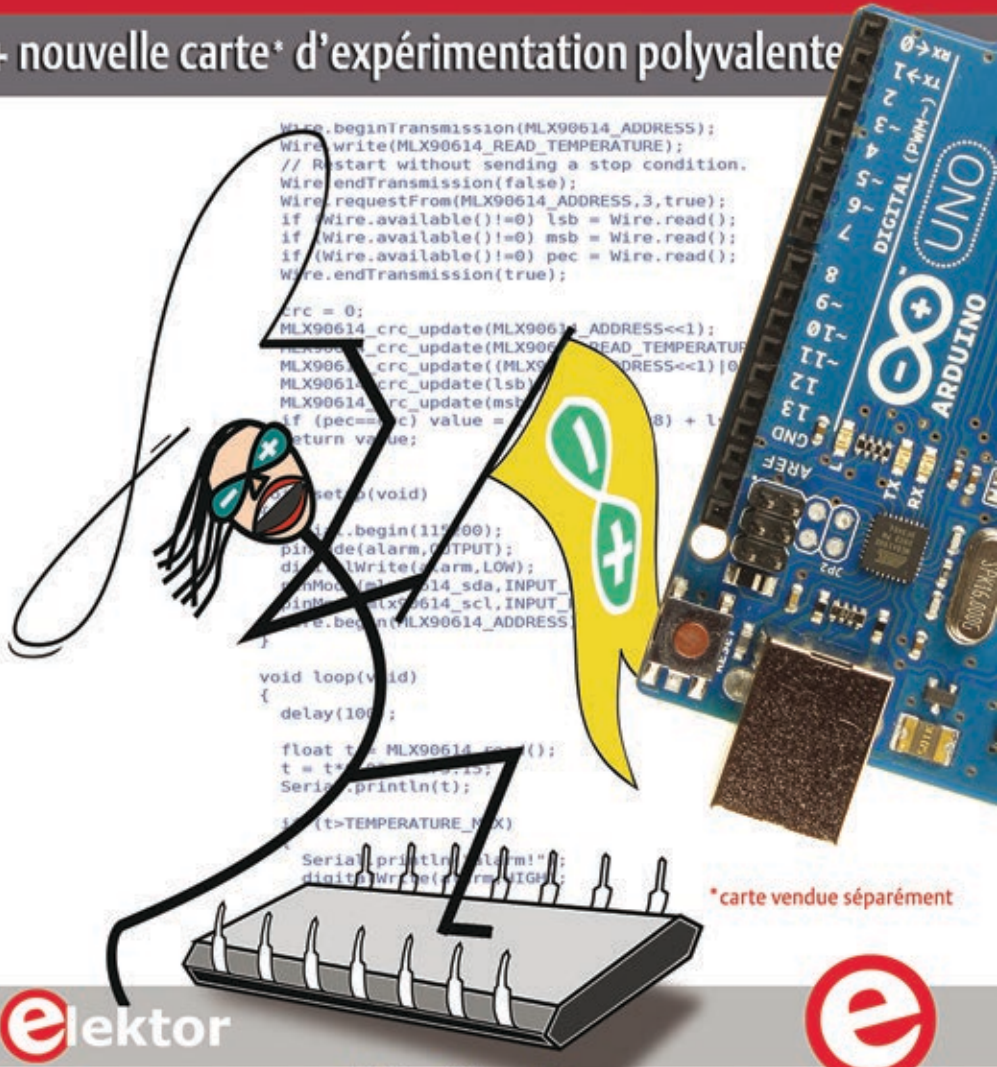
Clemens Valens

maîtrisez les microcontrôleurs à l'aide d'Arduino

2e édition revue et augmentée

+ nouvelle carte* d'expérimentation polyvalente



*carte vendue séparément

elektor

www.elektor.fr/arduino

Sommaire

2. Introduction

2.1 On frappe à la porte

Les microcontrôleurs ou micros sont des composants programmables fréquemment employés dans des applications de plus en plus diverses. Grâce à la diminution continuelle de leur coût et leurs performances toujours plus impressionnantes [*Toc toc toc ! Tiens, on frappe à la porte ? « Entrez ! »*], aujourd'hui il est souvent plus avantageux d'utiliser un micro plutôt que quelques composants discrets.

« Bonjour Denis, Mariline, qu'y a-t-il ? »

[*Denis et Mariline sont les rédacteurs qui relisent et corrigent le manuscrit de ce livre.*]

« Eh bien Clemens, je trouve ce début particulièrement rantanplan.

— Ah bon ?

— Oui, c'est le modèle du lieu commun ! Vous auriez par exemple pu interpeller votre lecteur et lui demander combien de microcontrôleurs il a rencontrés depuis qu'il s'est levé ce matin sans même s'en rendre compte.

— Ah ouais.

— Dans le radio-réveil, dans la chaudière de la salle de bain, dans la machine à café, dans l'ascenseur, dans la porte automatique dans la voiture, le train, le bus, etc. Si ça se trouve, il y en a eu des centaines !

— Il a une porte automatique dans sa voiture ?
— J'ai oublié une virgule.
— D'accord. Mais eh… il prend la voiture, le train et le bus ? Il habite si loin de son travail ?
— Vous m'avez très bien compris ! »

[*Denis et Mariline quittent la pièce.*]

Hum… Bon, j'en étais où ? Ah oui, les micros qui sont partout. Bref, même pour un simple clignotant, un microcontrôleur et une LED sont nettement mieux à leur affaire qu'un circuit classique. Le travail de dimensionnement des composants passifs ou actifs autrefois nécessaire est de nos jours souvent remplacé par la programmation d'un micro.

Le développement de circuits à microcontrôleur a longtemps été réservé aux spécialistes au sein des entreprises car les outils étaient chers et difficiles à mettre en œuvre. Tout cela a changé et réaliser un montage à microcontrôleur est désormais à la portée de l'électronicien amateur, voire le débutant (même moi j'y arrive). On ne compte plus les cartes à microcontrôleur bon marché ni les logiciels de programmation faciles et gratuits. Oui, les micros sont devenus banals.

Attention, banal ne veut pas dire enfantin. Même si les constructeurs s'efforcent de rendre cette technologie accessible au plus grand nombre, une bonne maîtrise du sujet est toujours nécessaire pour transformer en produit de qualité un prototype fabriqué avec des kits « grand public ».

