Montage

13

Un afficheur alphanumérique

Au sommaire :

- la définition d'un afficheur LCD ;
- le fonctionnement d'un tel afficheur ;
- le sketch complet ;
- l'analyse du schéma ;
- la réalisation du circuit ;
- un jeu ;
- un exercice complémentaire.

Qu'est-ce qu'un afficheur LCD ?

Que serait un microcontrôleur sans son afficheur pour correspondre avec le monde extérieur sans passer par l'ordinateur ou le Serial Monitor ? Bien sûr, on a déjà vu comment, par exemple, utiliser des afficheurs sept segments pour représenter des chiffres. S'il s'agit de représenter plusieurs chiffres ou des lettres ou encore des caractères spéciaux tels que par exemple *, #, %... les limites du possible sont vite atteintes. On utilise dans ces cas-là un afficheur à cristaux liquides ou LCD *(Liquid Cristal Display)* sous sa forme abrégée. Ces afficheurs contiennent des cristaux liquides capables de modifier leur orientation en fonction d'une tension appliquée, et de jouer ainsi plus ou moins sur l'incidence de la lumière.

Pour aller plus loin

Pour compléter ce chapitre, vous pouvez effectuer une recherche sur Internet sur les mots-clés :

- LCD ;
- LCD Module AVR ;
- Dot matrix display.

De tels éléments d'affichage utilisent en général des motifs composés de points *(Dot-Matrix)* pour représenter à peu près tous les signes (chiffres, lettres ou caractères spéciaux). Leur taille et leur équipement varient. La figure 13-1 en montre trois différents.

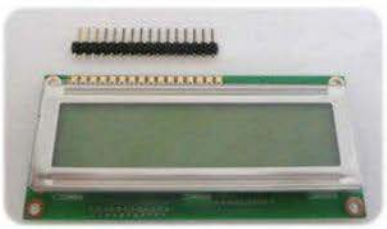
Figure 13-1▲
Plusieurs types d'afficheurs LCD

Le premier afficheur, LCD4884, avec une résolution de 84×48 pixels est déjà monté sur un shield ; il dispose d'un joystick miniature pour naviguer dans les menus, et peut être commandé directement avec la bibliothèque appropriée. Il peut même afficher des graphiques miniatures et se veut donc très souple pour représenter des éléments d'affichage. Le deuxième afficheur, *DMC-2047*, est équipé de 4 LED et d'une diode réceptrice à infrarouge (*IR*). Le troisième, de type HMC16223SG, est un afficheur à deux lignes avec un contrôleur compatible à celui de l'Hitachi *HDD44780*, sur lequel nous reviendrons bientôt. Pour une utilisation plus facile, beaucoup d'afficheurs sont dotés d'un contrôleur intégré qui commande les différents points ou segments. Si nous avions à le faire, le sketch serait beaucoup plus long. Dans l'environnement Arduino, un afficheur LCD avec pilote *HD44780* est relativement souvent utilisé. Ce pilote, qui s'est imposé comme le quasi-standard, est souvent adapté par beaucoup d'autres fabricants. La figure 13-2 montre un afficheur de ce type.

◀ **Figure 13-2**
Afficheur LCD

Il existe pour cet élément une bibliothèque livrée avec l'IDE Arduino. Vous pouvez bien entendu raccorder n'importe quel afficheur ou presque, à condition de trouver une bibliothèque appropriée ou de la créer vous-même. Nous utilisons pour notre montage l'afficheur ci-dessus, à 2 lignes de 16 caractères chacune.

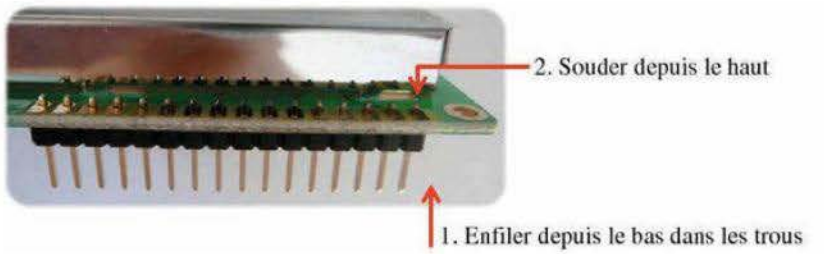
Composants nécessaires

	1 LCD HD44780 + barrette de 16 broches au pas de 2,54 mm
	1 trimmer de 10 kΩ ou 20 kΩ
	Plusieurs cavaliers flexibles de couleurs et de longueurs diverses

Remarque préliminaire sur l'utilisation de l'afficheur LCD

Si vous achetez un afficheur LCD tout neuf, il se peut que seuls des trous de connexion soient présents sur le circuit imprimé, comme vous pouvez le voir sur l'image ci-dessus. Vous pouvez alors, soit équiper de fils les contacts nécessaires et vous en servir plus tard pour le circuit sur la plaque d'essais, soit – et c'est le mieux – vous procurer une barrette à broches, comme vous pouvez le voir également sur l'image suivante.

