INNOVATIONS... MONTAGES FIABLES... ÉTUDES DÉTAILLÉES... ASSISTANCE LECTEUR

ELECTRONIQUE ET LOISIRS

ELECTRONIQUE magazine

www.electroniquemagazine.com

n°144 Automne 2018

- Barrière laser
- Pédale multi-effets
- Générateur HT modulaire
- Apprenons à utiliser RandA - 3
- Antiblackout - 2
- Cours MPLAB X - 4
- CAO KICad EDA - 4
- Indicateur d'état de batterie

Demoboard MP3

Videogame Arcade avec Raspberry Pi

N° 144 Septembre 2018

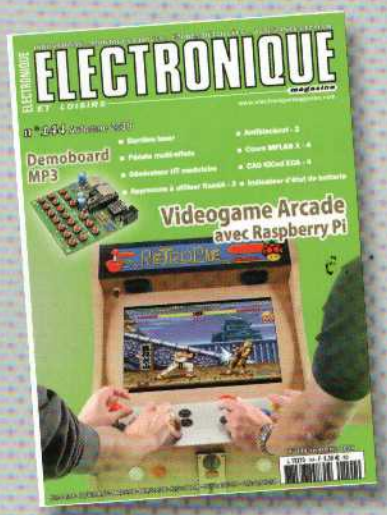
L 13270 - 144 - F: 8,30 € - RD

France 8,30€ - BEL/LUX : 8.80€ - PORT : 8.80€ - DOM/S : 8.30€ - NC/S : 990 CFP – POL/S : 1300 CFP – CAN : 12.70$ CAD

Sommaire

ARTICLES

Numéro 144
Automne 2018
100 pages



Les typons des circuits imprimés et les programmes **lorsqu'ils sont libres de droits** sont téléchargeables

L'EDITO AUTOMNE 2018

Chères lectrices et lecteurs,

Les vacances sont terminées avec hélas le retour aux habitudes quotidiennes. C'est pour cela que nous vous proposons dans ce numéro de septembre le projet RetroPie (basé sur un RaspberryPi) qui vous fera voyager dans le temps et qui vous permettra de (re)découvrir le charme des jeux d'arcade des années 80. Il s'agit de créer une véritable machine avec un coffret en bois et un monnayeur capable d'accepter de vraies pièces, comme celles que l'on trouvait dans les bars et les salles de jeux dans ces années là.

Un retour nostalgique du passé pour les lecteurs d'un certain âge et une opportunité pour les plus jeunes de découvrir comment leurs pères se sont amusés il y a de nombreuses années.

Ce projet représente l'esprit de notre magazine, à savoir apprendre l'électronique et la programmation en s'amusant.

Pour la mise en œuvre du projet RetroPie, nous n'avons pas utilisé une technologie vieille de 30 ans, mais plutôt un RaspberryPi avec un système d'exploitation Linux.

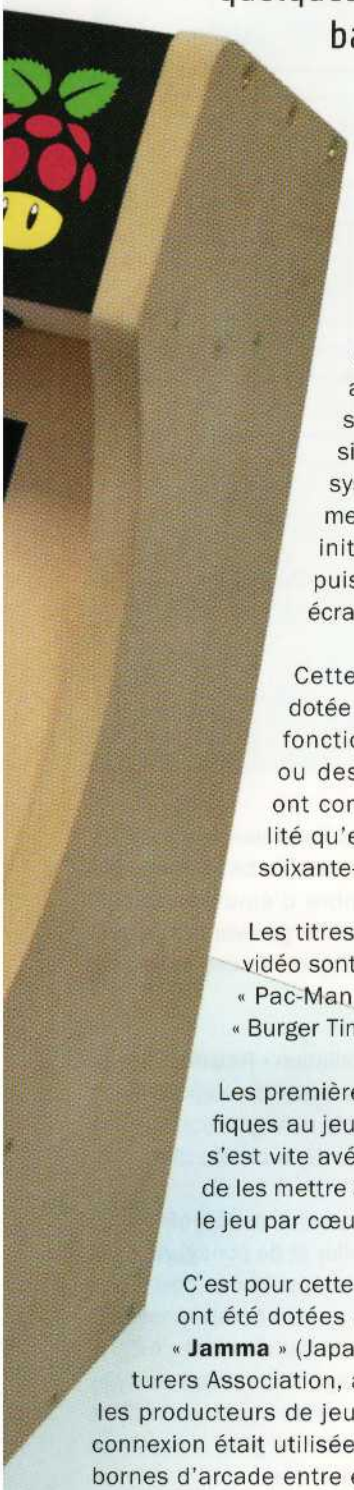
La Rédaction

à l'adresse **www.electroniquemagazine.com** dans le sommaire de la revue 144 et à l'onglet « Télécharger ».

JEU VIDÉO D'ARCADE AVEC RASPBERRYPI

de Roberto Testi

Nous vous proposons de reconstruire, à l'aide d'un RaspberryPi, d'une carte additionnelle décrite ici, d'un écran et de quelques boutons, un jeu vidéo de bar afin de redécouvrir le charme des jeux d'arcade des années 80.



Vous avez probablement entendu parler des jeux vidéo d'arcade, mais peut-être, si vous n'avez pas 35 ou 40 ans, vous ne savez pas ce que cela signifie. Un jeu vidéo d'arcade ou tout simplement un jeu vidéo de bar, est un système électronique inséré dans un meuble, généralement en bois, composé initialement d'une carte électronique puis d'un véritable ordinateur avec un écran à tube cathodique !

Cette machine de divertissement est dotée d'un monnayeur, c'est-à-dire qu'elle fonctionne avec des pièces de monnaie ou des jetons. Les jeux vidéo d'arcade ont connu un grand succès, tant en qualité qu'en popularité, de la fin des années soixante-dix à la fin des années 90.

Les titres qui ont marqué l'histoire des jeux vidéo sont « Space Invaders » publié en 1979, « Pac-Man » (1980), « Donkey Kong » (1981), « Burger Time » (1982) et beaucoup d'autres.

Les premières armoires produites étaient spécifiques au jeu qu'elles contenaient, mais le choix s'est vite avéré trop coûteux, car il était difficile de les mettre à jour face au public qui connaissait le jeu par cœur.

C'est pour cette raison que des armoires génériques ont été dotées de connecteurs standard nommés « **Jamma** » (Japan Amusement Machinery Manufacturers Association, association japonaise qui regroupe les producteurs de jeux vidéo d'arcade). Cette norme de connexion était utilisée pour brancher les systèmes et les bornes d'arcade entre eux. « Jamma » était un connecteur standard pour les cartes mère des jeux vidéo d'arcade, ce qui permettait de changer simplement la carte du jeu (nouveau jeu) au lieu de l'armoire entière.

Dans les années où la mode du jeu vidéo était à son zénith, il n'existait pas de technologie « domestique » bon marché (le PC était hors de prix à ces débuts) qui permettait au plus grand nombre d'en posséder un.

Ainsi, à l'époque, pour jouer à un jeu vidéo il fallait se rendre dans un bar ou dans une salle de jeu dédiée à ces machines. Il existait des versions moins abouties qui fonctionnaient sur de petites consoles bien moins puissantes (bien souvent avec un écran en noir et blanc à cristaux liquides).

Mais avec l'avènement des consoles plus modernes, les jeux d'arcade ont conservé le charme du passé et sont toujours appréciés du jeune public, même si les salles de jeux d'arcade n'existent plus.

De plus avec la disparition des jeux dans les bars, de nombreux amateurs ont essayé de récupérer des machines à restaurer pour les mettre dans leur maison ou dans leur garage.

Pour ceux qui ont raté cette occasion, nous proposons dans cet article une version qui, en profitant de la disponibilité de cartes à microprocesseur puissantes mais économiques telles que le RaspberryPi, vous permettra d'avoir un vrai jeu d'arcade à la maison.

Il suffit de construire une carte d'interface avec des boutons, un joystick et un monnayeur (pour les pièces de monnaie) afin de recréer l'ambiance d'une vraie machine de bar. Nous allons décrire toute l'électronique nécessaire et vous donner quelques suggestions pour construire le coffret. Il ne vous restera plus qu'à faire fonctionner votre imagination pour les décorations.

Le logiciel que nous vous proposons utilise des émulateurs de jeux d'arcade très répandus et qui sont disponibles depuis plus de 10 ans sur les PC.

Le projet

Comme « cerveau » du projet nous avons choisi la version **RaspberryPi 3** (mais il est possible d'utiliser la version 2) et comme logiciel « **RetroPie 4.3** » (ou ultérieur). Ce dernier est l'image Linux la plus complète en termes d'émulation de jeux

d'arcade et elle est plus facile à utiliser que les autres images nécessitant une configuration plus complexe.

Avec **RetroPie**, vous pouvez **émuler différents types de consoles du passé**, notamment la **Nintendo**, la **Game Boy**, la **Sega Master System**, l'**Amiga**, le **Commodore**, etc.

Dans notre cas, nous allons nous concentrer sur l'**émulateur MAME** (Multiple Arcade Machine Emulator), célèbre dans le monde entier car il permet d'émuler pratiquement tous les jeux des années 80 et 90, que nous avons pu trouver.

Dans notre projet, nous allons créer une véritable machine (avec le coffret) de type MAME (Multiple Arcade Machine Emulator) pour laquelle nous avons besoin des éléments suivants :

- un **RaspberryPi 2** ou **3** ;
- une carte **SD** de **16 Go** ou supérieure ;
- une alimentation pour le RaspberryPi (5 V 2 A via microUSB) ;
- un **joystick** de type arcade ;
- des boutons de type arcade ;
- des câbles pour connecter les joysticks et les boutons pour le RaspberryPi;
- un coffret pour fixer l'ensemble ;
- un câble HDMI ;
- un **moniteur** CVBS ou HDMI ;
- un câble audio (en option, car vous pouvez utiliser les haut-parleurs d'un téléviseur avec une interface HDMI) ;
- un clavier USB de type PC (pour la

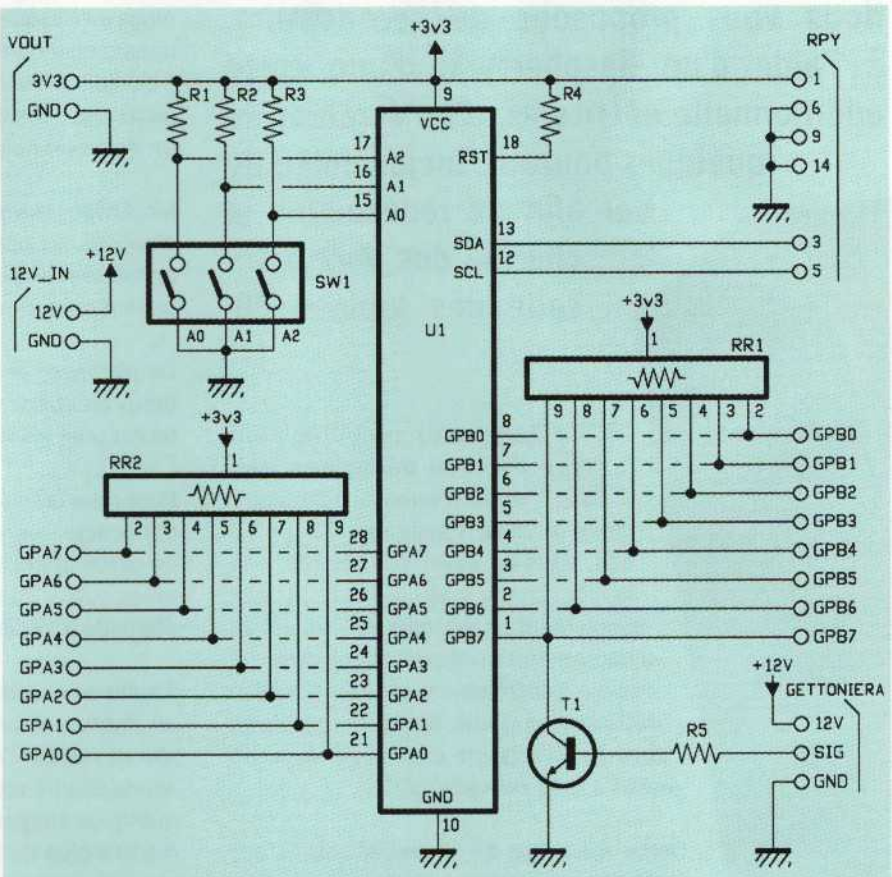


Schéma électrique de l'interface.

seule contenant le code que le microprocesseur du jeu vidéo utilisait à l'époque.

Dans les anciens jeux vidéo, pour reprogrammer (ou upgrader) l'unité il fallait remplacer physiquement la ROM, alors qu'avec un émulateur il suffit de charger le fichier appelé ROM correspondant au jeu désiré.

Avant de continuer, concentrons-nous

Le projet utilise Raspbian comme système d'exploitation de base et intègre un grand nombre d'émulateurs correspondant à un large éventail de systèmes informatiques de ces dernières décennies.

Le script d'installation « **RetroPie Setup Script** » est le cœur du logiciel pour l'installation et la configuration de tous les composants de la distribution.

Il existe déjà une image dotée d'une installation complète pour le script d'installation de RetroPie, et qui est fournie avec tous les émulateurs et toutes les fonctions prises en charge. L'image de la carte SD RetroPie est prête à l'emploi et fournit une installation complète de tous les systèmes et fonctionnalités pris en charge par le script d'installation de RetroPie.

Voici les principales caractéristiques :

- **installation complète de tous les émulateurs** disponibles pris en charge par le script d'installation RetroPie (voir le paragraphe « Émulateurs » ci-dessous) ;
- **démarrage automatique de la station d'émulation**, du « front-end » pour la navigation et l'exécution de la ROM. La station d'émulation est gérée par « Aloshi », il s'agit d'un « front-end » qui prend en charge la navigation sans clavier et qui contient également les sources des stations d'émulation de GitHub ;
- **écran de démarrage** (splash screen) **préconfiguré** et qui peut être personnalisé avec le script « RetroPie-Setup » ;
- actions « **SAMBA** » (interopérabilité des programmes Windows pour Linux) pour chaque système, afin de copier les ROM sur le RaspberryPi à partir du réseau ;
- outil « **daemon USB** » pour copier la ROM sur le RaspberryPi à partir d'une clé USB.

Émulateurs

RetroPie permet d'émuler de nombreux jeux d'arcade et de plates-formes de jeu des années 80/90. Il incorpore les émulateurs suivants :

- 3do ;
- Amstrad CPC ;
- Apple II ;
- Apple Macintosh ;
- Atari 2600 ;
- Atari 5200 and 8 bit series ;
- Atari 7800 ;
- Atari Jaguar ;
- Atari Lynx ;
- Atari ST/STE/TT/Falcon ;
- Commodore 64 ;
- Commodore Amiga ;
- FinalBurn Alpha ;

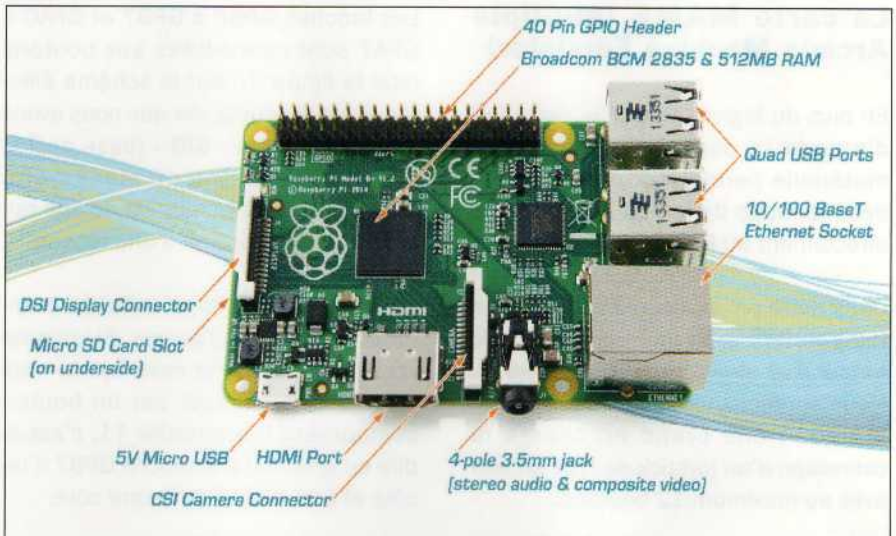


Figure 1 : ici les connexions disponibles pour un RaspberryPi 2B (identiques pour la version 3).

- Intellivision ;
- MAME ;
- MSX ;
- Neo Geo ;
- Neo Geo Pocket (Color) ;
- Nintendo 64 ;
- Nintendo DS ;
- Nintendo Entertainment System ;
- Nintendo Game Boy ;
- Nintendo Game Boy Color ;
- Nintendo Game Boy Advance ;
- Nintendo Virtual Boy ;
- PC ;
- PC Engine/TurboGrafx-16 ;
- PlayStation 1 ;
- ScummVM ;
- Sega 32X ;
- Sega CD ;
- Sega Dreamcast ;
- Sega Game Gear ;
- Sega Megadrive/Genesis ;
- Sega Master System ;
- Sega Saturn ;
- Sega SG-1000 ;
- Super Nintendo Entertainment System ;
- Vectrex ;
- Videopac or Odyssey2 ;
- WonderSwan (Color) ;
- Zmachine ;
- ZX Spectrum.

En plus de cela, il y a des émulateurs appelés « Ports » qui peuvent être installés au choix :

- KODI ;
- Minecraft Pi Edition ;
- Cave Story ;
- DOOM ;
- Duke Nukem 3D ;
- Quake Series ;
- Descent ;
- Super Mario War.

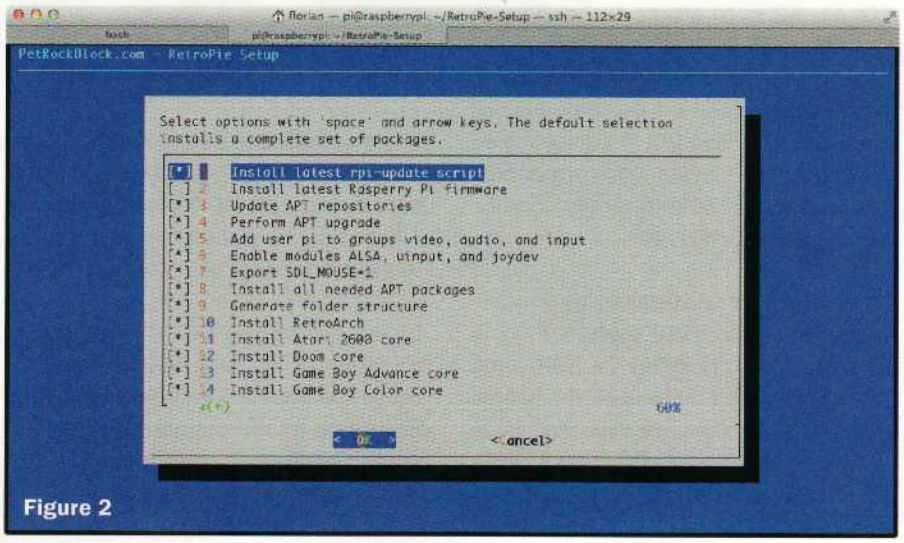


Figure 2

La carte MAME (Multiple Arcade Machine Emulator)

En plus du logiciel RetroPie, notre jeu d'arcade se compose d'une partie matérielle permettant de connecter des manettes de jeu ou des joysticks directement au RaspberryPi.

Dans notre projet, **nous interfaçons le RaspberryPi avec une carte réalisée autour d'un circuit intégré MCP23017**. Nous appelons cette dernière « **carte MAME** ». Elle prend en charge la connexion d'un joystick de type arcade avec au maximum 12 boutons.

Nous vous rappelons que pour la plupart des ROM MAME, il est nécessaire d'avoir 2 ou 3 boutons, alors que si vous voulez émuler d'autres plateformes, nous vous recommandons d'ajouter au moins 6 boutons pour le jeu et 2 boutons pour les fonctions de démarrage et de sélection.

La carte MAME ne fait rien d'autre que de se connecter au bus I2C du RaspberryPi via le circuit MCP23017 afin de transmettre l'état des boutons et du joystick. Le MCP23017 permet une extension des entrées/sorties, il est doté de 16 broches dont chacune peut être configurée en entrée ou en sortie.

Il s'interface grâce au bus I2C et permet de gérer 2 groupes de 8 entrées/sorties chacun. Il permet d'étendre le port GPIO du RaspberryPi, qui à lui seul ne pourrait pas tout gérer.

Grâce à cela, nous pouvons gérer 8 joysticks de type arcade, chacun avec un maximum de 12 boutons. Les broches A0, A1, A2 permettent de configurer l'adresse I2C de la carte MAME, nous pouvons en monter 8 au maximum (les 3 broches permettent 8 combinaisons d'adresses I2C soit 2^3). Si par exemple, vous utilisez 2 cartes, il faudra définir deux adresses I2C différentes à l'aide des broches A0, A1, A2.

L'**adresse est configurée** au moyen du commutateur **DIP switch SW1** à 3 voies. Les trois lignes de SW1 sont munies de 3 résistances de tirage (pull-up) permettant de maintenir à un niveau logique haut les broches A0, A1, A2 lorsque les contacts correspondant sont ouverts (OFF).

Les broches GPB0 à GPB7 et GPA0 à GPA7 sont connectées aux boutons (voir la figure 3). Sur le schéma électrique, vous pouvez voir que nous avons inséré la broche « **SIG** » (base de T1) pour le **mécanisme du monnayeur.** Cette broche est facultative, elle ne fait qu'émuler la pression d'une touche.

Vous trouverez les détails du monnayeur à la fin de l'article. Si vous ne voulez pas utiliser le monnayeur, vous pouvez le remplacer par un bouton contournant le transistor T1, c'est-à-dire en le reliant à la broche GPB7 d'un côté et à la masse de l'autre côté.

Toutes les lignes d'entrée du MCP23017 sont équipées de résistances de tirage (pull-up) grâce aux réseaux de résistances RR1 et RR2.

Réalisation pratique

Pour la « carrosserie » du jeu, vous pouvez choisir entre le « **bartop** » c'est-à-dire uniquement le **tableau de bord** et l'**écran** ou la **vraie** (grande) **cabine d'arcade**. Nous avons réalisé les deux modèles.

Le premier ressemble à un tableau de bord avec deux joysticks, des boutons et un monnayeur. Sur la face arrière se trouve une prise HDMI pour pouvoir connecter le jeu à n'importe quel téléviseur. Il est facilement transportable et n'est pas encombrant. Ses dimensions sont de 70 cm x 21 cm.

Le deuxième modèle est le jeu d'arcade classique, c'est-à-dire la grande cabine que l'on trouvait à l'époque dans les bars, avec le moniteur intégré ... pour les plus nostalgiques.

La structure peut être réalisée en panneaux de particules de 18 mm d'épaisseur ou avec du MDF (Medium Density Fiberboard ou panneaux de fibres de moyenne densité) de même épaisseur.

En ce qui concerne l'électronique, c'est-à-dire la carte MAME, le typon du circuit imprimé est disponible sur notre site en téléchargement dans le sommaire détaillé de la revue. Il s'agit d'un circuit imprimé simple face facilement reproductible.

Le montage des composants ne pose pas de difficultés car ils sont traditionnels (traversant). Le circuit intégré U1 sera doté d'un support. Vous devez l'orienter correctement.

Si vous utilisez le monnayeur, vous devez **monter le transistor T1** (en respectant son orientation), la **résistance R5** et le **bornier** dénommé « GETTONIERA » (cela veut dire monnayeur). **Sinon**, reliez la broche **GPB7 à la masse**. Reportez-vous à l'encadré intitulé « Plan de montage ».

La carte doit être complétée en soudant les borniers requis et, en correspondance avec le connecteur dénommé « RPY », une barrette mâle/femelle à 20 pôles sur deux rangées au pas de 2,54 mm doit être soudée.

La partie femelle doit se situer du côté de la face inférieure du circuit imprimé (côté cuivre), tandis que la partie mâle doit être du côté composants (face supérieure) de façon à pouvoir insérer 2 cartes (en sandwich).

Une fois la carte terminée, insérez-la dans le connecteur GPIO du RaspberryPi en faisant correspondre les points 1 et 2 des deux cartes.

Ensuite, connectez à l'aide des borniers les boutons que vous avez placés sur la face avant de la console que vous avez réalisée.

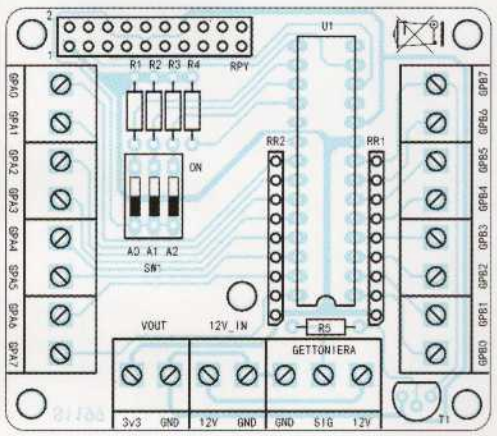
Faites de même avec le joystick et le mécanisme du monnayeur (sinon le bouton à installer en remplacement).

Si vous montez deux cartes MAME, la tension de 12 V (uniquement nécessaire si vous utilisez le monnayeur) doit être appliquée uniquement à celle sur laquelle le monnayeur est connecté.

Installation de l'image sur la carte SD

Une fois que vous avez installé tout le matériel nécessaire et assemblé le coffret, nous pouvons commencer par installer le logiciel. À l'aide d'un PC, téléchargez le fichier d'archive image « **RetroPie4_3.rar** » à partir de notre site.

Plan de montage de la carte MAME



Plan de câblage des composants de la carte MAME.

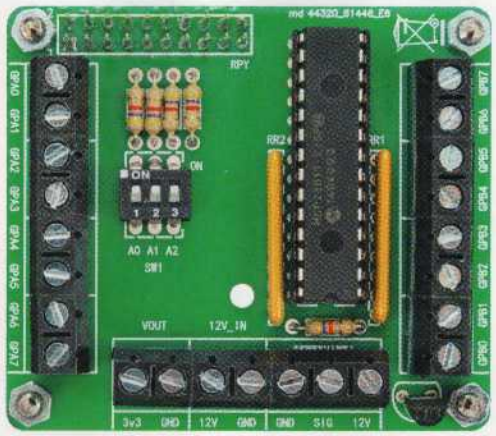


Photo de l'un de nos prototypes de la carte MAME.

Liste des composants

R1.....4,7 kΩ
R2.....4,7 kΩ
R3.....4,7 kΩ
R4.....4,7 kΩ
R5.....4,7 kΩ
RR1....réseau de résistances 8 x 4,7 kΩ + C
RR2....réseau de résistances 8 x 4,7 kΩ + C

U1.....MCP23017
SW1...DIP switch 3 voies
T1 .....BC547

Divers

Support circuit intégré 2 x 14 broches
Bornier 2 pôles (x 10)
Bornier 3 pôles

Barrette mâle/femelle 10 pôles (x 2)
Entretoise mâle/femelle 15 mm (x 4)
Vis 8 mm 3 MA (x 4)
Ecrou 3MA (x 4)

NB : le typon du circuit imprimé à l'échelle 1 est disponible en téléchargement dans le sommaire détaillé de la revue.

Il contient les configurations des boutons selon la disposition de la figure 5 ainsi que le BIOS des émulateurs. Il permet en plus la gestion du monnayeur. Décompressez le fichier que vous venez de télécharger, et installez le fichier « .img » sur la carte SD (attention la carte SD sera formatée).

Sous Windows, vous pouvez utiliser le programme « **Win32DiskImager** ». Sous Macintosh, vous pouvez utiliser le programme « RPI-sd card builder ». Sous Linux, vous pouvez utiliser, en faisant attention au chemin de la carte, la commande suivante :

```
sudo dd if=CHEMIN_FICHIER_IMG of=-
CHEMIN_CARTE_SD bs=1m
```

Nous vous recommandons d'utiliser une carte SD de 16 Go ou plus, car l'image prend environ 7,5 Go sans compter que nous devons insérer les différentes ROM des jeux.

À ce stade, vous devez obtenir un résultat semblable à celui de la figure 4 pour « Win32DiskImager » :

1. **sélectionnez l'image** que vous voulez écrire sur la carte SD (vous devez d'abord la décompresser) ;
2. **sélectionnez le périphérique** où se trouve la carte SD, assurez-vous qu'il s'agisse du bon périphérique, car la procédure efface complètement toutes les données présentes sur la carte SD ;
3. **cliquez** sur le bouton « **Write** », puis débranchez le périphérique lorsque le processus est terminé (le temps dépend des performances de votre PC). Maintenant, vous disposez d'une carte SD amorçable pour votre système.

Une fois la copie terminée, retirez la carte SD du PC et insérez-la dans le RaspberryPi. Revenons maintenant à la figure 5, où les noms des boutons sont ceux originaux des jeux d'arcade.

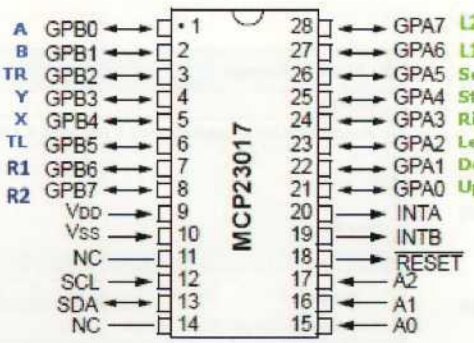
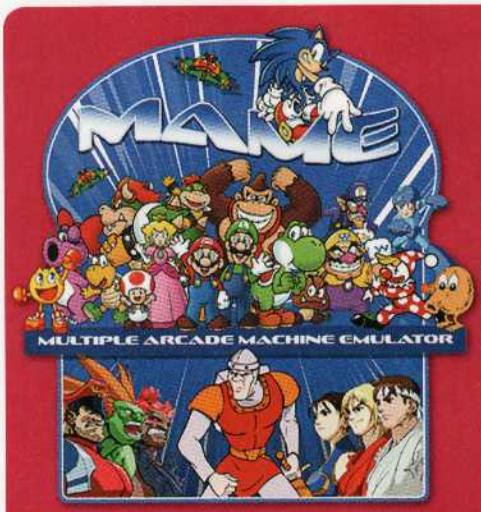


Figure 3 : brochage des entrées/sorties du MCP23017 et correspondance avec les boutons et les mouvements de la manette (joystick).

L'histoire de MAME



Le projet est né de la volonté de Nicola Salmoria, qui a commencé à travailler sur un émulateur universel le 24 décembre 1996 avec l'idée de combiner plusieurs émulateurs trouvés sur le net (y compris un émulateur primitif de Pac Man) dans un seul programme capable de faire fonctionner plusieurs jeux.

La première version, la « 0.1 », est sortie le 5 février 1997. Elle s'exécutait dans l'environnement MS DOS en lignes de commandes, elle était capable de faire « tourner » 5 jeux. En peu de temps, le projet a commencé à attirer l'attention d'autres développeurs, qui ont commencé à collaborer en insérant de nouveaux drivers pour le fonctionnement des jeux vidéo, grâce à une architecture particulière du code source.

En 1997 débute le développement en parallèle de « MAME32 » (maintenant MAMEUI), qui était une version adaptée à l'environnement Windows 32 bits avec l'ajout d'une interface graphique qui facilitait son utilisation. En 1998, la version « 0.34 » supportait près de 1000 jeux.

Depuis 2001, la version officielle de MAME n'est plus développée sous MS DOS mais sous Microsoft Windows, toujours sans interface graphique, en tant que programme de lignes de commandes. La dernière version stable de MAME est la « 0.135 », sortie le 1er novembre 2009. Elle supporte 4302 jeux (8334 en comptant aussi les différents clones). Aujourd'hui, le projet est coordonné par Angelo Salese.

MAME est composé de différentes parties capables d'émuler complètement les architectures typiques des jeux d'arcade. Dans la pratique, il « reconstruit », via programmation, tous les circuits internes qui permettaient de gérer les manettes, le moniteur et les effets sonores. La seule chose manquante est la partie logicielle des machines, c'est-à-dire les ROM originales des jeux, qui pour des raisons de copyright ne peuvent pas être légalement distribuées.

