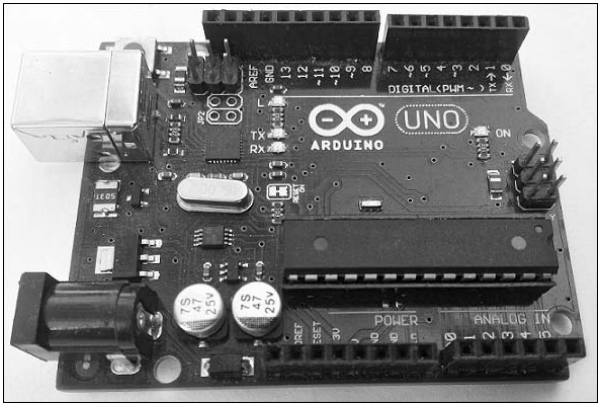
Chapitre 3
Matériel nécessaire

1. L'Arduino Uno

À l'heure où l'obsolescence condamne la plupart des objets électroniques âgés de quelques années, l'Arduino Uno fait figure d'exception. Il s'agit simplement de l'évolution du premier Arduino (cf. chapitre Présentation de l'Arduino - Évolution de l'Arduino). Mais il s'est déjà vendu à plus de 10 millions d'exemplaires et c'est loin d'être fini, parce que c'est l'Arduino le plus utilisé et le plus documenté.



L'Arduino Uno

Il est parfait pour débuter, car il n'est ni trop cher, ni trop grand. Il est suffisamment robuste et puissant pour commencer et possède un nombre d'entrées et de sorties suffisant à la plupart

des projets. De plus, il est compatible avec énormément de bibliothèques et de cartes d'extension, ce qui lui offre d'énormes possibilités d'évolution.

Caractéristiques techniques :

Microcontrôleur	ATmega328P
Tension d'entrée	7 V à 12 V
Tension de fonctionnement	5 V
Fréquence	16 MHz
Entrées analogiques	6
Sorties analogiques virtuelles (PWM)	6
Entrées/sorties numériques	14
EEPROM	1 ko
SRAM	2 ko
Flash	32 ko
Contrôleur USB	USB-B

La plupart des exemples présentés dans ce livre fonctionnent parfaitement avec l'Arduino Uno. Mais, parmi les autres modèles d'Arduino, certains sont peut-être plus adaptés à vos besoins. Vous en trouverez une description dans le chapitre qui leur est consacré (cf. Autres cartes Arduino).

2. Carte officielle, clones et contrefaçon ?

2.1 Un vrai Arduino

Choisissez une boutique fiable. Vous pouvez aussi consulter la liste des revendeurs de votre pays, sur le site officiel Arduino : https://www.arduino.cc/en/Main/Buy

Cette liste n'est bien sûr pas exhaustive, vous pouvez sûrement acheter ailleurs, mais méfiez-vous des prix trop intéressants : ils cachent souvent des contrefaçons.

L'Arduino Uno coûte généralement un peu plus de 20 €, ce n'est pas très cher si l'on considère qu'il est fabriqué dans un pays où les salaires et les conditions de travail sont correctes (Italie ou USA). De plus, cela permet de récompenser le travail des créateurs de cette merveilleuse petite carte.

2.2 Les clones 100 % compatibles

Ces cartes sont tout à fait légales tant qu'elles ne portent pas le nom d'Arduino. Elles sont souvent de bonne qualité et possèdent les mêmes fonctions que la carte officielle. Si elles sont beaucoup moins chères, c'est simplement parce qu'elles sont fabriquées en Chine.

Elles utilisent en général un autre type de contrôleur USB, ce qui peut (suivant votre configuration) vous obliger à télécharger et installer le bon driver. Cependant, une fois reconnue, la carte se comporte exactement comme un véritable Arduino.

2.3 Les clones spéciaux

Ils utilisent généralement le même microcontrôleur que les cartes officielles mais possèdent souvent un format. Ce type de carte ne cherche pas à imiter un modèle existant, mais plutôt à innover en proposant des options ou des fonctionnalités supplémentaires. Cela peut être l'ajout de LED, de capteurs, d'une matrice de LED (Rainbowduino), d'un lecteur de cartes SD (Pinguino), un design particulier (Diavolino, Boarduino), etc.

Ce sont des cartes intéressantes pour les utilisateurs avancés qui souhaitent bénéficier de ces nouvelles fonctions. Cependant, certaines innovations peuvent se faire au détriment de la compatibilité avec les cartes officielles.

