

Apprendre à afficher des messages et des grandeurs numériques sur un afficheur LCD alpha-numérique avec Arduino.



Ateliers Arduino

par X. HINAULT

www.mon-club-elec.fr



Tous droits réservés – 2012.

Ce document légèrement payant est soumis au droit d'auteur et est réservé à l'usage personnel.

Afin d'encourager la production de supports didactiques de qualité, ce document est légèrement payant.

La licence d'utilisation est attribuée pour un usage personnel uniquement, dans le cercle familial. Mise en ligne et diffusion non autorisées.

Si vous n'êtes pas le détenteur de la licence attribuée pour l'usage de ce document, soyez sympa, merci d'acheter votre exemplaire personnel ici : <https://monclubelec.dpdcart.com/>

Pour tout problème lié à l'utilisation de ce document, veuillez envoyer une copie ici : support@mon-club-elec.fr

Pour obtenir tout autres types de licence d'utilisation (enseignement, commercial, etc...), veuillez contacter l'auteur ici : support@mon-club-elec.fr

Vous avez constaté une erreur ? une coquille ? N'hésitez pas à nous le signaler à cette adresse : support@mon-club-elec.fr

Truc d'utilisation : visualiser ce document en mode diaporama dans le visionneur PDF. Navigation avec les flèches HAUT / BAS ou la souris.

En mode fenêtre, activer le panneau latéral vous facilitera la navigation dans le document. Bonne lecture !

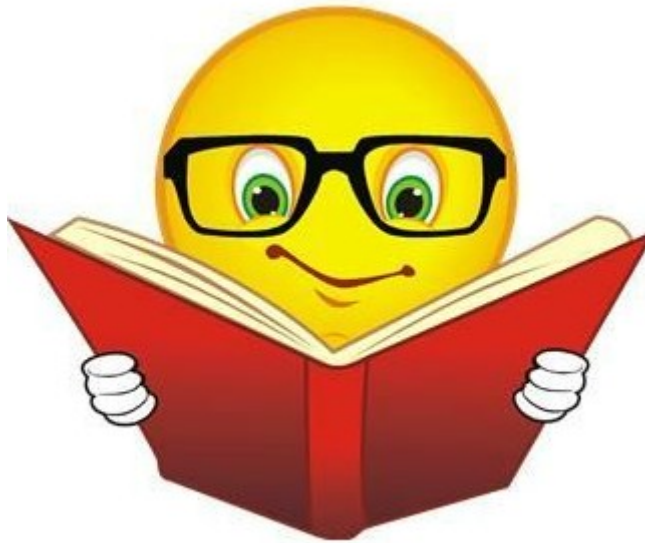
Lancer également le logiciel Arduino et connecter votre carte Arduino afin de pouvoir tester au fur et à mesure les codes d'exemples !

1. Intro

L'objectif ici est :

- d'écrire des programmes affichant des messages, des nombres
- de contrôler l'afficheur LCD depuis le terminal Série
- d'afficher le résultat de mesures et des résultats de calculs sur un afficheur LCD

... afin d'être en mesure de réaliser des applications utilisant un afficheur LCD pour afficher des messages

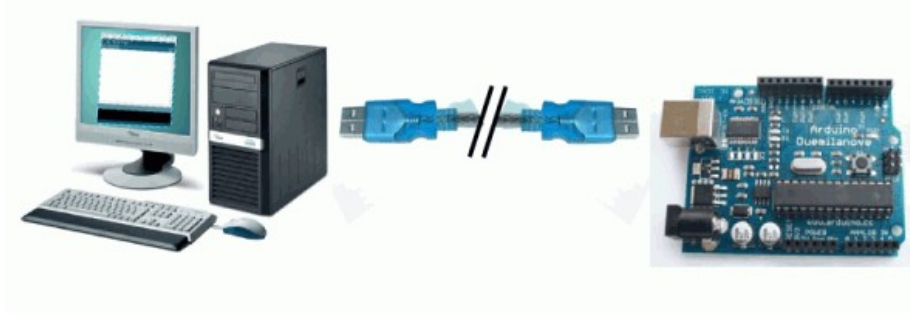


Prêt ? C'est parti !

2. Matériel nécessaire pour les ateliers Arduino

Pour cet atelier, vous aurez besoin de tout ou partie des éléments suivants pour pouvoir réaliser les exemples proposés :

De l'espace de développement Arduino

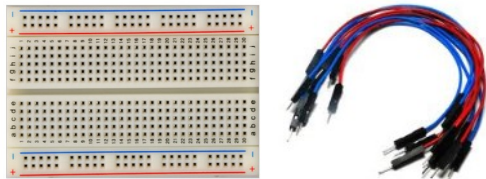


L'espace de développement Arduino associe :

- un ordinateur sous Windows, Mac Os X ou Gnu/Linux (Ubuntu)
- avec le logiciel Arduino installé (voir : <http://www.arduino.cc/>)
- un câble USB
- une carte Arduino UNO ou équivalente.

disponible chez : <http://shop.snootlab.com/> ou <http://www.gotronic.fr/>

Du nécessaire pour réaliser des montages sans soudure

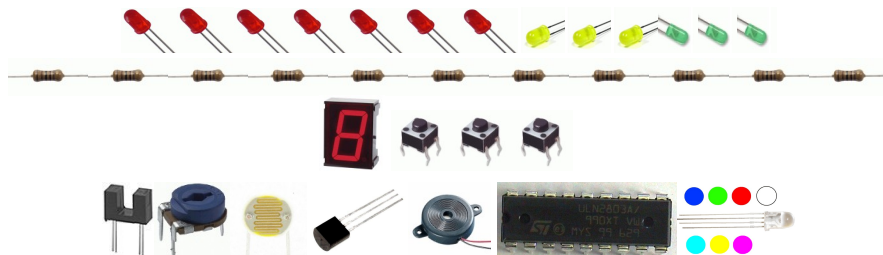


Pour réaliser des montages sans soudure, vous aurez besoin :

- d'une plaque d'essai ou breadboard moyenne (450 points)
- de quelques câbles souples (ou jumpers) mâle/mâle

disponible chez : <http://www.gotronic.fr/>

De quelques composants de base



Pour vous simplifier la vie, nous avons négocié ce kit pour vous !

Vous pouvez commander ce kit complet directement en 1 clic chez notre partenaire

<http://www.gotronic.fr/> avec le code express **701710**

GO TRONIC
ROBOTIQUE ET COMPOSANTS ÉLECTRONIQUES

Pour plus de détails, voir : http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERS

Pour les ateliers Arduino niveau débutant, vous devrez idéalement disposer des composants suivants :

- des LEDs 5mm Rouges(x20), Vertes (x5) et 3 Jaunes (x5)
- digit à cathode commune rouge 13mm (x1)
- Résistances (1/4w - 5%) de 270 Ohms (x20), 4,7K Ohms (x1), 1K Ohms (x1)
- mini bouton-poussoir (x3)
- Opto-fourche (x 1)
- Résistance variable linéaire 10K (x 1)
- Photo-résistance 7mm (x 1)
- Capteur de température LM35DZ (-55/+150°C - 10mV/°C) (x 1)
- Capsule son piézoélectrique (x 1)
- ULN 2803A (CI amplificateur 8 voies, 500mA/ voie) (x 1)
- LED 5mm multicolore RVB cathode commune (x 1)

3. Matériel spécifique nécessaire pour cet atelier

Pour cet atelier vous aurez besoin également :

D'un afficheur LCD alpha-numérique standard 4 lignes x 20 colonnes

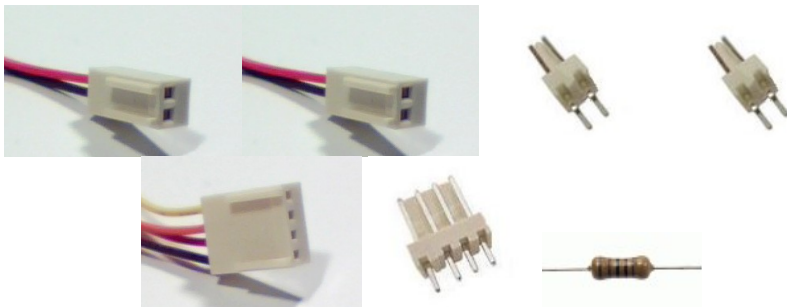


L'afficheur LCD alphanumérique standard 4 lignes x 20 colonnes est facile à utiliser avec la carte Arduino à l'aide de 6 broches numériques. Cet afficheur permet d'afficher des lettres et des chiffres sur 4 lignes et 20 colonnes. Un équipement idéal pour créer des applications autonomes réalisant des mesures, etc...

Existe en différents formats 2x8, 1x16, 2x16, etc... Le format 4x20 présente l'avantage d'être « à l'aise » pour afficher ses messages. Existe également en mode réflectif ou avec rétro-éclairage.

disponible chez : <http://www.gotronic.fr/> | De 8 à 20€ selon modèle
4x20 : code 03351 | 2x16 : code 03313

Du matériel nécessaire pour « préparer » un afficheur standard pour une utilisation simplifiée avec Arduino



Pour être utilisable facilement avec Arduino, l'afficheur LCD standard brut nécessite une petite préparation assez simple à l'aide d'éléments de connectique simples et une résistance :

- 2 connecteurs femelles sur fils – 2 contacts (code : 09825)
- 1 connecteur femelle sur fils – 4 contacts (code : 09827)
- 2 connecteurs droits – 2 contacts (code : 09815)
- 1 connecteur droit – 4 contacts (code : 09817)
- 1 résistance 1KOhms – 1/4w (code : 04036)

disponible chez : <http://www.gotronic.fr/> avec les codes indiqués

Des outils nécessaires pour réaliser des soudures



Pour réaliser des soudures, vous aurez besoin :

- d'un fer à souder à pointe fine, 25-30W et de son support
- de soudure d'étain 60% dite « électronique » - Ø 1mm
- d'un rouleau de scotch (pratique pour faire tenir les composants)
- d'une petite pince coupante
- d'une pompe à dessouder (pour rattraper le coup au besoin)

disponible chez : <http://www.gotronic.fr/> ou en magasin de bricolage

La préparation de l'afficheur LCD est présentée dans l'atelier précédent consacré aux afficheurs LCD alpha-numériques standards

4. Rappel : Fiche Technique : Afficheur LCD alpha-numérique standard

Description

Un afficheur alpha-numérique standard est un composant que vous connaissez bien et qui équipe toutes sortes de dispositifs de la vie courante. C'est un module qui permet d'afficher des messages à base de lettres et de chiffres assez simplement avec Arduino.