Des barrettes sont par exemple proposées avec une rangée de 40 broches et un pas de 2,54 mm. Coupez-les à une longueur de 15 broches en les pliant délicatement à l'endroit souhaité. Allez-y doucement car elles ont tendance à casser là où on ne veut pas. Enfilez ensuite les broches de la barrette depuis le bas dans les trous et soudez-les sur la face supérieure.

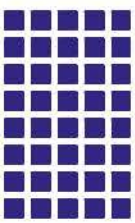


Vous pouvez ainsi brancher sans problème le module sur votre plaque d'essais.

Principes intéressants

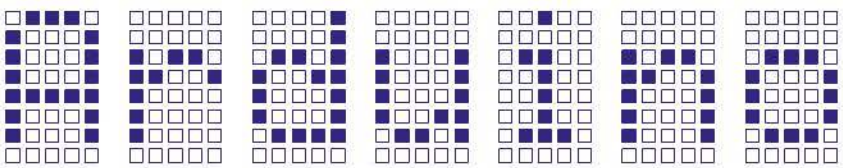
Avant d'utiliser l'afficheur LCD, voici quelques principes importants et intéressants à connaître. Comment un afficheur de ce type fonctionne-t-il ? Nous avons déjà vu que les différents caractères étaient composés à partir d'une matrice de points *(Dot-Matrix)*. *Dot* signifie *point* et se trouve être le plus petit élément représentable dans cette matrice. Tout caractère est construit avec une matrice de points 5 × 8.

Figure 13-3 ▶
La matrice de points 5 × 8 de l'afficheur LCD



Un emploi judicieux des différents points permet de générer les caractères les plus divers. La figure 13-4 montre le mot Arduino et les différents points à partir desquels les lettres sont composées.

Figure 13-4 ▶
Le mot Arduino composé à partir des différents points



La commande de l'afficheur est *parallèle*, autrement dit tous les bits de données sont envoyés en même temps au contrôleur. Il existe deux modes différents (4 bits et 8 bits), le mode 4 bits étant le plus utilisé parce qu'un nombre moindre de lignes de données doit être relié à l'afficheur, ce qui fait diminuer le coût.

Eh là, pas si vite ! Je ne suis pas idiot au point d'ignorer que si j'utilise 4 bits au lieu de 8, j'aurai un débit de données moindre et je pourrai transmettre moins d'informations différentes. Comment faire alors ?

C'est juste, Ardus ! Mais ça fonctionne sans diminution du volume d'informations. En mode 4 bits, les 8 bits d'informations à transmettre sont simplement scindés en deux moitiés : l'une contenant les quatre premiers bits, et l'autre contenant les quatre derniers bits. Un nombre binaire de 4 bits est appelé *nibble* (quartet) dans le traitement des données. Les 4 bits d'un nibble sont transmis en parallèle, et les deux nibbles d'un octet en série. Surtout, ne vous en faites pas. Le mode 4 bits est certes plus lent que le mode 8 bits, mais ça n'a ici aucune importance. Venons-en maintenant au module d'affichage LCD Hitachi *HDD44780*, à son brochage et aux branchements nécessaires. Il existe deux variantes différentes : celle à 16 broches qui possède un rétroéclairage, et celle à 14 broches qui n'en a pas besoin.

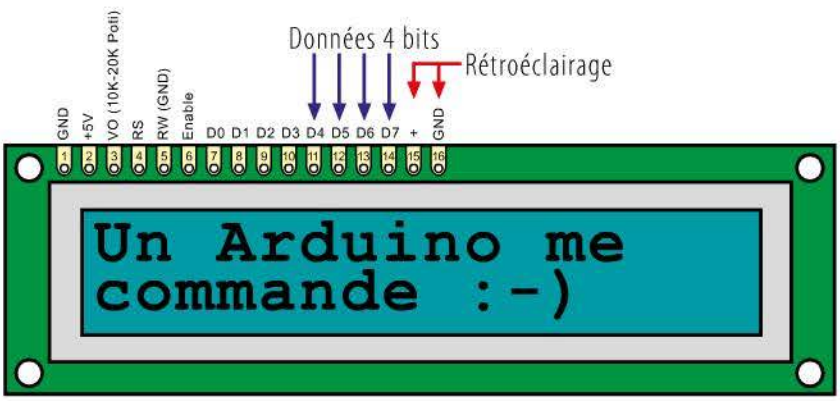


Figure 13-5
Branchements du module d'affichage

Sur les huit lignes de données, seules les quatre lignes supérieures (D4 à D7) sont nécessaires. Le tableau 13-1 donne l'affectation des broches et leur signification.

