

LCD

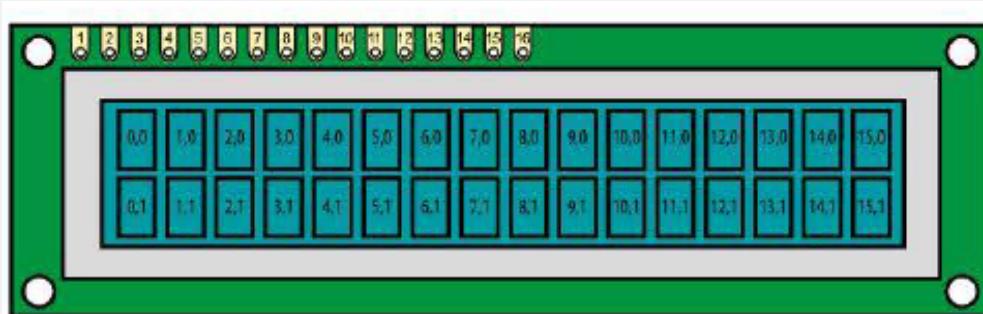
Objectifs

- Implémenter l'affichage dans un circuit à cristaux liquides (LCD)
 - Méthode filée
 - Méthode par le protocole **I2C** (Démonstration)
- Exercices



Circuits à cristaux liquides (LCD)

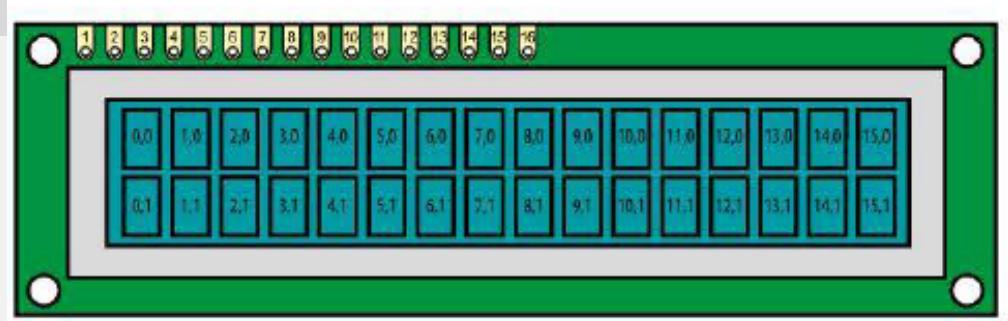
- BUT : afficher des caractères sur un écran LCD
- Matrice de pixels 5X 8 étendue sur une matrice en colonnes X lignes
 - Texte: exemple « Bonjour le monde ! »
 - Motifs programmables : exemple sourire