Selon les modèles, il existe des variantes, notamment :

- réflectif ou rétro-éclairé (consomme plus)
- avec LED de rétro-éclairage (utilisable dans l'obscurité)
- couleur de l'affichage : soit noir sur fond vert classique, soit blanc sur fond bleu, etc...

Il existe différentes tailles également :

- en 4 lignes x 20 colonnes, en 2 lignes x 8 colonnes, en 1 ligne x 16 colonnes.
- en pratique, un 4 lignes x 20 colonnes permet d'être à l'aise, et c'est celui que je conseille. Compter 20€ pièce. Les autres modèles sont moins chers, dès 8€.

Important

Toutes les variantes d'afficheurs dits « standards » (comme sur la photo) sont utilisables avec Arduino assez simplement comme nous allons le voir, jusqu'à 80 caractères (soit maximum 4 x 20) .

Ne pas confondre les afficheurs standards avec les afficheurs LCD « série » souvent vendus plus chers pour « économiser des broches » et nécessitant une librairie de communication spécifique. Nous ne traitons pas de ces afficheurs ici car ils sont particuliers à tel ou tel fabricant et pas toujours universels. Un afficheur LCD standard n'utilise que 6 broches numériques, ce qui n'est quand même pas la « mer à boire »... et ne justifie pas la différence de prix.

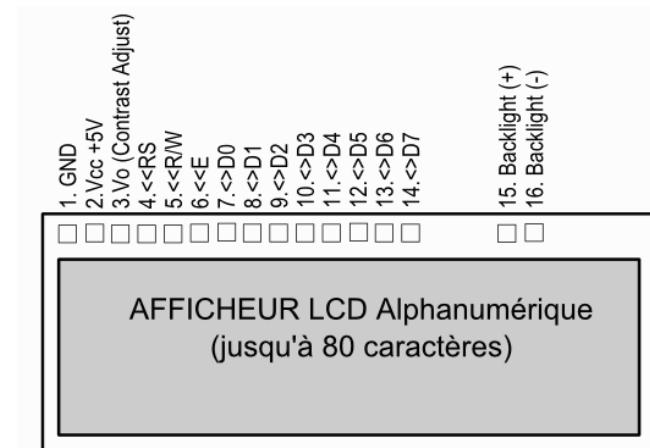
Brochage

Un afficheur LCD standard dispose typiquement :

- de 2 broches d'alimentation et d'une broche de réglage du contraste
- de 3 broches numériques de commande RS, E et RW
- de 8 broches numériques de données notées D0 à D7
- +/- de 2 broches correspondant à la LED de rétro-éclairage

Caractéristiques électriques

- Un afficheur LCD standard nécessite une tension d'**alimentation** typiquement de 5V et va consommer au plus quelques dizaines de mA : il sera donc directement utilisable et alimentable par le +5V de la carte Arduino (**pour mémoire, cette alimentation laisse 300mA dispo**).
- L'afficheur dispose par ailleurs de plusieurs **broches numériques** de contrôle qui sont de type numérique et connectables directement à la carte Arduino.
- Certains modèles enfin disposent d'une **LED interne de rétro-éclairage** qui permet d'utiliser le LCD dans l'obscurité. A utiliser comme une LED classique (càd avec une résistance en série) .



Les broches de l'afficheur LCD standard : ne vous laissez pas impressionner, c'est simple !

Mode de fonctionnement

Un afficheur LCD alpha-numérique standard peut fonctionner :

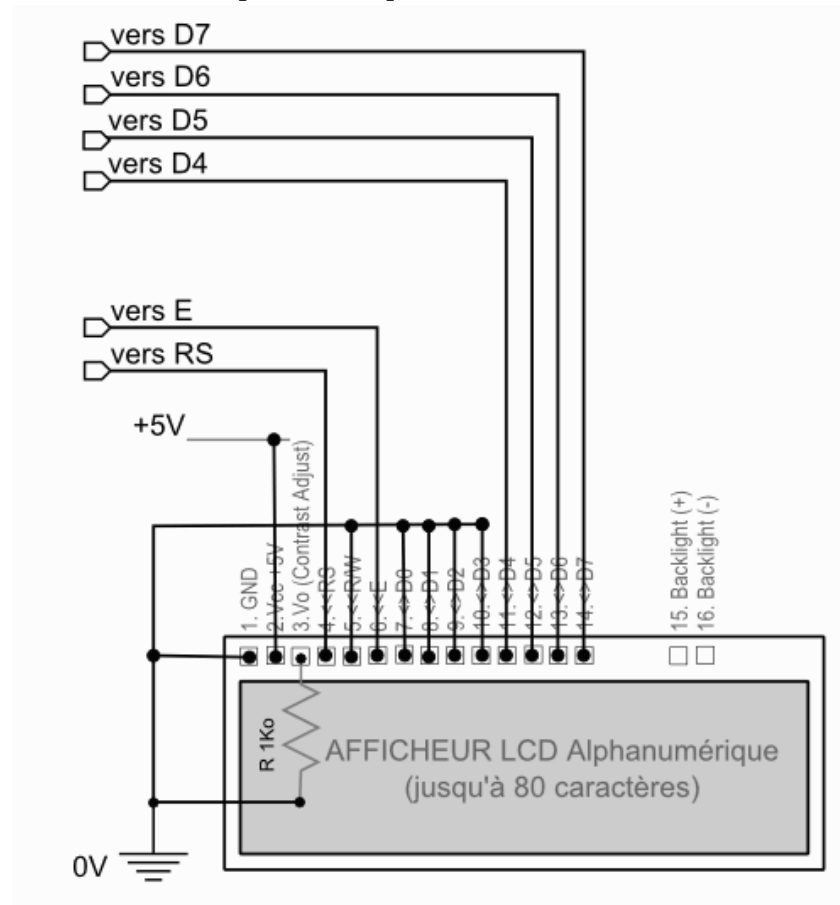
- soit en mode dits « 8 bits » et dans ce cas nécessite 8 broches de données + 3 broches de commandes soit 11 broches !
- soit en mode dits « 4bits » et dans ce cas nécessite 4 broches de données + 2 broches de commandes soit seulement **6 broches**. **C'est ce mode de fonctionnement que nous allons utiliser.**

5. Rappel : Préparation d'un afficheur LCD pour une utilisation simplifiée avec Arduino : le schéma théorique

Comme on vient de la dire, un afficheur LCD peut être utilisé en mode "4 bits" ou "8 bits". Dans le mode simplifié, dit « 4 bits », on n'utilise que 4 lignes de données pour envoyer les données à l'afficheur. C'est ce mode qui est le plus pratique. Les connexions à réaliser sont alors les suivantes :

- broches de commande **RS** et **E** connectées à **2 broches numériques en sortie** de la carte Arduino
- broches de données **D4** à **D7** connectées à **4 broches numériques en sortie** de la carte Arduino
- Le **+** et **-** connectées au **5V** et à la **masse (0V)**
- Une résistance de réglage du contraste entre le +5V et la broche Vo (en pratique 1Kohm – 1/4w fait l'affaire). On pourrait aussi utiliser une résistance variable, mais c'est plus compliqué ici, surtout qu'il suffit d'incliner plus ou moins l'afficheur pour régler le « contraste » apparent...
- la broche **RW** et les broches **Do - D3** non utilisées et connectées à la **masse (=0V)**.
- enfin, si l'afficheur intègre une LED de rétroéclairage, on la connectera comme une LED classique (pas utilisée ici).

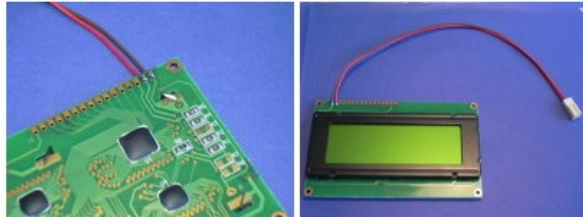
Voici le schéma de cette connexion simplifiée de l'afficheur LCD alpha-numérique :



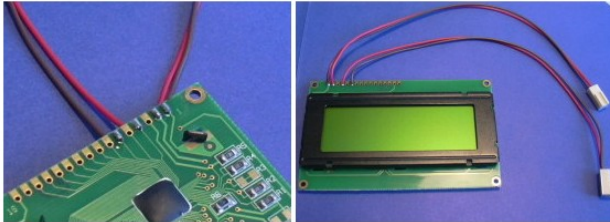
6. **Rappel : Préparation d'un afficheur LCD pour une utilisation simplifiée avec Arduino : description concrète**

La préparation de l'afficheur n'est pas très compliquée à réaliser et permettra ensuite d'utiliser très facilement l'afficheur avec une carte Arduino. Voici la procédure en images.