Tableau 13-1 ▶
Occupation des broches LCD pour la variante à 16 broches

Broche LCD	Broche Arduino	
1	GND	Masse
2	+5 V	+5 V
3	-	Réglage du contraste par un potentiomètre de 10 kΩ ou 20 kΩ
4	12	RS (Register Select)
5	GND	RW (Read/Write)/connecté à la masse (HIGH : Read/LOW : Write)
6	11	E (Enable)
11	5	Ligne de données D4
12	4	Ligne de données D5
13	3	Ligne de données D6
14	2	Ligne de données D7
15	-	Anode (+)/via résistance série 220 Ω !
16	GND	Cathode (–)

L'objectif du premier sketch LCD est de faire apparaître à l'écran la phrase « Un Arduino me commande : -) ».

Code du sketch

Ne vous laissez pas effrayer par la logique de commande relativement complexe. Nous allons nous servir d'une bibliothèque, qui nous permettra d'utiliser un afficheur LCD de manière très simple.

```
#include <LiquidCrystal.h>
#define RS 12      //Register Select
#define E 11       //Enable
#define D4 5       //Ligne de données 4
#define D5 4       //Ligne de données 5
#define D6 3       //Ligne de données 6
#define D7 2       //Ligne de données 7
#define COLS 16    //Nombre de colonnes
#define ROWS 2     //Nombre de lignes
LiquidCrystal lcd(RS, E, D4, D5, D6, D7);//Instanciation de l'objet

void setup(){
  lcd.begin(COLS, ROWS);          //Nombres de colonnes et lignes
  lcd.print("Un Arduino me");     //Affichage du texte
  lcd.setCursor(0, 1);            //Passer à la 2e ligne
  lcd.print("commande :-)");      //Affichage du texte
}

void loop(){/* vide */}
```

Revue de code

La bibliothèque `LiquidCrystal` doit être incorporée afin de pouvoir utiliser la fonctionnalité des commandes de l'afficheur LCD. Du point de vue logiciel, la variable suivante est nécessaire à notre montage.

Variable	Rôle
`lcd`	L'objet LCD

Tableau 13-2
Variable nécessaire et son rôle

Les paramètres suivants doivent être communiqués au constructeur pour générer un objet LCD :

- broche Register Select (`RS`) ;
- broche Enable (`E`) ;
- broches des lignes de données `D4` à `D7`.

```
LiquidCrystal lcd(RS, E, D4, D5, D6, D7);  //Instanciation de l'objet
```

La classe `LiquidCrystal` met une série de méthodes à disposition, car on ne peut envoyer un texte à l'afficheur LCD avec le seul constructeur. Pour que ce soit possible, il nous faut transmettre à l'objet afficheur quelques informations supplémentaires pour poursuivre l'initialisation. Les afficheurs LCD diffèrent quant au nombre de colonnes ou de lignes, et ce sont précisément ces informations qu'il lui faut. On voit bien que le constructeur ne dispose pas de tout pour une initialisation complète. Une méthode est ici nécessaire.

Méthode LCD : `begin`

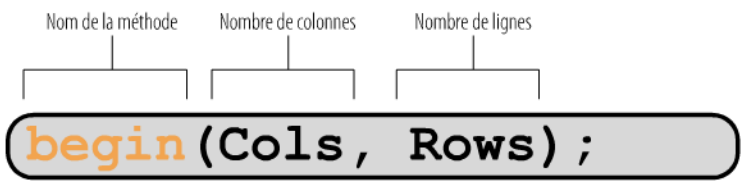


Figure 13-6
Méthode LCD `begin`

La méthode `begin` communique les nombres de colonnes et de lignes à l'objet LCD. Tout est alors prêt pour envoyer un texte.

Méthode LCD : `print`

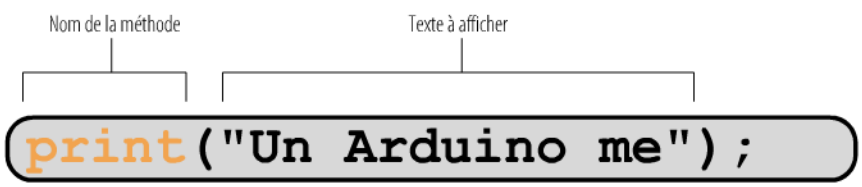


Figure 13-7
Méthode LCD `print`

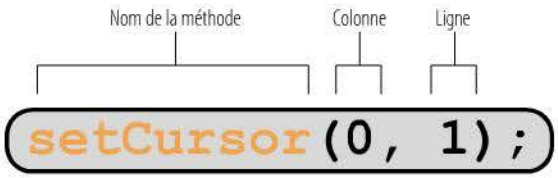
La méthode `print` indique à l'objet LCD ce qui doit être affiché à l'écran. Elle est comparable à celle du Serial Monitor.

