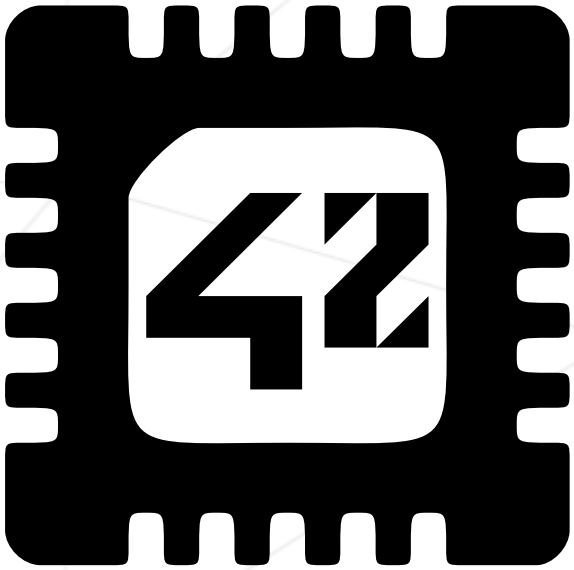

Piscine embarquée

Day 04 : Protocole I2C

contact@42chips.fr

Résumé: Two wires to rule them all.

Chapitre I

Préambule

Le protocole I2C pour (Inter-Integrated Circuit) ou TWI (Two Wire Interface) est un bus de communication utilisé pour connecter des périphériques à un microcontrôleur ou à un ordinateur. Il a été développé par Philips Semiconductors dans les années 1980 et est devenu un standard industriel largement utilisé dans de nombreux domaines, tels que l'électronique de consommation, l'automobile, l'aérospatiale et les communications.

Le protocole I2C permet à plusieurs périphériques de communiquer sur un seul bus en utilisant deux fils : un fil de données (SDA) et un fil d'horloge (SCL).
Un seul périphérique est autorisé à envoyer des données sur le bus à la fois, mais plusieurs périphériques peuvent être connectés au bus et être adressés individuellement.
Cela permet de réduire les coûts en utilisant moins de fils et en simplifiant les circuits imprimés.

Le protocole I2C utilise un schéma de maître-esclave pour la communication.
Le périphérique qui initie la communication est le maître, tandis que les périphériques qui reçoivent les commandes sont les esclaves.
Le maître envoie une série de bits d'adresse à chaque esclave pour identifier le périphérique cible, puis envoie des données au périphérique cible ou lit des données à partir de celui-ci.

Le protocole I2C permet également la communication de données en mode "point à point", c'est-à-dire que le maître peut communiquer directement avec un seul esclave à la fois.
Cela peut être utile dans les applications où il y a un grand nombre de périphériques sur le bus et où la latence de communication doit être minimisée.

En plus de la communication de données, le protocole I2C prend en charge plusieurs autres fonctionnalités, telles que la détection de collision de données, la gestion de l'alimentation et l'auto-identification des périphériques. Toutes ces fonctionnalités font du protocole I2C un choix populaire pour de nombreuses applications de communication de données.

Le protocole I2C est particulièrement utile dans les applications où il y a plusieurs périphériques et où l'espace est limité, car il nécessite seulement deux fils pour la communication.
Il est également facile à mettre en œuvre et à utiliser, ce qui en fait un choix populaire pour de nombreux projets.

Chapitre II

Consignes générales

Sauf contradiction explicite, les consignes suivantes seront valables pour tous les exercices.

- Le langage utilisé pour ce projet est le C.
- Il n'est pas nécessaire de coder à la norme de 42.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne prendrons en compte ni n'évaluerons un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Vos exercices seront évalués par des responsables de l'association 42Chips.
- Vous ne devez laisser aucun autre fichier que ceux explicitement specifiés par les énoncés des exercices dans votre répertoire lors de la peer-évaluation.
- Toutes les réponses à vos questions techniques se trouvent dans les `datasheets` ou sur Internet. À vous d'utiliser et d'abuser de ces sujets pour comprendre comment réaliser votre exercice.
- Vous devez utiliser la datasheet du microcontrôleur qui vous est fourni et commenter les parties importantes de votre programme en renseignant où vous avez trouvé les indices dans le document, et, si nécessaire, expliquer votre démarche. Ne faîtes pas des pavés non plus. Il faut que cela reste clair.
- Vous avez une question ? Demandez à votre voisin de droite ou de gauche. Vous pouvez demander sur le salon dédié dans le Discord de la piscine ou en dernier recours à un staff.