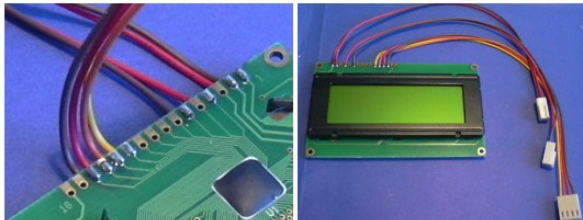
On commence par souder le connecteur 2 broches sur fils sur le + et - de l'afficheur :



On soude ensuite le connecteur 2 broches sur fils sur les broches RS et E :



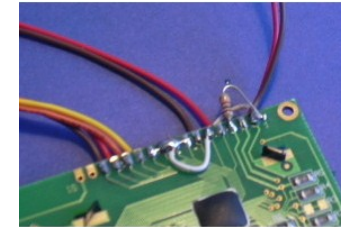
On soude ensuite le connecteur 4 broches sur fils sur les broches D4 à D7 :



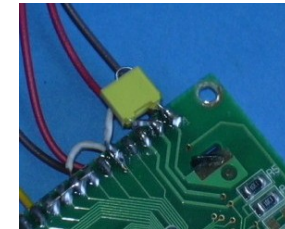
On soude ensuite entre-elles les broches D0 à D3, la broche RW et la broche 0V



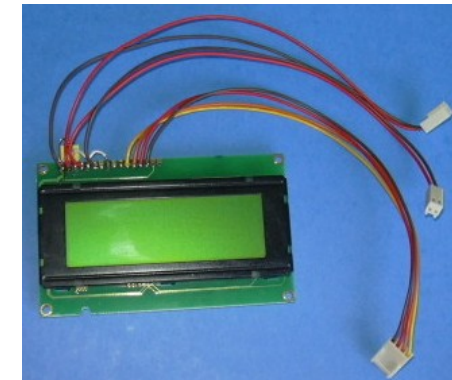
On soude la résistance de 1Ko entre le 0V et la broche Vo



On peut enfin souder un condensateur 100nF entre le + et le - (pas indispensable... limite les « parasites », c'est tout.)



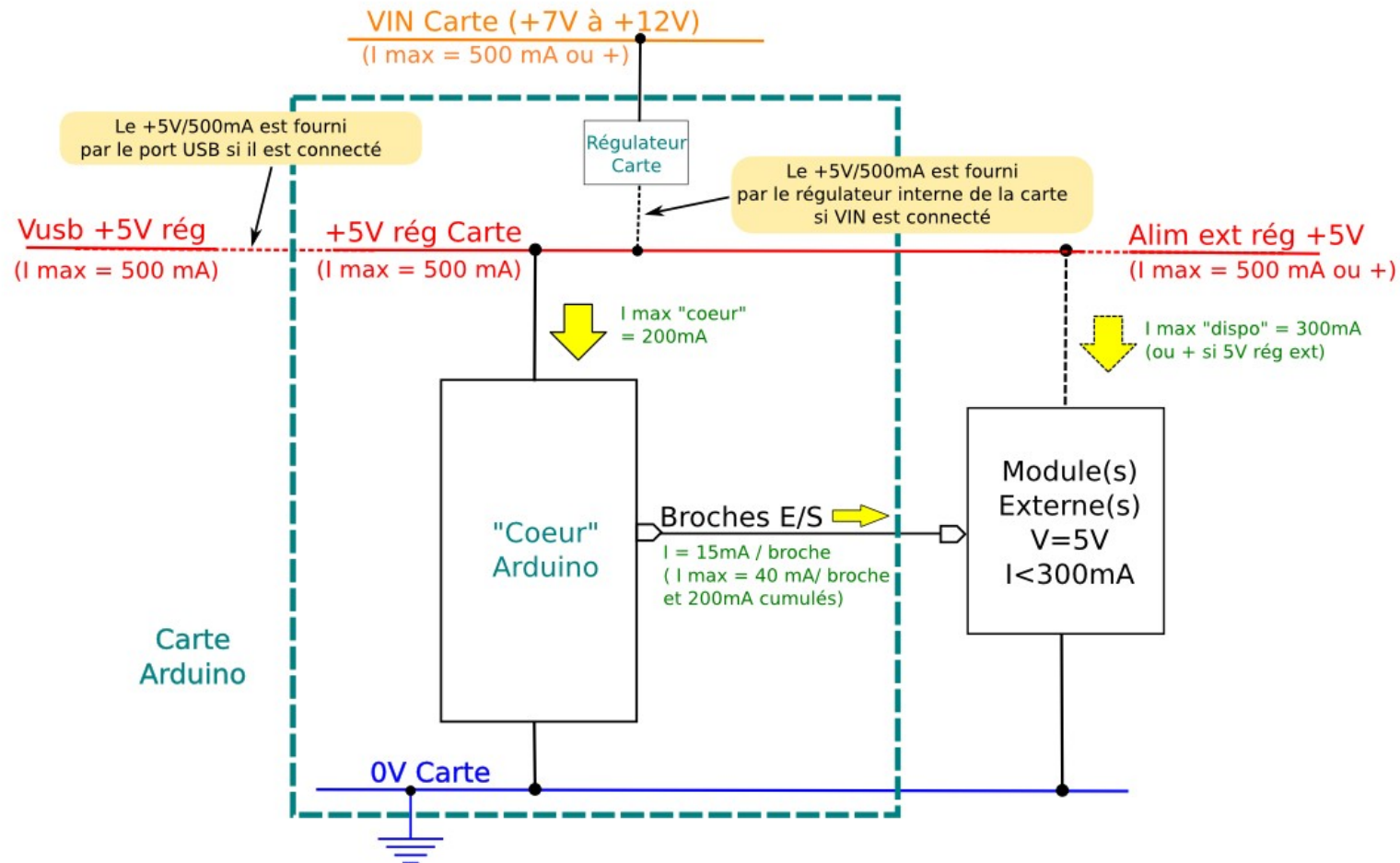
Voici à quoi ressemble le LCD préparé fini et prêt à l'emploi avec votre carte Arduino :



Rien de bien sorcier.. Cette fois, vous êtes prêts à utiliser votre LCD !

7. Rappel : Schéma électrique type d'utilisation d'un afficheur LCD avec une carte Arduino

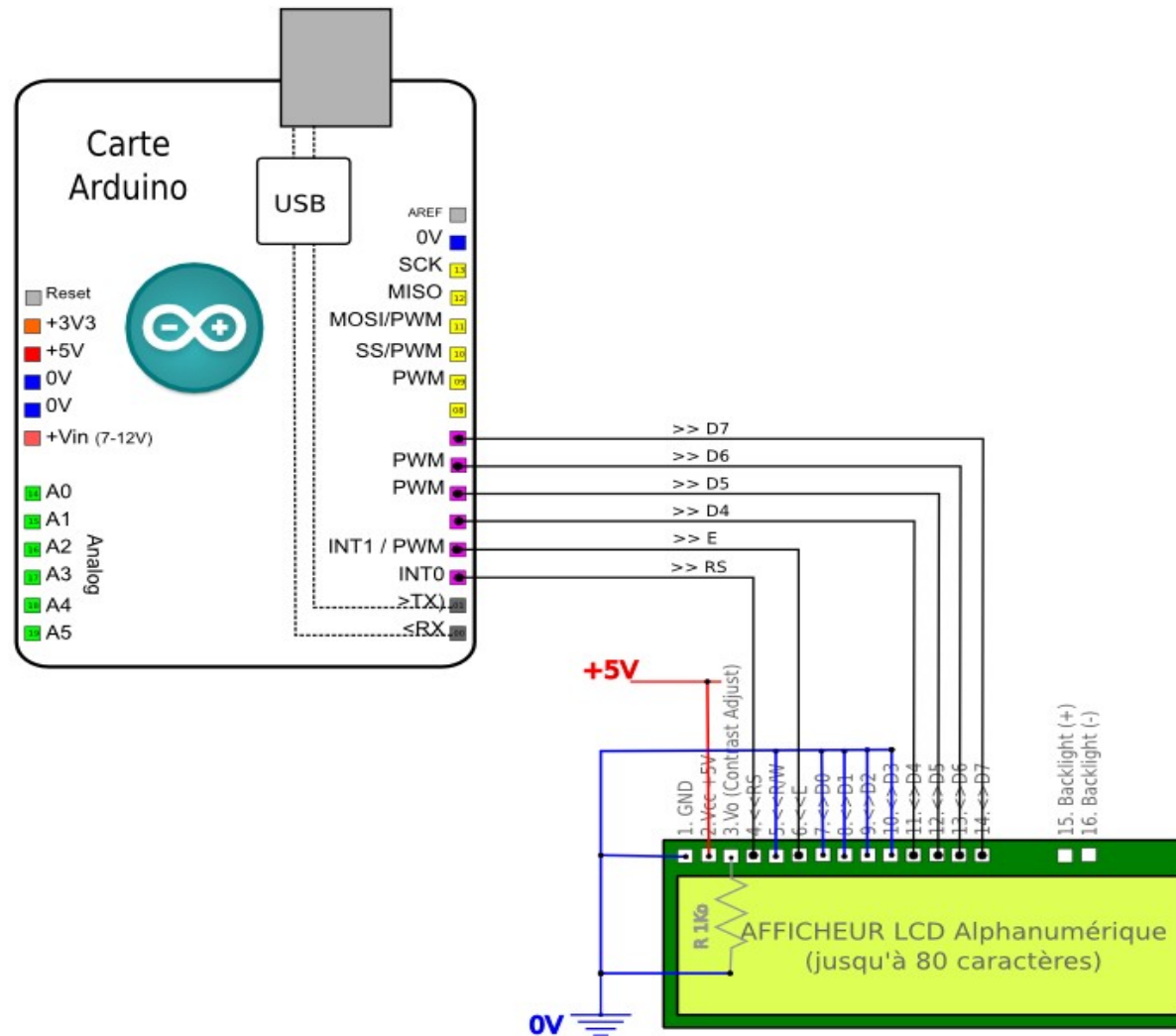
L'afficheur LCD ne va demander que quelques mA et est alimentable en 5V : on va donc pouvoir l'alimenter directement sur l'alimentation 5V de la carte Arduino (qui rappelons-le peut fournir 500mA – 200mA (conso du coeur Arduino) = 300mA disponibles environ).