Un moment, s'il vous plaît ! L'afficheur que vous utilisez a bien deux lignes. Comment avez-vous fait pour que le texte s'affiche sur la première ?

Quand aucune indication n'est donnée sur la position du texte à afficher, celui-ci se place au début de la première ligne. Comme vous pouvez le voir dans l'exemple, une autre ligne est occupée par du texte. Venons-en maintenant à la troisième méthode importante.

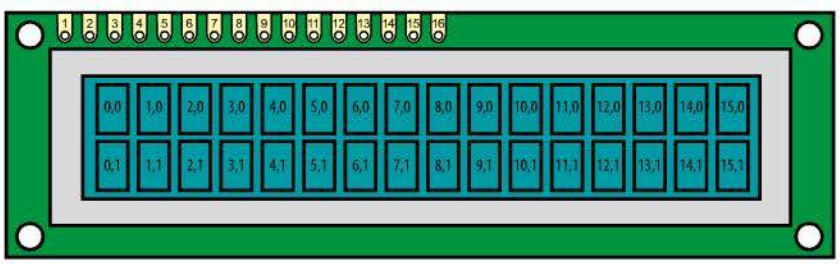
Méthode LCD : `setCursor`

Figure 13-8 ▶
Méthode LCD `setCursor`



La méthode `setCursor` permet de positionner le curseur à l'endroit où le texte suivant doit commencer. Elle est ici aussi – et comment pourrait-il en être autrement – basée sur zéro, autrement dit la première ligne ou colonne est pourvue de l'index `0`. Pour atteindre la deuxième ligne, vous devez – comme c'est le cas ici – utiliser la valeur `1`. La figure 13-9 peut vous aider à positionner l'affichage.

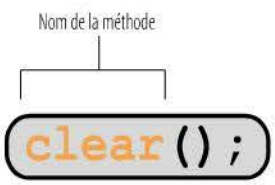
Figure 13-9 ▶
Coordonnées des différentes lignes accessibles avec `setCursor`



Avant d'oublier : vous pouvez naturellement tout effacer jusqu'au dernier caractère avec la méthode `clear`.

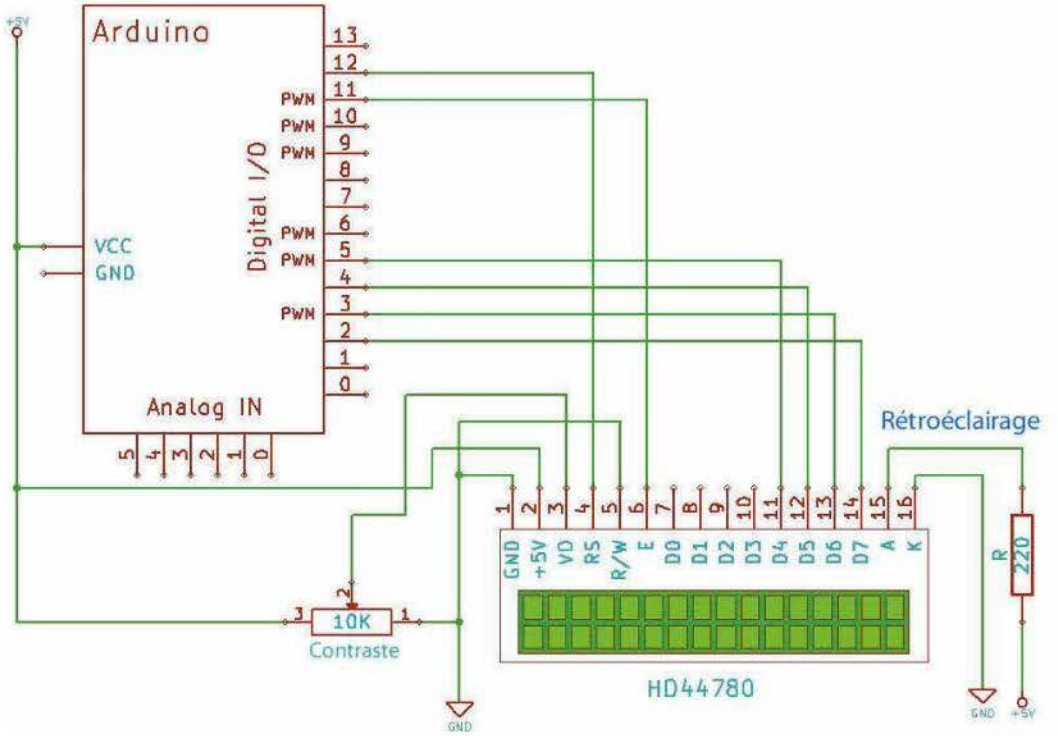
Méthode LCD : `clear`

Figure 13-10 ▶
Méthode LCD `clear`



Elle n'a pas de paramètre, efface tous les caractères de l'afficheur et positionne le curseur sur la coordonnée 0,0 dans le coin supérieur gauche.

