LIAISON I²C

1 . Présentation

Le soucis des concepteurs du bus I²C était de transmettre des informations à un grand nombre de récepteurs en réduisant le câblage.

La réduction du câblage présente un certain nombre d'avantages :

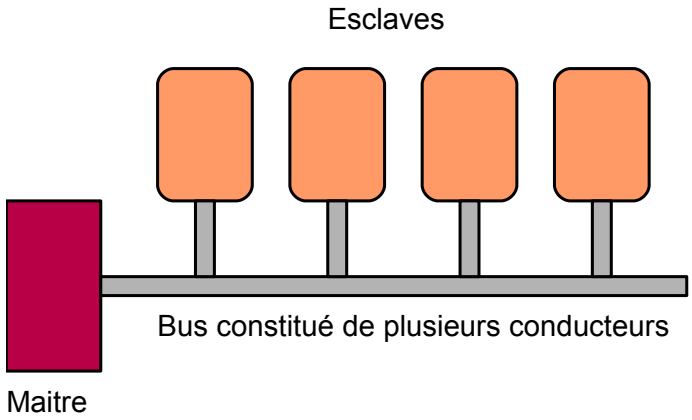
- diminution de la main d'œuvre et des risques de mal-façon
- réduction du coût de la matière première
- allègement et réduction des dimensions de l'appareil

En contre partie, il a fallu développer des composants spécifiques capables de communiquer grâce au bus I²C.

2 . Principe d'une communication par bus

Un bus est un ensemble de conducteurs portant des informations à plusieurs composants. Chaque composant est connecté en dérivation sur le bus. Cette disposition entraine les contraintes suivantes :

- chaque récepteur doit être capable de reconnaître que l'émetteur s'adresse à lui
- le court-circuit entre les broches similaires de deux composants différents doit être impossible
- la création d'un protocole précis

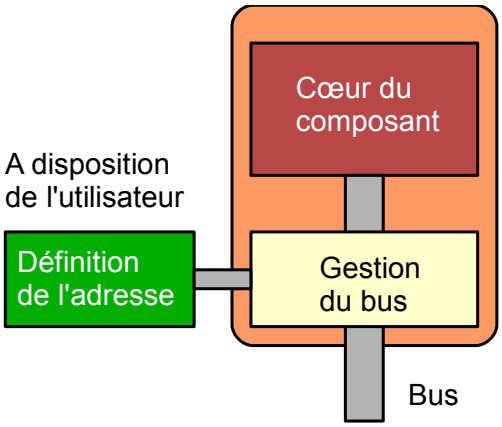


la communication par bus I²C met en œuvre deux types de composants,

- le maitre, qui est à l'initiative de tout transfert
- l'esclave, qui répond aux demandes du maitre.

Chaque type composant peut émettre ou recevoir des informations. Il ne peut y avoir qu'un seul émetteur et qu'un seul récepteur actif à un instant donné.

3 . Schéma fonctionnel d'un composant I²C



Le cœur du composant réalise la fonction principale c'est à dire la raison d'être du composant.

La gestion du bus s'occupe de la communication entre le maître et le cœur du composant. C'est la fonction qui connait le protocole I²C. Il s'agit de :

- reconnaître le début et la fin d'une transmission d'informations
- reconnaître que le maitre s'adresse ou non au composant
- d'extraire l'information utile et de la transmettre au cœur
- de réaliser l'opération inverse en cas de lecture

Chaque composant branché sur le bus I²C doit posséder une adresse unique. C'est le rôle du dispositif de définition de l'adresse.

4 . Principe de la définition de l'adresse du composant.

La définition complète de l'adresse se fait en trois parties :

- une partie interne figée par construction, qui correspond au type du composant, elle ne dépend pas du constructeur
- une partie externe laissée à la disposition de l'utilisateur
- la définition du sens de l'échange, écriture ou lecture.