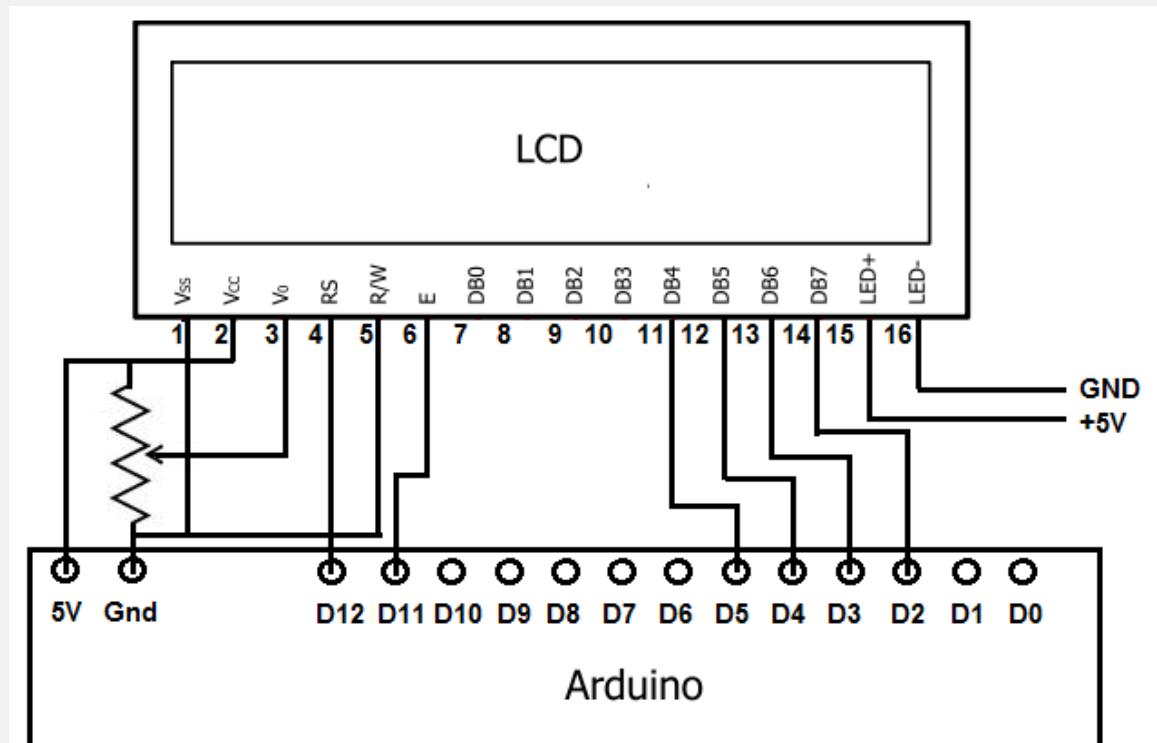
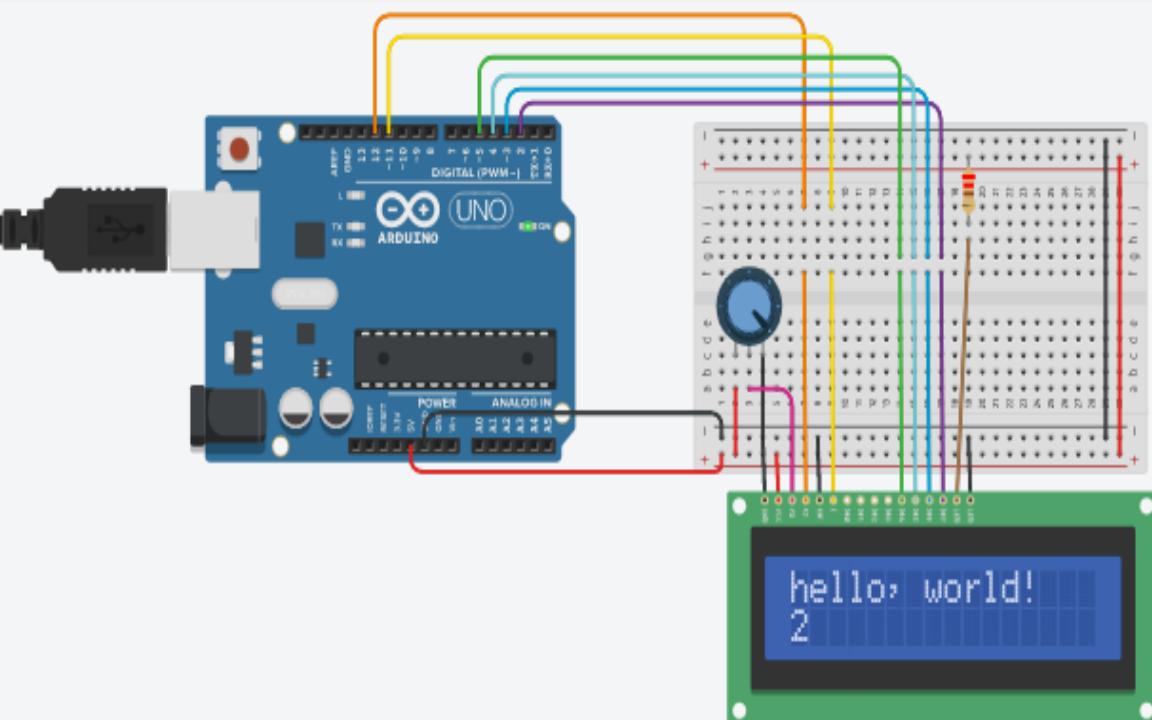
12. Standard character pattern		LLLL	LLLH	LLHL	LLHH	LHLL	LHLH	LHHL	LHHH	HLLL	HLLH	HLHL	HLHH	HHLL	HHLH
Upper 4bit	Lower 4bit	CG RAM (1)													
LLLL															
LLLH	(2)														
LLHL	(3)														
LLHH	(4)														
LHLL	(5)														
LHLH	(6)														
LHHL	(7)														
LLLL	(8)														

Circuits à cristaux liquides (LCD)

- Modèles variés offerts
 - Couleurs des pixels
 - Couleur du fond d'écran avec rétro-éclairage
 - Dimensions de la matrice
- 2 modes de programmation
 - Contrôle filée: minimum 8 (6 + 2) fils,
 - + 3 fils optionnels pour contrôler le rétro-éclairage
 - Contrôle par protocole **I2C**: 4 (2 + 2) fils !

