TP Transmission Numérique

Série Isochrone (Bus I²C)

Kimberley Jacquemot & Ayoub Ech Chamali

02/12/2022

I - Introduction

I.a. - Questions

1) Quel est le bloc logique interne à l'ATMEGA 328P qui devra être utilisé afin de mettre en œuvre ce type de transmission ?

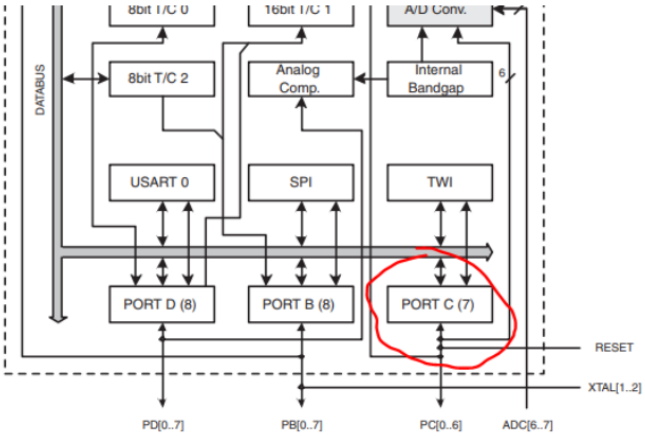


Figure 1 - Schéma logique Arduino Uno

On utilisera le TWI : Two Wires Interface. Nous l'utiliserons à travers le Port C de la carte.

2) Quel est le type de transmission mis en œuvre en I2C ?

C'est une transmission de type bus série synchrone bidirectionnel half-duplex (Isochrone)

3) Décrire ce que sont les deux états récessif et dominant.

L'état logique 0 ou « LOW » est l'état « dominant »

L'état logique 1 ou « HIGH » est l'état « récessif »

Le but est d'éviter que l'état dominant et récessif répondent simultanément. Pour cela, nous allons leur attribuer des adresses différentes.

4) Quelles sont les pattes qui représentent SDA (Data) et SCL (Clock) de l'ARDUINO UNO ?

Nous avons la description de ces deux pattes grâce à la documentation comme vous pourrez le voir ci-dessous.

- **SCL/ADC5/PCINT13 – Port C, Bit 5**

SCL, 2-wire Serial Interface Clock: When the TWEN bit in TWCR is set (one) to enable the 2-wire Serial Interface, pin PC5 is disconnected from the port and becomes the Serial Clock I/O pin for the 2-wire Serial Interface. In this mode, there is a spike filter on the pin to suppress spikes shorter than 50 ns on the input signal, and the pin is driven by an open drain driver with slew-rate limitation.

PC5 can also be used as ADC input Channel 5. Note that ADC input channel 5 uses digital power.

PCINT13: Pin Change Interrupt source 13. The PC5 pin can serve as an external interrupt source.

- **SDA/ADC4/PCINT12 – Port C, Bit 4**

SDA, 2-wire Serial Interface Data: When the TWEN bit in TWCR is set (one) to enable the 2-wire Serial Interface, pin PC4 is disconnected from the port and becomes the Serial Data I/O pin for the 2-wire Serial Interface. In this mode, there is a spike filter on the pin to suppress spikes shorter than 50 ns on the input signal, and the pin is driven by an open drain driver with slew-rate limitation.

PC4 can also be used as ADC input Channel 4. Note that ADC input channel 4 uses digital power.

PCINT12: Pin Change Interrupt source 12. The PC4 pin can serve as an external interrupt source.

Figure 2 - Description broches SCL et SDA

5) Quels sont tous les registres qui devront être utilisés, donnez leurs descriptions.

Nous utiliserons les registres TWBR, TWCR, TWSR, TWDR, TWAR. Vous pourrez les voir plus en détail dans le schéma réalisé ci-contre.