8. Utilisation d'un afficheur LCD « préparé » avec une carte Arduino : le montage

Le montage consiste à connecter :

- les 2 broches de commande RS et E sur 2 broches numériques Arduino,
- les 4 broches de données D4 à D7 sur 4 broches numériques Arduino,
- le +5V et le 0V

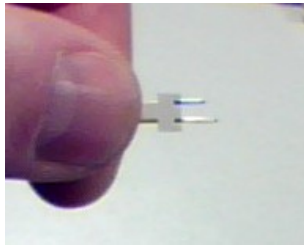


9. Utilisation d'un afficheur LCD préparé avec une carte Arduino : en images

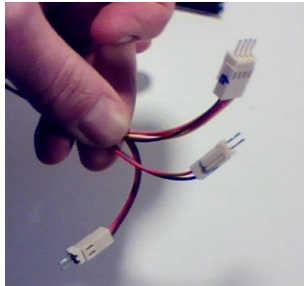
Commencer par prendre les connecteurs droits pour CI :



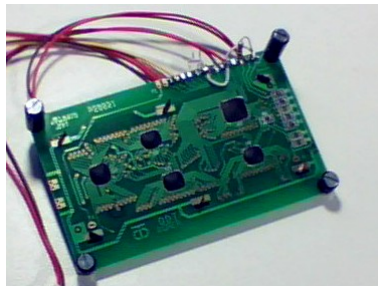
et à l'aide d'une pince, ressortir les broches droites de manière à ce qu'elles soient plus longues en partie extérieure :



Une fois fait pour tous les connecteurs droits, les mettre en place sur les connecteurs femelles sur les fils de l'afficheur LCD :

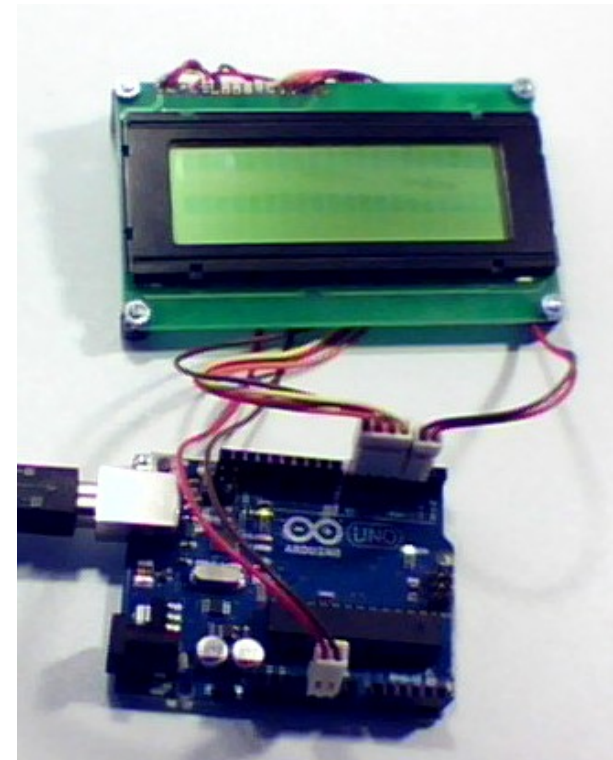


Dernière petite chose : on pourra mettre l'afficheur « sur pieds » à l'aide de 2 entretoises 5mm à l'avant et 2 entretoises 15mm à l'arrière de façon à ce qu'il soit légèrement incliné vers l'avant une fois posé sur ses pieds.



Une fois fait, il devient très facile et pratique d'utiliser l'afficheur LCD avec la carte Arduino :

- enficher le connecteur du +5V et 0V dans le connecteur 0V/+5V de la carte Arduino
- enficher le connecteur RS/E dans le connecteur des broches 2-3 de la carte Arduino,
- enficher le connecteur D4-D7 dans le connecteur des broches 4-7 de la carte Arduino.



Noter qu'un tel circuit pourra facilement être intégré dans un boîtier si l'on souhaite rendre l'application autonome.

10. Rappel : Langage Arduino : la librairie **LiquidCrystal** pour le contrôle des afficheurs LCD standards

Présentation

- Cette librairie Arduino permet à une carte Arduino de contrôler un afficheur LCD alphanumérique standard à cristaux liquides basé sur le circuit intégré Hitachi HD44780 (ou compatible), ce qui est le cas de la plupart des afficheurs alphanumériques LCD disponibles.
- La librairie fonctionne aussi bien en mode 4 bits qu'en mode 8 bits (càd utilisant 4 ou 8 broches numériques en plus des broches de contrôle RS, Enable et RW (optionnel)). Ainsi, en mode 4 bits, 6 broches numériques de la carte Arduino suffisent pour contrôler un afficheur LCD alphanumérique.

Inclusion

La librairie s'intègre dans un programme avec la ligne (pas de ; !!) :

```
#include <LiquidCrystal.h> // inclut la librairie Servo
```

Le constructeur de la classe

Le constructeur de la classe existe sous 4 formes correspondant à différents brochages possibles. Avec 6 broches, on utilisera la forme suivante :

```
LiquidCrystal lcd(rs, enable, d4, d5, d6, d7) ; // mode 4 bits - RW non connectée (le plus simple!)
```

avec :

- rs : broche numérique connectée à la broche RS de l'afficheur
- enable : broche numérique connectée à la broche E de l'afficheur
- d4 à d7 : broches numériques connectées aux broches D4 à D7 de l'afficheur.

Les fonctions de la librairie

La librairie dispose de nombreuses fonctions, à savoir :

- Fonctions d'initialisation : [begin\(\)](#)
- Fonctions d'écriture : [print\(\)](#) | [write\(\)](#)
- Fonctions de gestion de l'écran : [clear\(\)](#) | [display\(\)](#) | [noDisplay\(\)](#) |
- Fonctions de positionnement du curseur : [home\(\)](#) | [clear\(\)](#) | [setCursor\(\)](#)
- Fonctions modifiant l'aspect du curseur : [cursor\(\)](#) | [noCursor\(\)](#) | [blink\(\)](#) | [noBlink\(\)](#)
- Fonctions de contrôle du comportement du curseur : [autoscroll\(\)](#) | [noAutoscroll\(\)](#) | [leftToRight\(\)](#) | [rightToLeft\(\)](#)
- Fonctions d'effets visuels : [scrollDisplayLeft\(\)](#) | [scrollDisplayRight\(\)](#)
- Fonction de création de caractère personnalisé : [createChar\(\)](#)

Pour le détail complet des fonctions de la librairie, voir :

http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.LibrairieLCD

Principe d'utilisation

La structure type d'un programme utilisant un afficheur LCD va être la suivante :

Au niveau de l'entête déclarative

- Inclusion de la librairie **LiquidCrystal**
- Déclaration des constantes des broches utilisées avec le LCD
- Création d'un objet **LiquidCrystal** que l'on appellera lcd typiquement. On précise à ce niveau les broches utilisées en utilisant les constantes de broches déclarées précédemment.

Truc : je vous conseille de déclarer toutes les broches utilisées pour le LCD avec le nom de leur fonction RS, E, D4, D5.... De cette façon, vous pourrez appeler le constructeur sous la forme lcd(RS,E,D4,D5,D6,D7) ;

Au niveau de la fonction **setup()**

- Initialisation de l'afficheur avec la fonction lcd.[begin](#)(colonnes, lignes) où colonnes et lignes sont le nombre de ligne et de colonne de l'afficheur.
- Prendre l'habitude d'initialiser l'affichage avec l'appel de la fonction lcd.[clear](#)()
- On peut également à ce niveau afficher un rapide message d'accueil avec la fonction lcd.[print](#)(« texte ») suivi d'un effacement du LCD avec la fonction lcd.[clear](#)()
- On peut également à ce niveau réaliser l'affichage des messages fixes qui ne changeront pas ensuite (nom des valeurs par exemple)

Au niveau de la fonction **loop()**

- A ce niveau on utilisera selon les besoins toutes les fonctions utiles de la librairie pour se déplacer sur l'afficheur, afficher des caractères, effacer des messages, etc...



Cette librairie est présentée en détail dans l'atelier précédent dédié aux afficheurs LCD.

11. Rappel : « Hello world » : afficher un premier message sur un afficheur LCD.

Histoire de se « remettre en jambe », je vous redonne ici le premier programme que nous avons utilisé avec un afficheur LCD : ce programme affiche simplement le message « Hello World » sur l'afficheur LCD : rien de bien compliqué, mais comme ça vous êtes sûr que ça marche !

Entête déclarative

- On commence par importer la librairie **LiquidCrystal** (cette librairie est présentée en détail dans l'atelier précédent dédié aux afficheurs LCD)
- Ensuite on déclare l'ensemble des 6 broches utilisées pour le contrôle de l'afficheur LCD
- On déclare un objet **LiquidCrystal** qui représentera le LCD dans le reste du programme.

Fonction **setup()**

- On commence par initialiser l'afficheur à l'aide de la fonction **begin**(colonnes,lignes). Ici, afficheur 20 colonnes x 4 lignes. A adapter à votre situation le cas échéant.
- On initialise l'affichage avec la fonction **clear**() qui efface l'écran et positionne le curseur (invisible par défaut) en 0,0 (colonne 0, ligne 0) c-à-d dans le coin supérieur gauche de l'écran.
- Puis on affiche tout simplement un message à l'aide de la fonction... **print**(« texte ») **qui affiche le texte à l'emplacement du curseur !**

Note : Remarquer au passage la cohérence du langage Arduino qui propose la fonction **print()** aussi bien pour la classe **Serial** que la classe **LiquidCrystal**. Cette fonction sera également disponible pour les classes gérant une carte mémoire SD ou encore le réseau ethernet. Facile et puissant !

Fonction **loop()**

- Laissée vide ici.