2.2 Mais on va où ?

Ce livre a pour but de t'initier aux microcontrôleurs. Cette initiation reposera sur la plate-forme de prototypage rapide Arduino. Cette plate-forme, composée d'une carte de développement bon marché et d'un logiciel de programmation gratuit, permet d'introduire les différents aspects de la programmation des microcontrôleurs de façon accessible et à peu de frais.

Je présente les périphériques des microcontrôleurs les plus courants, le tout illustré par des exemples de programmation en Arduino. Ainsi tu pourras comprendre leur fonctionnement sans te perdre dans les détails. Ensuite nous examinerons chaque périphérique de plus près pour découvrir comment l'utiliser, ce qui te permettra de poursuivre. Ce livre n'est pas du type « 30 applications rigolotes avec Arduino ».

Il donne des applications rigolotes, mais il offre bien plus. À la fin du livre, si tu n'as rien sauté, tu seras capable de mettre en œuvre n'importe quel microcontrôleur et tu pourras quitter le cocon Arduino pour voler de tes propres ailes.

Pour profiter des chapitres qui suivent, je suppose que tu as un minimum de bagage en informatique et que tu sais :

- te servir d'un navigateur internet, de ton ordinateur et son gestionnaire de fichiers ;
- créer des dossiers et des fichiers ;
- manipuler des archives dans différents formats (notamment les plus courants pour ton système d'exploitation) ;
- retrouver tes téléchargements ;
- ce que sont un port USB et un port série ;
- ce que sont une résistance, un condensateur ou un transistor.

Pour me concentrer sur la programmation, je n'expliquerai pas tous les détails de l'électronique utilisée dans les exemples. Ce livre n'est pas une initiation à l'électronique. Aussi je dessinerai les schémas comme des schémas, et pas comme des œuvres d'art multicolores en 3D comme on en voit souvent dans des articles sur Arduino. De toute façon, les illustrations dans ce livre sont toutes en noir et blanc.

Bonne lecture !

4. Prototypage rapide à l'italienne

D'origine italienne, Arduino est une plate-forme de prototypage rapide pour applications à microcontrôleur. Cela veut dire qu'il s'agit d'un ensemble d'outils développé pour faciliter la conception de montages à base de microcontrôleur sans perdre trop de temps à en apprendre les tenants et aboutissants. Avec Arduino, toi aussi tu peux « faire du microcontrôleur » ou « faire de l'électronique embarquée », il suffit de te munir d'une petite carte électronique pas chère, d'installer un logiciel gratuit et de garder tes soirées libres (« voilà une offre qui ne se refuse pas »).

4.1 Le Parrain 1, 2 et 3

Arduino s'inscrit dans la lignée des projets Processing et Wiring et a pour but de vulgariser la technique, ce qui la rend accessible aux non-initiés, aux artistes, aux stylistes, aux passionnés et à ceux qui sont intéressés par la création d'objets ou d'environnements interactifs, mais qui n'ont pas suivi une formation technique et qui ne disposent pas forcément de beaucoup d'argent. Aussi les concepteurs d'Arduino ont-ils décidé que leur carte ne devrait pas coûter plus de 25 €, ce qui était en effet le prix de la première carte. Depuis, les prix ont grimpé un peu pour atteindre aujourd'hui environ 30 €.

Processing (`www.processing.org`) est un langage et un environnement de programmation à code source libre (*open source*) pour la création sur ordinateur d'images, d'animations et d'interactions. Processing a été développé « pour enseigner les fondements de la programmation dans un contexte visuel et pour servir comme carnet de croquis ou outil de production professionnel de logiciel ».

Wiring (`www.wiring.org.co`), italien comme Arduino, est un environnement de programmation avec cartes à microcontrôleur « pour explorer les arts électroniques, l'enseignement de la programmation et le prototypage rapide ».

Arduino (`www.arduino.cc`), conçu à la même école que Wiring, a simplifié encore plus le concept de Wiring, lui-même déjà assez facile à utiliser. A l'origine, Wiring n'était pas un projet libre mais, sous la pression de l'équipe Arduino, Wiring a cédé et a publié ses sources. Cela a permis à l'équipe Arduino de réaliser son rêve du microcontrôleur pour tout le monde.

Les premières cartes Arduino étaient plus élémentaires que celles de Wiring mais, au fil du temps, le matériel Arduino est devenu plus élaboré tandis que Wiring a étendu sa gamme en la simplifiant. Maintenant, les cartes Wiring S et Arduino Uno se ressemblent beaucoup.

Malgré l'équivalence des deux projets, Wiring n'a pas connu le succès fulgurant d'Arduino. Depuis le début, Arduino a tout fait pour répandre sa bonne parole puisque le code source, les schémas électriques, les dessins des platines et la documentation sont libres et ouverts ; là où Wiring a essayé de tout maîtriser. Au moment d'écrire ces lignes, plus de 600 000 cartes Arduino avaient été vendues dans le monde entier.

Les bibliothèques d'Arduino s'appuient sur celles de Wiring et l'environnement de développement intégré (EDI), basé sur Processing, semble identique à celui de Wiring. La seule différence notable est la couleur : orange pour Wiring et bleu clair pour Arduino. Processing quant à lui utilise les tons de gris dans un EDI avec les mêmes traits.

L'EDI de Wiring supporte les cartes Arduino mais l'EDI d'Arduino n'accepte pas le matériel Wiring. Au départ, Arduino se bornait à quelques microcontrôleurs AVR d'Atmel tandis que Wiring flirtait ouvertement avec les micros de Microchip, Texas Instruments ou encore ceux à base de cœur ARM. Arduino également s'est transformé en un système multi-plateforme, mais un peu « à l'insu de son plein gré ». La philosophie a été embrassée par des passionnés qui l'ont portée sur d'autres micros de constructeurs différents. Il a fallu quelques années à Arduino pour modifier l'EDI et ainsi permettre d'y intégrer d'autres compilateurs pour des micros différents mais, avec la version 1.5 et l'introduction de la carte Arduino Due avec son micro au cœur ARM Cortex M3, le but a été atteint.

