x22
#define SET_ADDRESS_POINTER 0x24

#define SET_TEXT_HOME_ADDRESS 0x40
#define SET_TEXT_AREA 0x41
#define SET_GRAPHIC_HOME_ADDRESS 0x42
#define SET_GRAPHIC_AREA 0x43

// Constante
#define NB_COL_GRAPH 40

// Prototypes
void write_d_aff(unsigned char data);
void write_c_aff(unsigned char command);
unsigned char check_status_ok();
unsigned char wait_status_ok();
unsigned char command(unsigned char cmd);
unsigned char dlcommand(unsigned char d1, unsigned char cmd);
unsigned char d2command(unsigned char d1, unsigned char d2, unsigned char cmd);
void delai_us_char(unsigned char ucdelai);
void initialisation_afficheur();
void draw_char(unsigned char dccar);
void draw_string(unsigned char * tableau);
void draw_hex8(unsigned char octet);
void goto_lico(unsigned char ligne, unsigned char colonne);
void clear_text();
void clear_graphics();
void clear_cgram();
void plot1(unsigned char x, unsigned char y);
void plot0(unsigned char x, unsigned char y);

```



Bus SPI



2

FISA S3E
Réseau de terrain

Bus SPI

FISA S3E A4 Réseaux de TERRAINS/BUS SPI / Résumé

1 – ARCHITECTURE DE LA LIAISON SPI

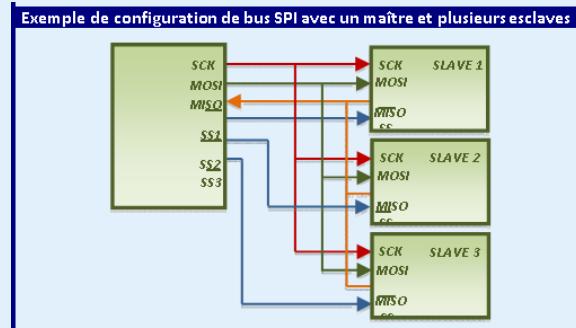
La liaison SPI est un **bus série** utilisé pour la **transmission synchrone** de données entre un maître et un ou plusieurs esclaves (**multipoints**). La transmission a lieu en **full duplex**.

MOSI (Master Out Slave In) : Sur la ligne MOSI le maître transmet des données à l'esclave.

MISO (Master In Slave Out) : Sur la ligne MISO l'esclave transmet des données au maître.

SCK (SPI Serial Clock) : Signal d'horloge, généré par le maître, qui synchronise la transmission. La fréquence de ce signal est fixée par le maître et est programmable.

SS (Slave Select) : Ce signal placé au NL0 permet de sélectionner (adresser) individuellement un esclave. Il y a autant de lignes SS que d'esclaves sur le bus.

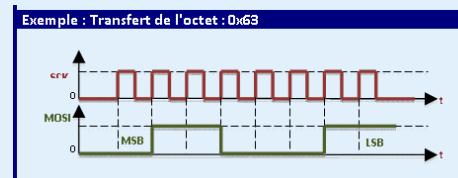


2 – TRAME DE TRANSFERT

Pour démarrer une transmission de données, le maître sélectionne un esclave en mettant sa ligne SS respective au NL0.

A chaque front d'horloge, le maître envoie un bit sur la ligne MOSI. De même il peut recevoir à chaque front d'horloge, sur la ligne MISO, un bit transmis par l'esclave. La transmission commence par le MSB.

La transmission de données est terminée dès que la ligne SS est remise au NL1.



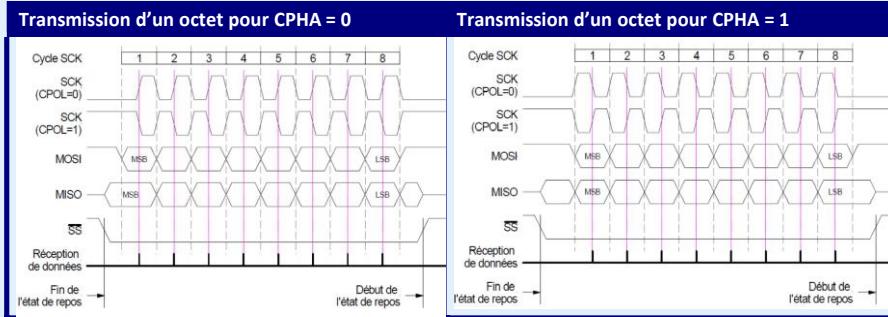
Le protocole du bus SPI permet de configurer le signal d'horloge (SCK) grâce à deux paramètres :

- **CPOL (Clock Polarity)** permet de configurer l'état logique de repos du signal d'horloge : Si **CPOL = 0** l'état de repos est le **NL0** et si **CPOL = 1** l'état de repos est le **NL1**.
- **Cpha (Clock Phase)** permet de définir le front actif, c'est-à-dire sur quel front d'horloge les données sont transmises : Si **Cpha = 0** le front actif est le **front de l'état de repos vers l'état actif** et si **Cpha = 1** le front actif est le **front de l'état actif vers l'état de repos**.

Ils en résultent quatre modes différents de transmission :

CPOL	Cpha	Etat de repos de la ligne SCK	Front actif du signal SCK
0	0	NL0	Front montant
0	1	NL0	Front descendant
1	0	NL1	Front descendant
1	1	NL1	Front montant

Pour une transmission de données correcte il est important que ces paramètres soient réglés de manière identique pour tous les appareils reliés au bus.



La fréquence du signal SCK est fixée par le maître en tenant compte des possibilités de l'esclave. Il s'agit très souvent d'un rapport de la fréquence de fonctionnement du maître ($f_{\mu C}/2$, $f_{\mu C}/4$, $f_{\mu C}/8$).

Pour un µC PIC, les réglages de la fréquence et des paramètres CPHA et CPOL sont disponibles par le biais d'un registre interne spécifique 8 bits (SPCON : Serial Peripheral CONtrol register) dans lequel chaque bit représente un paramètre. **Cas du 18F87K22 : Registres SSPxCON1 et SSPxSTAT.**

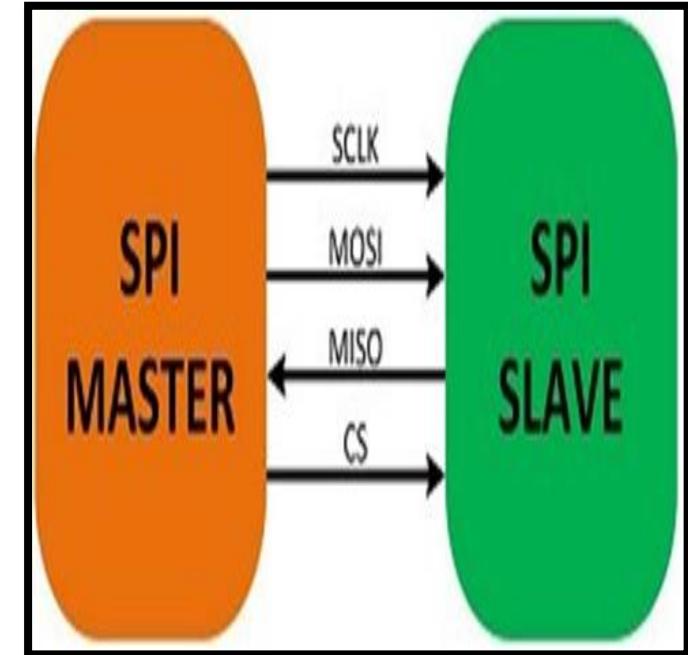
7	6	5	4	3	2	1	0
SPR2	SPEN	---	---	CPOL	Cpha	SPR1	SPRO
N° de bit						Description	
7, 1, 0						Serial Peripheral Rate : Bits permettant de régler la fréquence du signal d'horloge SCK	
6						Serial Peripheral Enable : 0 = Interface SPI inactive ; 1 = Interface SPI active	
3						Clock Polarity : 0 = Etat de repos des lignes au NL0 ; 1 = Etat de repos des lignes au NL1	
2						Clock Phase : 0 = Donnée acquise lors du front du signal SCK qui quitte l'état de repos et 1 = Donnée acquise lors du front du signal SCK qui retourne à l'état de repos	

[SPR2 SPR1 SPRO]	Fréquence d'horloge	Division de fréquence
000	$f_{\mu C}/2$	2
001	$f_{\mu C}/4$	4
010	$f_{\mu C}/8$	8
011	$f_{\mu C}/16$	16
100	$f_{\mu C}/32$	32
101	$f_{\mu C}/64$	64
110	$f_{\mu C}/128$	128
111	Horloge Externe	

FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

2.1 Généralités

Une liaison SPI (Serial Peripheral Interface) est un bus de données série synchrone qui opère en mode Full-duplex. Les circuits communiquent selon un schéma maître-esclaves, où le maître s'occupe totalement de la communication. Plusieurs esclaves peuvent coexister sur un même bus, dans ce cas, la sélection du destinataire se fait par une ligne dédiée entre le maître et l'esclave appelée chip select.



« **SCLK** — Serial Clock, Horloge (généré par le maître)

MOSI — Master Output, Slave Input (généré par le maître)

MISO — Master Input, Slave Output (généré par l'esclave)

CS — Chip(Slave) Select, Actif à l'état bas (généré par le maître)

FISA S3E A4 Réseaux de TERRAINS/ BUS SPI

2.2 Origines

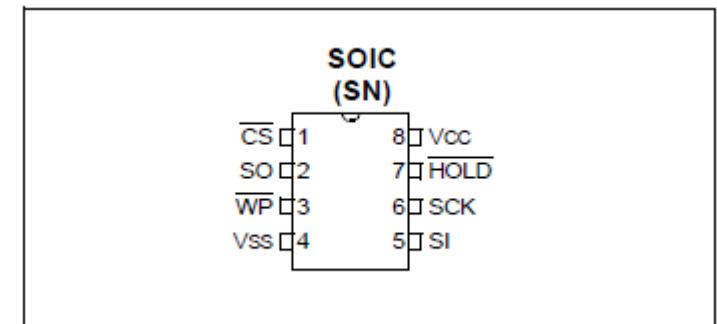
Le bus série SPI (Serial Peripheral Interface), à initialement été développé par Motorola.

D'autres fabricants (Microchip, Atmel, Texas Instrument...) ont adopté pour ce type de liaison et de nombreux composants sont apparus (mémoires, capteurs, micro contrôleurs...).

EXEMPLE : (Valise pédagogique Réseaux de terrains)



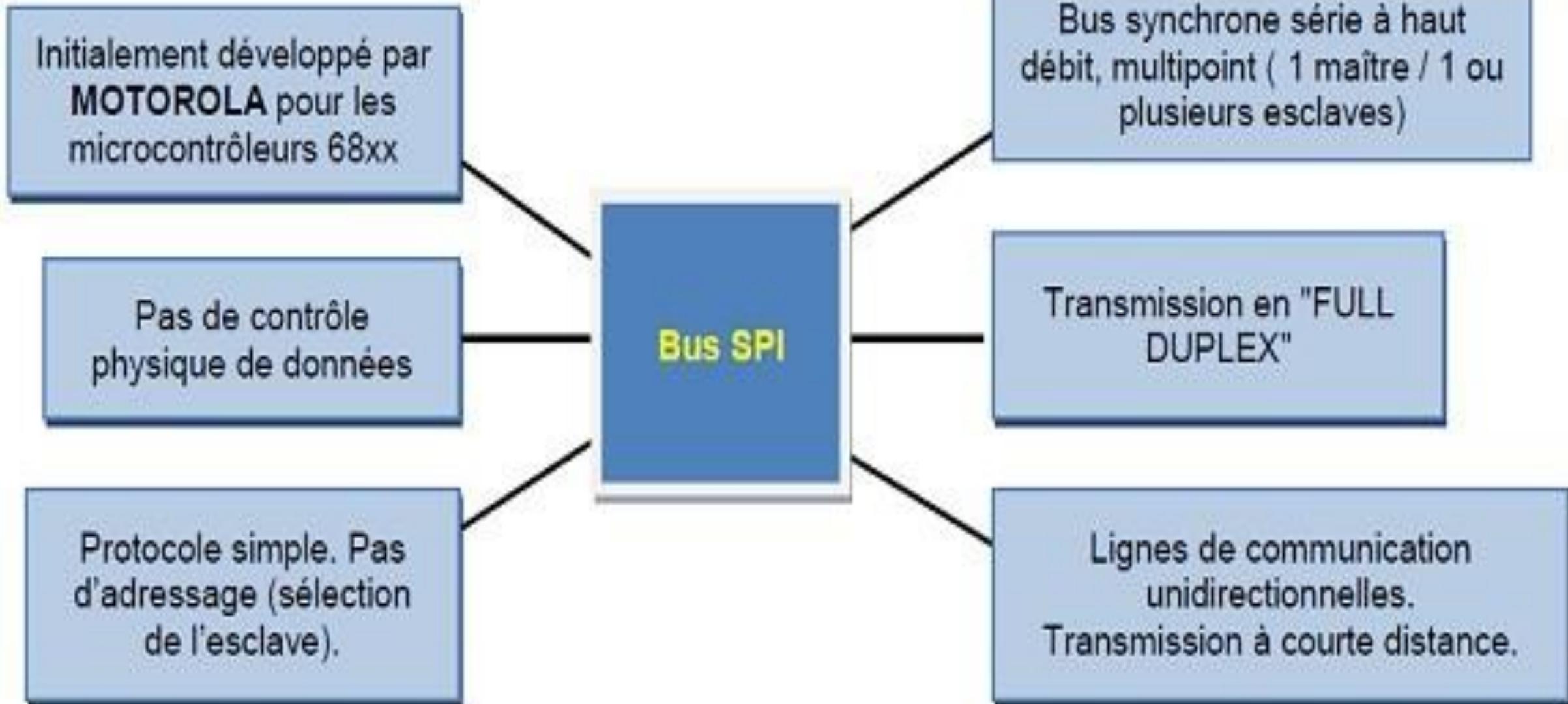
**25LC010A
25LC020A
25LC040A**
1K-4K SPI Serial EEPROM High Temp Family Data Sheet



Part Number	Density (bits)	Organization	VCC Range	Max. Speed (MHz)	Page Size (Bytes)	Temp. Range	Package
25LC010A	1K	128 x 8	2.5V-5.5V	5	16	H	SN
25LC020A	2K	256 x 8	2.5V-5.5V	5	16	H	SN
25LC040A	4K	512 x 8	2.5V-5.5V	5	16	H	SN

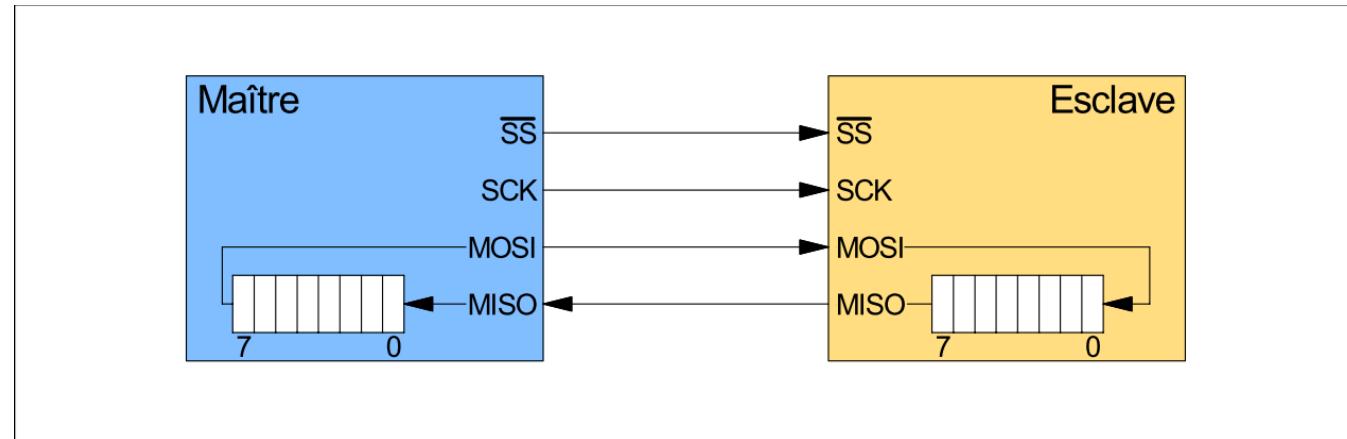
FISA S3E A4 Réseaux de TERRAINS/ BUS SPI

2.3 Philosophie



FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

2.4 Synoptique d'une liaison SPI Maître-Esclave



On utilise le principe du registre à décalage. Dans le cas ci-dessus, en 8 périodes d'horloge, l'octet passe du registre du maître à celui de l'esclave et réciproquement le contenu du registre d'esclave est passé dans celui du maître (full-duplex : simultanéité des transferts). Puisqu'il ne peut pas y avoir de collisions lors du transfert, il n'y a pas besoin d'arbitrage.

FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

2.5 Propriétés physiques du bus SPI

1. Le support physique

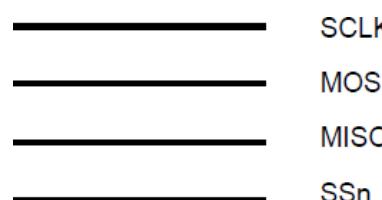
La ligne est constituée de 3 fils auxquels il faut ajouter les fils de sélection d'esclave.

Les données échangées sont des octets. La transmission s'effectue sur 2 fils monodirectionnels (nommés MOSI, MISO).

Une horloge indépendante fixée par le maître synchronise les échanges (en général sur front).

La fréquence de l'horloge de transmission est comprise entre 1 Mhz et 20 Mhz (selon les performances des circuits reliés au bus).

Il n'y a pas d'adressage des esclaves (comme sur un bus i2C par exemple). L'esclave devient actif au moyen d'une ligne de sélection de boîtier dédiée (généralement active à l'état bas).



SCLK (serial clock) :

Horloge du bus (produite par le maître)

MOSI (Master Out Slave In) :

Données du maître vers l'esclave actif

MISO (Master In Slave Out) :

Données de l'esclave actif vers le maître

SSn (Slave Select n) :

Sélection de l'esclave n à destination de la transmission

La fréquence d'horloge de transmission (maître) est paramétrable :

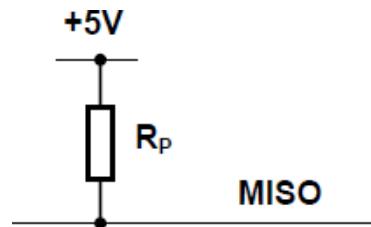
Par exemple pour un AT80C5112 en mode maître, les 3 bits du registre SPCON (Serial Peripheral CONtrol register) SPR2, SPR1 et SPR0 permettent de choisir parmi 7 fréquences, obtenues par division de la fréquence de l'horloge du microcontrôleur.

SPR2	SPR1	SPR0	Fréquence de l'horloge SPI
0	0	0	$F_{\mu c}/2$
0	0	1	$F_{\mu c}/4$
0	1	0	$F_{\mu c}/8$
0	1	1	$F_{\mu c}/16$
1	0	0	$F_{\mu c}/32$
1	0	1	$F_{\mu c}/64$
1	1	0	$F_{\mu c}/128$

5.1 Polarisation de la ligne MISO

Lorsque le bus est inutilisé, ce qui revient à dire qu'aucun esclave n'est sélectionné, la ligne MISO est à l'état haute impédance, ce qui ne permet pas d'en définir l'état logique.

On évite cela par l'utilisation d'une résistance de polarisation, de 5 à 50 kOhms, qui n'a aucune influence sur la vitesse de transmission (contrairement à ce qui se passe pour un bus i2C).



5.2 Rôle des bits CPOL et CPHA (pour un AT80C5112) dans le mode de service du bus SPI

Dans les caractéristiques du bus SPI du microcontrôleur AT80C5112, on peut déterminer, grâce à deux paramètres, les fronts où les données sont transmises (acquisitions des valeurs) et les moments où elles peuvent être modifiées.

Ces deux paramètres sont les bits CPOL (ClockPolarity) et CPHA (ClockPhase).

Il existe donc 4 modes de transmission différents (voir tableau ci-dessous). Pour une transmission correcte il faut que ces paramètres soient réglés de la même manière pour tous les composants reliés au bus.

Mode SPI	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1

Le **CPOL** détermine si au repos l'horloge est au niveau BAS (CPOL=0) ou HAUT (CPOL=1).

Le **CPHA** détermine à quel front de l'horloge les données sont transmises. CPHA=0 les données sont valides au premier front d'horloge, CPHA=1 elles sont valides au deuxième front.

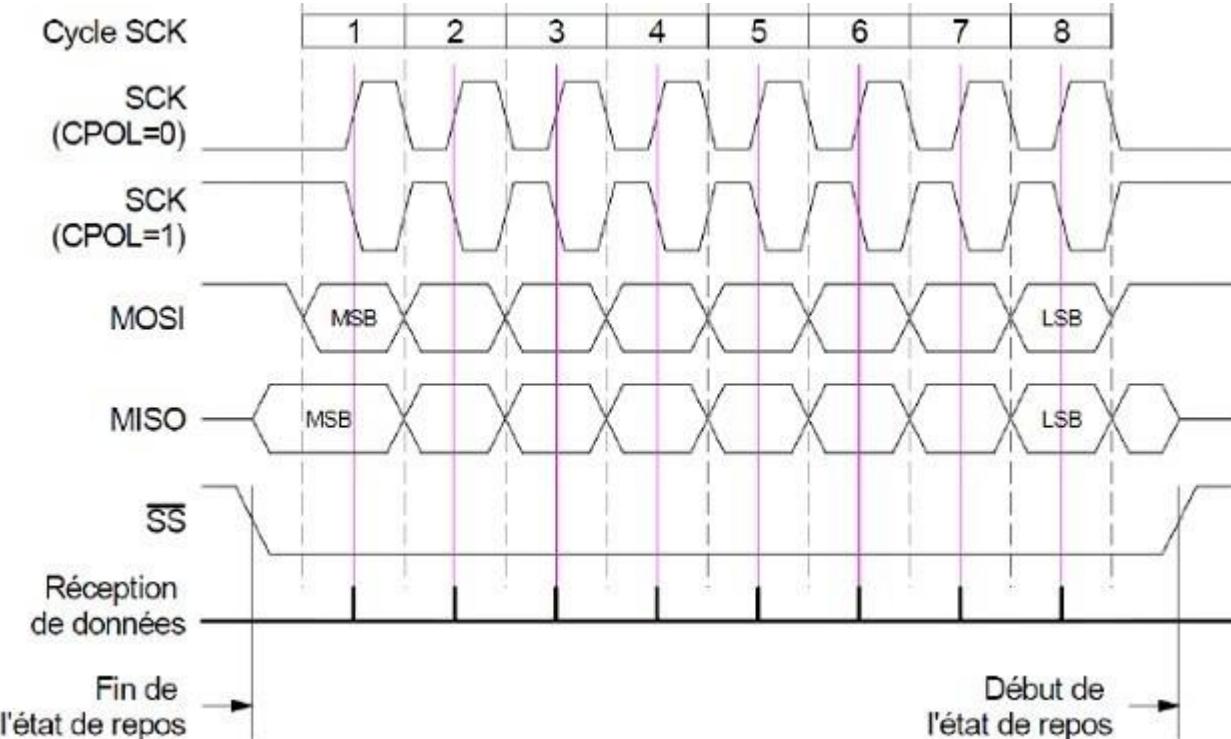
5.3 Format de transfert de données

Lorsque CPHA = 0, les données sont valides au premier front du signal d'horloge. La polarité CPOL détermine s'il s'agit d'un front montant ou descendant.

En effet, pour CPOL=0, au repos, l'horloge est au niveau BAS; le premier front est donc un front montant.

Pour CPOL=1, au repos, l'horloge est sur le niveau HAUT; le premier front est donc un front descendant.

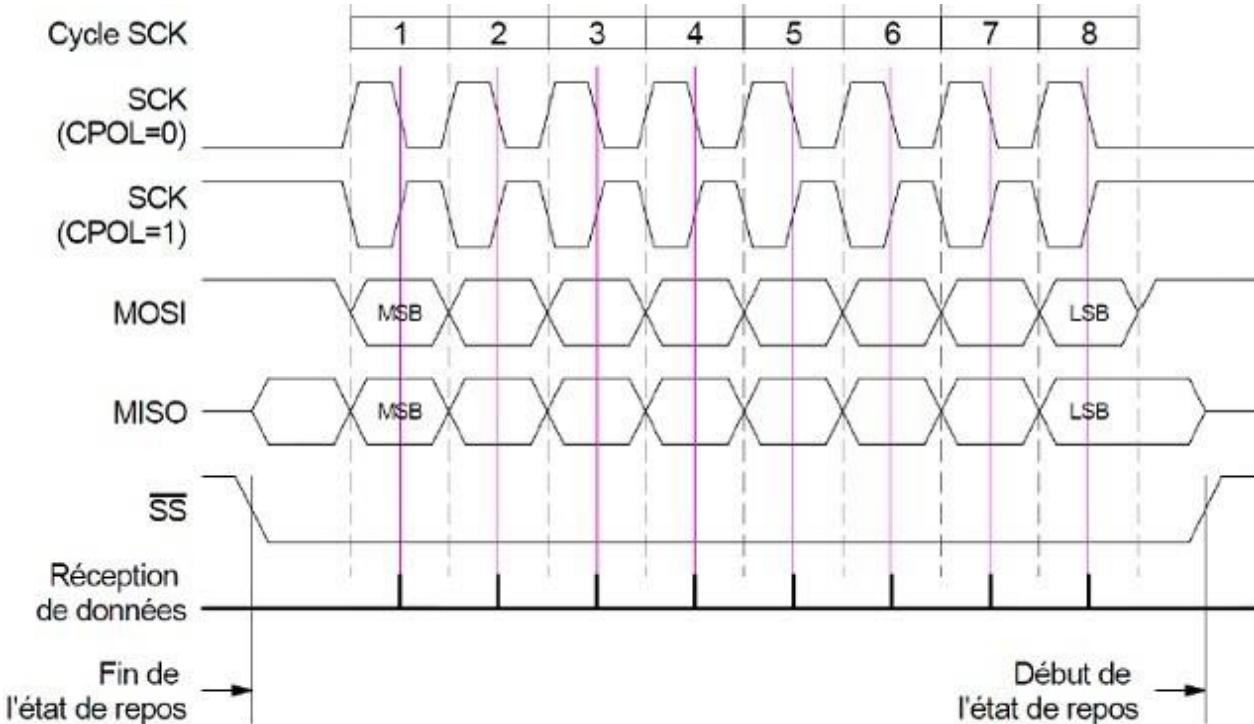
La polarité de l'horloge n'ayant pas d'influence sur le moment où le premier bit de données est valide elle n'a pas d'effet sur le format du transfert de données (voir figure ci-dessous).



Lorsque CPHA = 1, les données sont réceptionnées avec le deuxième front du signal d'horloge.

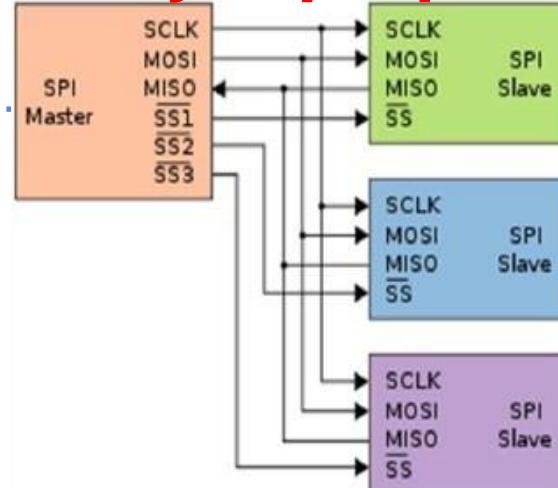
Pour CPOL=0, au repos, l'horloge est au niveau BAS et monte au niveau HAUT après le premier front, le deuxième front est donc un front descendant.

Pour CPOL=1, au repos, l'horloge est au niveau HAUT et descend au niveau BAS après le premier front; le deuxième front est donc un front montant.



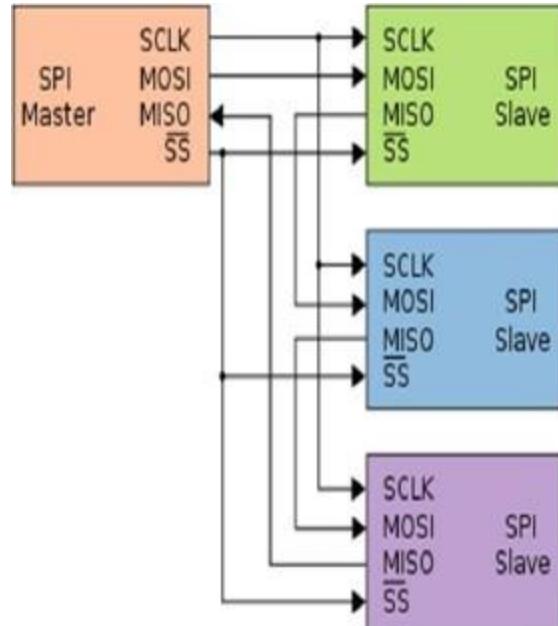
FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

2.6 Synoptique d'une liaison SPI Maître-Multi-Esclaves



Le maître sélectionne un seul et unique esclave avec lequel il veut rentrer en communication par la mise à niveau logique zéro de /SS 1 2 3, puis, après 8 fronts d'horloge, l'octet de donnée est transféré.

La patte MISO de l'esclave non sélectionné est à l'état haute impédance.



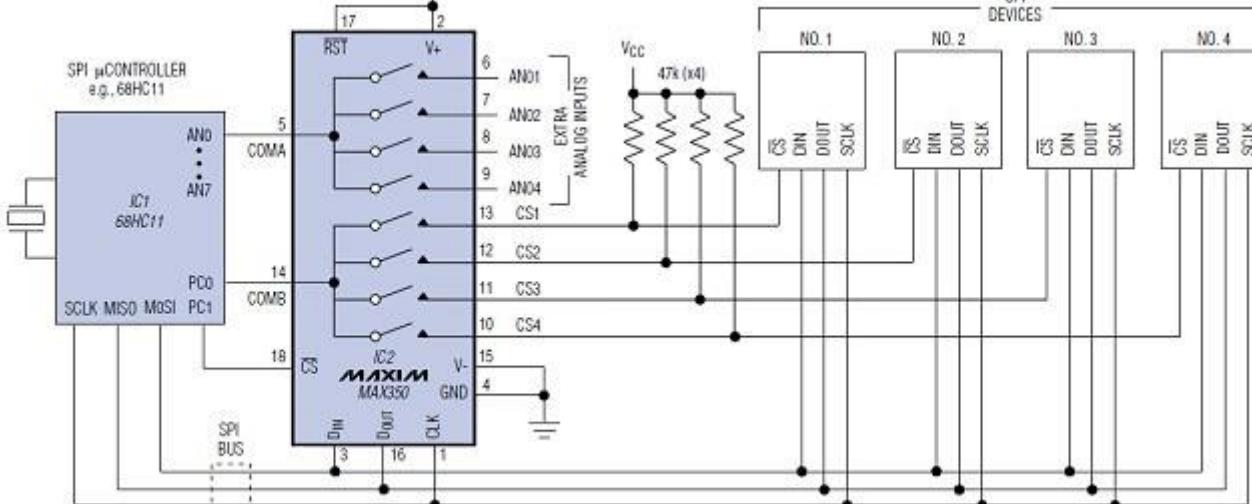
Le maître sélectionne tous les esclaves par la mise à niveau logique zéro de /SS , puis après 3*8 fronts d'horloge, les 3 octets de données sont transférés (dans le cas d'un octet par esclave).