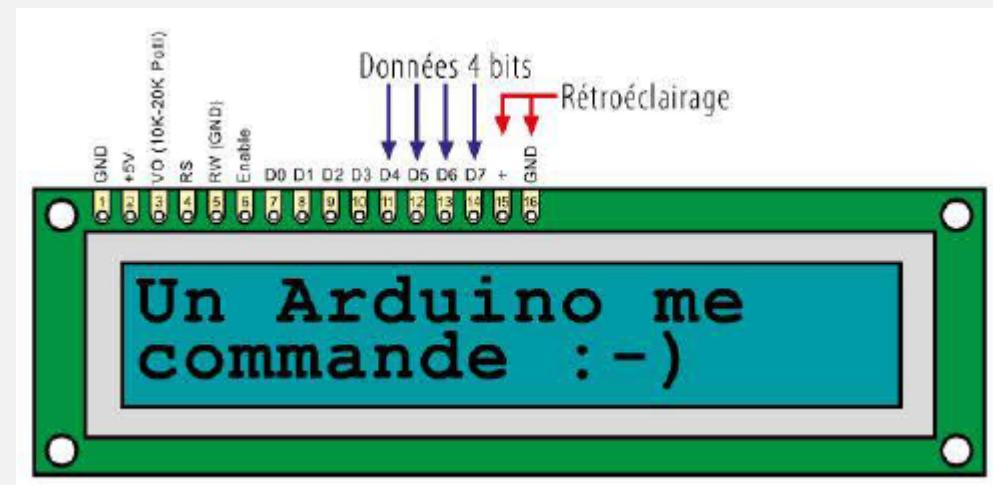


Méthode filée



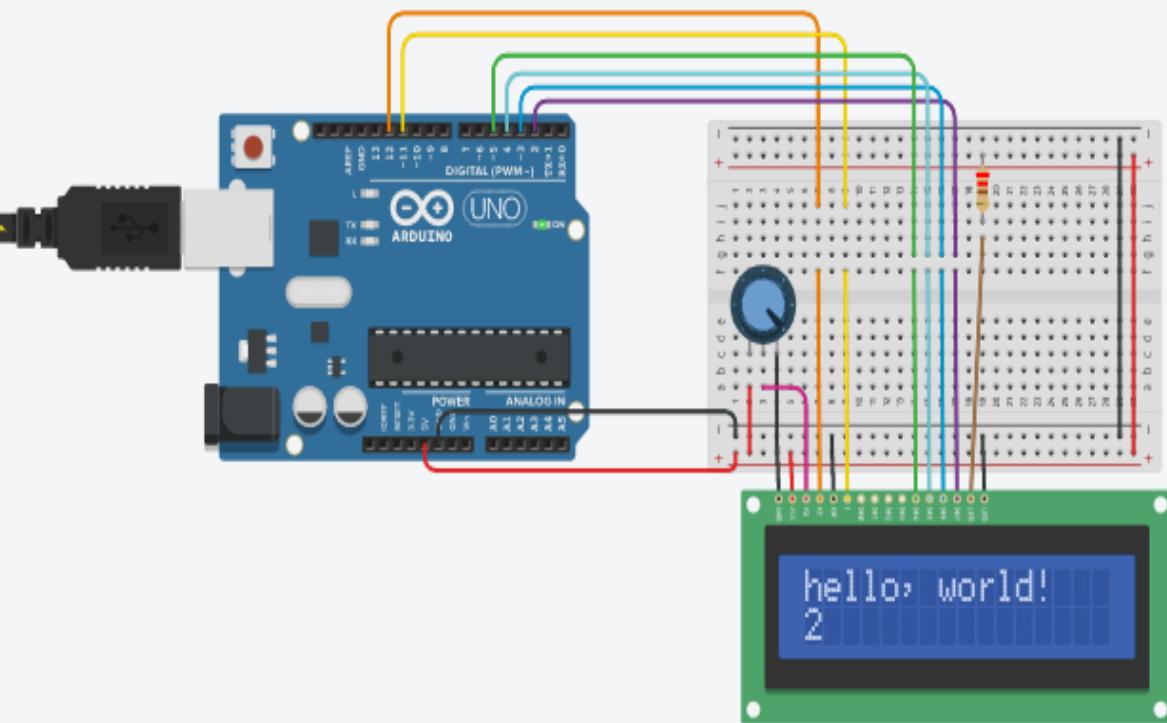
Circuits à cristaux liquides (LCD)

- Contrôlé par l'objet <LiquidCristal.h>
- Bornes
 - 1-2 Alimentation
 - GND et +5 V
 - V0: alimentation du rétro-éclairage
 - RS (Register select)
 - Valeur 0: indique le mode commande: exemple lcd.setCursor(0, 0);
 - Valeur 1: indique l'émission des caractères vers l'afficheur
 - R/W
 - Valeur 0 : mode écriture dans le registre du LCD
 - E: (Enable) LCD
 - Mis à 1 automatiquement durant la transmission des caractères
 - D7 à D0: bits de données étendues sur 8 bits
 - Sur 8 fils : D7 à D0 → un seul appel au microcontrôleur
 - Sur 4 fils : D4 à D7 → 2 appels au microcontrôleur, 4 bits à la fois
 - Rétro-éclairage
 - HIGH== allumé
 - LOW == éteint
- Tiré de: « Le grand livre d'Arduino - Eyrolles.pdf », Eric Bartman



