Présentation du bus I2C

Le bus I2C a été créé au début des années 80 par RTC Philips afin d'apporter une **solution simple et peu coûteuse** à la communication entre les circuits intégrés numériques à l'intérieur des appareils grand public (téléviseurs, magnétoscopes, etc.). Le principal avantage du bus I2C est de limiter le nombre de liaisons entre circuits intégrés.

1 - Structure et organisation du bus I2C

Le bus I2C est un bus de type **série synchrone** ne nécessitant que deux signaux.

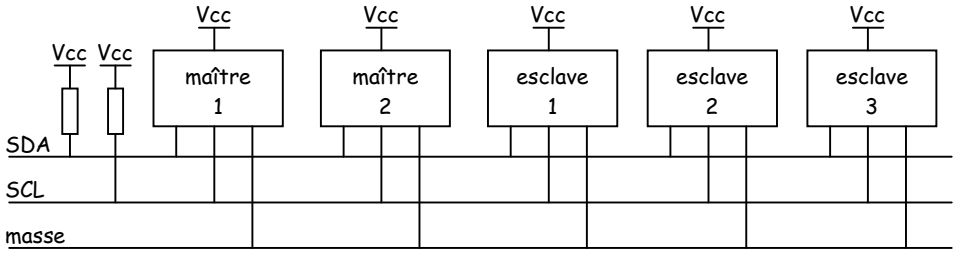
Rappel : Un bus série synchrone est un bus ou les données circulent sur un seul fil les unes après les autres au rythme (synchronisées par), un signal d'horloge envoyé sur un second fil.

Pour le bus I2C, ces deux fils s'appellent :

- **SDA** (serial data), le signal de donnée.
- **SCL** (serial clock), le signal d'horloge.

Ce bus permet la communication entre un circuit **maître** et un **circuit** esclave. Le montage peut comporter **plusieurs** maîtres et plusieurs esclaves. Le **maître** est le circuit qui émet le signal d'horloge de synchronisation, un seul maître à la fois peut envoyer ce signal. Les données peuvent circuler dans les **deux sens** sur le fil des données, de sorte que chaque circuit, qu'il soit maître ou esclave peut servir d'émetteur ou de récepteur (de données).

Les différents circuits sont placés en **parallèle** sur les lignes **SDA** et **SCL** comme le montre le schéma suivant :

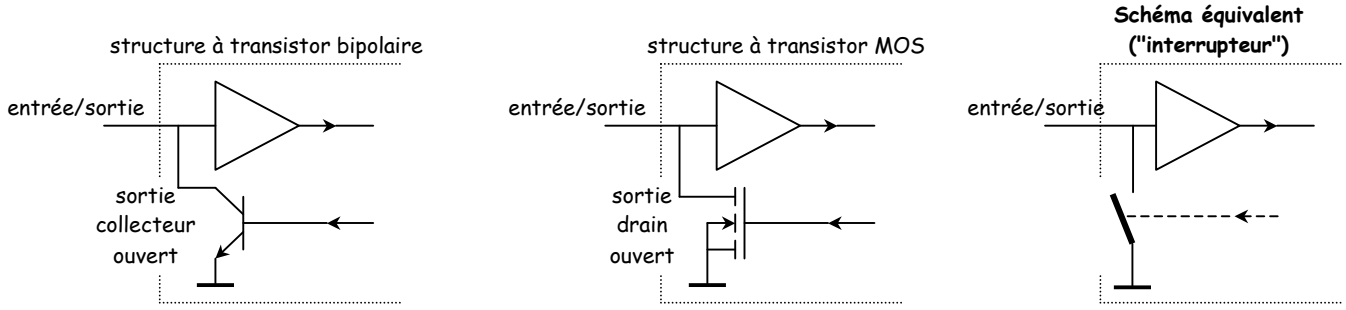


Au repos, c'est à dire lorsque aucun circuit n'émet, les signaux **SDA** et **SCL** sont au niveau logique **haut**.