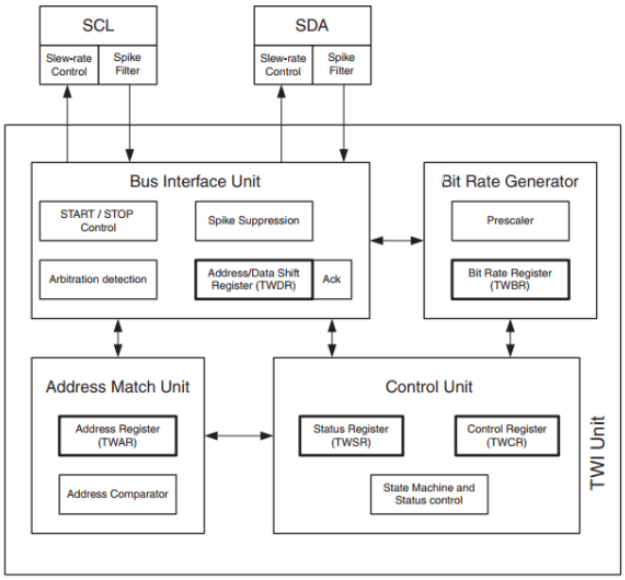
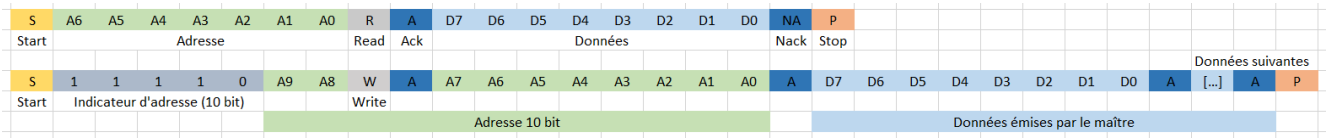


Figure 3 - Registres du TWI

6) Présentez par un schéma comment est structurée la trame I2C en version 8 et 10 bits.



7) Mettez en évidence sur le chronogramme les propriétés ISOCHRONE de la communication.

La principale différence avec l'isochrone est le 'start' et le 'restart' qui sont distingués et identifiés différemment.

8) Comment l'esclave reconnaît la version 8 ou 10 bits utilisée par le maître ?

L'esclave parvient à identifier les trames grâce à un indicateur envoyé avant toute informations par le maitre (avant adresse et données).

9) Quel sont les intérêts des bits ACK et R/W ?

Le bit ACK permet de transmettre au maître un acquiescement de l'esclave vers le maître, ainsi, le maître peut savoir que l'esclave a bien reçu l'information.

Les bits R/W correspond à Read/Write permettant de lire ou écrire les données.

10) Combien d'esclaves sont envisageables sur ce bus.

On pourrait avoir beaucoup d'esclaves, en fait c'est plus le matériel qui nous limitera. En effet, nous sommes limités par le nombre d'adresses disponibles. L'adresse des composants I²C est codée sur 7 bits, on a alors 2^7 adresses possibles. Il faut savoir que certaines adresses peuvent être réservées pour du broadcast par exemple.

11) Combien et quelles sont les vitesses de transmissions possibles, en déduire la durée d'un bit pour chacune des vitesses.

Vu que les données peuvent aller dans les deux sens (nous ne sommes pas en simplex), nous perdons en vitesse avec ce type de transmission. On a alors plusieurs vitesses :

- « Standard mode (Sm) » ≤ 100 kbit/s,
- « Fast mode (Fm) » ≤ 400 kbit/s,
- « Fast plus mode (Fm+) » ≤ 1 Mbit/s,
- « High-speed mode (Hs-mode) » ≤ 3,4 Mbit/s

12) Comment analysez-vous le rendement de la transmission dans cet échange entre les 2 cartes ?

Nous pouvons comparer SCA ainsi que SCL pour voir si les états se suivent bien.

13) Comment s'effectue la validation de l'échange qui est effectué ?

Dès que le maître détecte que SCL est au niveau haut, il teste SDA : il trouve un niveau bas (acquittement valide). Il peut alors lire le résultat de la conversion.