Programme de test

- Mise en œuvre



```
/*
 * LCD RS pin to digital pin 12
 * LCD Enable pin to digital pin 11
 * LCD D4 pin to digital pin 5
 * LCD D5 pin to digital pin 4
 * LCD D6 pin to digital pin 3
 * LCD D7 pin to digital pin 2
 * LCD R/W pin to ground
 * LCD VSS pin to ground
 * LCD VCC pin to 5V
 */
#include <LiquidCrystal.h>

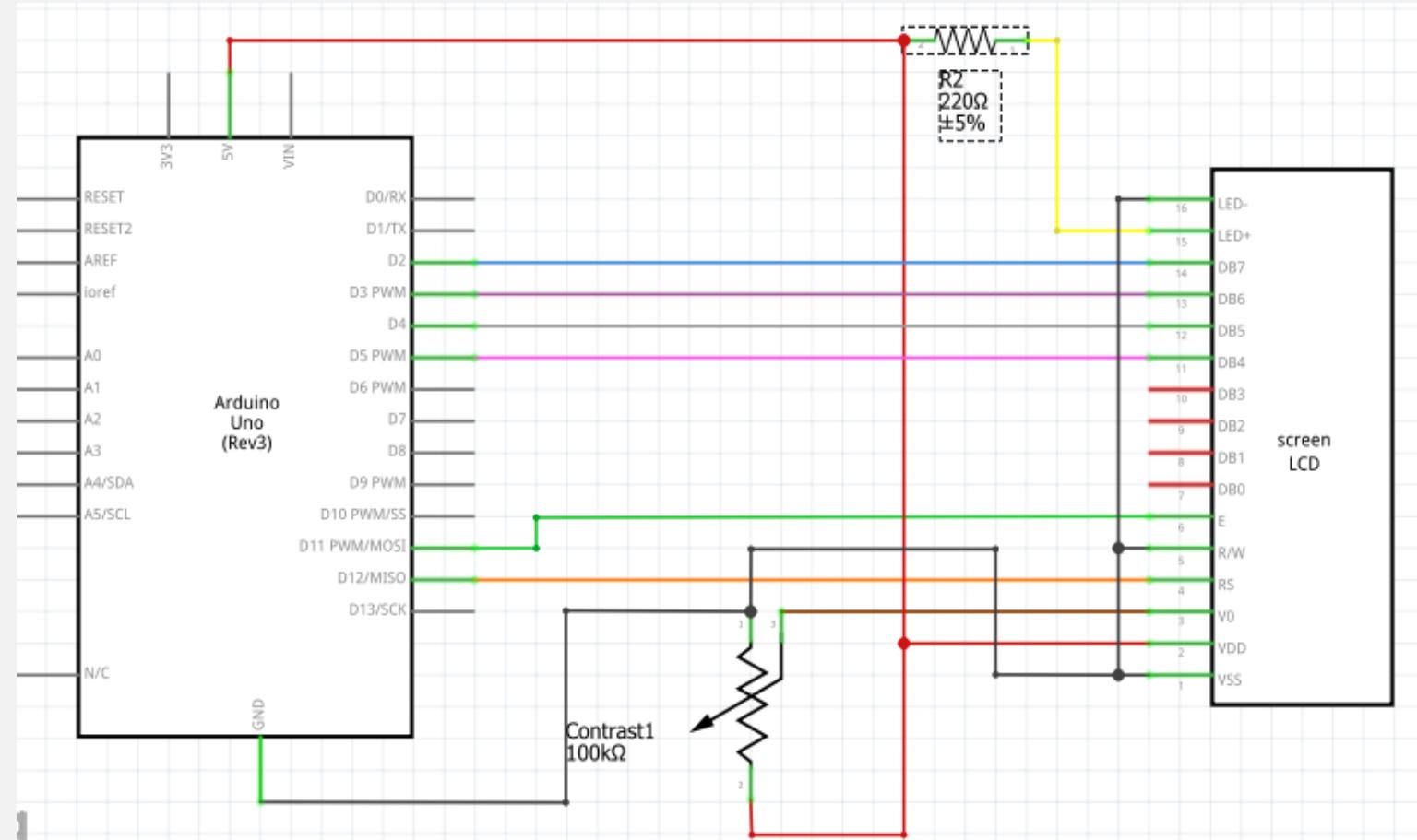
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
    // set up the LCD's number of columns and rows:
    lcd.begin(16, 2);

    // Print a message to the LCD.
    lcd.print("hello, world!");
}
```

Méthode filée

- Schéma du montage
 - Potentiomètre utilisé pour ajuster le rétro-éclairage (optionnel)