Cette disposition permet de réduire le nombre de lignes /SS, mais en contre partie il faudra un "buffer" plus grand dans le maître (ou une gestion du soft plus élaborée).

FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

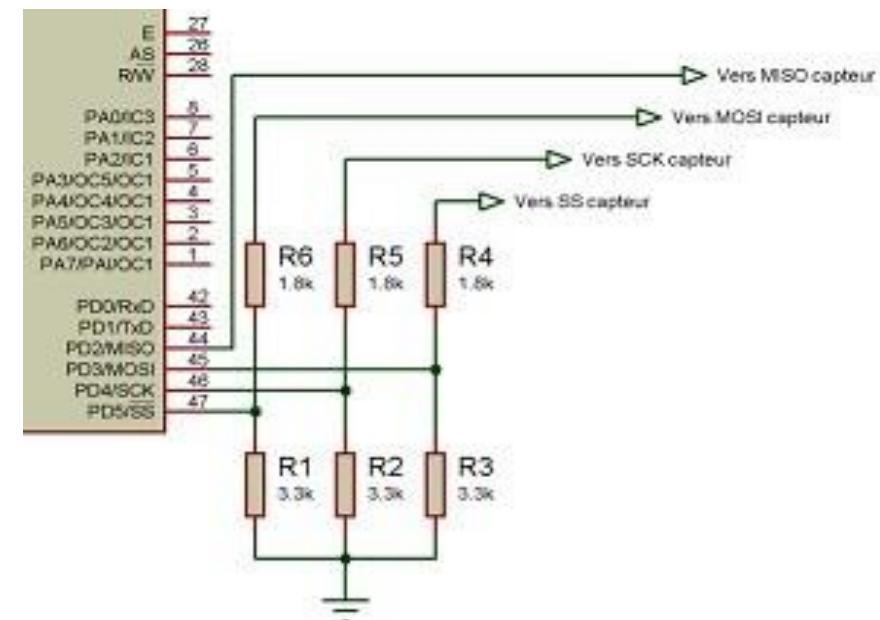
2.7 Bus SPI et niveaux électriques

Les signaux échangés sont de types TTL ou CMOS. Il pourra-t-être envisagé dans certains cas de placer des résistances de Pull-up de $47\text{k}\Omega$ (voir ci-dessous : utilisation d'un multiplexeur MAXIM MAX350).



Un multiplexeur double à 4 canaux accroît le nombre de canaux et d'entrées de sélection pouvant être utilisés par ce microcontrôleur.

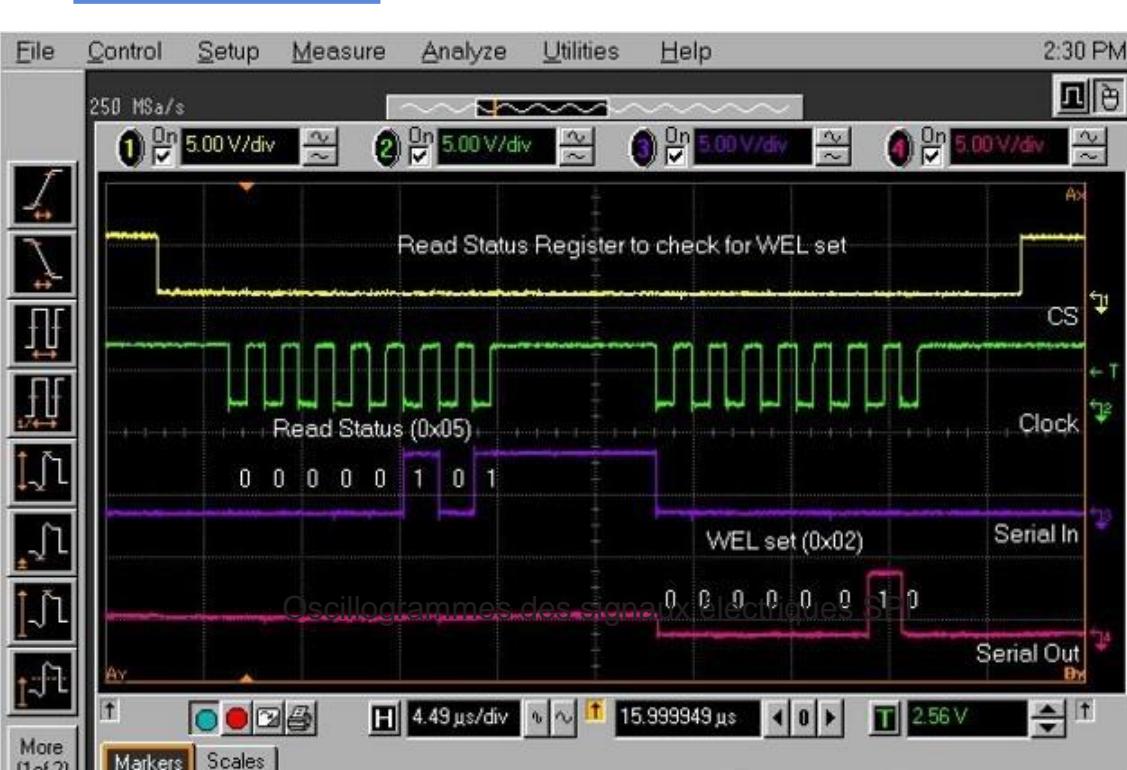
Si on utilise des composants de tension d'alimentation différente par exemple un µp en 5v et un capteur en 3.3v, il convient de procéder à une adaptation du niveau de tension des broches MOSI, SS, SCLK (la MISO n'est pas affectée car compatible avec le µp) par diviseur de tension à base de résistance $\frac{1}{4}$ de watt.



Oscillogrammes des signaux électriques SPI

4. Chronogramme

Toutes les liaisons SPI respectent le chronogramme général ci-dessous, mais quatre variantes existent au niveau de la prise en compte des données qui sont échangées.



FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

2.8 Les différents autres noms utilisés avec un bus SPI

SCK :

Il s'agit de l'horloge de synchronisation des échanges, elle est générée par le maître et est commune à tous les éléments du bus. Cela a pour avantage d'éviter que chaque composant possède son propre quartz avec les problèmes de précision que cela impose.

SDI, DI, SI, SDO, DO, SO :

Il s'agit de la patte de Serial Data In; dans ce cas de convention d'écriture il convient de relier la SDI du maître à la SDO du ou des esclaves.

Il s'agit de la patte de Serial Data Out; dans ce cas de convention d'écriture il convient de relier la SDO du maître à la SDI du ou des esclaves.

nCS, CS ,nSS, STE :

Il s'agit de la patte /SS Slave Select générée par le maître.

QSPI Queued Serial Peripheral Interface :

C'est un type de contrôleur dédié et spécifique qui permet certains transferts de données sans l'utilisation du processeur, mais en faisant appel à des pointeurs programmables, pointant une file d'attente de données.

Ce n'est pas un autre type de bus série.

FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

2.9 Avantages et Inconvénients du bus SPI

Avantages	Inconvénients
Communication en Full Duplex	Pas d'adressage possible
"Indépendant" du nombre de bits à transmettre	Utilisation sur très courte distance (même carte)
Pas de collision possible	Nécessite plus de fils que I ² C
Les esclaves utilisent l'horloge du maître pas de problème de précision de quartz	Pas d'acquittement (le maître ne sait pas s'il est écouté)
Beaucoup plus rapide que I ² C en mode standard	
Possibilité de configuration à plusieurs maîtres	

FISA S3E A4 Réseaux de TERRAINS/BUS SPI /

2.10 Constitution pour un système à microcontrôleur PIC

10.1 Le MSSP Control Register 1 (SSPCON1- 18F87K22 SSPxCON1)

bit 7 WCOL	: Write Collision Detect bit (Transmit mode only) 1 = Le registre SSPBUF register est écrit pendant que l'octet prévu est transmis (Doit être effacé par le programme) 0 = Pas de détection
bit 6 SSPOV	: Receive Overflow Indicator bit(1) SPI mode esclave: 1 = un nouvel octet est reçu pendant que le registre SSPBUF est encore en possession de l'ancien. En cas d'"overflow"(débordement) la donnée du SSPSR est perdue. Le débordement n'intervient que dans le mode Esclave. L'utilisateur doit lire le SSPBUF, seulement en cas de transmission de donnée, pour éviter le maintien de l'overflow il doit être effacer dans le soft. 0 =Pas de débotrdement
bit 5 SSPEN	: Master Synchronous Serial Port Enable bit 1 = validation de la configuration SCK, SDO, SDI and SS comme port série(2) 0 = pas de validation du mode SPI, les broches sont en E/S normales(2)
bit 4 CKP	: Clock Polarity Select bit 1 = horloge au niveau haut logique au repos 0 = horloge au niveau bas logique au repos
bit 3-0 SSPM3:SSPM0	: Master Synchronous Serial Port Mode Select bits 0101 = SPI mode Esclave, clock = SCK, SS non validée, SS peut être utilisée en E/S(3) 0100 = SPI mode Esclave, clock = SCK , SS (select slave) validée(3) 0011 = SPI mode Maître, clock = TMR2 output/2(3) 0010 = SPI mode Maître , clock = FOSC/64(3) 0001 = SPI mode Maître , clock = FOSC/16(3) 0000 = SPI mode Maître , clock = FOSC/4(3)

REGISTER 19-2: SSPCON1: MSSP CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV ⁽¹⁾	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

Note 1: Dans le mode Maître , le bit de dépassement n'est pas mis à 1 à chaque transmission ou nouvelle réception mais est initialisé par une écriture dans le registre SSPBUF.

2: Si validé, cette broche doit être configuré comme une entrée ou une sortie.

3: Les combinaisons non listées ici sont réservées ou implémentées pour l'I2C. CESI FISA S3E A4 BUS DE TERRAIN NAI 2020

10.2 Le MSSP Status Register (SSPSTAT 18F87K22 SSPxSTAT)

bit 7 SMP	: Sample bit Dans le mode Maître 1 = Echantillonnage de la donnée entrante à la fin du temps de sortie de la données sortante 0 = Echantillonnage de la donnée entrante au milieu du temps de sortie de la données sortante Dans le mode Esclave mis à 0
bit 6 CKE	: SPI Clock Select bit 1 = Transmission sur front de l'état actif vers état repos de l'horloge 0 = Transmission sur front de l'état repos vers état actif de l'horloge Dans le mode Esclave mis à 0
bit 0 BF	: Buffer Full Status bit (Receive mode only) 1 = Réception complète, SSPBUF plein 0 = Réception incomplète , SSPBUF vide

REGISTER 19-1: SSPSTAT: MSSP STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE ⁽¹⁾	D/A	P	S	R/W	UA	BF
bit 7							

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
-n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unknown

Les bits 1, 2, 3, 4 et 5 ne sont pas utilisés en mode SPI.

10.3 Le Serial Receive/Transmit Buffer Register SSPBUF

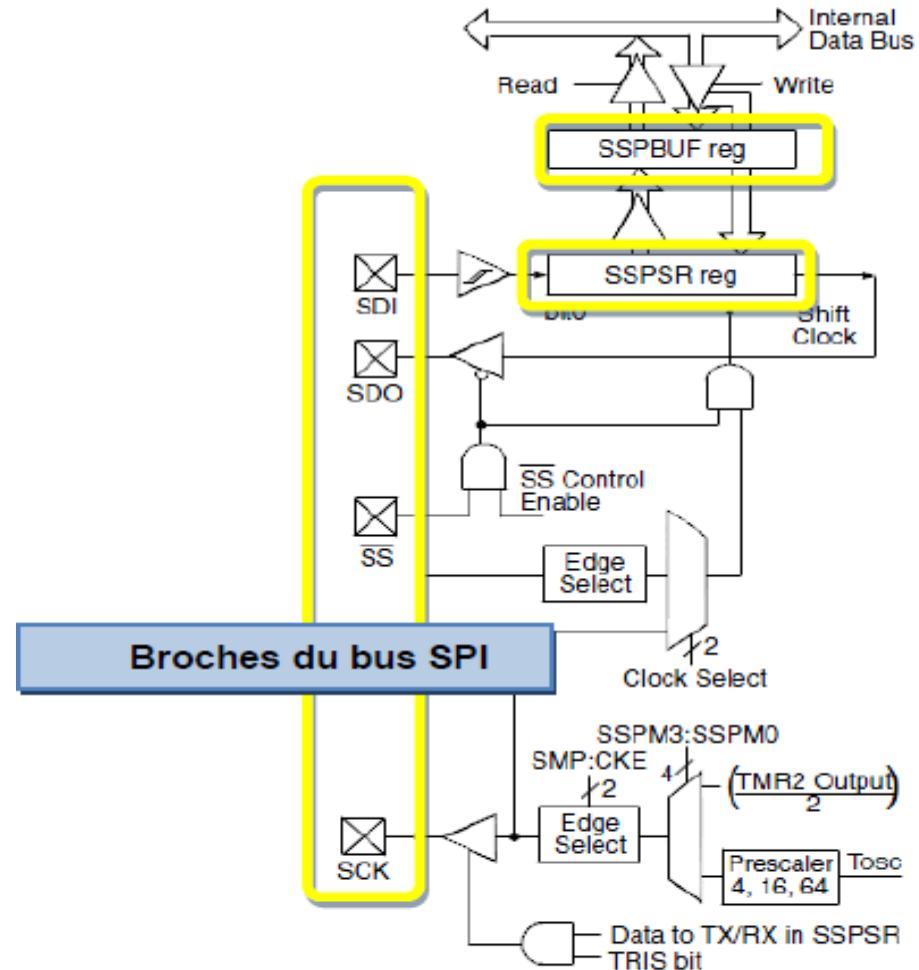
Ce registre n'est pas accessible en direct !

Ce registre est utilisé pour recevoir / envoyer les données de / vers l'extérieur du PIC.

Pendant les opérations de réception, les 2 registres SSPSR et SSPBUF créent un double tampon receveur.

Lorsque le SSPSR a reçu un octet, il le transfère au SSPBUF en provoquant la mise à 1 de SSPIF.

Pendant la transmission, SSPBUF n'est pas en double tampon. Une écriture dans SSPBUF va écrire aussi dans SSPSR.



Note: Only those pin functions relevant to SPI operation are shown here.

10.3 Cas du 18F87K22 (x prend la valeur 1 ou 2)

REGISTER 21-1: SSPxSTAT: MSSPx STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE ⁽¹⁾	D/A	P	S	R/W	UA	BF
bit 7				bit 0			

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
 -n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unknown

bit 7 **SMP:**
 Sample bit
SPI Master mode:
 1 = Input data is sampled at the end of data output time
 0 = Input data is sampled at the middle of data output time
SPI Slave mode:
 SMP must be cleared when SPI is used in Slave mode.

bit 6 **CKE:** SPI Clock Select bit⁽¹⁾
 1 = Transmit occurs on the transition from active to Idle clock state
 0 = Transmit occurs on the transition from Idle to active clock

state bit 5 **D/A:** Data/Address bit
 Used in I²C™ mode only.

bit 4 **P:** Stop bit
 Used in I²C mode only. This bit is cleared when the MSSPx module is disabled; SSPEN is cleared.

bit 3 **S:** Start bit
 Used in I²C mode only.

bit 2 **R/W:** Read/Write
 Information bit Used in I²C mode only.

bit 1 **UA:** Update
 Address bit Used in I²C mode only.

bit 0 **BF:** Buffer Full Status bit (Receive mode only)
 1 = Receive is complete, SSPxBUF is full
 0 = Receive is not complete, SSPxBUF is empty

Note 1: Polarity of clock state is set by the CKP bit (SSPxCON1<4>).

REGISTER 21-2: SSPxCON1: MSSPx CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV ⁽¹⁾	SSPEN ⁽²⁾	CKP	SSPM3 ⁽³⁾	SSPM2 ⁽³⁾	SSPM1 ⁽³⁾	SSPM0 ⁽³⁾
bit 7				bit 0			

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
 -n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unknown

bit 7 **WCOL:** Write Collision Detect bit
 1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
 0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator

bit⁽¹⁾ **SPI Slave mode:**
 1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. The user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).
 0 = No overflow

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit⁽²⁾

1 = Enables serial port and configures SCKx, SDIx, SDIx and SSx as serial port pins
 0 = Disables serial port and configures these pins as I/O port

pins bit 4 **CKP:** Clock Polarity Select bit

1 = Idle state for the clock is a high level
 0 = Idle state for the clock is a low level

bit 3-0 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits⁽³⁾

1010 = SPI Master mode: clock = Fosc/8
 0101 = SPI Slave mode: clock = SCKx pin; SSx pin control disabled; SSx can be used as I/O pin
 0100 = SPI Slave mode: clock = SCKx pin; SSx pin control enabled
 0011 = SPI Master mode: clock = TMR2 output/2
 0010 = SPI Master mode: clock = Fosc/64 0001 = SPI Master mode:
 clock = Fosc/16 0000 = SPI Master mode: clock = Fosc/4

Note 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.

2: When enabled, these pins must be properly configured as inputs or outputs.

3: Bit combinations not specifically listed here are either reserved or implemented in I²C™ mode only.

FISA S3E A4 Réseaux de TERRAINS/ BUS SPI

2.11 Compléments SD carda

Les cartes SD possèdent plusieurs interfaçages possibles dont un bus SPI. Le schéma ci-dessous présente la connectique des cartes SD, on y retrouve les broches spécifiques du bus SPI.

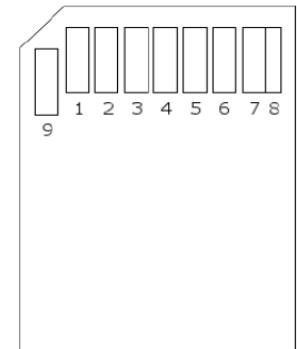
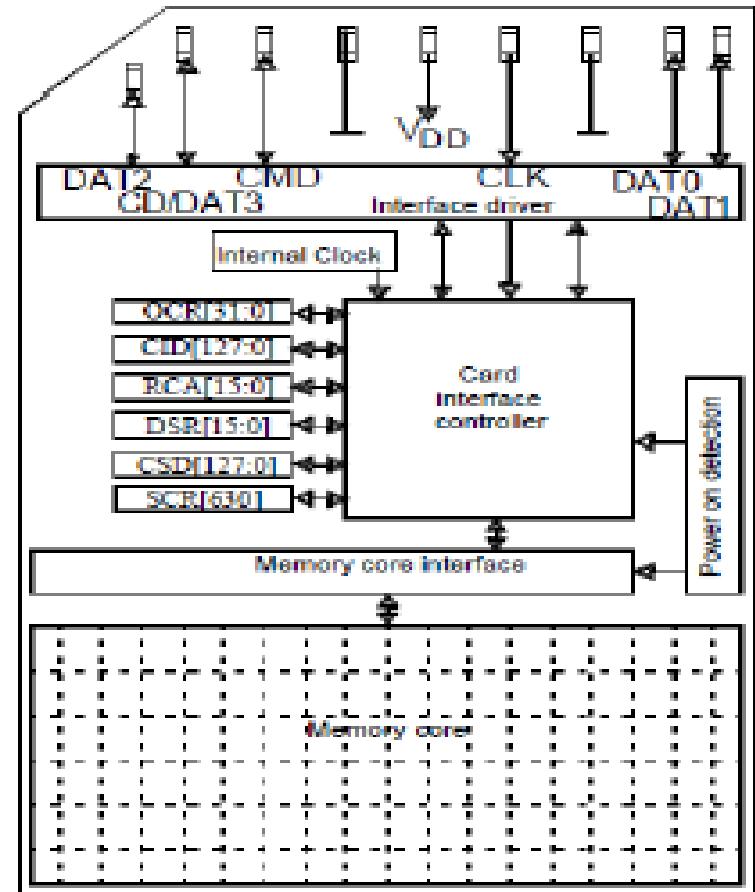
Avec un microcontrôleur incluant une interface SPI spécifique et une programmation pour les routines supplémentaires d'accès à la mémoire de la carte, on peut développer des applications sur ce type de carte (voir les liens proposés). Cependant, le protocole utilisé est complexe et, si l'on souhaite avoir une structure de fichier utilisable sur un ordinateur, on est confronté au problème de la FAT...

Pour plus d'infos :

[SPI and SD cards](#)

www.dejazzer.com/ee379/lecture_notes/lec12_sd_card.pdf

fr.wikihow.com/formater-une-carte-SD



Pin	Fonction SPI
1	Chip select
2	MOSI
3	GND
4	VCC
5	SCLK
6	GND
7	MISO
8	SD (SD only)
9	SD (SD only)

Bibliographie et sources

Internet : documents constructeurs motorola, microchips, Maxim.

Cours et diaporama sur les liaisons séries de T.Berenguer, P Monassier, B Nourry.

2.12 Travaux pratiques

Travaux préliminaires

Téléchargement de MPLABX directement sur le site Microchip :

<http://www.microchip.com/mplabx-ide-windows-installer>

Téléchargement du compilateur XC8

<http://www.microchip.com/mplabxc8windows>

Lancer le programme MPLAB X IDE

Créer un nouveau projet

File > New project

Choisir la catégorie Microchip Embedded puis Standalone project > Net

Choisir la famille Advanced 8-bit (PIC18) puis PIC18F87K22 > Net

Dans Hardware Tool, sélectionner PicKit 3 > Net

Dans Compiler toolchain, sélectionner XC8 > Net

Renseigner le nom du projet (ex : TP1_SPI)

Choisir l'emplacement du projet

Cocher "Set as main project" > Finish

Dans le projet, cliquer droit sur Source files > New > C main file
Entrer le nom > Finish

Dans le projet, cliquer droit sur Header files > Add existing item
Sélectionner le fichier xc.h dans <C:\program files\Microchip\xc8\include>
⇒ Non nécessaire si main.c inclut déjà #include <xc.h>

Copier les fichiers .c et .h fournis dans le dossier du projet.

Ajouter les .h dans Header du projet (cliquer droit sur Header files > Add existing item)
Ajouter les .c dans Sources du projet (cliquer droit sur Soures files > Add existing item)

Attention si vous obtenez l'erreur target device was not found lors de la programmation du pic vérifiez l'option d'alimentation du pickit :

Run > Set project configuration > Customize

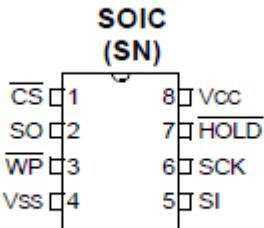
Cliquer sur Pickit à gauche dans Categories
Puis dans option categories sélectionner Power
Cocher power target from pickit et choisissez 5V



1K-4K SPI Serial EEPROM High Temp Family Data Sheet

25LC010A
25LC020A
25LC040A

25LC010A



Mise en œuvre du bus SPI

Objectifs du TP :

Découvrir l'environnement de développement.

Se familiariser avec l'interface graphique et le source afficheur.c fourni.

Lire et écrire dans un circuit mémoire SPI

Matériel utilisé :

- Malette Modèle : MDET Bus & Réseaux Kits de formation pour l'apprentissage des bus de communication
- Un PC équipé des logiciels MPLABX et son compilateur XC8.
- Un oscilloscope numérique,
- Un analyseur logique .
- Documentation des logiciels et du microcontrôleur.

Expérimentation n°1:

Saisir une chaîne de caractère au clavier et l'afficher en ligne 1 colonne 1.

Tracer un trait vertical en précisant la ligne , colonne, longueur.

Expérimentation n°2:

Saisir une chaîne de 16 caractères au clavier et l'afficher en ligne 1 colonne 1 .

Sauvegarder en mémoire SPI adresse 0.

Dumper la mémoire SPI (le 20 premiers octets) et démontrez la sauvegarde des données.



3

FISA S3E
Réseau de terrain

Bus I2C

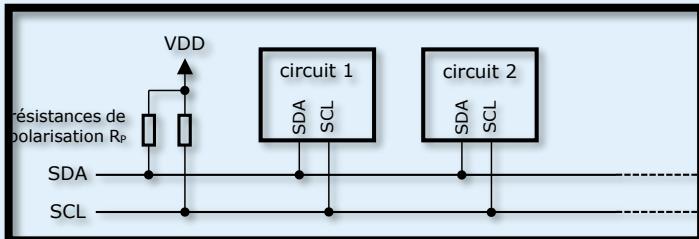
FISA S3E A4 Réseaux de TERRAINS/BUS I²C / 3 Résumé

« Inter Integrated Circuit » est un bus série développé dans les années 1980 par la société Philips pour faciliter la communication entre circuits intégrés tout en réduisant leur coût. Les derniers développements de ce bus lui confèrent un mode de transfert à grande vitesse à 3,4 Mbits/s.

1 DESCRIPTION MATERIELLE

C'est un bus série synchrone (l'horloge est transmise) constitué de 2 fils :

- la ligne de transmission de donnée série SDA (serial data)
- la ligne de transmission d'horloge SCL (serial clock)

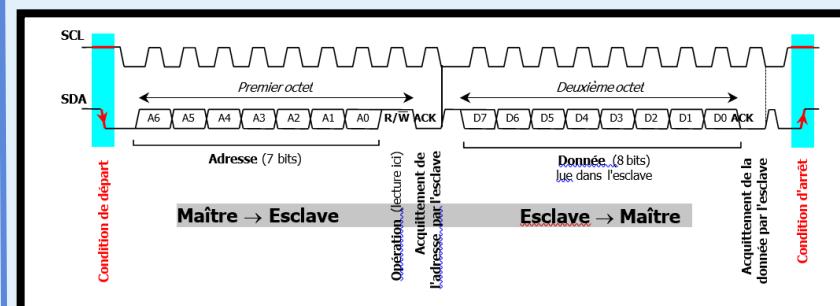


Plusieurs circuits peuvent être placés sur le bus sans risque de conflit grâce aux sorties à drain ou collecteur ouvert. En veille, chaque broche est en haute impédance : SDA et SCL sont amenées au niveau haut par 2 résistances RP reliées potentiel VDD.

2 – Protocole de communication

Il s'agit des règles afin que le dialogue soit possible, voir chronogrammes. Il se compose de plusieurs étapes :

1. la prise de parole par un maître si le bus est libre
2. la condition de départ émise par le maître
3. l'adresse de l'esclave destinataire
4. l'opération : lecture ou écriture de l'esclave
5. l'acquittement de l'adresse par l'esclave
6. l'octet transmis par le maître ou par l'esclave selon l'opération
7. l'acquittement de la transmission de l'octet
8. la condition d'arrêt générée par le maître terminant l'échange



3- Cas du 18F87K22

REGISTER 21-4: SSPxCON1: MSSPx CONTROL REGISTER 1 (I²C™ MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN ⁽¹⁾	CKP	SSPM3 ⁽²⁾	SSPM2 ⁽²⁾	SSPM1 ⁽²⁾	SSPM0 ⁽²⁾

00 en mode maître

REGISTER 21-4: SSPxCON1: MSSPx CONTROL REGISTER 1 (I ² C™ MODE)							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
bit 7							

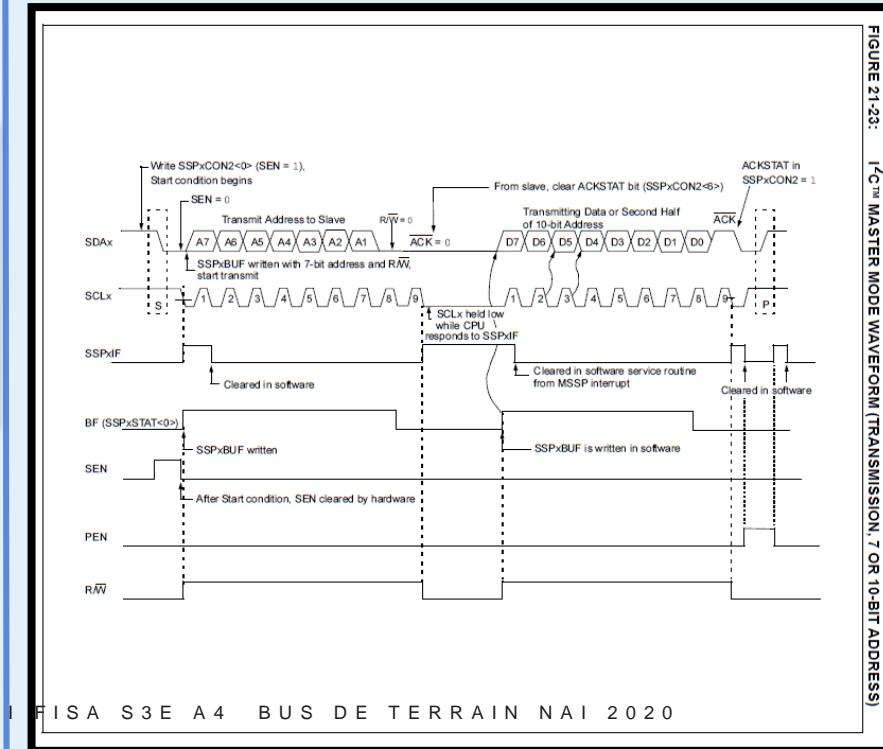
Registre de contrôle

REGISTER 21-5: SSPxCON2: MSSPx CONTROL REGISTER 2 (I²C™ MASTER MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT ⁽¹⁾	ACKEN ⁽²⁾	RCEN ⁽²⁾	PEN ⁽²⁾	RSEN ⁽²⁾	SEN ⁽²⁾
bit 7							

Gestion des échanges

SSPxADD baud rate



Sommaire



3 FISA S3E A4 Réseaux de TERRAINS/BUS I2C

• 3.1 Historique	40
• 3.2 Terminologie	41
• 3.3 Périphériques I ² C en bus	42
• 3.4 Adressage	44
• 3.5 Codage	45
• 3.6 Accès au bus	46
• 3.7 Registres du PIC 18F87K22	52
• 3.8 Cas du PCF 8591	54
• 3.9 Travaux pratiques	58

FISA S3E A4 Réseaux de TERRAINS/BUS I2C/

3.1 Historique



- But : faire communiquer à haute vitesse des composants électroniques de fonctions diverses (potentiomètres, RTC, mémoires, etc.) et d'origines diverses (Philips, Analog Device, Maxim, etc.) à l'aide d'un protocole standardisé simple et en limitant le câblage.