Schéma

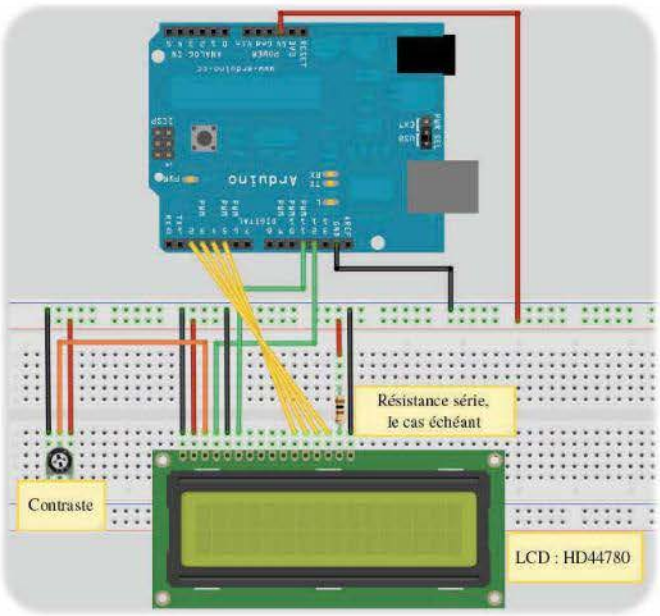


▲ **Figure 13-11**
Connexions de l'afficheur LCD

Attention !
Dans certaines variantes du HD44780, on peut brancher le rétroéclairage sur +5 V sans résistance série ; dans d'autres, une résistance dimensionnée en conséquence est nécessaire. Regardez la fiche technique avant de brancher la tension d'alimentation. Vous pouvez au pire laisser tomber le rétroéclairage. Si c'est trop sombre, vous pourrez toujours augmenter le contraste de manière à pouvoir lire quand même l'affichage.

Réalisation du circuit

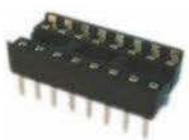
Figure 13-12 ▶
Réalisation de la commande du LCD avec Fritzing



Jeu : deviner un nombre

Quoi de mieux que le jeu où il faut deviner des nombres pour une réalisation avec l'afficheur LCD ? Si ça fonctionne, vous n'aurez plus besoin d'un ordinateur et gagnerez en indépendance grâce à l'unité d'affichage LCD. J'ai fixé, pour le besoin de la réalisation, le LCD sur une carte de circuit imprimé perforée, sur laquelle deux supports de circuit intégré à 16 broches sont posés l'un à côté de l'autre.

Figure 13-13 ▶
Support de circuit intégré à 16 broches



Un support comparable – mais avec plus de broches bien sûr – se trouve sur votre carte Arduino et maintient le microcontrôleur en position. De tels connecteurs sont vraiment utiles car si un circuit intégré vient à griller pour de bon, plus besoin de le dessouder péniblement. Il suffit de le remplacer. La carte mesure 10 × 5 cm.

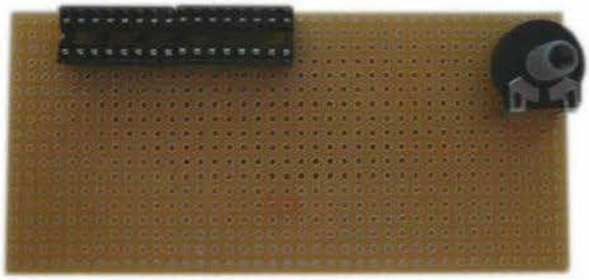


Figure 13-14
Carte-support pour l'afficheur LCD

Comme vous pouvez le constater, j'ai également placé dessus le trimmer pour le réglage du contraste. De l'autre côté de la carte, on voit comment j'ai relié entre elles les différentes broches des supports de circuit intégré. Les broches opposées ont toutes été reliées par plusieurs points de soudure.

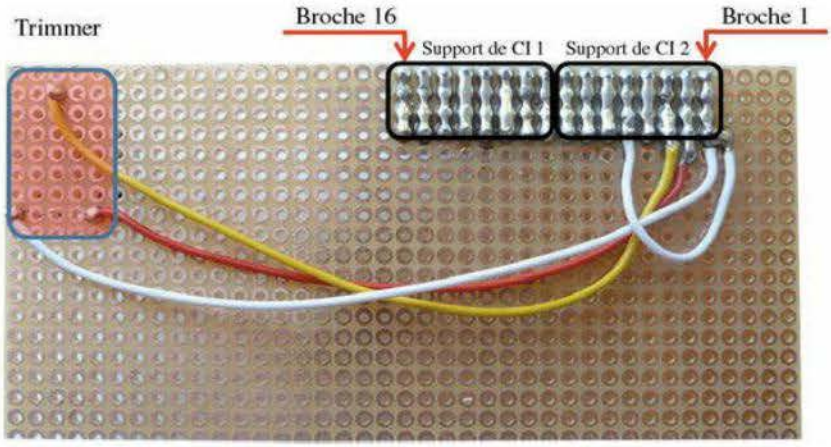


Figure 13-15
Face soudée de la carte-support pour l'afficheur LCD

Un *câblage volant* est parfois suffisant. La figure 13-16 montre l'afficheur LCD déjà fixé sur la carte. Ses broches sont insérées dans les contacts de la rangée inférieure des deux supports de circuit intégré. La rangée supérieure servira plus tard pour la connexion au shield.