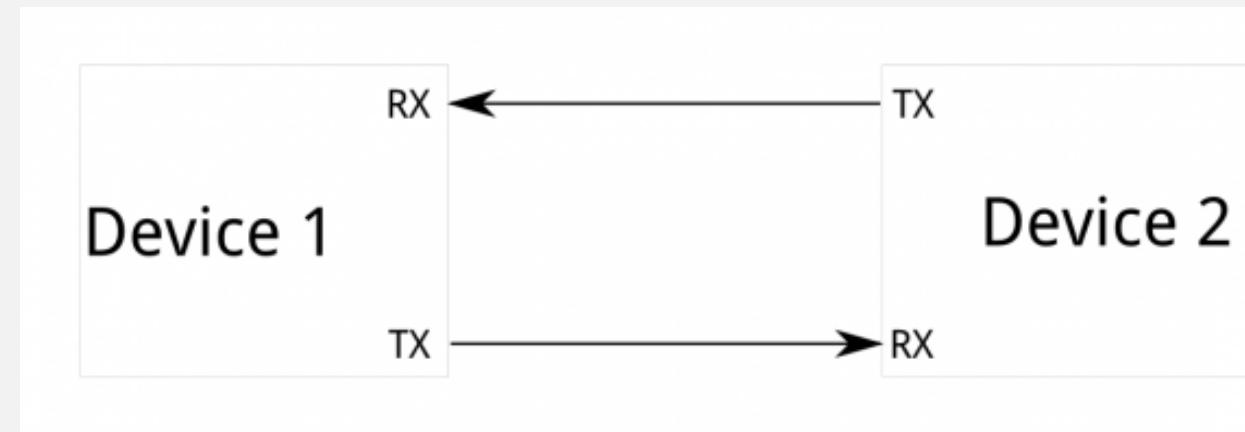


Mais plus j'avance, moins j'ai de sorties !

- Comment limiter le nombre de bornes utilisées ?
⇒ Utilisation d'un protocole de communication !
- Quand on parle de communication sur ce genre de plateforme, on trouve communément :
 - UART
 - SPI
 - I2C

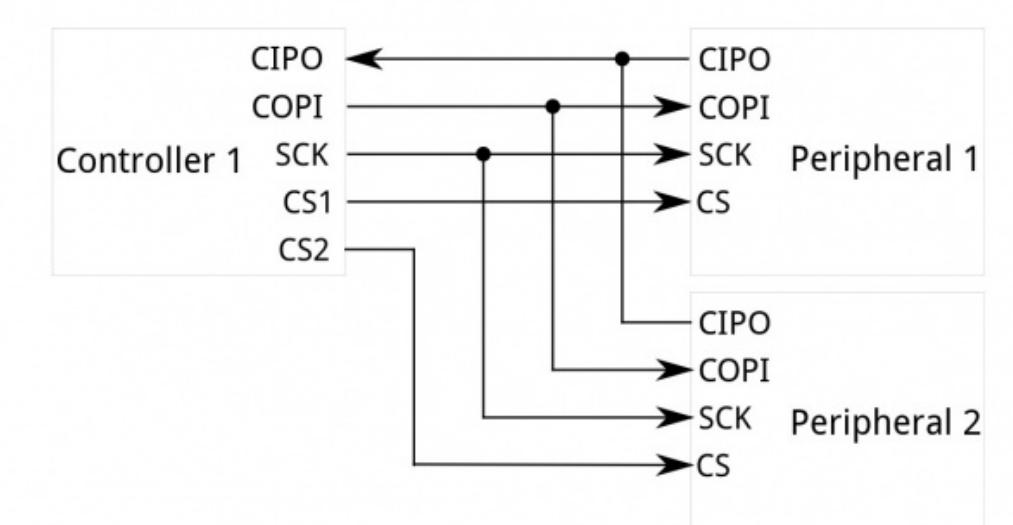
UART - Universal Asynchronous Receiver Transmitter

- Communication série : 1 bit à la fois
- Les deux périphériques doivent se mettre d'accord sur la vitesse de transmission
- 10 bits pour en transmettre 8
- Généralement limité à la communication de deux périphériques
- 230400 bits / s
- Bornes Arduino Uno : 0 et 1 de (\leftrightarrow) aux ports COM sur les anciens PCs)



SPI - Serial Peripheral Interface

- Communication série : 1 bit à la fois
- 3 bornes + 1 borne / périphérique
- Très rapide (jusqu'à 10 Mbits / s)
- Bornes Arduino Uno : COPI (11), CIPO (12), SCK (13), CS (ex. : 10)



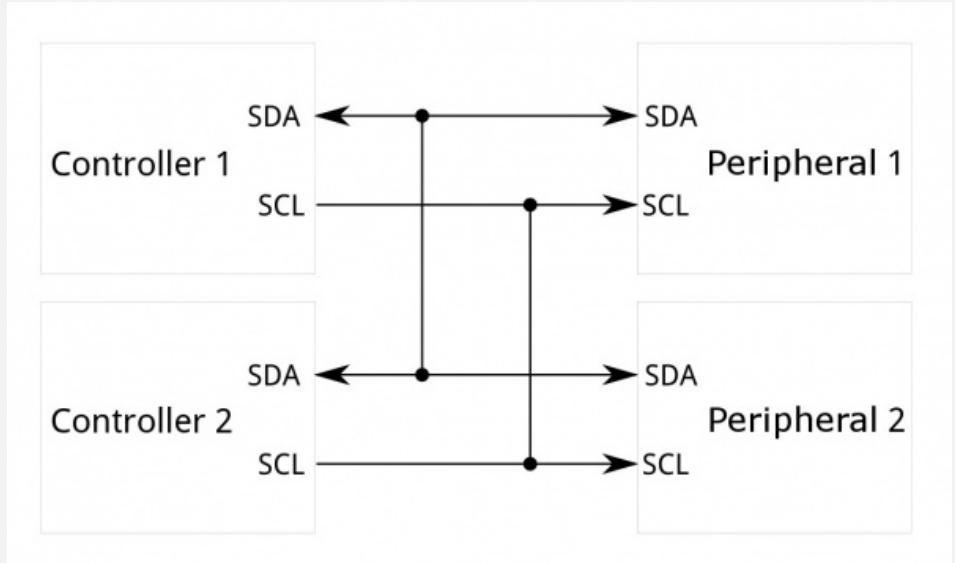
On trouvait anciennement la terminologie MISO (CIPO) et MOSI (COPI) avec M : master et S : slave.