- Réalisation : au début des années 80 Philips met au point les composants qui communiquent par une ligne de données SDA et une ligne d'horloge SCL. En 1982 la norme "Inter Integrated Circuit" est déposée. Aujourd'hui la division semi-conducteurs de Philips est NXP qui a publié la version 4.0 en 2012.

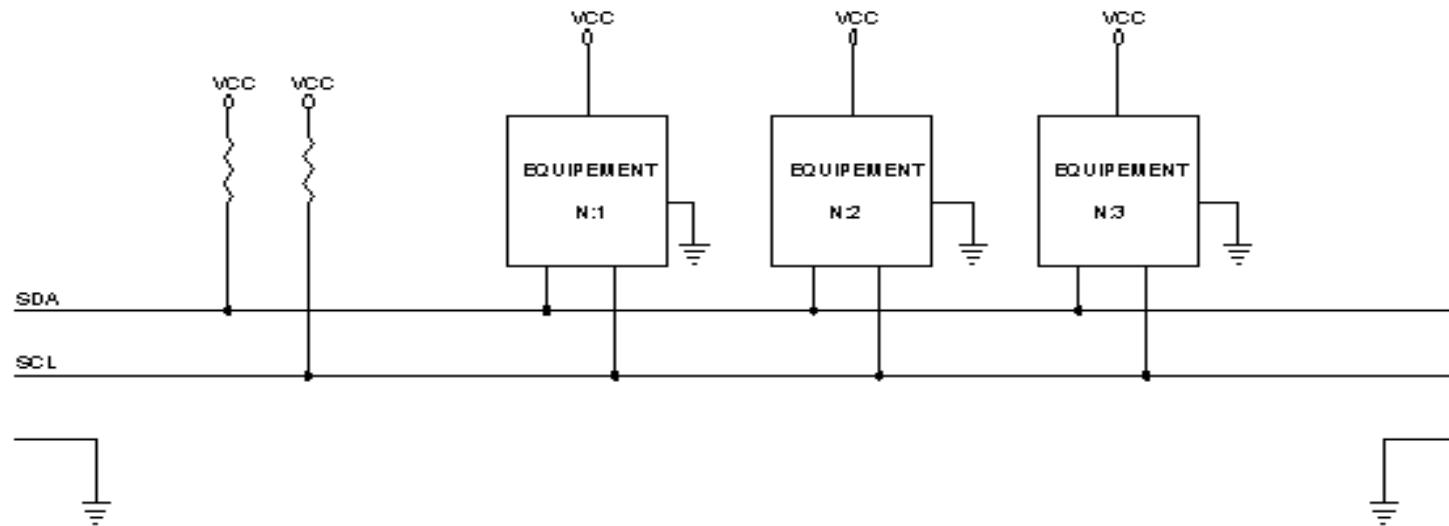
Terminologie

- SDA : Serial Data = ligne de transmission des données**
- SCL : Serial Clock = ligne du signal d'horloge**
- Émetteur : composant qui envoie des données sur le bus**
- Récepteur : composant qui reçoit les données présentes sur le bus**
- Maître : composant qui prend le contrôle du bus. Il génère l'horloge. Le maître peut être émetteur ou récepteur.**
- Esclave : composant adressé par le maître. L'esclave peut être émetteur ou récepteur.**
- Multi-maître : plusieurs maîtres peuvent tenter de contrôler le bus**
- Arbitrage : procédure permettant de choisir un seul maître alors que plusieurs tentent de prendre le contrôle du bus**
- Synchronisation : synchronisation des horloges de plusieurs maîtres**

FISA S3E A4 Réseaux de TERRAINS/BUS I²C/

3.3 Périphériques I²C en bus

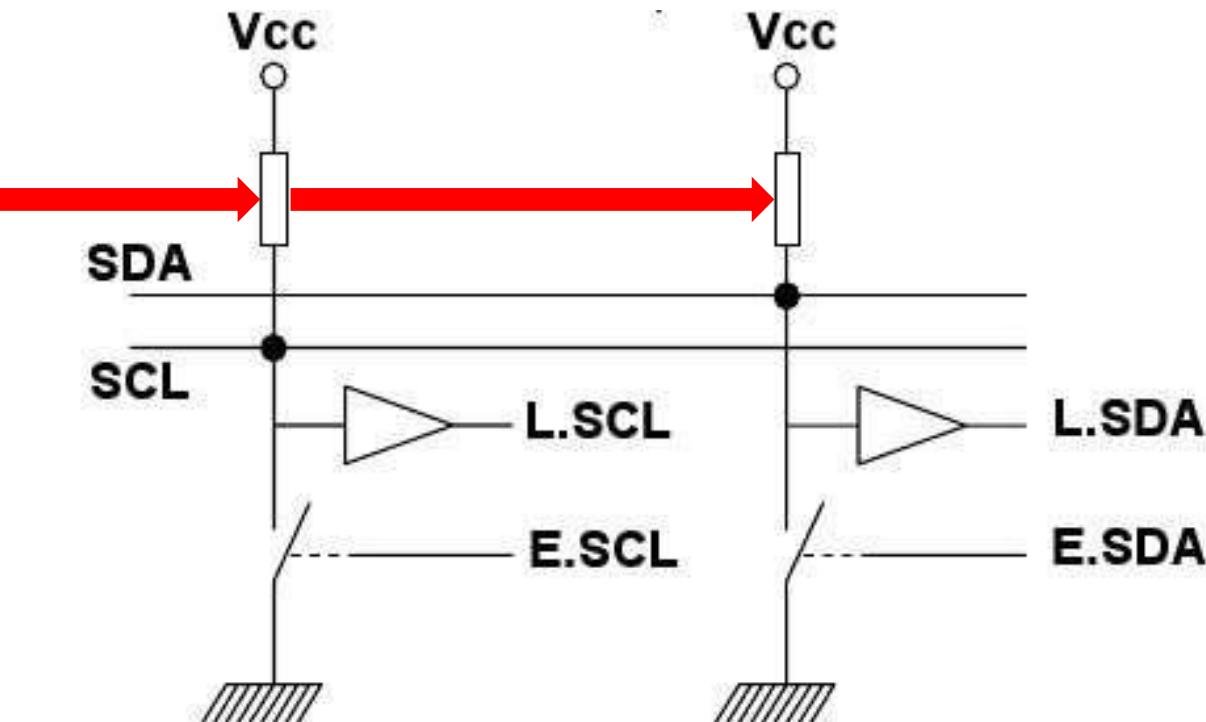
Les données sont envoyées en série sur la ligne SDA. La ligne SCL est l'horloge. Chaque équipement dispose d'une adresse unique.



E/S de type collecteur ou drain ouvert

ATTENTION :

prévoir des résistances
de pull-up (tirage) entre
Vcc et SDA et Vcc et SDL.
Généralement 4.7 kΩ



Avantages :

- Architecture ultra simple
- Les lignes SDA et SCL sont au repos à l'état haut. Elles peuvent être forcées uniquement à l'état bas.
- Pas de conflit électrique

FISA S3E A4 Réseaux de TERRAINS/BUS I2C/ 3.4 Adressage

Adressage

- Les adresses sont codées sur 7 bits
- Les adresses sont attribuées par le "I²C Committee" en fonction du type de périphérique
- Le 8ième bit indique que l'opération suivante sera une écriture (0) ou une lecture (1)**
- Lors d'une lecture le maître connaît le nombre d'octets qu'il attend

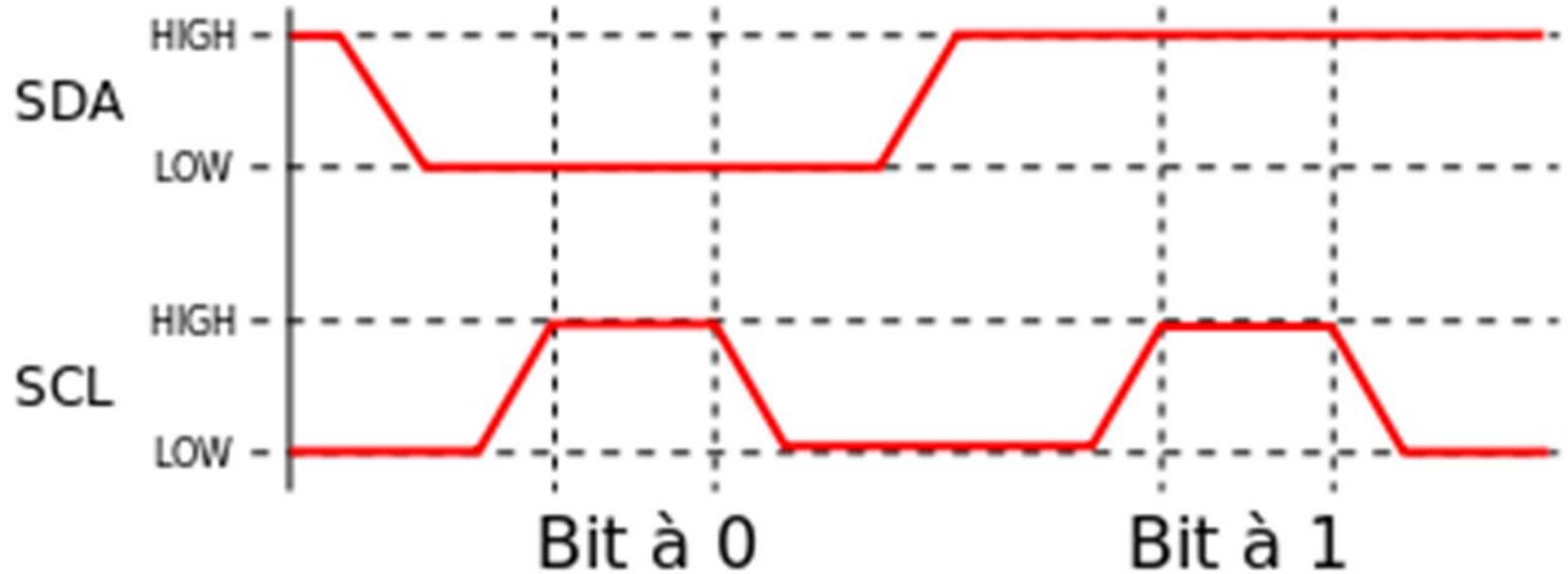
Adresses réservées : les adresses 0000 0xxx ne sont jamais attribuées à des composants

- 0000 0000 : appel général. Tous les périphériques renvoient un acquittement*
- 0000 0110 : reset. Tous les périphériques retournent à leur état initial*
- 0000 0010 : tous les périphériques reprennent leur adresse initiale*
- 0000 0100 : comme ci-dessus mais pour les circuit dont on peut définir l'adresse matériellement*

FISA S3E A4 Réseaux de TERRAINS/BUS I2C/ 3.5 Codage

Codage de bits : NRZ MSB first. Le niveau (« HIGH » ou « LOW ») de la ligne SDA doit être maintenu stable pendant le niveau « HIGH » sur la ligne SCL pour la lecture du bit.

Cette règle pourra être outrepassée pour des bits de contrôle.



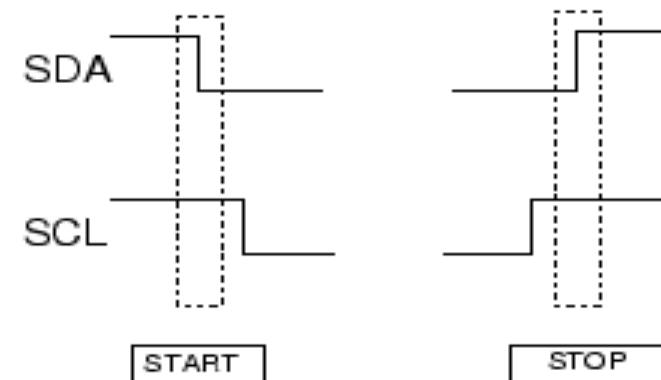
FISA S3E A4 Réseaux de TERRAINS/BUS I2C/

3.6 Accès au bus

Accès au bus

- Les lignes SDA et SCL sont à 1 lorsque le bus est disponible (idle)
- Le bus doit être "idle" depuis plus de 4.7µs pour prétendre en prendre le contrôle
- Pour prendre le contrôle du bus SDA passe à 0 et SCL reste à 1 (START)
- Le périphérique qui prend le contrôle du bus en devient le maître
- C'est le maître qui génère l'horloge (SCL)
- Le bus est libéré lorsque SDA passe à 1 et SCL reste à 1 (STOP)
- Les conditions de START et de STOP ne sont générées que par le maître

3.6.1 Conditions Start / Stop

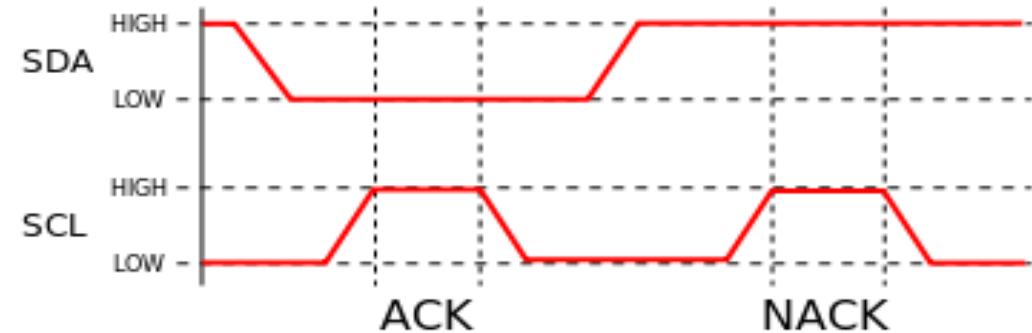


Connexion maître / esclave

Connexion du maître à l'esclave

- Le maître envoie les 7 bits d'adresses + le bit R/W soit 1 octets
- Après avoir reçu l'adresse correctement le récepteur accuse réception (Address-ACKnowledgment) en passant SDA à 0 pendant que SCL est à 1
- Le maître envoie ensuite un octet de données
- La bonne réception d'un octet de données est de même acquitté par un Data-ACKnowledgment.
- Si nécessaire un autre octet de données est envoyé puis acquitté
- Clôture de la connexion par une condition STOP

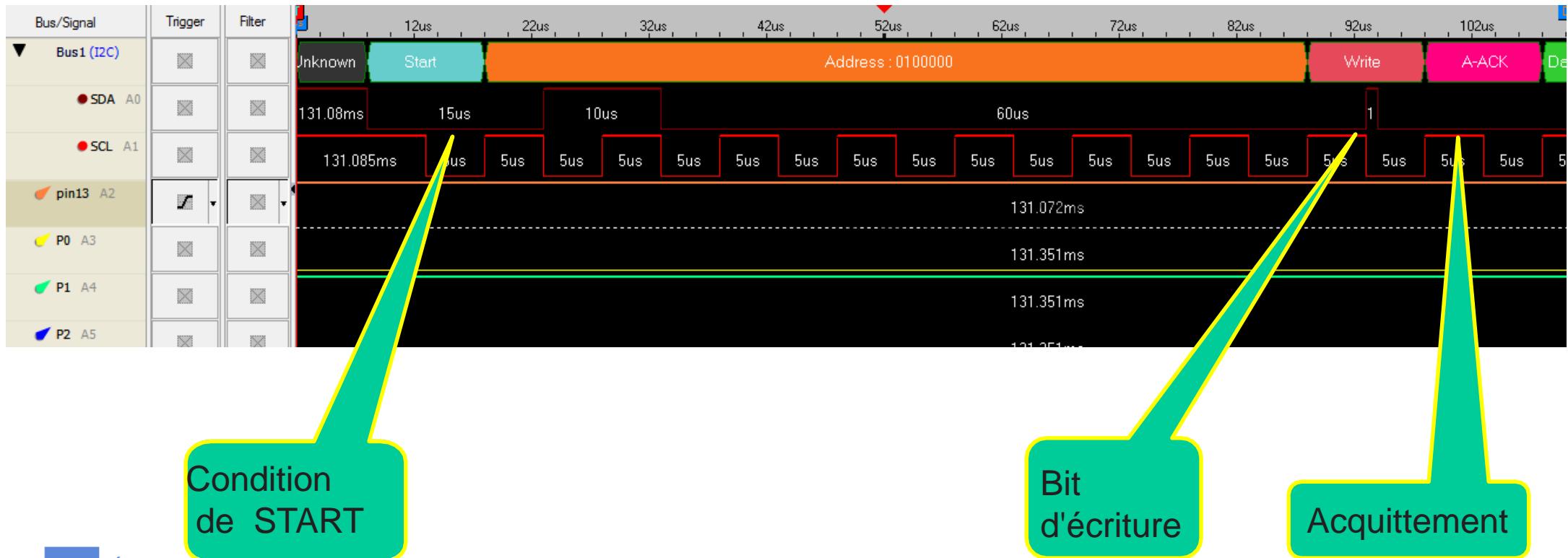
Acquittement



En cas d'erreur d'acquittement c'est une condition No-ACKnowledgment qui est envoyée

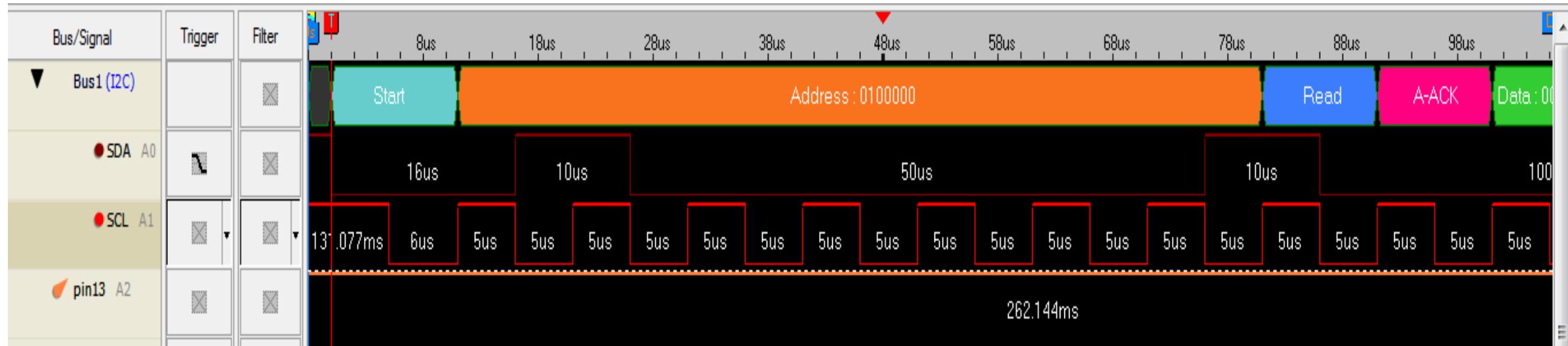
FISA S3E A4 Réseaux de TERRAINS/BUS I2C/

3.6.2 Prise de contrôle du bus en écriture



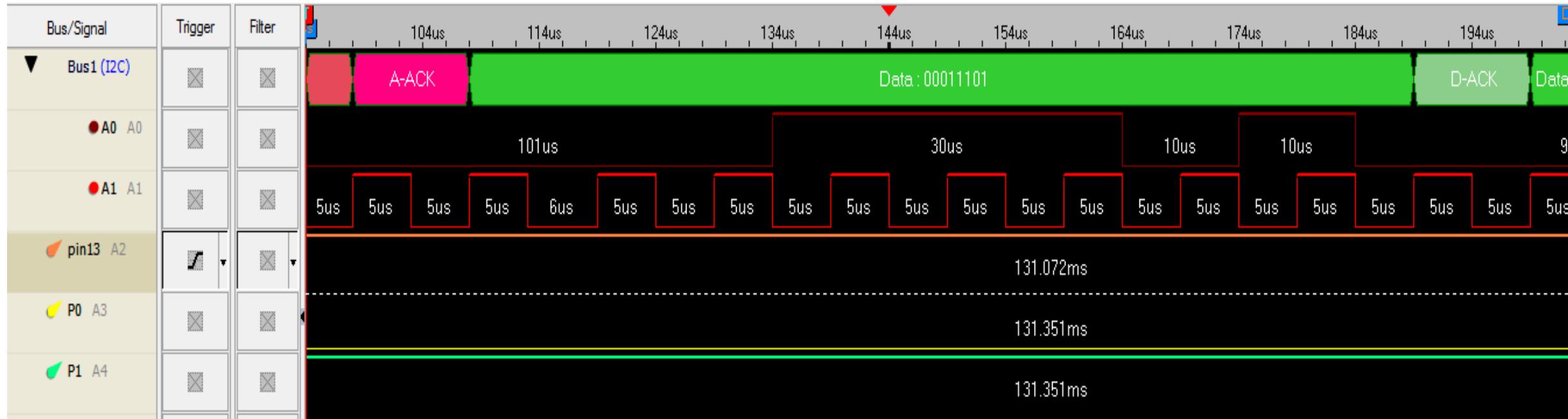
FISA S3E A4 Réseaux de TERRAINS/BUS I2C/

3.6.3 Prise de contrôle du bus en Lecture



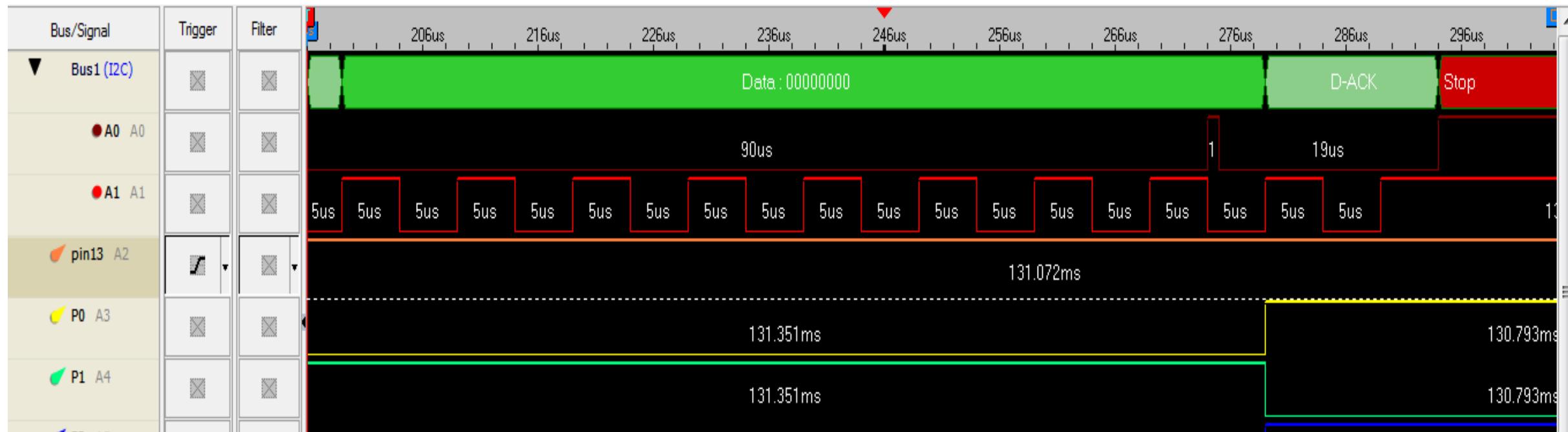
FISA S3E A4 Réseaux de TERRAINS/BUS I2C/

3.6.4 Transmission de données



FISA S3E A4 Réseaux de TERRAINS/BUS I2C/

3.6.5 Libération du bus



FISA S3E A4 Réseaux de TERRAINS/BUS I²C/

3.7 Registres 18F87K22

PIC18F87K22 FAMILY

REGISTER 21-3: SSPxSTAT: MSSPx STATUS REGISTER (I²C™ MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/A	P ⁽¹⁾	S ⁽¹⁾	R/W ^(2,3)	UA	BF
bit 7						bit 0	

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

bit 7	SMP : Slew Rate Control bit <i>In Master or Slave mode:</i> 1 = Slew rate control is disabled for Standard Speed mode (100 kHz and 1 MHz) 0 = Slew rate control is enabled for High-Speed mode (400 kHz)
bit 6	CKE : SMBus Select bit <i>In Master or Slave mode:</i> 1 = Enable SMBus-specific inputs 0 = Disable SMBus-specific inputs
bit 5	D/A : Data/Address bit <i>In Master mode:</i> Reserved. <i>In Slave mode:</i> 1 = Indicates that the last byte received or transmitted was data 0 = Indicates that the last byte received or transmitted was address
bit 4	P : Stop bit ⁽¹⁾ 1 = Indicates that a Stop bit has been detected last 0 = Stop bit was not detected last
bit 3	S : Start bit ⁽¹⁾ 1 = Indicates that a Start bit has been detected last 0 = Start bit was not detected last
bit 2	R/W : Read/Write Information bit ^(2,3) <i>In Slave mode:</i> 1 = Read 0 = Write <i>In Master mode:</i> 1 = Transmit is in progress 0 = Transmit is not in progress
bit 1	UA : Update Address bit (10-Bit Slave mode only) 1 = Indicates that the user needs to update the address in the SSPxADD register 0 = Address does not need to be updated
bit 0	BF : Buffer Full Status bit <i>In Transmit mode:</i> 1 = SSPxBUF is full 0 = SSPxBUF is empty <i>In Receive mode:</i> 1 = SSPxBUF is full (does not include the ACK and Stop bits) 0 = SSPxBUF is empty (does not include the ACK and Stop bits)

Note 1: This bit is cleared on Reset and when SSPEN is cleared.

- 2: This bit holds the R/W bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or not ACK bit.
- 3: ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSPx is in Active mode.

REGISTER 21-4: SSPxCON1: MSSPx CONTROL REGISTER 1 (I²C™ MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN ⁽¹⁾	CKP	SSPM3 ⁽²⁾	SSPM2 ⁽²⁾	SSPM1 ⁽²⁾	SSPM0 ⁽²⁾
bit 7	bit 0						

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared
		x = Bit is unknown

bit 7 **WCOL**: Write Collision Detect bit

In Master Transmit mode:

1 = A write to the SSPxBUF register was attempted while the I²C conditions were not valid for a transmission to be started (must be cleared in software)
0 = No collision

In Slave Transmit mode:

1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

In Receive mode (Master or Slave modes):

This is a "don't care" bit.
In Receive mode:

1 = A byte is received while the SSPxBUF register is still holding the previous byte (must be cleared in software)
0 = No overflow
In Transmit mode:

This is a "don't care" bit in Transmit mode.
SSPEN: Master Synchronous Serial Port Enable bit⁽¹⁾

1 = Enables the serial port and configures the SDAx and SCLx pins as the serial port pins
0 = Disables serial port and configures these pins as I/O port pins

CKP: SCKx Release Control bit

In Slave mode:
1 = Releases clock
0 = Holds clock low (clock stretch), used to ensure data setup time
In Master mode:
Unused in this mode.

SSPM<3:0>: Master Synchronous Serial Port Mode Select bits⁽²⁾

1111 = I²C Slave mode: 10-bit address with Start and Stop bit interrupts enabled
1110 = I²C Slave mode: 7-bit address with Start and Stop bit interrupts enabled

1011 = I²C Firmware Controlled Master mode (slave Idle)

1001 = Load SSPMSK register at SSPxADD SFR address^(3,4)

1000 = I²C Master mode: clock = FOSC/4 * (SSPxADD + 1)

0111 = I²C Slave mode: 10-bit address

0110 = I²C Slave mode: 7-bit address

Note 1: When enabled, the SDAx and SCLx pins must be configured as inputs.

2: Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.

3: When SSPM<3:0> = 1001, any reads or writes to the SSPxADD SFR address actually accesses the SSPxMSK register.

4: This mode is only available when 7-Bit Address Masking mode is selected (MSSPMASK Configuration bit is '1').

REGISTER 21-5: SSPxCON2: MSSPx CONTROL REGISTER 2 (I²C™ MASTER MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT ⁽¹⁾	ACKEN ⁽²⁾	RCEN ⁽²⁾	PEN ⁽²⁾	RSEN ⁽²⁾	SEN ⁽²⁾
bit 7							bit 0

Legend:

R = Readable bit W = Writable bit U = Unimplemented bit, read as '0'
 -n = Value at POR '1' = Bit is set '0' = Bit is cleared x = Bit is unknown

- bit 7 GCEN: General Call Enable bit
Unused in Master mode.
- bit 6 ACKSTAT: Acknowledge Status bit (Master Transmit mode only)
1 = Acknowledge was not received from slave
0 = Acknowledge was received from slave
- bit 5 ACKDT: Acknowledge Data bit (Master Receive mode only)⁽¹⁾
1 = Not Acknowledge
0 = Acknowledge
- bit 4 ACKEN: Acknowledge Sequence Enable bit⁽²⁾
1 = Initiates Acknowledge sequence on SDAx and SCLx pins and transmits ACKDT data bit.
Automatically cleared by hardware.
0 = Acknowledge sequence is Idle
- bit 3 RCEN: Receive Enable bit (Master Receive mode only)⁽²⁾
1 = Enables Receive mode for I²C™
0 = Receive is Idle
- bit 2 PEN: Stop Condition Enable bit⁽²⁾
1 = Initiates Stop condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Stop condition is Idle
- bit 1 RSEN: Repeated Start Condition Enable bit⁽²⁾
1 = Initiates Repeated Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Repeated Start condition is Idle
- bit 0 SEN: Start Condition Enable bit⁽²⁾
1 = Initiates Start condition on SDAx and SCLx pins. Automatically cleared by hardware.
0 = Start condition is Idle

Note 1: Value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.

2: If the I²C module is active, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

21.4.7 BAUD RATE

In I²C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPxADD register (Figure 21-19). When a write occurs to SSPxBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to 0 and stops until another reload has taken place. The BRG count is decremented twice per instruction cycle (FCY) on the Q2 and Q4 clocks. In I²C Master mode, the BRG is reloaded automatically.

Once the given operation is complete (i.e., transmission of the last data bit is followed by ACK), the internal clock will automatically stop counting and the SCLx pin will remain in its last state.

Table 21-3 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD. The SSPxADD BRG value of 0x00 is not supported.