2.4 La contrefaçon

Tout le monde a parfaitement le droit de fabriquer son propre clone. Pourtant l'Arduino peut être victime de contrefaçon. Cela s'explique certainement par la différence de prix entre le modèle officiel et un clone. Le site officiel Arduino explique comment distinguer le vrai du faux : https://support.arduino.cc/hc/en-us/articles/360020652100-How-to-spot-a-counterfeit-Arduino

Mais il faut aussi faire preuve de bon sens, si vous trouvez sur Internet un Arduino soi-disant fabriqué en Italie mais vendu au tiers du prix depuis la Chine, il y a peu de chances qu'il s'agisse d'un original.

Cependant, vous pouvez aussi l'acheter en toute bonne foi, s'il est proposé à un prix normal par un vendeur européen.

3. Les outils

3.1 Le multimètre

Exemples de multimètres (celui de droite est auto-range)

Il devient rapidement indispensable à tout bricoleur. Il permet de tester la tension (voltmètre), l'intensité (ampèremètre) et la polarité (distinguer le plus et le moins) d'une alimentation inconnue. Il est aussi très utile pour identifier rapidement la valeur d'une résistance (ohmmètre), ou pour vérifier que l'un des fils n'est pas coupé (testeur).

3.1.1 Choisir un multimètre

Il en existe à tous les tarifs. Les moins chers coûtent moins d'une dizaine d'euros alors que les plus chers peuvent atteindre plusieurs centaines. Pour un usage occasionnel, vous pouvez débuter avec un modèle à trente euros, mais évitez quand même de mesurer des tensions pouvant être dangereuses (supérieures à 50 V) avec ce type de multimètre.

Évitez cependant d'acheter le moins cher, ou préparez-vous à devoir rapidement rafistoler et/ou changer les fils, car ils sont souvent très fragiles et ne résistent pas très longtemps aux manipulations.

Si votre budget le permet, choisissez de préférence un modèle qui intègre l'auto-range (sélection automatique du calibre), plus confortable et plus facile à utiliser par un débutant.

3.1.2 Voltmètre

- Pour mesurer un courant continu, branchez le fil noir sur la fiche **COM** et le rouge sur la fiche **VΩmA**. Puis tournez le sélecteur jusqu'à **V=**.
- Si vous ne disposez pas de l'auto-range, choisissez une valeur supérieure à la tension supposée (20 pour mesurer une tension de 2 à 20 V ou 200 pour mesurer une tension de 20 à 200 V, etc.). Si vous ne connaissez pas la tension approximative, commencez par le plus élevé et réduisez ensuite pour obtenir plus de précision. Appliquez la pointe du fil rouge sur le (+) et la pointe du fil noir sur le (-). La valeur affichée à l'écran est la tension (en volts). Si la valeur est négative, cela veut dire que vous avez branché le rouge sur le (-) et le noir sur le (+). Ce n'est pas grave, mais ça permet de différencier le (+) du (-).
- Pour mesurer un courant alternatif, c'est le même principe, mais il faut sélectionner **V~** au lieu de **V=**.

Que ce soit en continu ou en alternatif, mesurez toujours la tension en parallèle.

3.1.3 Ampèremètre

- Pour mesurer l'intensité, sélectionnez **mA** et procédez comme pour mesurer la tension. Toutefois, l'ampèremètre doit toujours être branché en série avec un autre composant (une résistance).

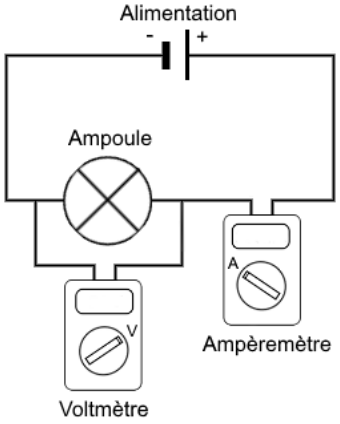
3.1.4 Ohmmètre

Ne l'utilisez que sur les composants qui ne sont pas alimentés en électricité.

- Sélectionnez **Ω**. Et faites comme pour le voltmètre ou l'ampèremètre. L'ohmmètre mesure la résistance d'un composant. Mais si vous faites se toucher les pointes rouge et noire, l'écran indique 0. Il est donc possible de l'utiliser aussi comme testeur, si votre multimètre ne possède pas cette fonction spécifique.

3.1.5 Testeur

Si vous possédez cette fonction, vous devez avoir un petit symbole représentant des ondes sonores. Le principe est simple. Si les fils noir et rouge sont reliés, vous entendez un petit bip. Cela permet facilement (et sans regarder l'écran) de savoir s'il y a une coupure dans un fil ou un circuit, une mauvaise soudure ou encore un faux contact. Il est également possible de tester un élément spécifique, comme par exemple un fusible ou un interrupteur.