D'un point de vue logique, MAME peut être divisé en 3 niveaux :

- le premier niveau gère l'émulation du matériel ;
- le second niveau contient toutes les fonctions générales et les modules qui agissent comme une passerelle entre le premier et le troisième niveau ;
- le troisième niveau présente l'interface de l'émulateur à l'utilisateur et se compose de l'interface graphique (si elle est présente) et de toutes les options qui permettent à l'émulateur d'être démarré et contrôlé.

Les deux premiers niveaux sont écrits en langage C afin d'avoir plus de rapidité et de portabilité. Aucune librairie spécifique externe n'est requise et le code peut être compilé par n'importe quelle plate-forme à l'aide d'un compilateur standard.

Le troisième niveau est dénommé « OSD » (Operating System Dependent) et contient tout le code spécifique aux différentes plate-formes.

Toutes les requêtes du système d'exploitation Windows ou Linux se trouvent dans cette couche, permettant ainsi une migration facile du code d'une plateforme à l'autre en modifiant uniquement ce niveau.

MAME utilise les trois premiers boutons du haut, tandis que Nintendo utilise les 6 premiers.

En fait, les boutons indiqués dans les instructions sont A, B, C, et correspondent sur la figure 5 respectivement aux boutons A, X et TR. Il y a ensuite le bouton « Start » (blanc) qui est en bas et « Select » (noir) qui est au-dessus de « Start », ces 2 boutons sont dupliqués pour jouer à deux joueurs.

En appuyant sur les deux, le système est réinitialisé et on quitte le jeu.

Pour tous les jeux, le bouton « **A** » permet la **sélection**, tandis que le bouton « **B** » permet de **quitter** le jeux (escape). La combinaison des boutons « Start » et « Select » permet de revenir au menu de l'émulateur dans lequel il se trouvait ou de revenir au menu de redémarrage du système.

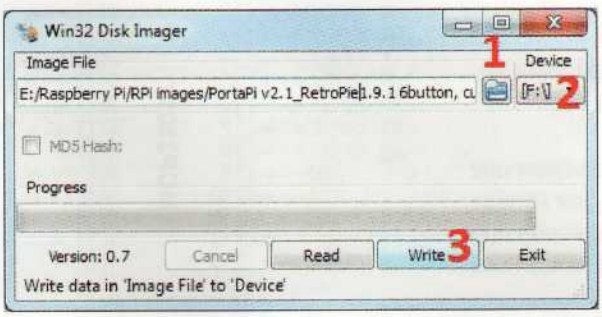


Figure 4 : création de la carte SD.

MAME, les ROM et les Copyrights

MAME est un logiciel téléchargeable qui peut être librement installé sur votre ordinateur, il est distribué librement.

Le code source est également gratuit. Par contre, il est illégal de vendre MAME ou son code source et de profiter de sa distribution. La règle stipule :

« La distribution de MAME avec des copies illégales de ROM sur le même support physique est strictement interdite.

Il est interdit de distribuer MAME de quelque manière que ce soit si vous vendez ou publiez des CD-ROM illégaux ou d'autres supports contenant des ROM. Ceci s'applique également si aucun revenu n'est tiré directement ou indirectement de cette vente.

Il est permis de rendre disponible des ROM et MAME sur le même site web, mais seulement en avertissant les utilisateurs du statut des droits d'auteur des ROM et en précisant que les utilisateurs n'ont pas à télécharger de ROM s'ils n'en détiennent pas les droits. »

Pour les ROM, le discours est différent. La possession de fichiers image des mémoires appartenant à des cartes logiques de jeux que vous ne possédez pas peut constituer une violation des droits d'auteur.

Dans la pratique, c'est comme faire une copie d'un jeu ou d'un CD de musique d'un ami au lieu de l'acheter.

En ce qui nous concerne, sur notre site, nous mettons des liens vers des ROM gratuites et fournissons des informations pour la recherche d'autres ROM libres de droit (autorisation des titulaires des droits).

Pour acquérir les droits afin d'utiliser une ou plusieurs ROM, vous pouvez demander la permission aux propriétaires légitimes, ou vous pouvez acheter des cartes logiques originales et utiliser légalement les ROM correspondantes. Une autre solution consiste à acheter un panneau de commande « HotRod Joystick » produit par Hanaho, avec lequel vous recevrez un ensemble de ROM Capcom absolument légales.

À partir du menu de l'émulateur, le bouton « B » permet de sortir (fonction ESC) et de revenir au menu des émulateurs disponibles. Vous pouvez les faires défiler à l'aide du joystick.

Pour effectuer les connexions entre la carte et les boutons, reportez-vous au **Tableau 1**. Les boutons des émulateurs utilisant « RetroArch » et les boutons de MAME sont déjà configurés.

À ce stade, il suffit de construire le boîtier, de vérifier le câblage de la carte et des différents boutons et joysticks, d'insérer les ROM des jeux et de commencer à jouer.

Dans notre fichier image, le monnayeur est activé, c'est-à-dire que le contact

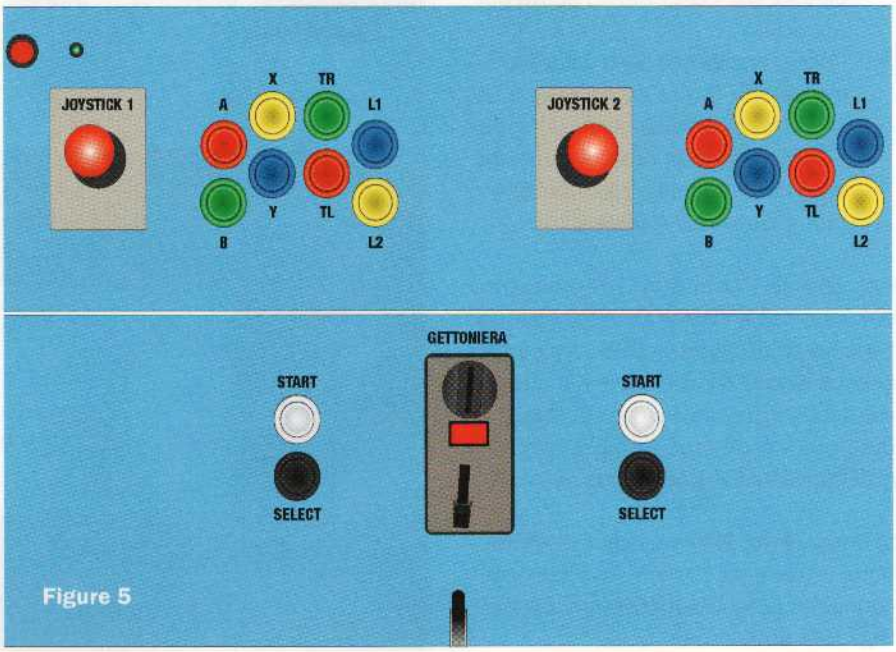


Figure 5

de ce dernier est connecté à l'entrée GPB7 de la carte.

Personnalisation de l'image

Si **vous ne souhaitez pas utiliser notre image** mais créer une **installation personnalisée**, par exemple pour changer le nombre de joueurs, les paramètres graphiques, etc., vous devez :

- **télécharger l'image de RetroPie** version 4.3 à l'adresse : **https://retropie.org.uk/download** ;
- **installer l'image sur la carte SD** comme décrit précédemment ;
- une fois l'image créée, **insérez la carte SD dans le RaspberryPi** puis connectez le clavier et le câble réseau (LAN). Après le démarrage, l'écran de la figure 6 apparaît, pour le moment il n'est pas possible de gérer une manette de jeu mais seulement le clavier ;
- **appuyez deux fois sur la touche F4** pour entrer dans l'invite de commande (prompt) et vérifiez l'adresse IP du RaspberryPi ;
- avec le clavier tapez la commande suivante qui vous permet d'entrer dans la **configuration avancée** : **sudo raspi-config** ;
- cliquez sur « **Expand file system** » pour augmenter l'espace de la carte SD (voir la figure 7). Sinon, il n'y aura pas assez d'espace pour installer les différents paquets et les ROM. Pour terminer l'opération, cliquez sur « OK » dans l'écran qui apparaît ;
- utilisez les flèches pour vous déplacer vers la droite de la fenêtre et choisissez « **Finish** » (voir la figure 8). Une nouvelle fenêtre apparaît vous demandant de redémarrer ou non. Cliquez sur « **Yes** ».

Une fois le RaspberryPi redémarré, déconnectez le clavier. En effet, lorsque les deux contrôleurs ont été configurés, si vous lancez RetroPie avec le clavier connecté, le logiciel ne reconnaîtra pas les boutons et vous serez obligé de redémarrer le RaspberryPi de nouveau.

Si vous devez effectuer des modifications à l'aide du clavier, vous devez le connecter uniquement après le démarrage de « Emulstation ».

Plus tard, cela nous aidera à configurer les boutons MAME.

À ce stade, vous devez **installer** le logiciel « **MobaXterm** » sur le PC. C'est un émulateur de terminal qui permet de contrôler à distance le RaspberryPi. Une fois que vous l'avez installé et ouvert, cliquez sur « session » en haut à gauche et dans la fenêtre qui apparaît, cliquez sur « **SSH** ».

Vous pouvez alors **entrer l'adresse IP** de votre RaspberryPi. Après l'avoir saisie, cliquez sur « OK » (voir la figure 9). Dans le champ à gauche de l'écran, cliquez sur l'adresse que vous venez de créer et entrez comme **utilisateur** « **pi** » et comme **mot de passe** « **raspberry** ».

À ce stade, vous pouvez exécuter des commandes pour installer et configurer les pilotes du joystick avec le MCP23017. Installez le noyau (kernel) Linux DKMS (Dynamic Kernel Module Support) à l'aide de la commande :

```
sudo apt-get install -y --force-yes dkms cpp-4.7 gcc-4.7 git joystick
```

Si une erreur apparaît, tapez la commande :

```
sudo apt-get install –f
```

Sinon continuez avec la commande :

```
wget http://www.niksula.hut.fi/~mhiienka/Rpi/linux-headers-rpi/linux-headers-`uname -r`_`uname -r`-2_armhf.deb
```

Tableau 1 : correspondance entre les broches GPIO de la carte et les boutons. La colonne RetroArch indique la valeur affectée dans le fichier de configuration RetroArch au bouton.

MCP21017 Broches (GPIO)	Nom du bouton	Équivalent à RetroArch
GPB0	A	8
GPB1	B	5
GPB2	TR	4
GPB3	Y	7
GPB4	X	3
GPB5	TL	6
GPB6	R1	//
GPB7	R2/SIG	1
GPA0	UP	trigger -1
GPA1	DOWN	trgger +1
GPA2	LEFT	trigger -0
GPA3	RIGHT	trigger +0
GPA4	START	11
GPA5	SELECT	10
GPA6	L1	0
GPA7	L2	9

À ce stade, un message vous demande si vous voulez télécharger et installer les pilotes audio (AP-MODE), appuyez sur « Y » (oui) pour exécuter. Ensuite, tapez les commandes suivantes :

```
sudo dpkg -i linux-headers-`uname -r`_`uname -r`-2_armhf.deb
sudo rm linux-headers-`uname -r`_`uname -r`-2_armhf.deb
```

Maintenant, vous devez récupérer le dossier « build » que vous trouverez

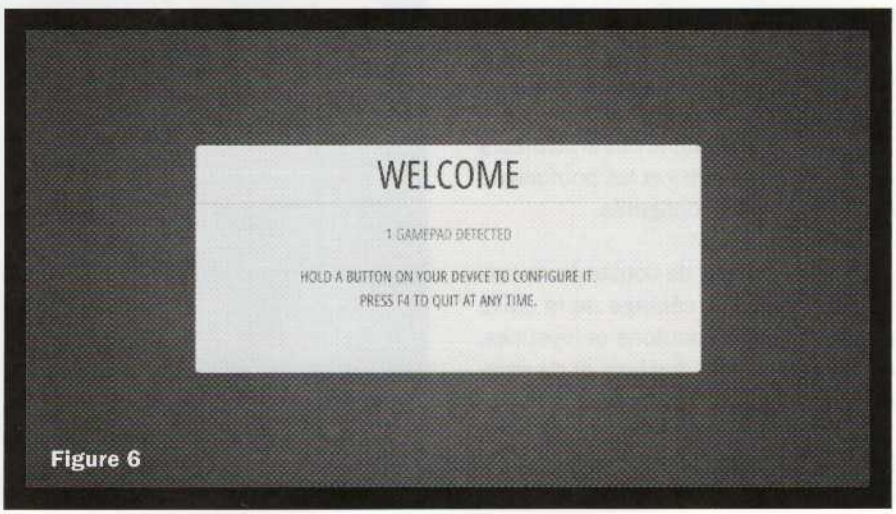


Figure 6

en téléchargement sur notre site web. Vous y trouverez les fichiers nécessaires au fonctionnement des joysticks.

Décompressez le dossier et copiez le via « MobaXterm » dans le répertoire : « **../home/pi** ».

Entrez dans le dossier « DEBIAN » à l'aide de la commande :

cd /home/pi/build/mk-arcade-joystick-rpi-0.1.4/DEBIAN/

Modifiez les droits d'administration de deux fichiers, sinon lors de l'installation, vous ne pourrez pas configurer les joysticks.Pour cela, tapez les commandes suivantes :

chmod 775 postinst
chmod 775 prerm

Revenez maintenant au dossier principal avec la commande « cd » puis tapez la commande suivante : **cd build**

Autorisez l'installation des pilotes suivants :

sed -i "s/\$MKVERSION/0.1.4/g" /home/pi/build/mk-arcade-joystick-rpi-0.1.4/usr/src/mk_arcade_joystick_rpi-0.1.4/* /home/pi/build/mk-arcade-joystick-rpi-0.1.4/DEBIAN/control /home/pi/build/mk-arcade-joystick-rpi-0.1.4/DEBIAN/prerm

Créez le fichier de gestion des deux contrôleurs :

sudo dpkg-deb --build "mk-arcade-joystick-rpi-0.1.4/"
sudo dpkg -i mk-arcade-joystick-rpi-0.1.4.deb

et créez les adresses des joysticks :

sudo modprobe mk_arcade_joystick_rpi map=0x20,0x21

Dans ce cas, nous avons pris un exemple avec deux contrôleurs, les adresses sont respectivement « 0x20 » et « 0x21 ».

L'adresse est déterminée par le DIP switch qui figure sur la carte MAME, conformément au tableau 2.

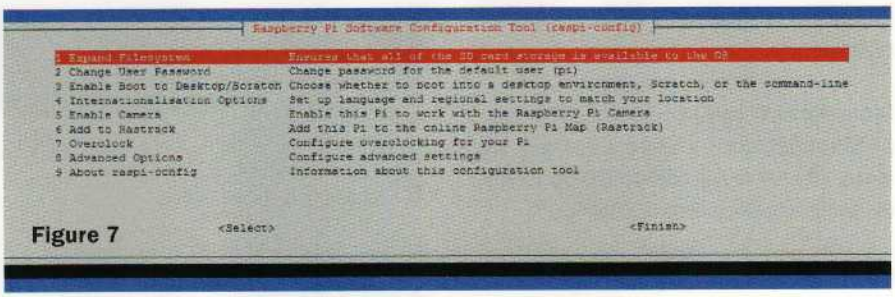


Figure 7

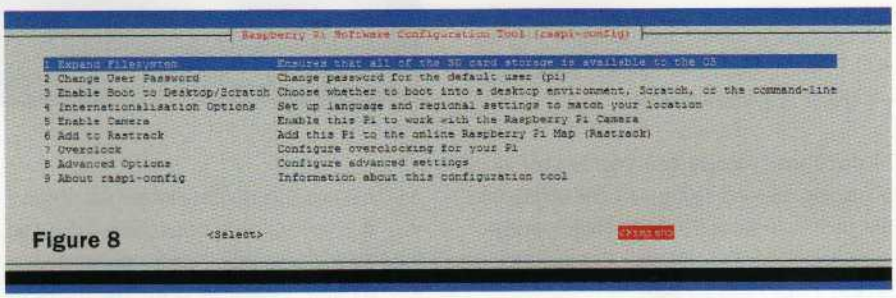


Figure 8

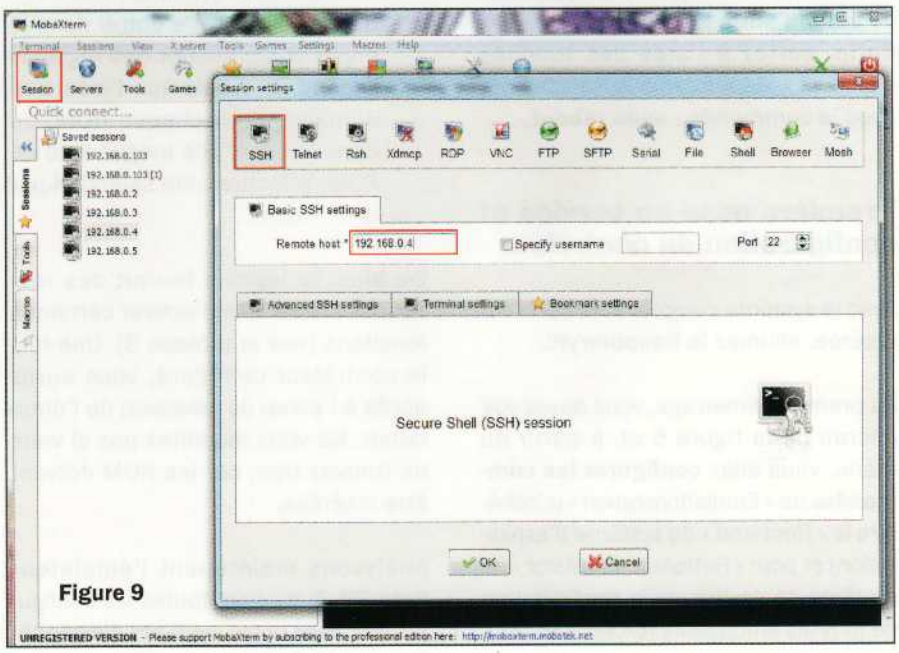


Figure 9

Ouvrez le fichier « modules » avec l'éditeur de texte à l'aide de la commande :

sudo nano /etc/modules

et insérez les lignes suivantes :

mk_arcade_joystick_rpi map=0x20,0x21
i2c-bcm2708
i2c-dev

Enregistrez les modifications en appuyant sur les touches « CTRL + O » en écrasant le fichier.

Tableau 2

DIP switch	A2	A1	A0	Adresse I2C
000	ON	ON	ON	0x20
001	ON	ON	OFF	0x21
010	ON	OFF	ON	0x22
011	ON	OFF	OFF	0x23
100	OFF	ON	ON	0x24
101	OFF	ON	OFF	0x25
110	OFF	OFF	ON	0x26
111	OFF	OFF	OFF	0x27

Ensuite, appuyez sur « CTRL + X » pour quitter.

Maintenant, ouvrez le fichier « raspi-blacklist.conf » avec l'éditeur de texte, et insérez un dièse (#) au début de la ligne « #i2c-bcm2708 » :

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
#i2c-bcm2708
```

Enregistrez à l'aide des touches « CTRL + O » en écrasant le fichier, puis avec « CTRL + X » pour quitter.

Maintenant, vous pouvez tester les boutons et vérifier que l'installation est correcte à l'aide de la commande :