4.2 Pâtes, fromage et sauce tomate

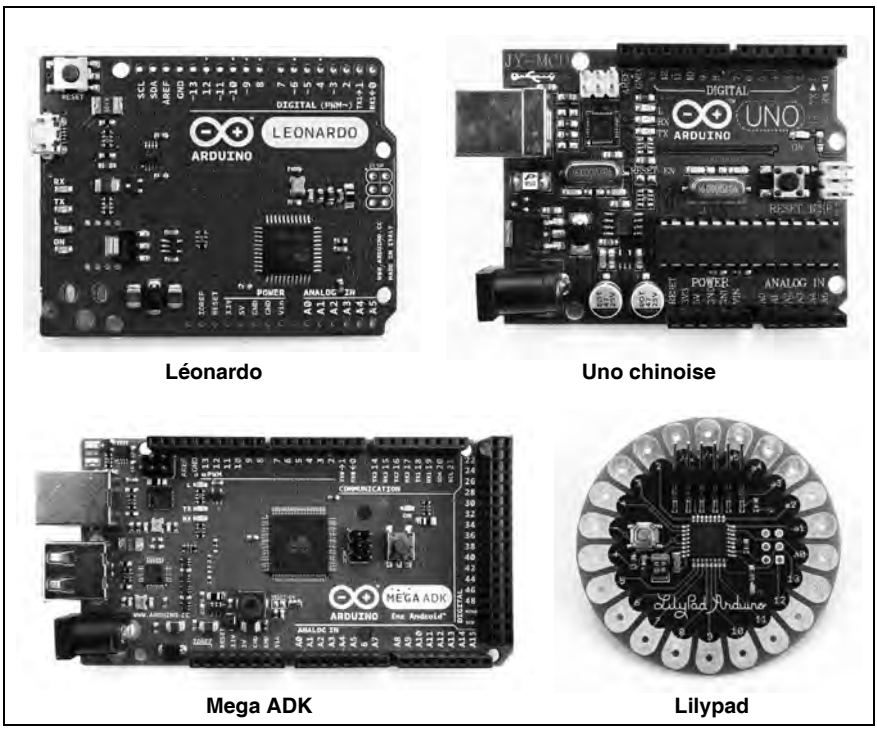
La plate-forme Arduino est construite sur trois piliers :

1. le pilier *Matériel* qui regroupe une collection de cartes à microcontrôleur et de cartes d'extension. Les schémas électriques des cartes et les dessins des circuits imprimés sont disponibles et gratuits ;

2. le pilier *Logiciel* qui comprend les outils de programmation et une bibliothèque étendue de fonctions de haut niveau. Tout est gratuit, à code source libre et multi-plateforme ;

3. le pilier *Distribution-Communication* sous forme d'un site internet par lequel le matériel et le logiciel Arduino sont mis à disposition des intéressés. L'adresse est `www.arduino.cc` (`arduino.cc` marche également). Pour les curieux, .cc est le domaine national réservé au territoire des îles Cocos (anciennement les îles Keeling), un territoire extérieur à l'Australie. Ce domaine a probablement été choisi pour son faible coût, mais je m'égare.

Le site internet est *la* référence pour tout ce qui concerne officiellement Arduino. Ce site regroupe les mises à jour logicielles et matérielles et toutes les informations nécessaires pour leur mise en œuvre. Si tu cherches un revendeur de matériel authentique, c'est ici que tu le dénicheras. Un forum pour poser les questions qui ne trouvent pas de réponse dans ce livre est également accessible. Le site est en anglais, mais il existe des forums dans d'autres langues, dont un en français.



Léonardo **Uno chinoise**

Mega ADK **Lilypad**

Figure 4-1 - Quelques cartes Arduino moins courantes : la Léonardo (sans connecteurs) est équipée d'un ATmega32U4 pour réaliser des dispositifs USB, la Lilypad en forme de pâquerette cible les applications de mode, la Mega ADK est capable de communiquer avec les dispositifs Android et une Uno chinoise qui ne porte pas la mention « made in Italy ».

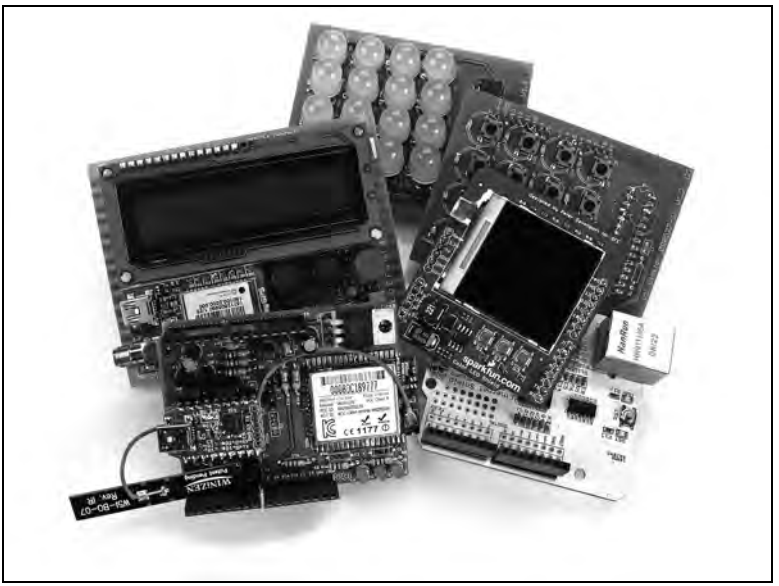


Figure 4-2 - Une collection de cartes d'extension compatibles avec Arduino. La photo montre un *shield* Wifi/Bluetooth, un *shield* Ethernet (blanc), un *shield* avec écran de téléphone portable, avec clavier matriciel (voir chapitre 6, page 73), un *monome* 10h (avec les 16 grosses LED blanches, voir aussi `www.monome.org`) et, au milieu à gauche, un *shield* d'expérimentation que nous construirons dans le chapitre 8 (page 159).

Côté matériel, Arduino est une carte à microcontrôleur avec un port USB. Plusieurs modèles existent, mais les plus courantes sont la petite carte au format Uno (Diecimila, Duemilanove…) et la plus grande au format Mega 2560 (Mega, Mega 1280, Mega ADK…). Il y en a d'autres comme la Mini, la Micro, la Nano ou encore la Lilypad en forme de pâquerette prévue pour les applications *wearable*, c'est-à-dire à intégrer dans les vêtements. La carte Arduino Due est différente des autres car elle est équipée d'un micro à 32 bits au lieu de 8 bits et elle est donc beaucoup plus puissante que les autres[1]. Il existe aussi une carte nommée Yún qui est une sorte de mélange d'une carte Linux et d'une Arduino Léonardo. La Yún vise les applications sans fil et dans les nuages.