14) Comment s'appelle ce type de transmissions ?

C'est une transmission Simplex.

15) Quelle analyse de la fiabilité de ce bus faite vous compte tenu du mode de fonctionnement ?

Nous analysons l'information reçue par l'esclave et on la compare avec l'information envoyée.

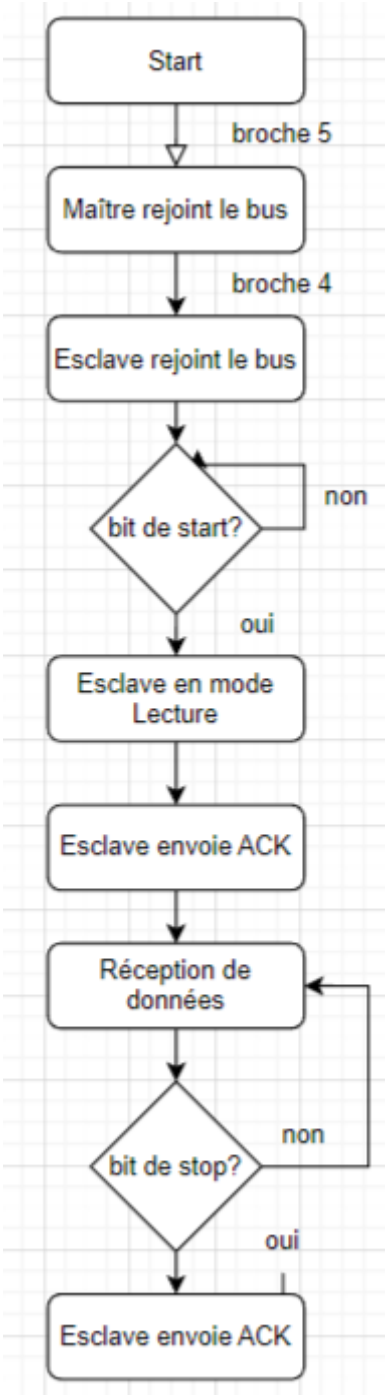
16) Comment l'esclave procède pour annoncer au maître qu'il est occupé ?

Si l'esclave est occupé il ne pourra alors ni recevoir ou transmettre de données, il fera passer la ligne SCL à l'état bas. Cela oblige le maître à attendre que le dispositif esclave libère la ligne SCL.

17) Il y a-t-il isolation galvanique entre les 2 cartes ARDUINO sollicitées par cette communication ?

Non puisque les deux circuits sont directement reliés.

18) Faites un grafcet des étapes logiques consacrées à cet échange puis à l'affichage du message.



19) Quels sont les codes fonctions que vous devez définir, Quelles valeurs choisissez-vous, Argumentez vos choix.

Dans cet exemple nous utilisons une carte Arduino Uno, nous devrons définir les adresses maître et esclave, nous utiliserons les broches A4 et A5 (Entrées/Sorties Analogues).

20) Ecrire :
    a. un Sketch pour la carte qui devra être en mode maître o un autre Sketch pour la carte qui devra être en mode esclave. Ces 2 Sketchs doivent permettre d'échanger une zone de mémoire RAM entre les 2 cartes ARDUINO. La zone RAM au sein de l'esclave doit contenir les éléments suivants :
        i. valeur codée binaire de pi,
        ii. valeur codée binaire de e,
        iii. Message : « Polytech-Nancy » Le Sketch du maître doit permettre de lire le contenu de la mémoire RAM de la carte en mode esclave et la ramener chez elle puis l'afficher sur le port terminal.