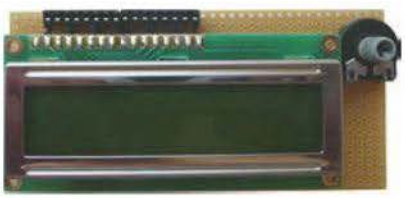


Figure 13-16
Afficheur LCD sur sa carte-support

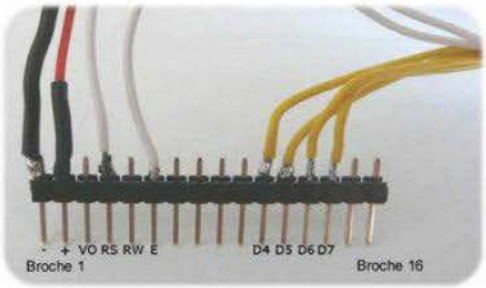
Il ne manque plus maintenant que les lignes de raccordement à votre clavier analogique. Les liaisons passent par les barrettes à broches que vous connaissez bien.

Il vous faut :

- 1 barrette à 16 broches ;
- 2 barrettes à 8 broches ;
- 1 barrette à 6 broches.

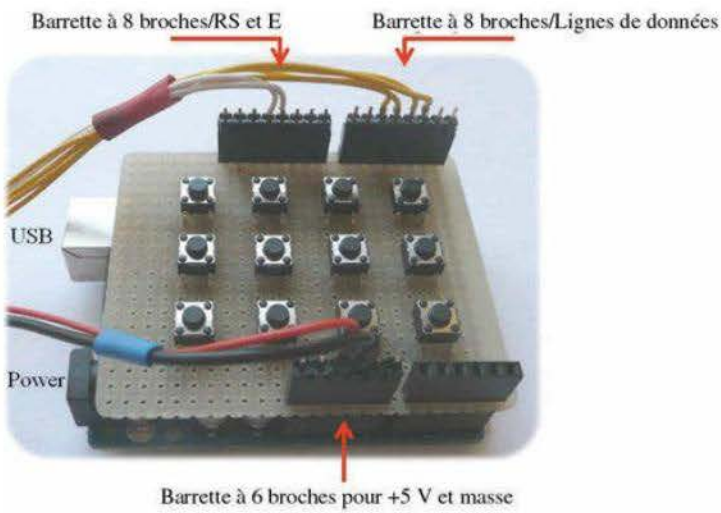
La barrette à 16 broches est raccordée à la carte-support sur laquelle se trouve l'afficheur LCD. La figure 13-17 illustre les lignes de raccordement analogique.

Figure 13-17 ▶
Barrette à 16 broches



Les deux barrettes à 8 broches et la barrette à 6 broches sont branchées sur le clavier.

Figure 13-18 ▶
Clavier analogique avec ses trois barrettes à broches



En utilisant les couleurs et l'affectation des broches indiquée, vous ne devriez pas avoir de problème pour construire le petit faisceau de câbles avec les barrettes à broches. La figure 13-19 montre à nouveau les trois composants, à savoir la carte-support avec l'afficheur LCD, la carte Arduino et le shield du clavier superposé, tous reliés entre eux.

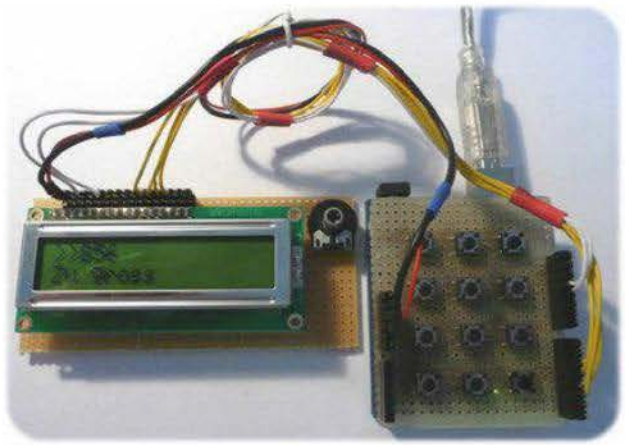


Figure 13-19
Réalisation du circuit complet pour le jeu des nombres à deviner

Attention !