1. La Due et la Yún ne sont pas traitées dans ce livre car elles compliquent considérablement les choses. Un peu de la simplicité d'Arduino a été sacrifiée à des micros beaucoup plus puissants.

7.4 Récréation : le Misophone

Ouf, après ce long exercice pour mettre en œuvre un régulateur PID, il est temps de se reposer. Pour cela, je te propose un montage sympathique qui nécessite un peu de mécanique, de l'électronique et de la programmation.

As-tu déjà entendu parler de la misophonie ? Probablement pas, car il s'agit d'une maladie peu connue, identifiée seulement dans les années 1990. La misophonie, à ne pas confondre avec la mysophobie, la phobie des poussières, est « une hypersensibilité aux bruits quotidiens, plus particulièrement à ceux que fait l'entourage en respirant ou en mangeant. Le malade peut en venir à des sentiments extrêmes de colère ou de panique, voire à des idées violentes envers la personne qui produit les sons. »[1] Quand, il y a quelque temps, j'ai visionné sur l'internet une vidéo sur une fourchette électronique qui émet un son au moment où l'aliment piqué sur elle entre en contact avec la bouche de son utilisateur, j'ai tout de suite pensé à la misophonie. Le but de la fourchette musicale, nommée *EaTheremin* par ses inventeurs japonais, est d'inciter les enfants à manger des légumes en les amusant. J'ai des doutes sur l'efficacité de cette approche assez surprenante mais une telle fourchette pourrait peut-être aider à sortir de leur isolement social les gens qui souffrent de misophonie. En effet, équipée de quelques fourchettes musicales, la personne souffrant de cette maladie pourrait à nouveau inviter des amis à venir manger chez elle, les bruits émanant de la bouche de ses hôtes seraient rendus inaudibles par le chant harmonieux des fourchettes musicales. Alors, comment réaliser une telle fourchette (que j'ai rebaptisée Misophone) ?

La fourchette tenue dans la main de l'utilisateur, l'aliment piqué sur la fourchette et le corps de l'utilisateur forment un circuit électrique qui se ferme quand l'aliment touche un endroit conducteur de l'utilisateur, comme sa bouche, mais un bout de peau nue fonctionne aussi bien. La fréquence du son obtenu dépend de la résistance électrique de l'aliment mais également du corps de l'utilisateur. Nous avons donc besoin d'un circuit capable de produire un son dont la fréquence est contrôlée par une résistance. Un tel circuit est assez facile à fabriquer avec uniquement des composants analogiques, mais un microcontrôleur aussi fait l'affaire. Il suffit de mesurer la résistance à l'aide d'une entrée analogique puis de délivrer sur une sortie numérique un signal carré dont la fréquence varie en fonction de la résistance mesurée. Pour avoir un ordre de grandeur, la résistance entre ma main et ma bouche est, selon mon multimètre, d'environ 1 MΩ. Avec une pile bouton de 3 V, j'ai mesuré des courants autour d'une dizaine, parfois même plusieurs dizaines, de

1. Source : *A Misophonia UK information leaflet*, réf. M001 FRENCH, www.misophonia-uk.org.

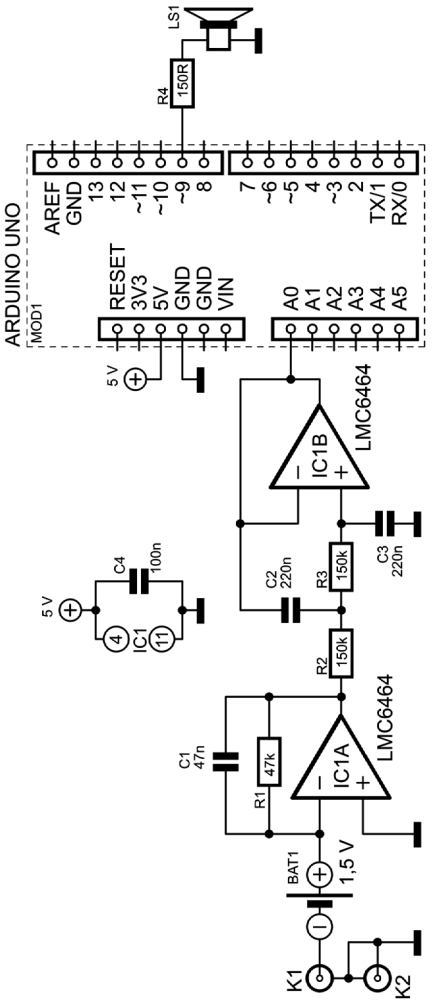


Figure 7-12 - Le circuit du Misophone.

microampères. Cela m'a donné l'idée de traiter la fourchette musicale comme un capteur qui fournit un courant dont l'intensité est une fonction de la résistance corporelle. Une résistance ou un ampli-op peut convertir ce courant en tension que le micro pourra numériser. Un filtre passe-bas inséré juste avant l'entrée du micro élimine les ronflements à 50 Hz dus au secteur et captés par le corps. Il est aussi possible d'implémenter ce filtre en logiciel, ce qui simplifiera encore plus le montage. Avec une fréquence de coupure assez basse (j'ai choisi 5 Hz), on obtient également une certaine lenteur qui se traduit en glissandos sympathiques. Des vibratos

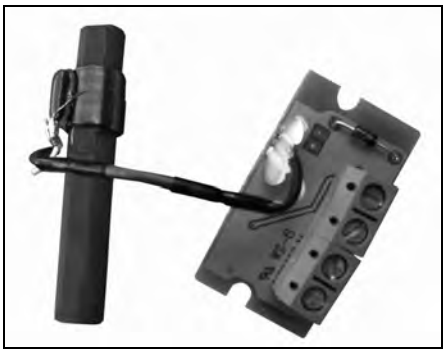


Figure 9-1 - Un module récepteur DCF77 bon marché et facile à trouver.