```
jstest /dev/input/js0
```

Ensuite, modifiez « js0 » par « js1 » afin de tester l'autre joystick.
Enfin, sortez à l'aide des touches « CTRL+Z ». Redémarrez le RaspberryPi avec la commande : **sudo reboot**.

Première mise en service et configuration du contrôleur

Avec le système complet et la carte SD insérée, allumez le RaspberryPi.

Au premier démarrage, vous devez voir l'écran de la figure 6 et, à partir du menu, vous allez **configurer les commandes** de « Emulationstation » (c'est-à-dire le « front-end » du système d'exploitation) et pour « RetroArch Emulator », le système de gestion de la configuration de tous les émulateurs (excepté MAME).

Maintenez pressé n'importe quel bouton de la console jusqu'à ce que l'écran de la figure 10 apparaisse, puis le menu de la figure 11.

Lorsque vous aurez configuré les mouvements du joystick et les boutons « A-B-X-Y-LB-LT-RB-LB-START » et « SELECT », maintenez pressé un bouton déjà configuré. Le texte suivant apparaît : « hold for 2S to skip ». La configuration se termine et les autres commandes seront ignorées.

Dans cette phase, les fonctions des boutons sont :

Figure 10

- **A** : permet de confirmer un choix ;
- **B** : permet de quitter l'émulateur et de retourner à l'écran précédent ;
- **START** : permet d'entrer dans le menu pour modifier les boutons et redémarrer le système ;
- **SELECT** : permet d'entrer dans le menu et d'effectuer des réglages secondaires, tels que l'insertion d'images pour chaque ROM, ou dans le cas d'une longue liste de ROM, effectuer une sélection par la 1ère lettre.

De plus, le logiciel fournit des raccourcis clavier afin d'activer certaines fonctions (voir le tableau 3). Une fois le contrôleur configuré, vous aurez accès à l'écran de sélection de l'émulateur. Ne vous inquiétez pas si vous ne trouvez rien, car les ROM doivent être insérées.

Analysons maintenant l'émulateur RetroPie. Il contient toutes les configurations pour notre système d'exploitation (voir la figure 12).

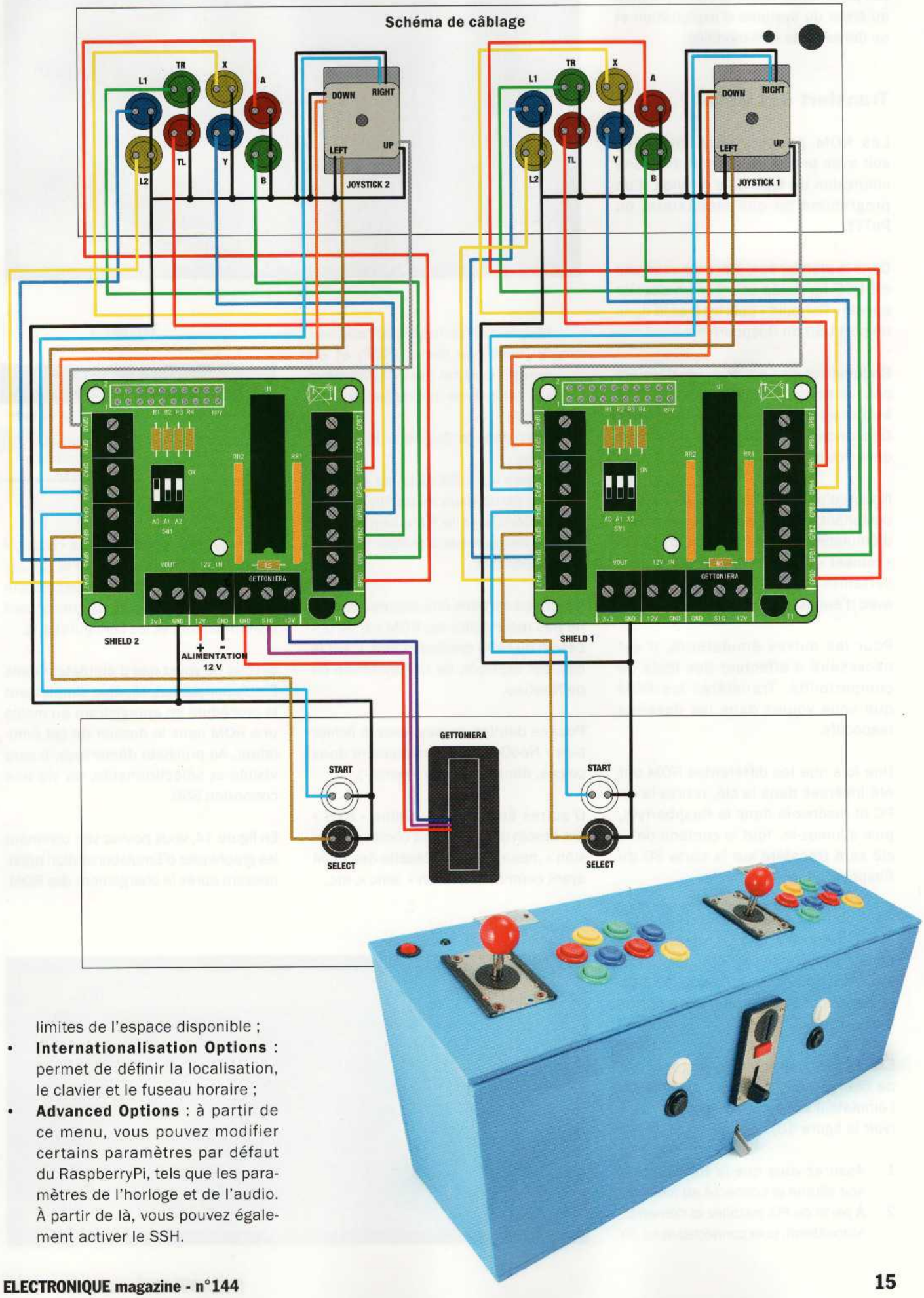
Nous vous recommandons d'utiliser un clavier pour effectuer les configurations, cela est plus pratique et rapide, sinon vous pouvez utiliser le contrôleur Arcade qui vient d'être installé.

Voici les différents types de configurations :

- **Configure audio setting** : configuration des paramètres audio, permet de sélectionner si l'audio doit être transmis via la sortie HDMI (par défaut) ou via la prise jack, et de régler le volume de sortie ;
- **Configure retroarch/launch retroarch rgui** : permet de modifier les paramètres des émulateurs, de l'audio, de la vidéo, du contrôleur etc. MAME n'est pas concerné, vous devrez modifier manuellement les paramètres ;
- **Configure retroarch/joystick** : réglages à modifier uniquement si vous utilisez pour jouer un clavier USB ou un joystick USB (contrôleur PS3, PC, XBOX, etc.) ;
- **Configure retroarch net play** : permet de modifier les paramètres réseau uniquement si le RaspberryPi est toujours connecté au réseau.
- **Configure splashscreen** : configuration de l'écran de démarrage (splash screen), permet de choisir parmi 30 thèmes l'image de démarrage du système ;
- **Configure wifi** : configuration du Wi-Fi ; grâce à son Wi-Fi intégré le RaspberryPi peut être connecté au réseau Wi-Fi en permanence. Choisissez la connexion souhaitée et les informations d'identification pour vous connecter ;
- **Show Ip addres** : visualisation de l'adresse IP du RaspberryPi ;
- **RaspberryPi configuration tool raspi-config** : configuration du RaspberryPi, permet d'accéder au menu de configuration du RaspberryPi, normalement effectué à l'aide de la commande : **sudo raspi-config**.

Vous trouverez les éléments suivants :

- **Expand Filesystem** : élargit la partition de la carte SD, dans les



limites de l'espace disponible ;

- **Internationalisation Options** : permet de définir la localisation, le clavier et le fuseau horaire ;
- **Advanced Options** : à partir de ce menu, vous pouvez modifier certains paramètres par défaut du RaspberryPi, tels que les paramètres de l'horloge et de l'audio. À partir de là, vous pouvez également activer le SSH.

Les paramètres restants correspondent au cœur du système d'exploitation et ne doivent pas être modifiés.

Transfert des ROM

Les ROM peuvent être chargées soit avec une clé USB, soit avec une connexion de type SSH à l'aide d'un programme tel que MobaXterm ou PuTTY.

Dans le premier cas, procurez-vous une clé USB formatée et créez un dossier appelé « Retropie » puis insérez-la dans un port USB du RaspberryPi.

Redémarrez ce dernier et les dossiers pour chaque émulateur seront créés automatiquement sur la clé USB. Débranchez la clé USB et insérez-la dans votre PC.

Notez qu'un dossier « roms » a été créé, contenant autant de sous-dossiers que d'émulateurs. Procurez-vous le **MAME** « **romset** » version « **037b5** », bien que certaines ROM puissent fonctionner avec d'autres « romset ».

Pour les autres émulateurs, il est nécessaire d'effectuer des tests de compatibilité. Transférez les ROM que vous voulez dans les dossiers respectifs.

Une fois que les différentes ROM ont été insérées dans la clé, retirez-la du PC et insérez-la dans le RaspberryPi, puis allumez-le. Tout le contenu de la clé sera transféré sur la carte SD du RaspberryPi.

Pour connaitre la fin du téléchargement, regardez la LED verte. Lorsqu'elle s'arrête de clignoter, le processus est terminé. Vous n'avez plus qu'à sélectionner l'émulateur et démarrer une partie.

Examinons maintenant la procédure de chargement des ROM à partir de l'émulateur de terminal via MobaXterm (voir la figure 13).

1. Assurez-vous que le RaspberryPi soit allumé et connecté au routeur ;
2. À partir du PC, installez et démarrez MobaXterm, puis connectez-vous au RaspberryPi en indiquant l'adresse IP (assignée dans DHCP) et en entrant comme nom d'utilisateur « pi » et comme mot de passe « raspberry » ;
3. Allez dans le dossier « RetroPie/roms » ;
4. Copiez les ROM dans les dossiers des émulateurs respectifs ;
5. Redémarrez le RaspberryPi afin de visualiser et d'utiliser les ROM disponibles.

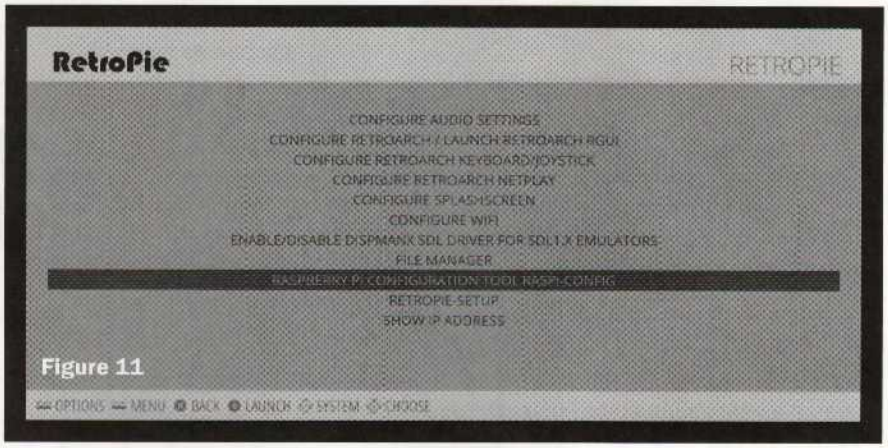


Figure 11

Notez que certains émulateurs peuvent ne pas reconnaître les ROM car ils ont besoin du BIOS correspondant. C'est le cas, par exemple, de la Playstation ou de NeoGeo.

Pour ce dernier, sauvegardez le fichier bios « NeoGeo.zip » (uniquement dans ce cas, dans le dossier « roms »).

D'autres émulateurs comme « NES » ont besoin de ROM ayant comme extension « .nes ». SNES nécessite des ROM ayant comme extension « .smc », etc.

Tableau 3

Raccourci clavier	Action
Select + Start	Sortie (ESC ou Exit)
Select + X	Menu RGUI
Select + B	Réinitialisation (Reset)

Vous trouverez à cette adresse « https://github.com/RetroPie/RetroPie-Setup/wiki » les BIOS dont vous avez besoin ainsi que des instructions concernant les émulateurs et les configurations.

Si vous ne voyez pas d'émulateur dans Emulationstation, répétez simplement la procédure en enregistrant au moins une ROM dans le dossier de cet émulateur. Au prochain démarrage, il sera visible et sélectionnable, ou via une connexion SSH.

En figure 14, vous pouvez voir comment les graphiques d'Emulationstation apparaissent après le chargement des ROM.

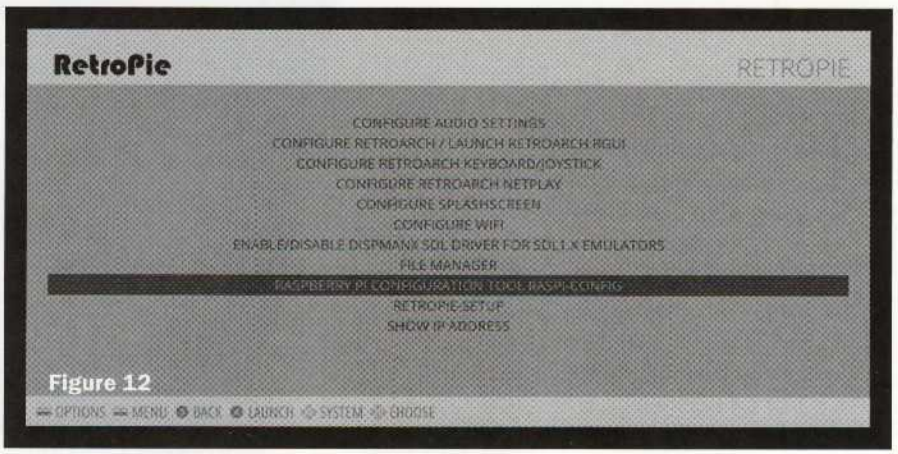


Figure 12

Installation du **monnayeur**

Le **monnayeur n'est pas indispensable**, mais nous voulions recréer l'ambiance des jeux d'arcade. Il ne fait que simuler la pression d'un bouton et envoyer au RaspberryPi un nombre d'impulsions correspondant à la valeur de la pièce insérée.

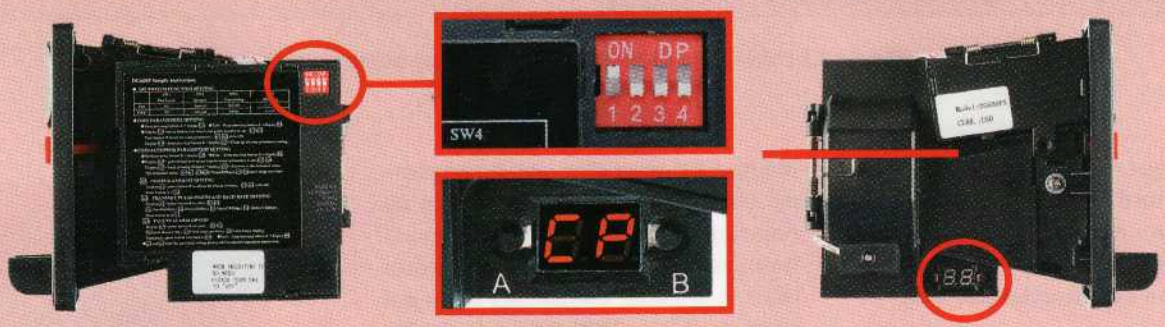
Dans ce projet, nous avons programmé 3 pièces : 50 centimes = 1 crédit, 1 euro = 2 crédits et 2 euros = 5 crédits.

Le monnayeur peut fonctionner avec une interface de type RS232 ou comme un simple contact (c'est comme cela que nous l'utilisons). Sur le côté gauche se trouvent deux boutons et un afficheur à 7 segments qui nous permettront de programmer les pièces à volonté. En haut se trouve un DIP switch qui permet de configurer le mode de fonctionnement du monnayeur. Dans notre cas, nous devons régler le « dip1 » sur ON et les autres sur OFF.

Une fois cela effectué, commençons par programmer la première pièce. Maintenez le bouton « A » enfoncé jusqu'à ce que « CP » apparaisse. Appuyez alors une fois sur le bouton « A » et vous verrez « C1 » apparaître (cela signifie « pièce 1 » ou première pièce).

À ce stade, appuyez sur le bouton « B », vous verrez le nombre augmenter. Cela équivaut au nombre d'impulsions que le monnayeur enverra à chaque fois que nous insérerons cette pièce. Mettez « 01 » et insérez une pièce de 50 centimes. Vous entendrez des « bips » signifiant que la valeur de la pièce a été acceptée. Dorénavant, à chaque fois que vous insérerez une pièce de 50 centimes, le monnayeur émettra une impulsion, c'est-à-dire l'équivalent dans le jeu d'un crédit. Répétez la procédure pour les autres pièces :

C1 : 01 équivaut à un crédit (50 centimes)
C2 : 02 équivaut à deux crédits (1 euro)
C3 : 05 équivaut à cinq crédits (2 euros)



Avec le joystick, vous vous déplacez vers la droite et vers la gauche, tandis qu'avec la touche « A » vous entrez dans l'émulateur. Sélectionnez un jeu et lancez-le avec la touche « A ».

Le bouton « B » est utilisé par le « front-end » Emulationstation, pour quitter le menu d'un émulateur et revenir à l'écran de sélection (écran principal).

Le site web **www.mamedev.org permet de télécharger gratuitement des ROM de manière légale**, car les détenteurs des droits d'auteur ont donné leur accord ou les droits d'auteur ont expiré.

Par exemple, pour tester l'émulateur, téléchargez la ROM de STAR FIRE en vous connectant à : **http://mamedev.org/roms/**.

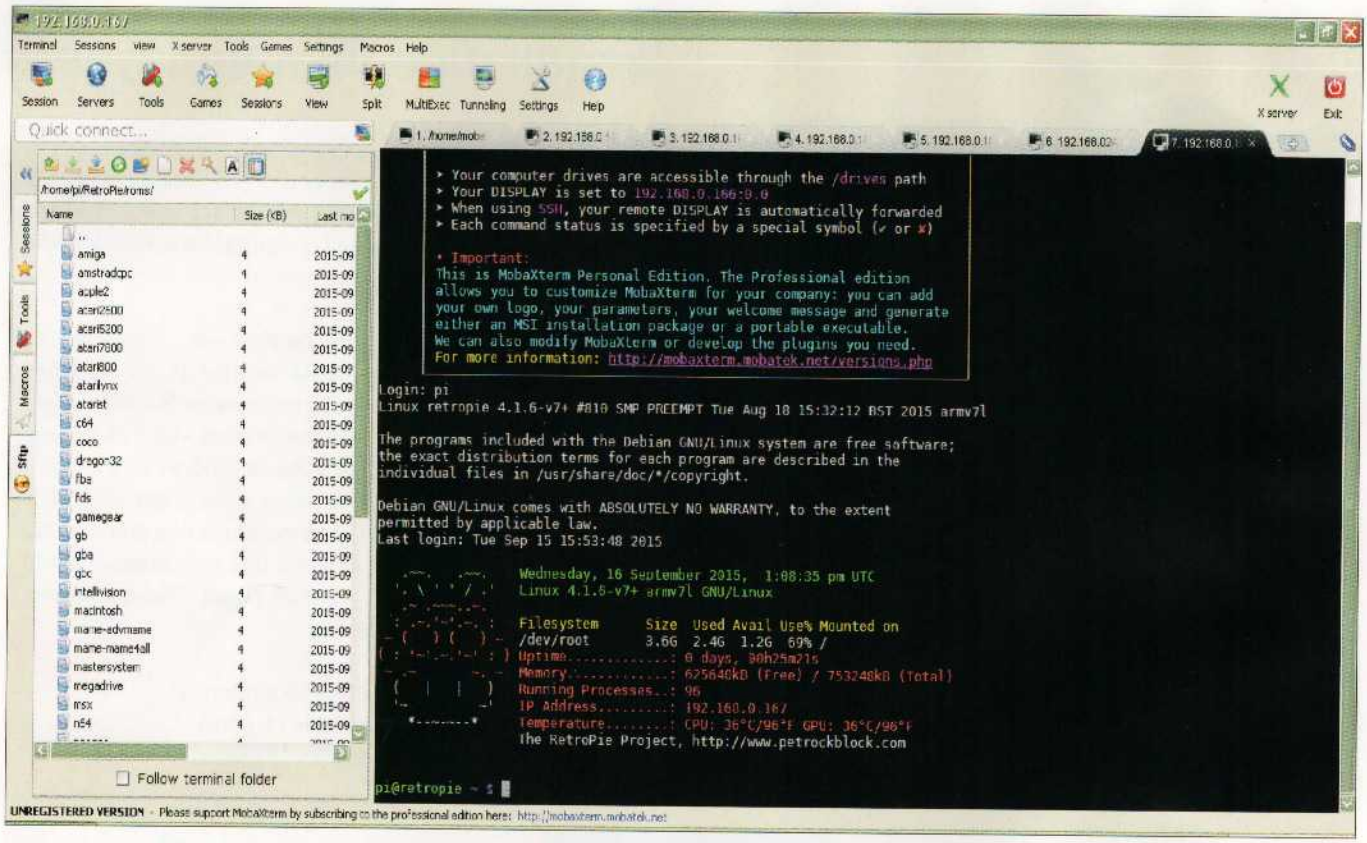


Figure 13 : chargement de la ROM à l'aide de MobaXterm.

Téléchargez en cliquant sur le lien « **Download the Star Fire ROM images** » après avoir coché l'option « **I understand that these ROM images are for non-commercial use only** ».

Maintenant, chargez la ROM (starfire.zip) dans l'émulateur MAME en la copiant dans le dossier « mame-mame4all » via MobaXterm ou une clé USB, puis redémarrez le RaspberryPi.

À partir de Emulationstation, sélectionnez le MAME et avec le joystick cherchez « Star Fire », puis démarrez le jeu en appuyant sur la touche A.

Maintenant, choisissez les touches à utiliser :

1. Appuyez sur la touche « TAB » pour entrer dans le menu de configuration ;
2. Aller à INPUT (Général) ;
3. Affectez aux touches les fonctions souhaitées et définissez une touche pour terminer le jeu en modifiant l'interface utilisateur actuelle, puis quittez le menu de configuration.

Maintenant vous êtes prêt à commencer votre jeu vidéo. Rappelons que de nombreux sites proposent un grand nombre de ROM, mais pour rester dans la légalité, **choisissez ceux qui proposent des ROM libres de droits**. Vous pouvez voir sur note site le jeu vidéo d'arcade en action.

Vous trouverez les plans du coffret aux formats « .dxf » et « .pdf » à réaliser, ainsi que l'**image** de **RetroPie** en téléchargement sur notre site **www.electroniquemagazine.com** dans le sommaire détaillé de la revue (allez vers le bas de la page dans la section « Télécharger »).

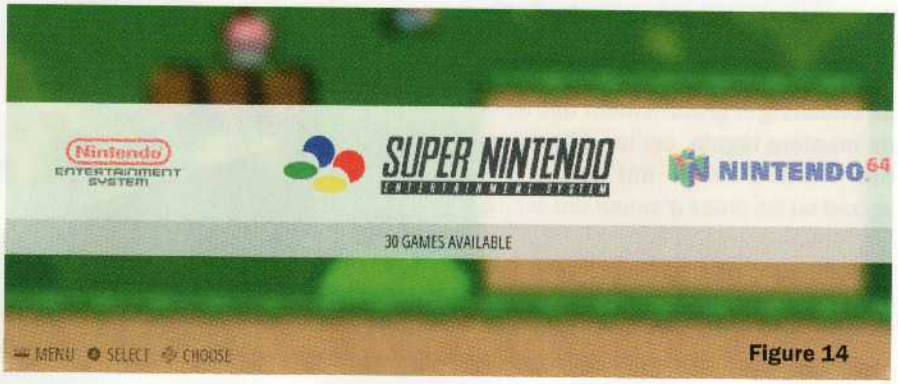


Figure 14

DEMOBOARD MP3

Dans cet article, nous testons l'énorme potentiel du module lecteur audio DFR0299, idéal pour Arduino mais aussi pour de nombreuses applications autonomes.

de **Boris Landoni**

Il existe de nombreuses applications fixes et mobiles dans lesquelles nous avons besoin de reproduire des messages vocaux. Par exemple, une boîte installée à l'entrée d'un magasin et dotée d'un capteur de proximité qui attire l'attention des clients grâce à un message vocal, ou des balances parlantes qui vous invitent à vous peser, ou des panneaux d'avertissement qui indiquent le chemin d'accès de l'entrée d'un local ou encore une voix dans une auto qui vous indique de boucler la ceinture et de rouler à vitesse modérée lorsque vous dépassez la limite autorisée.

Dans le passé, le seul moyen d'obtenir ces fonctions étaient d'utiliser des disques miniatures ou des bandes magnétiques.Cependant, grâce à la création de synthétiseurs vocaux (Speech Circuit) comme le premier MSM6322 du fabricant OKI, il était alors possible de reproduire un son à partir de semi-conducteurs.

L'arrivée des Chipcorders ISD (maintenant Nuvoton ...) a développé l'ère des enregistreurs et des reproducteurs vocaux numériques.

Cependant, le secteur a rapidement évolué impliquant l'utilisation de supports de stockage de masse capables de surmonter les contraintes imposées par la faible quantité de mémoire des ISD.

Voici donc le module **DFR0299** qui est un **lecteur MP3**. Pour vous montrer tout le potentiel de ce module, nous avons décidé de concevoir et de vous proposer dans cet article une carte de développement spécifique pour ce module. Elle sera donc la plate-forme de développement pour différentes applications ou simplement pour programmer plusieurs dispositifs.

Le circuit

Avant de procéder à l'analyse du schéma électrique de la carte demoboard MP3, considérons qu'il est utile de résumer les caractéristiques du module MP3, détaillées par ailleurs dans l'encadré intitulé « Caractéristiques techniques ».

Le module est capable de **reproduire** directement des **fichiers MP3** et **WAV** stockés sur une carte SD dont la **capacité maximale est de 32 Go**. Cette dernière doit obligatoirement être **formatée** dans un système de fichier **FAT16** ou **FAT32**, et insérée dans l'emplacement approprié.

Cependant, le module dispose également d'une **interface USB** de type « **Device** », que nous avons reportée sur un connecteur USB de type « A » et qui lui permet de **lire des données à partir d'une clé USB**, à condition que la capacité maximale n'excède pas 32 Go et qu'elle soit formatée comme déjà mentionné pour la SD-Card.

La particularité de ce module est qu'il a été **conçu et construit pour l'environnement Arduino**. Pour pouvoir le gérer, une librairie spécifique a été développée permettant de sélectionner une piste à jouer, de régler le volume, etc. Mais le **module** peut également **fonctionner de manière autonome**.

C'est la manière que nous avons adoptée dans notre cas, nous l'utiliserons de cette façon grâce à un nombre imposant de boutons qui permettent de contrôler toutes les fonctions.

Comme le **module est contrôlable via une interface série** à un niveau TTL, nous utilisons dans notre demoboard un convertisseur série/USB pour l'interfacer avec un ordinateur. Le port série est disponible et accessible au niveau du circuit.

Le module contient un décodeur capable de décompresser l'audio au format MP3 et un microcontrôleur capable d'accéder, via le bus SPI, aux données contenues dans une carte SD. Au fur et à mesure que le flux de données est lu, le décodeur le transforme en signal audio non compressé.

Celui-ci est ensuite amplifié par une **sortie BF mono intégrée** d'une puissance de 3 W, ce qui est plus que suffisant pour piloter un haut-parleur afin de tester les différentes possibilités du module. Le HP se connecte entre la broche 8 (SPK1 → +HP) et la broche 6 (SPK2 → -HP).

Si vous avez besoin de plus de puissance, utilisez les **sorties audio stéréo** « **DAC_R** » (broche 4) et « **DAC_L** » (broche 5) pour piloter un amplificateur BF de puissance.

N'oubliez pas de relier chaque sortie avec une référence à la masse (GND).

Toutes les **fonctions du module sont gérées par des boutons poussoirs** disposés sur le circuit imprimé de la carte, comme indiqué dans le schéma électrique.

Comme vous pouvez le voir, étant donné qu'il s'agit d'une carte de développement, tous les contacts sont reportés vers l'extérieur, y compris les lignes « TX » et « RX » de l'UART interne du module.

Cette dernière fonctionne à 9600 bps car le module a été développé pour Arduino, il n'est donc pas nécessaire d'avoir une vitesse de transmission série élevée.

Pour détecter l'état de plusieurs boutons, un stratagème a été adopté. En effet, nous utilisons le convertisseur A/N du microcontrôleur qui est accessible via les broches ADKEY1(12) et ADKEY2(13). Cette solution est intéressante, sinon les détections des états des 20 boutons nécessiteraient autant de broches du microcontrôleur.

Le module ne dispose pas d'autant de broches, il est contenu dans un circuit imprimé ayant 16 broches au pas de 2,54 mm réparties de chaque côté. Dans celles-ci doivent passer l'alimentation, les sorties audio stéréo, la connexion USB, le port série, et la sortie haut-parleur.

L'état de chaque bouton poussoir est détecté par la chute de tension produite lors de la pression, il est ainsi possible de déterminer l'entrée à laquelle il est connecté. Cette tension dépend de la résistance insérée en série avec chaque bouton.

Pour être précis, chacune des entrées **ADKEY1** et **ADKEY2** est dotée en interne d'une résistance de pull-up qui en série avec chaque résistance de chaque bouton poussoir constitue un **diviseur de tension**. En choisissant la valeur de chaque résistance de

CARACTÉRISTIQUES TECHNIQUES

- Tension d'alimentation : de **8 à 12 VDC** ;
- Consommation maximale de courant : **500 mA** ;
- Commande à l'aide de **boutons poussoirs** ;
- Formats de fichier pris en charge : **MP3 et WAV** ;
- Morceaux reproductibles : **14** ;
- Source de données : microSD et USB **≤ 32 Go** ;
- Sélection automatique ou manuelle des données ;
- Sortie pour haut-parleur mono **3 W / 8 Ω** ;
- Sortie stéréo pour amplificateur BF externe.

manière adéquate, lorsqu'un bouton est pressé il **provoque une certaine chute de tension unique** qui permet de **l'identifier**.

Dans le programme se trouve une table contenant les valeurs des tensions correspondantes à chaque résistance des boutons. Cela permet donc la lecture de l'état de chaque bouton car les résistances en série avec les 10 boutons ont des valeurs différentes.

Par exemple, une pression sur la touche « Play Mode » applique une tension sur ADKEY1 qui est différente de la touche « Loop » ou encore « Pause/Play ». Chaque bouton correspond à une tension unique. Le principe est identique pour l'entrée ADKEY2.

Il est évident que **chaque touche doit être pressée une à la fois**, sinon il y aurait deux résistances en parallèle et le convertisseur A/N lirait une valeur anormale qui n'existe pas dans la table des valeurs des tensions.

Les touches sont divisées en groupes fonctionnels, celles qui activent la reproduction directe des pistes (dans ce cas, la commande s'effectue via les touches qui permettent la lecture de 14 fichiers audio) et celles qui contrôlent le mode de fonctionnement.

Les touches 01 à 14, lorsqu'elles sont pressées (une à la fois), effectuent automatiquement la reproduction des pistes dans l'ordre correspondant à celui du support de stockage. Ainsi, une pression sur la touche 01 fait jouer directement la première piste, sur 02 la seconde piste, et ainsi de suite jusqu'à la 14ème piste.

Il est à noter que les 14 touches liées aux segments de la mémoire dans lesquels sont stockées les mêmes pistes ont deux modes d'activation. Une pression courte commence la lecture tandis qu'une pression prolongée provoque la lecture de la piste en boucle.

La touche « **Play Mode** permet de déterminer le **mode de lecture**, c'est-à-dire est-ce que la chanson en cours doit être lue jusqu'à la fin ou faut-il passer immédiatement à la piste sélectionnée

schéma **ÉLECTRIQUE**

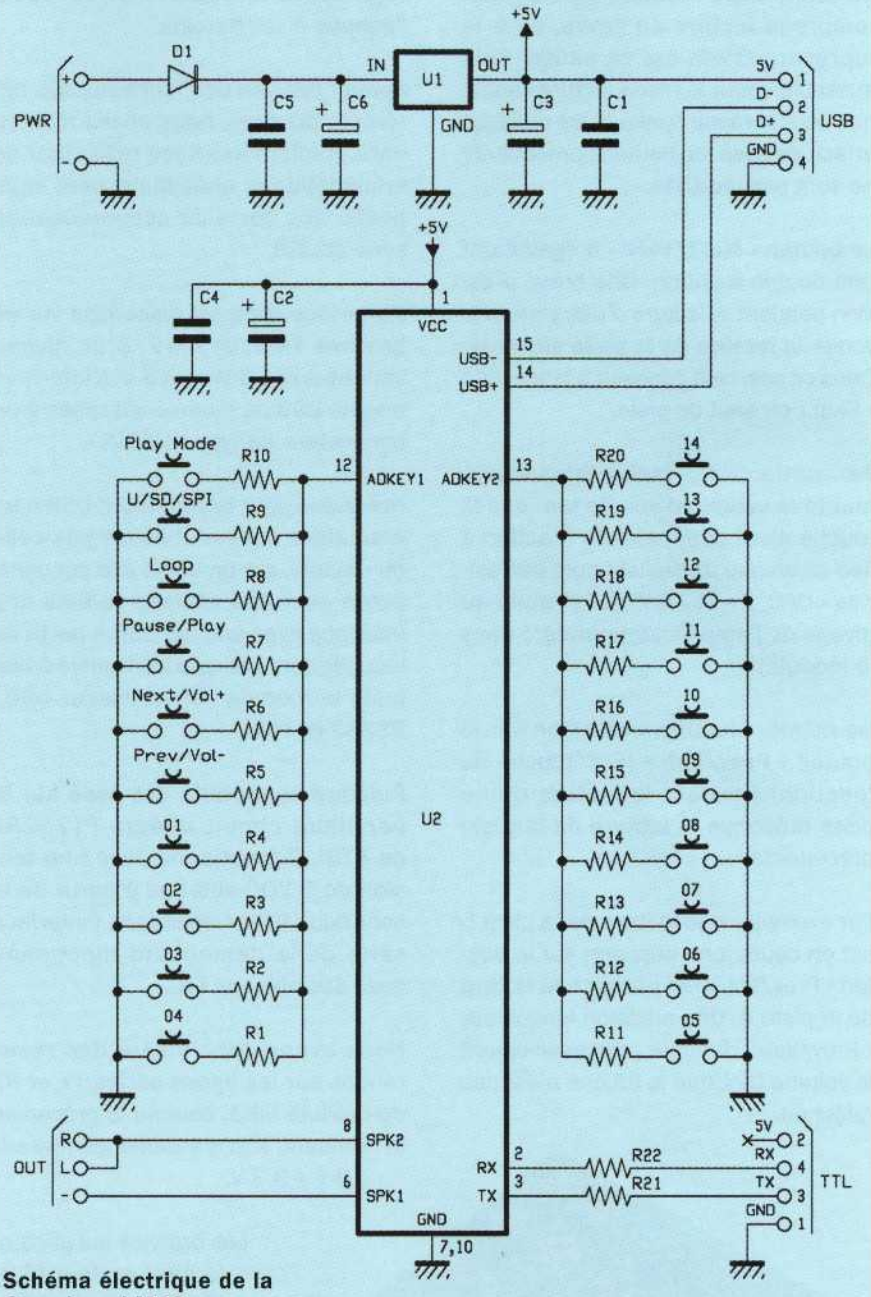


Schéma électrique de la Demoboard MP3.

(mode interruption/non interrompu) ? » Chaque fois que la touche est pressée, le mode est inversé.

La touche « **U/SD/SPI** » permet de définir manuellement la **sélection de la source** à partir de laquelle les fichiers doivent être lus. Chaque pression fait passer dans l'ordre de la source USB à la carte SD puis au bus SPI. La troisième pression active le mode veille, puis le cycle recommence.

La troisième touche de fonction « **Loop** » permet de basculer entre les modes de **lecture simple ou en boucle**.

Dans le premier cas, la piste choisie avec les touches 01 à 14 n'est reproduite qu'une fois à chaque pression, alors que dans le second mode (cyclique) la piste recommence du début de manière infinie tant que l'alimentation du circuit est présente, ou tant que la touche « Loop » n'est pas pressée de

nouveau ou encore tant que la touche « Pause/Play » n'est pas pressée.

En effet, cette dernière sert à interrompre la lecture en cours, ou à la reprendre si elle est en pause. Cela aussi bien pour le mode lecture simple que pour le mode cyclique, les réglages effectués avec les boutons précédents ne sont pas modifiés.

Le bouton « **Next/Vol+** » a également une double fonction. Une brève pression pendant la lecture d'une piste provoque la **lecture de la piste suivante**. Dans ce cas, cela équivaut à la fonction « Skip » ou saut de piste.

Par contre, une **pression longue augmente le volume** d'écoute tant que la touche n'est pas relâchée (l'action a lieu au niveau du signal audio des sorties « DAC_R » et « DAC_L » et aussi au niveau de l'amplificateur intégré dans le module).

De même, une brève pression sur le bouton « **Prev/Vol-** » (5ème touche de fonction) pendant la lecture d'une piste provoque la lecture de la piste précédente.

Par exemple, si la lecture de la piste 5 est en cours, une pression sur le bouton « Prev/Vol- » fait passer à la lecture de la piste 4. Une pression longue sur « Prev/Vol- » diminue progressivement le volume tant que la touche n'est pas relâchée.

En ce qui concerne le volume d'écoute, notez qu'au démarrage le volume est au niveau le plus élevé, il faudra donc l'ajuster avec la touche « Prev/Vol- » pour l'adapter à vos besoins.

Après l'analyse des fonctions des différents boutons, nous allons maintenant étudier l'interface utilisateur de la demoboard, essentiellement composée des ports de communication série et USB.

L'interface série est disponible via les broches TX(3) et RX(2) à un niveau variant entre 0 V et 3,3 V. L'interface interne USB du module est reliée à un connecteur de type « USB-A ».

Notez que pour la connexion USB à un ordinateur nous n'utilisons pas celle du module, car en usine elle est configurée pour être utilisée comme une interface avec une clé USB à partir de laquelle sont chargés les fichiers à lire, mais le module convertisseur USB/RS232 **ET782**.

Puisque ce module est basé sur le populaire circuit intégré FT232RL de FTDI, il fonctionne avec une tension de 5 VDC obtenue à partir de la connexion USB. Cependant, l'interface série de la demoboard fonctionne avec des niveaux TTL.

Nous avons donc inséré des résistances sur les lignes séries TX et RX du module MP3, comme le préconise le fabricant, afin d'adapter les niveaux à 0 V / 3,3 V.

Les broches qui pilotent le haut-parleur (6 et 8) sont disponibles sur une prise jack de 3,5 mm, le support de la carte SD est directement soudé au circuit imprimé du module (le connecteur n'apparait pas dans le schéma électrique).

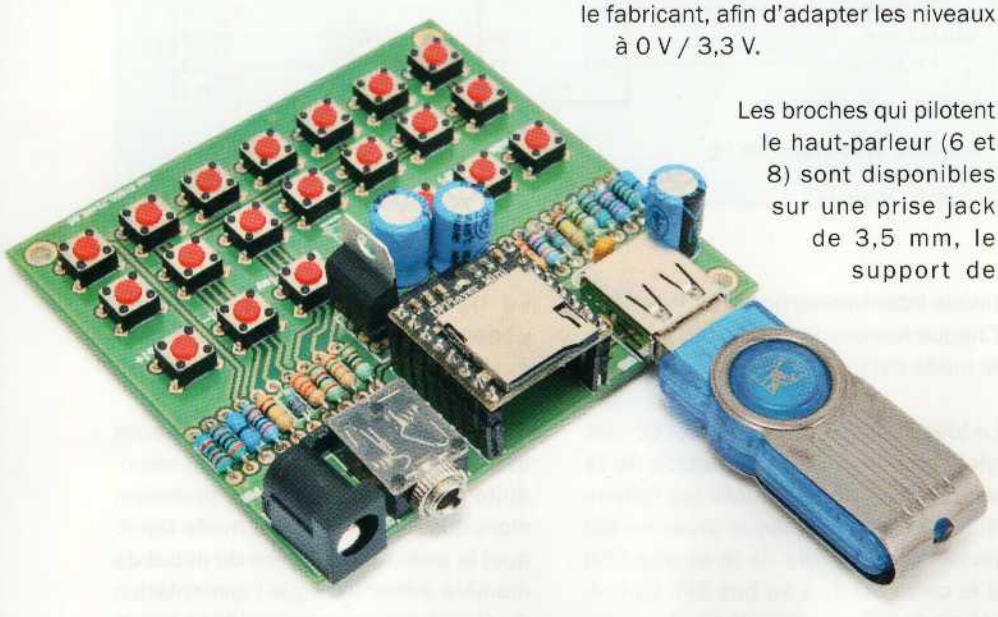


Figure 1 : lecture des fichiers à partir d'une clé USB insérée dans son connecteur.

L'ensemble du circuit est alimenté par une tension continue d'une valeur comprise entre **9 VDC** et **12 VDC**. Le régulateur U1 (7805) permet d'obtenir une tension stabilisée de 5 VDC.

La diode **D1 protège** la demoboard contre toute **inversion de polarité**. Les condensateurs C5 et C6 filtrent la tension d'entrée (au niveau des points + PWR et -PWR), tandis que la tension stabilisée en sortie du régulateur U1 (5 VDC) est filtrée par les condensateurs C1 et C3.

Les condensateurs C2 et C4 filtrent localement la tension arrivant au module, empêchant ainsi les perturbations provenant de l'alimentation de parasiter le fonctionnement du microcontrôleur du module MP3.

Réalisation pratique

Nous pouvons maintenant passer à la construction de la carte Demoboard MP3. Elle nécessite un circuit imprimé double face, dont vous pouvez télécharger sur notre site dans le sommaire détaillé de la revue les typons et les fichiers Gerber pour une fabrication professionnelle.

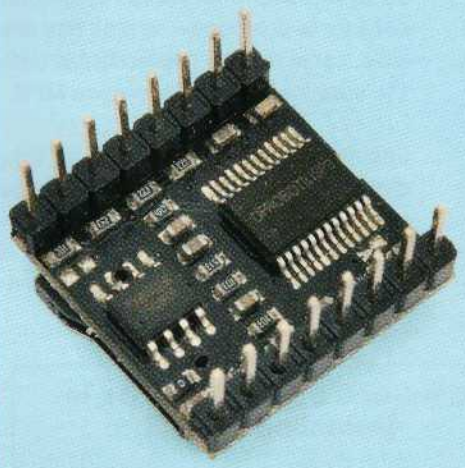
Le hobbyiste pourra réaliser le circuit imprimé par photogravure en utilisant les typons au format Acrobat (pdf) à l'échelle 1.

Notez que pour plus de facilité tous les composants sont traversants (traditionnels), excepté le module qui est en CMS mais qui est vendu déjà monter. Il n'y aura donc aucune difficulté pour réaliser la carte.

Une fois le circuit imprimé gravé et percé, commencez par souder les composants ayant un profil bas, c'est à dire les résistances, la diode D1 en orientant correctement sa bague.

Soudez ensuite les barrettes femelles de 8 contacts chacune qui accueilleront le module MP3.

Le module MP3 audio



Le module MP3 DFPlayer vu de dessous.

Le composant que nous utilisons dans la demoboard est un lecteur MP3 basé sur un circuit spécialisé DFR0299 du fabricant DFROBOT (www.dfrobot.com).Il permet de lire des fichiers audio au format MP3 ou WAV sans devoir utiliser de matériel coûteux et exigeant. En fait, le DFR0299 est un décodeur complet qui lit, à partir d'un support de stockage de masse comme une carte SD ou une clé USB, et décrypte un fichier audio. Ensuite, il l'amplifie pour l'envoyer vers un haut-parleur (8 Ω / 3W) ou vers un amplificateur externe. Il est donc idéal pour la lecture de fichiers audio standard MP3, mais il ne dispose pas de ressources de calcul (CODEC compressés)) dont le traitement direct nécessiterait l'ajout d'un microcontrôleur.

Le module a été conçu pour être géré par Arduino, celui-ci est nécessaire pour l'envoi de commandes via l'interface série pour gérer la lecture. Tout le reste est effectué par le DFR0299. Ce module peut également être utilisé de manière autonome. Il doit être alimenté en 5 VDC, sa consommation est de 450 mA dans le cas où la puissance est délivrée dans un haut-parleur de 3 W / 8 Ω. Il est nécessaire de lui adjoindre quelques boutons comme dans le cas de la demoboard. Il peut aussi être utilisé en combinaison avec un microcontrôleur équipé d'un port série TTL. Il est possible de le contrôler via les broches d'entrées séries ou analogiques.

Il peut gérer jusqu'à 100 dossiers, chaque dossier pouvant contenir jusqu'à 1 000 pistes MP3. Le réglage du volume, qu'il soit commandé par le port série ou via les boutons sur la carte, s'effectue sur 30 niveaux (± à partir de la position centrale).

L'alimentation peut varier de 3,2 VDC au minimum jusqu'à 5 VDC au maximum, le courant consommé en veille est de l'ordre de 20 mA. Les principales caractéristiques du module sont :

- décodage MP3 11172-3 et I6013813-3 layer3 ;
- possibilité d'égalisation : Pop, Rock, Jazz, Classique etc. ;
- fréquences d'échantillonnage en lecture : 8/11,025/12/16/22,05/24/32/44,1/48 kHz ;
- DAC de 24 bits qui permet une plage dynamique de 90 dB et un rapport signal sur bruit de 85 dB ;
- prise en charge des systèmes de fichiers FAT16 et FAT32 ;
- capacité maximale de la mémoire adressable : 32 Go clé USB ou carte SD ;
- dimensions : 20 mm x 20 mm x 13 mm.

Bien que dans notre projet nous utilisions le module en mono, il dispose d'un décodeur MP3 stéréo dont les sorties sont disponibles sur les broches 4 et 5 qui sont au niveau ligne (bas niveau et haute impédance), afin de piloter un amplificateur, une table de mixage ou encore des écouteurs de 300 Ω.

Numéro	Nom	Description	Note
1	VCC	Tension d'entrée	3,2 VDC à 5 VDC (4,2 V typique)
2	RX	Entrée série UART	
3	TX	Sortie série UART	
4	DAC_R	Sortie audio droite	sortie bas niveau canal droit
5	DAC_L	Sortie audio gauche	sortie bas niveau canal gauche
6	SPK2	Haut-parleur	sortie en pont BF (vers le - HP)
7	GND	Ground	Masse
8	SPK1	Haut-parleur	sortie en pont BF (vers le + HP)
9	IO1	Trigger port 1	commande précédente/volume vers le bas
10	GND	Ground	Masse
11	IO2	Trigger port 2	commande suivante/volume vers le haut
12	ADKEY1	AD port 1	lecture de la 1ère piste
13	ADKEY2	AD port 2	lecture de la 5ème piste
14	USB+	USB+ DP	USB Data -
15	USB-	USB- DM	USB Data +
16	Busy	Statuts de lecture	niveau bas pendant la lecture et haut au repos

Brochage du module MP3.

L'amplificateur BF intégré dans le module est de type monophonique, il amplifie le mélange des canaux gauche et droit. L'étage final est basé sur le circuit intégré YX8002A de Thaieasyelec, Il est capable de délivrer 3 W en continu avec une distorsion de moins de 10 % grâce à une structure finale en pont.