Prises de mesures avec un voltmètre et un ampèremètre

3.2 Le fer à souder

Exemples de fers à souder

Comme pour le multimètre, il faut trouver un compromis entre la station de soudage de pro et le budget d'un amateur. Cela dit, il n'y a rien de plus frustrant qu'un mauvais fer à souder à moins de dix euros. Il ne chauffe pas assez. La panne (le côté qui chauffe au bout du fer) s'abîme et se déforme rapidement. Et il faut finalement en racheter un autre. Ce qui n'est pas spécialement économique.

La petite station que l'on voit sur la photo ne coûte qu'une trentaine d'euros, mais convient parfaitement à un usage occasionnel ou pour débuter. Évidemment, si vous soudez beaucoup, il vaut mieux investir un peu plus.

Certains petits accessoires peuvent aussi être très utiles :

La troisième main, un support modulable, avec des pinces crocodiles et qui permet de maintenir en place les pièces à souder. Il existe des modèles avec une loupe et même parfois une petite lampe.

Une pompe et/ou de la tresse à dessouder, très utiles pour retirer un composant sur un circuit imprimé ou afin le récupérer ou de le changer (s'il est défectueux).

Un support qui ne craint rien, comme une vieille table ou une planche à poser pour protéger le plan de travail. Vous éviterez ainsi de brûler votre bureau ou de faire un trou dans la nappe de la salle à manger.

N'oubliez pas le fil de soudure. Pour l'électronique, choisissez un fil assez fin (pas plus de 1 mm de diamètre).

Évitez de respirer les fumées. Même si les fils récents ne devraient plus contenir de plomb, ils emploient d'autres produits chimiques qui sont également toxiques. Vous pouvez utiliser un extracteur de fumée (avec un filtre à charbon) ou au moins un petit ventilateur afin d'éviter d'être exposé aux vapeurs (ou de souder en apnée).

Le secret pour réaliser une bonne soudure c'est de chauffer le support avant d'appliquer l'étain, sinon l'étain reste sur la panne du fer au lieu d'aller sur le support. Évidemment, c'est une question de dosage, si vous le chauffez trop, cela peut aussi abîmer le support. Un bon entraînement si vous débutez consiste à étamer un fil électrique. Il faut d'abord dénuder le fil, puis recouvrir la partie en cuivre d'une petite couche d'étain. Cela évite qu'il s'effiloche et le prépare à être soudé par la suite.

Le fer doit être à la bonne température (environ 350°) et nettoyé régulièrement en essuyant la panne sur une éponge métallique ou humide.

Chapitre 3

Projet 1 - Décodeur de message infrarouge

1. Présentation

La télécommande infrarouge (IR) fait partie de notre quotidien depuis le milieu des années 1970. Une multitude d'appareils sont pilotables à distance grâce à cette technologie : téléviseur, porte de garage, voiture, box internet, casque audio, etc.

Après la réalisation de ce premier projet, nous serons capables de vérifier le bon fonctionnement d'une télécommande IR, de déchiffrer les signaux qu'elle envoie, et de les enregistrer afin de les utiliser dans le second projet et de pouvoir ainsi piloter n'importe quel appareil IR à partir d'une Arduino.

1.1 Principe de fonctionnement

Après avoir terminé le montage, il sera possible de pointer une télécommande dans sa direction. L'appui sur les boutons de celle-ci déclenchera le décodage et l'affichage des trames d'informations IR dans le **moniteur série** de l'IDE Arduino.

1.2 Notions abordées

Ce premier projet permettra d'évoquer l'utilisation du **moniteur série** de l'Arduino, de faire appel aux librairies, de différencier les entrées analogiques et numériques de l'Arduino, et de s'initier aux protocoles d'échanges de données en électronique.

2. Matériel nécessaire

Le montage est construit autour d'un composant principal, le récepteur IR VS1838B.

Récepteur IR VS1838B

Le VS1838B est d'un rapport qualité/prix assez intéressant puisqu'il est possible de le trouver pour moins d'un euro. Il capte les ondes IR au travers de sa photodiode, le signal est amplifié pour être ensuite filtré, et finalement traité de manière logique afin d'en extraire des trames d'impulsions. Ces impulsions sont alors disponibles sur sa broche de signal S.

Pour réaliser ce montage, nous utilisons également une carte Arduino Uno afin de réaliser toute la partie « intelligente » de décodage et d'affichage des signaux.