4.1 Constitution générale de l'adresse du composant

F3	F2	F1	F0	A2	A1	A0	Sens

F3 F2 F1 F0 : partie figée par construction

A2 A1 A0 : partie laissée à la disposition de l'utilisateur (voir ci-dessous)

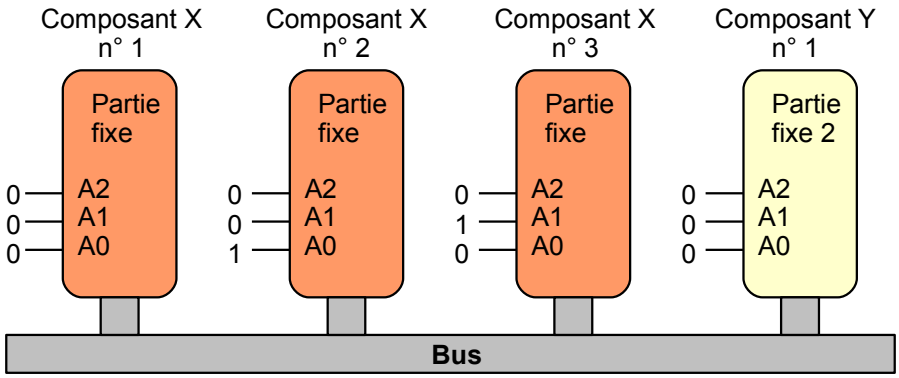
Sens : bit de sens du transfert (0 = écriture 1 = lecture)

Particularité : les documentations présentent l'adresse du composant sans le bit de sens. Il faut donc manipuler cette adresse avant de l'utiliser dans un échange. (Voir un exemple d'établissement de l'octet d'adresse complet § 6.1)

4.2 Plusieurs composants identiques sur le même bus

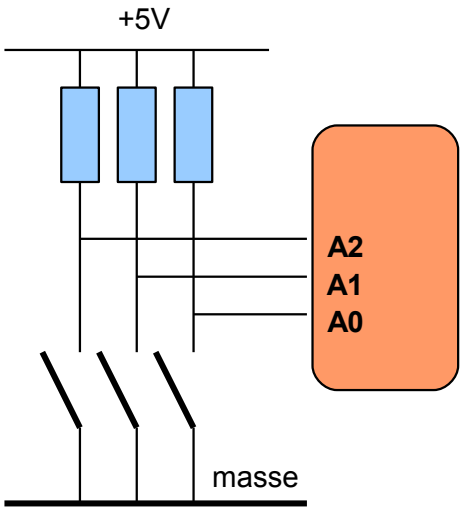
Le protocole I²C accepte plusieurs composants identiques sur le même bus. La différentiation se fait par les valeurs A2 A1 A0.

Les composants de type différents sont reconnus, en premier lieu, par la partie fixe de leur adresse.



4.3 Principe de l'intervention de l'utilisateur

Le composant possède trois broches que l'utilisateur peut porter au 0 logique ou au 1 logique pour compléter l'adresse.



Le dispositif se compose de trois résistors de rappel, et de trois interrupteurs.

Si l'adresse est figée, le dispositif peut être plus simple, il suffit de relier directement les broches A0, A1, A2 au niveau logique voulu (0 ou 5 V)

4.4 Adresse d'une cellule interne au composant.

Un composant peut être constitué de une cellule à plusieurs milliers de cellules internes, que l'on appelle parfois registres. Par exemple le PCF 8574 ne comporte qu'un seul registre interne alors que la mémoire 24LC32 comporte 4096 cellules mémoire.

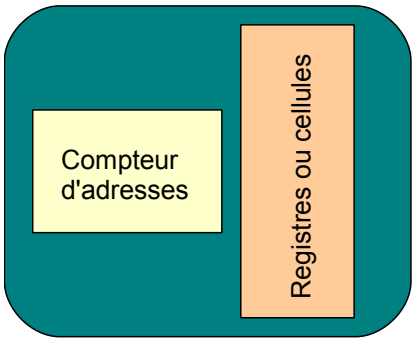
Une fois le composant sélectionné comme indiqué ci-dessus, il faut choisir le registre ou la cellule auquel on souhaite d'adresser.

Le choix se fait par l'écriture de un ou plusieurs octets à la suite de l'écriture de l'octet de sélection du composant.

Le dialogue avec la partie interne du composant peut se faire de manière :

- ciblée, on s'adresse à un seul registre ou à une seule cellule (random access)
- séquentielle, on s'adresse à des registres ou des cellules adjacents.

Schéma de la constitution interne d'un composant comprenant plusieurs registres.



Composant

Le schéma montre

- l'ensemble des registres ou des cellules
- le compteur des adresses.

Le protocole de communication indique qu'il faut d'abord sélectionner le composant puis on peut choisir une cellule interne comme le montrent les exemples ci-dessous.

Si on souhaite s'adresser à des registres consécutifs, il suffit de répéter l'opération de lecture ou d'écriture car le compteur s'incrémente à chaque opération.