Les lignes séries fonctionnent à des niveaux de 0 V /3,3 V, mais elles peuvent être interfacées avec une logique en 5 V. Dans ce cas, le fabricant suggère de placer une résistance de 1 kΩ en série sur la ligne RX, de manière à limiter le courant d'entrée sur cette broche à des valeurs sûres.

Ces barrettes forment une sorte de support sur mesure pour le module DFR0299.

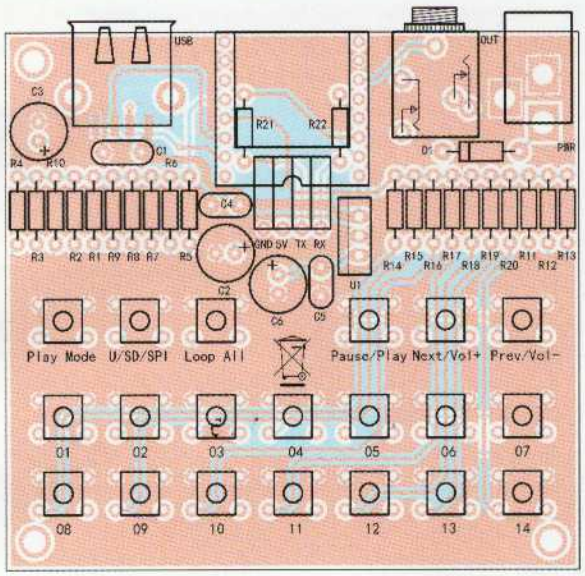
Continuez en soudant les condensateurs non polarisés puis les boutons poussoirs. Ensuite, passez aux condensateurs électrolytiques en prêtant une attention particulière à leur polarité.

Pour cela aidez-vous du plan de câblage des composants. Soudez ensuite le régulateur en l'insérant verticalement. Sa partie métallique (méplat) doit faire face au condensateur C5.

Continuez avec le connecteur femelle à 4 pôles coudé à 90°, qui doit être soudé sous le module MP3 très proche du circuit imprimé. Nous recommandons de le positionner le plus bas possible, car le module convertisseur USB/TTL sera inséré sous le module MP3.

Ensuite, soudez la prise USB, la fiche d'alimentation dont le point central est le positif et la prise jack de 3,5 mm pour la sortie audio. Enfin, montez le module MP3 sur les deux rangées de

Plan de montage de la Demoboard MP3



Plan de câblage des composants de la Demoboard MP3.

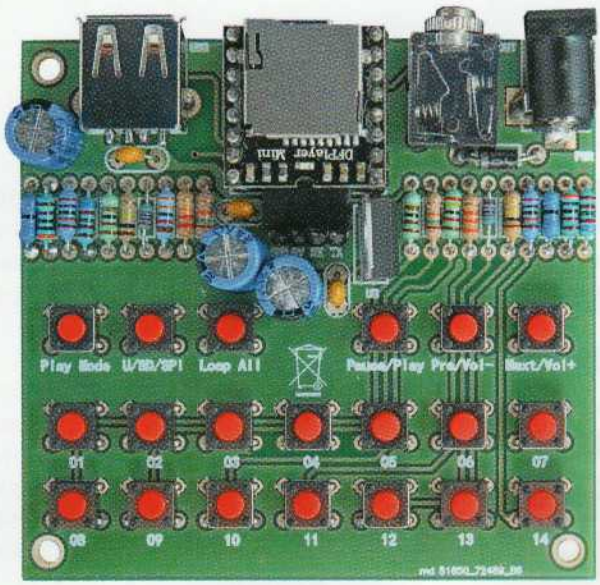


Photo de l'un de nos prototypes de la Demoboard MP3.

Liste des composants de la Demoboard MP3

R1..... 9,1 kΩ 1%
R2..... 6,2 kΩ 1%
R3..... 3 kΩ 1%
R4..... 0 Ω
R5..... 15 kΩ 1%
R6..... 24 kΩ 1%
R7..... 33 kΩ 1%
R8..... 51 kΩ 1%
R9..... 100 kΩ 1%
R10.... 200 kΩ 1%
R11 ... 0 Ω
R12 ... 3 kΩ 1%
R13 ... 6,2 kΩ 1%
R14.... 9,1 kΩ 1%
R15 ... 15 kΩ 1%
R16.... 24 kΩ 1%
R17.... 33 kΩ 1%
R18 ... 51 kΩ 1%
R19 ... 100 kΩ 1%
R20 ... 200 kΩ 1%
R21.... 1 kΩ
R22 ... 1 kΩ
C1..... 100 nF céramique
C2..... 220 µF / 16 V électrolytique
C3..... 220 µF / 16 V électrolytique
C4..... 100 nF céramique
C5..... 100 nF céramique
C6..... 220 µF / 16 V électrolytique
D1..... 1N4007
U1..... 7805
U2..... DFR0299
USB ... connecteur USB-A pour circuit imprimé
OUT.... prise jack stereo 3,5mm pour circuit imprimé
TTL..... module USB/TTL (ET782)

Divers

fiche d'alimentation pour circuit imprimé
microswitch (x 20)
barrette femelle 8 pôles (x 2)
barrette femelle 4 pôles coudée à 90°

NB : les typons des circuits imprimés à l'échelle 1 sont disponibles en téléchargement dans le sommaire détaillé de la revue.

L'interface vocale
Ford de 3ème génération

La dernière technologie de reconnaissance et de synthèse vocale de Ford se nomme « SYNC3 ». Elle est apparue sur certains modèles de la marque en 2017. Il s'agit d'une évolution du système original « SYNC » qui faisait partie des premières technologies de connectivité, d'information et de divertissement pour véhicules particuliers.

Le nouveau système « SYNC3 » permet l'utilisation de systèmes d'infodivertissement embarqués avec des commandes vocales simples. Il permet aussi de passer et de recevoir un appel téléphonique, envoyer et écouter des SMS, trouver des points d'intérêts ou encore écouter de la musique.

Le nouveau système « SYNC3 » intègre toutes les fonctionnalités de son prédécesseur « SYNC2 », auquel il ajoute la possibilité de faire défiler et agrandir les images sur l'écran comme sur un smartphone et des commandes vocales avancées pour des fonctionnalités supplémentaires telles que le contrôle de la climatisation.

Il est également compatible avec les technologies des plateformes d'infodivertissement telles que Android Auto de Google et Apple CarPlay. Cela permet de connecter directement à la voiture un périphérique Apple ou Android pour utiliser des applications dédiées à la navigation par satellite ou au multimédia.

Avec le système « SYNC3 », grâce à la fonction « mirroring », l'écran LCD du véhicule affiche une réplique exacte de l'écran du smartphone.

barrettes en l'orientant correctement. Son repère en forme de « U » doit se situer face au condensateur C6. Vérifiez de nouveau l'orientation de tous les composants polarisés ainsi que celle du module.

Une fois l'assemblage terminé et la vérification effectuée, vous pouvez tester la Demoboard.

Pour l'alimenter, vous devez utiliser une source de tension continue comprise entre 9 V et 12 V capable de délivrer 500 mA si vous avez opté pour le branchement d'un haut-parleur. Sinon, une source d'alimentation de 200 mA suffit si vous connectez un casque ou un amplificateur BF externe.

Au niveau de la prise jack vous pouvez connecter soit directement un haut-parleur de 8 Ω / 3 W ou encore un petit étage de puissance BF monophonique d'une dizaine de watts avec un potentiomètre sur son entrée pour régler le volume.

Vous pouvez alors y relier une petite enceinte de quelques watts afin d'obtenir une meilleure qualité que celle provenant d'un simple HP.

Si vous avez l'intention de lire des fichiers à partir d'une carte SD, insérez-la dans l'emplacement qui se trouve sur le module MP3 avant de connecter ce dernier sur ses barrettes, sinon laissez l'emplacement SD vide si vous utilisez un clé USB.

Notez que l'emplacement de la carte SD se trouve sur la partie supérieure du module, elle est donc accessible. Si par la suite vous devez utiliser et/ou changer de carte SD, faites-le délicatement en débranchant au préalable l'alimentation de la Demoboard.

Si vous voulez **connecter la Demoboard à un PC, utilisez la prise USB du module ET782M** et pas celle présente sur le circuit imprimé où vient s'insérer la clé USB à partir de laquelle sont lus les fichiers audio.

Appuyez sur une touche entre 01 et 14 et vérifiez que le fichier correspondant est lu.

Ajustez le volume avec la touche « Prev/Vol- », car il sera probablement trop élevé. En effet, le module démarre automatiquement avec le niveau de volume le plus élevé.

Utilisation

Comme nous l'avons mentionné, la Demoboard prend en charge les fichiers MP3 et WAV et effectue une lecture automatique depuis une clé USB ou une carte SD. Attention, il faut **insérer soit la clé USB soit la carte SD mais pas les deux à la fois**. Dès que l'un des 2 types de périphériques est inséré, appuyez sur la touche « U/SD/SPI » pour sélectionner la source des fichiers audio, sinon le module ne reproduira rien.

Pour que le module MP3 reconnaisse et reproduise correctement les fichiers audio, vous devez les enregistrer sur le support de masse dans le format : « **0001_NomPiste.mp3** » où le nom du fichier doit commencer par l'identifiant numérique composé des 4 caractères, c'est-à-dire le numéro de la piste dans un format à 4 chiffres : 001, 002, 003 jusqu'à 0014.

À la place « NomPiste » vous pouvez donner le nom que vous voulez, l'important est qu'il soit précédé du format à 4 chiffres que vous voulez assigner avec le tiret bas (touche 8 ou underscore).

Tableau 1 – Syntaxe des commandes

Élément	Fonction	Description
$S	bit de démarrage 0x7E	tous les retours des commandes commencent avec
VER	version	informations sur la version de la carte
Len	nombre d'octets suivants	0 - 2999
CMD	commande à exécuter	indique l'opération que le module doit effectuer
Feedback	retour de la commande	1 = envoie du retour; 0 = pas de retour
para1	paramètre 1	demande l'octet haut des données
para2	paramètre 2	demande l'octet bas des données
checksum	somme de contrôle ou parité	vérification (ne comprend pas le bit de démarrage $)
$O	bit d'arrêt	valeur 0xEF

Tableau 2 – Commandes séries pour les fonctions de lecture.

Commande	Fonction	Paramètre (16 bits)
0x01	Suivant	
0x02	Précédent	
0x03	Suivi spécial (NUM)	0 - 2999
0x04	Augmente le volume	
0x05	Diminue le volume	
0x06	Spécifie le volume	0 - 30
0x07	Paramétrage de l'égaliseur 0/1/2/3/4/5	Normal/Pop/Rock/Jazz/Classic/Bass
0x08	Définit le mode de lecture (0/1/2/3)	répétition/répétition du dossier/simple répétition/aléatoire
0x09	Définit la source de lecture(0/1/2/3/4)	U/TF/AUX/SLEEP/FLASH
0x0A	Entre en veille ; faible consommation	
0x0B	Fonctionnement normal	
0x0C	Réinitialisation du module	
0x0D	Lecture	
0x0E	Pause	
0x0F	Spécifie le dossier à lire	1 - 10 (à besoin d'être défini par l'utilisateur)
0x10	Réglage du volume	[DH=1 : réglage volume][DL : réglage du gain 0 -31]
0x11	Répète la lecture	[1 : commence la répétition][0 : arrête la lecture]

Par exemple : « 0001_minette.mp3 » ou encore « 0005_toto.mp3 ». De cette façon, vous pouvez contrôler la Demoboard correctement à partir des boutons, ou à l'aide de commutateurs CMOS ou encore avec des lignes d'un microcontrôleur configurées en sorties. Ainsi une corrélation est établie entre la piste à reproduire et la ligne à activer.

Rappelez-vous que l'interface série/USB permet de contrôler la Demoboard à partir d'un ordinateur, cependant aucun logiciel spécifique n'a été développé.

La gestion peut être réalisée à l'aide d'un logiciel d'émulation de terminal (Windows Hyperterminal, par exemple) en suivant la syntaxe et les commandes spécifiées par le constructeur du module (DFRobot) dans le manuel d'utilisateur.

Rappelez-vous aussi que les fichiers audio sont recherchés par le microcontrôleur qui gère le module MP3 en commençant par le **dossier racine** (root) de la mémoire de masse et en procédant par **ordre alphabétique**.

Le module prend en charge un maximum de 100 dossiers et un maximum de 1000 pistes peuvent être stockées dans chaque dossier. Il est évident que les 14 premières pistes peuvent être adressées par les boutons, la reproduction des autres ne peut être effectuée que par l'interface série avec des commandes appropriées.

Les formats de fichiers supportés sont conformes à la norme « ISO 11172-3 » et au décodage audio « ISO13813-3 layer3 ». Les taux d'échantillonnages supportés sont les suivants : 8/11,02 5/12/16/22,05/24/32/44,1 /48 kHz.

Le contrôle du volume s'effectue sur 30 pas, du minimum au maximum. Ceci est valable à la fois pour le contrôle manuel du volume par les boutons et pour le contrôle via le port série.

À partir du port série, il est également possible de choisir parmi l'un des 6 effets d'égalisation du son (par exemple Normal/Pop/Rock/Jazz/Classic/Bass).

En terme de communication au niveau du port série, le protocole prévoit de paramétrer le PC de la manière suivante :

- bits de données : 1 ;
- pas de somme de contrôle (pas de checksum ou pas de parité) ;
- pas de contrôle de flux.

La syntaxe des commandes à envoyer via le port série doit être dans ce format : « $S VER Len CMD Feedback para1 para2 checksum $O ».

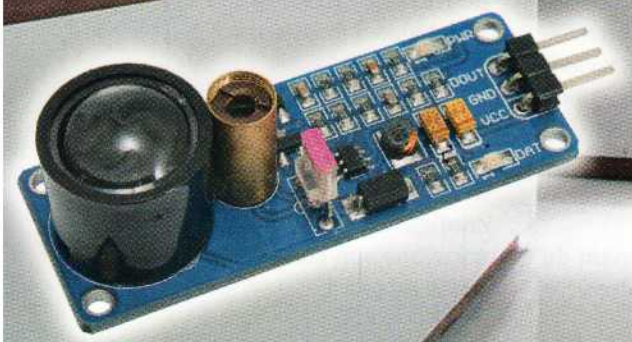
Comme vous pouvez le voir, « $S » correspond au bit de démarrage et chaque chaîne se termine par le bit d'arrêt « $O ».

Le tableau 2 reporte les commandes principales à envoyer au module, il s'agit des commandes régissant les fonctions de lecture des pistes audio.

Notez que les paramètres doivent être présents uniquement dans les commandes qui le nécessitent, telles que celles qui indiquent la piste à lire, le niveau du volume d'écoute, etc. ■

Figure 2 : pour relier la Demoboard à un PC, vous devez insérer le convertisseur USB/série dans le connecteur approprié.

BARRIÈRE LASER



Ce montage détecte la présence
et donc la rencontre avec un objet
grâce à la combinaison d'une diode laser émettant un rayon
de lumière et d'un phototransistor qui détecte le rayon réfléchi.

de Alessandro Sottocornola

Ce type de détecteur particulier est utilisé dans les applications robotiques, mais aussi dans les systèmes de contrôle industriel ou encore les systèmes optiques. Il permet de déceler la proximité ou la présence de divers objets, en détectant l'interruption de la lumière ou son reflet sur une surface de l'objet.

Les radars routiers de première génération étaient basés sur la réflexion de deux faisceaux lumineux qui constituaient deux barrières lumineuses qui étaient alors coupées par le passage d'un véhicule. La vitesse était détectée en comptant le temps écoulé entre la coupure des deux faisceaux, on obtenait ainsi la distance parcourue entre les deux points (deux faisceaux), on en déduisait la vitesse.

Dans cet article, nous vous proposons la **conception** d'une **barrière laser qui n'est pas de type à interruption**, c'est-à-dire que l'objet à détecter ne doit pas passer entre un émetteur et un photodétecteur pour relever sa présence, mais il peut être **détecté par réflexion**.

Notre circuit **projette un faisceau laser dont la lumière est concentrée et collimatée**. Un objet qui passe devant lui réfléchit une partie qui est renvoyée par une lentille et concentrée sur la surface sensible d'un photodétecteur.

Pour rappel, **une lumière collimatée est une lumière dont les rayonnements sont quasiment parallèles à l'infini**. Cela permet d'obtenir une lumière qui ne se disperse pas avec la

distance, ou qui sera très peu dispersée dans la réalité.

Par conséquent, avec cette barrière, **l'objet à détecter ne doit pas interrompre le faisceau laser mais il doit être dirigé vers le photodétecteur**.

Ce type de barrière a l'avantage de ne pas nécessiter le positionnement à une certaine distance du faisceau émetteur par rapport au photodétecteur. Cela simplifie considérablement le câblage, et permet de concentrer toute l'électronique en un seul endroit.

Ce type de capteur est idéal pour les systèmes mobiles, tels que les robots.

La distance de détection typique est de 80 cm, mais elle peut atteindre un maximum de 1,5 m. Cela dépend de la façon dont la surface de l'objet absorbe la lumière ou la reflète.

Le circuit peut être utilisé pour détecter des obstacles, tels que des pièces de monnaies dans un appareil ou encore des personnes pour ouvrir une porte, etc.

Schéma électrique

Examinons le schéma électrique du circuit. Nous utilisons un laser comme émetteur et un module, dénommé **H2**, faisant office de récepteur et qui comporte un phototransistor.

Le faisceau émis par l'émetteur est réfléchi sur la surface frontale de l'objet et frappe la lentille qui sert à concentrer la lumière réfléchie sur le phototransistor du récepteur.

Ainsi, le phototransistor entre en conduction. Le récepteur ne détecte que la lumière rouge émise et réfléchie par le laser, et filtre les perturbations provenant d'autres sources lumineuses.

Le laser est constitué d'une diode laser refroidie dans un boîtier approprié, d'un régulateur de courant et d'une lentille collimatrice. La **diode laser** est alimentée (**pilotée**) par le transistor **Q1** (un BC817 monté en émetteur commun), dont la base est pilotée par un signal rectangulaire de **180 kHz** généré par un oscillateur. Ce dernier est nommé **H1** dans le schéma électrique.

À chaque **niveau logique haut** présent sur la broche 1 de l'oscillateur (la broche 2 est l'alimentation 5 VDC de l'oscillateur), le **transistor entre en saturation**, se met donc à **conduire** et alimente ainsi la diode laser.

Celle-ci émet alors une impulsion lumineuse. Lorsque le signal sur la base de Q1 est nul, le transistor est bloqué et la diode laser est désactivée (éteinte).

Lorsque la diode laser est allumée, son courant est limité par la résistance R5 montée en série avec le collecteur du transistor Q1. L'onde lumineuse est envoyée frontalement et lorsqu'elle heurte un objet dont la surface en réfléchit au moins une partie, elle revient vers le circuit pour atteindre la lentille du phototransistor. Ainsi, la lentille fait converger la lumière vers la jonction du phototransistor.

Si le faisceau lumineux provenant de la diode laser est réfléchi et atteint le phototransistor avec une intensité suffisante, la jonction de ce dernier, qui contient des charges libres (c'est-à-dire des électrons, car le transistor est un NPN), engendre un courant de base qui est ensuite amplifié par le champ électrique présent au niveau du collecteur.

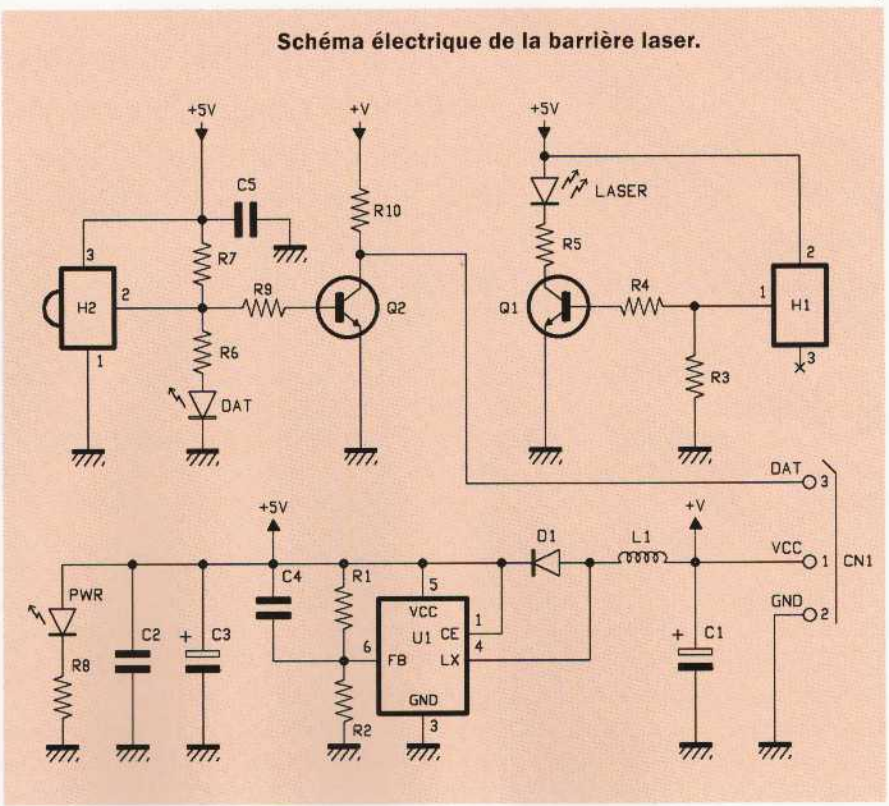
En effet, la tension d'alimentation de 5 VDC est reliée au collecteur du phototransistor au niveau de la broche 3. La masse, qui est reliée à la broche 1 de H2, fait référence en interne à la résistance d'émetteur.

Sur la broche 2 du phototransistor est disponible le signal de son émetteur. Ce type de **phototransistor est sensible à la lumière infrarouge et visible**. Il est configuré en collecteur commun, de sorte que son collecteur est alimenté par la broche 1.

CARACTÉRISTIQUES TECHNIQUES

- détection laser par réflexion ;
- distance de détection de 8 cm à 150 cm ;
- sortie digitale niveau TTL ;
- tension d'alimentation : de 2,5 VDC à 12 VDC ;
- courant consommé : 150 mA ;
- dimensions en mm : 47,7 x 17,9 x 20.

Schéma électrique de la barrière laser.



La broche 2 est à 0 V en l'absence de lumière et à environ 4 V selon la quantité de lumière reçue sur sa base qui correspond à la jonction exposée au faisceau lumineux.

Le transistor Q2, un BC817, est également monté en émetteur commun. Dans les conditions de repos (broche 2 de H2 à 0 V) le transistor est saturé par l'intermédiaire de la résistance de tirage (pull-up) R7, qui à l'aide de R9 polarise la base. Cela a pour effet d'allumer la LED nommée DAT, dont le courant est limité par la résistance R6.

Notez que dans cette condition, la tension au niveau du collecteur de Q2 est proche de 0 V car il est conducteur.

Lorsque le **phototransistor est frappé par une lumière d'intensité suffisante**, la **tension** de son collecteur **chute de 5 V** à environ un peu moins de **0,7 V**.

Cette tension n'est pas suffisante pour polariser la base de Q2, ce dernier se bloque et **son collecteur** (de Q2) prend un **niveau logique haut** à l'aide de R10.

Ainsi, pour **chaque impulsion lumineuse réfléchie** qui atteint le phototransistor, la **broche DAT**(3) du connecteur CN1 **passe d'un niveau logique 0 à un niveau logique 1**. Donc, la broche DAT est au repos lorsque la diode laser n'émet pas de lumière.

Cette broche peut être lue par un microcontrôleur afin de détecter la présence d'un objet. Un niveau logique haut correspond à une tension de 5 V et un niveau bas à environ 0 V, de sorte qu'il est possible d'interfacer la barrière laser avec des systèmes logiques fonctionnant à des niveaux TTL.

La LED DAT répète de manière inversée l'état de la sortie. En fait, lorsque la LED DAT est allumée (le phototransistor ne reçoit pas de lumière, il n'y a pas d'objet), la sortie DAT(3) est à un niveau logique bas. En présence d'un objet, la LED s'éteint et la sortie DAT(3) passe à un niveau logique 1.

Notez que le choix de piloter la diode laser avec un signal rectangulaire au lieu d'un signal continu, provient de la nécessité d'immuniser le fonctionnement de la barrière laser contre les perturbations lumineuses du milieu ambiant.

En effet, si la diode laser était continument allumée, la sortie DAT(3) présenterait une tension susceptible d'être influencée par la lumière ambiante, car le phototransistor pourrait être constamment conducteur s'il était suffisamment éclairé par une source lumineuse ambiante de grande intensité et de longueur d'onde correspondant à sa sensibilité.

Par contre, en émettant des impulsions lumineuses à une certaine fréquence, nous pouvons nous assurer qu'en disposant un filtre sur le photorécepteur, il ne détectera pas les sources lumineuses continues.

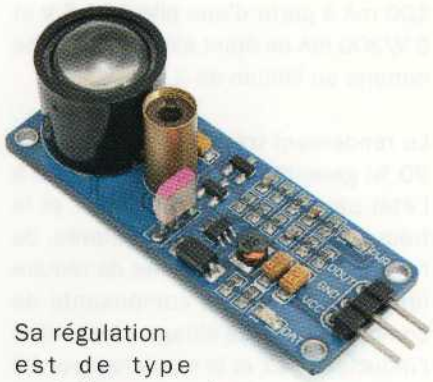
En effet, le photorécepteur H2 est équipé en sortie (sur la broche 2) d'un **filtre** qui ne laisse passer que les signaux d'une fréquence de **180 kHz** correspondant aux impulsions de la diode laser.

L'utilisation d'un filtre passe-bande à la sortie du circuit discrimine les signaux, en éliminant toutes les impulsions lumineuses sans rapport avec le système de détection de notre circuit. De plus, il rend possible l'interfaçage du circuit avec un microcontrôleur.

Notez que la fréquence élevée d'allumage et d'extinction de la diode laser n'interfère pas avec la détection des impulsions, même si celles-ci sont courtes, elles sont plus longues que le retard introduit par la réflexion de la lumière sur le phototransistor.

Le circuit peut être alimenté par une tension comprise entre 2,5 VDC et 12 VDC, grâce à la présence d'un **régulateur à découpage**. Il s'agit du circuit U1 (PT1301) qui fournit 5 VDC en sortie, quelles que soient les conditions de charge et la tension présente sur son entrée, c'est-à-dire entre le contact « VCC » et la masse du connecteur CN1.

La tension d'alimentation appliquée sur le connecteur CN1 est localement filtrée par le condensateur électrolytique C1. Le régulateur à découpage **PT1301 est un convertisseur compact DC/DC** (continu/continu) à **haut rendement**.



Sa régulation est de type contrôle en mode courant adaptatif PWM ou « **Adaptive Current Mode PWM** ».

À l'intérieur du circuit intégré U1, nous trouvons un amplificateur d'erreur, un générateur de signal en dents de scie (générateur de rampe), un comparateur de tension pour implémenter la modulation PWM.

Celle-ci permet de comparer le signal en dents de scie par rapport à la composante continue de contre-réaction appliquée sur la broche « FB » à l'aide du diviseur de tension R1/ R2. Cette composante de contre-réaction est filtrée par le condensateur C4 en parallèle sur la résistance R1.

L'étage de sortie du circuit U1 est constitué d'un transistor MOSFET à enrichissement de type canal N pour piloter l'inductance et obtenir ainsi des impulsions PWM en sortie.

Le MOSFET interne au circuit PT1301 peut commuter des courants jusqu'à 300 mA, mais si vous avez besoin de gérer des courants plus importants, vous pouvez utiliser la sortie auxiliaire (broche EXT de U1) pour contrôler un MOSFET externe de plus grande puissance.

Dans ce cas, la sortie correspondant au drain du MOSFET interne, c'est-à-dire la broche LX, se connecte en parallèle au drain du MOSFET externe.

La particularité du circuit PT1301 est qu'il peut démarrer et fonctionner avec des tensions inférieures à 1 V (le minimum est de 0,8 V) appliquées sur son entrée, ce qui le rend approprié pour les applications fonctionnant sur batterie.

Le circuit peut fournir une tension de sortie de 3,3 V avec un courant de

100 mA à partir d'une pile de 1,5 V et 5 V/300 mA en étant alimenté par une batterie au lithium de 3,6 V.

Le rendement très élevé (supérieur à 90 %) garantit une faible résistance à l'état passant (ON) du MOSFET et la fréquence de commutation élevée, de l'ordre de 500 kHz, permet de réduire les dimensions des composants de commutation et de filtrage, c'est-à-dire l'inductance L1 et le condensateur C3 (et également C4, qui agit comme un filtre résiduel à 500 kHz).

Le fonctionnement en commutation est le suivant. Au démarrage, à travers l'inductance L1 et la diode D1, le courant atteint la broche 5 (VCC) du circuit U1. Les condensateurs **C3** et **C4 se chargent** et, immédiatement après, le circuit intégré commence à fonctionner.

Son **comparateur** interne envoie une **impulsion positive sur la grille du MOSFET** interne. Ce dernier devient alors conducteur (état ON), le drain et la source sont alors court-circuités et un courant circule vers la masse. Pendant ce temps, le reste du circuit intégré est alimenté par la tension présente aux extrémités des condensateurs C3 et C4.

L'inductance, quant à elle, emmagasine de l'énergie et peu après le blocage du MOSFET, l'**énergie stockée dans L1 est transférée** (car l'inductance a tendance à maintenir le courant qui circulait avant le blocage du MOSFET, générant ainsi une surtension inverse de polarité positive sur l'anode de la diode) **à travers la diode D1**, à partir de laquelle nous obtenons en sortie du convertisseur DC/DC une tension de **5 VDC**.

La LED connectée en série avec la résistance R8 indique la présence de la tension de sortie du circuit U1.

La tension présente sur la broche FB atteint le comparateur interne, ce dernier permet d'ajuster la largeur de l'impulsion de la tension envoyée sur la grille du MOSFET.

En fonction de la **tension** de la **grille**, le **MOSFET est plus ou moins conducteur**, c'est-à-dire qu'il court-circuite la self L1 vers la masse pendant une période d'autant plus grande que la tension présente sur la broche FB est faible et vice versa.

Lorsque le **MOSFET conduit**, la self **L1 emmagasine de l'énergie**. Le cycle recommence ainsi tant que le circuit est alimenté. Le circuit U1 fonctionne toujours car il est alimenté par la broche CE qui est directement connectée à la tension VCC. Lorsque la broche CE est reliée à la masse, le circuit PT1301 se met en mode veille, le convertisseur est alors au repos et ne consomme que 14 µA.

Plan de montage de la barrière laser

Plan de câblage des composants de la barrière laser.

Photo de l'un de nos prototypes de la barrière laser.

Liste des composants de la barrière laser

R1.....2,7 MΩ boîtier 0603
R2.....910 kΩ boîtier 0603
R3.....22 Ω boîtier 0603
R4.....22 Ω boîtier 0603
R5.....120 Ω boîtier 0603
R6.....1 kΩ boîtier 0603
R7.....4,7 kΩ boîtier 0603
R8.....1 kΩ boîtier 0603
R9.....1 kΩ boîtier 0603
R10....10 kΩ boîtier 0603
C1.....100 µF/16 V boîtier 1206
C2.....1 µF/10 V céramique boîtier 0603
C3.....100 µF/10 V boîtier 1206
C4.....100 pF/10 V céramique boîtier 0603
C5.....100 nF/10 V céramique boîtier 0603

L1 ..... inductance 10 µH
D1..... 1N5819 SMD
U1..... PT1301
Q1..... BC817
Q2..... BC817
Diode laser
H1..... oscillateur 180 kHz
H2..... phototransistor IR
DAT .... LED rouge boîtier 0805
PWR... LED verte boîtier 0805

Divers

Barrette mâle 3 pôles

NB : les typons des circuits imprimés à l'échelle 1 sont disponibles en téléchargement dans le sommaire détaillé de la revue.

Réalisation pratique

Eh bien, après l'étude théorique, passons à la pratique. Le circuit comporte deux faces, comme d'habitude vous pouvez télécharger sur notre site web dans le sommaire détaillé de la revue les typons à l'échelle 1 : 1 et/ou les fichiers Gerber pour une fabrication professionnelle.

Les composants utilisés sont presque tous des modèles CMS, vous devez donc être très soigneux pour la construction.

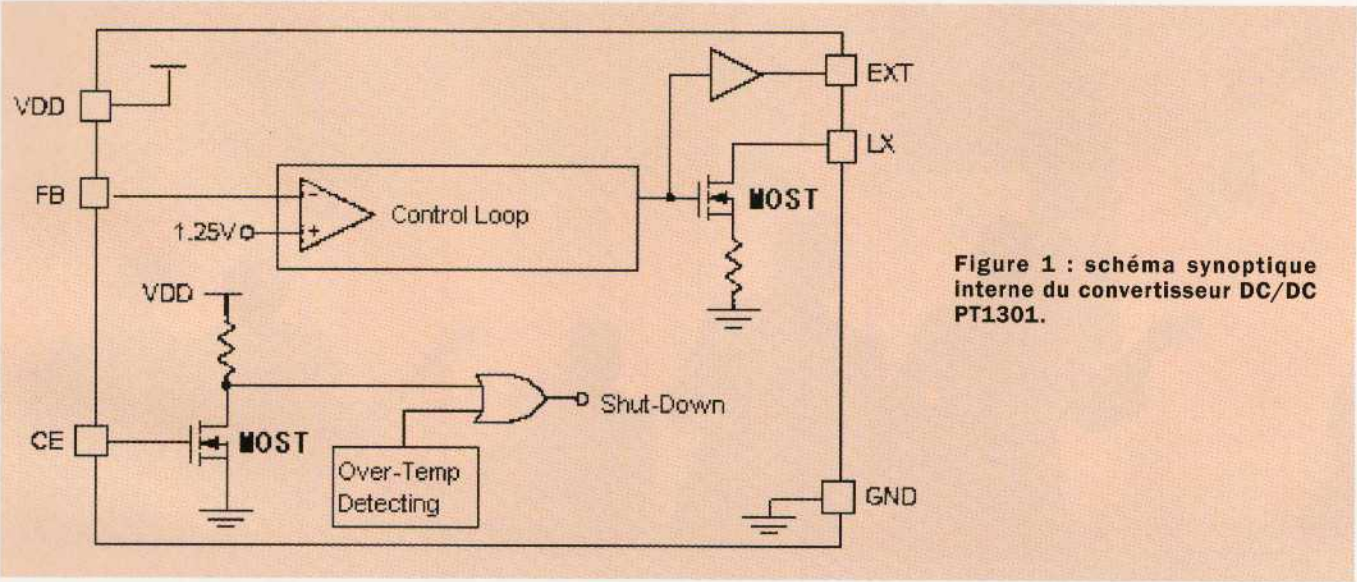


Figure 1 : schéma synoptique interne du convertisseur DC/DC PT1301.

Ceux d'entre vous qui ne se sentent pas capables de manipuler les composants CMS, pourront se procurer la barrière laser dans le commerce. Elle est disponible à la vente déjà montée sous forme de module prêt à l'emploi.

Ceux qui préfèrent la construire doivent être équipés d'une bonne dose de patience, d'un minimum d'expertise, d'un **fer à souder de 20 W** avec une pointe de **0,2 mm de Ø** et de la soudure très fine de 0,5 mm de Ø.

Il est préférable d'utiliser une pince à épiler pour placer les composants et une bonne loupe pour vérifier que chaque composant est placé au bon emplacement et correctement soudé (éviter les bavures ou les cheveux d'ange).

Commencez par souder les résistances (boitiers 0603) ainsi que les condensateurs non polarisés. Ensuite, continuez avec les condensateurs polarisés en respectant leur orientation (C1 et C3).

Continuez avec la diode Schottky D1 et les LED toujours en prêtant une attention particulière. Vérifiez plusieurs fois l'orientation avant de les souder, car le dessoudage de composants n'est jamais simple.

Pour le circuit intégré et les transistors, utilisez une pointe de colle pour les maintenir sur le circuit avant de les souder.

La broche 1 de U1, marquée par un point sur le boîtier, doit se situer du côté de l'oscillateur H1.

Ensuite, soudez la self L1, l'oscillateur H1, la diode laser et le phototransistor ainsi que la barrette mâle au pas de 2,54 mm pour les connexions.

Le phototransistor est de type à grande lentille, spécialement conçu pour les télémètres lasers et les dispositifs qui doivent intercepter une lumière laser réfléchie.

Si vous le changez par un modèle traditionnel, la distance de détection ne sera pas la même que celle annoncée.

Quant à la diode laser, vous pouvez prendre n'importe quel modèle de quelques milliwatts, émettant dans l'infrarouge.

Le composant n'est pas critique, de sorte que vous pouvez le prendre d'un petit pointeur laser que l'on trouve dans le commerce, à condition qu'il émette dans l'infrarouge (ou proche de l'infrarouge) qu'il puisse être alimenté en 5 VDC.

La consommation de la diode laser ne doit pas dépasser 200 mA.

Nous donnons ci-après le code source d'un sketch fonctionnant sous Arduino pour une application typique, afin de tester le montage.

Exemple d'application typique