Il faut des résistances de rappels (*"pull-up")* sur les fils **SDA** et **SCL** car les sorties des circuits sont de type collecteur ou drain ouvert (≈ **interrupteurs ouverts**). Ces résistances établissent un niveau logique haut sur les lignes SDA et SCL lorsque toutes les sorties sont à l'état haute impédance (≈ *circuits ouverts*). Les sorties reliées ensembles forment un **ET câblé**, c'est à dire que **dès qu'un circuit applique en sortie un niveau logique bas, toute la ligne est à 0 quel que soit l'état des autres sorties**.

Pour éviter les conflits, **un maître qui veut émettre doit attendre que le bus soit au repos**.

Structure des entrées/sorties SDA et SCL :



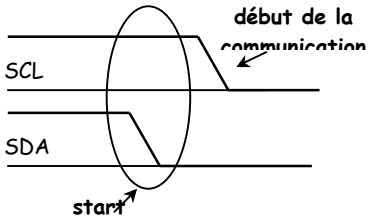
2 - Le protocole I2C

Le **protocole** est l'équivalent du langage chez l'être humain, c'est la façon avec laquelle les circuits vont communiquer entre eux.

❑ Prise de contrôle du bus par un maître et début de communication :

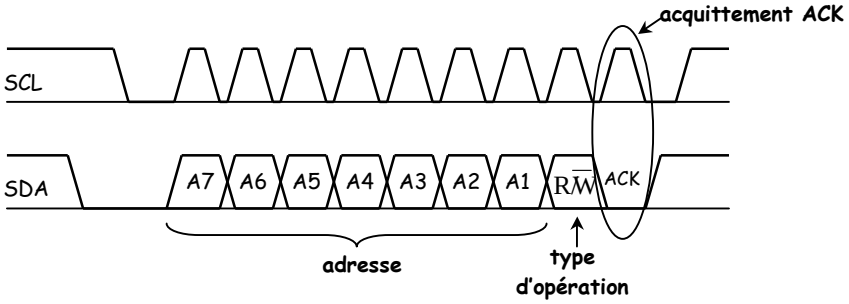
Le bus étant initialement libre, **SDA** et **SCL** sont à **1**. Un maître prend le contrôle du bus en effectuant un **START** : il met SDA à 0, SCL restant à 1.

Au cours de la communication, l'horloge SCL est envoyée par le maître et SDA ne peut changer d'état que lorsque SCL est à 0.



❑ Choix de l'esclave et du type d'opération par le maître et réponse de l'esclave :

Après avoir pris le contrôle du bus, le maître envoie une **adresse sur 7 bits** pour sélectionner un esclave (MSB en premier). Le **huitième bit**, appelé $R/\overline{W}$ (Read/Write), indique si le maître souhaite faire une opération de lecture ou d'écriture. Simultanément, le maître génère le signal d'horloge SCL comportant huit créneaux.



Tous les **esclaves observent** l'adresse envoyée par le maître. L'esclave qui reconnaît son adresse répond au maître en **maintenant SDA** au niveau logique bas : c'est le **bit d'acquittement** appelé **ACK** (acknoledge). L'esclave est prêt à communiquer avec le maître. Les autres esclaves restent au repos. Si le bit $R/\overline{W}$ envoyé par le maître est à :

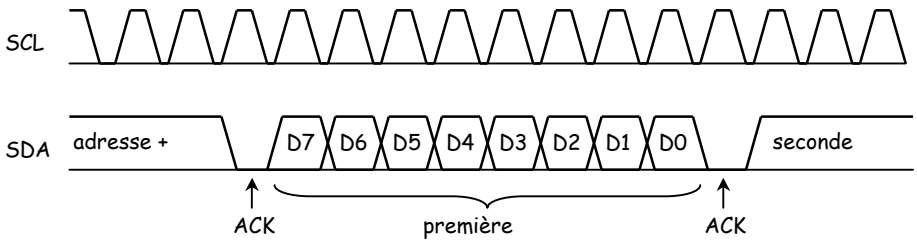
- **1** (lecture), l'esclave prend la parole.
- **0** (écriture), l'esclave se met à l'écoute du maître.

❑ Envoi des données :

Chaque donnée est formée d'un **octet** envoyé bit par bit sur la ligne SDA. Au cours de la communication, plusieurs octets peuvent être transmis.