Voir la présentation de la mémoire 24LC32 pour lire un complément sur le compteur d'adresses interne et son auto-incrémentation.

5 . Protocole I²C

La transmission des informations se fait de manière série c'est à dire que les octets constituant le message sont décomposés en leurs chiffres binaires. À un instant donné, le bus ne véhicule qu'un un seul chiffre binaire.

5.1 Constitution physique

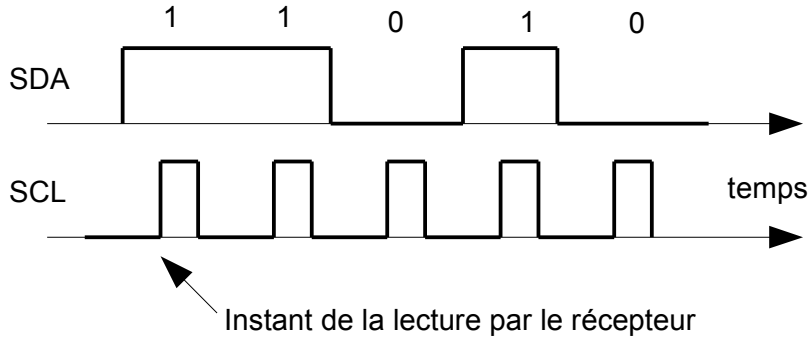
La transmission de type série nécessite deux informations :

- **quoi**, c'est le chiffre binaire dont on parle ci-dessus
- **quand**, c'est à dire à quel instant le récepteur doit lire l'information[1].

Pour satisfaire ces deux contraintes, le protocole I²C utilise un bus de trois fils,

- la masse
- un fil portant les données, **SDA** (quoi)
- un fil portant des impulsions régulières formant une sorte de métronome, **SCL (**quand)

5.2 Les niveaux logiques et leur concordance



SCL est toujours imposée par le maitre que ce soit en écriture ou en lecture.

L'émetteur donne ses informations par SDA.

Le récepteur lit chaque chiffre binaire à un instant précis. Pendant le niveau bas de SCL, les données peuvent changer, c'est interdit pendant le niveau haut.

1 Ou à quel instant l'émetteur, si ce n'est pas le maitre, doit déposer son information

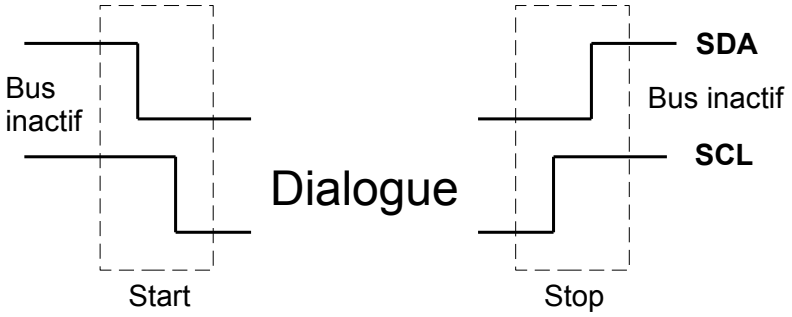
5.3 Le repos des lignes du bus

Lorsque le bus est inactif, les lignes SDA et SCL sont au niveau haut (5 V) par rapport à la masse.

5.4 Marque de début et marque de fin de transmission

Le maitre, qui est à l'initiative de la transmission, avertit de son intention de commencer un dialogue par une séquence particulière sur les lignes SDA et SCL : la condition de Start

Lorsqu'il le dialogue est terminé, il applique une autre séquence aux lignes SDA et SCL : la condition de Stop.



5.5 Format des informations véhiculées par le bus

Les informations se présentent toujours sous la forme d'octets. Ces informations peuvent être :

- l'adresse du composant auquel le maitre veut s'adresser
- l'adresse d'une partie interne du composant pouvant contenir un octet
- la donnée

5.6 Le dialogue

Définition du sens de l'échange :

Lorsque le maitre envoie une information vers un esclave, c'est une écriture.

Lorsque le maître va chercher une donnée, c'est une lecture.

Principe général du dialogue (du protocole)

La première information qui suit une condition de Start est l'adresse du composant auquel le maître veut s'adresser.

Le récepteur, maître ou esclave, annonce qu'il a bien reçu l'information envoyée par un acquittement (Ack) Voir ci-dessous.