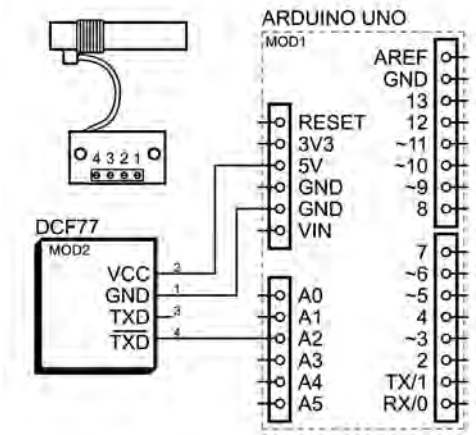


Figure 9-2 - Les modules récepteurs DCF77 sont faciles à mettre en œuvre.

nécessaire, celle intégrée dans le micro convient parfaitement. Pour une réception optimale, oriente l'axe de l'antenne, la barre de ferrite avec la bobine, perpendiculairement à Francfort.

Le schéma montre comment j'ai connecté le module à une carte Arduino. J'ai utilisé la sortie inversée (broche 4), du coup mes bits horaires sont actifs au niveau bas, mais cela ne change rien au fonctionnement. Puisque les valeurs des bits sont codées en durées, leurs niveaux n'ont pas d'importance. Comme entrée, j'ai choisi l'entrée analogique numéro deux car elle était encore libre sur ma carte d'expérimentation. Voici un petit croquis qui permet de vérifier que le module « voit » bien l'émetteur à Francfort et si les bits sont actifs au niveau haut ou bas.

```
/*
 * DCF77 polling pulse measuring
 */

int dcf77 = A2;
int pulse_prev;
unsigned long time_falling;
unsigned long time_rising;

void setup(void)
{
  Serial.begin(115200);
  pinMode(dcf77,INPUT_PULLUP);
  pulse_prev = digitalRead(dcf77);
  time_falling = millis();
  time_rising = millis();
}

void loop(void)
{
  unsigned long time = millis();
  int pulse = digitalRead(dcf77);

  if (pulse==1 && pulse_prev==0)
  {
    // rising edge
    time_rising = time;
    Serial.print("lo: ");
    Serial.println(time_rising-time_falling);
  }
  else if (pulse==0 && pulse_prev==1)
  {
    // falling edge
    time_falling = time;
    Serial.print("hi: ");
    Serial.println(time_falling-time_rising);
  }

  pulse_prev = pulse;
}
```

À chaque passage, la fonction *loop* demande d'abord le nombre de millisecondes écoulées depuis le lancement du croquis. C'est la fonction *millis* qui fournit cette valeur. Il existe une fonction similaire, *micros*, qui fournit le nombre de microsecondes écoulées depuis le lancement du programme. Puisque nous nous attendons à des bits qui durent au moins 100 ms, une résolution en microsecondes ne présente pas d'avantages.

```
        Serial.print("/20");
        Serial.println(year);
      }
      else Serial.println("error");
      bits = "";
      seconds = 0;
    }
  }
}
```

J'ai aussi resserré un peu les marges pour les durées des bits. Grâce à ce réglage et aux tests de parité, le décodage est ainsi assez robuste. Pour encore renforcer la robustesse, tu peux rajouter des contrôles sur les valeurs décodées. L'heure ne doit pas dépasser 23, les minutes 59, etc. Je te laisse cet exercice.

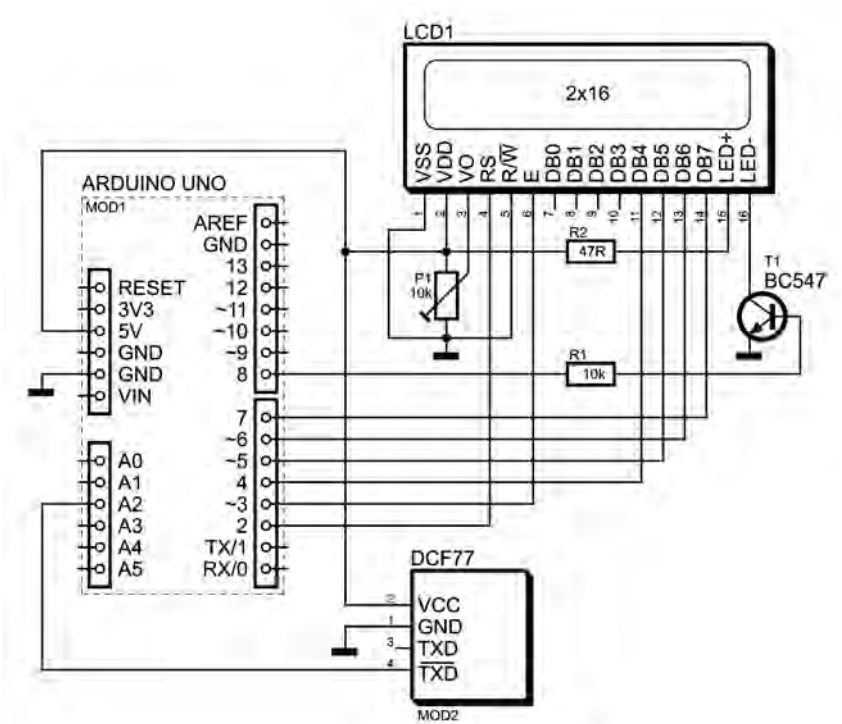


Figure 9-3 - Un LCD permet d'afficher l'heure reçue par le module récepteur DCF77.

9.4 millis et micros, deux petites fonctions

L'écriture du croquis pour décoder les trames DCF77 était relativement facile grâce à la fonction *millis* qui nous a permis de mesurer la durée des bits sans beaucoup d'effort. Cette fonction utilise un des modules temporisateur/compteur du micro pour compter le nombre de millisecondes écoulées depuis le démarrage du croquis. Le micro de la carte Arduino Uno possède trois de ces modules, celui de la carte Mega en a six. Les modules ne sont pas tous pareils. Le temporisateur/compteur 0 est un module à 8 bits, équipé de deux sorties indépendantes et doté de fonctions MLI. Il a été prévu par son concepteur pour gérer des tâches de synchronisation et pour produire des formes d'ondes rectangulaires.

Le temporisateur/compteur 1 est le plus complet et le plus précis des trois, il offre une résolution de 16 bits, deux sorties, de la MLI et de la capture d'événements. Ce module est à utiliser quand il faut produire des fréquences ou des longueurs d'impulsions avec exactitude ou quand il faut mesurer des durées avec beaucoup de précision. Le micro de la Mega est équipé de quatre de ces modules (1, 3, 4 et 5) qui, en plus, ont chacun trois sorties au lieu de deux.