FIGURE 21-19: BAUD RATE GENERATOR BLOCK DIAGRAM

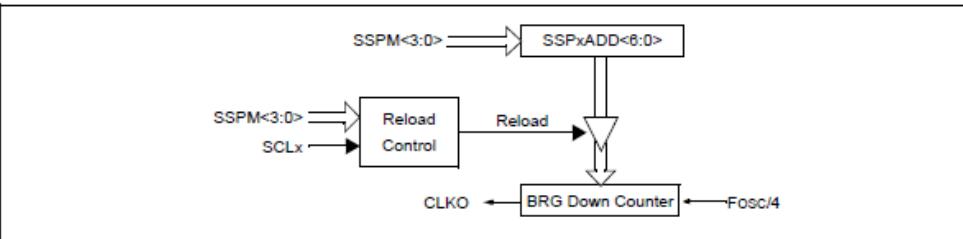


TABLE 21-3: I²C™ CLOCK RATE w/BRG

Fosc	FCY	FCY * 2	BRG Value	FSCL (2 Rollovers of BRG)
40 MHz	10 MHz	20 MHz	18h	400 kHz
40 MHz	10 MHz	20 MHz	1Fh	312.5 kHz
40 MHz	10 MHz	20 MHz	63h	100 kHz
16 MHz	4 MHz	8 MHz	09h	400 kHz
16 MHz	4 MHz	8 MHz	0Ch	308 kHz
16 MHz	4 MHz	8 MHz	27h	100 kHz
4 MHz	1 MHz	2 MHz	02h	333 kHz
4 MHz	1 MHz	2 MHz	09h	100 kHz
16 MHz	4 MHz	8 MHz	03h	1 MHz ⁽¹⁾

Note 1: A minimum of 16 MHz Fosc is required to get 1 MHz I²C.

21.4.7.1 Baud Rate and Module Interdependence

Because MSSP1 and MSSP2 are independent, they can operate simultaneously in I²C Master mode at different baud rates. This is done by using different BRG reload values for each module.

Because this mode derives its basic clock source from the system clock, any changes to the clock will affect both modules in the same proportion. It may be possible to change one or both baud rates back to a previous value by changing the BRG reload value.

FISA S3E A4 Réseaux de TERRAINS/BUS I2C/

3.8 Cas du PCF8591

Cas PCF8591

8-bit A/D and D/A converter (voir TP Microcontrôleur 1 : N°3 CAN-CAN)

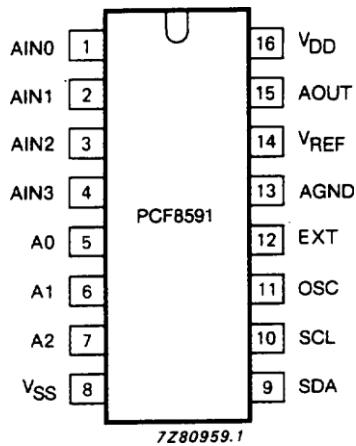
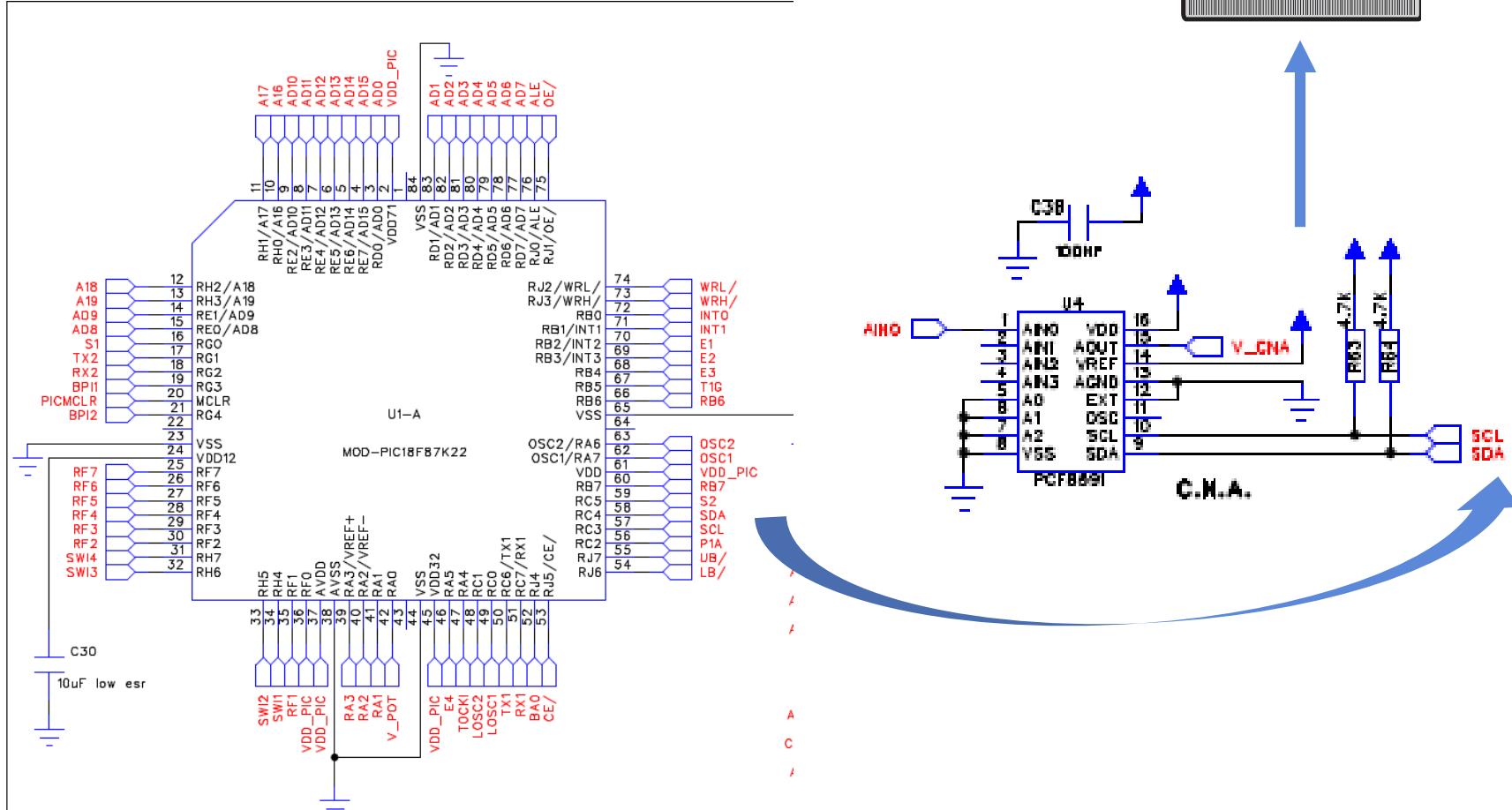


Fig.2 Pinning diagram.

SYMBOL	PIN	DESCRIPTION
AIN0	1	
AIN1	2	
AIN2	3	
AIN3	4	
A0	5	
A1	6	hardware address
A2	7	
VSS	8	
SDA	9	I2C-bus data input/output
SCL	10	I2C-bus clock input
OSC	11	oscillator input/output
EXT	12	external/internal switch for oscillator input
AGND	13	analog ground
VREF	14	voltage reference input
AOUT	15	analog output (D/A converter)
VDD	16	positive supply voltage

Environnement du TP3



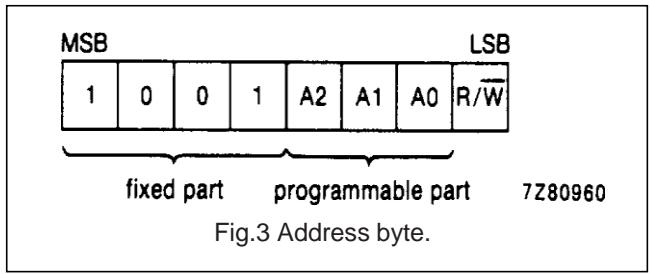
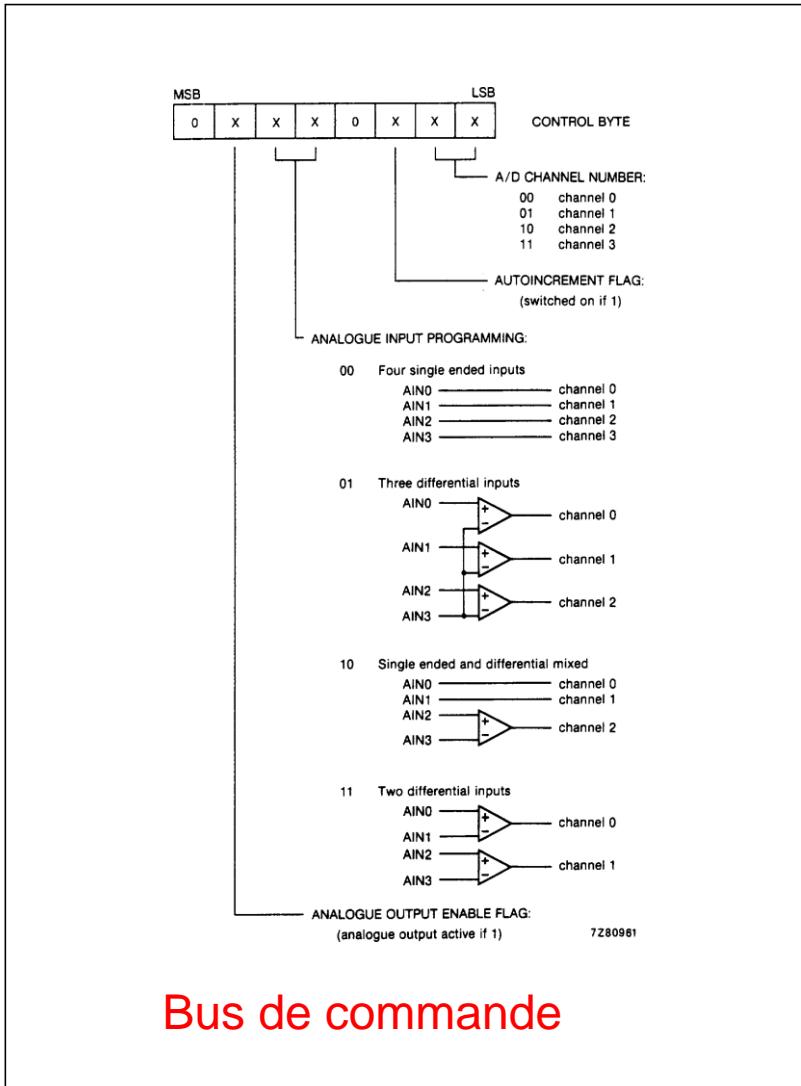
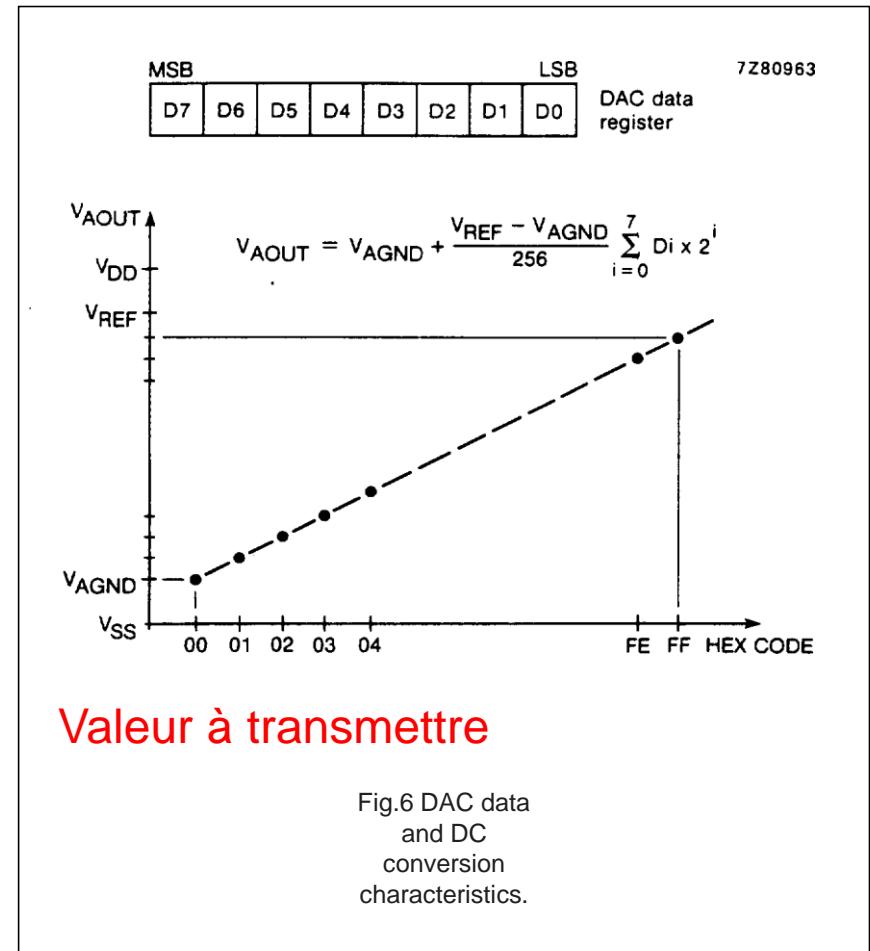


Fig.3 Address byte.

Adresse du composant



Bus de commande



Valeur à transmettre

Fig.6 DAC data
and DC
conversion
characteristics.

Programmation Initialisation

```
void I2C_Init(void)
{
    SSP1STAT = 0x00;
    SSP1CON1 = 0x00;
    SSP1CON1bits.SSPEN = 1; // Enables the serial port and configures the SDAx and SCLx pins as the serial port pins
    SSP1CON1bits.SSPM = 0x08; // 1000 = I2C Master mode
    SSP1CON2 = 0x00;
    SSP1ADD = 0x1A; // Baud rate pour I2C à 100kHz
}
```

Ecriture sur le bus I2C

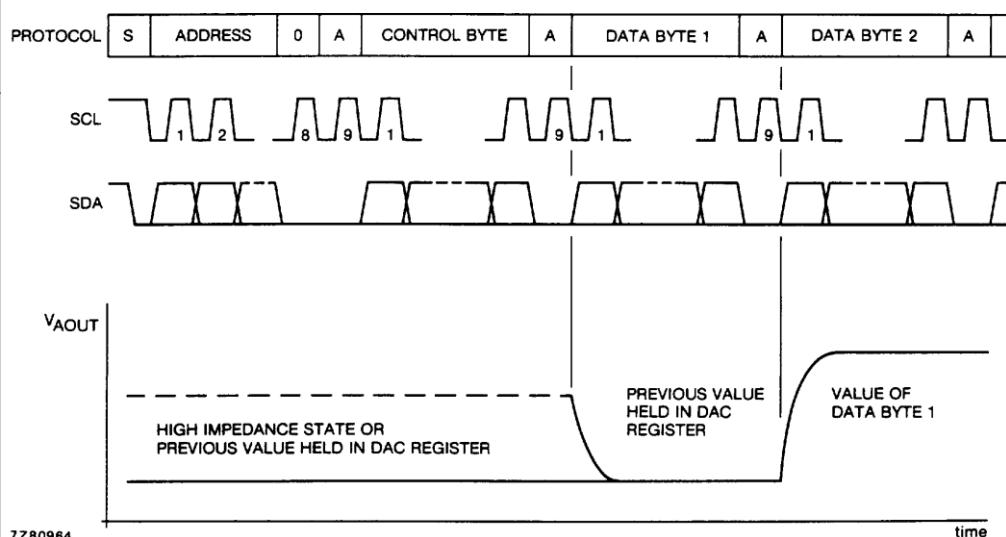
```
SSP1CON2bits.SEN = 1; // Envoi du bit de start
while(PIR1bits.SSP1IF == 0){}
PIR1bits.SSP1IF = 0;

SSP1BUF = 0x90; // adresse du convertisseur dig/ana
while(SSP1CON2bits.ACKSTAT == 1){}; // tant que acquittement du convertisseur non reçu
while(PIR1bits.SSP1IF == 0){}
PIR1bits.SSP1IF = 0;

SSP1BUF = 0x40; // bits de contrôle
while(SSP1CON2bits.ACKSTAT == 1){}; // tant que acquittement du convertisseur non reçu
while(PIR1bits.SSP1IF == 0){}
PIR1bits.SSP1IF = 0;

SSP1BUF = VAL; // bits de données 0 à 255
while(SSP1CON2bits.ACKSTAT == 1){}; // tant que acquittement du convertisseur non reçu
while(PIR1bits.SSP1IF == 0){}
PIR1bits.SSP1IF = 0;
```

```
SSP1CON2bits.PEN = 1; // Envoi du bit de stop
while(PIR1bits.SSP1IF == 0){}
PIR1bits.SSP1IF = 0;
```



Récapitulatif

Fig.7 D/A conversion sequence.

FISA S3E A4 Réseaux de TERRAINS/BUS I2C /

3.9 Travaux pratiques

Travaux préliminaires

Téléchargement de MPLABX directement sur le site Microchip :

<http://www.microchip.com/mplabx-ide-windows-installer>

Téléchargement du compilateur XC8

<http://www.microchip.com/mplabxc8windows>

Lancer le programme MPLAB X IDE

Créer un nouveau projet

File > New project

Choisir la catégorie Microchip Embedded puis Standalone project > Net

Choisir la famille Advanced 8-bit (PIC18) puis PIC18F87K22 > Net

Dans Hardware Tool, sélectionner PicKit 3 > Net

Dans Compiler toolchain, sélectionner XC8 > Net

Renseigner le nom du projet (ex : TP1_SPI)

Choisir l'emplacement du projet

Cocher "Set as main project" > Finish

Dans le projet, cliquer droit sur Source files > New > C main file
Entrer le nom > Finish

Dans le projet, cliquer droit sur Header files > Add existing item
Sélectionner le fichier xc.h dans <C:\program files\Microchip\xc8\include>
⇒ Non nécessaire si main.c inclut déjà #include <xc.h>

Copier les fichiers .c et .h fournis dans le dossier du projet.

Ajouter les .h dans Header du projet (cliquer droit sur Header files > Add existing item)
Ajouter les .c dans Sources du projet (cliquer droit sur Soures files > Add existing item)

Attention si vous obtenez l'erreur target device was not found lors de la programmation du pic vérifiez l'option d'alimentation du pickit :

Run > Set project configuration > Customize

Cliquer sur Pickit à gauche dans Categories
Puis dans option categories sélectionner Power
Cocher power target from pickit et choisissez 5V

TP N° 2 FISA S3E A4 Réseaux de TERRAINS/ BUS I2C

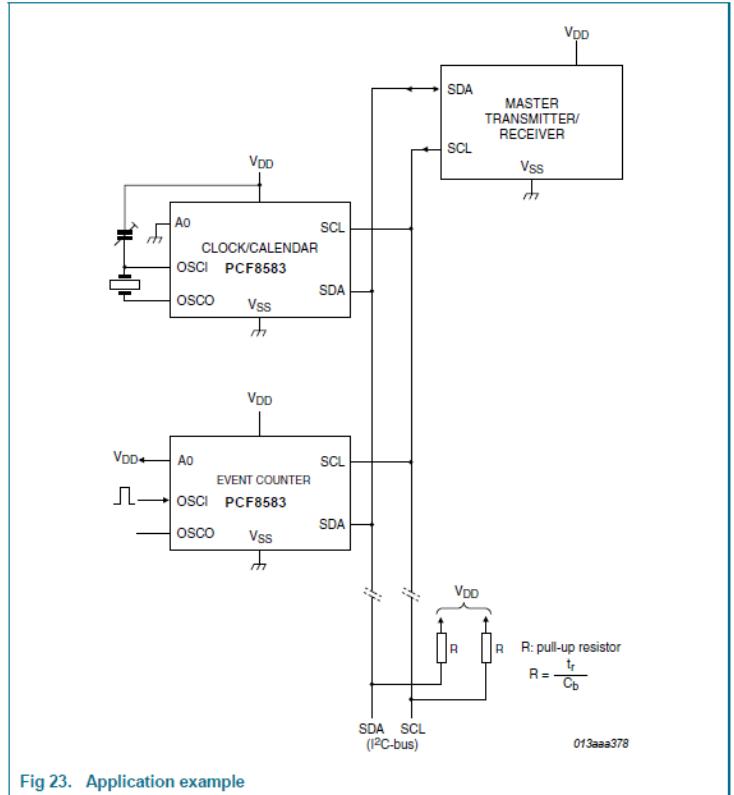


Fig 23. Application example

Mise en œuvre du bus I2C

Objectifs du TP :

Approfondir le bus I2C.

Se familiariser avec l'interface graphique et la source afficheur.c fourni.

Lire et écrire dans un circuit mémoire I2C.

Matériel utilisé :

- Malette Modèle : MDET Bus &Réseaux Kits de formation pour l'apprentissage des bus de communication
- Un PC équipé des logiciels MPLABX et son compilateur XC8.
- Un oscilloscope numérique,
- Un analyseur logique .
- Documentation des logiciels et du microcontrôleur.

Expérimentation n°1:

Configurer le composant PCF 8383 en Event Compteur.

Ecrire une application met à zéro le compteur à l'appui sur un bouton poussoir.

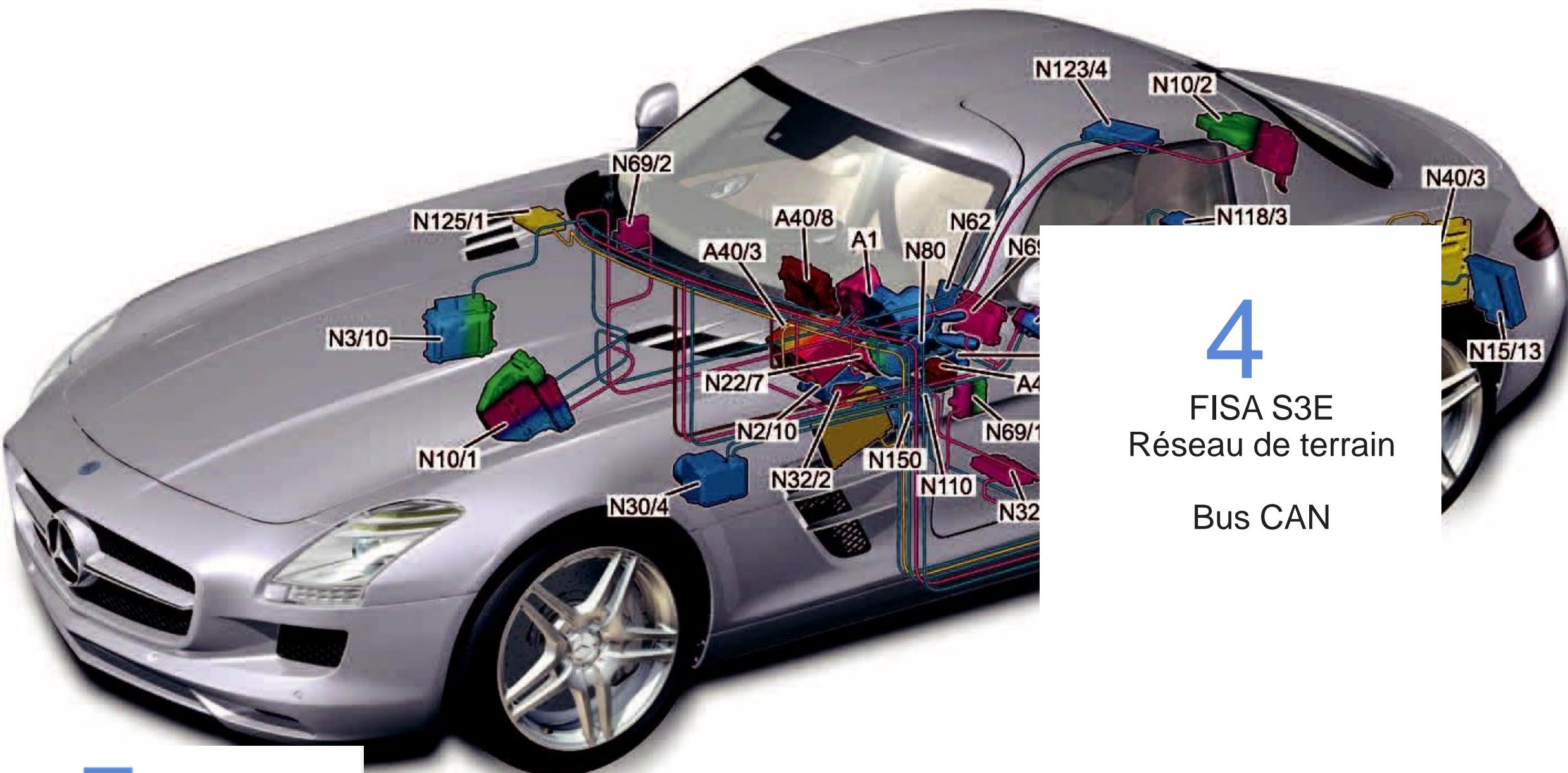
Faire tourner la turbine et afficher la valeur du compteur.

Expérimentation n°2: Dosage

Saisir au clavier une valeur pré-déterminée à charger dans le compteur.

Après remise à zéro logicielle, compter à l'aide de la turbine jusqu'à l'atteinte de la valeur pré-déterminée.

A l'atteinte de la valeur pré-déterminée, générer une interruption qui positionne un voyant en alarme et qui déclenche l'inscription « ALARME » à l'écran.



4

FISA S3E Réseau de terrain Bus CAN

FISA S3E A4 Réseaux de TERRAINS/BUS CAN / 4 Résumé

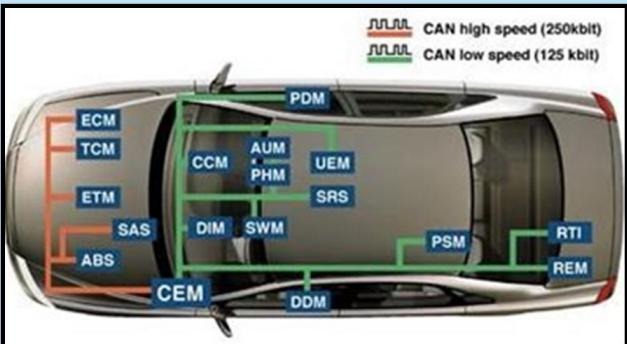
Il existe sous deux versions :

CAN2.0A : trame standard identificateur de 11 bits (CAN standard) ;

CAN2.0B trame plus longue avec identificateur sur 29 bits (CAN étendu).

Il existe également deux types différenciés par leur débit :

le CAN LOW Speed ; le CAN High Speed.



1- Principe de l'arbitrage

Le procédé d'attribution du bus est basé sur le principe de "**l'arbitrage bit à bit**", selon lequel les nœuds 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 écrase la valeur 1.

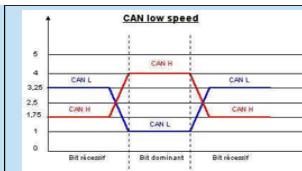
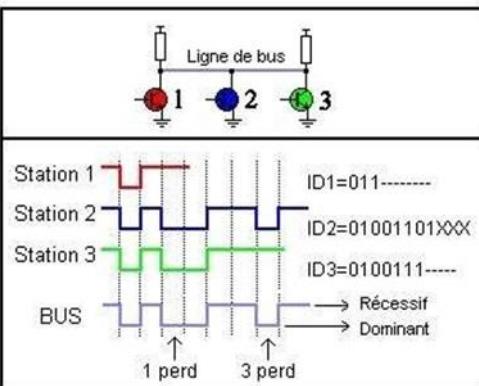
On appelle donc "état dominant" l'état logique 0, et "état récessif" l'état logique 1. Lors de l'arbitrage bit à bit, dès qu'une station émettrice se trouve en état récessif et détecte un état dominant, elle perd la compétition et arrête d'émettre. Tous les perdants deviennent automatiquement des récepteurs du message, et ne tentent à nouveau d'émettre que lorsque le bus se libère.

2 Les signaux du bus CAN

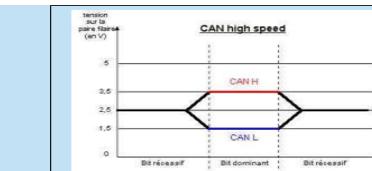
Paire filaire différentielle sur 2 fils ,les niveaux logiques (récessifs et dominants) sont obtenus par la différence de potentiel entre les deux voies CAN L et CAN H.

Les niveaux de tension sur CANL et CANH dépendent du type Low Speed ou High Speed du bus.

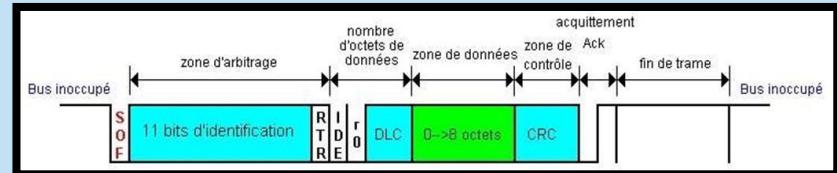
Ces niveaux de tension correspondent à un codage dit **NRZ** (No Return to Zero : il n'y a jamais de courant nul sur la ligne. La masse n'est plus utilisée et les niveaux logiques correspondent à 2 niveaux de tensions distincts).



Débit max: 125Kbits/s



Débit max: 1Mbits/s



3-Constitution d'une trame en format standard

Une trame se répartie en 7 champs:



Le début de trame SOF (Start Of Frame), 1 bit dominant ; la ligne étant précédemment au repos.



Composée de 12 bits (zone d'identification de la trame (11 bits + RTR)) : Ou 32 bits pour CAN 2.0B

- les 11 premiers indiquent l'**identité** du contenu du message, et servent également à l'**arbitrage** (gestion des priorités)
- bit RTR (**Remote Transmission Request**) : détermine s'il s'agit d'une trame de données (ex : régime moteur) ou d'une **demande** de message (ex : demande de T° eau). Le bit à 0 (dominant) pour une trame de données et le bit à 1 (récessif) pour une trame de demande.



Champ de commande constitué de 6 bits :

- les 2 premiers serviront pour une éventuelle évolution du protocole (**bits de réserve**) ;
- les 4 derniers permettent de coder le **nombre d'octets du champ de données**.



Ce champ contient de 0 à 8 octets de données (64 bits maxi)



Zone CRC (**Cyclic Redundancy Code**) de 15 bits : Ces bits sont recalculés à la réception et comparés aux bits reçus. S'il y a une différence, une erreur CRC est déclarée.



Zone d'acquittement (**ACKnowledge**) composé d'un bit à l'état récessif ainsi qu'un bit séparateur ACK. Le premier bit doit être forcé à l'état dominant par les stations ayant bien reçu cette trame.



Zone de fin de trame EOF (**End Of Frame**), 7 bits récessifs (à l'état 1).