Si vous soudez sur les barrettes à broches des fils d'alimentation ou de masse qui se trouvent directement l'un à côté de l'autre, vous risquez fort d'avoir un jour ou l'autre, par déplacement, un court-circuit entre ces contacts ou les fils avoisinants. Aussi ai-je pris soin de mettre chaque soudure sous gaine thermo-rétractable pour l'isoler.

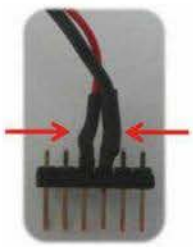


Figure 13-20
Barrette à 6 pôles avec deux morceaux de gaine thermo rétractable (flèches rouges)

Voici maintenant le code complet, qui est déjà un peu plus conséquent.

```
#include <LiquidCrystal.h>
#include "MyAnalogKeyPad.h"
#define analogPinKeyPad 0     //Définition de la broche analogique
#define MIN 10                //Limite inférieure du nombre aléatoire
#define MAX 1000              //Limite supérieure du nombre aléatoire
#define RS 12                 //Broche Register Select du LCD
#define E 11                  //Broche Enable du LCD
#define D4 5                  //Ligne de données LCD broche 4
#define D5 4                  //Ligne de données LCD broche 5
#define D6 3                  //Ligne de données LCD broche 6
#define D7 2                  //Ligne de données LCD broche 7
#define COLS 16               //Nombre de colonnes LCD
#define ROWS 2                //Nombre de lignes LCD
int arduinoNumber, tries;  //Le nombre généré, nombre d'essais
```

```
char yourNumber[5];          //Nombre à 5 chiffres maxi
byte place;
MyAnalogKeyPad myOwnKeyPad(analogPinKeyPad);  //Instanciation clavier
LiquidCrystal lcd(RS, E, D4, D5, D6, D7);    //Instanciation LCD

void setup(){
  myOwnKeyPad.setDebounceTime(500);  //Réglage temps
                                     //de rebond 500 ms
  lcd.begin(COLS, ROWS);          //Nombres de lignes et de colonnes
  lcd.blink();                    //Faire clignoter le curseur
  startSequence();                //Appel de la séquence de démarrage
}

void loop(){
  char myKey = myOwnKeyPad.readKey(); //Lecture de
                                      //la touche appuyée
  if(myKey != KEY_NOT_PRESSED){       //Interrogation si une touche
                                      // quelconque appuyée
    yourNumber[place] = myKey;
    place++;
    lcd.print(myKey);                 //Afficher la touche sur le LCD
}

  if(place == int(log10(MAX))+1){
    tries++;
    int a = atoi(yourNumber);
    if(a == arduinoNumber){
      lcd.clear();                  //Effacer écran LCD
      lcd.print("Exact !!!");       //Affichage sur LCD
      lcd.setCursor(0, 1);          //Positionnement curseur
                                    //sur 2e ligne
      lcd.print("Essai :" + String(tries));
      delay (4000);                 //Attendre 4 secondes
      tries = 0;                    //Remise à zéro du nombre d'essais
      startSequence();              // Appel de startSequence
  }
  else if(a < arduinoNumber){
    lcd.setCursor(0, 1);          //Positionnement curseur sur
                                  //2e ligne
    lcd.print("Trop petit");      //Affichage sur LCD
    lcd.setCursor(0, 0);          //Positionnement curseur sur
                                  //1re ligne
}
  else{
    lcd.setCursor(0, 1);        //Positionnement curseur sur
                                //2e ligne
    lcd.print("Trop grand");    //Affichage sur LCD
    lcd.setCursor(0, 0);        //Positionnement curseur sur
```

```
                                //1re ligne
  }
  lcd.setCursor(2, 0);          //Positionnement curseur sur
                                //3e emplacement de la 1re ligne
  place = 0;
 }
}

int randomNumber(int minimum, int maximum){
  randomSeed(analogRead(5));
  return random(minimum, maximum + 1);
}

void startSequence(){
  arduinoNumber = randomnumber (MIN, MAX);  //Générer le nombre
                                            //à deviner
  lcd.clear();                              //Effacer écran LCD
  lcd.print("Devine un nombre");            //Affiche sur LCD
  lcd.setCursor(0, 1);                      //Positionnement curseur
                                            //sur 2e ligne
  lcd.print("de" + String(MIN) + " - " + String(MAX));
  delay(4000);                              //Attendre 4 secondes
  lcd.clear();                              //Effacer écran LCD
  lcd.print(">>");                          //Affiche sur LCD
}
```

Je ne souhaite pas trop m'étendre sur le sketch pour l'instant. J'ai fait en sorte que l'affichage ait lieu sur le LCD. J'ai ajouté aussi une méthode qui affiche un curseur clignotant à l'écran (figure 13-21). La fonction `atoi` (**A**SCII **to** **I**nteger) convertit une chaîne de caractères en un entier.