Dépendamment du fabricant on peut aussi trouver :

- M pour main et S pour second
- I pour initiator et R pour responder
- Etc.

I2C - Inter-Integrated Circuit

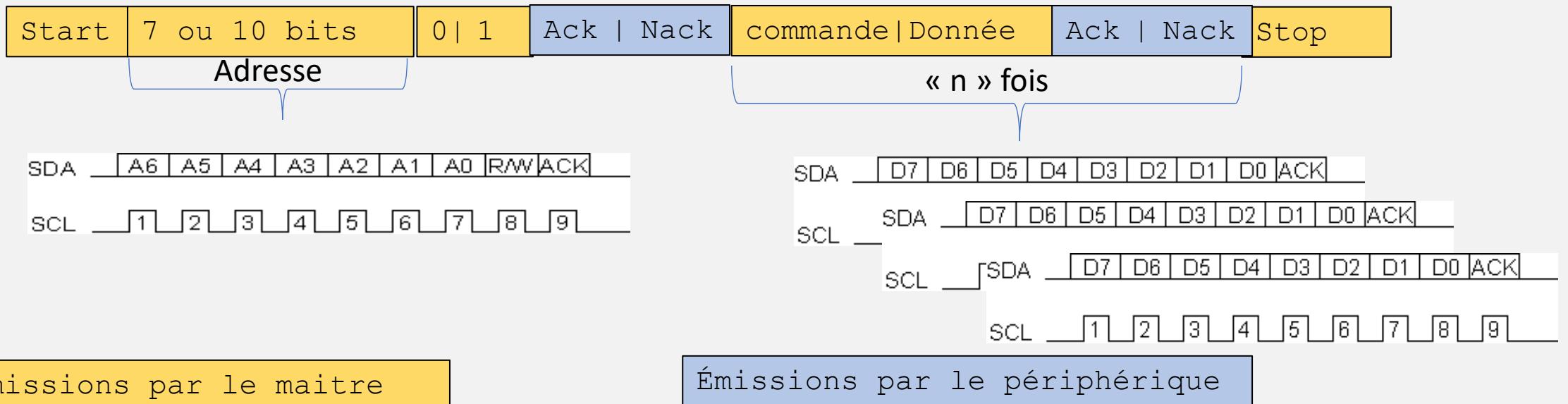
- Communication série : 1 bit à la fois
- Chaque périphérique possède une adresse pouvant s'étendre sur 7 bits (10 bits en mode étendu)
- Le contrôleur construit une trame qu'il envoie sur le lien SDA
- Bornes Arduino Uno SDA (A4), SCL (A5)



Les périphériques captent les messages si l'adresse incluse dans la trame correspond à un périphérique,
celui-ci prend action sur le message
Les autres périphériques ignorent le message