Fonctionnement

Une fois la carte Arduino programmée, le message s'affiche : cool !



```
// ateliers Arduino par X. HINAULT - Tous droits réservés - 2012
// licence GPL v3 - www.mon-club-elec.fr

// afficher un message simple sur un afficheur LCD standard

//--- entete déclarative ---
#include <LiquidCrystal.h> // inclusion de la librairie LCD

// déclaration des broches de l'afficheur
const int RS=2; // broche RS
const int E=3; // broche E
const int D4=4; // broche D4
const int D5=5; // broche D5
const int D6=6; // broche D6
const int D7=7; // broche D7

LiquidCrystal lcd(RS,E,D4,D5,D6,D7); // déclaration objet représentant lcd

//--- la fonction setup() : exécutée au début et 1 seule fois
void setup() {

    // écrire ici les instructions à exécuter au début
    lcd.begin(20,4); // initialise LCD colonnes x lignes

    lcd.clear(); // efface LCD + se place en 0,0
    lcd.print("Hello World"); // affiche le message

} // fin de la fonction setup()

//--- la fonction loop() : exécutée ensuite en boucle sans fin
void loop() {

    // écrire ici les instructions à exécuter en boucle

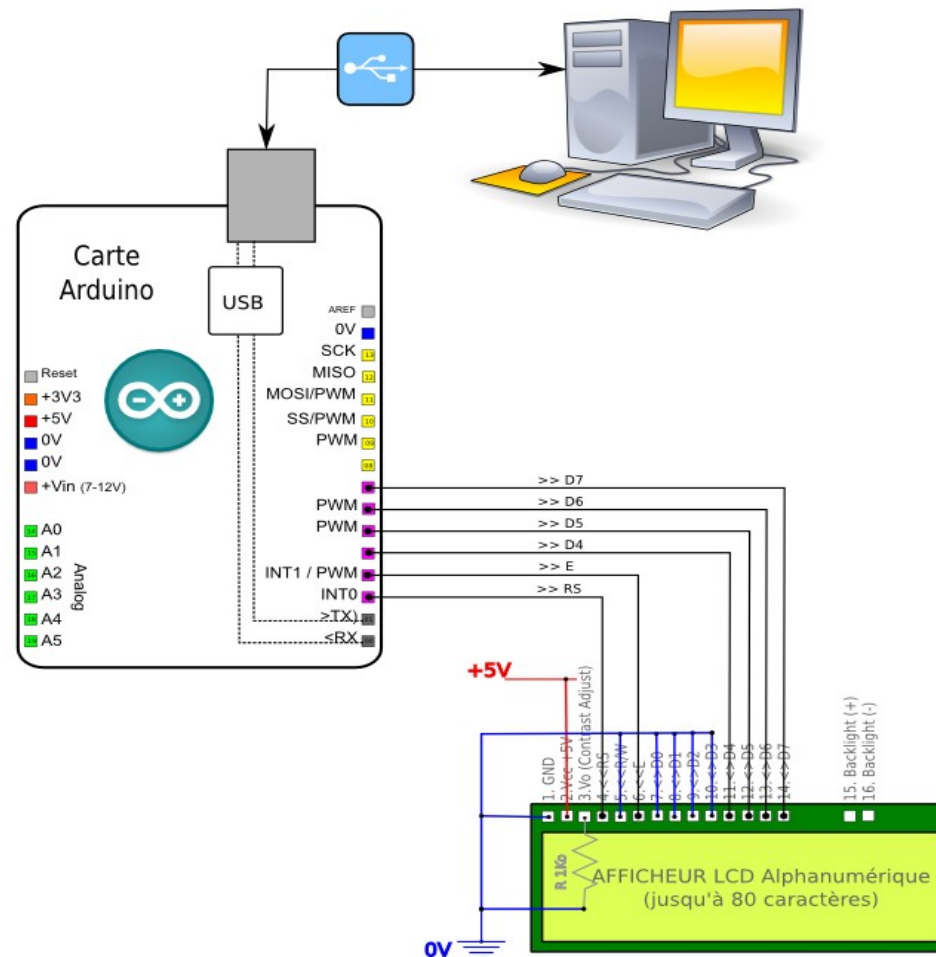
} // fin de la fonction loop()

// NB : les lignes précédées de // sont des commentaires
```

12. Afficheur LCD standard : afficher une chaîne reçue sur le port Série : le montage

De la même façon que précédemment, le montage consiste à connecter :

- les 2 broches de commande RS et E sur 2 broches numériques Arduino,
- les 4 broches de données D4 à D7 sur 4 broches numériques Arduino,
- le +5V et le 0V
- et la carte Arduino restera connectée au port série pour communiquer avec le poste fixe.



13. Afficheur LCD standard : afficher une chaîne reçue sur le port Série : le programme

Ce que nous allons faire ici :

Nous allons à présent contrôler l'afficheur LCD à partir du port Série : en fait, nous allons ici envoyer depuis le Terminal série une chaîne qui sera automatiquement affichée sur l'afficheur LCD sur la première ligne de l'afficheur. Normalement, pas de souci pour vous : vous savez faire avec ce que vous connaissez !

Entête déclarative

Déclarations pour l'afficheur LCD

- On commence par importer la librairie **LiquidCrystal**
- Ensuite on déclare l'ensemble des 6 broches utilisées pour le contrôle de l'afficheur LCD
- On déclare un objet **LiquidCrystal** qui représentera le LCD dans le reste du programme.

Variables pour réception sur port Série

- On déclare les variables utiles pour la réception de la chaîne sur le port Série

Tableau de chaîne pour affichage sur le LCD

- On déclare également un tableau de **String** pour gérer l'affichage des chaînes reçues sur l'afficheur.

```
//affiche ligne à ligne sur afficheur LCD standard chaîne reçue sur port
Série

//--- Librairies incluses ---
#include <LiquidCrystal.h> // inclusion de la librairie LCD

//-- déclaration des broches de l'afficheurs --
const int RS=2; // broche RS
const int E=3; // broche E
const int D4=4; // broche D4
const int D5=5; // broche D5
const int D6=6; // broche D6
const int D7=7; // broche D7

LiquidCrystal lcd(RS,E,D4,D5,D6,D7); // déclaration objet représentant
lcd

//----- variables pour la réception sur le port série
int octetReception=0; // variable de réception octet
char caractereReception=0; // variable de réception caractère
String chaineReception=""; // déclare un objet String vide

//----- variables pour affichages sur les lignes de l'afficheur -----
String lignesLCD[4] = { "", "", "", "" }; // tableau de String - 1 par ligne
```

Fonction **setup()**

Initialisation de la communication Série

- On initialise la communication série à 115200 bauds.
- On affiche un message d'accueil dans le Terminal Série.

Initialisation de l'afficheur LCD

- On commence par initialiser l'afficheur à l'aide de la fonction **begin**(colonnes,lignes). Ici, afficheur 20 colonnes x 4 lignes. A adapter à votre situation le cas échéant.
- On initialise l'affichage avec la fonction **clear()** suivie d'une rapide pause.
- Puis on affiche un message de test pendant 1 seconde avec la fonction **print**(« Texte») suivi de la fonction **delay()**.
- Et à nouveau la fonction **clear()** suivie d'une rapide pause.

```
//--- la fonction setup() : exécutée au début et 1 seule fois
void setup() {

    //----- Initialisation port Série ----
    Serial.begin(115200); // initialise communication série
    //-- utiliser la meme vitesse dans le Terminal Série
    Serial.println("Arduino ok !");
    Serial.println("Saisir chaine + clic Send.");

    //---- Initialisation afficheur LCD -----
    lcd.begin(20,4); // initialise LCD colonnes x lignes

    lcd.clear(); // efface LCD + se place en 0,0
    delay(10); // courte pause après clear()

    lcd.print("LCD OK !"); // affiche le message
    delay(1000); // pause

    lcd.clear(); // efface LCD + se place en 0,0
    delay(10); // courte pause après clear()

} // fin de la fonction setup()
```

Fonction **loop()**

Réception de la chaîne sur le port Série

- Comme cela a été vu dans les ateliers consacrés au port Série, à l'aide d'une boucle `while()`, on teste l'arrivée d'un caractère sur le port Série, caractère qui est ajouté à la chaîne de réception.
- Ceci se répète tant qu'un caractère est présent.

Gestion de la chaîne reçue

- Une fois la chaîne reçue,
 - on réalise un décalage des lignes au sein du tableau de **String** utilisé pour la gestion des lignes de l'afficheur,
 - puis on mémorise la chaîne dans le **String** de la première ligne
 - les 4 **String** sont ensuite affichés après avoir effacé l'afficheur.
- L'ensemble abouti à un décalage vers le bas des chaînes déjà reçues à la réception d'une nouvelle chaîne.

Le langage Arduino permet ainsi de combiner facilement plusieurs fonctionnalités : ici le port Série, la manipulation des chaînes de caractères **String** et l'afficheur LCD.