Sommaire



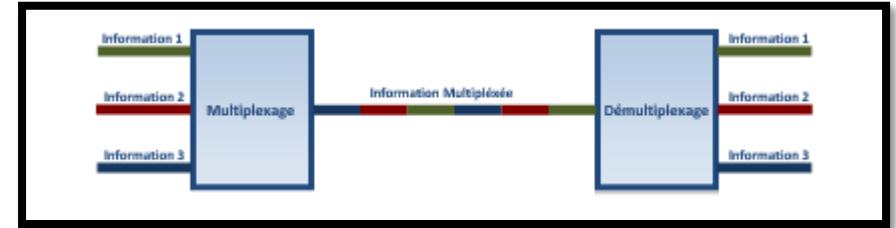
4 FISA S3E A4 Réseaux de TERRAINS/BUS CAN

• 4.1 Historique	63
• 4.2 Le bus CAN	64
• 4.3 Protocole CAN et couches OSI	66
• 4.4 Support de transmission	67
• 4.5 Codage de l'information et Stuffing	68
• 4.6 Les informations transmises sur le bus	69
• 4.7 Trame de données (DATA FRAME)	70
• 4.8 Trame de requête (REMOTE FRAME)	76
• 4.9 Travaux pratiques	78



FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

4.1 Historique



La société BOSCH développe dès le début des années 1980 une solution de multiplexage des informations circulant à bord de la voiture. Le bus CAN (Control Area Network) est alors développé et sera normalisé dès 1983. En 1985, Mercedes commercialise la première voiture (classe S) équipée d'un bus CAN et de cinq unités de calcul. Aujourd'hui, beaucoup de véhicules sont totalement multiplexés. Leurs réseaux permettent de relier entre plus d'une trentaine de calculateurs.

Domaines d'application

Le CAN est un réseau de terrain (ou réseau embarqué) qui occupe aujourd'hui une position de leader sur le marché automobile.

D'autres secteurs utilisent également le bus CAN :

- véhicules industriels
- matériel agricole
- bateaux
- avions
- production industrielle
- automatismes
- la domotique ...



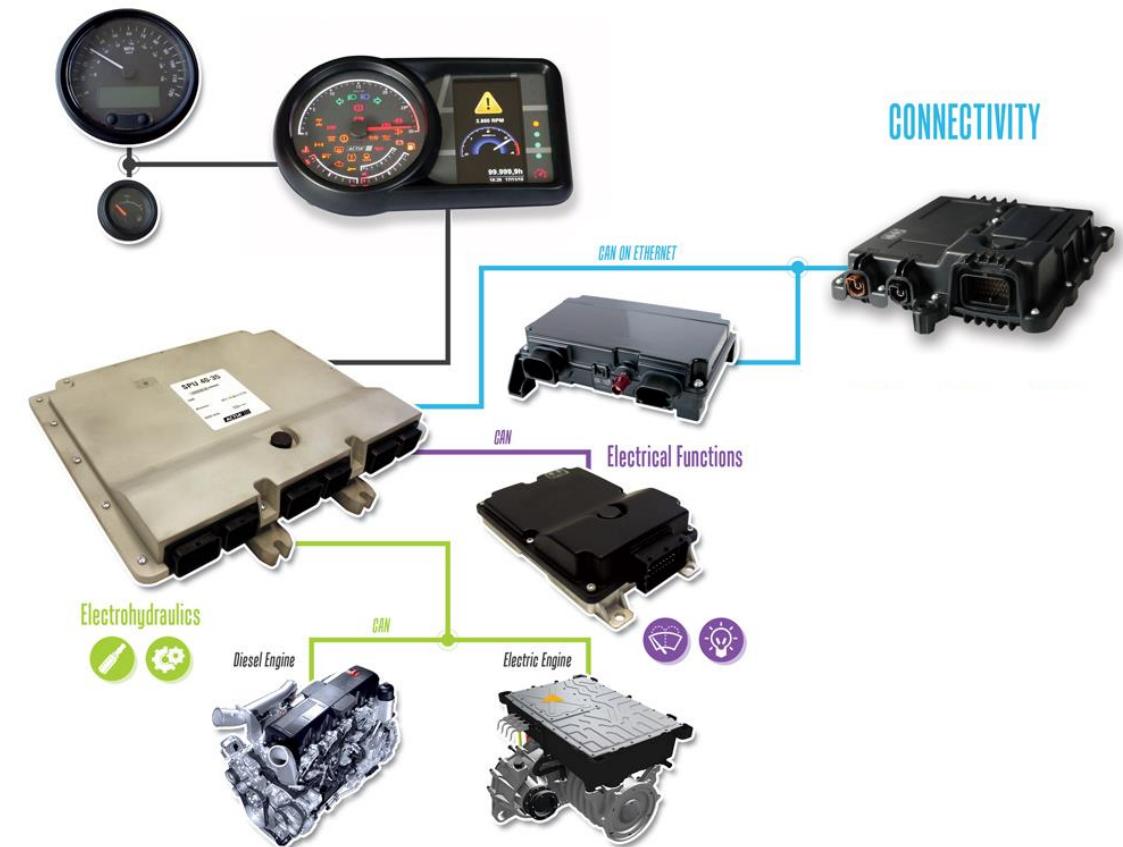
FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

4.2 Le bus CAN

Le bus CAN (Controller Area Network) est un bus série de terrain car il doit fonctionner dans un environnement limité et sévère (milieu industriel, atelier, voiture...) permettant la **transmission asynchrone** de données numériques. Il s'agit d'un bus multiplexé.

Les particularités de ce bus sont :

- bus multi maîtres où tous les participants ont les mêmes droits
- fiabilité élevée des mécanismes de protection du protocole

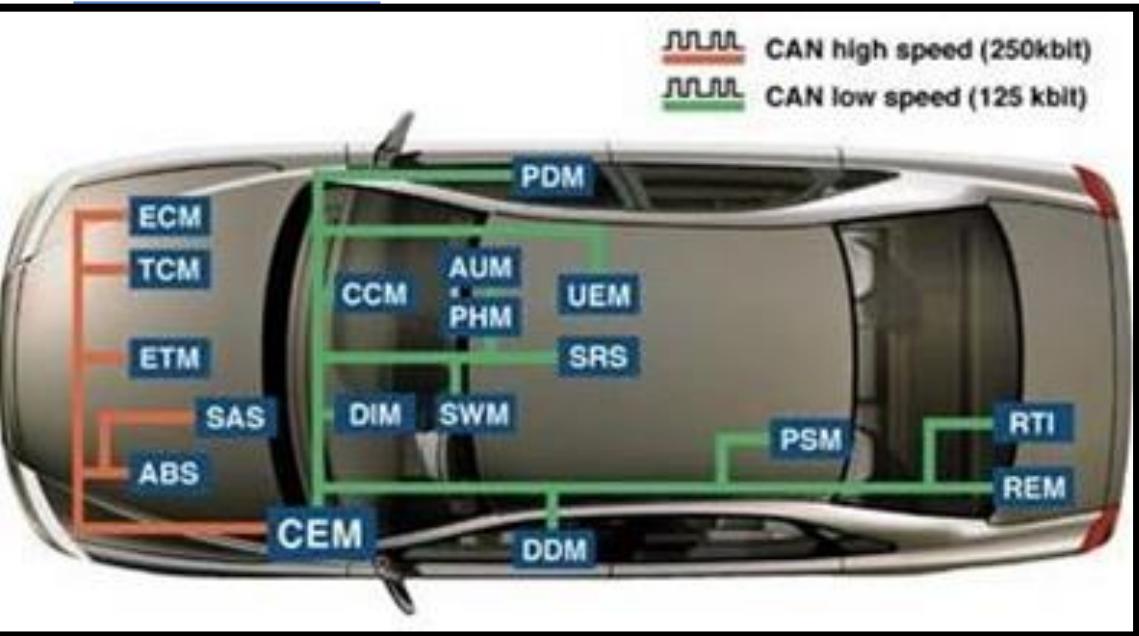


Il existe 2 protocoles CAN :

- le protocole **CAN 2.0A** dit CAN standard qui possède un identificateur de 11 bits ce qui permet de délivrer 2048 messages différents ;*
- le protocole **CAN 2.0B** ou CAN étendu qui possède un identificateur 29 bits soit plus de 536 millions de messages différents.*

Pour chacun des 2 protocoles il existe 2 types d'interfaces (normes de transmission) :

- le **CAN LowSpeed ou CAN-LS** (norme ISO 11519). Ce bus présente un débit max de **125 kbits/s** et peut raccorder jusqu'à **20 nœuds**. Dans l'automobile, il est appelé bus confort et utilisé pour la climatisation, la radio, le tableau de bord ... ;*
- le **CAN Highspeed ou CAN-HS**. Ce bus présente un débit max de **1 Mbits/s** et peut raccorder jusqu'à 30 nœuds. Dans l'automobile, il est utilisé pour les équipements de sécurité (freinage, moteur ...) ;*



FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

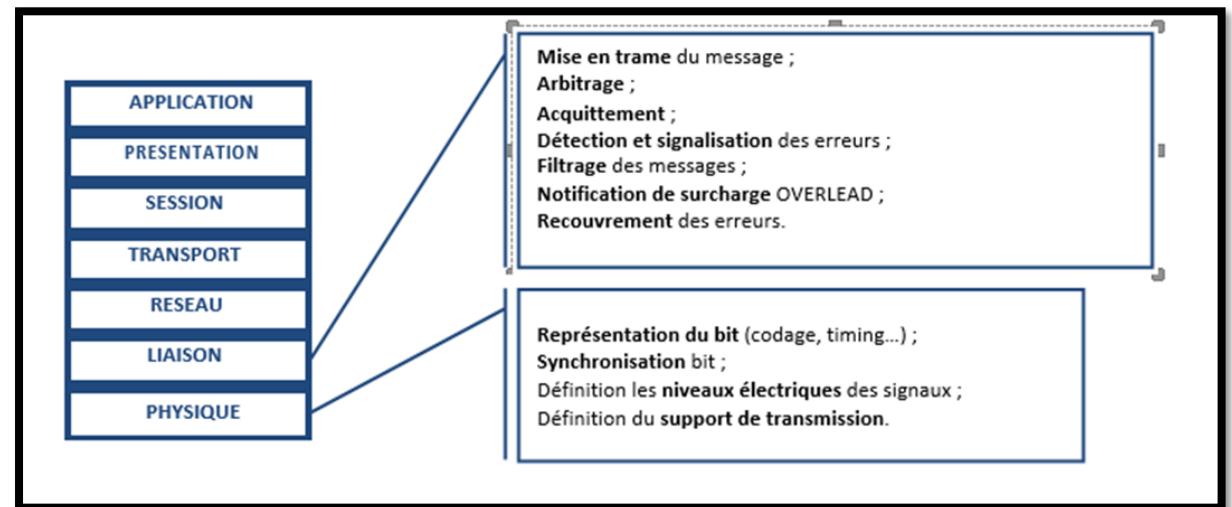
4.3 Protocole CAN et couches OSI

La structure du protocole du bus CAN possède implicitement les principales propriétés suivantes :

- hiérarchisation des messages ;
- garantie des temps de latence ;
- souplesse de configuration ;
- réception de multiples sources avec synchronisation temporelle ;
- fonctionnement multi maître ;
- détections et signalisations d'erreurs, retransmission automatique des messages altérés dès que le bus est de nouveau au repos. ;
- distinction d'erreurs (temporaire ou non-fonctionnalité permanente au niveau d'un nœud) ;
- déconnexion automatique des nœuds défectueux.

Le protocole CAN ne couvre seulement que deux des sept couches du modèle d'interconnexion **des systèmes ouverts OSI**

On retrouve ainsi dans le protocole CAN, la couche liaison de données (couche 2) et la couche physique (couche 1).



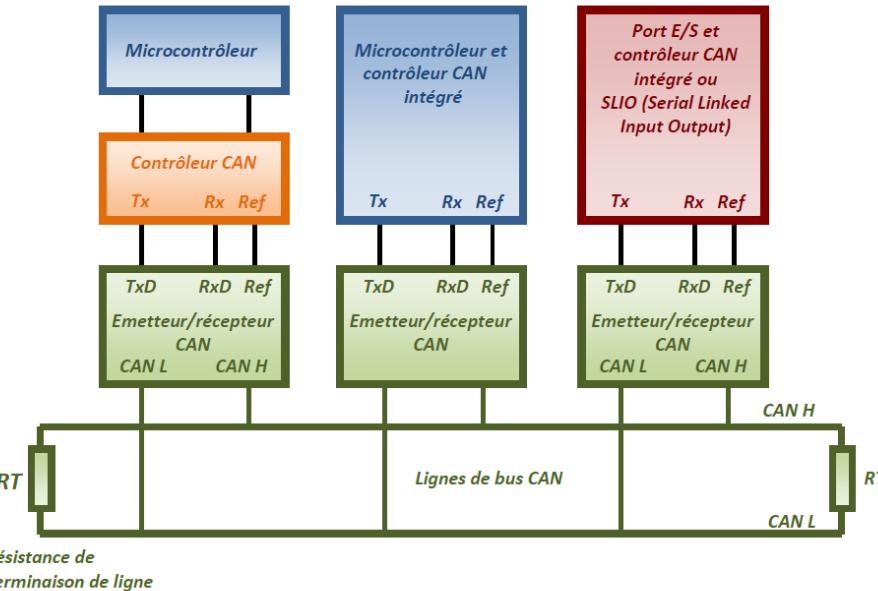
FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

4.4 Support de transmission

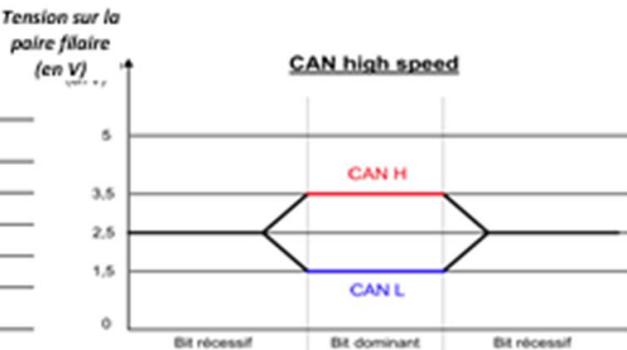
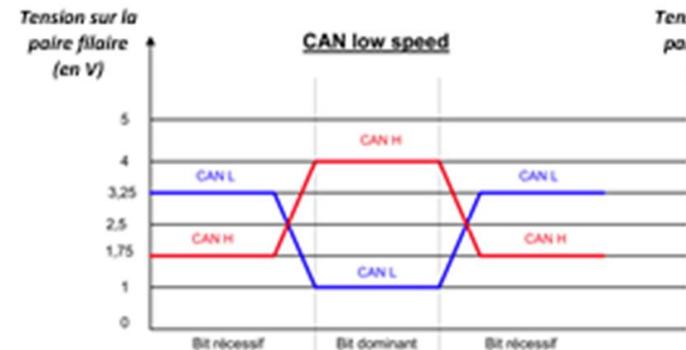
La transmission des données est effectuée sur une paire filaire différentielle torsadée. La ligne est donc constituée de deux fils : CAN L (CAN Low) et CAN H (CAN High). Le Signal CAN transmis est donc obtenu par la différence de tension entre les deux lignes. La ligne du bus doit se terminer par des résistances de terminaison.

Un nœud du bus CAN requiert pour fonctionnement au sein du réseau, microcontrôleur, un gestionnaire de protocole CAN (ou contrôleur CAN) et une interface de ligne CAN (ou Emetteur / Récepteur CAN)

Paramètres	CAN LS	CAN HS
Débit	125 kb/s	125 kb/s à 1 Mb/s
Nombre de nœuds sur le bus	2 à 20	2 à 30
Courant de sortie (mode émission)	> 1 mA sur 2,2 kΩ	25 à 50 mA sur 60Ω
Niveau dominant (NL0)	CAN H = 4V CAN L = 1V	CAN H = 3,5 V CAN L = 1,5 V
Niveau récessif (NL1)	CAN H = 1,75V CAN L = 3,25V	CAN H = 2,5 V CAN L = 2,5 V
Caractéristique du câble	30 pF entre les câbles de ligne	2 × 120 Ω
Tensions d'alimentation	5 V	5 V

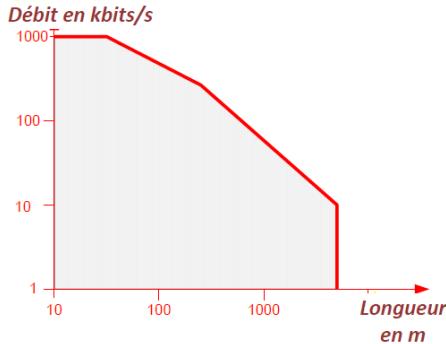


Les nœuds sont câblés sur le bus de telle manière qu'en cas d'émission simultanée de deux nœuds, le NL0 s'impose par rapport au NL1 : Le NL0 est donc appelé état dominant et le NL1, état récessif.



FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

4.5 Codage de l'information et Stuffing



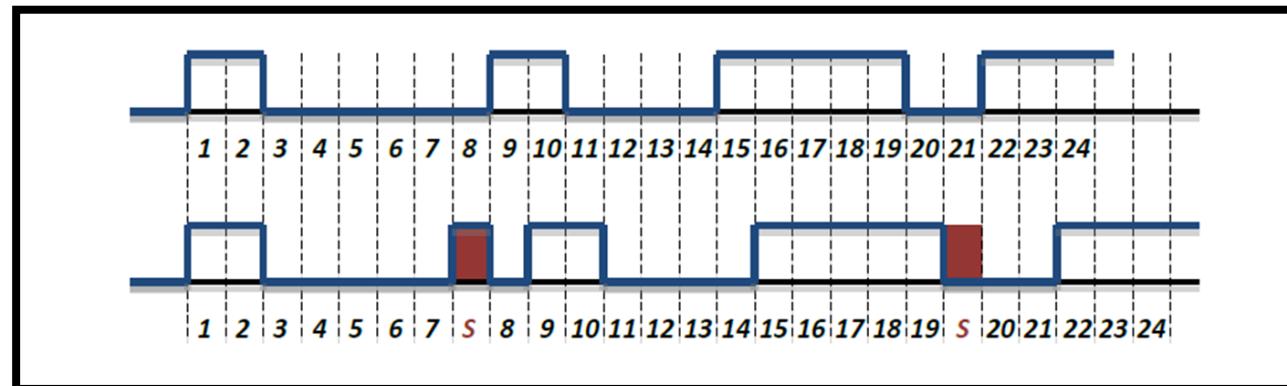
Débit	Longueur	Durée d'un bit
1 Mbit/s	30 m	1 µs
800 kbit/s	50 m	1,25 µs
500 kbit/s	100 m	2 µs
250 kbit/s	250 m	4 µs
125 kbit/s	500 m	8 µs
62,5 kbit/s	1000 m	16 µs
20 kbit/s	2500 m	50 µs
10 kbit/s	5000 m	100 µs

Les bits transitant sur le bus sont codés avec la méthode NRZ (Non Return to Zero). Pendant la durée totale du bit, le niveau de tension de la ligne est maintenu, c'est à dire sa valeur reste constante qu'elle soit dominante ou récessive.

Une des caractéristiques du codage NRZ est que le niveau du bit est maintenu pendant toute sa durée. Cela pose des problèmes de fiabilité si un grand nombre de bits identiques se succèdent. La technique du Bit Stuffing impose au transmetteur d'ajouter automatiquement un bit de valeur opposée lorsqu'il détecte 5 bits consécutifs identiques dans les valeurs à transmettre.

La transmission différentielle du signal sur le bus CAN assure l'immunité électromagnétique car les deux lignes du bus sont affectées de la même manière par un signal perturbateur.

La longueur du bus et le débit maximum sont liés par la courbe suivante :



4.6 Les informations transmises sur le bus

Le concept de communication du bus CAN est celui de la diffusion d'information (**broadcast**) :

- chaque station connectée au réseau écoute les trames transmises par les stations émettrices.
Chaque nœud décide quoi faire du message, s'il doit y répondre ou non, s'il doit agir ou non
- le protocole CAN autorise différents nœuds à accéder simultanément au bus. Un procédé rapide et fiable d'arbitrage détermine le nœud qui émet en premier.

Il existe 4 types de trames circulant sur le bus CAN :

- **trame de données** (data frame) : trame qui transporte des données ;
- **trame de requête** (remote frame) : trame émise par un nœud désirant recevoir une trame de données (l'identificateur est le même pour les deux trames dans ce cas) ;
- **trame de surcharge** (overload frame) : trame qui indique qu'une station est surchargée pendant un certain laps de temps. Elle permet d'introduire un délai entre 2 trames (elle ne contient pas d'information) ;
- **trame d'erreur** (error frame) : trame transmise lors de la détection d'une erreur. Elle indique la nature des erreurs (de bit, de stuffing, de CRC, d'acquittement...) et permet la correction en demandant de renvoyer à nouveau la donnée.

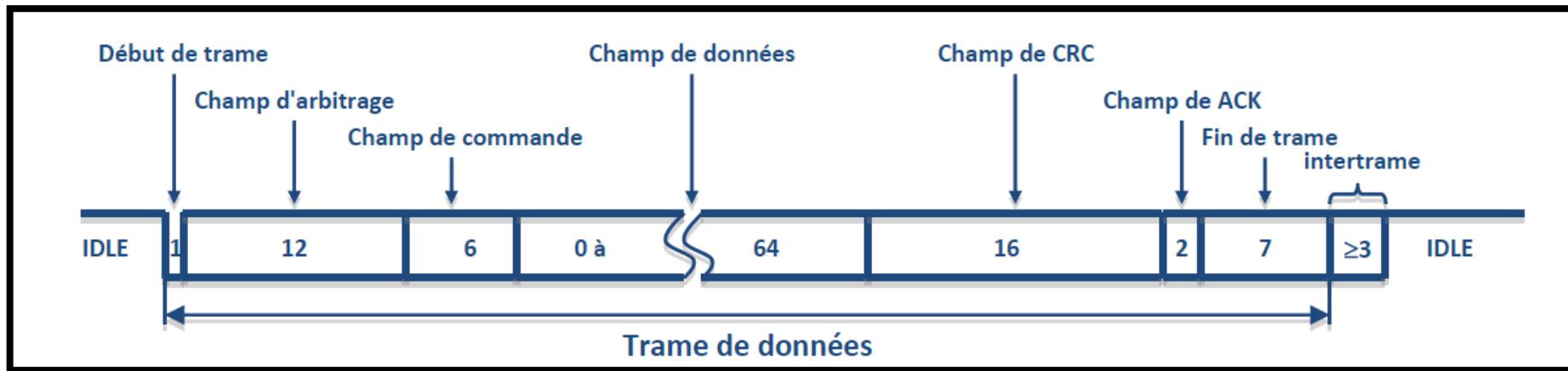
Les trames de données et trames de requêtes sont séparées des trames précédentes par un champ de bits appelé espace intertrame (interframe space). Il s'agit d'une suite de plusieurs bits récessifs (NL1).

4.7 Trame de données (DATA FRAME)

La trame de données standard CAN 2.0A se décompose en 7 parties champs :

début de trame (1 bit dominant) : Start Of Frame (SOF) ; champ d'arbitrage (12 bits) : arbitration field ; champ de commande (6 bits) : control field ; champ de données (0 à 64 bits) : data field ; champ de CRC (16 bits) : CRC sequence ; champ d'acquittement (2 bits) : ACKnowledgement field ; fin de trame (7 bits) : End Of Frame (EOF).

On trouve ensuite une 8ème zone dite espace intertrame (**interframe**) qui appartient à la trame.

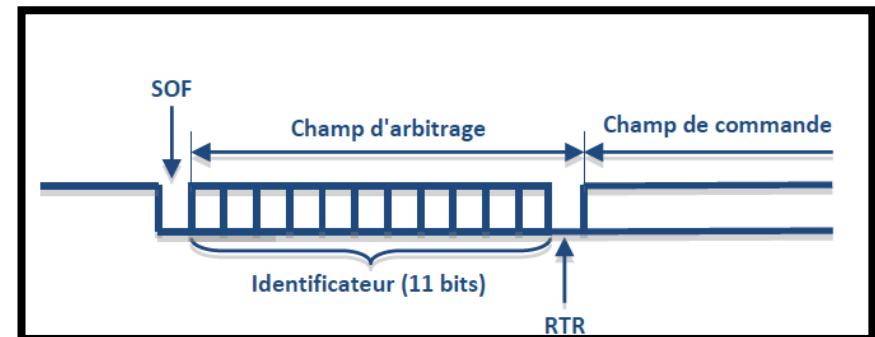


FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

Dès que le bus est libre (bus idle), n'importe quel noeud relié au réseau peut émettre un nouveau message.

Le bit SOF (début de trame de données) est dominant. Il signale à toutes les stations le début d'un échange. Cet échange ne peut démarrer que si le bus était précédemment au repos.

Dans le **champ d'arbitrage**, les identificateurs de chaque message permettent de définir quel message est prioritaire sur tel autre.

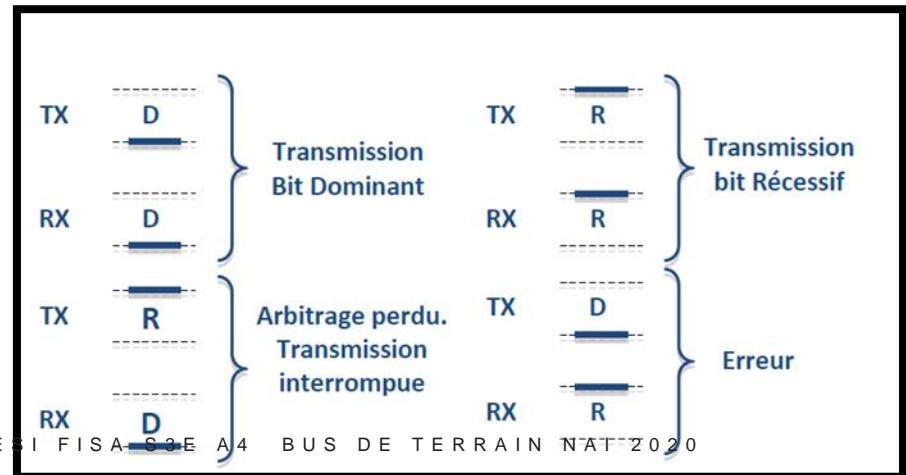


La longueur de l'identificateur est de 11 bits. Les bits sont transmis dans l'ordre ID10 à ID0 (du bit de poids fort au bit de poids faible). Les 7 premiers bits de poids fort (ID10 à ID4) ne doivent pas être tous récessifs. L'identificateur a la forme suivante : ID = (ID10 ID9 ID8 ID7 ID6 ID5 ID4 X). Les 4 derniers bits ne sont pas historiquement utilisés. Le nombre maximal d'identificateurs est donc de 2032.

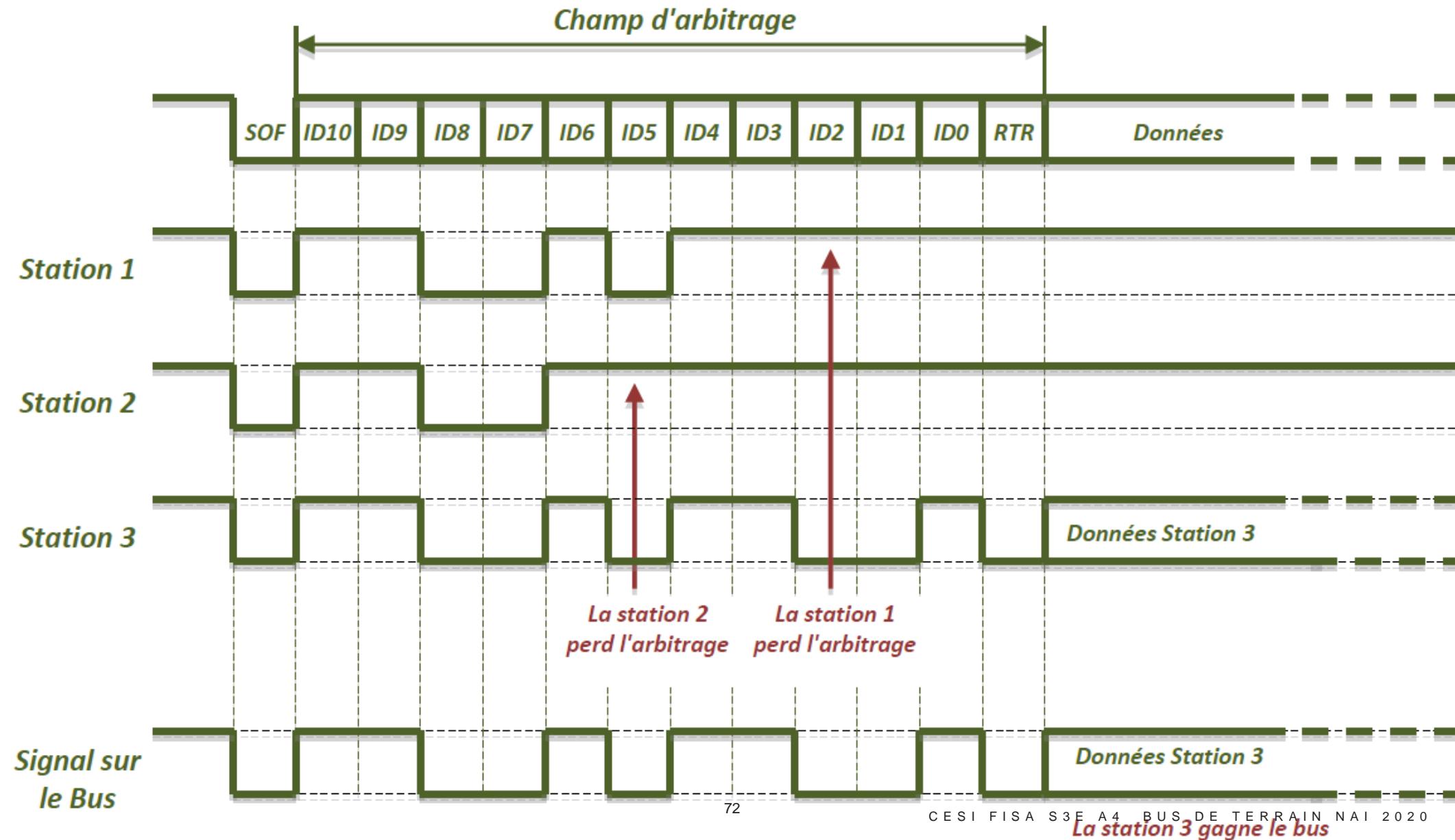
Pour une trame de données, le bit RTR doit être dominant.

Pendant le champ d'arbitrage, les bits transmis et reçus sont comparés par l'interface CAN.

Le champ d'arbitrage est constitué des bits de l'identificateur ainsi que du bit RTR (Remote Transmission Request).