```
int laser_din=2;

void setup()
{
  pinMode(laser_din,INPUT);
  Serial.begin(9600);
}
void loop()
{
  if(digitalRead(laser_din)==LOW)
  {
    Serial.println(«PAS Obstacle !»);
  }
  else
  {
    Serial.println(«Obstacle !»);
  }
  delay(500);
}
```

La connexion à Arduino s'effectue de la manière suivante :

CN1		Arduino
DOUT	→	D2
GND	→	GND
VCC	→	5 V

NB : ne jamais pointer le laser directement dans les yeux, cela pourrait provoquer des lésions irréversibles de la rétine !

PÉDALE MULTI-EFFETS

de ***Boris Landoni***

Nous vous avons présenté précédemment un circuit d'effet trémolo pour guitare électrique, nous complétons ce montage avec une pédale multi-effets proposant deux types de distorsions fuzz : symétrique et asymétrique. De plus, elle dispose d'un contrôle de volume et de tonalité et se relie à une guitare électrique.

Il n'y a pas si longtemps, dans le numéro 139, nous avons publié un amplificateur pour guitare spécialement conçu pour la maison et dans le numéro 141, un circuit d'effet trémolo pour guitare électrique.

Nous complétons ces montages en vous proposant une **pédale multi-effets**. Il s'agit d'une pédale qui peut être utilisée avec le pied et qui intègre différentes fonctions. Elle permet deux types de **distorsions fuzz** : **symétrique**

et **asymétrique**, ainsi qu'un **overdrive** et un **réglage** du contrôle de **tonalité**.

L'effet fuzz est de type à distorsion « **clipping** » (écrêtage) du signal, c'est à dire l'écrêtage de la demi-onde positive et négative, mais cet effet est possible à la fois de manière symétrique, asymétrique ou semi-asymétrique.

Il en résulte deux effets sonores différents qui modifient le son de la guitare de différentes manières (l'un au niveau de la première harmonique et l'autre au niveau de la seconde harmonique).

Le contrôle de tonalité est de type classique, c'est-à-dire qu'il corrige la courbe de réponse du signal en renforçant ou en affaiblissant les graves et les aigus.

Contrairement aux préamplificateurs Hi-Fi, il n'y a qu'un seul contrôle, donc si vous augmentez les basses fréquences, le correcteur atténuera les hautes fréquences et vice versa. Cela rend le son plus plat ou grave.

En ce qui concerne l'overdrive, il s'agit d'un effet similaire à celui du gain Les deux ajustent la distorsion de l'amplificateur qui est relié à la pédale.

L'effet **overdrive** consiste essentiellement en une amplification supplémentaire qui peut **augmenter considérablement le niveau du son** de la guitare, de manière à **saturer l'entrée** de l'amplificateur (ou de la table de mixage, ou autre). Pour cette raison, en augmentant l'overdrive vous augmentez le volume et donc la distorsion.

Caractéristiques techniques

- **Tension d'alimentation : 9 VDC ;**
- **Consommation de courant : 40 mA ;**
- **Contrôles du volume et de la tonalité ;**
- **Effets : overdrive, distorsion symétrique/asymétrique/semi-asymétrique ;**
- **Alimentation par batterie ou secteur ;**
- **Connexions par jack 6,35 mm ;**
- **Buffer en entrée et en sortie ;**
- **Dimensions : 115 x 65 x 25 mm.**

Donc si vous n'insérez pas l'effet fuzz, vous ajustez le niveau audio, tandis qu'avec l'effet inséré, vous diminuez ou augmentez le niveau du son de la guitare qui déclenche la distorsion.

Analysons maintenant le schéma électrique de la pédale multi-effets. Pour l'instant considérons qu'un effet de distorsion déforme le signal de l'étage de sortie, alors que l'effet overdrive sature l'étage d'entrée.

Schéma électrique

Notre pédale se compose de **4 étages** à **amplificateurs opérationnels**, dont deux sont des amplificateurs, tandis que l'un procure à la fois une amplification et une distorsion. Le dernier étage est utilisé pour le contrôle de la tonalité.

En vous référant au schéma électrique, vous constatez que le signal d'entrée atteint le circuit via la prise IN (SK1) qui est un jack 6,35 mm stéréo, mais dont le **contact du second canal est utilisé pour commander l'activation** de l'alimentation lorsqu'il est inséré.

Ce contact met à la masse la base du transistor T1 dès que le jack est inséré, ce qui a pour effet de saturer le transistor (il devient conducteur) car sa base est polarisée par la résistance R10 reliée à la tension d'alimentation à travers la diode D4 qui protège contre une inversion de la polarité.

Dans les conditions de repos (jack non présent) le transistor T1 est bloqué, cette condition implique également que le transistor T2 est bloqué, donc **si le jack n'est pas inséré** dans la prise SK1, le **circuit ne consomme** théoriquement **aucun courant**.

Cette configuration est extrêmement pratique car elle dispense de l'utilisation d'un interrupteur d'allumage et de sa manipulation. Vous disposez donc d'une liberté d'utilisation maximale, car le circuit n'est alimenté (et donc consomme) que lorsque le jack est inséré dans la prise d'entrée.

Cela signifie que **vous n'avez pas à vous préoccuper de déconnecter l'alimentation lorsque vous ne vous servez plus de la pédale**. En cas de fonctionnement sur batterie, il n'y a pas de risque de décharge prématurée.

En insérant le jack de la guitare, le contact est déconnecté de la masse et « libère » donc la base du transistor T1 de la masse (GND). Ce **dernier devient conducteur**, ce qui a pour effet de mettre à la masse la broche de la résistance R12 qui est reliée au collecteur de T1.

Il en découle que la base du transistor T2 (NPN) se trouve alors polarisée. **T2 devient conducteur** et son collecteur alimente la ligne « +9V » avec une tension inférieure à celle de l'alimentation présente sur la cathode de D4.

Le transistor **T2 fonctionne comme un interrupteur statique**. La tension est filtrée par les condensateurs C2 et C4, puis par le réseau R/C constitué par la résistance R18 et le condensateur C13 afin d'obtenir la tension « **+9Vr** ».

Cette dernière alimente les **diviseurs de tension qui fournissent les tensions de références aux amplificateurs opérationnels**. Cela est **nécessaire pour amener la tension de sortie des AOP à environ la moitié de leur tension** d'alimentation, afin de permettre une excursion positive et négative du signal de sortie.

De plus, les AOP alimentés à partir d'une tension unique doivent être exempts de perturbations et nécessitent donc une alimentation filtrée correctement.

Par exemple, une ondulation pourrait se produire dans le circuit lorsqu'il est alimenté à partir du secteur (bruit du 50 Hz ou ronflement), ou encore ils pourraient amplifier un bruit de fond déjà présent dans la chaîne du signal, et donc engendrer un bruit perceptible à l'écoute.

En effet, si une perturbation entre dans les entrées des amplificateurs opérationnels, elle est amplifiée. À moins qu'elle atteigne en même temps l'entrée inverseuse et non-inverseuse avec la même amplitude, ce qui est peu courant dans la réalité.

Selon la loi de Murphy, le pire est toujours certain ! C'est pourquoi la tension de référence doit être parfaitement filtrée.

L'alimentation prélevée sur le collecteur du transistor T2 est moins filtrée car elle sert à alimenter les AOP au niveau de leurs étages de sortie ainsi que la LED de signalisation LD1.

Cette dernière est alimentée en insérant l'effet via la pédale, c'est-à-dire via le commutateur SW2 qui est actionné avec le pied. Il se situe sur le couvercle de l'appareil. Cet inverseur commute à la fois l'alimentation de la LED et les lignes d'entrée et de sortie du signal.

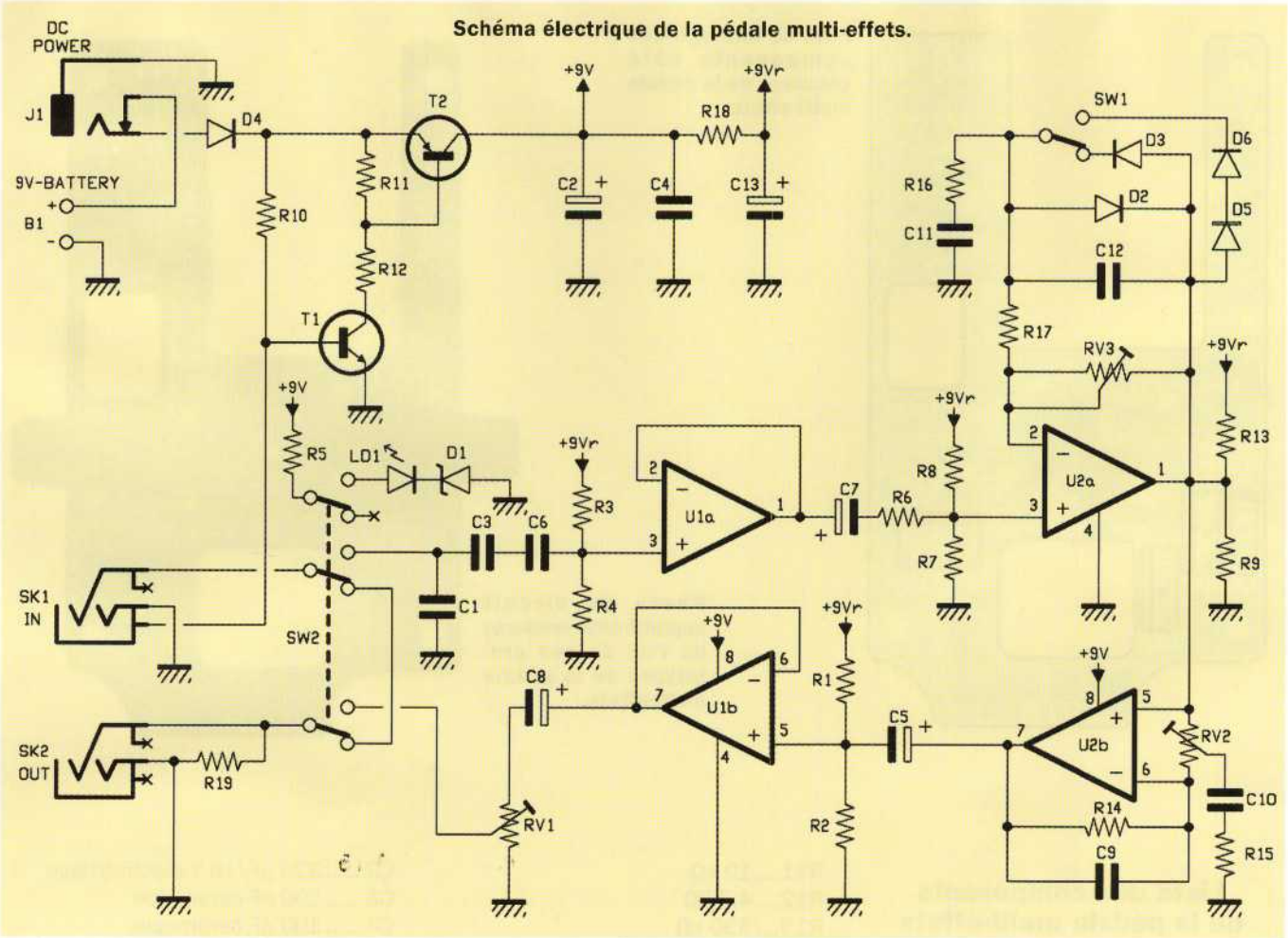
Notez la configuration particulière de la LED qui n'est allumée que, lorsque le circuit est alimenté, par une pression sur la pédale qui insère l'effet.

La différence de potentiel entre la tension d'alimentation sur la cathode de D4 et celle de la polarisation directe de LD1 est absorbée par la résistance R5 et la diode zener D1 connectée en série. Celle-ci maintient une tension de 3,3 V sur ses extrémités.

Le signal arrivant sur la prise d'entrée SK1 dans les conditions de repos, c'est-à-dire le circuit alimenté mais le jack non inséré, est directement dévié par le triple inverseur sur la prise de sortie SK2. Dans cette condition LD1 n'est pas alimentée, elle est donc éteinte. Cependant le circuit est alimenté.

Lorsque la pédale est enfoncée, le signal d'entrée arrive aux extrémités du condensateur **C1** qui **permet de filtrer les éventuelles perturbations**

Schéma électrique de la pédale multi-effets.



captées par les câbles. Ensuite, les deux condensateurs en série **C3** et **C6 bloquent toute composante continue** qui pourrait être présente avec le signal audio d'entrée.

Ce dernier atteint le **premier étage** à amplificateur opérationnel, c'est-à-dire l'**entrée non-inverseuse** (broche 3) de **U1a**, configuré en **étage tampon** (buffer) avec un **gain unitaire**. Ce type de montage présente une **haute impédance d'entrée**, ce qui évite de charger la sortie de la guitare (ou tout autre circuit) relié sur l'entrée IN.

La **polarisation de U1a** est **déterminée par le diviseur de tension** formé par les résistances **R3** et **R4**. Ainsi, l'entrée non-inverseuse est polarisée à environ 4,5 V pour que la sortie de U1a soit, au repos, à un potentiel permettant l'excursion du signal audio autour de cette valeur.

Comme nous l'avons évoqué, les condensateurs C3 et C6 bloquent la composante continue, tout comme C7, qui ne laisse passer que le signal audio provenant de U1a et qui est appliqué à travers R6 au second amplificateur opérationnel. Ce dernier, U2a, est configuré en tant qu'amplificateur non-inverseur dont le gain en tension Gv est déterminé par la formule :

Gv = RV3 / R6

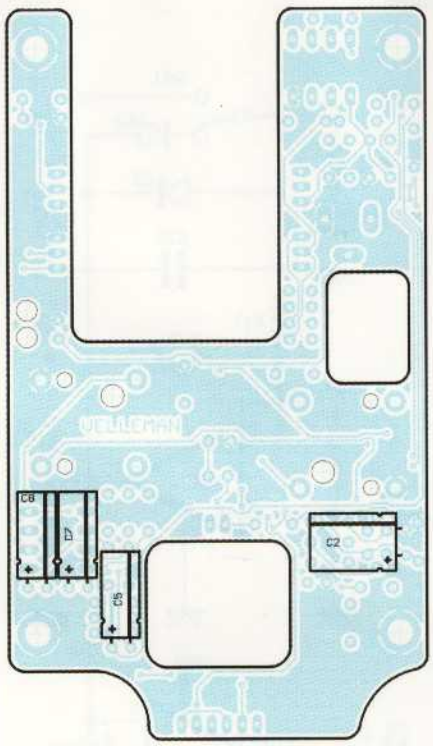
où RV3 est la résistance insérée par le potentiomètre RV3, monté en contre-réaction entre la sortie et l'entrée inverseuse de U2a, afin de permettre d'ajuster le niveau.

La formule est approximative car en réalité, en fonction de la position du commutateur SW1, il faut tenir compte de la valeur introduite par la résistance R17 et les résistances dynamiques des diodes D5 et D6 en série (ou de D2 et D3 en parallèle).

Le commutateur **SW1 permet de choisir l'effet de distorsion** à introduire, c'est-à-dire soit le classique effet fuzz qui correspond à une distorsion symétrique des deux crêtes du signal, soit une distorsion asymétrique qui agit plus nettement sur la demi-onde négative que sur la demi-onde positive.

Étant donné que dans le premier cas, il n'y a que **la seule diode D2 en contre-réaction** pour la demi-onde négative et **deux diodes** pour la demi-onde positive, le gain en présence du signal, si l'interrupteur SW1 qui insère la distorsion fuzz est fermé sur la cathode de D3, est calculé en considérant la résistance parallèle introduite par RV3, R17 et la diode en série (demi-onde).

Puisque la valeur de R17 est relativement plus petite par rapport à la résistance maximale insérée par le potentiomètre RV3, la **distorsion intervient donc plus brusquement**, c'est-à-dire déjà à de **faibles amplitudes du signal appliqué** à l'entrée du circuit, et inversement. **RV3** effectue donc le contrôle de l'**overdrive**.



Plan de câblage des composants côté soudures de la pédale multi-effets.

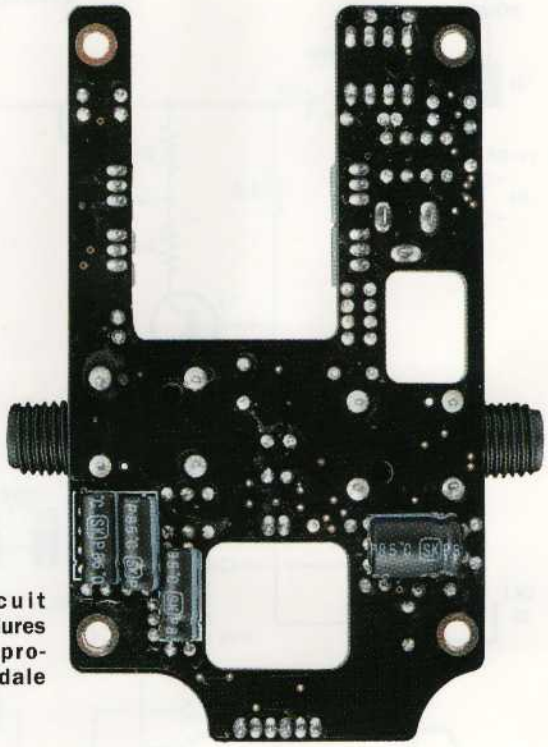


Photo du circuit imprimé côté soudures de l'un de nos prototypes de la pédale multi-effets.

Liste des composants de la pédale multi-effets

R1..... 330 kΩ
R2..... 470 kΩ
R3..... 330 kΩ
R4..... 470 kΩ
R5..... 390 Ω
R6..... 10 kΩ
R7..... 470 kΩ
R8..... 330 kΩ
R9..... 470 kΩ
R10.... 1 MΩ
R11.... 10 kΩ
R12.... 4,7 kΩ
R13.... 330 kΩ
R14.... 10 kΩ
R15.... 470 Ω
R16.... 470 Ω
R17.... 33 kΩ
R18.... 100 kΩ
R19.... 1 MΩ
RV1.... potentiomètre 100 kΩ
RV2.... potentiomètre 25 kΩ
RV3.... potentiomètre 1 MΩ

C1..... 470 pF céramique
C2..... 220 µF/16 V électrolytique
C3..... 100 nF céramique
C4..... 100 nF céramique
C5..... 10 µF/63 V électrolytique
C6..... 47 nF céramique
C7..... 10 µF/63 V électrolytique
C8..... 10 µF/63 V électrolytique
C9..... 10 nF céramique
C10.... 22 nF céramique
C11.... 47 nF céramique
C12.... 47 pF céramique
C13.... 10 µF 63 VL électrolytique

D1..... zener 3,3V 400mW

Le même contrôle s'applique également si l'interrupteur SW1 est fermé sur la cathode de la diode D6. Dans ce cas, nous avons une distorsion fuzz asymétrique. Le gain est déterminé par la résistance parallèle introduite par RV3, R17 et les diodes D5 et D6 en série. Dans ce cas également, le potentiomètre RV3 règle le niveau du signal d'entrée à partir duquel la distorsion commence.

Notez que la différence entre les deux types de distorsions fuzz réside dans la quantité et la qualité des harmoniques produites par le signal distordu.

Avec l'interrupteur SW1 positionné sur la cathode de D3 qui se trouve en parallèle avec le potentiomètre de contre-réaction RV3, sont insérées les diodes D1 et D2 et la résistance R17. Ces dernières se trouvent alors en parallèle et réduisent la tension aux bornes de la résistance R18 et donc la tension de sortie pour créer un son distordu typique causé par l'écrêtage.

L'**amplitude** du signal de sortie de **U2a** (V_{out}) est égale à la **somme de la chute de tension** due à la polarisation directe de la diode (0,6 V) et de la **résistance introduite** par R17 et RV3.

La tension de sortie peut être calculée de façon approximative par la formule :

$$V_{out} = V_{in} * R17 / R6$$

où V_{in} correspond à la tension appliquée sur le condensateur C7.

En réalité, dans le calcul, il faut tenir compte de la résistance introduite en parallèle par R6, R7 et R8.

À l'inverse, en positionnant l'interrupteur SW1 sur la cathode de D6, une demi-onde positive d'amplitude prévue s'additionne au 0,6 V et la demi-onde

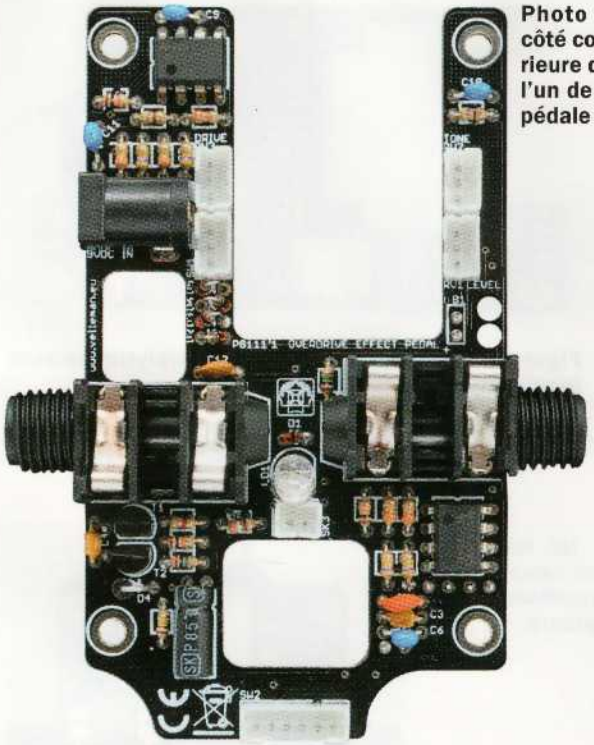
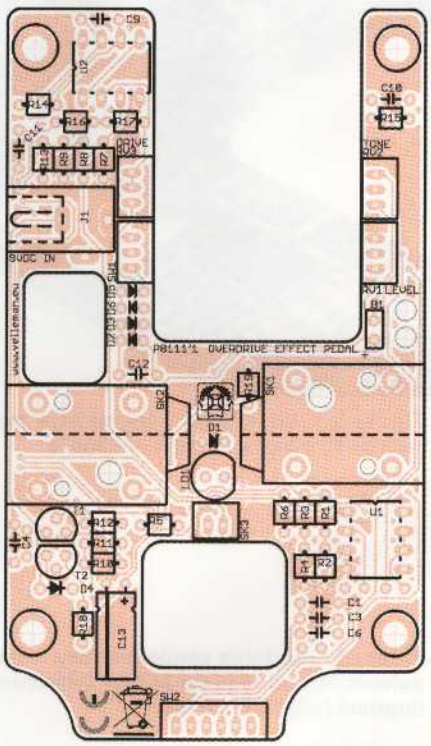


Photo du circuit imprimé côté composants (face supérieure du circuit imprimé) de l'un de nos prototypes de la pédale multi-effets.



Plan de câblage des composants côté composants (face supérieure du circuit imprimé) de la pédale multi-effets.

D2..... 1N4148
D3..... 1N4148
D4..... 1N4007
D5..... 1N4148
D6..... 1N4148

T1 ..... BC547
T2 ..... BC557

LD1.... LED 5 mm bleu

U1..... NE5532P
U2..... NE5532P
SW1... interrupteur à levier

SW2... double interrupteur

Divers

Fiche d'alimentation pour circuit imprimé
Clip pour pile 9V
Connecteur jack 6,35 mm pour circuit imprimé (x2)
Bouton pour potentiomètre (x3)
Connecteur mâle 3 pôles JST 2 mm (x4)
Connecteur femelle 3 pôles JST 2 mm fils volants (x4)

Connecteur mâle 6 pôles JST 2 mm
Connecteur femelle 6 pôles JST 2 mm fils volants
Connecteur mâle 2 pôles JST 2 mm
Connecteur femelle 2 pôles JST 2 mm fils volants
Boitier métallique pour circuit imprimé de 97 mm x 56 mm

NB : les typons des circuits imprimés à l'échelle 1 sont disponibles en téléchargement dans le sommaire détaillé de la revue.

négative est par contre écrêtée dans les deux sens.

En raison de l'effet de distorsion, l'amplitude du signal de sortie de U2a est limitée par rapport à ce qu'elle serait si seulement RV3 interviendrait, car l'amplificateur U2b est inséré dans le circuit, ce qui augmente encore l'amplitude et permet en même temps, grâce aux composants qui l'entourent, de contrôler la tonalité.

Le réglage est effectué grâce au potentiomètre **RV2** (Tone), qui en plus d'assurer la polarisation, détermine la fréquence de coupure du filtre R/C constitué par RV2 et le condensateur C10 et celle du filtre R/C parallèle de contre-réaction constitué par R14 et C9.

Le signal est ensuite amplifié et corrigé en fréquence par l'amplificateur opérationnel U2b pour atteindre, à travers le condensateur C5 de découplage qui bloque la composante continue présente entre la broche 7 de U2b et la masse, le dernier étage constitué par l'amplificateur opérationnel U1b configuré en étage tampon non-inverseur avec un gain unitaire.

Le diviseur de tension formé par les résistances R1 et R2 polarise l'entrée non-inverseuse pour assurer l'amplification des deux demi-ondes du signal.

Le circuit **U1b se comporte comme un adaptateur d'impédance**, le signal de sortie BF sort de sa broche 7 en passant à travers le condensateur de découplage C8 pour atteindre le potentiomètre **RV1**, utilisé comme **volume général** (Level).

Lorsque l'effet est activé, le commutateur SW2 transmet le signal BF à la prise jack de sortie (OUT) nommée SK2.

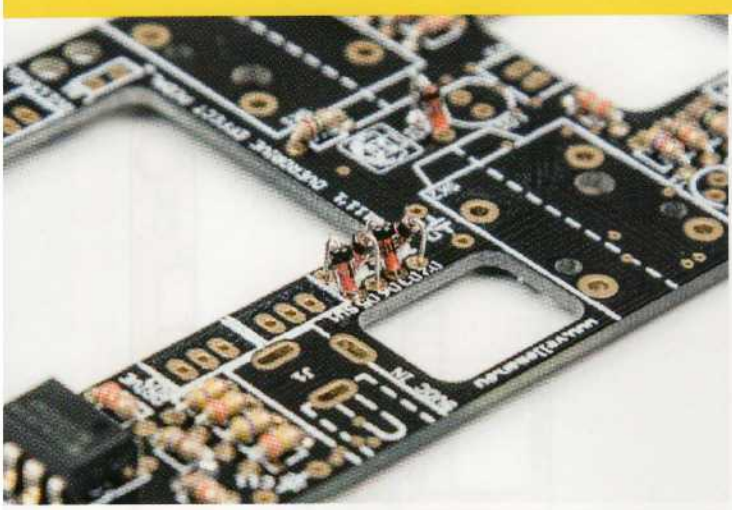


Figure 1 : les diodes doivent être montées debout, comme le montre l'image ci-dessus.

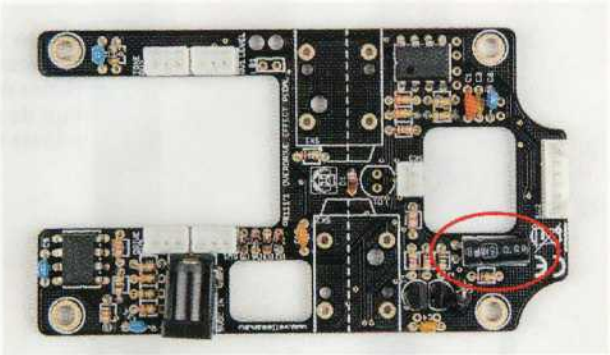
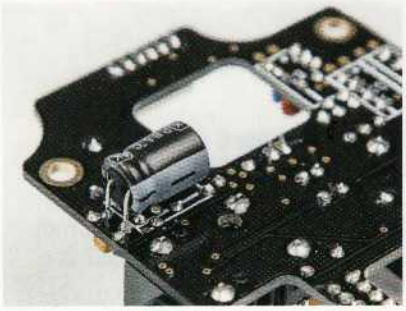


Figure 2 : Les condensateurs électrolytiques sont montés horizontalement.

Figure 3 : certains condensateurs électrolytiques doivent être soudés sur la face inférieure du circuit imprimé (côté soudures).



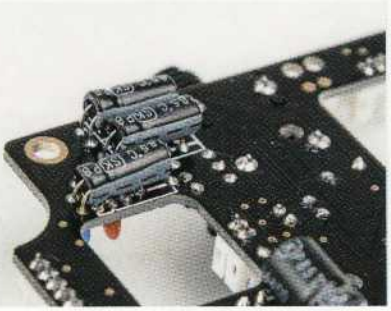
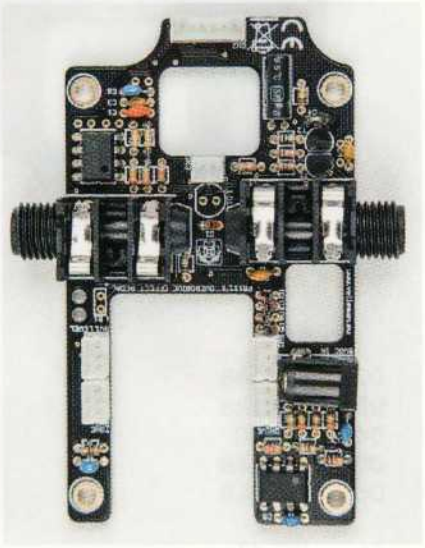


Figure 4 : ici, les prises jacks et les connecteurs pour les potentiomètres et les interrupteurs.



La résistance R19 fait office de charge afin de drainer le courant accumulé dans le condensateur C8 lorsqu'il n'y a pas de charge en sortie.

Cela permet d'éviter des « clocs » dans la chaîne d'amplification lorsque les jacks sont insérés/désinsérés. Les amplificateurs opérationnels sont de type NE5532 à faible bruit.

Terminons l'analyse du schéma électrique par l'alimentation. Le **circuit peut être alimenté** soit avec une **pile de 9 V** reliée aux points « +B1 » et « -B1 » à l'aide d'un connecteur clip à pression, soit avec une **alimentation externe stabilisée** reliée à la fiche d'alimentation J1.

L'alimentation prioritaire étant celle externe, car d'après le schéma, le positif de la pile passe par le contact de coupure de la fiche J1.

Lorsqu'il n'y a pas de connecteur dans la fiche J1 (pas d'alimentation externe), le contact interne est fermé et porte le positif de la pile (le négatif est toujours connecté) sur l'anode de la diode D4 et permet ainsi d'alimenter le montage.

À l'inverse, en présence d'un connecteur dans la fiche J1 (utilisation d'une alimentation externe), le positif de la pile est interrompu et l'anode de D4 est alimentée par le contact central de la prise d'alimentation, c'est-à-dire par alimentation externe.

Réalisation pratique

Nous venons d'étudier le fonctionnement de la pédale multi-effets, nous pouvons maintenant passer à sa construction. Dans un premier temps, il faut réaliser le circuit imprimé double face sur lequel tous les composants seront soudés.

Ensuite, il sera installé dans un boîtier métallique spécial, dont le couvercle devra faire apparaître l'interrupteur, la LED, les trois potentiomètres et le commutateur fuzz.

Pour graver le circuit imprimé, vous devez d'abord télécharger sur notre site web dans le sommaire détaillé de la revue les typons des deux faces du circuit.

Une fois gravé et percé, vous pouvez passer à l'assemblage des composants. Comme d'habitude commencez par souder les composants ayant un profil bas, c'est-à-dire les résistances et les diodes en faisant attention à la polarité indiquée et en les **disposant verticalement** comme le montre la **figure 1**.

Les connexions

Notre pédale multi-effets se connecte entre une guitare électrique et un amplificateur ou une table de mixage (en haut du dessin) ou en série avec d'autres effets, par exemple l'effet trémolo (en bas du dessin).

Montez aussi verticalement la LED en ajustant la longueur des pattes pour qu'elle puisse arriver à fleur du couvercle du boîtier.

Continuez en soudant les supports des circuits intégrés DIP à 2 x 4 broches pour les amplificateurs opérationnels NE5532. **Leurs repères détrompeurs** en forme de « **U** » **doivent être orientés comme indiqué dans le plan de câblage** des composants.

Soudez ensuite les condensateurs non polarisés puis les électrolytiques en respectant la polarité et en les disposant en hauteur pour ensuite plier leurs pattes à 90° afin de les souder horizontalement sur le circuit imprimé (voir la figure 2).

Cette astuce permet de rendre le circuit aussi mince que possible pour un meilleur positionnement dans le boîtier.

Veillez à ce que les condensateurs électrolytiques **C2**, **C5**, **C7** et **C8 soient montés du côté de la face inférieure** du circuit imprimé (côté soudures), comme indiqué en figure 3.

Continuez avec les transistors, leurs méplats sont en face l'un de l'autre, puis montez la prise d'alimentation. Soudez ensuite les deux prises jacks de 6,35 mm. Soudez enfin les 3 potentiomètres, le triple inverseur SW2 (effet) et l'interrupteur SW1 (distorsion) qui doivent d'abord être montés sur le couvercle et reliés au circuit imprimé à l'aide de fils. Reportez-vous à la figure 4 et au plan de câblage.

Vous avez la possibilité de relier les potentiomètres, l'inverseur et l'interrupteur à l'aide des connecteurs JST au pas de 2,54 mm, ce qui permet un montage et un démontage plus faciles de la pédale d'effets.

Une fois le montage terminé, connectez une pile de 9 V et profitez de la pédale multi-effets, en évitant toutefois de déranger les voisins !

Ci-dessus la pédale d'effet trémolo décrite dans le numéro 141.

Kit complet mini RetroPie avec moniteur (sans boîtier)

Kit original à assembler, vous permettant de réaliser un mini coffret de jeux vidéo d'arcade pour un seul joueur, comme ceux présents dans les années 80 et 90 dans les bars et les salles de jeux. Le système est basé sur un RaspberryPi 3 avec un moniteur couleur de 7''. Il utilise la dernière version de la distribution RetroPie, dérivée de Debian Jessie, avec laquelle vous pouvez émuler différents types de consoles, y compris Nintendo, Game Boy, MAME, Sega Master System, Amiga, Commodore, etc. Grâce à ses dimensions contenues (250 mm x250 mm x 250 mm), il est possible de l'utiliser n'importe où.

Réf.: MINIRETROPIE
Prix: 192,00€

Le kit comprend un RaspberryPi 3, une carte MAMe RetroPie, un moniteur couleur 7'', une carte SD de 8 Go avec le système RetroPie déjà installé, deux haut-parleurs, 8 boutons d'arcade, un joystick, un adaptateur secteur et tous les pièces nécessaires pour compléter et utiliser immédiatement la station de jeux vidéo arcade.

Attention le boîtier réalisé avec une imprimante 3D est disponible séparément.

Boîtier réalisé avec une imprimante 3D pour le kit mini RetroPie

Boîtier en plastique fabriqué avec une imprimante 3D, à utiliser avec le kit mini RetroPie. Vous pouvez choisir la couleur du PLA parmi les suivantes : noir, blanc, gris, rouge, orange, jaune, rose, vert, bleu.

Réf.: CASEMINICABINET
Prix: 169,00€

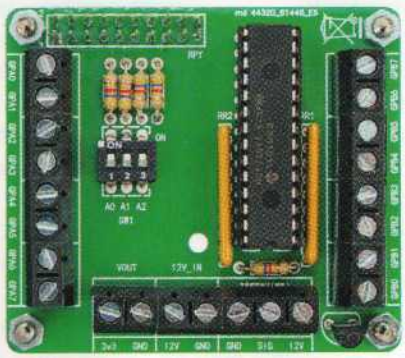
Header Marquee

Front Panel

Bezel Sticker

Carte MAME RetroPie pour RaspberryPi

Carte MAME en kit basée sur le circuit MCP23017. Elle permet la connexion entre un joystick avec un maximum de 12 boutons directement sur un RaspberryPi. Elle communique avec le RaspberryPi via le bus I2C en envoyant l'état des boutons et des contacts du joystick. La carte est vendue en kit à monter.



Réf.: ET1199K
Prix: 12,90€

COMELEC
CD 908 - 13720 BELCODÈNE Tél. : 04 42 70 63 90 Fax : 04 42 70 63 95 www.comelec.fr

Kit complet RetroPie 1 & 2 joueurs sans moniteur et sans boîtier

Ce kit permet de construire une station de jeu vidéo d'arcade pour un joueur, comme ceux présents dans les années 80 dans les bars. Le système est basé sur un RaspberryPi 3 et utilise comme logiciel RetroPie dérivé de Debian Jessie, avec lequel vous pouvez émuler différents types de consoles, y compris Nintendo, Game Boy, MAME, Sega Master System, Amiga, Commodore, etc. Le kit comprend toutes les pièces nécessaires à la fabrication de la station de jeu vidéo d'arcade à l'exception du moniteur et du coffret. Vous aurez donc besoin d'un moniteur avec une entrée HDMI ou un adaptateur HDMI/VGA et de fabriquer le coffret. **Les plans de ce dernier** sont disponibles sur le site **www.electroniquemagazine.com**. Vous pouvez ajouter en supplément le monnayeur (pas obligatoire pour le fonctionnement du jeu).

Le kit comprend pour un seul joueur :

- 1 RaspberryPi 3 ;
- 2 boutons d'arcade rouge ;
- 2 boutons d'arcade jaune ;
- 2 boutons d'arcade vert ;
- 1 bouton d'arcade bleu ;
- 1 bouton d'arcade blanc avec le symbole d'un joueur ;
- 1 bouton d'arcade noir ;
- 1 manette de jeu rose 4 axes ;
- 1 carte MAME pour le RaspberryPi 3 ;
- 30 cosses faston femelles 5 mm ;
- 1 câble à 8 pôles de 2,5 mètres pour le câblage des boutons ;
- 1 carte SD de 8 Go ;
- 1 câble HDMI de 70 cm ;
- 1 câble adaptateur micro USB ;
- 1 alimentation 5 V 2 A ;
- 1 câble FTP de 1 m CAT5E RJ45.

Réf.: RETROPIEKIT_N
Prix: 120,50€

Pour un second joueur, il faut ajouter les composants suivants :

- 2 boutons d'arcade rouge ;
- 2 boutons d'arcade jaune ;
- 2 boutons d'arcade vert ;
- 1 bouton d'arcade bleu ;
- 1 bouton d'arcade blanc avec le symbole d'un joueur ;
- 1 bouton d'arcade noir ;
- 1 manette de jeu rose 4 axes ;
- 1 carte MAME pour le RaspberryPi 3 ;
- 100 cosses faston femelles 5 mm.

Monnayeur programmable (6 types de pièces)

Monnayeur entièrement programmable capable d'identifier en analysant l'épaisseur, le diamètre et le temps de chute, jusqu'à 6 types de pièces différentes. Il dispose d'un bouton de restitution des pièces. Il dispose d'une sortie série ou par impulsion avec la possibilité de sélectionner la vitesse de transmission. Il est doté de 2 boutons et d'un afficheur à 7 segments pour la programmation. Après avoir programmé le monnayeur, insérez simplement une pièce et lisez la sortie série pour trouver la valeur sous la forme d'octets binaires. Tout ce dont vous avez besoin pour faire fonctionner le monnayeur sont des pièces de monnaie, une alimentation 12 VDC et un récipient pour entreposer les pièces acceptées.

Caractéristiques techniques :

- reconnaît jusqu'à 6 types de pièces différents ;
- accepte des pièces de diamètre compris entre 17 mm et 30,5 mm et d'épaisseur comprise entre 1,25 mm et 3,2 mm ;
- sortie série ou par impulsion avec possibilité de sélectionner le Baud Rate ;
- possibilité de rejeter les pièces insérées ;
- connecteur industriel standard à 10 broches pour port parallèle ;
- connecteur à 5 broches pour le port série ;
- alimentation 12 VDC 50 mA en mode veille, 450 mA au moment de l'insertion de la pièce ;
- bouton de déblocage en cas de bourrage de pièce ;
- température de fonctionnement de 0 ° C à + 50 ° C.

Réf.: GETTON1
Prix: 71,30€

Installé dans un système électrique domestique d'une puissance maximale de 6 kW, l'anti-blackout permet en cas de dépassement de la puissance maximale, de désactiver un ou plusieurs appareils selon une séquence que vous aurez au préalable déterminée (programmée). Seconde et dernière partie : le récepteur (actionneur).

STOP AU BLACKOUT!

Deuxième partie

de Roberto Prestianni

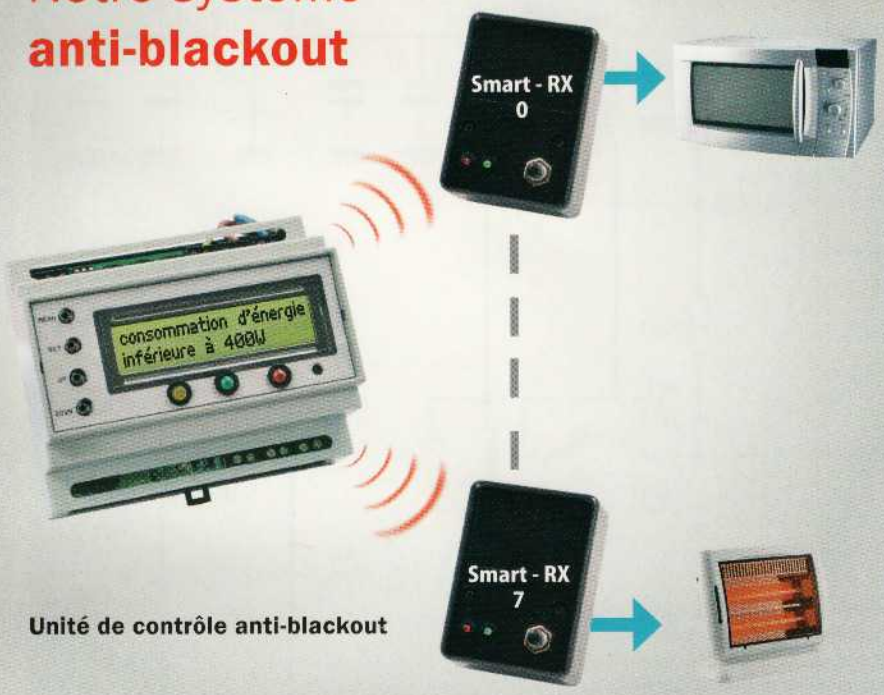
Devoir faire attention au nombre d'appareils que nous connectons à l'ensemble du circuit électrique de la maison (ou appartement), est l'une des choses les plus ennuyeuses. Pensez à vérifier si vous pouvez repasser ou utiliser le four micro-ondes pendant que vous écoutez de la musique sur la chaîne Hi-Fi...

Pour cela, vous devez comptabiliser la consommation de chaque appareil pour savoir si la consommation maximale est atteinte, cela devient très vite fastidieux ! Pour un électricien cela peut être facile, mais imaginez une personne qui n'est pas habituée ou qui ne connait pas les systèmes électriques.

De plus, depuis l'avènement des compteurs électroniques, les installations électriques ont tendance à disjoncter plus facilement. Apparemment, ils ne supportent pas bien le dépassement de la puissance liée au contrat d'électricité.

Beaucoup d'utilisateurs se retrouvent ainsi déconnectés du réseau à cause d'un dépassement de consommation de quelques watts !

Notre système **anti-blackout**



Unité de contrôle anti-blackout

Dans de telles situations, il est très pratique de disposer d'un système capable de contrôler en temps réel la consommation de courant et d'organiser le fonctionnement des appareils afin de la maintenir dans les limites autorisées.

C'est ce que fait notre anti-blackout, il permet le contrôle à distance et donc la coupure d'un ou plusieurs appareils considérés comme secondaires, par exemple une chaîne Hi-Fi, un téléviseur, lorsque la limite de consommation est atteinte pour ensuite les reconnecter lorsqu'elle revient à une valeur acceptable. Dans le précédent numéro 143 d'Electronique et Loisirs Magazine, nous avons décrit le projet en analysant l'unité centrale du système anti-blackout.

Nous allons maintenant décrire les modules récepteurs, appelés « **Smart-Rx** », qui sont reliés aux appareils à déconnecter. Il s'agit de **récepteurs radio** conçus pour fonctionner avec l'unité centrale. Le « Smart-Rx » est un récepteur spécial radiocommandé qui fonctionne selon le principe de **codage Motorola**.

Grâce au réglage d'un cavalier, il peut être commandé en tant qu'actionneur dédié à l'anti-blackout ou être commandé par une télécommande compatible avec les codeurs de types « MC1450xx » fonctionnant avec une fréquence d'horloge de 1,7 kHz.

Selon la fonction pour laquelle le récepteur a été conçu, le relais commute l'alimentation de la charge (l'appareil) connectée au système.

La philosophie du projet est telle que le récepteur doit contrôler l'alimentation d'un ou plusieurs appareils dont vous pensez pouvoir vous passer en cas de dépassement de la puissance maximale de votre compteur.

Selon le nombre d'appareils à commander, il vous faut un nombre plus ou moins important de « Smart-Rx ». Gardez à l'esprit que l'anti-blackout peut gérer un maximum de sept niveaux de priorité.

Le récepteur est facile à installer. Il possède une prise intégrée dans le boîtier à partir de laquelle il prélève la tension secteur, et une prise volante pour relier la charge à commander à l'aide d'un relais.

Grâce à la fonction d'auto-apprentissage du codage, l'intégration d'un ou plusieurs « Smart-Rx » est rapide et facile.

De plus, comme l'**activation** et la **désactivation** d'une charge s'effectuent avec **deux codes distincts**, il ne peut pas arriver que l'appareil commandé soit éteint lorsqu'un ordre d'allumage est envoyé et vice versa.

Chaque récepteur peut cependant être utilisé comme un simple interrupteur radiocommandé de type marche/arrêt ou « ON/OFF ».

Le schéma électrique

Le circuit est basé sur l'utilisation d'un microcontrôleur **PIC16F88**, qui implémente le décodage Motorola, l'interprétation et l'exécution des différentes commandes provenant de l'unité de contrôle ainsi que l'auto-apprentissage des codes.

Lors de la phase d'auto-apprentissage, le récepteur reçoit une trame de données provenant de l'unité de contrôle anti-blackout. Le microcontrôleur extrait alors le code de base du système et le niveau de priorité qui lui est affecté.

L'alimentation est obtenue directement à partir de la tension secteur 230 VAC (la même que celle qui alimente la charge à contrôler), à travers le transformateur TR1 constitué de deux enroulements secondaires de 9 V. La tension est ainsi redressée par un redresseur double-alternance constitué de deux diodes D1 et D2.

La tension unidirectionnelle ainsi obtenue est filtrée par les condensateurs C1 et C2 puis stabilisée à 5 VDC par le régulateur de tension U1, un 78L05.

Les condensateurs C3 et C4 éliminent toute ondulation résiduelle en sortie.

La tension unidirectionnelle présente sur les cathodes de D1 et D2 a une valeur proche de 12 V, elle sert à alimenter la bobine du relais RL1 qui permettra la commutation de la charge (l'appareil) connectée en sortie.

Il est à noter que l'utilisation du transformateur augmente l'immunité du microcontrôleur aux nombreuses perturbations provenant de l'alimentation secteur 230 VAC.

Nous aurions pu utiliser un montage sans transformateur, mais plus propice aux perturbations. En effet, la commutation d'une charge fortement inductive (machine à laver, four, etc.) peut provoquer des perturbations importantes qui compromettraient le fonctionnement du PIC s'il n'y avait pas le transformateur.

En renonçant au transformateur et en optant pour une alimentation directement couplée à la tension secteur, les perturbations du réseau électrique passeraient plus facilement.

La broche RA1 du PIC est configurée en sortie de manière à piloter la LED verte LD2 qui visualise la réception des codes et certaines fonctions particulières. La broche RB4, ainsi que les autres entrées du PORTB, dispose d'une résistance de pull-up intégrée. Elle est donc connectée directement au bouton SW1.

La broche RA2, configurée en sortie, alimente le relais à travers le transistor Q1. Ce dernier a sa base connectée en série avec la LED LD1 et la résistance R1. Cette LED rouge s'allume en même temps que le relais s'enclenche, signalant la déconnexion de la charge correspondante.

La diode D3 empêche toute oscillation de la bobine du relais RL1 et C5 filtre les perturbations hautes fréquences qui pourraient se propager sur la ligne d'alimentation 5 V appliquée au PIC. Il est primordial de placer physiquement le condensateur proche du PIC.

Pour **recevoir le signal modulé** et **codé**, nous utilisons le module HF **U3**. Il s'agit d'un module économique de type **Aurel AC-RX2**. Compte tenu de la compatibilité du brochage de ce dernier avec d'autres modules HF Aurel fonctionnant en **433,92 MHz**, et compte tenu de la présence du cavalier JP4 permettant l'activation éventuelle de l'AGC, le module HF AC-RX2 peut très bien être remplacé par un autre module HF ayant de meilleures performances.

Le montage comprend **deux cavaliers** de configuration, **JP2** et **JP3**, qui permettent de définir le **mode de fonctionnement** du « **Smart-Rx** ». Le premier cavalier permet de spécifier si la commutation du relais doit être contrôlée par un ou deux boutons d'un transmetteur Motorola de type TX1CSAW ou TX2CSAW. Le second cavalier spécifie le type de fonctionnement du « Smart-Rx », c'est-à-dire si la charge est contrôlée par l'anti-blackout ou par une simple radiocommande « ON/OFF ».

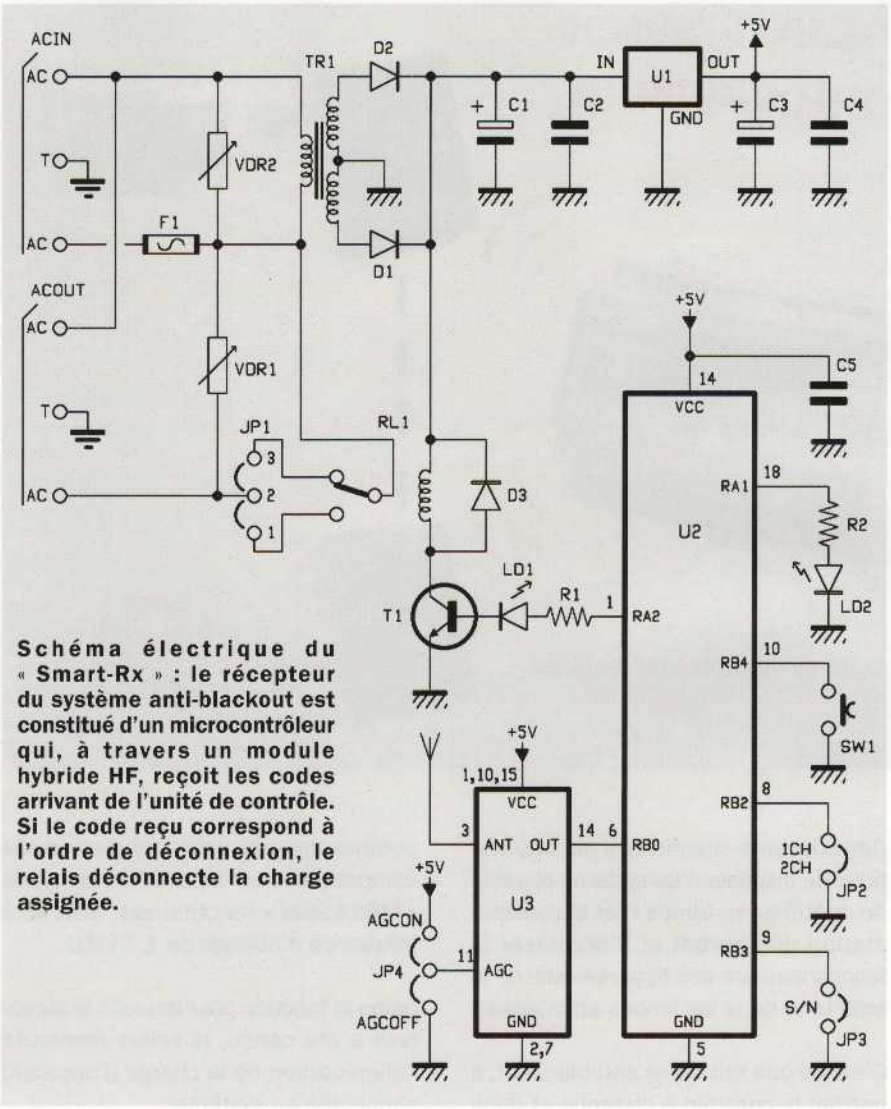


Schéma électrique du « Smart-Rx » : le récepteur du système anti-blackout est constitué d'un microcontrôleur qui, à travers un module hybride HF, reçoit les codes arrivant de l'unité de contrôle. Si le code reçu correspond à l'ordre de déconnexion, le relais déconnecte la charge assignée.

Le cavalier **JP1** permet de sélectionner le **mode de commutation du relais** (NF ou NO), qui, comme nous le verrons, dépend du mode de fonctionnement utilisé. Le tableau 1 montre la configuration des 3 cavaliers en fonction des deux modes de fonctionnement prévus.

Notons enfin la présence des varistances VDR1 et VDR2 : la première protège les contacts du relais contre les pics de tensions générés pendant les phases d'ouverture et de fermeture (du contact), notamment lorsque les charges sont de nature inductive. La seconde, placée à l'entrée du secteur, tend à éliminer les éventuelles surtensions qui peuvent apparaître sur la ligne 230 VAC.

Le firmware

Le firmware (programme) du PIC est disponible en téléchargement sur le site dans le sommaire détaillé de la revue. Vous y trouverez le projet complet avec tous les fichiers source écrits en langage C. Comme vous pouvez le voir sur la figure, le projet MPLAB ne comporte que deux fichiers « **Attuatore_Rx_file.c** » et « **pic16f87.h** », à part ceux de type « inclusion » propres au langage C et appelés à l'aide de l'instruction « **#include** ».

Le premier fichier contient la **boucle principale** « **main ()** », la routine d'interruption, la routine « **acquisition_**

Listing 4