Carte Arduino Uno

Remarque

Comme pour tous les montages présentés dans ce livre, toute autre carte Arduino « classique » (Mega, Nano, Due...) peut également faire l'affaire moyennant quelques adaptations simples du montage et/ou du programme. C'est une bonne manière pour le lecteur de s'approprier le projet.

3. Schéma et montage

Comme dans tout protocole d'échange d'informations, il y a un émetteur (ici, la télécommande IR), et un récepteur (l'appareil à piloter).

Nous décrirons brièvement le fonctionnement de l'émetteur pour nous attarder ensuite sur la raison d'être de ce premier projet : le récepteur IR.

3.1 La télécommande IR

Il existe plusieurs technologies de télécommande (infrarouge, radio, ultrason, etc.). Celle qui nous intéresse ici permet d'envoyer des signaux vers un récepteur IR grâce à un composant électronique bien connu et largement utilisé : la LED (*Light Emitting Diode*, ou DEL en français, diode électroluminescente). Les LED peuvent émettre dans toutes les couleurs mais la première à avoir vu le jour, en 1962, fut la LED IR qui émet donc dans la lumière non visible.

Remarque

Il est possible, malgré tout, de voir la LED d'une télécommande IR fonctionner en l'observant à travers un appareil photo numérique comme celui d'un smartphone.

Comment reconnaît-on une télécommande IR ? S'il est nécessaire de pointer la télécommande vers l'appareil à piloter pour que cela fonctionne, c'est qu'elle est infrarouge. Autre indice : nous pouvons distinguer une petite coque (la LED) sur la partie supérieure de la télécommande.

Télécommande IR avec sa LED d'émission

Il existe aussi des modèles avec un cache de protection en plastique noir mais transparent afin de laisser passer les infrarouges.

Autres télécommandes IR

L'émission de messages IR sera vue plus en détail dans le projet 2 - Télécommande infrarouge.

3.2 Le récepteur IR

Toute émission de signal IR par un émetteur est destinée à être reçue par un récepteur IR. Bien sûr, le protocole d'échange doit être connu des deux appareils pour qu'ils puissent se comprendre. C'est pourquoi il y a autant de télécommandes que d'appareils télécommandés, il serait effectivement fâcheux que la porte du garage réagisse à la télécommande du téléviseur.

En revanche, le récepteur que nous allons construire ici sera capable de comprendre toutes les télécommandes IR, rien de dangereux à cela puisqu'il ne pilotera finalement rien.

Brochage du récepteur IR

Le récepteur IR VS1838B dispose de trois pattes (ou pin), de gauche à droite :

- la sortie de signal S, sur laquelle les impulsions IR peuvent être récupérées ;
- la pin G (pour GND ou *Ground*, comprenez 0 volt) ;
- l'alimentation Vin, capable de supporter entre 3 et 5 V.

3.3 Montage

Le montage est relativement rapide puisqu'il consiste à relier un unique composant à l'Arduino Uno. Câblez le récepteur VS1838B de la manière suivante :

- La sortie de signal S, à relier à l'entrée numérique 7 de l'Arduino.
- La pin G, à relier au GND de l'Arduino.
- L'alimentation Vin, à relier à la sortie 5 V de l'Arduino.

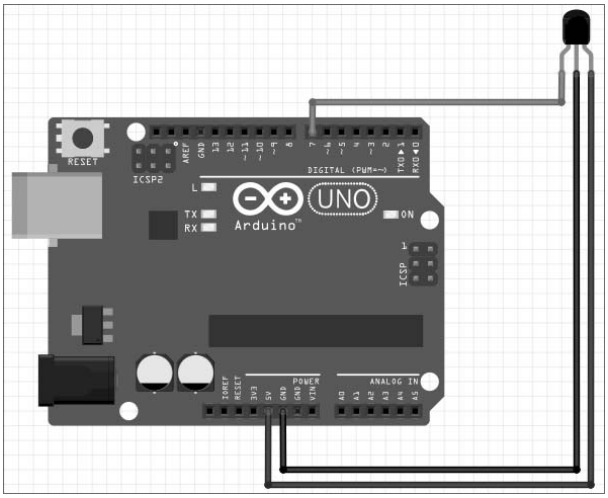


Schéma de montage du décodeur IR

Entrée analogique/numérique

Pourquoi relier la pin de signal du VS1838B à une entrée numérique de l'Arduino ? Il existe deux types d'entrées :

- Les entrées numériques ou digitales permettent de recevoir un signal de type HIGH ou LOW (1 ou 0 pour parler en binaire) selon que le voltage appliqué à la pin d'entrée soit respectivement de 5 V ou 0 V.