C'est relativement simple et efficace !
(Ce programme ne fait que 6Ko...)

```
//--- la fonction loop() : exécutée ensuite en boucle sans fin
void loop() {

    while (Serial.available()>0) { // si un caractère en réception

        octetReception=Serial.read(); // lit le 1er octet de la file
        d'attente

        caractereReception=char(octetReception); // récupère le caractère à
        partir du code Ascii

        chaineReception=chaineReception+caractereReception; // ajoute la
        caractère au String

        delay(1); // laisse le temps au caractères d'arriver

    } // fin while - fin de réception de la chaîne

    //----- une fois la chaîne reçue
    if (chaineReception!="") { // la chaîne n'est pas vide
        Serial.print("Chaîne reçue : ");
        Serial.println(chaineReception);

        //--- ajoute la chaîne dans le tableau de String pour déclencher vers
        le bas ---
        lignesLCD[3]=lignesLCD[2]; // copie ligne n-1 dans ligne n
        lignesLCD[2]=lignesLCD[1]; // copie ligne n-1 dans ligne n
        lignesLCD[1]=lignesLCD[0]; // copie ligne n-1 dans ligne n
        lignesLCD[0]=chaineReception.substring(0,20); // met les 20 premiers
        caractères de la chaîne reçue sur la première ligne

        //----- affiche les String de lignes
        lcd.clear(); // efface LCD
        delay(10); // pause après clear

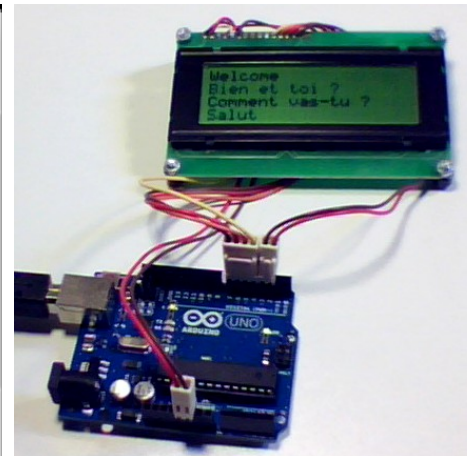
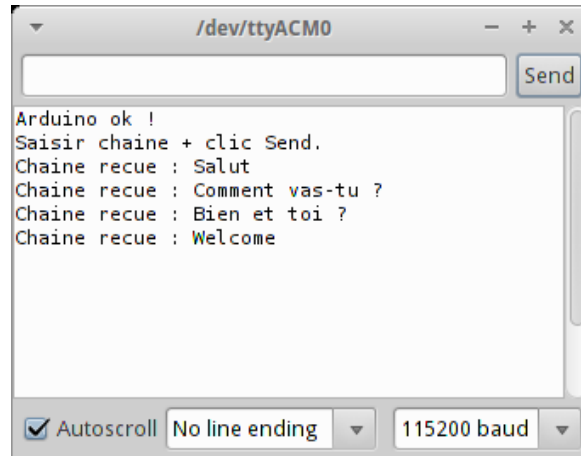
        for (int i=0; i<4; i++) {
            lcd.setCursor(0, i); // se positionne en début de ligne
            lcd.print(lignesLCD[i]); // affiche la chaîne de ligne i
        } // fin for

        //--- RAZ chaîne réception
        chaineReception=""; // Vide la chaîne
    } // fin if chaîneReception pas vide

} // fin de la fonction loop()
```

Fonctionnement du programme

- Une fois la carte Arduino programmée :
 - saisir une chaîne et clic sur <Send> :
 - la chaîne saisie (les 20 premiers caractères en fait) sera (seront) affiché(s) sur la première ligne de l'afficheur.
- Chaque nouvelle chaîne reçue entraînera un décalage vers le bas des lignes précédentes.



Sympa non ?

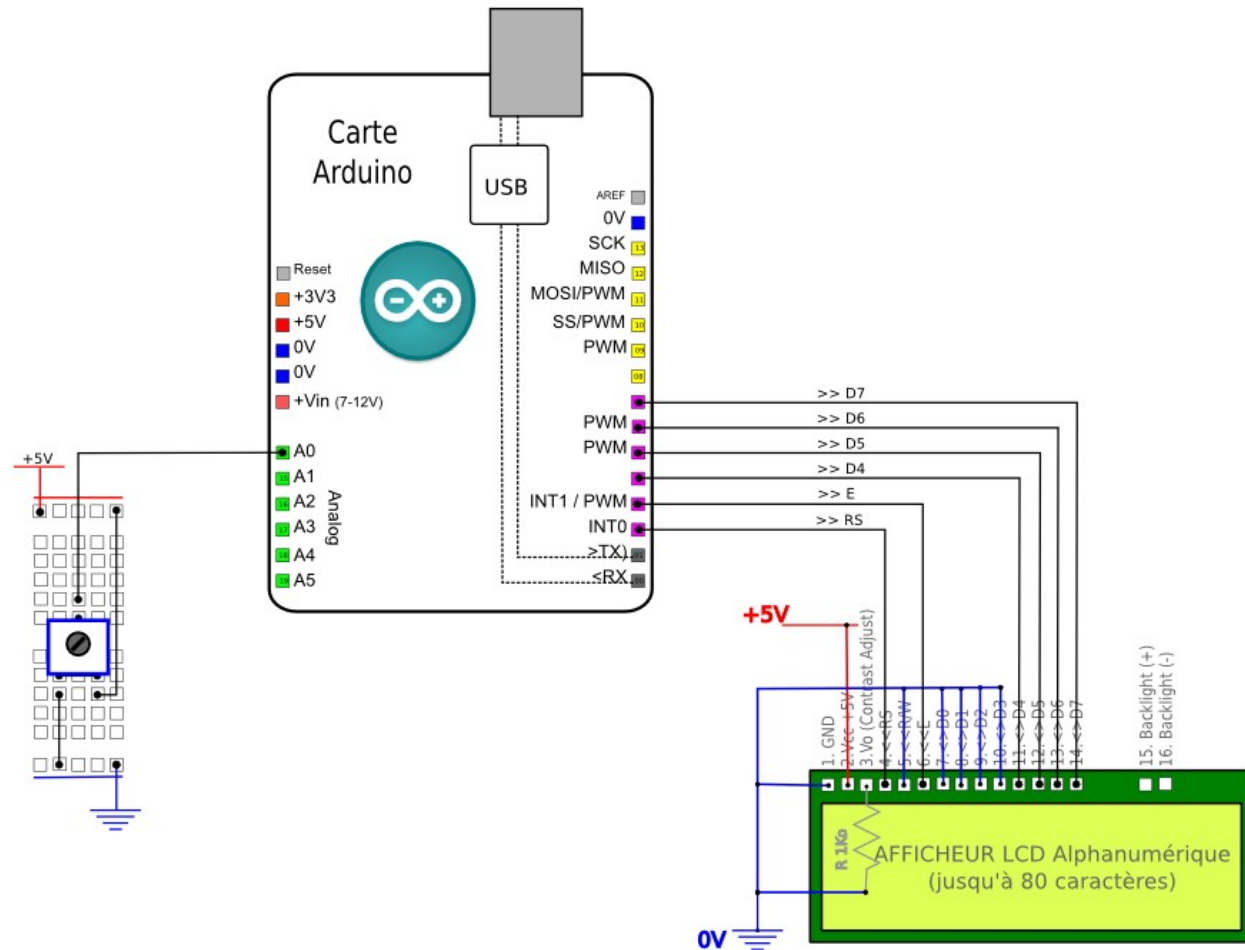
Pour info, la même chose est réalisable de façon encore plus directe à partir d'une interface Processing où chaque caractère saisi au clavier du PC sera immédiatement retranscrit sur l'afficheur LCD, comme nous le verrons dans un autre atelier dédié aux interfaces Processing.

Just for fun ! Mais aussi très pratique pour régler des valeurs de paramètres depuis le PC...

14. Afficheur LCD standard et conversion analogique numérique : un simple voltmètre 5V : le montage

De la même façon que précédemment, le montage consiste à connecter :

- l'afficheur LCD « préparé » sur la carte Arduino :
 - les 2 broches de commande RS et E sur 2 broches numériques Arduino,
 - les 4 broches de données D4 à D7 sur 4 broches numériques Arduino,
 - le +5V et le 0V
- de plus, on connecte une résistance variable sur une broche analogique de la carte Arduino.



15. Afficheur LCD standard et conversion analogique numérique : un simple voltmètre 5V : le programme

Ce que nous allons faire ici :

Ici, je vous propose de créer un véritable voltmètre 5V avec une précision de 5mV et affichage sur LCD : c'est assez simple comme vous allez le voir. Une résistance variable est connectée sur une broche analogique de la carte Arduino, une mesure de la tension sur cette broche est réalisée et le résultat est affiché sur l'afficheur LCD. Dans une visée didactique, on affichera ici la valeur brute de la mesure et la valeur de la tension calculée, sur 2 lignes de l'afficheur. Place au code !

Entête déclarative

Déclarations pour l'afficheur LCD

- On commence par importer la librairie **LiquidCrystal**
- Ensuite on déclare l'ensemble des 6 broches utilisées pour le contrôle de l'afficheur LCD
- On déclare un objet **LiquidCrystal** qui représentera le LCD dans le reste du programme.

Déclarations pour la mesure analogique

- On déclare une variable **int** pour stocker la mesure brute et une variable **float** pour stocker le résultat du calcul de la tension.

```
// voltmètre 5V sur afficheur LCD standard

//--- librairie incluse ---
#include <LiquidCrystal.h> // inclusion de la librairie LCD

//-- déclaration des broches de l'afficheurs --
const int RS=2; // broche RS
const int E=3; // broche E
const int D4=4; // broche D4
const int D5=5; // broche D5
const int D6=6; // broche D6
const int D7=7; // broche D7

const int RVar=0; //declaration constante de broche analogique

// --- Déclaration des variables globales ---
int mesureBrute=0; // Variable pour acquisition résultat brut de
conversion analogique numérique
float mesure; // variable à virgule pour calcul valeur en millivolts

//---- Objets utiles ---
LiquidCrystal lcd(RS,E,D4,D5,D6,D7); // déclaration objet représentant
lcd
```