```
if(test==0)
{
//******** Reconnaissance et mise en œuvre des commandes ********************************

if(swcode_temp[6]==OPENs && (swcode_temp[7]==ONEs || swcode_temp[7]==OPENs))
{
if(swcode_temp[7]==ONEs && swcode_temp[8]==ZEROs)          // -------->Priorité
{
        priority_print();
        DelayMs(250);
        DelayMs(250);
        DelayMs(250);
        DelayMs(250);
}

if(swcode_temp[7]==ONEs && swcode_temp[8]==ONEs)           // -------->Réactivation générale des charges
{
        out=load_anti_ON;                                  // Allume la charge
        lamp_led();                                        // 3 clignotements de la LED
}
if(swcode_temp[7]==ONEs && swcode_temp[8]==OPENs)          // -------->Désactivation générale d'urgence
{
        out=load_anti_OFF;                                 // Éteint la charge
}
if(swcode_temp[7]==OPENs && swcode_temp[8]==ZEROs)         // -------->Remplacement forcé du code de base
{
        LED=1;
        DelayMs(250);DelayMs(250);DelayMs(250);DelayMs(250);  // En attente avant l'acquisition du code
        read_base_priority(0);                             // Sans mise à jour du niveau de priorité
        LED=0;
}
}
else                                                       // Lit le niveau de priorité et active/désactive la
charge
{
if(priority == (3*swcode_temp[6] + swcode_temp[7]))        // Comparaison du niveau de priorité
{
if(swcode_temp[8]==OPENs)        out=load_anti_ON;         // Allume la charge
else if(swcode_temp[8]==ZEROs) out=load_anti_OFF;          // Éteint la charge


//********* Sauvegarde dan l'EEPROM du nouveau statut de la sortie ********************************
WREN=1;                                                    // Ecrit l'activation
write_EEPROM(EEADD_out_state,out);                         // Initialise la valeur d'écriture à l'adresse