I2C.ino

```
#include <Wire.h> //La bibliothèque nous permettra de communiquer avec la RAM de l'arduino
#define I2C_SLAVE_ADDRESS A4

void setup()
{
  Wire.begin(); // Le maître entre dans le bus I2C
  Serial.begin(9600); //Configuration de la vitesse de transmission
  Serial.println(F("-----------------------------------I am the Master")); // On distingue le maître de l'esclave
  delay(1000);
}
void loop()
{
Wire.beginTransmission(I2C_SLAVE_ADDRESS); //On commence la communication avec l'esclave tant qu'il est là
Wire.write("3.14"); //Donnée à transmettre
Wire.endTransmission(); // Arrêter la transmission
delay(1000);

//De même pour les autres données
Wire.beginTransmission(I2C_SLAVE_ADDRESS);
Wire.write('e');
Wire.endTransmission();
delay(1000);
Wire.beginTransmission(I2C_SLAVE_ADDRESS);
Wire.write("Polytech-Nancy");
Wire.endTransmission();
delay(1000);
}
```

Figure 4 - Arduino Sketch pour le maître I²C

I2C_Slave.ino

```ino
#include <Wire.h> //La bibliothèque nous permettra de communiquer avec la RAM de l'arduino
#define I2C_SLAVE1_ADDRESS A4

void setup()
{
  Wire.begin(I2C_SLAVE1_ADDRESS); //L'esclave rentre dans le bus à son adresse
  Serial.begin(9600);
  Serial.println("-------------------------------------I am Slave"); //On distingue l'esclave du maître
  delay(1000);
  Wire.onReceive(receiveEvent); //Lorsque l'on reçoit un 'start' du maître, on appelle notre fonction qui va lire le message
}

void loop()
{
  delay(100);
}

void receiveEvent(int howMany)
{
  String data = ""; //On créer la variable qui va stocker les chars que l'esclave reçoit
  while( Wire.available()){
    data += (char)Wire.read();
  }
    Serial.println(data); //On affiche les données
}
```