Méthode I2C

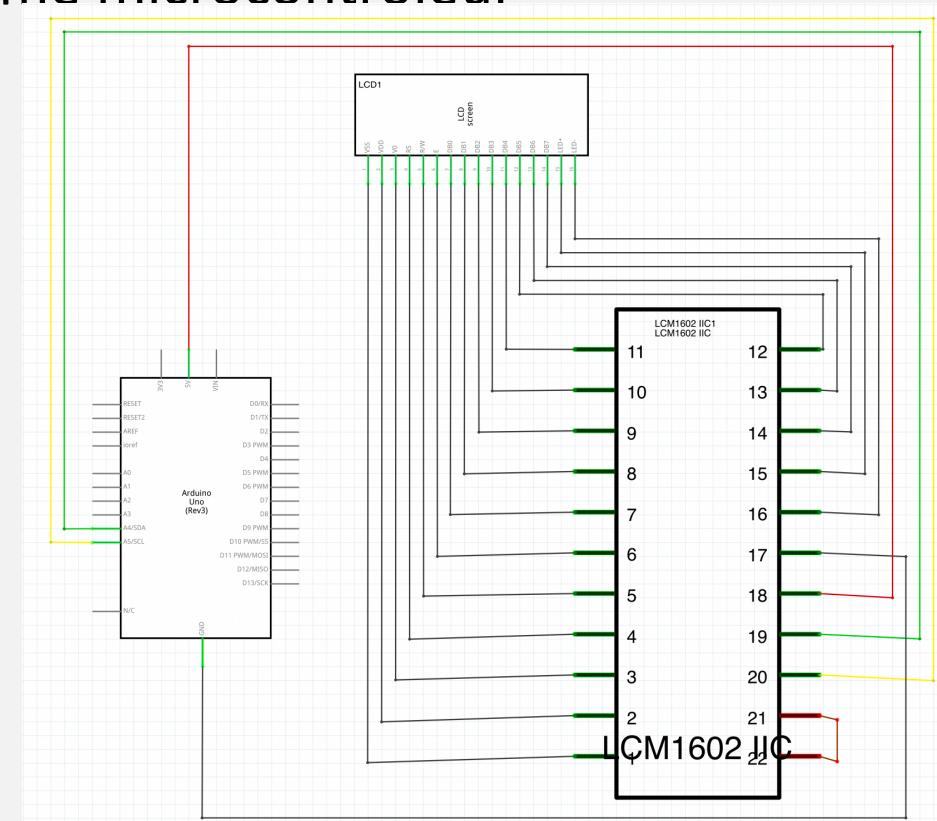
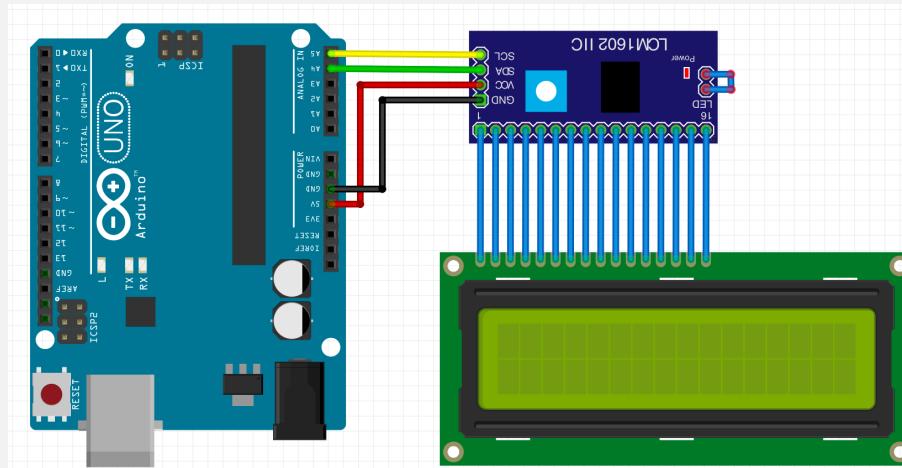
- Principe simplifié
 - Chaque trame est composé d'octets contenant l'adresse du périphérique, le « sujet » (0 = commande; 1 = donnée), les informations et autres informations
 - Durant la transmission, le périphérique concerné valide réception du message
 - À la fin, le contrôleur envoie un signal « fin de transmission »



I2C - LCD

- Buts

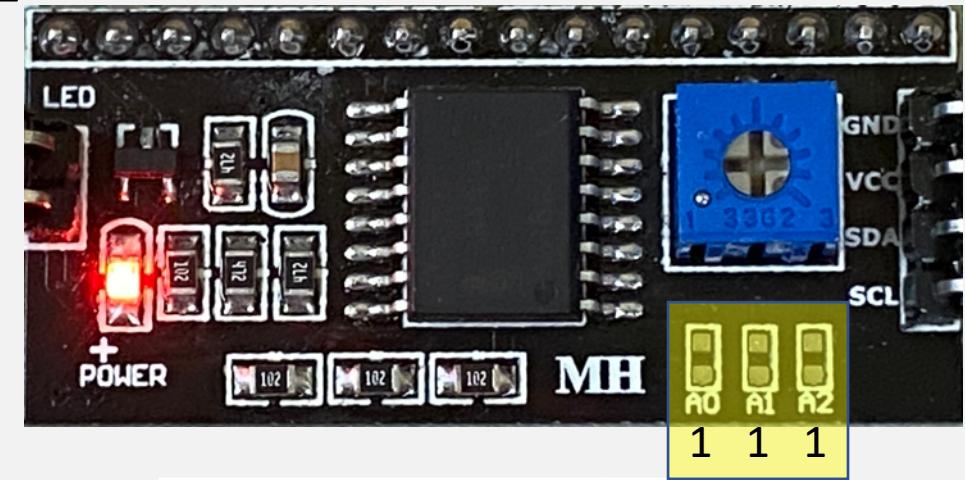
- Implémenter plusieurs périphériques sur un même microcontrôleur
- 4 fils suffisent pour un écran LCD
 - Alimentation: GND et + 5V
 - Noté SDA : fils de transmission des données en série
 - Noté SCL : fils des pulsations de synchronisation



Méthode I2C

- Mise en œuvre du modèle 1602A: LiquidCrystal_I2C.h
 - Adresse soudée sur 3 bits marquées A0, A1, A2
 - Idée pouvoir avoir n périphériques
 - Programmation identique à la méthode Filée en changeant la classe utilisée

```
10
11 //initialize the liquid crystal library
12 //the first parameter is the I2C address
13 //the second parameter is how many rows are on your screen
14 //the third parameter is how many columns are on your screen
15 LiquidCrystal_I2C lcd(0x27, 16, 2);
16
17 void setup() {
18
19     //initialize lcd screen
20     lcd.init();
21     // turn on the backlight
22     lcd.backlight();
23 }
```