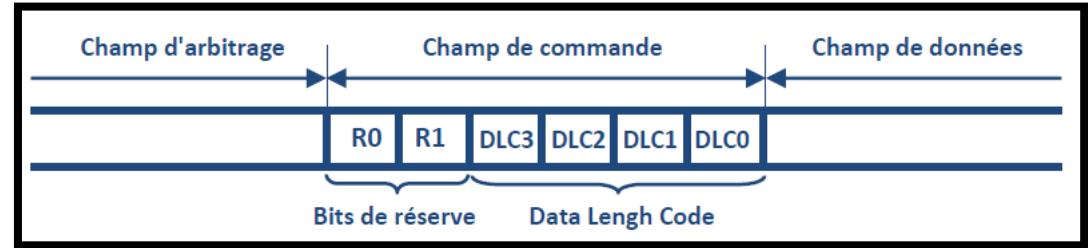


Exemple d'arbitrage entre 3 stations



FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

Le **champ de commande** est constitué de 6 bits.



Les bits R0 et R1 sont les bits de réserves (dominants en trame CAN 2.0A). Ils sont en réserve pour des usages ultérieurs et permettent d'assurer des compatibilités avec de futures évolutions.

Les 4 derniers bits du champ de commande sont les bits DLC (Data Length Code) indiquant le nombre d'octets qui seront contenus dans le champ de données.

Nb d'octets	DLC3	DLC2	DLC1	DLC0
0	D	D	D	D
1	D	D	D	R
2	D	D	R	D
3	D	D	R	R
4	D	R	D	D
5	D	R	D	R
6	D	R	R	D
7	D	R	R	R
8	R	D	D	D

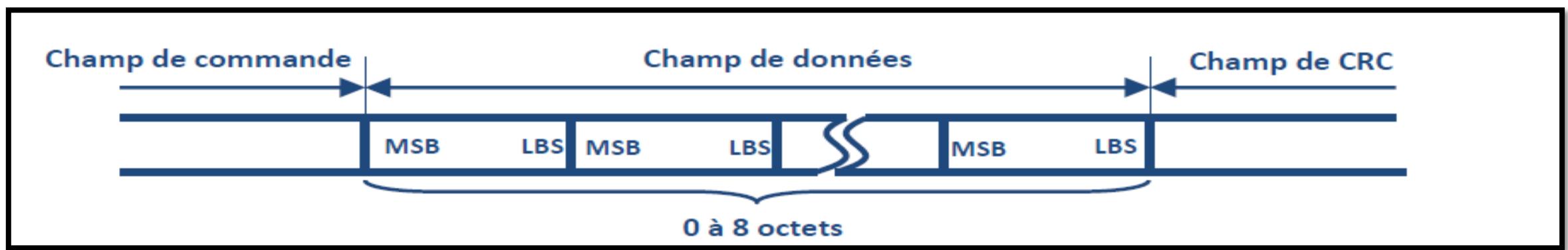
CESI FISA S3E A4 BUS DE TERRAIN NAI 2010

FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

Champ de données

Le champ de données contient les données utiles transmises.

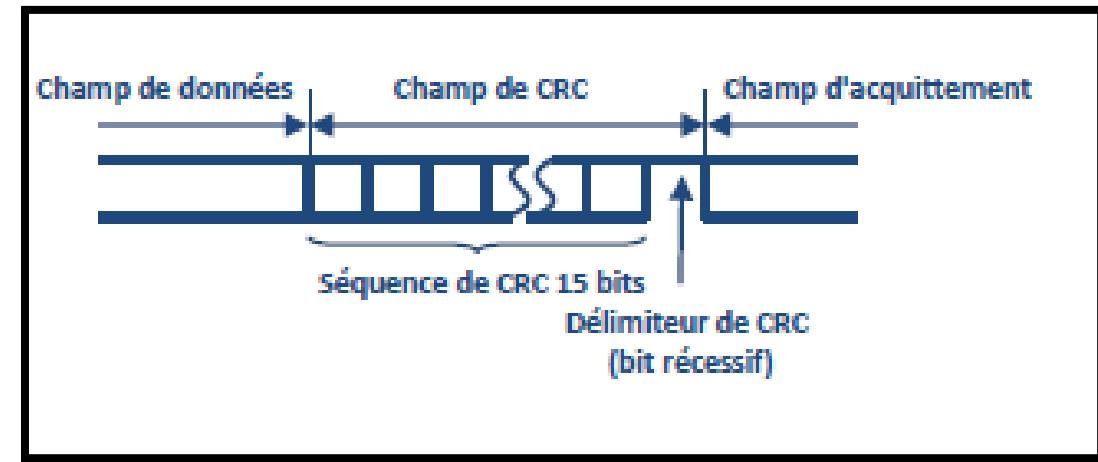
Il peut être composé de 0 à 8 octets. Dans chaque octet, le MSB est transmis en premier.



Champ de CRC

Le champ de CRC est composé d'une séquence de 15 bits suivi du CRC Delimiter (1 bit récessif).

La séquence de CRC (Cyclic Redundancy Code) permet de vérifier l'intégrité des données transmises. Il s'agit d'un polynôme calculé de la même manière par l'émetteur et le récepteur à partir des bits du SOF, du champ d'arbitrage, du champ de contrôle et du champ de donnée.



Fin de trame

La trame de donnée se termine par une séquence de 7 bits récessifs. Ce champ possède une structure fixe. Les logiques de codage (à l'émission) et de décodage (aux réceptions), de bit stuffing sont désactivées pendant la séquence du champ de fin de trame.

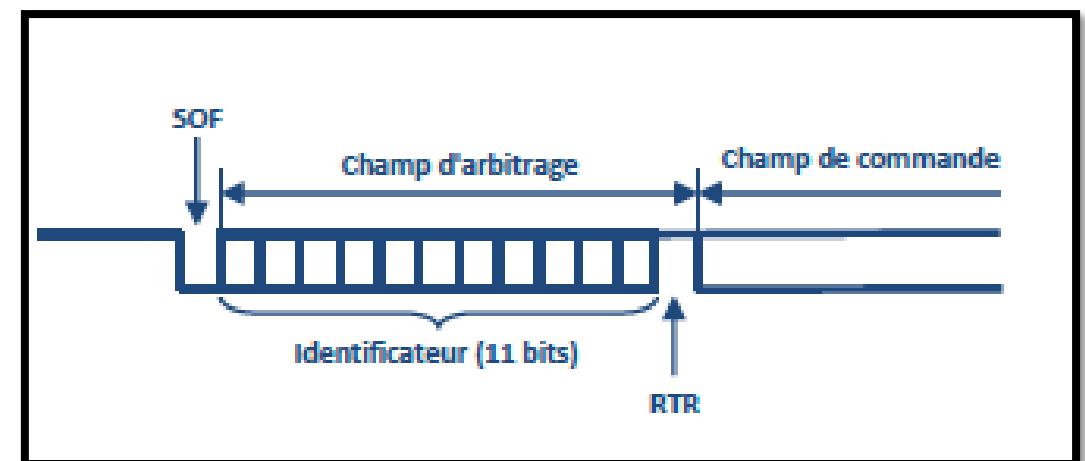
4.8 Trame de requête (REMOTE FRAME)

Constitution d'une trame de requête

Une station nécessitant des données particulières peut initialiser la demande d'une transmission en envoyant une trame de requête (remote frame). La structure d'une trame de requête est identique à celle de donnée hormis le champ de données qui est optionnel dans ce type de trame.

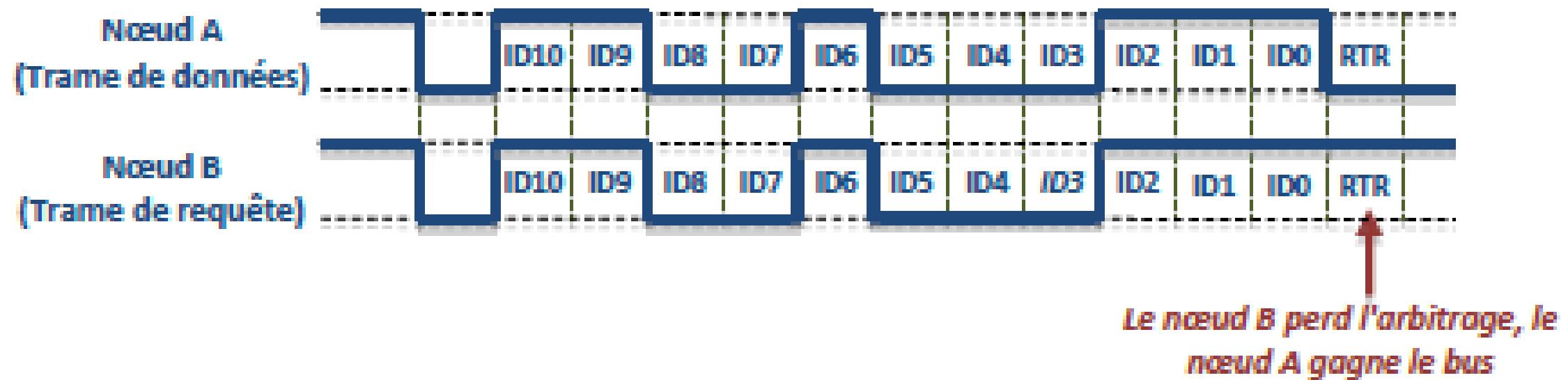
Champ d'arbitrage

Contrairement à la trame de données le bit RTR du champ d'arbitrage d'une trame de requête est un bit récessif. C'est donc ce bit qui différencie une trame de données d'un trame de requête.



Arbitrage trame de requête et trame de données

Arbitrage entre une trame de données et une trame de requête portant le même identificateur : la trame de donnée est prioritaire sur la trame de requête.



FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

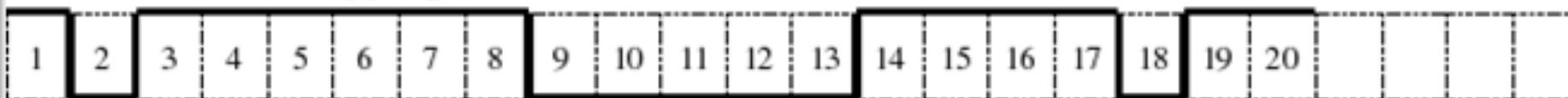
4.9 Travaux pratiques

4.9.1 Exercice Stuffing

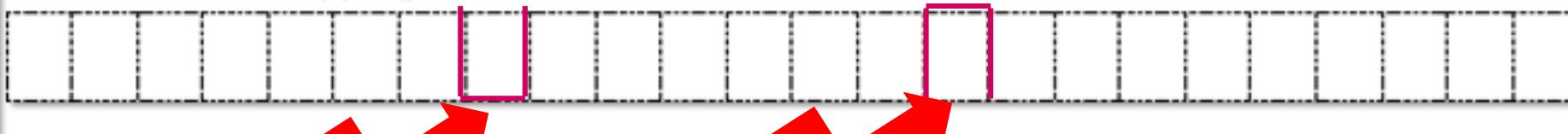
Compléter le diagramme suivant en respectant la méthode de Bit-Stuffing.

Réponse :

Trame avant *stuffing*



Trame avec *stuffing*



STUFF

STUFF

4.9.2 Prise de contrôle du CAN

Compléter le troisième chronogramme (résultante sur le bus) et indiquer le nœud qui a réussi à émettre sa trame.

Réponses :

Noeud 2



Noeud 1



Etat du bus (résultante)



Le nœud 2 gagne le bus.

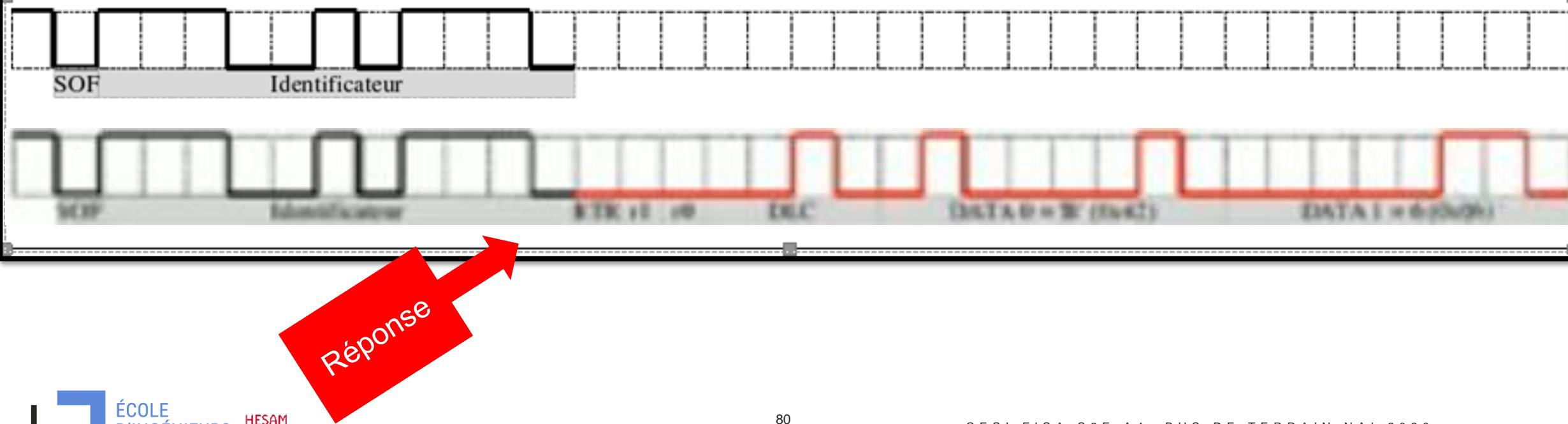
2 gagne
le bus

4.9.3 Trame CAN

Exercice n°12

Compléter la trame (jusqu'au champ CRC exclu) dans le cas où un noeud émet les données 'B' (0x42) suivi de la valeur 6.

Réponse :



FISA S3E A4 Réseaux de TERRAINS/BUS CAN/

4.9.4 Travaux pratiques sur mallette

Observation du bus CAN

Objectifs du TP :

Approfondir le bus CAN

Visualiser des trames CAN

Comprendre la puissance de bus CAN et pourquoi ce bus est autant utilisé aujourd'hui dans les véhicules

Matériel utilisé :

- Malette Modèle : MDET Bus & Réseaux Kits de formation pour l'apprentissage des bus de communication
- Un PC équipé du logiciel compatible avec l'interface OBD
- Interface prise OBD - USB
- Un oscilloscope numérique
- Un analyseur logique
- Documentation

Expérimentation n°1:

Utiliser l'interface OBD – USB pour visualiser les trames CAN présentes sur le bus CAN de la mallette. Les trames sont émises par une carte de simulation d'un véhicule.

Visualiser un trame CAN à l'oscilloscope ou analyseur logique et la décoder à la main en identifiant les champs de la trames.

Expérimentation n°2: Envoi d'une trame CAN depuis le PIC

Envoyer une trame CAN sur le bus CAN de la mallette à partir du PIC.

Vous choisirez l'identifiant, la taille et les données à transmettre.

Le type de bus et sa vitesse vous seront donnés.

Visualiser la trame CAN à l'oscilloscope ou analyseur logique et la décoder à la main en identifiant les champs de la trames afin de vérifier qu'elle corresponde à la trame envoyée.



5

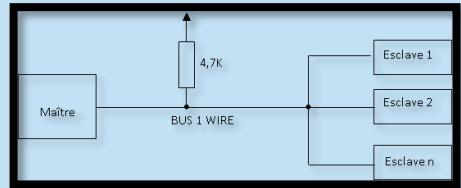
FISA S3E
Réseau de terrain

Bus 1 WIRE

FISA S3E A4 Réseaux de TERRAINS/BUS 1 WIRE /

5 Résumé

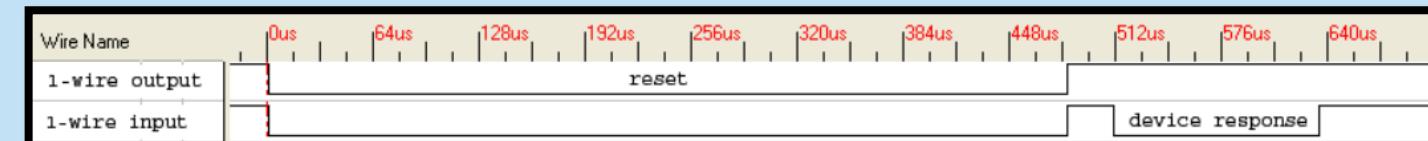
Permet la communication sur 1 fil (1 WIRE), plus le fil de masse.



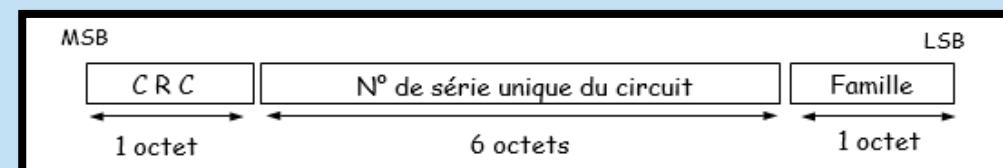
1-Reset

L'état repos du bus à l'état haut.

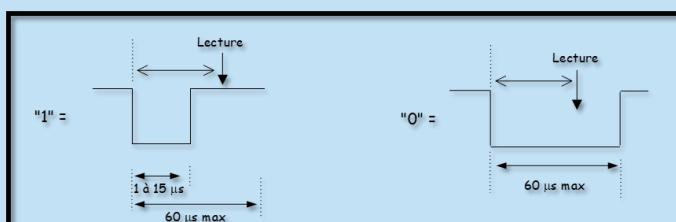
Reset



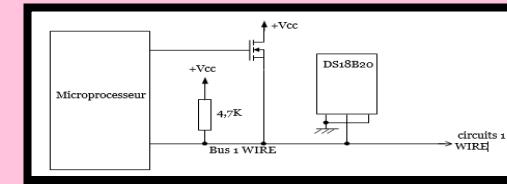
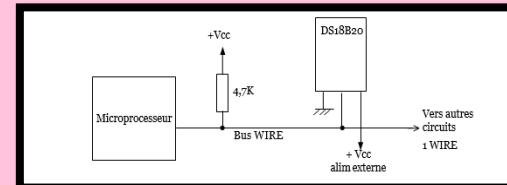
2- Adresse unique d'un circuit



3- Emission 1 / Emission 0



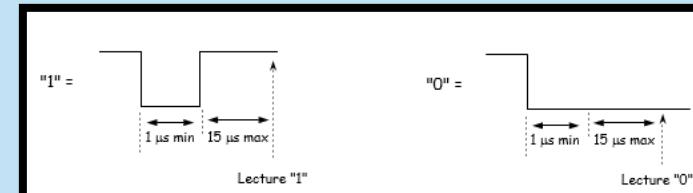
Alimentation



Alimentation Parasite

3-

Réception 1 / Réception 0



4- Commande ROM

0x33'= READ ROM

0x55 = MATCH ROM

0xCC= SKIP ROM

0xEC'= CONDITIONAL SEARCH

0xF0 = SEARCH ROM

5. CRC

index	0	94	188	226	97	63	221	131	194	156	126	32	163	253	31	65
16	157	195	33	127	252	162	64	30	95	1	227	189	62	96	130	220
32	35	125	159	193	66	28	254	160	225	191	93	3	128	222	60	98
48	190	224	4	92	223	129	99	61	124	34	192	158	29	67	161	255
64	70	24	250	164	39	121	155	197	132	218	56	102	229	187	89	7
80	219	133	103	57	186	228	6	88	25	71	165	251	120	38	196	154
96	101	59	217	135	4	90	184	230	167	249	27	69	198	152	122	36
112	248	166	68	26	153	199	37	123	58	100	134	216	91	5	231	185
128	140	210	48	110	237	179	81	15	78	16	242	172	47	113	147	205
144	17	79	173	243	112	46	204	146	211	141	111	49	178	236	14	80
160	175	241	19	77	206	144	114	44	109	51	209	143	12	82	176	238
176	50	108	142	208	83	13	239	177	240	174	76	18	145	207	45	115
192	202	148	118	40	171	245	23	73	8	86	180	234	105	55	213	139
208	87	9	235	181	54	104	138	212	149	203	41	119	244	170	72	22
224	233	183	85	11	136	214	52	106	43	117	151	201	74	20	246	168
240	116	42	200	150	21	75	169	247	182	232	10	84	215	137	107	53

Soit fcrc(x) la fonction qui lit la valeur dans la table à la position x

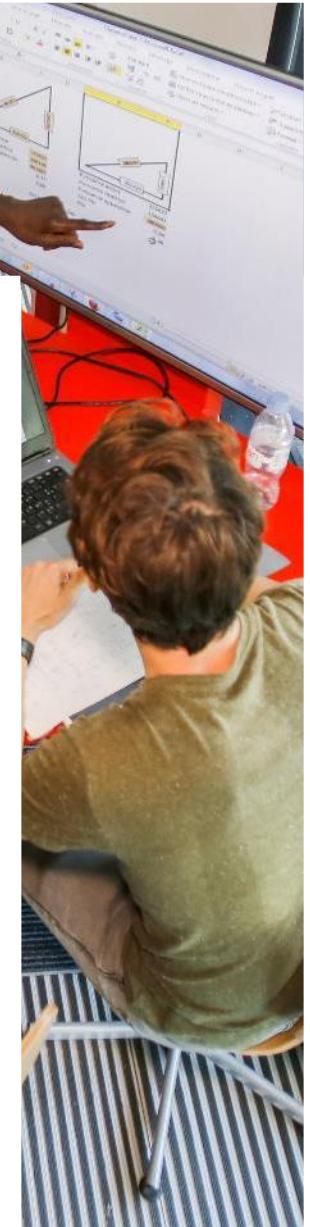
CRC= (00 xor LSB); /* Famille*/
CRC= fcrc(CRC) xor O2; /*Octet2*/
CRC= fcrc(CRC) xor O3; /*Octet3*/
CRC= fcrc(CRC) xor O4; /*Octet4*/
CRC= fcrc (CRC)xor O5; /*Octet5*/
CRC= fcrc (CRC)xor O6; /*Octet6*/
CRC= fcrc (CRC)xor O7; /*Octet7*/

Sommaire



5 FISA S3E A4 Réseaux de TERRAINS/BUS 1 WIRE

• 5.1 Historique	85
• 5.2 Réception d'un bit par le maître:	87
• 5.3 Commande ROM	87
• 5.3 Calcul du CRC	89
• 5.4 Liste des composants 1-WIRE	91
• 5.5 Bibliographie et sources	92
• 5.6 Travaux pratiques sur mallette	93



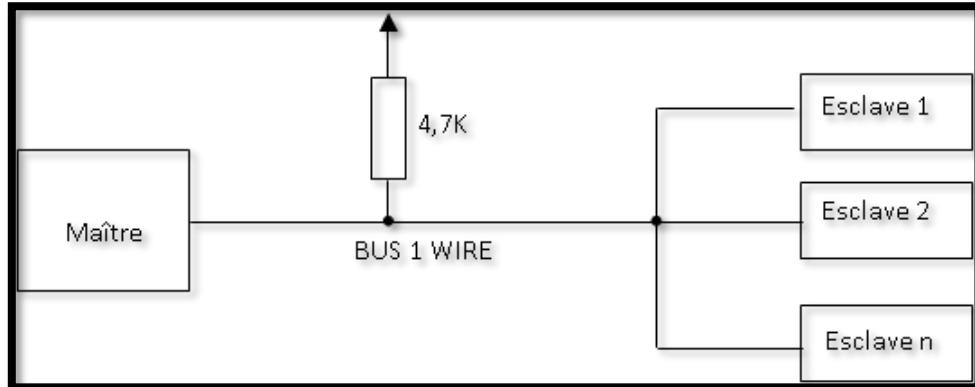
FISA S3E A4 Réseaux de TERRAINS/BUS 1 WIRE/

5.1 Historique

Le bus **1 WIRE** de **DALLAS**, permet de connecter et de faire dialoguer entre eux des circuits sur un seul fil.

Ce système de bus utilise un seul maître, qui pourra dialoguer avec un ou plusieurs esclaves.

Toutes les commandes et données sont envoyées avec le bit LSB en tête. Le fil unique du bus doit être tiré au +Vcc par une résistance de 4,7K . L'état repos du bus est donc un état haut.



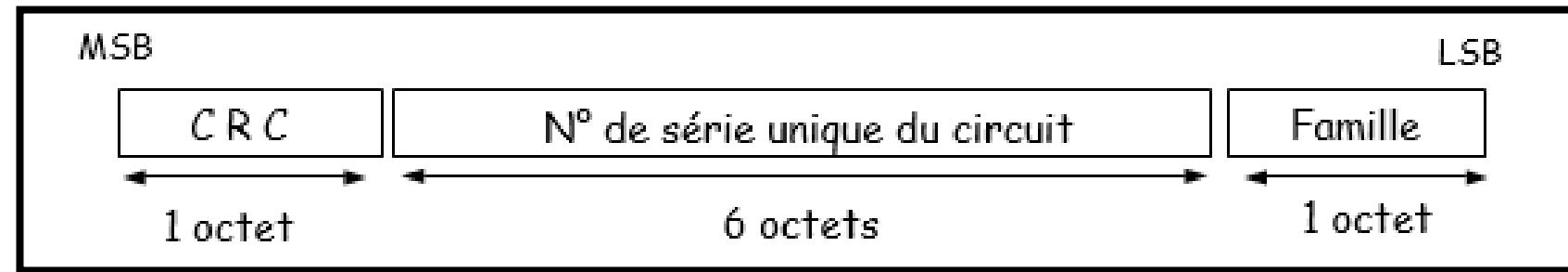
Si le bus est maintenu à l'état bas plus de 480 µs par le maître, tous les composants sur le bus sont remis à zéro.

C'est le pulse d'initialisation ou de Reset.

Après un délai de 15 à 60 µs, le ou les esclaves raccordés, forcent le bus à l'état bas pendant 60 à 240 µs pour signaler leur présence.

Chaque circuit possède une adresse physique unique, gravée dans la puce à la fabrication.

Cette adresse est constituée de 64 bits soit 8 octets. Le premier octet détermine le type de famille auquel appartient le circuit. Les 6 octets suivants, constituent le code propre du circuit. Le dernier octet est le CRC. C'est un octet de contrôle calculé à partir des 56 bits précédents.



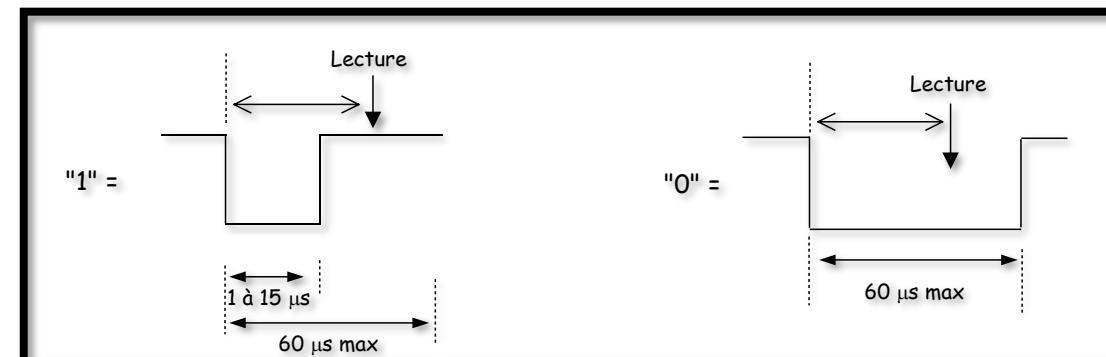
Toute transaction entre un maître et un ou plusieurs esclaves, débute par une initialisation, constituée par l'envoi du pulse de Reset par le maître. Le maître doit ensuite envoyer une commande de type ROM qui est propre au protocole 1 Wire, et que tous les circuits de ce type vont reconnaître. Cela va permettre entre autre de sélectionner un circuit parmi les différents esclaves qui ont répondu présents au pulse de Reset. Le dialogue et l'échange de données pourra ensuite commencer, entre le maître et l'esclave sélectionné.

Emission d'un bit du maître vers l'esclave:

Le maître force le bus à "0" pendant 1 à 15 µs. L'esclave va lire le bus entre 15 et 45 µs après le front descendant (valeur typique 30 µs).

Si on veut émettre un "1", il faut repasser le bus à "1" immédiatement, et ne plus rien faire jusqu'à $t = 60 \mu s$. Pour émettre un "0" il faut laisser le bus à "0" jusqu'à $t = 60 \mu s$, puis repasser le bus à "1".

La durée du bit est donc de 60 µs, ce qui donne un débit de 16 kbits/sec.

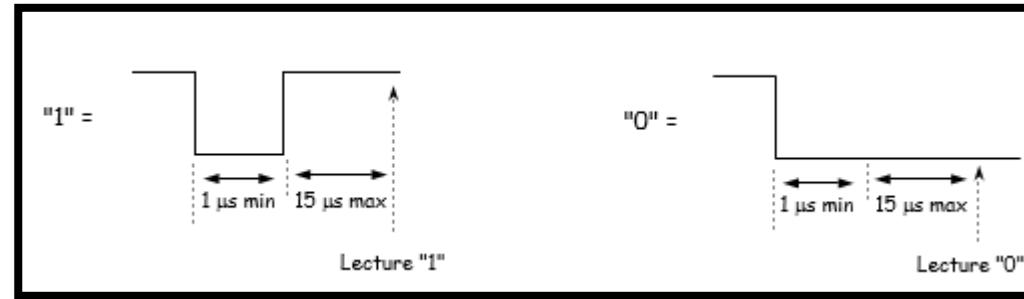


FISA S3E A4 Réseaux de TERRAINS/ BUS 1 WIRE/

5.2 Réception d'un bit par le maître:

Le maître force le bus à "0" pendant au moins 1 µs. Si l'esclave veut émettre un "1", il laisse le bus libre donc tiré à "1". Pour émettre un "0", l'esclave doit tirer le bus à "0" pendant 15 µs au minimum.

Le maître devra donc dans tous les cas lire le bus 15 µs maximum après avoir tiré le bus à "0" pendant 1 µs. L'état du bus donnera alors le bit transmis par l'esclave.



5.3 COMMANDES ROM:

Ces commandes sont constituées d'un octet que le maître devra envoyer après avoir fait un reset.

0x33'=READ ROM

Cette commande ne peut être utilisée que s'il n'y a qu'un seul esclave sur le bus. Celui ci répond alors ces 64 bits de code.

0x55 = MATCH ROM

Cette commande suivi de 64 bits de code, va permettre au maître de sélectionner un esclave particulier.

0xCC= SKIP ROM

Commande d'appel général, pour adresser tous les esclaves. Cette fonction est utile pour adresser un esclave qui est seul sur le bus, sans avoir à envoyer les 64 bits de son code.

0xEC'=CONDITIONAL SEARCH

Cette commande fonctionne comme la commande SEARCH ROM, à la différence que seul les circuits ayant une condition bien spécifiée participent à la recherche. Par exemple les circuits de mesure de la température qui ont le Flag d'alarme actif ou les port E/S qui ont leur sortie à "1".