Fonction **setup()**

Initialisation de l'afficheur LCD

- On commence par initialiser l'afficheur à l'aide de la fonction **begin**(colonnes,lignes). Ici, afficheur 20 colonnes x 4 lignes. A adapter à votre situation le cas échéant.
- On initialise l'affichage avec la fonction **clear()** suivie d'une rapide pause.
- Puis on affiche un message de test pendant 1 seconde avec la fonction **print**(« Texte») suivi de la fonction **delay()**.
- Et à nouveau la fonction **clear()** suivie d'une rapide pause.

```
//--- la fonction setup() : exécutée au début et 1 seule fois
void setup() {

    lcd.begin(20,4); // initialise LCD colonnes x lignes

    lcd.clear(); // efface LCD + se place en 0,0
    delay(10); // courte pause après clear()

    lcd.print("LCD OK !"); // affiche le message
    delay(1000); // pause

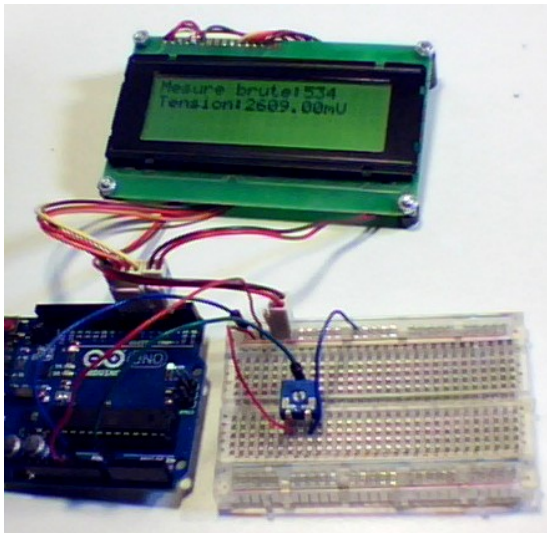
    lcd.clear(); // efface LCD + se place en 0,0
    delay(10); // courte pause après clear()

} // fin de la fonction setup()
```

Fonction `loop()`

- on réalise une mesure à l'aide de la fonction `analogRead(brocheAnalogique)` : le résultat est stocké dans une variable.
- on convertit la valeur brute en la valeur de la tension correspondante à l'aide de l'instruction `map()` qui réalise une règle de 3 de l'échelle 0-1023 vers l'échelle 0-5000mV.
- ensuite, on affiche le résultat sur l'afficheur LCD à l'aide :
 - des fonctions de positionnement `home()` et `setCursor()`
 - et de la fonction `lcd.print()`
- On réalise ensuite une pause d'une demi-seconde entre 2 mesures avec l'instruction `delay(100)`.

Remarquer la souplesse avec laquelle il est possible de créer facilement des applications avec des messages combinant des chaînes de texte et des variables entières ou décimales sur un afficheur LCD avec le langage Arduino ! Encore une fois, simple et efficace !



```
//--- la fonction loop() : exécutée ensuite en boucle sans fin
void loop() {

    // acquisition conversion analogique-numerique (CAN) sur la voie
    analogique
    mesureBrute=analogRead(RVar);

    // calcul de la valeur a virgule en millivolts
    // = convertit la valeur brute 0 - 1023 en valeur 0 - 5000 mV

    mesure=map(mesureBrute,0,1023,0.0,5000.0);

    //---- calcul équivalent ----
    //mesure=mesureBrute;
    //mesure=mesure*5000.0;
    // mesure=mesure/1023.0;

    // affiche valeur numerique entiere puis à virgule au format decimal

    //--- 1ère ligne ---
    lcd.home();
    lcd.print("Mesure brute:");
    lcd.print(mesureBrute);
    lcd.print(" "); // espace de propreté

    lcd.setCursor(0,1); // début 2ème ligne
    lcd.print("Tension:");
    lcd.print(mesure,2); // affichage avec 2 virgules
    lcd.print("mV ");

    delay(100); // entre 2 mesures

} // fin de la fonction loop()
```

Fonctionnement du programme

- Une fois la carte Arduino programmée, la mesure brute et la tension s'affichent en direct sur l'écran : modifier la résistance variable, la mesure se modifie en conséquence.

Bravo, vous venez de créer vous-même votre premier voltmètre ! Sympa non ?

Avec un code que de 5 Ko en plus !

16. Rappel : Fiche composant : le capteur analogique linéaire de température LM35

Description

Le capteur de température LM35 est un capteur de température linéaire fonctionnant en 5V.



Fiche technique : <http://www.mon-club-elec.fr/datasheet/LM35.pdf>

Brochage

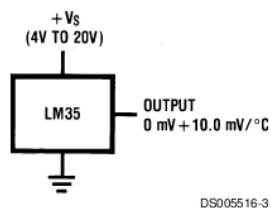
3 broches : le 0V (GND), le +5V (+Vs) et la tension de sortie (Vout)



TL/H/5516-2

Order Number LM35CZ,
LM35CAZ or LM35DZ

Schéma fonctionnel



DS005516-3

FIGURE 1. Basic Centigrade Temperature Sensor
(+2°C to +150°C)

Caractéristiques

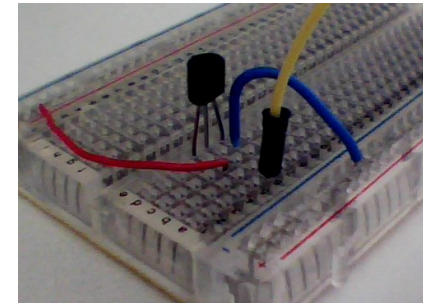
- mesurant la température entre -55 et +150°C
- dont la sortie est à 0V à 0°C
- avec une variation de 10mV par °C
- alimentable en 5V

Grâce à ces informations, on pourra utiliser précisément le capteur pour réaliser un thermomètre avec Arduino !

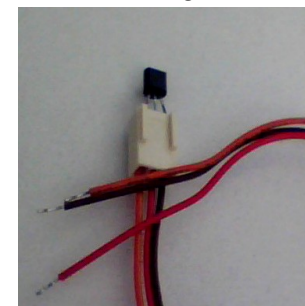
Exemple d'utilisation sur plaque d'essai avec Arduino

On connecte :

- la broche Vs au +5V
- la broche GND au 0V
- la broche Vout à une broche analogique de la carte Arduino.



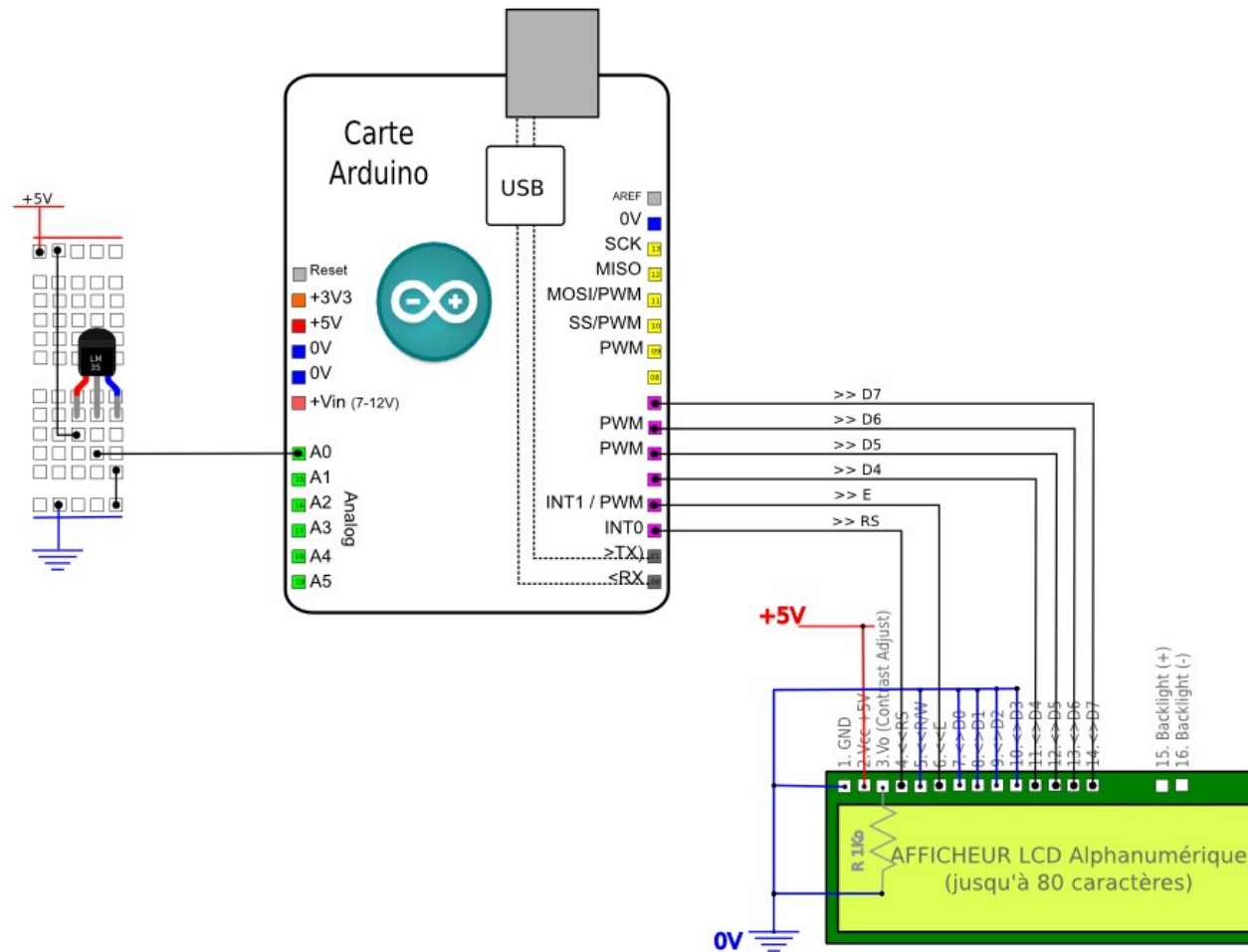
Truc : on peut utiliser un connecteur 3 broches sur fils pour le montage !