Méthode LCD : `blink`

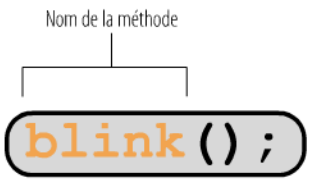


Figure 13-21
Méthode LCD `blink`

`blink` est appelée une seule fois dans la fonction `setup` et fait clignoter un curseur à l'endroit où on va écrire. Quand on regarde le début du sketch, on voit qu'il est tout à fait possible d'incorporer plusieurs bibliothèques dans un projet.

Il n'y a théoriquement aucune limite. La mémoire flash finit quand même à un moment par laisser entendre qu'elle est pleine et qu'aucun code ne peut plus être ajouté.

Il y a une chose que je ne comprends pas et je ne sais plus si vous l'avez déjà expliquée ou pas. On trouve par exemple la ligne :

```
lcd.print("de" + String(MIN) + "-" + String(MAX));.
```

Autrement dit, on affiche des chaînes de caractères et on utilise l'opérateur +. Mais comment fait-on pour additionner des chaînes de caractères ? Ça ne marche qu'avec des nombres, non ?

Bien sûr, Ardus ! Seules des valeurs mathématiques peuvent être additionnées. L'opérateur + ne peut évidemment rien additionner dans le cas de chaînes de caractères. Comment le pourrait-il ? Les différentes chaînes de caractères sont simplement réunies en une seule. On dit aussi qu'elles sont concaténées. Si maintenant, comme dans notre sketch, des valeurs numériques font partie de la chaîne de caractères à afficher, elles doivent être préalablement converties en une `string`. C'est la fonction `string` qui s'en charge, comme dans `String(MIN)`.

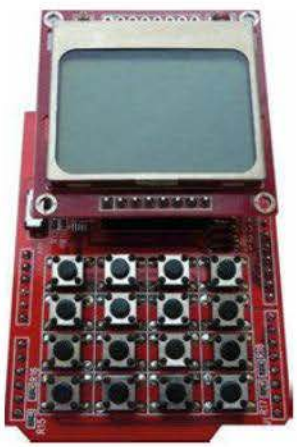
Pour aller plus loin

Voici quelques liens intéressants à propos d'Arduino et de LCD :

- *http://www.arduino.cc/en/Tutorial/LiquidCrystal*
- *http://www.arduino.cc/en/Reference/LiquidCrystal*
- *http://www.sparkfun.com/datasheets/LCD/HD44780.pdf*
- *http://arduino.cc/fr* (en français)

La figure 13-22 montre encore un shield pour clavier prêt à l'emploi, capable de recevoir un LCD.

Figure 13-22 ▶
Shield pour clavier 4 × 4 avec interface pour afficheur 5110 LCD



Pour du prêt à l'emploi somme toute relativement compact, le shield pour clavier 4 × 4 se veut idéal avec un porte-écran approprié. L'écran monochrome, qui a une résolution de 84 × 48 pixels, est

compatible avec l'afficheur 3310 LCD, pour lequel une bibliothèque Arduino existe.

Pour aller plus loin

Pour compléter ce chapitre, vous pouvez effectuer une recherche sur Internet sur les mots-clés :

- Nokia 5110 LCD ;
- 3310 LCD.

Problèmes courants

Si, après avoir raccordé le LCD et chargé le sketch, vous ne voyez rien à l'écran, vérifiez les points suivants.

- Le câblage est-il correct ?
- Pas de court-circuit ?
- Le trimmer du contraste est-il correctement branché ? Augmentez le cas échéant le contraste jusqu'à ce que vous puissiez voir quelque chose à l'écran.

Qu'avez-vous appris ?

- Vous avez raccordé pour la première fois un élément d'affichage, capable non seulement de clignoter mais aussi d'afficher des nombres et du texte.
- La bibliothèque `LiquidCrystal` vous a permis de commander facilement un LCD avec un contrôleur HDD44780.
- Vous avez ensuite clairement transposé le jeu des nombres à deviner.
- D'autres types de LCD vous ont été présentés pour vos expériences à venir, de telle sorte que vos créations puissent être sans limite.

Exercice complémentaire

Réfléchissez un peu à une serrure à code de sécurité, du type de celles installées aux entrées des zones sensibles.

Un code à plusieurs chiffres doit être saisi pour ouvrir la porte. On est bien entendu informé en cas de saisie erronée que le code chiffré

composé est trop bas ou trop élevé. Vous pouvez par exemple brancher un servomoteur désengageant un pêne du système de fermeture en cas de code correct. Il faut attendre un certain temps, par exemple trois minutes, après avoir tapé trois fois de suite un code erroné. Créez un contrôle d'accès à votre chambre pour décourager les colocataires ou proches trop curieux.