Figure 5 - Sketch Arduino de l'esclave I²C

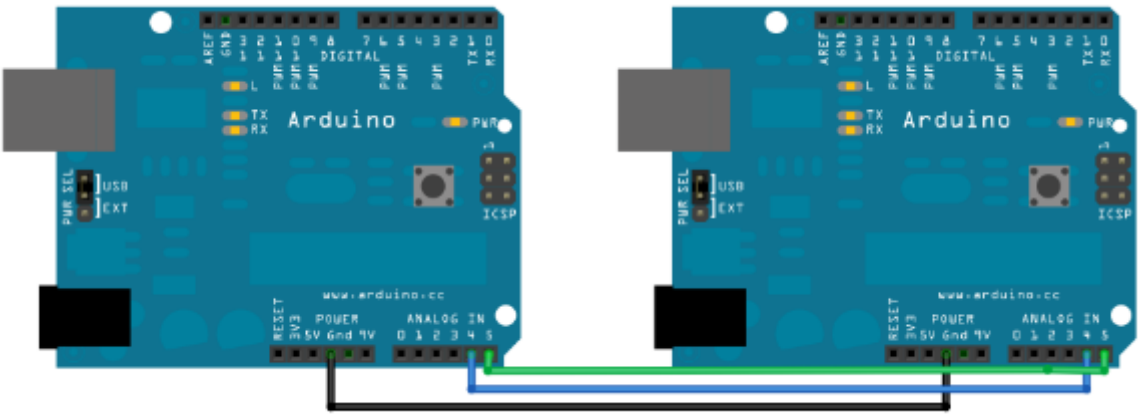


Figure 6 - Montage Arduino I²C

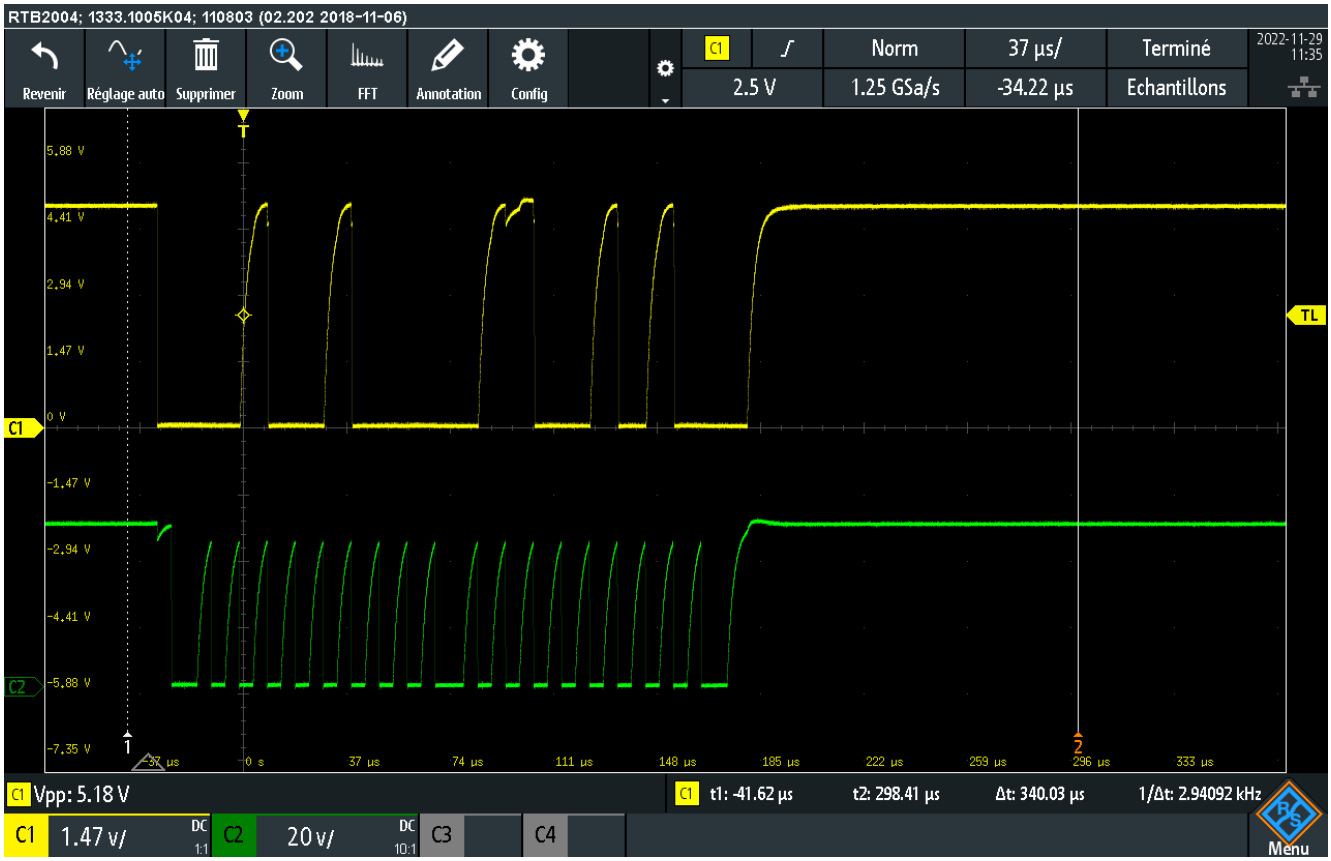


Figure 7 - Transmission "3,14"

Vous pourrez voir sur le graphique ci-dessus le signal SDA (jaune) ainsi que le signal SDL (vert) pour la transmission du 3,14.