Le temporisateur/compteur 2 est un module à usage général. Il offre une résolution de 8 bits, il dispose de deux sorties et il peut être cadencé par un quartz de montre de 32 768 Hz. Cela le rend particulièrement bien adapté à des applications qui ont besoin de l'heure. Malheureusement, dans le monde d'Arduino, cette fonction n'est pas exploitable car l'entrée du quartz est déjà occupée par le quartz de 16 MHz.

Pour les besoins d'Arduino, le temporisateur/compteur 0 est réservé aux fonctions *millis*, *micros*, *delay* et *delayMicroseconds* et il compte à une fréquence de presque 1 kHz, 976,6 Hz pour être exact. À intervalle régulier, environ toutes les 42 ms, le compteur de millisecondes est corrigé à cause de cette différence de fréquence. La valeur retournée par *millis* est toujours en retard, jusqu'à presque une milliseconde entière. Ce compteur n'est donc pas très précis. Un compteur de microsecondes proprement dit n'existe pas, cette valeur est calculée à chaque appel de la fonction *micros*. Contrairement au compteur de millisecondes, la valeur retournée par *micros* est exacte (ou presque, car elle ne tient pas compte du temps nécessaire pour exécuter la fonction). Vu l'imprécision de *millis*, la fonction *delay* utilise *micros*. La fonction *delayMicroseconds* effectue ses propres calculs et ne dépend de personne. Elle est assez précise, sauf pour de tout petits délais de seulement une ou deux µs.

9.5 C'est qui, Émilie ?

Sur le micro de la Uno, les compteurs/temporisateurs ont tous deux sorties et ils savent tous faire de la MLI, ce qui explique les six sorties MLI utilisables par *analogWrite*. Le micro de la Mega ne possède pas seize de ces sorties comme tu pourrais le croire, mais quinze, car la sortie supplémentaire du temporisateur/compteur 1 (par rapport à la Uno) partage une broche avec la deuxième sortie du temporisateur/compteur 0. Note que la documentation d'Arduino a un peu de mal à suivre l'évolution du matériel et du logiciel, une mise à jour ne sera pas du luxe. Au moment d'écrire ces lignes, elle affirme que la carte Mega n'a que douze sorties MLI (ce qui était le cas pour l'ancienne Mega).

9.5.1 Deux types de MLI

Les signaux MLI produits par *analogWrite* sont censés être indépendants de la sortie : quelle que soit la sortie MLI que tu utilises, le signal produit doit être le même. Cependant ce n'est pas le cas. Bien que les compteurs/temporisateurs fonctionnent tous en mode 8 bits avec une horloge de 16 MHz / 64 = 250 kHz (diviseur (*prescaler*) de 64), ils ne produisent pas pour autant la même fréquence. Les broches 5 et 6 qui sont reliées au compteur 0 tournent à presque 977 Hz (16 MHz / 64 / 256), les autres produisent une fréquence d'un peu plus de 490 Hz (16 MHz / 64 / 255 / 2). Les signaux MLI sur les broches 5 et 6 (4 et 13 sur la Mega) sont plus rapides mais moins symétriques que les autres qui, eux, bénéficient d'une option du micro nommé MLI à phase correcte (*Phase Correct PWM*). Le temporisateur/compteur 0 fonctionne en mode MLI rapide (*Fast PWM*). Cette différence est due au fait que le compteur 0 est aussi utilisé pour le compteur de millisecondes. Si tu utilises le compteur 0 en mode MLI à phase correcte, ce qui est tout à fait possible, le compteur de millisecondes ne fonctionnera plus, les valeurs fournies par *millis* et *micros* seront fausses et *delay* sera trop lente.

Pour produire un signal MLI, le compteur est comparé à une valeur de seuil. Si le compteur est en dessous de ce seuil, le signal MLI est 0 logique. Si le compteur dépasse le seuil, le signal MLI est 1 logique[1]. En mode MLI rapide, le compteur compte de 0 à 255, puis repart de 0, car 255 + 1 = 0 pour un registre à 8 bits. L'horloge du compteur est donc divisée par 256. La forme d'onde produite par le compteur est en dent de scie et les impulsions du signal MLI se trouvent toujours à gauche de la valeur maximale du compteur. En revanche, en mode MLI à phase correcte, le compteur ne repart pas de 0, mais change de direction. Après 255, la valeur suivante est 254 au lieu de 0. À chaque fois que le compteur atteint la valeur

1. Les niveaux peuvent être inversés.

maximale ou minimale, il change de direction. La forme d'onde produite par le compteur est alors triangulaire. Du coup, le compteur croise deux fois la valeur de seuil, en montant et en descendant, avec comme résultat que les impulsions du signal MLI sont maintenant centrées autour de la valeur maximale du compteur. La fréquence de ce signal est égale à un peu plus que la moitié de celle obtenue avec la MLI rapide, car l'horloge du compteur est dans ce cas divisée par 255 × 2. Pour mettre en évidence la différence entre les deux techniques, compare avec un oscilloscope les deux canaux du même module pendant qu'ils produisent deux rapports cycliques différents.

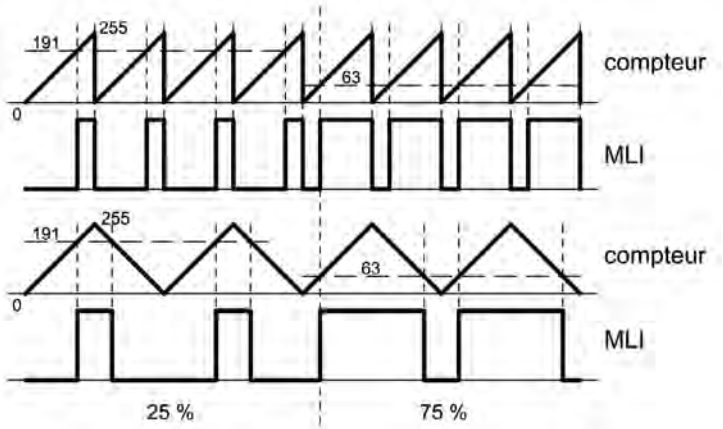


Figure 9-4 - La différence entre la MLI rapide (haut) et la MLI à phase correcte (bas).