17. Afficheur LCD standard et conversion analogique numérique : un simple thermomètre : le montage

De la même façon que précédemment, le montage consiste à connecter :

- l'afficheur LCD « préparé » sur la carte Arduino :
 - les 2 broches de commande RS et E sur 2 broches numériques Arduino,
 - les 4 broches de données D4 à D7 sur 4 broches numériques Arduino,
 - le +5V et le 0V
- de plus, *on connecte un capteur de température LM35 sur une broche analogique de la carte Arduino.*



18. Afficheur LCD standard et conversion analogique numérique : un simple thermomètre : le programme

Ce que nous allons faire ici :

Dans la même veine que ce que nous venons de faire, je vous propose ici de réaliser un simple thermomètre numérique à affichage digital, ni plus ni moins ! On utilise ici le capteur de température analogique linéaire LM35 qui fournit une tension en sortie valant $0\text{ V} + 10\text{mV}/^{\circ}\text{C}$ (voir l'atelier consacré aux mesures analogiques avec Arduino pour plus de détails). Il devient dès lors très facile d'obtenir la température avec une précision d'un demi-degré Celsius. C'est tout bête, mais ce thermomètre vous l'aurez fait vous-même ! Et ce montage pourra servir de base à toutes sortes d'applications concrètes : station météo, surveillance d'aquarium, de serre, de cave à vin, etc... Allez, on se lance !

Entête déclarative

Déclarations pour l'afficheur LCD

- On commence par importer la librairie **LiquidCrystal**
- Ensuite on déclare l'ensemble des 6 broches utilisées pour le contrôle de l'afficheur LCD
- On déclare un objet **LiquidCrystal** qui représentera le LCD dans le reste du programme.

Déclarations pour la mesure analogique

- On déclare une variable **int** pour stocker la mesure brute et une variable **float** pour stocker le résultat du calcul de la tension.
- On déclare également une variable **float** pour le calcul de la température.

```
// thermomètre à LM35 avec affichage sur LCD standard

//--- inclusion librairies ---
#include <LiquidCrystal.h> // inclusion de la librairie LCD

//-- déclaration des broches de l'afficheurs --
const int RS=2; // broche RS
const int E=3; // broche E
const int D4=4; // broche D4
const int D5=5; // broche D5
const int D6=6; // broche D6
const int D7=7; // broche D7

const int RVar=0; //déclaration constante de broche analogique

// --- Déclaration des variables globales ---
int mesureBrute=0; // Variable pour acquisition résultat brut de
conversion analogique numérique
float mesure; // variable à virgule pour calcul valeur en millivolts
float temperature; // variable à virgule pour stockage temperature

//---- Objets utiles ---
LiquidCrystal lcd(RS,E,D4,D5,D6,D7); // déclaration objet représentant
lcd
```

Fonction **setup()**

Initialisation de l'afficheur LCD

- On commence par initialiser l'afficheur à l'aide de la fonction **begin**(colonnes,lignes). Ici, afficheur 20 colonnes x 4 lignes. A adapter à votre situation le cas échéant.
- On initialise l'affichage avec la fonction **clear()** suivie d'une rapide pause.
- Puis on affiche un message de test pendant 1 seconde avec la fonction **print**(« Texte») suivi de la fonction **delay()**.
- Et à nouveau la fonction **clear()** suivie d'une rapide pause.

```
//--- la fonction setup() : exécutée au début et 1 seule fois
void setup() {

    lcd.begin(20,4); // initialise LCD colonnes x lignes

    lcd.clear(); // efface LCD + se place en 0,0
    delay(10); // courte pause après clear()

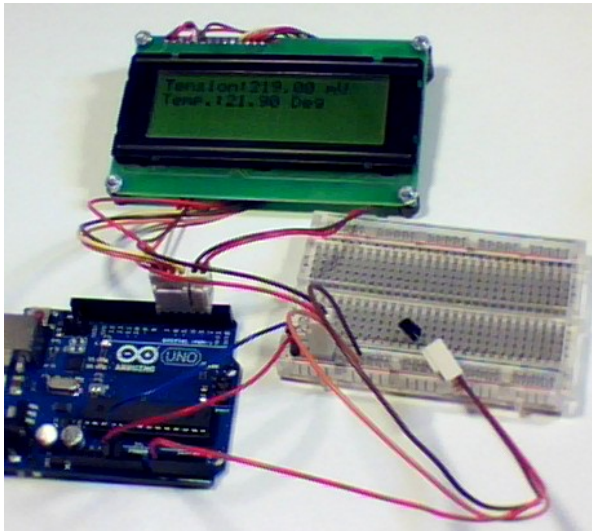
    lcd.print("LCD OK !"); // affiche le message
    delay(1000); // pause

    lcd.clear(); // efface LCD + se place en 0,0
    delay(10); // courte pause après clear()

} // fin de la fonction setup()
```

Fonction `loop()`

- on réalise une mesure à l'aide de la fonction `analogRead(brocheAnalogique)` : le résultat est stocké dans une variable.
- on convertit la valeur brute en la valeur de la tension correspondante à l'aide de l'instruction `map()` qui réalise une règle de 3 de l'échelle 0-1023 vers l'échelle 0-5000mV.
- ensuite, on affiche le résultat sur l'afficheur LCD à l'aide :
 - des fonctions de positionnement `home()` et `setCursor()`
 - et de la fonction `lcd.print()`
- On réalise ensuite une pause d'une demi-seconde entre 2 mesures avec l'instruction `delay(100)`.



```
//--- la fonction loop() : exécutée ensuite en boucle sans fin
void loop() {

    // acquisition conversion analogique-numerique (CAN) sur la voie
    // analogique
    mesureBrute=analogRead(RVar);

    // calcule de la valeur a virgule en millivolts
    // = convertit la valeur brute 0 - 1023 en valeur 0 - 5000 mV

    mesure=map(mesureBrute,0,1023,0.0,5000.0);

    //---- calcul équivalent ----
    //mesure=mesureBrute;
    //mesure=mesure*5000.0;
    // mesure=mesure/1023.0;

    // affiche valeur numerique entière puis à virgule au format décimal

    //--- 1ère ligne ---
    lcd.home();
    lcd.print("Tension:");
    lcd.print(mesure,2); // affichage avec 2 virgules
    lcd.print(" mV ");

    temperature=mesure/10.0; // calcul temperature 0 + 10mV/°C

    lcd.setCursor(0,1); // début 2ème ligne
    lcd.print("Temp.:");
    lcd.print(temperature,2); //affichage avec 2 virgules
    lcd.print(" Deg ");

    delay(100); // entre 2 mesures

} // fin de la fonction loop()
```

Fonctionnement du programme

- Une fois la carte Arduino programmée, la tension mesurée et la température s'affichent en direct sur l'écran : prenez le thermomètre dans vos doigts, la mesure se modifie en conséquence.

Bravo, vous venez de créer vous-même votre premier thermomètre à affichage digital ! Sympa non ?

Et toujours avec un code que de seulement 5 Ko !

Nous verrons ultérieurement comment stocker les valeurs obtenues dans une carte SD et les visualiser ensuite sur ordinateur... ce qui donne des perspectives intéressantes pour des surveillances de température, etc... mais ce sera l'objet d'un autre atelier.

19. Les éléments du langage Arduino étudiés dans cet atelier

Les fonctions usuelles de la librairie **LiquidCrystal**, à savoir :

- Fonctions d'initialisation : [begin\(\)](#)
- Fonctions d'écriture : [print\(\)](#) | [write\(\)](#)
- Fonctions de gestion de l'écran : [clear\(\)](#) | [display\(\)](#) | [noDisplay\(\)](#) |
- Fonctions de positionnement du curseur : [home\(\)](#) | [clear\(\)](#) | [setCursor\(\)](#)
- Fonctions modifiant l'aspect du curseur : [cursor\(\)](#) | [noCursor\(\)](#) | [blink\(\)](#) | [noBlink\(\)](#)

La documentation complète du langage Arduino en français est disponible ici :
http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.ReferenceMaxi

20. *A présent, vous devriez être capable :*

- d'écrire un programme capable de recevoir des chaînes de caractères en provenance du Terminal Série avec Arduino pour l'afficher sur LCD
- d'afficher des valeurs numériques et des résultats de calcul sur afficheur LCD alpha-numérique
- de réaliser des mesures et de les afficher sur un afficheur LCD alpha-numérique

Table des matières

Intro |
Matériel nécessaire pour les ateliers Arduino |
Matériel spécifique nécessaire pour cet atelier |
Rappel : Fiche Technique : Afficheur LCD alpha-numérique standard |
Rappel : Préparation d'un afficheur LCD pour une utilisation simplifiée avec Arduino : le schéma théorique |
Rappel : Préparation d'un afficheur LCD pour une utilisation simplifiée avec Arduino : description concrète |
Rappel : Schéma électrique type d'utilisation d'un afficheur LCD avec une carte Arduino |
Utilisation d'un afficheur LCD « préparé » avec une carte Arduino : le montage |
Utilisation d'un afficheur LCD préparé avec une carte Arduino : en images |
Rappel : Langage Arduino : la librairie LiquidCrystal pour le contrôle des afficheurs LCD standards |
Rappel : « Hello world » : afficher un premier message sur un afficheur LCD. |
Afficheur LCD standard : afficher une chaîne reçue sur le port Série : le montage |
Afficheur LCD standard : afficher une chaîne reçue sur le port Série : le programme |
Afficheur LCD standard et conversion analogique numérique : un simple voltmètre 5V : le montage |
Afficheur LCD standard et conversion analogique numérique : un simple voltmètre 5V : le programme |
Rappel : Fiche composant : le capteur analogique linéaire de température LM35 |
Afficheur LCD standard et conversion analogique numérique : un simple thermomètre : le montage |
Afficheur LCD standard et conversion analogique numérique : un simple thermomètre : le programme |
Les éléments du langage Arduino étudiés dans cet atelier |
A présent, vous devriez être capable : |

Bravo !
vous avez terminé cet atelier Arduino !



Prêt pour la suite ? Retrouvez de nombreux autres thèmes d'ateliers Arduino ici :

http://www.mon-club-elec.fr/pmwiki_mon_club_elec/pmwiki.php?n=MAIN.ATELIERS