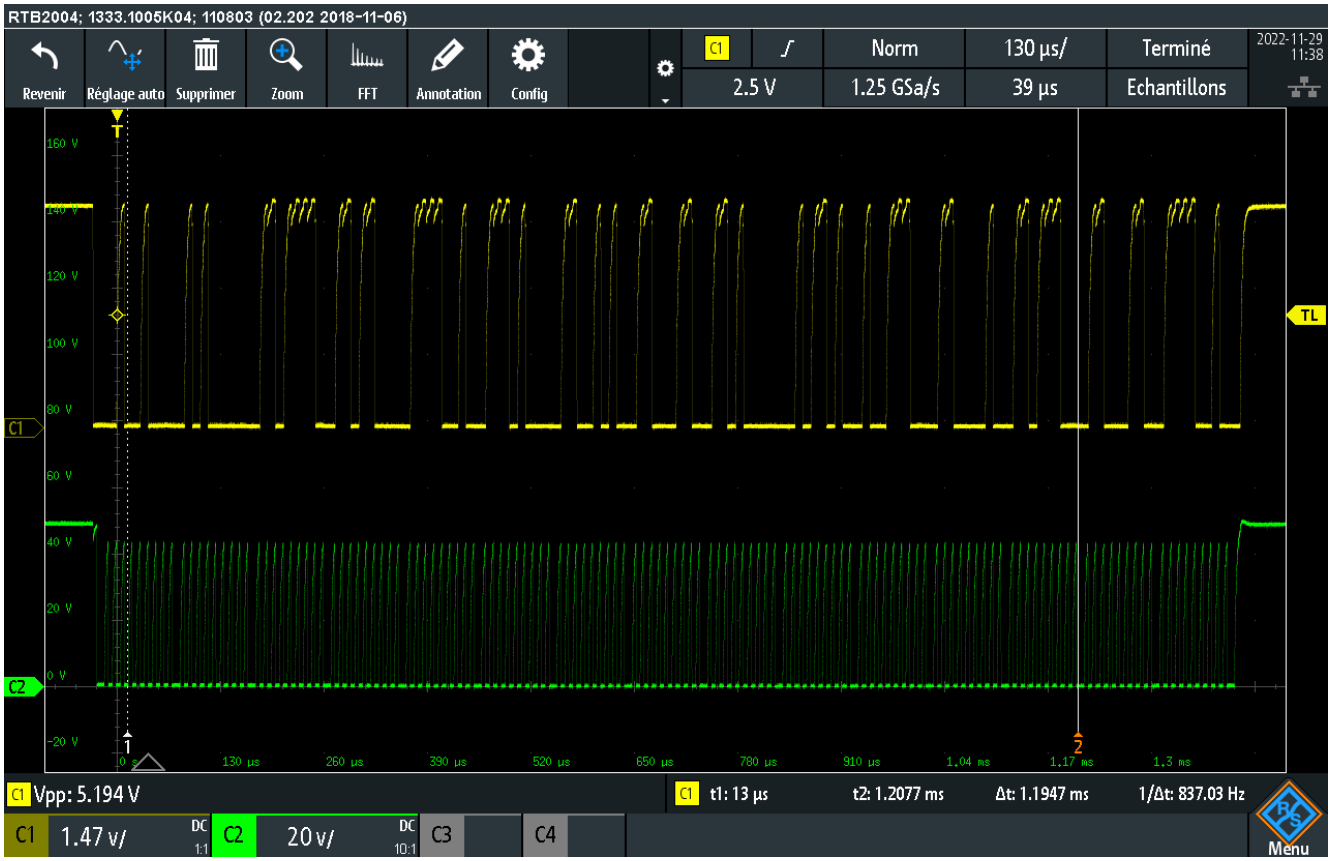


Figure 8 - Transmission "Polytech-Nancy"

On peut voir que l'analyse/lecture de trame peut rapidement se compliquer, il devient assez dur de détecter un décalage ou bien une erreur à l'œil nu.

21) En déduire le rendement de ce mode de transmission.

Nous recevons bien les informations dans une trame « Polytech Nancy » nous mettons à présent 1,4ms. Ce qui était auparavant la transmission d'un octet.

22) Quelles sont en résumé les caractéristiques principales de l'I2C ?

L'I²C permet de transmettre et recevoir des données dans une communication double, l'esclave peut répondre au maître et cela nous permet de temporiser avec un délai automatiquement pour laisser les deux appareils synchronisés constamment. Avec une bonne configuration, l'I²C peut être incroyablement efficace.

23) Que concluez-vous des échanges avec ce type de communication I2C ?

L'I²C est beaucoup plus efficace et évite des erreurs sur des données plus importantes grâce à ses indicateurs.