9.6 Le Maître du Temps

À part *analogWrite*, l'API d'Arduino n'offre pas d'autres fonctions pour jouer avec des signaux MLI. C'est dommage, car le constructeur du micro est assez fier des capacités MLI de ses AVR. Alors, pour mettre en valeur les efforts du fondeur, je te présente ici une application spectaculaire et unique au monde (restons modestes) : un émetteur de signal horaire DCF77 à signal MLI.

Les horloges radiopilotées se synchronisent sur le signal horaire DCF77 (ou un autre émetteur) qu'elles reçoivent par voie hertzienne. Les signaux horaires des divers émetteurs, pas uniquement celui du DCF77, sont tellement peu protégés et leurs formats sont si bien documentés qu'ils forment des proies faciles pour des malveillants. En effet, il n'est pas compliqué de réaliser son propre émetteur de signal horaire compatible et d'embrouiller les horloges radiopilotées proches.

Ainsi elles affichent l'heure que tu veux au lieu de l'heure absolue. L'utilisation des brouilleurs de GPS ou téléphones portables est tout à fait légale sous certaines conditions, alors pourquoi pas un brouilleur de signal horaire ? C'est marrant, non ? Alors, comment faire ?

Le signal émis par l'émetteur DCF77 est assez simple. Il s'agit d'une porteuse, un signal sinusoïdal à 77,5 kHz, modulée en amplitude par l'information[1]. Pour envoyer un bit, l'amplitude de la porteuse est diminuée à 20% de sa valeur non modulée. La durée de la modulation détermine la valeur du bit. Un niveau logique 0 a une durée de 100 ms, un niveau logique 1 dure deux fois plus longtemps, 200 ms. Plusieurs techniques peuvent être employées pour produire un tel signal, comme un oscillateur sinusoïdal précis suivi par un atténuateur à deux niveaux (20% et 100%), contrôlé par un micro qui fournit les bits horaires. Or, produire une fréquence de 77,5 kHz est tout à fait à la portée d'un micro et ce serait donc un peu dommage de ne pas profiter de cette capacité. Le micro de la carte Arduino est cadencé à 16 MHz. Si nous divisons cette fréquence par 206, nous obtenons une fréquence de 77 670 Hz, soit une erreur de seulement 0,2%. Les récepteurs DCF77 du commerce ne sont probablement pas assez chatouilleux pour se laisser perturber par une si petite différence.

L'étape suivante est d'appliquer la modulation d'amplitude (MA, les anglophones disent AM, de *amplitude modulation*) à la porteuse. La solution qui vient à l'esprit est un pont diviseur de tension avec deux résistances, mais elle nécessite une broche supplémentaire du micro pour contrôler le diviseur. Or, il est possible de faire encore plus simple en utilisant la modulation de largeur d'impulsions ou MLI. À l'aide d'un « peu » de mathématique, l'analyse de Fourier en l'occurrence, il est possible de démontrer que chaque signal périodique peut être construit à partir d'une série de signaux sinusoïdaux. Le premier signal sinusoïdal dans la série est appelé le fondamental et sa fréquence est la même que celle du signal périodique. Les autres signaux sinusoïdaux sont les harmoniques. Leurs fréquences sont des multiples entiers de la fréquence fondamentale et leurs phases et amplitudes dépendent de la forme d'onde du signal périodique. La suite des signaux sinusoïdaux forme ce que l'on appelle le spectre du signal périodique et, comme l'empreinte digitale chez l'homme, il est unique pour chaque forme d'onde[2]. Ainsi

1. En réalité, le format n'est pas si simple que ça car le signal est aussi en partie modulé en phase par une séquence pseudo-aléatoire qui permet de corriger l'heure reçue en fonction de la distance entre l'émetteur et le récepteur. Heureusement pour nous, la grande majorité des récepteurs du commerce ne tiennent pas compte de cette information.
2. Si la valeur moyenne du signal périodique n'est pas égale à zéro, le spectre contiendra aussi une composante continue (0 Hz).

un signal carré contient uniquement des harmoniques dont les fréquences sont des multiples impairs de la fréquence fondamentale. Dans notre exemple, nous pouvons construire un signal carré à partir de signaux sinusoïdaux avec des fréquences de 77,5 kHz, de 232,5 kHz (×3), de 387,5 kHz (×5) et ainsi de suite.

Quand le rapport cyclique du signal carré dévie des 50%, les harmoniques pairs apparaissent également, comme celui à 155 kHz (×2) ou celui à 310 kHz (×4). La distribution des amplitudes change aussi. Non seulement les amplitudes des harmoniques dépendent du rapport cyclique, mais aussi celle du fondamental. Elle est au maximum pour un rapport cyclique de 50% et elle diminue quand le rapport cyclique change. Nous pouvons exploiter cette caractéristique pour moduler l'amplitude du fondamental. Pour notre brouilleur de DCF77, il suffit de choisir un signal rectangulaire avec un rapport cyclique en dessous de 50% et dont l'amplitude du fondamental est égale à seulement un cinquième de l'amplitude du fondamental du signal carré.

Quel rapport cyclique faut-il choisir ? Si tu es fort en maths, tu pourras le calculer, si tu ne l'es pas, comme moi, tu utilises ton oscilloscope ou multimètre. Il existe une autre solution : te procurer un logiciel d'analyse de Fourier (il y en a plein sur l'internet) et essayer plusieurs signaux jusqu'à obtenir le résultat voulu. J'ai utilisé la méthode suivante :
un filtre accordé sur la fréquence fondamentale permet d'extraire un beau signal sinusoïdal de 77,5 kHz modulé en amplitude. Ce filtre doit être sélectif sans pour autant être un filtre de compétition. Il doit filtrer tous les harmoniques à partir du premier à 155 kHz. La modulation d'amplitude modifie aussi le niveau moyen du signal, un filtre passe-bande qui par sa nature élimine le niveau moyen est donc le meilleur choix. J'ai obtenu de bons résultats avec un filtre actif du quatrième ordre (figure 9-5). Avec un oscilloscope ou un multimètre connecté à la sortie du filtre, tu peux mesurer l'amplitude du fondamental pour des rapports cycliques différents et ainsi trouver la bonne valeur.