0xF0 = SEARCH ROM

Cette commande va permettre de rechercher bit à bit les codes de tous les esclaves raccordés au bus 1 Wire.

En réponse à cette commande, les esclaves envoient leur premier bit, puis ce même bit inversé. Le maître émet à son tour ce premier bit. Les esclaves qui reconnaissent leur 1er bit restent à l'écoute, et les autres s'éliminent et ne répondront plus. Les esclaves toujours présents vont maintenant envoyer leur 2eme bit, puis ce même 2eme bit mais inversé. Le maître comme précédemment va émettre ce 2eme bit. Les esclaves qui ne reconnaissent pas leur 2eme bit vont s'éliminer.

Quand le maître reçoit le bit et son inverse à : 1 1 c'est qu'il n'y a pas de circuit sur le bus 1Wire.

Quand le maître reçoit le bit et son inverse à : 0 0 c'est qu'il y a conflit, car des esclaves ont un "1" et des autres un "0" à cette position. Dans ce cas il enverra en réponse un bit à "0" pour ne garder que les circuits ayant un "0" à cette position et éliminer ceux qui ont un "1".

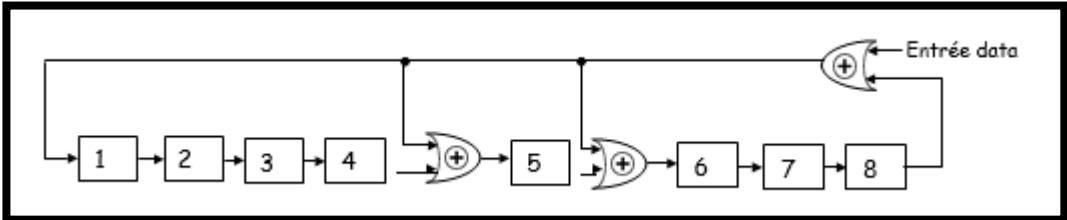
Quand le maître reçoit le bit et son inverse à : 0 1 c'est qu'il n'y a que des circuits ayant un bit à "0" à cette position. Il enverra un "0" pour garder tous ces circuits. Et s'il a reçu 1 0 il enverra "1", car le bit de cette position est à "1" et on gardera les circuits.

Le principe général de la recherche est de **DESELECTER** les uns après les autres les circuits à chaque conflit sur les différentes positions des bits. A la fin de chaque étape de recherche, le maître connaît un nouveau code de 64 bits complet d'un circuit. L'étape suivante est identique jusqu'au niveau de la dernière décision après le conflit. Le maître part alors dans la direction opposée, il enverra un "1" alors qu'il n'avait gardé que les circuits ayant un "0" à cette position. Ainsi, bit par bit, on va arriver à lire les 64 bits de tous les esclaves. Le maître va ainsi savoir combien il y a d'esclaves sur le bus et quelles sont leurs codes propres.

5.4 Calcul du CRC

Le polynôme générateur du CRC est :

Sa représentation est la suivante: $X^8 + X^5 + X^4 + 1$



Calcul pratique du CRC:

Quand on aura reçu les 56 premiers bits du circuit, soit 7 octets, on devra calculer l'octet de CRC pour le comparer à celui que l'on va recevoir avec les 8 bits restants.

Pour éviter les calculs complexes du polynôme, on va utiliser une table indexée de 256 valeurs décimales.

index	0	94	188	226	97	63	221	131	194	156	126	32	163	253	31	65
0	0	94	188	226	97	63	221	131	194	156	126	32	163	253	31	65
16	157	195	33	127	252	162	64	30	95	1	227	189	62	96	130	220
32	35	125	159	193	66	28	254	160	225	191	93	3	128	222	60	98
48	190	224	4	92	223	129	99	61	124	34	192	158	29	67	161	255
64	70	24	250	164	39	121	155	197	132	218	56	102	229	187	89	7
80	219	133	103	57	186	228	6	88	25	71	165	251	120	38	196	154
96	101	59	217	135	4	90	184	230	167	249	27	69	198	152	122	36
112	248	166	68	26	153	199	37	123	58	100	134	216	91	5	231	185
128	140	210	48	110	237	179	81	15	78	16	242	172	47	113	147	205
144	17	79	173	243	112	46	204	146	211	141	111	49	178	236	14	80
160	175	241	19	77	206	144	114	44	109	51	209	143	12	82	176	238
176	50	108	142	208	83	13	239	177	240	174	76	18	145	207	45	115
192	202	148	118	40	171	245	23	73	8	86	180	234	105	55	213	139
208	87	9	235	181	54	104	138	212	149	203	41	119	244	170	72	22
224	233	183	85	11	136	214	52	106	43	117	151	201	74	20	246	168
240	116	42	200	150	21	75	169	247	182	232	10	84	215	137	107	53

Exemple: On vient de recevoir les 7 octets suivants :

00 00 04 0C 38 F0 01

Le CRC que l'on va calculer est initialisé au départ à la valeur 00. On part du code de famille qui est le 1er octet: 01

On fait le OU Exclusif entre le CRC et l'octet soit : 0x00 XOR 0x01.

Le résultat est 0x01 soit 0d01. Cette valeur est l'index de la table qui va donner le CRC . On trouve à l'index 01 de la table la valeur décimale 94 soit 0x5E'. Donc CRC = 5E.

On fait maintenant le OU Exclusif entre le nouveau CRC et le 2eme octet. 0x5E XOR 0xF0 donne : 0xAE soit 174 en décimal. Cette valeur d'index nous donne dans la table un nouveau CRC calculé de 176 soit 0xB0'.

Le OU Exclusif avec le 3eme octet donne : 0xB0' XOR 0x38' = 0x88' soit l'index décimal de 136 qui donne dans la table: CRC = 0d78 ou 0x4E.

Le OU Exclusif avec le 4eme octet donne : 0x4E XOR 0x0C = 0x42 soit l'index décimal de 66 qui donne dans la table: CRC = 0d250 ou 0xFA.

Le OU Exclusif avec le 5eme octet donne : 0xFA XOR 0x04 = 0xFE soit l'index décimal de 254 qui donne dans la table: CRC = 0d107 ou 0x6B.

Le OU Exclusif avec le 6eme octet donne : 0x6B XOR 0x00 = 0x6B soit l'index décimal de 107 qui donne dans la table: CRC = 0d69 ou 0x45.

Le OU Exclusif avec le 7eme octet donne : 0x45 XOR 0x00= 0x45 soit l'index décimal de 69 qui donne dans la table: CRC = 0d121 ou 0x79.

Le CRC des 7 octets reçu est donc 0x79 qu'il faudra comparer au 8eme octet que l'on va recevoir.

Soit fcrc(x) la fonction qui lit la valeur dans la table a la position x

```
CRC= (00 xor LSB); /* Famille*/  
CRC= fcrc(CRC) xor O2; /*Octet2*/  
CRC= fcrc(CRC) xor O3; /*Octet3*/  
CRC= fcrc(CRC) xor O4; /*Octet4*/  
CRC= fcrc (CRC)xor O5; /*Octet5*/  
CRC= fcrc (CRC)xor O6; /*Octet6*/  
CRC= fcrc (CRC)xor O7; /*Octet7*/
```

FISA S3E A4 Réseaux de TERRAINS/BUS 1 WIRE/ 5.5 Liste des composants 1-WIRE

(Les + utilisés)

Code	iButton	Famille	Description
2401	1990A	1	Numéro de série 1-Wire
1425	1991	2	Mémoire (en bloc) à accès protégé par mot de passe (1 153 bits)
2404	1994	4	Horloge temps-réel, timer et NVRAM (4 ko)
2413		5	Interrupteur ou détecteur d'état (d'une double bascule)
	1993	6	NVRAM (4 ko)
	1992	8	NVRAM (1 ko)
2502	1982	9	NVRAM (1 ko)
	1995	0A	NVRAM (16 ko)
2505	1985	0B	EPROM (16 ko)
	1996	0C	NVRAM (64 à 256 ko)
2506	1986	0F	EPROM (64 ko)
18S20	1920	10	Thermomètre
2406, 2407		12	Double bascule et EPROM d'1 ko
2430A	1971	14	EPPROM 256 bits et registre OTP (mémoire morte) 64 bits
	1963S	18	NVRAM (4) et calcul SHA-1
	1963L	1A	NVRAM (4 ko) et compteur d'écritures
	28E04-100	1C	EEPROM 4 096 bits avec PIO
2423		1D	NVRAM (4 ko) et compteur d'impulsions

Code	iButton	Famille	Description
24011990A			1Numéro de série 1-Wire
1425	1991		Mémoire (en bloc) à accès protégé par mot de passe (1 153 bits)
2404	1994	4	Horloge temps-réel, timer et NVRAM (4 ko)
2409	1F		Coupleur deux canaux (obsolète le 14/10/2019)
	1921		Enregistreur de températures Thermocron
1822		21	Thermomètre digital
2433	1973		EEPROM (4 ko)
2415	1904	24	Horloge temps réel (RTC)
2438		26	Capteur de température, convertisseur A/N
2417		27	Horloge temps réel avec interrupteur
18B20		28	Capteur de température avec résolution ajustable
2408		29	Bascule adressable 8 canaux
2890	2C		Potentiomètre numérique, 1 canal
2431	2D		EEPROM (1 ko)
2760		30	Capteur de température, courant et convertisseur A/N
24321961S		33	EEPROM et calcul SHA-1

Code	iButton	Famille	Description
	1977	37	EEPROM 32 ko protégée par mot de passe
	24133A		Bascule adressable, deux canaux
	1922, 1923	41	Enregistreur de mesures (température et humidité)
	1981	91	EPROM 512 bits
2480			Driver de ligne série vers 1-Wire

5.6 Bibliographie et sources

1 WIRE D. MENESPLIER 2004

FISA S3E A4 Réseaux de TERRAINS/BUS 1 WIRE/

5.7 Travaux pratiques sur mallette

Gestion d'un composant DALLAS 1 WIRE DS18B20

Objectifs du TP :

Concevoir un driver 1 WIRE

Matériel utilisé :

- Malette Modèle : MDET Bus &Réseaux Kits de formation pour l'apprentissage des bus de communication
- Un PC équipé du logiciel MPLABX
- Un oscilloscope numérique
- Un analyseur logique
- Documentation

Expérimentation :

Concevoir une application en C bas niveau qui permet : la gestion du capteur de température,

- Encoder les sous programmes suivant :

--char Lecture 1 WIRE (unsigned char *VAL, char * Dest);

- VAL pointeur sur la chaîne de caractère lue
- Dest Périphérique de destination
- Retour 0 lecture OK, 0xFF Erreur

--char Ecriture 1 WIRE (unsigned char VAL, char * Dest);

- VAL valeur à écrire,
- Dest Périphérique de destination

- Char CRC (char * trame1wire)

- Retourne le CRC du message

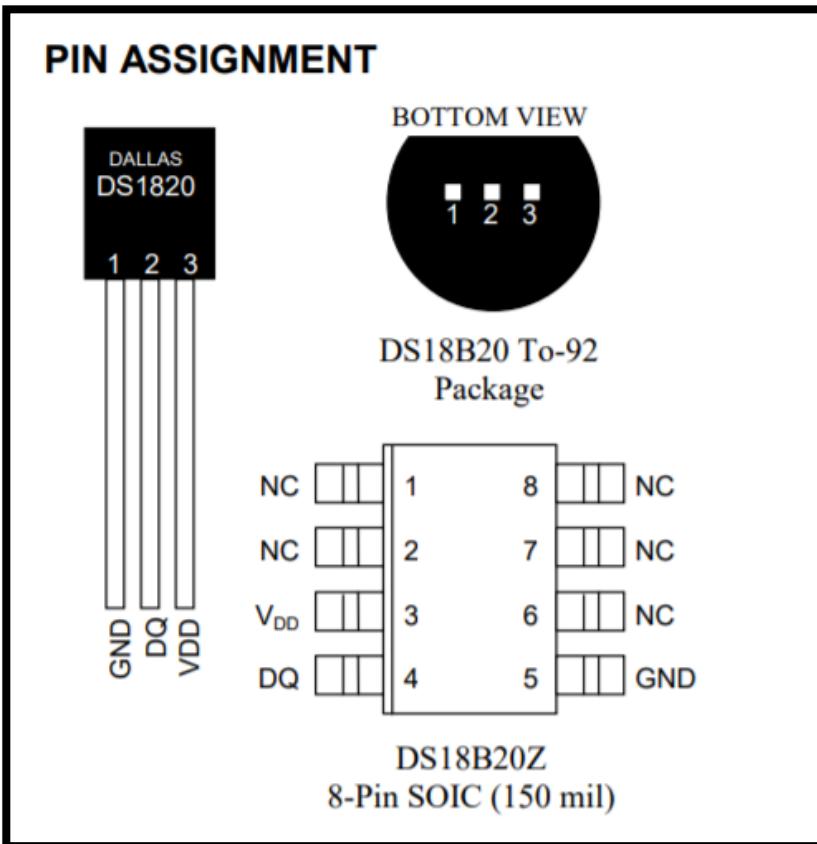
void Température (char x, char y, char * dest)

Affiche à l'écran toute les 30 s la valeur de température lue à l'emplacement x,y..

DS 18B20



www.dalsemi.com



DS18B20 Programmable Resolution 1-Wire® Digital Thermometer

FEATURES

- Unique 1-Wire interface requires only one port pin for communication
- Multidrop capability simplifies distributed temperature sensing applications
- Requires no external components
- Can be powered from data line. Power supply range is 3.0V to 5.5V
- Zero standby power required
- Measures temperatures from -55°C to +125°C. Fahrenheit equivalent is -67°F to +257°F
- ±0.5°C accuracy from -10°C to +85°C
- Thermometer resolution is programmable from 9 to 12 bits
- Converts 12-bit temperature to digital word in 750 ms (max.)
- User-definable, nonvolatile temperature alarm settings
- Alarm search command identifies and addresses devices whose temperature is outside of programmed limits (temperature alarm condition)
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system

CAPTEUR de TEMPERATURE DS 18B20

Ces circuits possèdent un code unique sur 64 bits comme tous les circuits 1 WIRE.

Le code famille est 0x28, suivi de 6 octets propre au circuit et d'un octet de CRC.

La détection de présence de ce circuit se fait en envoyant le pulse de Reset, qui est un état bas pendant au moins 480 µs.

Quand un circuit DS 18B20 est présent sur le bus il le signale en maintenant le bus à l'état bas pendant 60 à 240 µs.

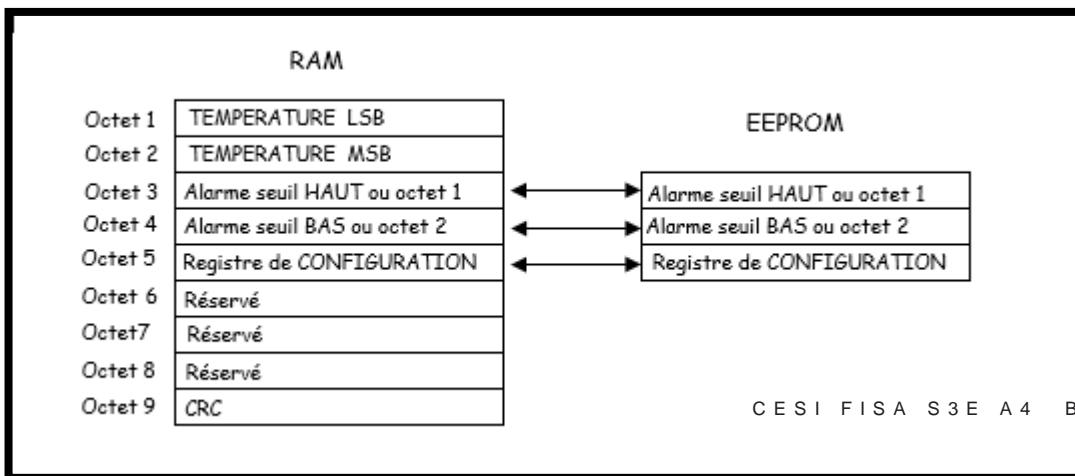
Toute transaction avec un tel circuit doit démarrer par un pulse de Reset suivi de l'envoi d'un commande ROM. On pourra après envoyer une commande de fonction propre à ce type de circuit.

Si le circuit est seul sur le bus 1 Wire, la commande ROM peut être l'appel général SKIP ROM = 0xCC. Si ce n'est pas le cas, il faudra connaître les 64 bits propre du circuit que l'on veut atteindre et utiliser la commande MATCH ROM = 0x55 suivi des 8 octets du code.

Une recherche préalable des 8 octets de code sera faite par la commande READ ROM = 0x33 si le circuit est seul ou bien par SEARCH ROM = 0xF0 s'il y a plusieurs circuits sur le bus.

MEMOIRE INTERNE:

Elle est constituée d'une zone RAM de 9 octets et d'une zone EEPROM non volatile de 3 octets.



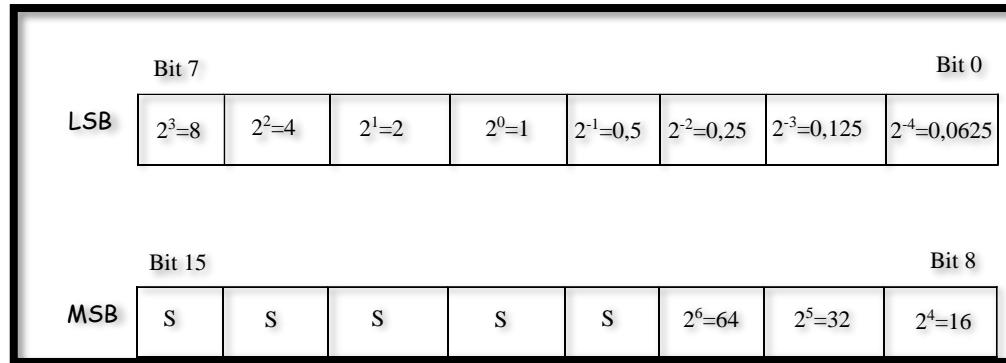
OCTET 1 et 2: TEMPERATURE LSB et MSB

La température est donnée sur 16 bits en complément à 2 entre -55°C et $+125^{\circ}\text{C}$.

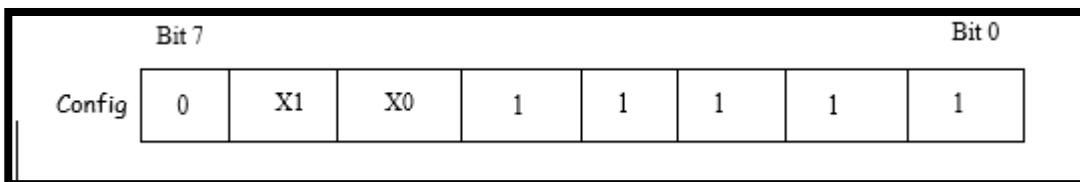
S = Signe de la température. Ce bit est à "1" si elle est négative et à "0" si elle est positive.

Si le signe est positif (bit S=0) la valeur absolue de la température sera donnée par les bits significatifs de LSB et MSB .

Par contre si la température est négative (bit S=1), la valeur absolue sera obtenue en complémentant la valeur des bits significatifs de LSB et MSB et en ajoutant 1 au résultat.



OCTET 5: Registre de CONFIGURATION



Seuls deux bits sont significatifs dans ce registre: X1 et X0. Ces bits permettent de choisir la résolution.

X 1	X 2	RESOLUTION	Temps conversion
0	0	9 bits	94 ms
0	1	10 bits	188 ms
1	0	11 bits	375 ms
1	1	12 bits	750 ms

En résolution 11 bits, le bit 0 n'est pas défini.

Le bit de poids min est le bit 1 de valeur : 0,125 °C.

En résolution 10 bits, les bit 0 et 1 ne sont pas définis.

Le bit de poids min est le bit 2 de valeur 0,25 °C.

En résolution 9 bits, les bit 0, 1 et 2 ne sont pas définis.

Le bit de poids min est le bit 3 de valeur 0,5 °C.

CODES de COMMANDES:

Après avoir envoyé une commande ROM pour adresser un DS18B20 esclave, le maître doit envoyer un code de commande.

Début de conversion: 0x44

Cette commande lance la conversion de température. Le résultat est rangé dans les 2 octets LSB et MSB . Le temps de conversion dépend de la résolution choisie.

Le maître doit interroger le DS18B20 qui répond par un bit à "0" tant que la conversion n'est pas terminée.

Quand l'opération est terminée, l'esclave répond par un bit à "1".

ECRITURE en RAM: 0x4E

Seuls les octets 3, 4 et 5 de la zone RAM peuvent être écrits. Il s'agit des octets: Alarme seuil haut, Alarme seuil bas et Configuration.

Le maître doit commencer par envoyer en premier le LSB de l'octet 3. Tous les octets seront ensuite envoyés avec le LSB en tête.

Il doit impérativement envoyer les 3 octets, avant de faire un reset, pour que l'écriture soit effective.

LECTURE de la RAM: 0xBE

Les 9 octets de la RAM sont envoyés vers le maître. L'esclave commence par le bit 0 du premier octet et transmet ainsi les 9 octets de sa RAM.

Le maître peut interrompre à tout moment la lecture en faisant un Reset.

COPIE RAM en EEPROM: 0x48

Copie des octets 3, 4 et 5 de la zone RAM dans la zone EEPROM pour sauvegarde en cas de coupure d'alimentation. RECOPIE EEPROM en RAM: 0xB8

Cette commande récupère en EEPROM les octets Alarme seuil haut, Alarme seuil bas et Configuration pour les placer en RAM dans les octets 3, 4 et 5.

ALIMENTATION PARASITE: 0xB4

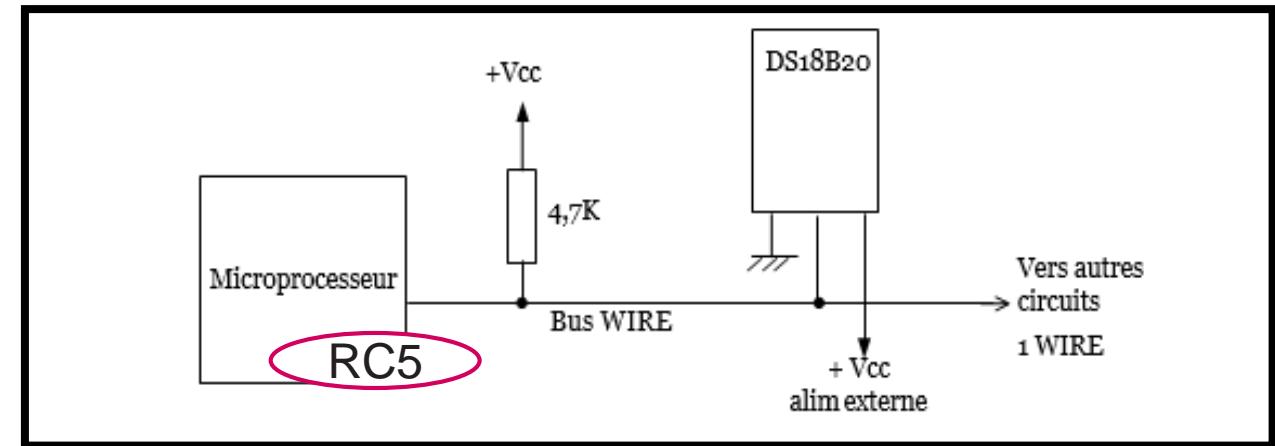
L'esclave répond à cette commande par un bit à "1" s'il fonctionne avec une alimentation extérieure, c'est à dire sur 3 fils. Et par le bit à "0" s'il est en mode d'alimentation parasite, sur 2 fils.

ALIMENTATION du DS 18B20:

Il y a 2 méthodes pour alimenter ce circuit.

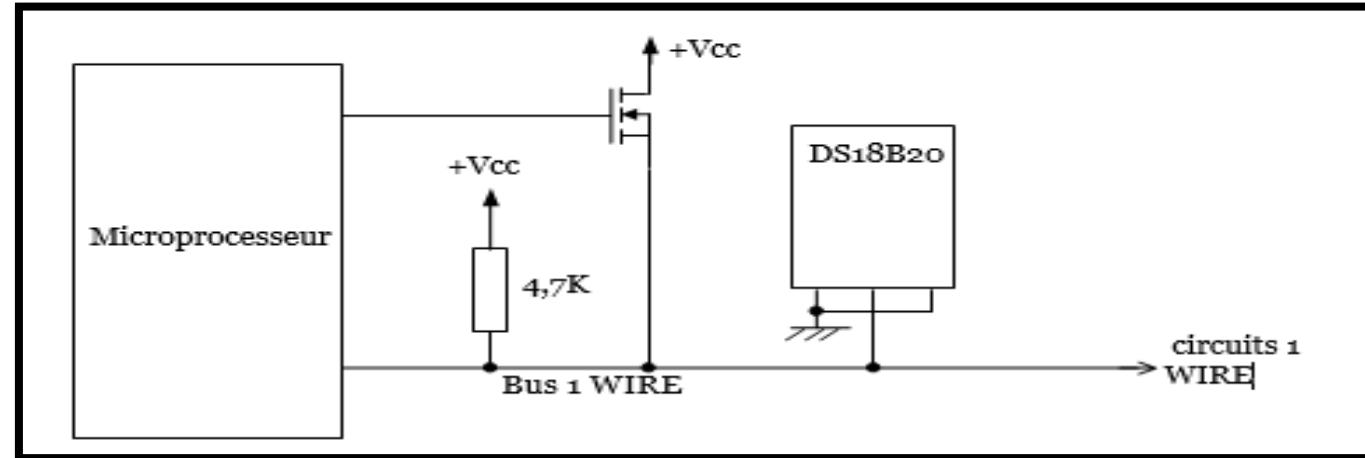
- Soit par une alimentation extérieure entre 3V et 5V :

Dans le cas du TP on utilise
RC5 et la broche



ALIMENTATION du DS 18B20:

Soit par une alimentation parasite : 1 WIRE



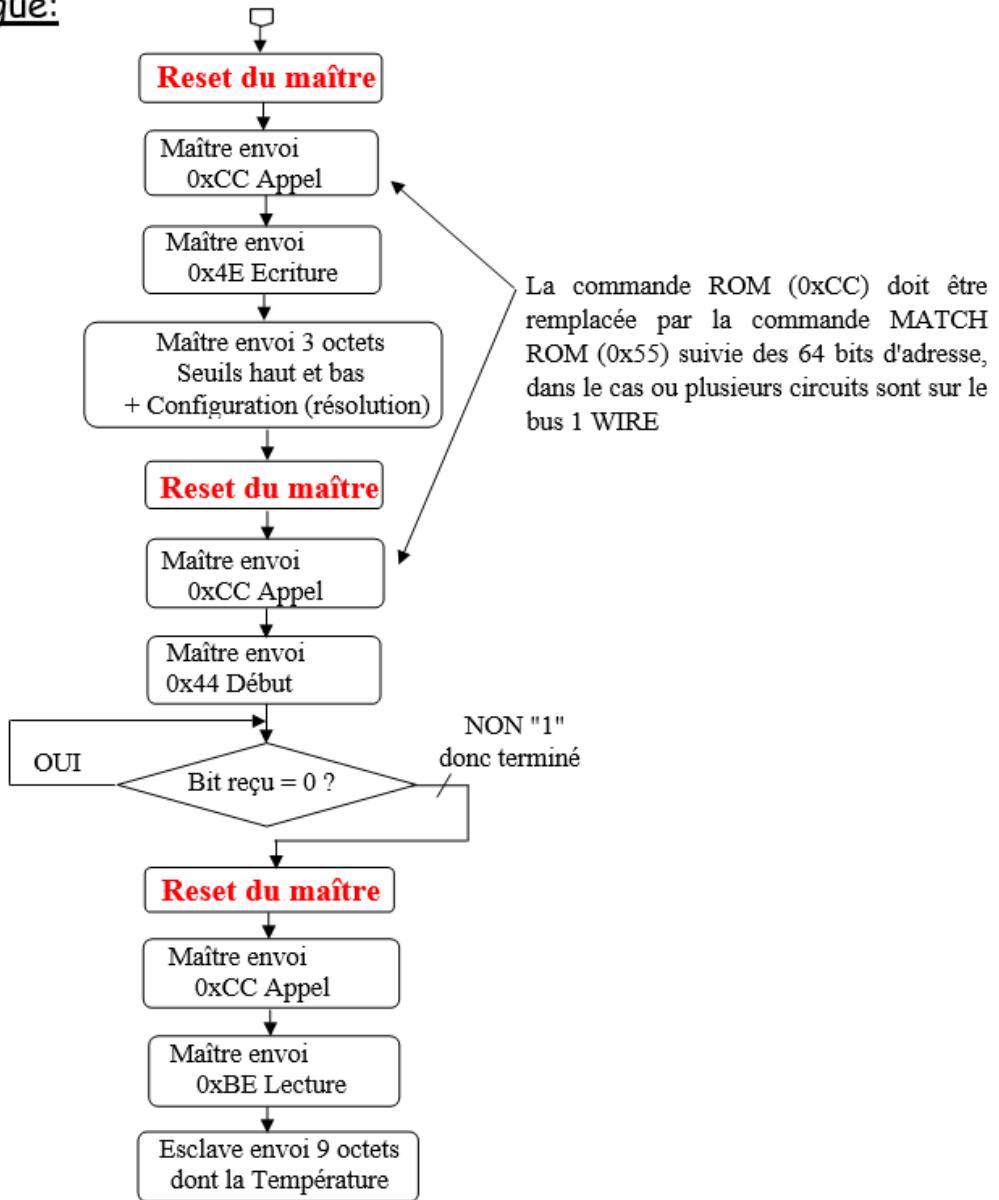
Seulement, quand le circuit est en mode conversion ou en mode écriture en EEPROM, le courant d'alimentation doit être d'au moins 1,5mA. Dans ce cas l'alimentation parasite ne suffit plus.

Il faut pour cela mettre le bus 1 WIRE directement au +Vcc. C'est ce qui sera fait par la mise en conduction du transistor MOSFET.

Cette opération devra intervenir dans les 10 µs qui suivent l'envoi de la commande "Conversion Température" ou "Copie RAM en EEPROM".

Ensuite le bus devra rester au +Vcc pendant toute la durée de la conversion de température (750 ms) ou de l'écriture en EEPROM (10ms). Aucune autre activité sur le bus 1 WIRE ne peut avoir lieu pendant ce temps là, à la différence du mode d'alimentation extérieure, où ce temps peut être mis à profit pour envoyer d'autres données sur le bus.