Adresse	A2	A1	A0
0x20	0	0	0
0x21	0	0	1
...			
0x27	1	1	1

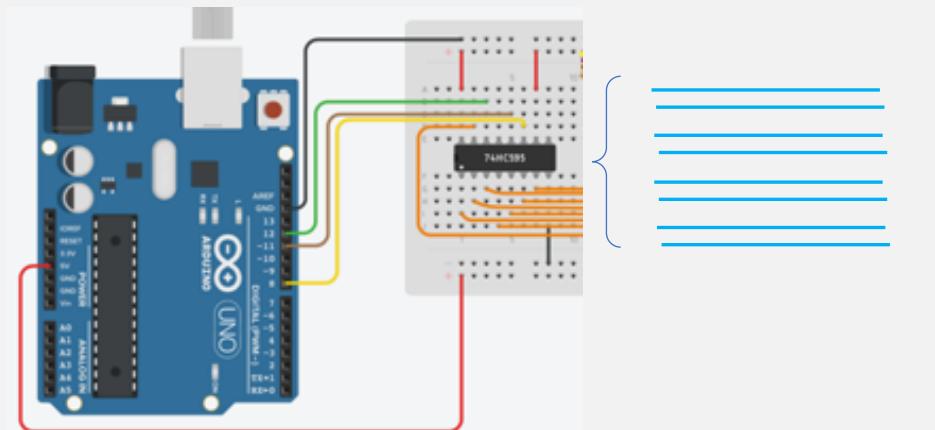
Résumé

- Mise en œuvre d'un écran LCD 1602A
 - Branchement filé
 - 6 bornes au minimum
 - Contrôle et transfère des messages par l'objet `LiquidCrystal.h`
 - Protocole I2C
 - Seulement 2 bornes
 - Contrôle et transfère des messages par l'objet `LiquidCrystal_I2C.h`
 - Possibilité de communiquer avec plusieurs périphériques sans utiliser de nouvelles bornes

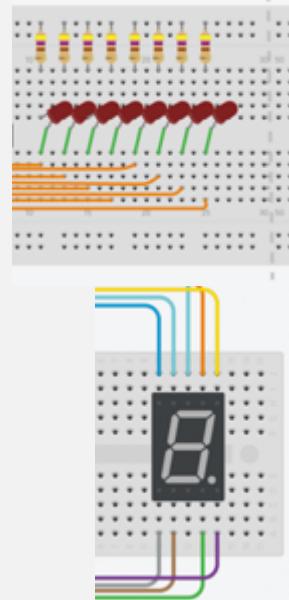
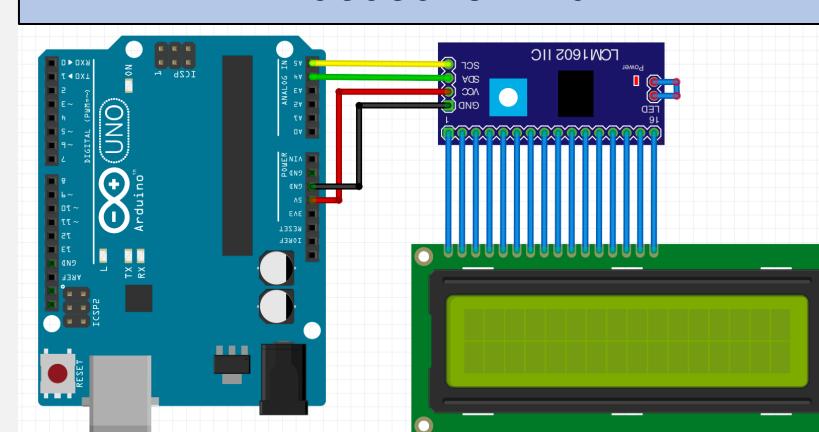
Résumé général

- Usage réduit de fils au microcontrôleur
 - 4 fils en entrée, 8 fils en sortie
- Éclairage de DELS
 - Registre à décalage
- Écran LCD
 - Protocole I2C

Registre à décalage



Protocole I2C



Références

- <https://www.tinkercad.com/things/6aCJ66So2yC> : Programme de test et montage de base
- <https://www.ardumotive.com/i2clcden.html> : méthode I2C
- https://www.engineersgarage.com/knowledge_share/character-lcd-pinout-working/: documentation explicite
- <https://www.robot-electronics.co.uk/i2c-tutorial>: trames du protocole (avancée)
- <https://fr.wikipedia.org/wiki/I2C>: description du protocole
- <https://www.arduino.cc/en/Reference/LiquidCrystal> : documentation de la bibliothèque LiquidCrystal
- <https://github.com/arduino-libraries/LiquidCrystal> : code source de la bibliothèque LiquiCrystal
- <https://platformio.org/lib/show/887/LiquidCrystal/installation> : ajout dans PIO
- https://platformio.org/lib/show/576/LiquidCrystal_I2C/installation : ajout dans PIO, version I2C