La dernière étape côté matériel consiste à équiper l'étage de sortie d'un amplificateur et d'une antenne. La fréquence sur laquelle émettent les horloges atomiques est tellement basse qu'il faut une antenne immense pour obtenir une portée exploitable. La longueur d'onde λ à 77,5 kHz est de 3 871 m et l'antenne optimale de $\lambda/4$ fait donc presque 968 m de long. Même une antenne moins efficace de $\lambda/20$ nécessite toujours un câble, fil ou conducteur d'une longueur de presque 200 m. Ce genre d'antennes peut être construit en roulant le conducteur autour d'un cadre pour obtenir une antenne… cadre. J'ai choisi une autre approche, beaucoup moins fatigante, puisque j'ai tout simplement cannibalisé l'antenne d'un module récepteur DCF77. C'est une antenne à barre de ferrite qui ne supporte pas de fortes puissances, car la ferrite sature rapidement, mais qui présente l'avantage

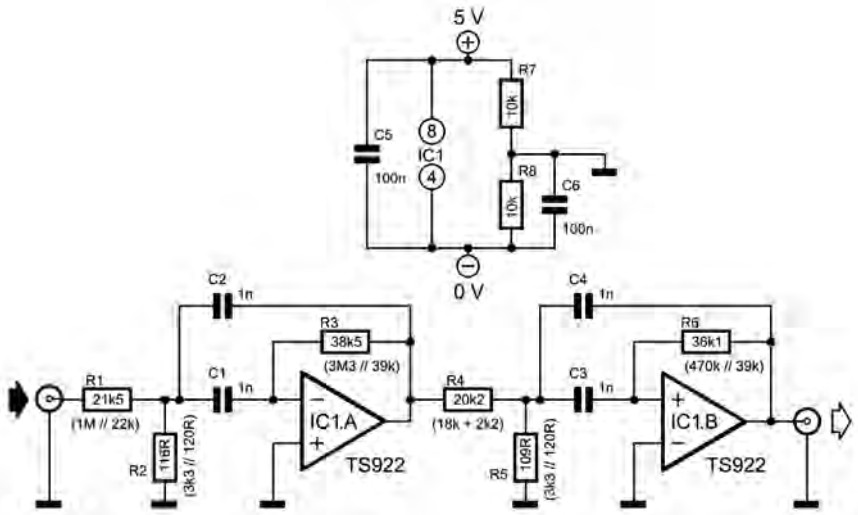


Figure 9-5 - Un filtre passe-bande Bessel du quatrième ordre permet de visualiser sur un oscilloscope le fondamental d'un signal MLI. Ce filtre a été calculé pour une fréquence centrale de 77,5 kHz. Les valeurs entre parenthèses permettent de réaliser les valeurs non standard. Attention à l'alimentation asymétrique.

non négligeable d'être déjà accordée sur la bonne fréquence à l'aide d'un condensateur soudé sur la bobine et par le placement de la bobine sur la barre de ferrite. Avec cette antenne, j'ai obtenu une portée d'environ deux mètres avec un module récepteur DCF77 identique. Pour une horloge de commerce bon marché, la portée était de 20 cm. Pour améliorer la portée, un amplificateur de plusieurs dizaines de watts et une bonne antenne d'émission sont nécessaires.

9.6.1 Émetteur DCF77 : MLI de haut vol

Pour mon montage final, je n'ai utilisé ni filtre ni amplificateur. Puisque l'antenne d'émission est un filtre LC et que le récepteur est équipé d'une antenne similaire, et puisque Dame Nature a tendance à atténuer les fréquences aiguës d'un signal, les harmoniques n'auront pas beaucoup d'influence sur le signal. Du coup, le rôle de l'amplificateur peut être endossé par le micro et j'ai pu connecter l'antenne à la broche 9 de la carte Arduino simplement à travers un condensateur pour bloquer la tension continue qui, dans ce montage, ne sert à rien.

maîtrisez les microcontrôleurs à l'aide d'Arduino

Clemens Valens

Pour cette 2e édition, l'auteur a créé une carte* d'expérimentation polyvalente à coupler avec une carte* Arduino. Ensemble, elles permettent de réaliser grand nombre des montages du livre et de présenter de nouveaux exercices. Ceux-ci fournissent à leur tour l'occasion de découvrir de nouvelles techniques.

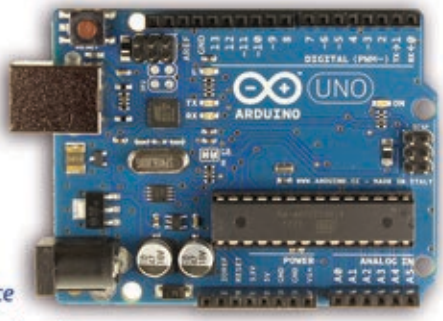
SOMMAIRE

Ce livre est pour vous si vous êtes :

- ✓ débutant en microcontrôleurs
- ✓ utilisateur de cartes Arduino (bricoleur, artiste, ...) qui souhaite approfondir ses connaissances
- ✓ étudiant en électronique
- ✓ enseignant à la recherche d'idées

Points forts

- ✓ Objectif double :
  1. Théorie : apprentissage général de la programmation des microcontrôleurs
  2. Pratique : montages sur carte Arduino avec l'environnement de développement Arduino
- ✓ Réalisations très originales
- ✓ Matériel peu coûteux ; logiciel gratuit et *open source*



L'ambition de cet ouvrage est de vous faire *entrer* dans le monde Arduino puis de vous en faire ressortir victorieux pour vous emmener plus loin dans l'apprentissage de la programmation des microcontrôleurs. Il met la théorie en pratique sur une carte Arduino avec l'environnement de programmation Arduino. Après ce parcours initiatique inédit, plaisant et ludique, vous programmerez vous-même n'importe quel µC. Ce livre sera donc votre premier livre sur les microcontrôleurs *avec une fin heureuse* !
Tous les programmes présentés peuvent être téléchargés.

*cartes vendues séparément

Changez de loisirs, devenez dresseur de puces !

C. Valens est responsable du laboratoire et du site Elektor.Labs. Amoureux de l'électronique, il élabore pour le plaisir, mais parfois aussi pour son employeur, des montages à µC. Polyglotte, il parle le C, le C++, le PASCAL, le BASIC et des dialectes de l'assembleur. Clemens passe une bonne partie de ses journées sur son ordinateur pendant que sa femme, leurs deux enfants et leurs deux chats s'efforcent d'attirer son attention (seuls les chats y parviennent).

Diff. Geodif – 42,50 €

elektor www.elektor.fr/arduino

9 782866 611958