Exemple de dialogue:





6

FISA S3E
Réseau de terrain
Bus ARINC 429

FISA S3E A4 Réseaux de TERRAINS/BUS ARIN 429/

6 Résumé

Catégorie : BUS AVIONIQUE

Exemple d'utilisation: Airbus A310,... A340, Boeing du 727 au 767

Avantages

Il est déterministe puisque n'ayant qu'un seul émetteur par bus, il n'y a aucun risque de collision.

Inconvénients

Dans l'avionique moderne, il est plus que limité par son faible débit et ses possibilités d'adressage (labels),

Il n'y a pas de somme de contrôle (hormis la parité) permettant de vérifier l'intégrité de données,

Débit: Deux débits sont normalisés:

- * Lo speed 12.5 kbits/seconde.
- High speed 100 kbits/seconde.

Support physique: Paires torsadées. Codage RZ Bipolaire avec retour à zéro.

Niveaux de tension

Niveaux logique RZ : 1 , NULL, 0

Niveau	Ligne A <> Ligne B Côté émetteur	Ligne A <> Ligne B Côté récepteur
HIGH	+10.0 V ± 1.0 V	de +6.5 à +13 V
NULL	0 V ± 0.5V	de +2.5 à -2.5 V
LOW	-10.0 V ± 1.0 V	de -6.5 à -13 V



Le mot ARInc 429 est codé en 32 bits et utilise un format incluant 5 champs primaires, en général :

- 8 bits pour le label (nature de l'information)
- le SDI (Source / Destination Identifier)
- Data Field (le champ de données)
- le SSM (Sign / Status Matrix)
- un bit de parité (Odd parity bit)



Le LABEL

La particularité du label est son sens de lecture: il est transmis en sens inverse sur le bus par rapport aux autres champs du mot ARInc.

Le schéma ci-dessous donne la valeur maximale que peut atteindre le label:²

Bit	1	2	3	4	5	6	7	8
Codage binaire	1	1	1	1	1	1	1	1
Codage octal	3		7		7			

Le bit de poids faible est transmis en premier. Le LABEL est transmis en entête du mot, soit les huit premiers bits. On a donc, sur le bus ARINC, l'ordre de transmission des 32 bits suivant :

8, 7, 6, 5, 4, 3, 2, 1, 9, 10, 11, 12, 13 ... 32.

Sommaire



6 FISA S3E A4 Réseaux de TERRAINS/BUS ARINC 429

• 6.1 Historique	103
• 6.2 Support	104
• 6.3 Encodage des bits	105
• 6.4 Niveaux	105
• 6.5 Temps et vitesse	105
• 6.6 Avantages/Inconvénients / BUS ARINC 429	114
• 6.7 Travaux pratiques sur mallette	115



FISA S3E A4 Réseaux de TERRAINS/ BUS ARINC 429/

6.1 Historique

L'avionique est l'électronique appliquée aux techniques aéronautiques et spatiales.

ARINC 429 :

Description :

L'ARINC 429 est un des plus anciens bus avionique. Développé par *l'Aeronautical Radio INCorporation* en 1977, il est encore utilisé aujourd'hui sur des nouvelles plates-formes même si d'autres bus plus récents sont plus fréquemment retenus.

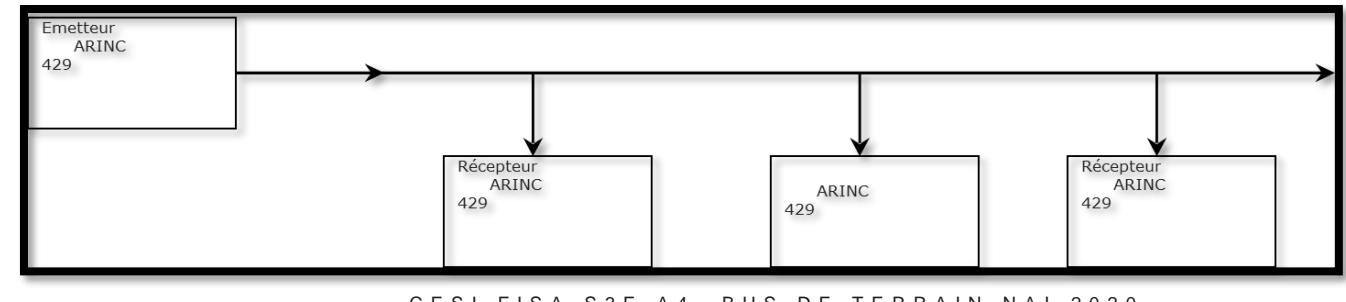
Ce bus est un bus de données simple utilisant un seul émetteur et de 1 à 20 récepteurs par bus.

On le retrouve dans des avions tels que les Airbus A310 /A340 et dans de nombreux autres systèmes avioniques. Les normes ARInc sont divisées en quatre familles : ARInc 400, 500, 600 et 700.

Les normes les plus utilisées sont : ARInc 419, 429, 575, 615, 629.

La majorité des équipements avioniques utilisent l'ARInc 429 pour leur bus de données.

Actuellement, l'Airbus A340 et le Boeing 777 utilisent l'ARInc 629 (version améliorée de l'ARInc 429).



FISA S3E A4 Réseaux de TERRAINS/ BUS ARINC 429/

6.2 Support

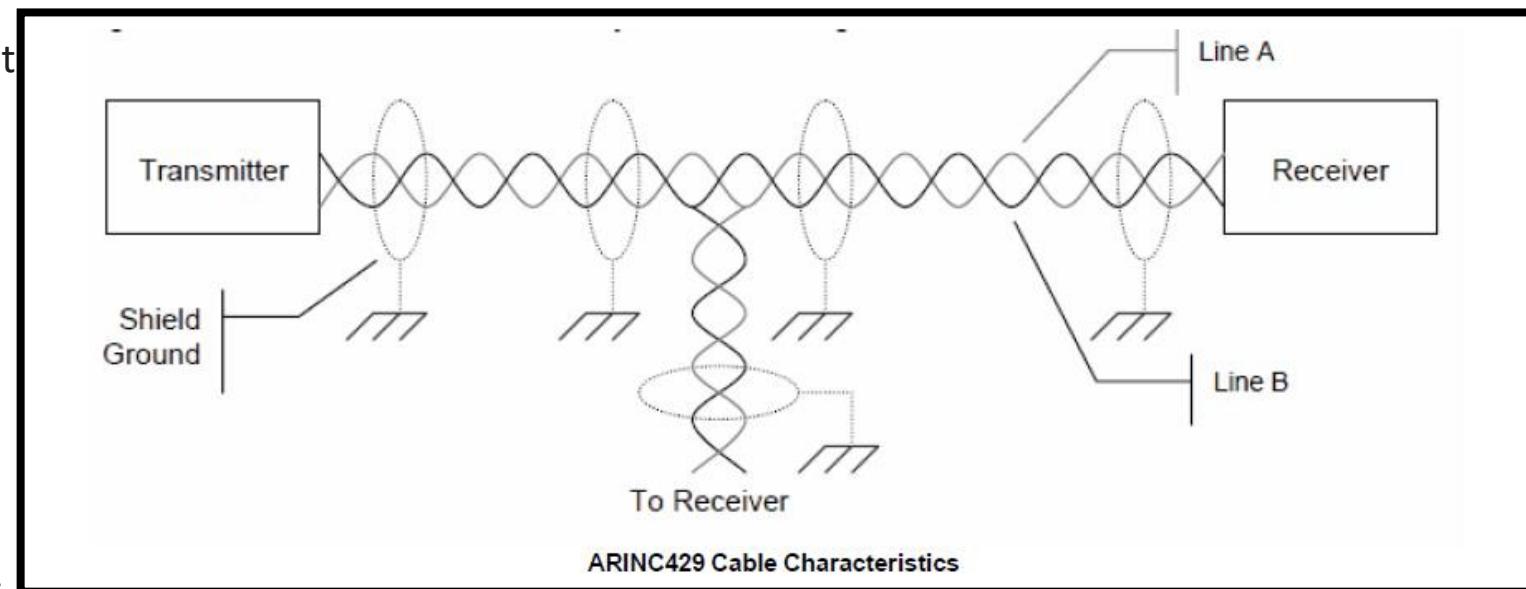
La connexion est réalisée par l'intermédiaire d'une paire torsadée blindée composée de deux brins (ou lignes) « A » et « B ».

Le transfert des données est réalisé en différentiel entre les deux lignes de la paire.

L'impédance de l'émetteur doit être maintenue en permanence à $75 \pm 5 \Omega$ quel que soit le niveau HIGH, LOW ou NULL, divisée de manière égale entre les deux lignes de la paire. Cette valeur a été choisie pour être approximativement égale à l'impédance caractéristique de la paire blindée.

La résistance de chaque récepteur, entre les deux lignes (« A » et « B ») et mais aussi entre chacune des lignes et la masse, doit être maintenue supérieure à $12 \text{ k}\Omega$. Idem pour la capacitance, qui doit être inférieure à 50 pF entre les deux lignes, mais aussi entre chacune des lignes et la masse.

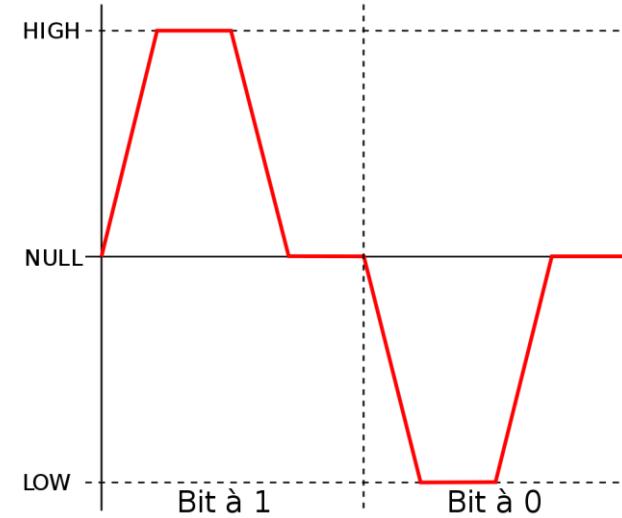
La résistance totale des récept



FISA S3E A4 Réseaux de TERRAINS/ BUS ARINC 429/

6.3 Encodage des bits

L'encodage utilisé est de type bipolaire avec retour à 0 répondant à la forme suivante : RZ



6.4 Niveaux

Les trois niveaux utilisés pour l'encodage entre les deux brins (A et B) de la paire blindée sont

Niveau	Ligne A <=> Ligne B Côté émetteur	Ligne A <=> Ligne B Côté récepteur
HIGH	+10.0 V ± 1.0 V	de +6.5 à +13 V
NULL	0 V ± 0.5V	de +2.5 à -2.5 V
LOW	-10.0 V ± 1.0 V	de -6.5 à -13 V

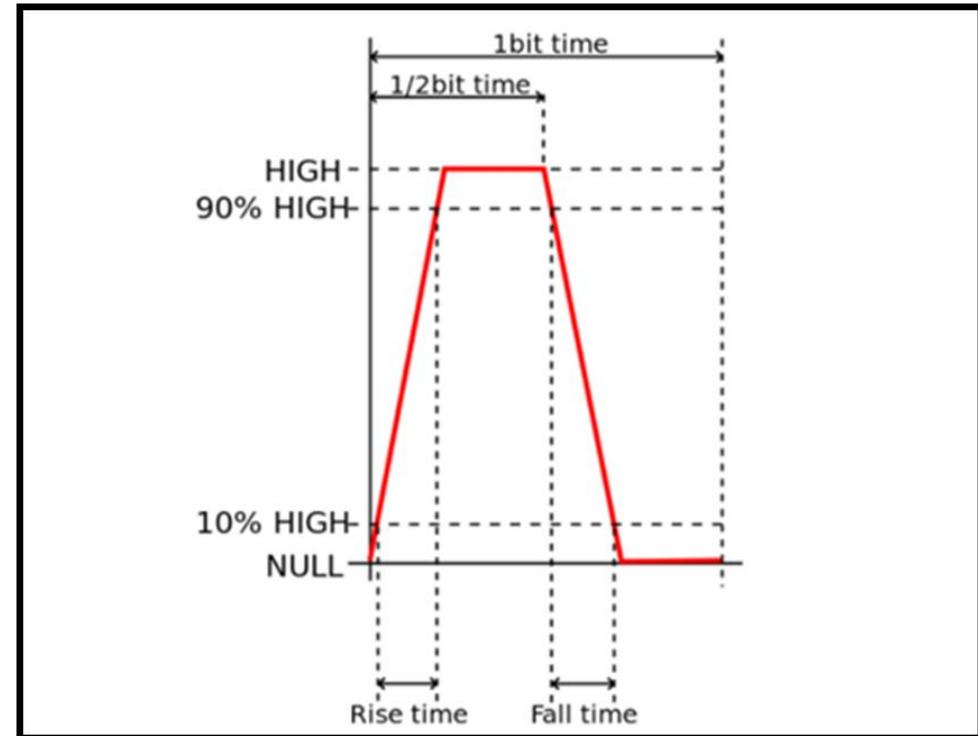
Lorsque l'émetteur ne transmet rien, il place le bus à l'état NULL.

6.5 Temps et vitesse

Il existe deux vitesses de transmission 12,5 kbit/s (Lo SPEED) et 100 kbit/s (Hight SPEED). Pour

chacune de ces vitesses, la norme impose des temps caractéristiques :

- Le taux de transfert (Bit rate),
- la durée d'un bit (1bit time),
- la durée d'un demi-bit (1/2bit time),
- Le temps de montée (Rise time),
- le temps de descente (Fall time).



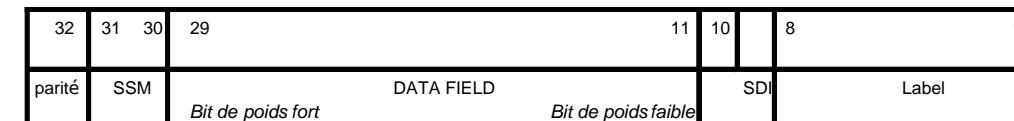
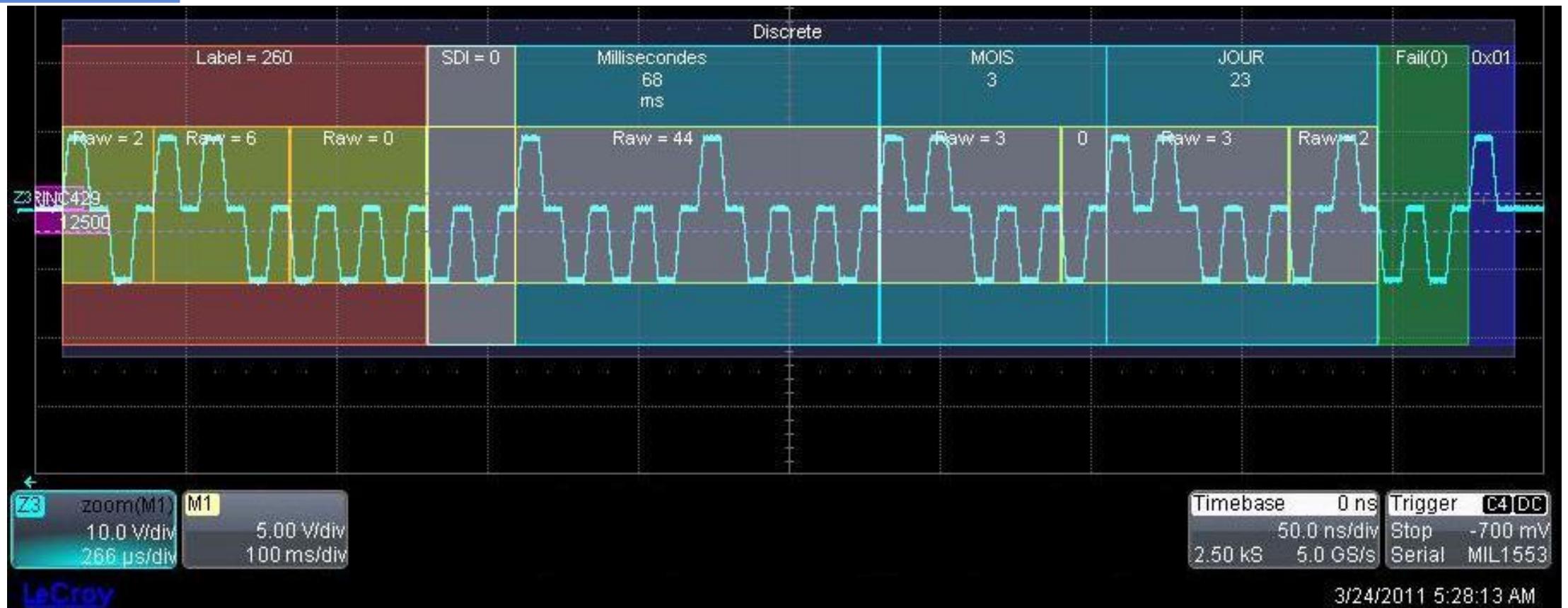
Bit rate	1bit time	1/2bit time	Rise time	Fall time
100kbits/s ± 1 %	10µs ± 0,25µs	5µs ± 0,25µs	1,5µs ± 0,5µs	1,5µs ± 0,5µs
12,5kbits/s ± 1 %	80µs ± 2µs	40µs ± 2µs	10µs ± 5µs	10µs ± 5µs

Exemple d'un mot

L'image ci-dessous montre un mot ARINC 429, produit par exemple par un générateur NAV429 23, et capturé à l'aide d'un oscilloscope.

Dans ce cas particulier, les bits 11 à 29 (données) contiennent des jours, des mois et des millisecondes.

Un oscilloscope restitue un signal dans l'ordre chronologique d'apparition de l'information. Ceci explique pourquoi le label est au début de la transmission. (Voir Ordre de transmission des bits).



FISA S3E A4 Réseaux de TERRAINS/ BUS ARINC 429/

6.5 Composition d'un mot ARINC 429

Le mot ARINC 429 est codé en 32 bits et utilise un format incluant 5 champs primaires, en général :

- 8 bits pour le label (nature de l'information)
- le SDI (*Source / Destination Identifier*)
- *Data Field* (le champ de données)
- le SSM (*Sign / Status Matrix*)
- un bit de parité (*Odd parity bit*)



Trames

Les trames sont composées de plusieurs mots de 32 bits espacés par au moins quatre bits à l'état NULL (utiles pour la synchronisation sur le mot suivant).

L'ordre des mots n'est pas imposé

Schéma d'organisation d'un mot.

D'autre part, les données du Data Field et du label peuvent être codés de deux manières différentes, fonction de la nature de l'information à envoyer et du récepteur. Ces deux formats sont le BCD et le BNR.

6.5.1 Le Label

Le label est une partie importante du mot **ARInc 429** car il conditionne le format du Data Field et joue un rôle d'identification en combinaison avec le SDI.

Tous les récepteurs LRU sont programmés pour ne retenir que les données nécessaires à leurs propres opérations.

Pour cela, chacun d'eux identifie les 8 premiers bits du mot ARInc, **le label**, codé en octal, représentant soit le type d'information contenu dans le mot, soit le rôle du mot pour la maintenance (mode auto test), ou bien le mode de transfert des données.

Il peut aussi contenir des instructions ou des fonctions de report d'informations.

La particularité du label est son sens de lecture: il est transmis en sens inverse sur le bus par rapport aux autres champs du mot **ARInc**.

Le schéma ci-dessous donne la valeur maximale que peut atteindre le label:

Bit	1 MSB	2	3	4	5	6	7	8 LSB
Codage binaire	1	1	1	1	1	1	1	1
Codage octal	3		7		7		7	



P :	bit de parité (impaire)	bit 32.
SSM :	Sign/Status Matrix	bit 31 (MSB) et bit30 (LSB).
Données:		bits 11 à 29.
SDI :	Source/Destination Identifier	bit 10 (MSB) et bit 9 (LSB).
LABEL :	identifiant des données ,	MSB bit 1 à LSB bit8.

De par la simplicité de sa topologie et des protocoles utilisés, ce bus est d'une très grande fiabilité. Et comme il n'y a qu'un seul émetteur par paire de fils, l'ARINC 429 est bien évidemment déterministe.

6.5.2 Ordre de transmission:

Le bit de poids faible est transmis en premier. Le LABEL est transmis en entête du mot, soit les huit premiers bits. On a donc, sur le bus ARINC, l'ordre de transmission des 32 bits suivant :

8, 7, 6, 5, 4, 3, 2, 1, 9, 10, 11, 12, 13 ... 32.

6.5.3 Formats d'encodage de l'information : BCD / BNR

Toutes les données **ARInc** sont transmises dans les 32 bits du mot, mais les données peuvent être codées différemment :

- BCD (Binary Coded Decimal)
- BNR (le complément par deux de la notation binaire) => utilisé pour la transmission d'altitude.
- Données discrètes
- Données de maintenance
- Caractères alphabet ISO #5 (ASCII)

Encodage du Data Field en BNR

Le BNR est aussi un format standard du Data Field en ARInc 429.

Ce type de codage écrit la donnée comme un nombre binaire, c'est le même format que sur les ordinateurs modernes.

Format standard BNR

Le bit 29 est le bit du signe et le bit 28 est le bit le plus important du champ de données, il représente la moitié de la valeur maximum des paramètres définis.

Les bits successifs représentent l'incrément d'une série de fraction binaire.

Si le bit 29 est « 1 » le signe est **négatif (ou Sud, Ouest, Gauche, De, Sous)**, si c'est « 0 » le signe est **positif (ou Nord, Est, Droite, vers, au dessus)**.

Exemple : La figure ci-dessous montre un encodage en BNR, le message utilise le label 103 qui correspond à la vitesse de l'air.

32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	1
P	SSM		Data																		SDI		LABEL		
0	1	1	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	103	

Exemple d'encodage en BNR

En se référant aux spécifications de l'ARInc 429, on sait que le facteur d'échelle pour cet exemple est de 512 (29) et 10 bits sont utilisés (bits 29 au 20).

Un zéro dans le bit 29 montre que le signe de la valeur de la vitesse de l'air est positive.

La valeur numérique transmise est obtenue en multipliant le facteur d'échelle, déterminé par le type de donnée associée au label, par le ratio indiqué par chaque bit successif et en additionnant l'ensemble des valeurs obtenues.

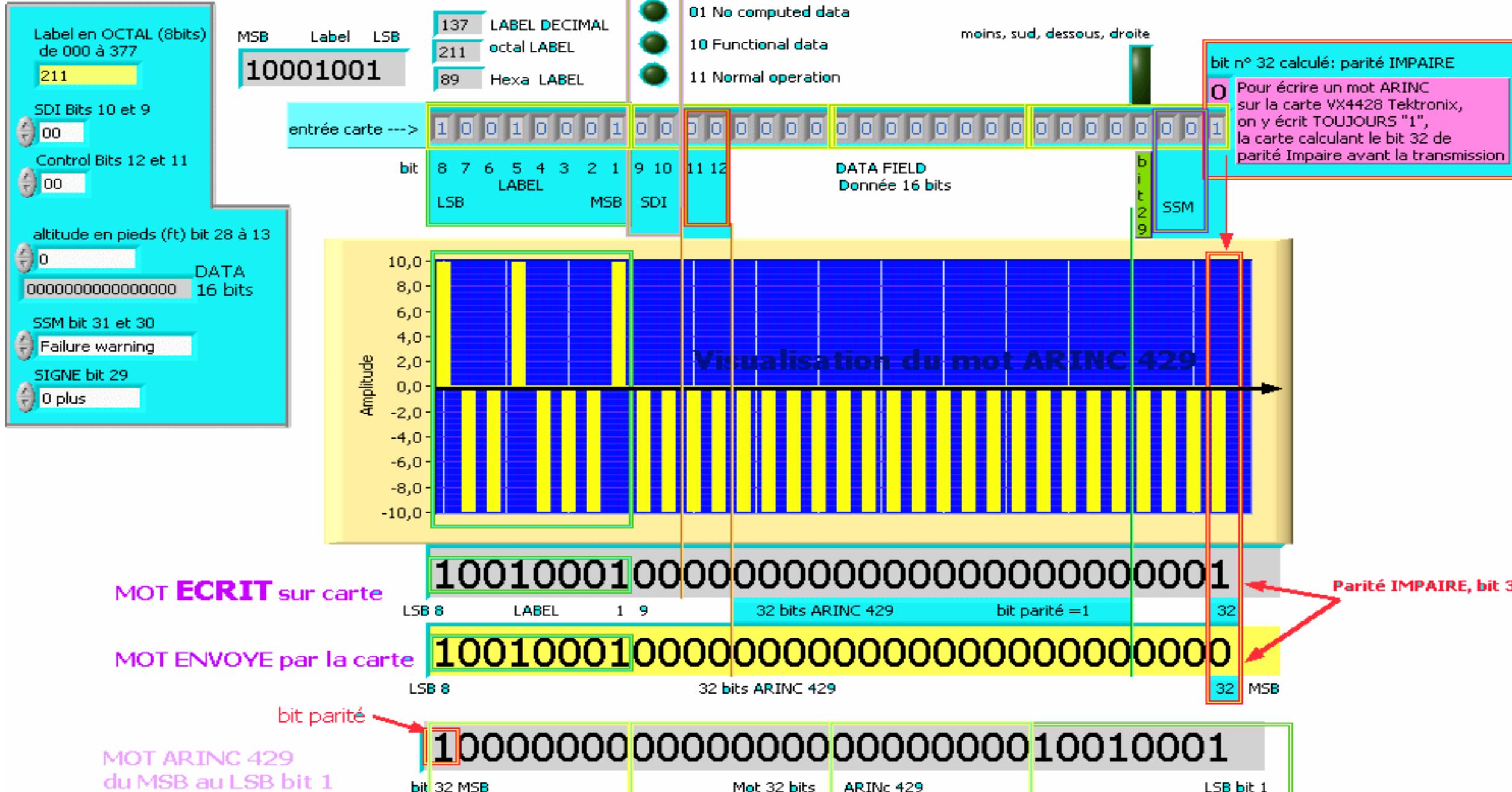
Le bit 28 est la moitié du facteur d'échelle, le bit 27 et le quart du facteur d'échelle, le bit 26 est le huitième du facteur d'échelle et ainsi de suite.

$$\text{Vitesse de l'air} = 512 \cdot \frac{1}{2} + 512 \cdot \frac{1}{64} + 512 \cdot \frac{1}{128} = 268 \text{KNotes}$$

Le nœud (symbole kn1, kta ou nd) est une unité de mesure de la vitesse utilisée en navigation maritime et aérienne. Un nœud est égal à un mille marin par heure, soit 1,852 km/h

Détermination d'un mot 32 bits ARINC 429 pour altimètre (16 bits - résolution 1 pieds)

pour AIRBUS A300-600



Example: FREQUENCY = 109.30 MHz

Exemple de codage des mots de 32 bits :
 Frequency control word bit assignments

			en BCD																en OCTAL				en OCTAL										
5	3	P SSM	10 MHz				1 MHz				0.1 MHz				0.01 MHz				1	2	4	L S B	en OCTAL										
32	31	30	29	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		
			4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	0	0	0	0	0	0	1	1	0	1	1	0	0	0		
			1	0	0	0	1	0	0	1	0	0	1	1	0	0	1	1	0	0	0	0	0	0	1	1	0	1	1	0	0	0	
			0	9	3													3	3	0													

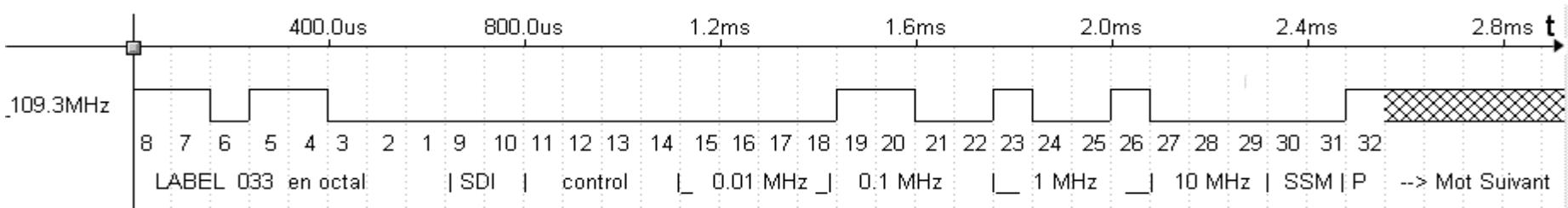
Note :

- 1 bits 13 et 14 reserved for control functions
- 2 used to define category of ground facilities (installations)
- 3 Sign Status Matrix (donnée de contrôle) defined by following table

	Bit 31	Bit 30
Normal operation	0	0
No computed data	0	1
Functional test	1	0
Undefined	1	1

4 Source destination identifier : 2 bits.

5 P : parity bit



6.6 Avantages/Inconvénients / BUS ARINC 429

Avantages

La topologie est simple (facile d'approche, fiable, etc.).

Il est déterministe puisque n'ayant qu'un seul émetteur par bus, il n'y a aucun risque de collision.

Inconvénients

Le nombre et donc le poids des nombreuses paires de fils à installer est important,

Dans l'avionique moderne, il est plus que limité par son faible débit et ses possibilités d'adressage (labels),

Il n'y a pas de somme de contrôle (hormis la parité) permettant de vérifier l'intégrité de données,

Le récepteur ne peut pas acquitter la bonne réception des données, hormis si on met en place un second bus en retour.

FISA S3E A4 Réseaux de TERRAINS/BUS ARINC 429/

6.7 Travaux pratiques sur mallette

Gestion d'un composant DALLAS 1 WIRE DS18B20

Objectifs du TP :

Concevoir un Analiseur de trame ARINC 429

Matériel utilisé :

- Malette Modèle : MDET Bus & Réseaux Kits de formation pour l'apprentissage des bus de communication
- Un PC équipé du logiciel MPLABX
- Un oscilloscope numérique
- Un analyseur logique
- Documentation
- Un convertisseur Arinc 429 / Rs232 via Usb

Expérimentation n°1:

Visualiser un trame ARINC 429 à l'oscilloscope ou analyseur logique et la décoder à la main en identifiant les champs de la trames.

On émet la fréquence de communication soit :

101,6 MHZ (Label Octal 033, Normal Opération)

Expérimentation n°2: Envoi d'une trame ARINC depuis le PIC

Sur l'interface ARINC 429 / USB, reception de trame depuis le PC sous Visual Basic:

Utiliser l'interface ARINC 429 – USB pour transférer les trames ARINC 429 . Les trames sont convertie par le boitier ARINC 429 USB: Serie.

On développera coté PC un outil de lecture / écriture sur le port série de données Arinc 429 en représentant graphiquement les données émises et reçues.



CAMPUS D'ENSEIGNEMENT SUPÉRIEUR ET DE FORMATION PROFESSIONNELLE



Merci pour votre
attention.

Nicolas ANTINI

Stéphane DUVAL