Si l'information contient plusieurs octets, ils sont envoyés ou lus les uns après les autres et séparés par un acquittement.

Il faut distinguer le cas de l'échange en écriture du cas de l'échange en lecture.

En écriture (pour simplifier, on ne mentionne pas l'acquittement)

Tout l'échange se fait avec le bit de sens en écriture.

- la condition de Start
- adresse du composant auquel le maître veut s'adresser.

La suite du protocole peut différer d'un composant à l'autre mais on trouve :

- l'adresse de la partie interne à laquelle le maitre veut s'adresser, sur un ou plusieurs octets. Les octets sont envoyés les uns à la suite des autres et séparés par un acquittement.
- la donnée, sur un ou plusieurs octets. Les octets sont envoyés les uns à la suite des autres et séparés par un acquittement.

- la condition de Stop

En lecture

L'échange se fait en deux étapes une première partie avec le bit de sens en écriture, une seconde partie avec le bit de sens en lecture.

Pendant la première partie, l'utilisateur prépare le composant à la lecture en disant quel registre il veut lire. Ce dialogue se fait avec le bit de sens en écriture.

En seconde partie, le maître réalise les opérations de lecture, pour cela, le bit de sens est en lecture. La séquence de lecture commence par une condition de Start car tout octet suivant un Start est considéré comme une adresse de composant.

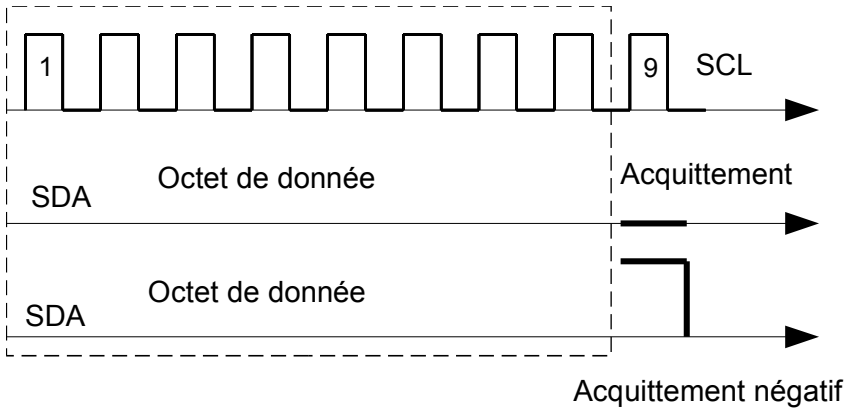
- la condition de Start
- adresse du composant auquel le maître veut s'adresser. (bit de sens en écriture)
- acquittement de la part du composant
- envoi de l'adresse du registre auquel on veut s'adresser
- acquittement de la part du composant
- nouvelle condition de Start
- adresse du composant auquel le maître veut s'adresser (le bit de sens est en lecture)
- acquittement de la part du composant
- lecture de la première information par le maître
- acquittement de la part du maître
- … *série de lectures*
- lecture de la dernière information par le maître
- acquittement négatif de la part du maître
- la condition de Stop

Attention ce schéma peut différer d'un composant à l'autre. Il faut lire la documentation du constructeur.

5.7 Acquittement et acquittement négatif

Pour traiter un octet, aussi bien en écriture qu'en lecture, le maître envoie une série de 8 impulsions sur la ligne SCL. Pour réaliser l'acquittement, le maître envoie une 9° impulsion pendant laquelle le récepteur, maître ou esclave :

- force SDA à 0 pour un acquittement
- force SDA à 1 pour un acquittement négatif.



Exemple de schéma mis en œuvre pour une liaison avec un LM75

Pour une lecture on trouvera la séquence suivante

Start SelW Ack Choix Ack Start SelR Ack OctetMSB Ack OctetLSB Nack Stop

souligné, l'acquittement de la part du Maître

Pour une écriture on trouvera la séquence suivante

Start SelW Ack Choix Ack OctetMSB Ack OctetLSB Ack Stop

SelW	: l'octet d'adresse du composant se termine par le bit de sens en écriture
SelR	: le bit de sens est en lecture.
Ack	: acquittement du récepteur (S)
Nack	: acquittement négatif du maître (M)
Choix	: octet de choix du registre auquel on veut s'adresser
OctetMSB	: octet de poids fort de la donnée
OctetLSB	: octet de poids fable de la donnée
Start	: présentation de la condition de Start (= début de transmission)
Stop	: présentation de la condition de Stop (= fin de transmission)

6 . Stratégie de programmation

La stratégie de programmation diffère selon le microcontrôleur mis en œuvre. Dans tous les cas il faudra utiliser les sous-programmes pour alléger le programme.