Chapitre III

Je vous ai compris !

	Exercice : 00
	Two Wire Interface
	Dossier de rendu : *ex*00/
	Fichiers à rendre : `Makefile, *.c, *.h`
	Fonctions Autorisées : `avr/io.h, util/twi.h`

- Le microcontrôleur AVR ATmega328P possède 1 interface I2C que vous devez utiliser dans cet exercice pour communiquer avec un capteur de température AHT20 (U3).
- Vous devez écrire une fonction `i2c_init` qui initialise l'I2C sur le microcontrôleur.
- L'I2C du MCU doit être configuré afin que la fréquence de communication soit de 100 kHz.
- Vous devez écrire une fonction `i2c_start` qui démarre une transmission I2C entre le microcontrôleur et le capteur.
- Le programme devra retourner sur votre ordinateur les valeurs de statut après chaque envoi de donnée.
- Vous devez écrire une fonction `i2c_stop` qui interrompt la communication entre le microcontrôleur et le capteur.

```
void i2c_init(void)
void i2c_start(void)
void i2c_stop(void)
```

	Exercice : 01
	Brute Data
	Dossier de rendu : *ex*01/
	Fichiers à rendre : `Makefile, *.c, *.h`
	Fonctions Autorisées : `avr/io.h, util/twi.h, util/delay.h`

- Complétez le programme précédent, en ajoutant une fonction `i2c_write` qui écrira le contenu du registre TWDR du microcontrôleur et l'enverra dans le capteur de température.
- Vous devez écrire une fonction `i2c_read` qui affichera le contenu du registre TWDR après la mesure par le capteur.
- Vous devez écrire une fonction `print_hex_value` qui écrira le contenu des 7 octets d'une mesure du capteur AHT20 sans modification sur la sortie standard de votre PC.
- Vous devez faire afficher le retour de valeurs dans une boucle qui se répètera avec une fréquence respectant les préconisations constructeur.

```
void i2c_write(unsigned char data)
void i2c_read(void)
void print_hex_value(char c)
```

```
0C 79 9A A6 4E 3C F2
0C 79 83 76 4E 29 78
0C 79 8D 36 4E 19 5A
0C 79 7E 06 4E 82 91
```

Soyez attentif au délai demandé entre la fin de la commande d'écriture et la lecture des trames de données.

	Exercice : 02
	Ça va chauffer !
	Dossier de rendu : *ex*02/
	Fichiers à rendre : `Makefile, *.c, *.h`
	Fonctions Autorisées : `avr/io.h, util/twi.h, util/delay.h, dtostrf()`

- Complétez le programme précédent afin qu'il affiche la température et l'humidité en lieu et place des valeurs brutes obtenues plus haut.
- Les valeurs de température et d'humidité devront être affichées en respectant les paramètres suivants :
  - la précision affichée sur le terminal doit respecter les préconisations données dans la datasheet, arrondies à la dizaine supérieure
    (ex : 0.0031 -> 0.01 ou 0.1 -> 0.1) ;
  - la température sera en degrés Celsius (°C), l'humidité en pourcentage (%) ;
  - la mesure affichée doit être la moyenne des 3 dernières mesures prises par le programme. Le comportement pour les deux premières mesures devra être cohérent avec cette règle.

```
Temperature: 17°C, Humidity: 43,4%
Temperature: 18°C, Humidity: 43,6%
Temperature: 18°C, Humidity: 43,5%
Temperature: 17°C, Humidity: 43,8%
```

Attention, dans l'exemple ci-dessus, la résolution et la précision ne sont pas celles qui vous sont demandées et ne sert que pour visualiser l'affichage du résultat tel que demandé dans l'exercice.