Remarque

En réalité, le seuil est plus bas, et tout voltage supérieur à environ 2,5 V est considéré comme étant un signal HIGH. Inversement, tout voltage inférieur à 2,5 V est considéré comme étant un signalement LOW. Cette explication est elle-même très simplifiée mais l'idée est là. À noter également que ce seuil peut varier légèrement selon les composants, ou les fabricants.

- Les entrées analogiques, qui ne se limitent pas à 0 ou 1 mais acceptent des valeurs proportionnelles à la tension appliquée. Elles peuvent aller de 0 à 255, 1023 voire plus, tout dépend de la capacité du convertisseur analogique/numérique (CAN, ou DAC (en anglais), *Digital Analog Converter*) implémenté dans le microcontrôleur. Si la tension appliquée à l'entrée est de 0 V, la valeur résultante sera 0. Si la tension est de 5 V, le résultat sera la valeur maximale autorisée par le DAC. Celui de l'Arduino est un DAC 10 bits, les valeurs des entrées analogiques peuvent donc varier entre 0 et 1023.

Dans le cas de ce montage, la sortie « signal » du VS1838B est soit à l'état HIGH, soit à l'état LOW selon les impulsions reçues par le capteur. Il n'y a donc pas de valeur intermédiaire possible. Dans ce cas d'usage, il est préférable de choisir une entrée numérique.

4. Programmation de l'Arduino

La structure d'un programme (ou *sketch*, nom fréquemment rencontré sur les sites web destinés à l'Arduino) est composée de plusieurs blocs :

- l'initialisation (ou *setup*) ;
- la boucle d'exécution (ou *loop*).

Mais avant cela, il est nécessaire de déclarer l'ensemble des librairies, des constantes et des variables globales utilisées dans nos deux fonctions `setup()` et `loop()`.

4.1 Les déclarations préliminaires

La majorité des programmes Arduino nécessitent l'utilisation d'une ou plusieurs librairies. Une librairie (ou bibliothèque) est un ensemble de déclarations et de fonctions destinées à un usage particulier. Le nombre de bibliothèques disponibles est impressionnant, et il existe pratiquement une bibliothèque pour chaque usage. L'étape de recherche de librairie fait donc partie de la démarche classique dans un projet Arduino. La multitude de sites web ou de chaînes YouTube traitant de l'Arduino est une source inépuisable d'informations, mais qu'il faut parfois traiter avec un peu de recul et un œil aguerri. Le plus simple reste sans doute d'ouvrir le gestionnaire de bibliothèque de l'IDE Arduino en cliquant sur **Outils - Gérer les bibliothèques** afin d'afficher la fenêtre suivante, et d'avoir un aperçu et une brève description de toutes les librairies disponibles :

Fenêtre de gestion des bibliothèques

N'hésitez pas à naviguer dans ce menu pour vous faire une idée de toutes les possibilités offertes.

Pour ce premier projet, nous incluons la librairie **IRremote** qui nous dispense de programmer toute la partie liée à la réception des trames d'impulsions sur la pin 7 de l'Arduino, leur décodage, la reconnaissance des protocoles, et bien d'autres choses :

```
#include <IRremote.h>

#define PIN_SIG 7
```

La seconde ligne définit la constante `PIN_SIG` à une valeur de 7, cette constante nous servira dans la suite du programme pour faire référence à la pin de l'Arduino sur laquelle est connectée la pin de signal du VS1838B.

Pourquoi utiliser une constante de type macro `#define` plutôt qu'une variable ou mieux une constante C++ ? Pour plusieurs raisons :

- Cette valeur ne va jamais changer au cours de l'exécution du programme, elle est constante.
- Pour optimiser le code : à la compilation du programme, l'IDE Arduino va remplacer toutes les références à `PIN_SIG` par sa valeur, ce qui se traduit par un gain de temps d'exécution (accès direct à la valeur, sans référencement), et de place (pas d'espace mémoire réservé pour la variable).
- Le langage utilisé sur Arduino est très souvent le C, parfois le C++. Il est donc plus prudent d'utiliser `#define` plutôt que `const`.

Sur des programmes volumineux, cela peut vraiment faire une différence surtout en termes d'espace mémoire. Prenez donc l'habitude d'utiliser les macros de préprocesseur (`#define`, `#ifdef`, etc.)

4.2 La fonction setup()

Le programme à télécharger dans l'Arduino doit effectuer plusieurs tâches :

- Démarrer le gestionnaire de communication série.
- Initialiser toutes les variables nécessaires.
- Configurer les pins de l'Arduino.

C'est la partie de « setup ». Tout programme Arduino dispose d'une fonction `setup()` permettant d'initialiser et de mettre en place tout ce qui sera utile par la suite.