//while(WR) continue;                                      // Attend la fin de l'écriture avant EEPROM_READ
WREN=0;                                                    // Ecrit la désactivation
}
}
}
```

code() » pour la lecture complète des **codes Motorola**, la routine « **read_code()** » qui extrait à partir du signal de sortie du module HF **un seul mot du code**, la routine « **read_base_priority()** » qui extrait le **code de base** ainsi que le **niveau de priorité** inclus dans le code, la routine « **set_freq()** » qui permet d'adapter la **fréquence d'échantillonnage des bits série entrants**, la routine « **priority_print()** » qui affiche le **niveau de priorité** assigné à l'actionneur à l'aide de plusieurs clignotements de la LED verte LD2, la routine « **write_EEPROM()** » qui mémorise les paramètres de fonctionnement et autres dans la mémoire EEPROM, ainsi que les routines « **lamp_led()** » et « **more_lamp()** » respectivement pour un **seul clignotement** et de **multiples clignotements** de la LED de signalisation.

De plus, sont définies également les **routines de retards** « **halfperiod()** » pour la fréquence d'échantillonnage du code, « **Delay100Us()** » pour la

Ceci est suivi par l'**initialisation de certains blocs de la mémoire** dans lesquels sont stockés : le code de base, le niveau de priorité, l'état initial de la ligne RA2 (commande du relais) et la valeur initiale qui définit la fréquence de lecture des codes provenant du module U3.

Ensuite, sont définies **toutes les variables utilisées** par le firmware (et donc tous les registres généraux) de types « **bit** » (bit unique), de type « **unsigned char** » (simple octet non signé), de type « **unsigned short** » (double octet non signé) ainsi que les deux tableaux de 9 éléments « unsigned char » pour la gestion des codes Motorola.

Après cela, les **lignes d'entrées/sorties** du microcontrôleur **sont configurées** et les routines du firmware sont exécutées.

génération de retards qui sont des multiples de 0,1 ms et enfin « **DelayMs()** » et « **DelayMss()** » pour la génération de retards multiples de 1 ms.

Avant d'étudier l'organigramme du firmware, examinons les premières lignes de code du fichier source « Attuatore_Rx_file.c ». Tout d'abord, nous trouvons l'inclusion des fichiers « **header** » (.h) nécessaires pour le langage C ainsi que les définitions de certaines fonctions associées au PIC (« htc.h »). Ensuite, nous trouvons les **bits de configuration** qui paramètrent l'utilisation de l'oscillateur interne (INTIO), l'activation du PWRT (Power-up Timer) et la désactivation de la ligne de reset « MCLR ».

Après cela, nous trouvons les **définitions des constantes**, c'est-à-dire les paramètres et les adresses des données de la mémoire EEPROM tels que les codes des bits à 3 états (ONE, ZERO, OPEN), les constantes de définition des retards (par exemple « wait_10sec » pour la génération d'un retard de 10 secondes), la valeur par défaut du niveau de priorité (« priority »), les limites minimales et maximales relatives à la fréquence de lecture des codes Motorola ainsi que l'adresse du premier bit du code de base mémorisé dans l'EEPROM.

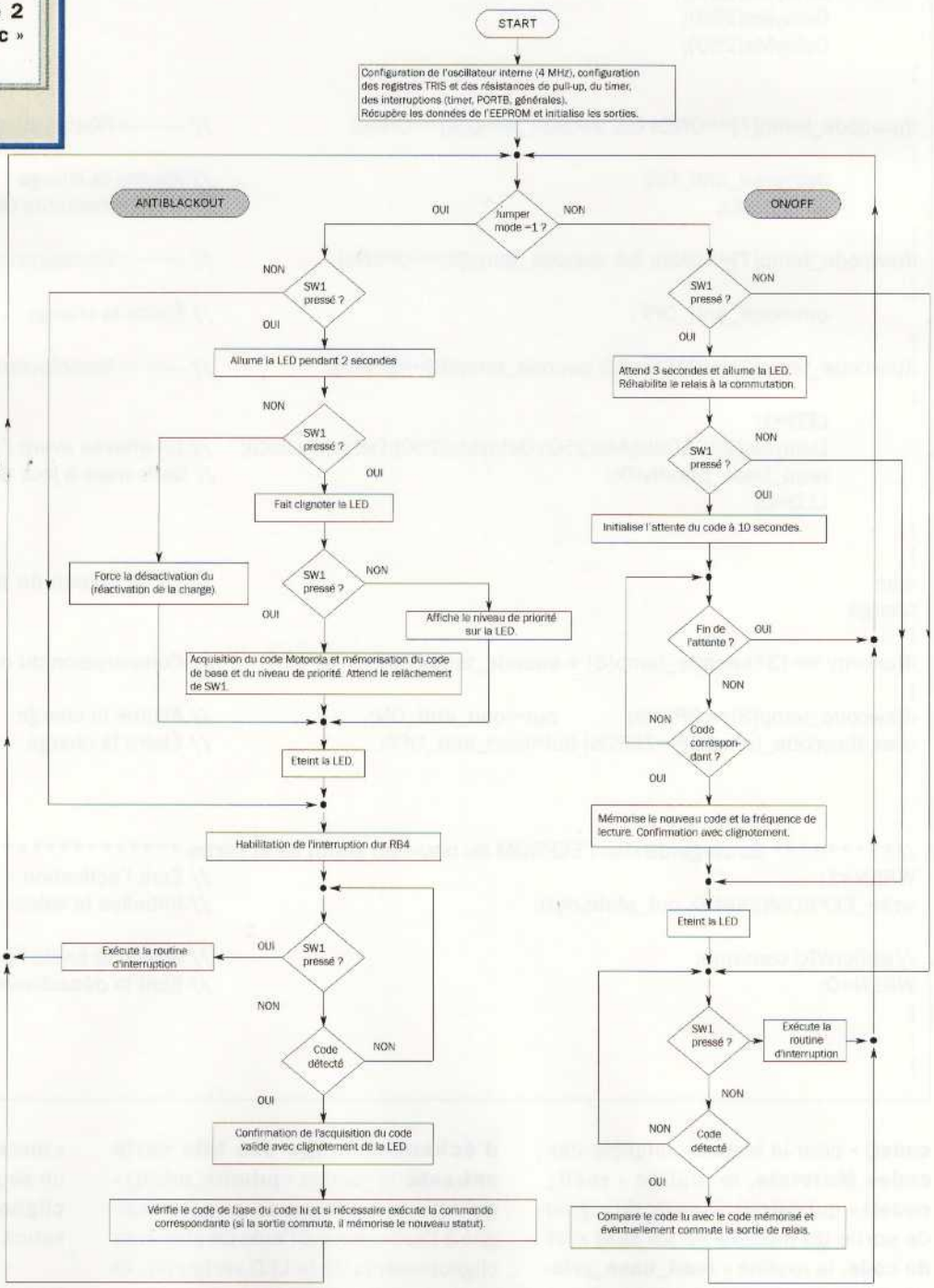


Diagramme de flux (flow-chart) : l'organigramme montre le fonctionnement du firmware et les deux modes de fonctionnement prévus après l'initialisation (RX anti-blackout ou RX télécommandé).

Si nous examinons maintenant l'organigramme (flow-chart), **lorsque le microcontrôleur est mis sous tension** et après les configurations des registres et des ports d'entrées/sorties, le firmware **teste la configuration du cavalier MODE** (JP3).

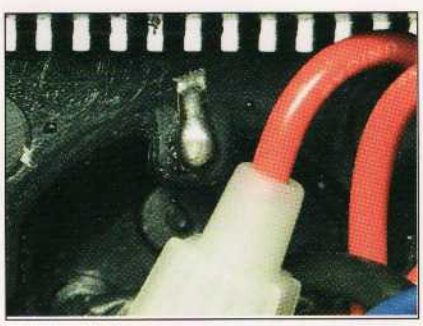
Si le cavalier est **ouvert**, la partie du programme qui implémente l'actionneur pour l'**anti-blackout** (« Smart-Rx ») sera exécutée (la ligne se trouve à un niveau logique 1 à cause de la résistance de pull-up intégrée). Si le cavalier est **fermé**, la partie du programme relative au simple fonctionnement en tant que radiocommande « **ON/OFF** » sera exécutée.

Sur le côté gauche du diagramme de flux, vous pouvez voir que l'état de SW1 est d'abord vérifié et, s'il n'est pas pressé, le programme prépare la lecture de tous les codes Motorola entrant dans le PIC. Sinon, après un bref délai, la LED LD2 s'allume et le bouton est à nouveau vérifié.

S'il est pressé, le programme reste en attente quelques secondes pour détecter un éventuel code valide et, si nécessaire, assigner un nouveau code de base ainsi que le niveau de priorité à l'actionneur correspondant.

Inversement, si SW1 n'est pas pressé, le niveau de priorité précédemment défini est affiché. Dans ce cas, la LED verte LD2 clignote pendant un court instant si le niveau de priorité est égal à « 0 », sinon elle clignote un nombre de fois égal au niveau de priorité.

Ensuite, le programme effectue la routine de réception de toutes les commandes et leur exécution.



Connexion de la broche de terre dans le boîtier.

Si SW1 est pressé pendant cette routine, le programme part en interruption pour revenir au début du cycle principal.

D'autre part, si un code Motorola est détecté et comprend le code de base du système, alors la commande correspondante est exécutée. Le listing 4 montre cette partie particulière du programme.

Nous allons maintenant examiner la partie à droite de l'organigramme, c'est-à-dire correspondant à un fonctionnement en simple radiocommande « ON/OFF ».

Ici aussi, le bouton SW1 est testé une première fois et, s'il n'est pas pressé, toute la partie du programme liée à l'auto-apprentissage du code est ignorée. Sinon, après un délai de 3 secondes environ, la LED verte s'allume et le relais est activé pour la commutation.

Si, entre-temps, le bouton SW1 est maintenu enfoncé, l'auto-apprentissage d'un nouveau code Motorola est activé pour un fonctionnement de la charge en mode « ON/OFF » (marche/arrêt) avec une télécommande compatible. Au bout de 10 secondes, le programme revient à l'exécution du cycle principal.

En bas de l'organigramme, vous pouvez voir la partie du programme qui attend continuellement un code Motorola valide. Lorsque cela se produit, le code reçu est comparé à celui en mémoire et éventuellement la sortie du relais peut être commutée.

Comme nous l'avons mentionné précédemment, le **code est constitué de 9 bits comprenant le code de base** (les 6 premiers bits) et un « **code spécifique** » de 3 bits qui représente le niveau de priorité de l'actionneur que l'unité de contrôle veut commander ainsi que son état (de l'actionneur). Le tableau 2 montre les différentes configurations du code de base (qui est commun).

Le programme commence par la vérification de la variable « test » à l'aide d'un flag (drapeau). Si ce dernier est à « 0 », cela implique que le code reçu est valide.

En examinant les tableaux, nous pouvons comprendre pourquoi tous les codes ayant le bit 7 égal à « H » (aussi appelé « OPEN ») et le bit 8 égal à « 1 » ou « H », sont des commandes spéciales compréhensibles par les « Smart-Rx ». Les autres codes ont les bits 7 et 8 qui spécifient le niveau de priorité, tandis que le bit 9 indique l'état que doit prendre l'actionneur.

Si, par exemple, l'unité de contrôle ayant un code de base égal à « 111HHH », émet le code suivant « 111HHH -10-H », cela signifie qu'elle commande à tous les récepteurs appartenant au niveau de priorité 3, l'activation de la charge du « Smart-Rx » du deuxième canal.

En examinant le code, une série de « if (...) » identifient la commande spéciale qui doit être exécutée. Une fois détecté le niveau de priorité spécifié par la centrale, le programme change l'état du relais (activation/désactivation de la charge) et le nouveau statut est mémorisé dans l'EEPROM.

Notez que les états des relais des « Smart-Rx » en mode anti-blackout uniquement sont mémorisés, même lorsque le système n'est pas alimenté.

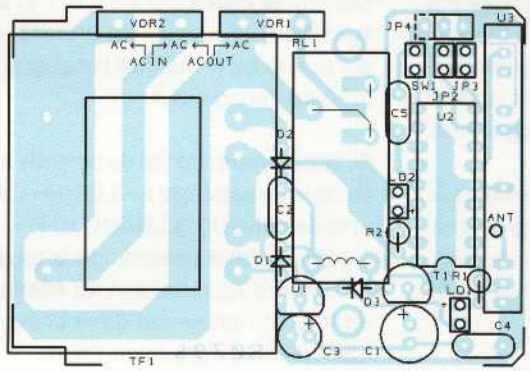
Réalisation pratique

Pour construire un ou plusieurs « Smart-Rx », vous devez d'abord télécharger les typons à l'échelle 1 et/ou les fichiers GERBER qui se trouvent sur notre site dans le sommaire détaillé de la revue en bas de page.

Après avoir gravé et percé le ou les circuits imprimés, commencez par configurer JP1, en effectuant un pont de soudure entre les points reliés à la borne « ACout » et l'autre au contact NF du relais dans le cas où vous utilisez le mode anti-blackout, sinon au contact NO du relais si vous utilisez le(s) « Smart-Rx » en mode radiocommandé « ON/OFF ».

Commencez par souder verticalement les deux résistances et les trois diodes D1, D2 et D3. Ensuite continuez avec les condensateurs non polarisés C2, C4 et C5 puis soudez les condensateurs polarisés C1 et C3 en prêtant attention à leur orientation (les + sont orientés vers le relais).

Plan de montage



Plan de câblage des composants du « Smart-Rx ».

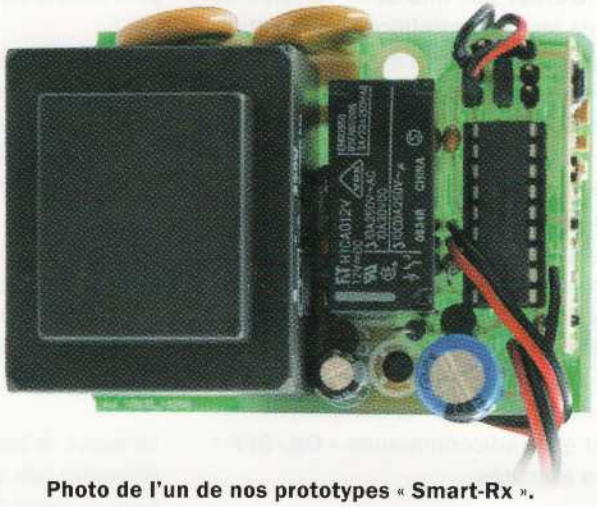


Photo de l'un de nos prototypes « Smart-Rx ».

Liste des composants

R1.....220 Ω
R2.....220 Ω

C1.....470 µF/25V électrolytique
C2.....100 nF multicouche
C3.....100 µF/25V électrolytique
C4.....100 nF multicouche
C5.....100 nF multicouche

D1.....1N4007
D2.....1N4007
D3.....1N4007

LD1....LED 5 mm rouge
LD2....LED 5 mm verte

T1 .....BC547

U1.....78L05
U2.....PIC16F88
U3.....AC-RX2 ou compatible

SW1...bouton poussoir

VDR1 .varistance 275 VAC
VDR2 .varistance 275 VAC

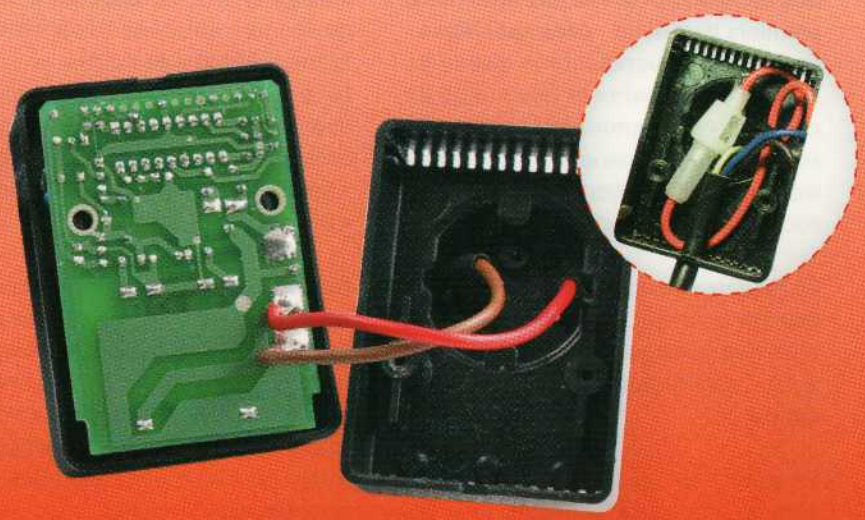
F1 .....fusible 10 A
RL1....relais 12 V H1CA012V : code commande radiosapres.fr : 790-3329
TR1....transformateur 220 VAC / 2 x 9 VAC 3 VA

Divers

Antenne (fil de cuivre émaillé de 17 cm de longueur)

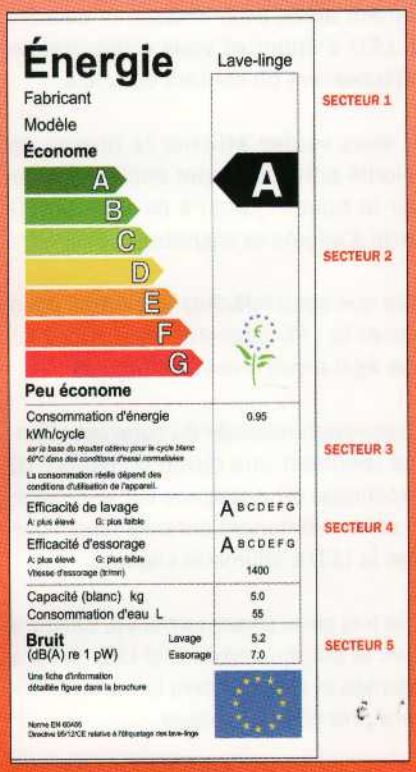
Barrette mâle 2 pôles (x2)
Barrette mâle 3 pôles (x3)
Porte fusible 10 A
Support pour circuit intégré 2 x 9 broches
Enjoliveur de panneau pour LED 5 mm
Boitier plastique PPB-11860-WT pour alimentation 52 x 78 x 38,5 mm avec prise secteur (www.farnell.fr)
Prise volante + câble tripolaire

NB : les typons des circuits imprimés à l'échelle 1 sont disponibles en téléchargement dans le sommaire détaillé de la revue.



Aux électrodes internes du boîtier sont reliés 2 morceaux de fil. Ils doivent être connectés ensuite aux points « ACin ». Si vous le souhaitez, vous pouvez insérer un porte-fusible volant, en série avec l'un des conducteurs. Entre le point commun « ACin » et le point indépendant « ACout », connectez le câble à 3 conducteurs pour la charge.

Peu de consommation, Pas de blackout !



Une façon d'éviter les pannes à la maison, causées par une consommation excessive, est certainement d'acheter des appareils à faible consommation. Cependant, il est important d'apprendre à lire les étiquettes énergétiques avant l'achat d'un appareil.

L'étiquette est une échelle de référence pour la consommation électrique des appareils ménagers et des ampoules. Elle est divisée en 7 classes, de la classe A (basse consommation) à la classe G (forte consommation), dont les valeurs limites varient pour chaque appareil électroménager.

Il existe également des appareils de classe « A+ » et « A++ », encore plus efficaces que ceux de la classe A. Différents modèles d'une même classe peuvent être comparés en fonction de la consommation d'énergie estimée pour une utilisation dans des conditions standard prédéfinies.

Cette valeur est indiquée sur l'étiquette dans la partie située en dessous de la classe d'énergie, elle est exprimée en consommation annuelle (kWh/an) ou par cycle d'utilisation (kWh/cycle).

Dans l'étiquette, le secteur 1 identifie l'appareil avec le modèle et le fabricant, le secteur 2 montre les classes d'efficacité énergétique. Le secteur 3 indique la consommation d'énergie, exprimée en kWh/an ou en kWh/cycle. Le secteur 4 fournit les spécifications des performances de l'appareil. Le secteur 5 reporte le niveau de bruit de l'appareil et indique qu'une fiche détaillée contenant les caractéristiques de l'appareil est disponible pour l'utilisateur/consommateur.

Ensuite, soudez le support du PIC, les barrettes des cavaliers, le relais, les varistances et le module HF Aurel. Son brochage est tel qu'il ne peut être inséré que dans un seul sens.

Pour l'antenne, utilisez du fil émaillé rigide de sorte qu'il puisse être orienté afin d'obtenir la meilleure réception possible. N'oubliez pas de gratter l'extrémité du fil de cuivre émaillé avant de le souder.

Les deux LED doivent être prolongées avec des fils ayant une longueur de 3,5 cm à 4 cm et, éventuellement, recouvrez les soudures avec de la gaine thermorétractable. Une fois tous les composants soudés, configurez les cavaliers en fonction du mode de fonctionnement souhaité. Pour le mode anti-blackout, laissez JP3 ouvert. Insérez le PIC dans son support en l'ayant au préalable convenablement programmé.

À ce stade, vous devez préparer le boîtier en plastique ayant la prise secteur intégrée.

Vous devez percer un trou d'environ 7 mm sur la moitié inférieure du couvercle afin de faire passer le câble à 3 conducteurs de la prise volante.

Soudez le fil de terre du câble à trois fils à la borne de terre du boîtier, puis sur l'un des fils insérez un porte-fusible. Soudez une extrémité du porte-fusible à l'une des 2 broches du boîtier puis l'autre fil à la broche restante (points « ACin », c'est-à-dire la phase et le neutre).

Reliez les fils de phase et de neutre restants aux points « ACout ». Il convient de noter que « ACin » et « ACout » ont une borne en commun.

Fixez le câble à l'intérieur du boîtier à l'aide d'un serre-câble. Il est important que le corps du porte-fusible soit logé correctement dans le boîtier afin qu'il ne gêne pas la disposition du circuit imprimé à l'intérieur du boîtier.

Percez des trous dans la face avant du boîtier de ∅ 6 mm correspondant aux 2 LED de 5 mm, il faut tenir compte de l'épaisseur des enjoliveurs. Sinon percez à 5 mm et fixez les LED avec un point de colle (sans enjoliveur).

Enfin, percez le trou correspondant au bouton, utilisez éventuellement un capuchon. Fermez le boîtier à l'aide du couvercle, les deux LED et le bouton étant montés sur le couvercle.

Si une distance (ou portée) HF plus élevée est souhaitée, le fil de l'antenne doit passer à travers un trou à réaliser près du module HF, afin de le sortir du boitier. Ce dernier doit ensuite être fermé en serrant les vis de fermeture.

Si vous ne trouvez pas de porte-fusible volant d'une capacité de 10 A ou plus, vous pouvez en utiliser un de 5 A, en prenant soin, de calibrer le fusible correctement (fusible de 5 A, soit une puissance maximale de 1 kW sous 230 VAC).

En ce qui concerne le pouvoir de coupure du relais, sa capacité maximale est de 10 A.

Tableau 1 : la configuration pour le mode anti-blackout s'effectue à l'aide de JP1. Le cavalier JP2 spécifie si l'activation et la désactivation en mode ON/OFF doivent être commandées par deux émetteurs séparés, ou par un seul. JP3, selon son état, définit le mode de fonctionnement.

Remarque : NF et NO signifient que vous devez utiliser le contact du relais normalement fermé ou normalement ouvert.

Jumper	Ouvert	Fermé	NF	NO
JP1	-	-	ANTI-BLACKOUT	ON/OFF
JP2	Double code	Code unique	-	-
JP3	ANTI-BLACKOUT	ON/OFF	-	-

Si vous souhaitez contrôler des charges plus importantes, vous devrez utiliser un relais dont les contacts peuvent supporter des puissances plus grandes.

Nous vous conseillons également d'étamer abondamment toutes les pistes de cuivre du circuit imprimé en relation avec la tension secteur.

Enfin, **par mesure de sécurité, n'ouvrez pas le boîtier du « Smart-Rx » lorsqu'il est relié à une prise secteur, même s'il n'est pas en fonctionnement**. Il y a **risque d'électrocution**.

Tests et codage

Pour tester le système, connectez simplement le récepteur « Smart-Rx » à une prise électrique et transmettez n'importe quel code Motorola à l'aide d'une télécommande compatible. Le bon fonctionnement de l'appareil sera confirmé par le clignotement de la LED verte.

À ce stade, il ne reste plus qu'à coder le « Smart-Rx » et à le préparer à fonctionner en conjonction avec l'anti-blackout ou avec une télécommande en mode « ON/OFF ».

Fonctionnement avec le « Smart-Rx »

Pour forcer l'activation de la charge, maintenez le bouton enfoncé jusqu'à ce que la LED verte s'allume. Immédiatement après avoir relâché le bouton, la LED s'éteint et vous entendrez le relâchement du contact du relais.

Si vous voulez afficher le niveau de priorité précédemment entré, appuyez sur le bouton jusqu'à ce que la LED verte s'allume et clignote.

Dès que vous relâchez le bouton, vous verrez la LED clignoter un nombre de fois égal à son niveau de priorité.

L'auto-apprentissage du code est effectué (pendant une durée d'environ 10 secondes) en appuyant sur le bouton et en le maintenant enfoncé jusqu'à ce que la LED s'allume et clignote.

Une fois cette phase terminée, toujours avec le bouton enfoncé, la LED restera allumée et à ce moment le « Smart-Rx » sera prêt pour le codage.

Le codage du système

Les commandes transmises par l'unité de contrôle anti-blackout sont des trames standard de type codage Motorola MC14502XX, chacune contenant l'état des 9 bits à trois états. Dans notre système, nous avons prévu que les 6 premiers bits constituent le code de base, qui est identique pour tous les récepteurs et l'identification de l'unité de contrôle, tandis que les 3 bits restants définissent le niveau de priorité et d'autres commandes spéciales.

Tableau 2

	Priorité	Bit 1 à 6	Bit 7	Bit 8	Bit 9
Priorité	0	Code de base	0	0	0/1/OPEN
Priorité	1	Code de base	0	1	0/1/OPEN
Priorité	2	Code de base	0	OPEN	0/1/OPEN
Priorité	3	Code de base	1	0	0/1/OPEN
Priorité	4	Code de base	1	1	0/1/OPEN
Priorité	5	Code de base	1	OPEN	0/1/OPEN
Priorité	6	Code de base	OPEN	0	0/1/OPEN
Code spécial	Priorité	Code de base	OPEN	1	0
Code spécial	Active tout	Code de base	OPEN	1	1
Code spécial	Désactive tout	Code de base	OPEN	1	OPEN
Code spécial	Change le code de base	Code de base	OPEN	OPEN	0
Code spécial	Change le code de base	Code de base	OPEN	OPEN	1
Code spécial	Séquence de codage	Code de base	OPEN	OPEN	OPEN

Le Tableau 2 résume l'état des 9 bits concernant les commandes correspondant à la désactivation des récepteurs associés aux priorités 0 à 6 et des commandes spéciales telles que l'activation ou la désactivation simultanée de tous les récepteurs. Le Tableau 3 résume les 9 combinaisons correspondant à la variation des 3 derniers bits.

Tableau 3

Priorité	Bit 1 à 6	Bit 7	Bit 8	Bit 9
0	Code de base	0	0	0/1/OPEN
1	Code de base	0	1	0/1/OPEN
2	Code de base	0	OPEN	0/1/OPEN
3	Code de base	1	0	0/1/OPEN
4	Code de base	1	1	0/1/OPEN
5	Code de base	1	OPEN	0/1/OPEN
6	Code de base	OPEN	0	0/1/OPEN
Réservé	Code de base	OPEN	1	0/1/OPEN
Réservé	Code de base	OPEN	OPEN	0/1/OPEN

Ensuite, relâchez le bouton et transmettez le code.

Rappelez-vous que les acquisitions du code de base et du niveau de priorité s'effectuent en détectant un seul code, qui peut être émis par l'unité de contrôle ou par n'importe quel émetteur Motorola.

Si vous souhaitez utiliser l'unité de contrôle et en supposant que vous l'ayez déjà configuré avec le code de base choisi pour le système, au moyen de la commande 7 vous devrez entrer le niveau de priorité que vous souhaitez attribuer au « Smart-Rx ».

Immédiatement après, en utilisant la commande 12, l'unité de contrôle émettra en continu la séquence de codage complète avec un code de base et un niveau de priorité défini (reportez-vous aux tableaux 2 et 3).

Le clignotement de la LED jaune confirmera l'émission du code désiré et, à ce stade, vous devrez placer le « Smart-Rx » en mode auto-apprentissage. Une fois le code lu et mémorisé, la LED verte émettra un flash.

Si vous souhaitez utiliser un transmetteur commun de type Motorola, vous devrez observer quelques précautions. Pour tout émetteur que vous utilisez, référez-vous au Tableau 3.

Dans le cas de télécommandes de types **TX1CSAW** et **TX2CSAW**, vous devez effectuer une **configuration à l'aide des DIP-switch**, ou par **auto-apprentissage** dans le cas d'un émetteur **TX-4C**.

Le dernier bit (le neuvième) est sans importance car son but est d'encoder le « Smart-Rx », donc ne vous inquiétez pas si certaines télécommandes ne comportent que 8 commutateurs DIP-switch.

Les émetteurs à autoapprentissage doivent être encodés avec les 9 bits, vous pouvez affecter l'une des trois valeurs possibles « 1 », « 0 », « H » au 9ième bit.

Fonctionnement avec une télécommande « ON/OFF »

Pour utiliser un « Smart-Rx » avec une télécommande, vous devez d'abord définir le code de l'émetteur que vous souhaitez utiliser (si vous utilisez un émetteur avec un codeur à PIC, vous pouvez affecter n'importe quelle valeur au 9ième bit).

Après avoir effectué cela, insérez le « Smart-Rx » dans une prise et appuyez sur le bouton pendant environ 3 secondes, jusqu'à ce que la LED verte clignote.

À la fin de l'attente, la même LED restera allumée et seulement alors vous pourrez relâcher le bouton.

Le récepteur a donc activé le mode auto-apprentissage, vous devez donc maintenant émettre le code de commande avec l'émetteur pour le fonctionnement en « ON/OFF » (dans un délai maximum de 10 secondes). L'actionneur vous donnera une confirmation de la

Avec les télécommandes standard

Tableau 4

Priorité	Bit 1 à 6	Bit 7	Bit 8	Bit 9
0	Code de base	0	0	1
1	Code de base	0	1	1
2	Code de base	0	OPEN	1
3	Code de base	1	0	1
4	Code de base	1	1	1
5	Code de base	1	OPEN	1
6	Code de base	OPEN	0	1

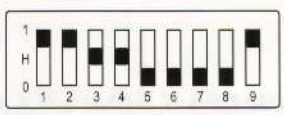
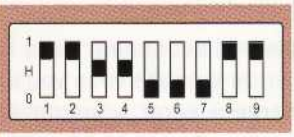
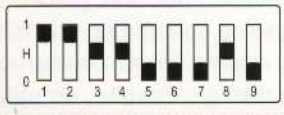
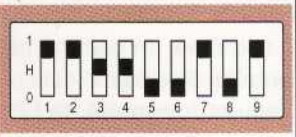
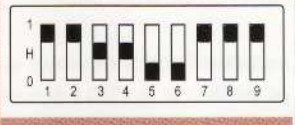
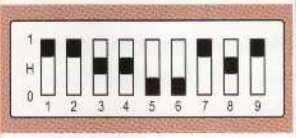
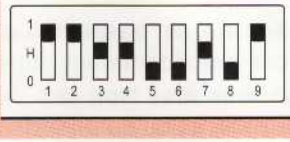
Comme l'anti-blackout émet des commandes contenant les codes Motorola de la série MC1450XX, rien n'interdit d'utiliser les récepteurs « Smart-Rx » avec des télécommandes fonctionnant avec un encodage de type MC145028 avec une fréquence d'horloge de 1,7 kHz.

Il est aussi possible de piloter un « Smart-Rx » configuré pour fonctionner avec un émetteur ayant un codage de type MC145026, fonctionnant à 1,7 kHz.

Dans le premier cas, les DIP-switch de la télécommande doivent être configurés pour émettre des codes correspondant aux 6 premiers bits du code de base du système (définis dans menu) et les 3 bits restants sur les valeurs de priorité choisies.

Dans le second cas, les 6 premiers DIP-switch de l'émetteur doivent être réglés comme souhaité et les 3 derniers doivent être configurés selon les 7 combinaisons prévues pour les priorités. Vous devez effectuer l'apprentissage des « Smart-Rx ».

Nous illustrons ci-contre la configuration des DIP-switch pour les 7 priorités possibles. Comme vous le voyez, les 6 premiers sont fixes et les 3 derniers variables.

Configuration des DIP	Priorité
	0
	1
	2
	3
	4
	5
	6

mémorisation du code par un bref clignotement de la LED verte.

Si vous avez laissé **JP2 ouvert**, alors avec le bouton gauche d'une télécommande de type TX2CSAW vous allumez la charge, tandis qu'avec le bouton droit vous l'éteignez.