Le CUBLOC possède des instructions assez complètes. En revanche, le programmeur de PicBasic doit rédiger la plupart des instructions.

Le principe est d'écrire ses sous-programmes le plus universels possibles et d'utiliser des arguments pour donner les valeurs particulières

Le CUBLOC donne la possibilité de mettre en oeuvre des sous-programmes avec des arguments, la tâche est facilitée. Pour le PicBasic, il faut affecter les valeurs particulières à des variables que les sous-programmes vont exploiter.

Je conseille de placer Start et Stop dans le programme principal pour délimiter la zone de la liaison I²C.

6.1 Exemple de programme de gestion du LM75

```
Const Device = CB320
' lecture température sur LM75
' définition des constantes
    Const adr = &H48            'adresse proprement dite du LM75
    Const chx = 0    'octet de sélection des registres, ici on ne s'adresse qu'au registre de
température
'définition des variables
    Dim vadr As Byte            'adresse complète contenant le bit de sens
    Dim xx As Byte              'octet  a tout faire
    Dim temp As Byte            'contient les degrés entiers
    Dim demi As Byte            'contient le demi degré
    Set I2c 9,8
debut:
Do
    I2cstart
    selW adr        ' l'adresse du composant est passée en argument,
    selR adr
    lire            'le SP s'occupe de tout
    I2cstop
    Delay 1000
Loop
    End
' ========== sous programmes
Sub selW(ad As Byte)
    vadr = ad * 2 + 0       'construction de l'adresse complète avec sens en écriture
    xx = I2cwrite(vadr)  'utilisation des instructions de la panoplie du CUBLOC
    xx = I2cwrite(chx)   'le choix du registre est ici, est-ce une bonne idée, je ne sais pas.
    End Sub
Sub selR(ad As Byte)
    I2cstart                'la répétition du Start[2]
    vadr = ad * 2 + 1       'construction de l'adresse complète avec sens en lecture[3]
    xx = I2cwrite(vadr)
    End Sub
Sub lire()
    temp = I2cread(0)           ' premier octet
    demi = I2creadna(0)         ' dernier octet de donnée.
    End Sub
```

2 Est-ce une bonne idée ? Cela facilite la lecture du programme principal mais brouille le schéma classique.
3 Ça c'est bien

6.2 Exemples de sous-programmes pour le PicBasic

```
'---------- condition de début de transmission
I2Cstart :
out sda, 1   'par précaution
out scl,1
out sda,0
return
'---------- condition de fin de transmission
I2Cstop :
out sda, 0   'par précaution
out scl,1
out sda,1
return
'------ Sélection du composant en écriture, la donnée est dans la variable adr (adr est modifiée)
SelW:
gosub I2Cstart      'on peut également expliciter au lieu d'appeler le SP
adr = adr * 2       'décalage de une place vers la gauche
adr = adr + 0       'introduction du bit de sens (par cohérence avec la lecture)
shiftout scl, sda, mode, adr   'mode = 2 pour l'attente d'un ack
'------- Sélection du composant en lecture, la donnée est dans la variable adr (adr est modifiée)
SelR:
gosub I2Cstart      'on peut également expliciter au lieu d'appeler le SP
adr = adr * 2       'décalage de une place vers la gauche
adr = adr + 1       'introduction du bit de sens
shiftout scl, sda, mode, adr   'mode = 2 pour l'attente d'un ack
'---------- écriture d'un octet, la donnée est dans la variable don
OctetW:
shiftout scl, sda, mode, don
return
'---------- lecture d'un octet courant
OctetR:
don = shiftin(scl, sda, mode)       ' mode est en général  égal à 3
out scl, 0                          'par précaution
out sda,0                           'génération d'un Ack = impulsion sur scl avec sda = 0
out scl, 1
out scl, 0
return
'---------- lecture d'un octet de fin
OctetRF:
don = shiftin(scl, sda, mode)       ' mode est en général  égal à 3
out scl, 0                          'par précaution
out sda,1                           'génération d'un Nack = impulsion sur scl avec sda = 1
out scl, 1
out scl, 0
return
```