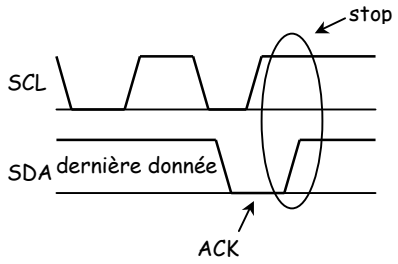
- s'il s'agit d'une opération de **lecture**, les données sont envoyées au maître par l'esclave.
- s'il s'agit d'une opération **d'écriture**, les données sont envoyées à l'esclave par le maître.

D'abord l'émetteur envoie les 8 bits du premier octet puis le récepteur fait l'acquittement en maintenant la ligne SDA à 0 (bit ACK). L'émetteur envoie ensuite les 8 bits du second octet et le récepteur fait l'acquittement. Le même cycle se répète jusqu'à ce que toutes les données aient été envoyées. Dans les deux cas, l'horloge SCL est générée par le maître.



❑ Fin de communication et libération du bus :

Pendant la communication, les changements d'état sur **SDA** se produisent lorsque **SCL** est à **0**. Pour mettre fin à la communication, le maître effectue un **STOP** : il met d'abord SCL à 1 puis ramène SDA à 1. C'est le changement d'état de SDA alors que SCL est à 1 qui met fin à la communication.



❑ Synchronisation du maître et de l'esclave :

Habituellement, la ligne **SCL** fonctionne en sortie pour un maître et en entrée pour un esclave car c'est le maître qui génère le signal d'horloge. Mais lorsqu'un esclave veut obliger un maître à ralentir la communication, l'esclave maintient la ligne SCL à 0 pour empêcher le maître de la ramener à 1. La communication reprend son cours lorsque l'esclave libère la ligne SCL. Cette opération s'appelle la **synchronisation**.

❑ Arbitrage en cas de conflit :

Un maître ne **peut pas prendre le contrôle du bus** si celui-ci est déjà **occupé**. Mais il peut arriver que deux maîtres cherchent à prendre le contrôle du bus en même temps, à quelques nanosecondes près. Dans ce cas l'arbitrage se fait sur la ligne SDA. Le premier des deux maîtres qui veut établir la communication doit vérifier que SDA=1 (ligne libre), si cette dernière est à 0, cela signifie que la ligne est occupée, il doit cesser d'émettre et se placer en récepteur au cas où la donnée envoyée par l'autre maître lui serait destinée.

3 - Performances et limites d'emploi du bus I2C

Le bus I2C est réservé à de **courtes liaisons** : entre composants sur la même carte ou entre cartes par fils de quelques centimètres.

Les spécifications du bus I2C datant de 1992 imposent :

- l'adressage se fait sur 10 bits soit 1024 adresses possibles.
- la vitesse de transmission est limitée à 400 Kbits/seconde, soit 50 Koctets/seconde.
- la sensibilité au bruit est moindre grâce à des entrées trigger de Schmitt.
- les sorties d'un circuit non alimenté sont mises en haute impédance afin de ne pas bloquer le bus.

4 - Exemples de circuits intégrés pour le bus I2C

Le **maître** est souvent un **microcontrôleur** possédant une interface I2C.

En l'absence de microcontrôleur, ou si celui-ci n'est pas doté d'un interface I2C, on utilise une interface Bus parallèle/Bus I2C. Cet interface est alors maître.

Les **esclaves** sont des circuits intégrés intégrant un interface pour le bus I2C et réalisant la même fonction que des circuits intégrés couramment utilisés en électronique :

- CAN ou CNA : PCF8591
- mémoire vive : PCF8570
- EEPROM : PCF8582
- horloge calendrier : PCF8576
- Port d'entrée / Sortie parallèles (8) : PCF8574
- Afficheurs LCD
- …..

Du fait de sa simplicité de mise en œuvre, les modules et shields Arduino utilisent souvent un bus I2C.
Ils sont repérables grâce aux marquages **SCL** et **SDA** sur les connecteurs et doivent être connectés aux prises **I2C** des shields d'extension.