Notez qu'en **utilisant une télécommande avec 2 boutons** (2 canaux), vous devez **associer à un bouton** le code « **ON** » à l'aide du **bit 9** positionné sur **1** (activation) et à l'**autre bouton** le code « **OFF** », en utilisant un codage similaire au précédent mais avec le bit 9 placé à **0**.

Si **JP2 est fermé**, avec **un seul bouton** vous commandez à la fois l'activation et la désactivation (**ON/OFF**). Dans ce cas, il suffira d'utiliser un seul code pour les deux types d'émetteur (1 et 2 boutons).

Le Tableau 4 montre un exemple de configuration des commutateurs DIP-switch pour l'attribution du code de base « 11HH00 » aux 7 niveaux de priorité (le 9ième bit, comme expliqué précédemment, doit être positionné sur « 1 »).

Installation des « Smart-Rx »

Avant d'installer l'actionneur dans la prise à partir de laquelle vous voulez relier la charge à gérer, vous devez d'abord l'encoder. Pour cela, placez-le près de l'unité de contrôle et, une fois connecté au réseau, suivez les instructions sur le codage. Ensuite, en utilisant la commande 12, faites-en sorte que le code soit émis en continu par l'unité de contrôle. Si le signal est correctement reçu et interprété, la LED verte clignotera continuellement.

À ce stade, installez l'actionneur sur sa prise finale et vérifiez si la LED clignote toujours. Si c'est le cas, vous devrez orienter l'antenne aussi bien que possible pour assurer une réception correcte du signal.

Si les choses ne se passent pas comme décrit précédemment, cela signifie que vous êtes trop loin de l'unité de contrôle et/ou qu'il y a trop d'obstacles qui réduisent excessivement la puissance du signal HF.

Si vous pensez que l'emplacement du « Smart-Rx » est dans un endroit éloigné et donc difficile à atteindre par le signal HF, vous pouvez utiliser un module HF ayant une plus grande sensibilité que celle du module AC-RX2.

Un test de fonctionnement du système peut être effectué en configurant l'unité de contrôle avec une limite de puissance de, par exemple, 1 000 W et en appliquant une charge d'une puissance supérieure (1200 W ou plus) à l'actionneur.

Si le niveau de priorité du « Smart-Rx » est de 3, vous verrez l'unité de contrôle détecter la surcharge et désactiver les niveaux 0-1-2-3 en quelques secondes.

Ce dernier, en effet, sera réactivé toutes les 30 secondes de sorte que l'unité de contrôle vérifie si la surcharge est toujours présente et, après environ 3 secondes, la même charge sera à nouveau coupée. Cela se produit indéfiniment tant que la surcharge persiste.

Un autre test que vous pouvez effectuer immédiatement est le comportement lors d'une surcharge incontrôlée. En déconnectant tous les actionneurs, connectez la charge de 1200 W directement à une prise de courant.

L'unité de contrôle désactivera tous les niveaux possibles, après quoi elle signalera l'impossibilité de ramener la consommation à un niveau inférieur au maximum défini, jusqu'à ce que l'intervention du relais interne désactive complètement l'installation électrique via le compteur (fonction optionnelle).

Dans ce cas également, après environ 30 secondes, l'unité de contrôle testera si la surcharge est toujours présente et décidera de réactiver tous les niveaux mis à « OFF » dès que possible, ou isolera de nouveau l'installation électrique.

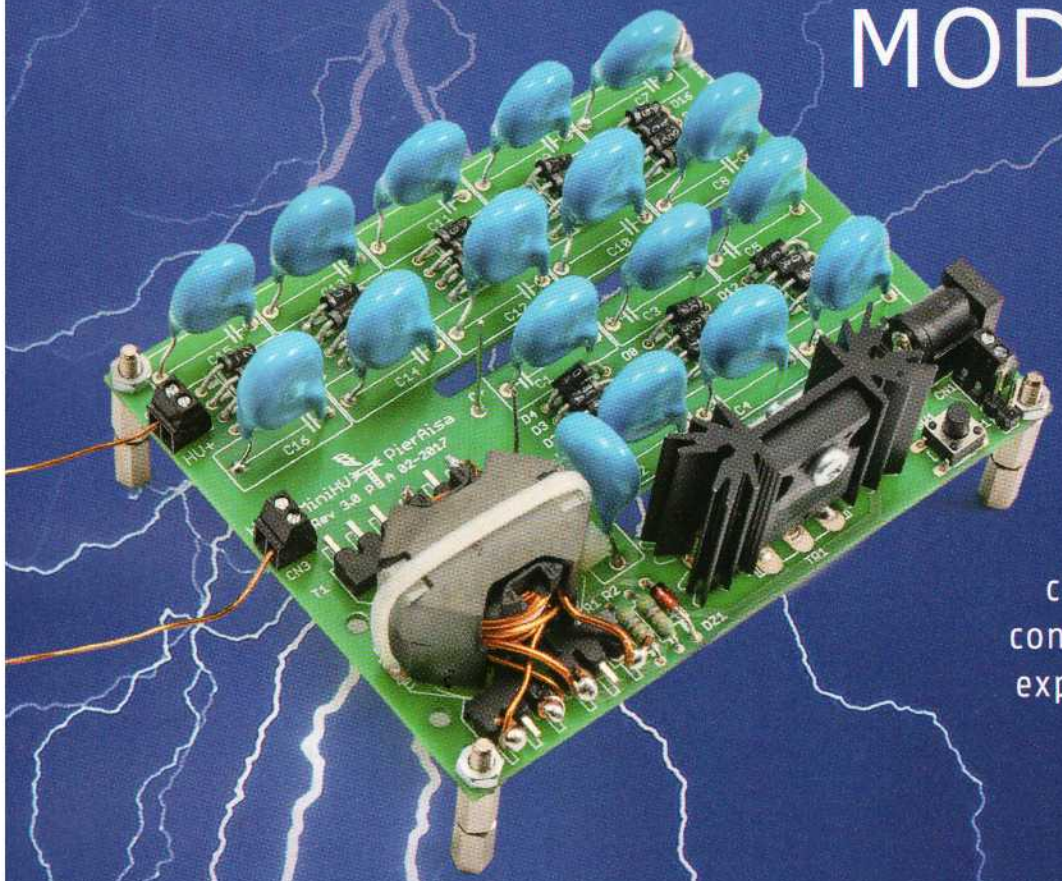
Configuration initiale

Une fois les récepteurs terminés, le système a besoin de réglages initiaux pour intervenir au bon moment et contrôler de manière adéquate tous les actionneurs dispersés dans l'appartement ou la maison. La séquence recommandée est celle décrite ci-après :

1. **Entrez la puissance maximale autorisée** : en utilisant la commande 3, entrez la puissance maximale au-delà de laquelle le système interviendra dans le contrôle de la surcharge, en désactivant les actionneurs d'un ou plusieurs niveaux de priorité. Si, par exemple, l'alimentation électrique a une limite de puissance nominale de 3 kW, cette même valeur peut être la limite pour l'unité de contrôle. Cependant, il est conseillé d'effectuer quelques tests pour s'assurer que ce réglage garantisse à 100 % que le compteur ne disjoncte pas lors d'une situation de surcharge réelle. La valeur par défaut est 6 kW ;
2. **Entrez le niveau de priorité le plus élevé** : en utilisant la commande 4, entrez le niveau de priorité le plus élevé (de 1 à 6) géré par l'anti-blackout. Les niveaux, au maximum, sont au nombre de 7, mais ce nombre peut être réduit en fonction de vos besoins. En effet, afin d'éviter que le système perde du temps à désactiver des niveaux inutilisés (avec le risque de disjonction du compteur électrique), il est nécessaire que chacun d'entre eux, s'il est sur OFF, désactive efficacement un ou plusieurs appareils. La valeur par défaut est 6 ;
3. **Entrez le code de base** : les 6 premiers bits du codage sont réservés au code de base, c'est-à-dire le code d'identification du système anti-blackout (729 combinaisons sont possibles, pour modifier le codage utilisez la commande 7). Le code par défaut est « HHHHHH » ;
4. **Activation du buzzer** : en utilisant la commande 10, il est possible d'activer ou de désactiver le buzzer. La valeur par défaut est « ON », le buzzer signale la condition de surcharge lorsque la désactivation de tous les niveaux de priorité ne permet pas de faire descendre la consommation en dessous de la puissance maximale définie.

GÉNÉRATEUR DE HAUTE TENSION MODULAIRE

de Pier Alessandro Aisa



Entrons facilement dans le monde de la haute tension et en toute sécurité, grâce à ce générateur modulaire conçu pour de nombreuses expériences intéressantes.

Réaliser des expériences avec la haute tension est quelque chose de vraiment fascinant, car les différences de potentiel très élevées (nous parlons de plusieurs milliers de volts) déclenchent divers phénomènes dans l'air ambiant, allant de la décharge d'arcs électriques à la création de halos brillants (visibles dans l'obscurité par effet d'ionisation de l'air ambient) près des conducteurs et encore plus aux niveau des pointes, jusqu'à ce que vous arriviez à l'éclairage apparemment inexpliqué des tubes au néon placés à proximité du générateur.

Malheureusement, travailler avec des milliers de volts n'est pas identique à la manipulation de circuits alimentés en basse tension (5 V à 12 V). Cela impose donc des précautions particulières afin d'éviter d'avoir des brûlures et/ou s'électrocuter.

Le projet proposé dans cet article s'adresse à tous ceux qui sont intrigués et/ou fascinés par la haute tension et tous les phénomènes qui y sont liés tels que les décharges électriques, l'effet corona, le vent ionique, le principe de pointe et l'ionisation de l'air, mais qui pour des raisons de peur ou de sécurité ne les ont jamais mis en œuvre.

Notre carte électronique, que nous avons appelée « **MiniHV** », peut être utilisée comme un laboratoire d'expérimentation afin de se familiariser avec la haute tension et comprendre ses effets dans l'environnement.

Vous pourrez ainsi apprendre et observer les concepts liés à la quantité de charge, au champ électrique, à l'isolation électrique, et cela sans prendre de risques car la puissance pouvant être produite par le montage reste limitée.

Dans notre vie quotidienne, les hautes tensions sont en réalité très présentes et proches de nous. Il suffit de porter une paire de chaussures à semelles en caoutchouc et de marcher sur le tapis de la maison, ou lorsque nous approchons la clé de la portière de la voiture ou encore quand nous touchons la portière après avoir roulé, nous apercevons un petit arc électrique ou ressentons une décharge.

Dans ce cas, les tensions peuvent atteindre 20 000 V et plus, mais nous ne ressentons qu'une gêne ou une légère douleur lorsque nous entrons en contact avec un arc électrique, car malgré la haute tension, le courant de décharge est minime, de l'ordre de quelques microampères.

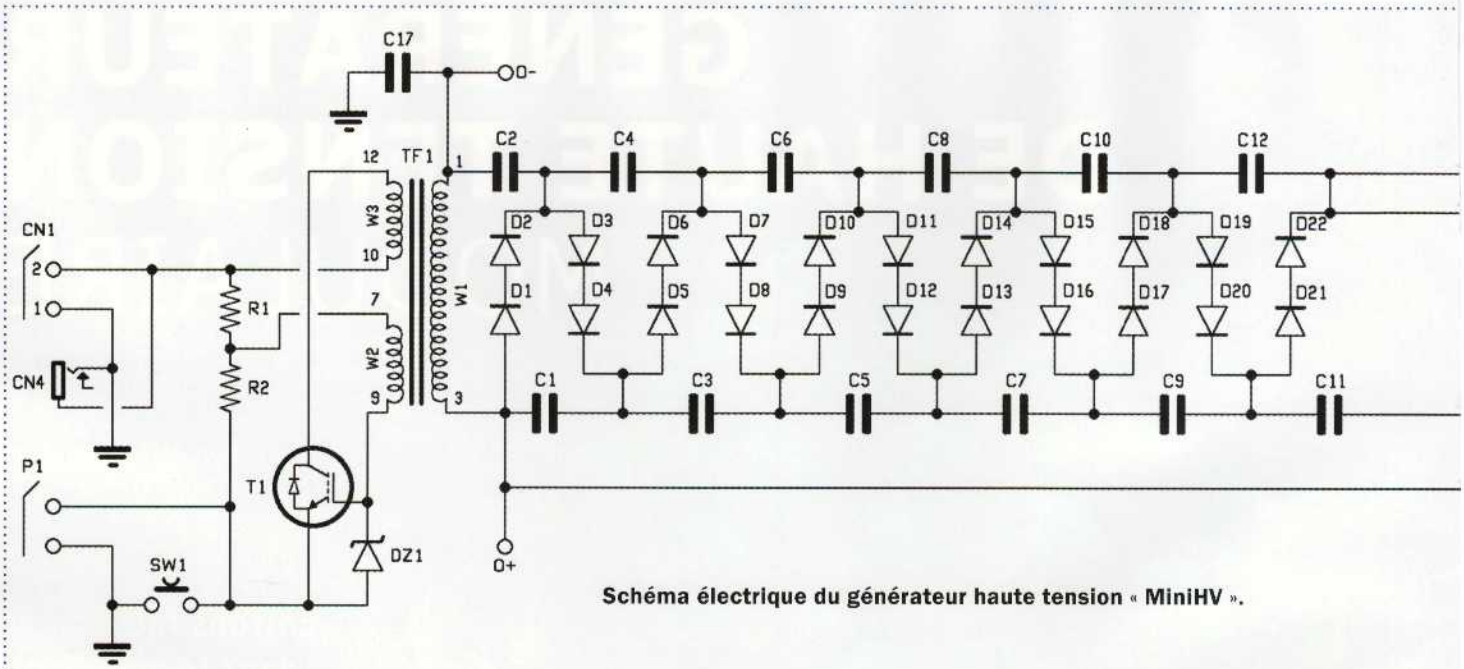


Schéma électrique du générateur haute tension « MiniHV ».

Le cas de la foudre est très différent, car en plus des tensions très élevées qui peuvent facilement atteindre un million de volts, la décharge électrique (le courant) est très élevée et atteint des valeurs destructrices et mortelles pour l'homme (quelques centaines d'ampères).

Le projet que nous vous soumettons appartient à la première catégorie, c'est-à-dire qu'il permet de générer des tensions élevées, mais avec un faible courant disponible.

Dans tous les cas, **nous vous invitons à lire les instructions de l'encadré intitulé « Quelques précautions » avant de vous servir du montage**, afin de ne pas entrer en contact direct avec la haute tension et de ne pas prendre de choc électrique. D'autre part, **nous déconseillons l'utilisation de ce montage aux personnes ayant des faiblesses cardiaques ou disposant d'un pacemaker**.

Le circuit proposé permet de générer une haute tension continue sur des électrodes et de découvrir le charme des phénomènes physiques suivants :

- l'**arc électrique en continu** ;
- l'**effet corona** et la **production d'ozone** ;
- l'**effet de pointe** et le vent ionique ;
- l'**ionisation de l'air**.

Principe de fonctionnement

Le but que nous nous sommes fixés pour concevoir ce générateur haute tension était de fabriquer une carte électronique compacte qui soit modulaire, de sorte que la tension de sortie puisse être augmentée en ajoutant plusieurs cartes en série au niveau de la sortie.

À partir d'une basse tension de seulement 12 V, avec ce montage, vous pouvez dépasser 50000 V en utilisant les trois principes suivants :

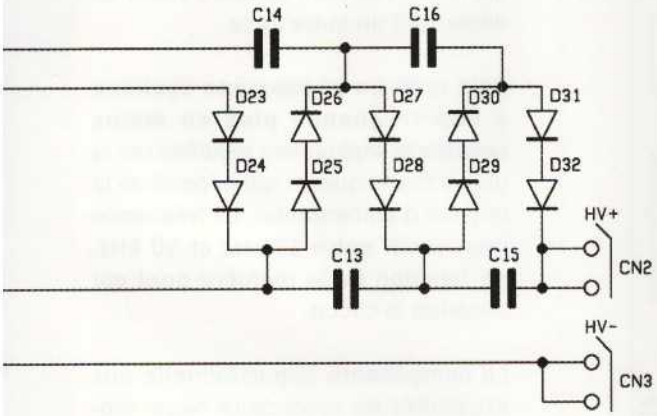
- **génération d'une extra-tension impulsionnelle sur l'enroulement primaire** du transformateur ;
- **augmentation de la tension impulsionnelle** grâce à un **rapport de transformation élevé** (rapport entre le nombre de spires de l'enroulement secondaire et de l'enroulement primaire) du transformateur ;
- redressement et élévation de la tension continue grâce à un **multiplicateur de Cockcroft-Walton**.

Pour générer la haute tension impulsionnelle de départ, nous utilisons un **circuit oscillateur** basé sur un transistor **IGBT**, dont la grille est polarisée par le diviseur de tension constitué par les résistances R1 et R2 et l'enroulement W2 du transformateur.

Ce circuit remplit deux fonctions : il déclenche l'oscillation et augmente la tension, puis la dirige vers le multiplicateur qui se trouve après le secondaire W1 du transformateur. Ce dernier dispose donc d'un enroulement primaire qui est W3, d'un enroulement secondaire élévateur W1 et d'un enroulement de contre-réaction (de rétroaction ou de retour) qui est W2. L'IGBT que nous avons utilisé est un FGH60N60SMD, dont le V_{CE} est de 600 V et le courant de collecteur de 60 A.

CARACTÉRISTISQUES TECHNIQUES

- Tension d'alimentation : 12 VDC
- Courant consommé : 3,4 A
- Tension de sortie maximale : 45 kV
- Fréquence d'oscillation : de 10 kHz à 30 kHz
- Puissance consommée : 40 W



Le schéma électrique

En examinant le schéma électrique du générateur haute tension, nous constatons qu'il est divisé en deux parties : une qui est la partie basse tension et l'autre la partie haute tension. Ces deux parties sont couplées par le transformateur T1. Le connecteur CN4 est utilisé pour alimenter le montage avec une tension continue de 12 V.

Le bornier à vis CN1 permet de distribuer l'alimentation 12 V à un autre module afin d'augmenter la tension de sortie, les tensions de sorties de chaque carte étant alors connectées en série.

Pour comprendre le fonctionnement du circuit, considérons que le bouton SW1 de la carte est enfoncé. Cela a pour effet de relier le négatif de l'oscillateur à la masse de l'alimentation.

La tension de 12 V alimente alors le **diviseur de tension** formé par **R1** et **R2**, de sorte que la **tension aux bornes de R2** traverse **l'enroulement secondaire W2**, initialement inerte vis-à-vis de la grille de l'IGBT.

Ce dernier **entre en conduction** (saturation ou état fermé) du fait qu'une tension positive apparaît entre sa grille et son émetteur et qui excède la tension de seuil $V_{GE(Th)}$ de l'IGBT.

Il s'ensuit **l'apparition d'un courant important traversant l'enroulement W3** du transformateur, car son collecteur est pratiquement au potentiel de son émetteur (proche de 0 V).

Dès qu'un courant circule dans l'enroulement **W3**, **deux tensions induites sont générées** dans les deux enroulements secondaires. Une tension induite très élevée apparaît aux extrémités de **W3**, et qui va au multiplicateur de tension.

Une autre tension induite apparaît aux extrémités de **W2**, compte tenu du sens de l'enroulement par rapport à celui de W3, un **potentiel négatif apparaît** sur la **grille** de **T1** et bloque l'IGBT.

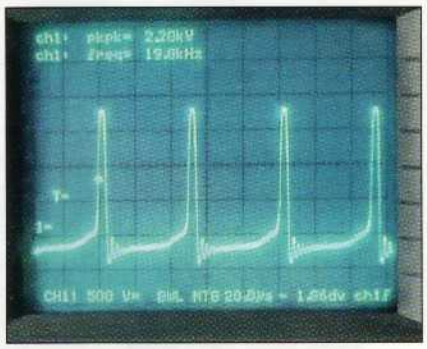


Figure 1 : tension impulsionnelle mesurée sur l'enroulement W1 (aux points test 0+ et 0-).

Lorsque **le courant est brusquement interrompu**, l'enroulement **W3 produit une extra-tension inverse impulsionnelle** qui est annulée par la diode interne de l'IGBT. Cela entraîne **une impulsion de courte durée à haute énergie**, ce qui provoque un pic de tension de plus de 2000 V aux bornes de l'enroulement W1.

L'effet **Corona** et la production d'**ozone**

Lorsqu'une différence de potentiel élevée, supérieure à plusieurs milliers de volts, est appliquée entre deux électrodes immergées dans un milieu gazeux, un phénomène d'ionisation par avalanche se produit. Cela provoque une accélération des particules.

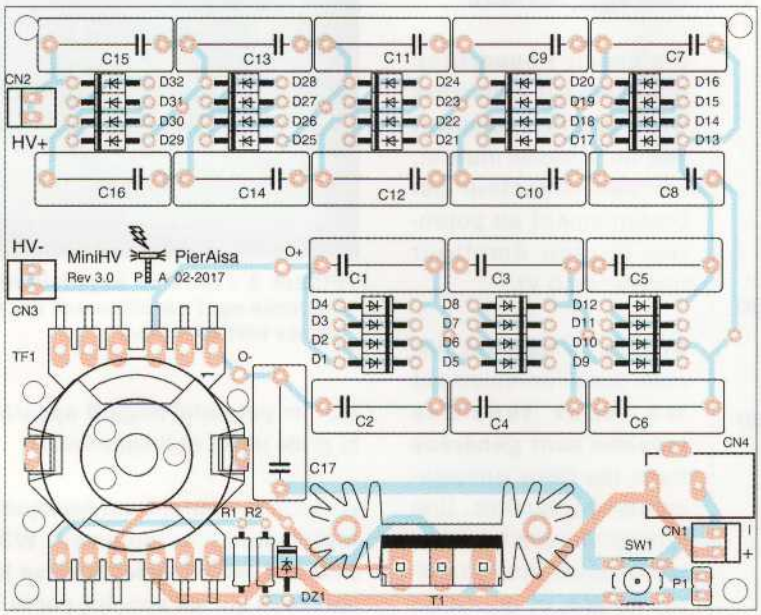
Ainsi, des collisions entre les particules et les molécules présentes dans l'air se produisent, libérant de très grandes quantités de charges libres dans l'air. Un nuage de charges se forme autour des électrodes, générant un courant électrique visible à l'œil nu et émettant une lumière violette avec des crépitements.

Ce phénomène est appelé « effet corona », ou « effet de couronne ». Il s'agit de décharges électriques partielles provoquées par l'ionisation du milieu entourant les électrodes. Ce phénomène apparaît quand le potentiel électrique dépasse une « valeur critique » (mais dont les conditions ne permettent pas la formation d'un arc).

Il se manifeste par l'apparition de points lumineux bleuâtres ou de lignes lumineuses violettes ou encore d'une longue traînée lumineuse qui se forme autour des électrodes. La couronne négative apparaît plus grande que la couronne positive, car les électrons peuvent s'éloigner de la zone d'ionisation et ainsi l'air ionisé peut s'étendre au-delà.

L'effet corona est un excellent générateur d'ozone O_3. La couronne négative en génère une quantité beaucoup plus grande que la couronne positive. Après quelques secondes de fonctionnement du « MiniHV », vous sentirez une odeur âcre, comme lors d'un orage, due précisément à la production de molécules d'ozone.

Plan de montage du générateur « MiniHV »



Plan de câblage des composants du générateur « MiniHV ».

Photo de l'un de nos prototypes du générateur « MiniHV ».

Liste des composants du générateur « MiniHV »

R1..... 470 Ω 1/2W
R2..... 470 Ω 1/2W
C1 à C17... 2,2 nF céramique 7,5 kV
D1 à D32... GP02-40
DZ1.... BZX85C12
T1 ..... FGH60N60SMD
TF1 .... transformateur (voir le texte)
SW1... microswitch

Divers

Bornier 2 pôles au pas de 2,54 mm (x3)
Fiche d'alimentation pour circuit imprimé
Cavalier

NB : les typons des circuits imprimés à l'échelle 1 sont disponibles en téléchargement dans le sommaire détaillée de la revue.

Le blocage de T1 ne dure pas longtemps, car cette impulsion induit une autre tension positive sur la grille, ce qui fait de nouveau conduire l'IGBT en démarrant un autre cycle.

Cela crée un **phénomène cyclique à une fréquence plus ou moins constante**, légèrement modifiée par la dérive thermique, et qui dépend de la tension d'alimentation. La fréquence peut varier entre 10 kHz et 30 kHz, en fonction de la manière dont est construit le circuit.

La composante impulsionnelle aux extrémités du secondaire haute tension est bidirectionnelle (la figure 1 montre la forme de l'onde de la tension aux extrémités de l'enroulement W1) et est **redressée et multipliée** par les cellules multiplicatrices de type de **Cockcroft-Walton**.

Le montage comporte **8 cellules** (ou étages). Dans chaque cellule, le condensateur est chargé lors de l'alternance (demi-période) négative à l'amplitude maximale de la tension alternative (Vpk).

La tension de sortie est égale à la tension alternative d'entrée à laquelle s'ajoute la tension constante du condensateur, la tension de sortie maximale devient donc **2 Vpk**.

La tension de sortie est certes continue, mais elle oscille fortement au rythme de la tension alternative d'entrée.

Pour résumer, nous pouvons dire qu'à travers les diodes et les condensateurs, la charge est piégée et transférée à la cellule suivante en ajoutant la tension présente sur chaque condensateur précédent.

Chaque cellule est constituée de deux condensateurs et de quatre diodes (en réalité deux diodes suffiraient, mais dans notre cas, comme il est parfois difficile de se procurer les composants adéquates pouvant supporter des tensions inverses élevées.

Nous utilisons deux diodes en série dans chaque cellule) qui doivent supporter une tension assez élevée, d'où la nécessité d'utiliser des modèles de

diodes et des condensateurs haute tension. Les borniers CN2 et CN3 sont connectés aux bornes de sorties de la haute tension.

Vous pouvez connecter différents types d'électrodes en fonction du phénomène que vous souhaitez observer, en respectant la polarité « HV+ » pour la borne positive et « HV- » pour la borne négative.

Les diodes D1 à D32 et les condensateurs C1 à C16 constituent les 8 cellules multiplicatrices du générateur de Cockcroft-Walton.

Les diodes ont été doublées (2 en série dans chaque cellule) par rapport à un générateur classique afin de permettre l'utilisation de diodes dont la tension inverse est de 4 KV. Elles sont plus faciles à trouver dans le commerce et ont une chute de tension directe inférieure.

Le condensateur C17 permet de donner une référence par rapport à la masse pour la partie haute tension.

Le cavalier P1, s'il est inséré, permet de contourner le bouton SW1. Le générateur s'allume dès que la tension d'alimentation 12 V est appliquée, sans devoir appuyer sur le bouton SW1.

Nous avons prévu des points test sur le circuit imprimé afin de mesurer la tension à des endroits intermédiaires du multiplicateur, cela permet de détecter d'éventuelles pannes.

Faites attention et utilisez des appareils pouvant mesurer des tensions élevées.

Les multimètres classiques sont à proscrire dans le cas où vous voulez effectuer une mesure directe (voir la figure 7).

La diode zener **DZ1 protège la grille** du transistor **IGBT** contre les **surtensions**. Les résistances R1 et R2 polarisent à l'allumage le transistor à environ 9 V.

Réalisation pratique

Pour construire le générateur « MiniHV », un circuit imprimé double face un peu particulier a été fabriqué. Il faudra le reproduire le plus fidèlement possible pour obtenir les résultats souhaités. Vous pouvez télécharger les typons sur notre site dans le sommaire détaillé de la revue.

En effet, **compte tenu des tensions importantes**, la **forme et l'espacement entre les pistes doivent être reproduits à l'identique**.

Pour garantir une bonne isolation, **il est nécessaire de réaliser les découpes prévues sur le circuit imprimé** (traits blanc épais sur la photo du prototype), de manière à séparer la zone haute tension du reste du circuit.

Cela évitera l'apparition d'arcs électriques du fait des fortes différences de potentiels qui sont présentes entre les pistes en fonctionnement.

Une fois le circuit imprimé gravé et percé, sans oublier les découpes, commencez comme d'habitude par souder les composants ayant un profil bas (les résistances, les diodes, le poussoir).

Ensuite, continuez en soudant la fiche d'alimentation, les borniers et les condensateurs.

Montez le transistor IGBT sur un radiateur ayant une résistance thermique de 14 °C/W en le fixant au moyen d'une vis M3 avec un écrou. Sa partie métallique, qui est contre le radiateur, doit être positionnée vers l'intérieur du montage (les inscriptions sur le transistor doivent être orientées vers l'extérieur de la carte).

Le transformateur TF1 doit être réalisé sur un **noyau en ferrite de type RM10 en matériau N87** avec un coefficient **AL= 4,2 µH**. Il s'agit d'un noyau **EPCOS** (codes commandes : 1422723 et 2355131 chez farnell.fr).

La disposition des broches est en ligne afin d'augmenter l'isolation des enroulements primaires et secondaires.

Dans un premier temps, commencez par bobiner l'enroulement W1 constitué de 100 spires de fil de cuivre émaillé de diamètre de 0,315 mm dans le sens des aiguilles d'une montre.

Ensuite, dénudez les extrémités du fil et soudez le début de l'enroulement sur la broche 1 du corps du transformateur et la fin de l'enroulement sur la broche 3.

Enrobez d'une couche de ruban isolant le bobinage que vous venez de réaliser.

Ensuite, commencez le bobinage de l'enroulement W2 en utilisant un fil de cuivre émaillé de 0,75 mm de diamètre.

L'enroulement W2 est constitué de 4 spires bobinées dans le sens des aiguilles d'une montre.

Dénudez les extrémités du fil de cuivre émaillé et soudez le début du bobinage W2 sur la broche 9 du corps du transformateur et la fin du bobinage sur la broche 7.

Enfin, il faut réaliser l'enroulement W3. Il est constitué de 5 spires de fil de cuivre émaillé de 1 mm de diamètre bobiné dans le sens des aiguilles d'une montre.

Dénudez les extrémités du fil de cuivre émaillé et soudez le début du bobinage W3 sur la broche 10 du corps du transformateur et la fin du bobinage sur la broche 12.

À ce stade, vous pouvez insérer les bobinages qui se trouvent sur le corps du transformateur à l'intérieur des deux demi-noyaux. Ensuite, verrouillez le noyau à l'aide des accessoires appropriés.

Le circuit imprimé est également adapté à l'utilisation de transformateurs toroïdaux, pour ceux qui veulent expérimenter d'autres modes de fonctionnement du générateur « MiniHV ».

Vous pouvez tester différents types de matériaux et de nombres de spires, tout en respectant la polarité des enroulements.

La polarité est primordiale car l'enroulement primaire doit induire dans l'enroulement secondaire de rétroaction (broches 7 et 9) une tension de polarité telle qu'elle déclenche l'oscillation.

En résumé, lorsque l'IGBT est conducteur, la broche 9 du transformateur

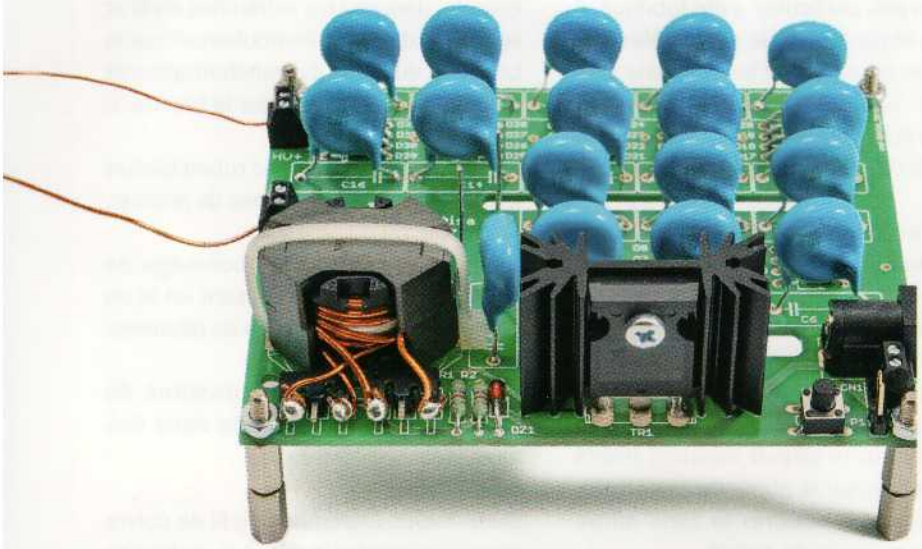


Figure 2 : disposition des électrodes.

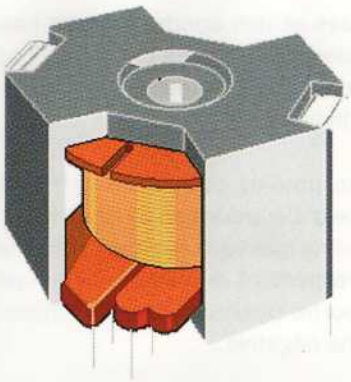


Figure 3 : assemblage du transformateur TF1.

doit fournir une impulsion négative qui bloque le transistor.

Une fois le transformateur réalisé et soudé, insérez quatre entretoises hexagonales pour maintenir la carte à une certaine distance du plan de travail.

Enfin, insérez deux fils de cuivre rigides dénudés d'environ 5 cm de long et de 1 mm de diamètre dans les borniers de sortie. Les fils doivent être disposés comme indiqué en figure 2.

Comment obtenir un arc électrique

Afin de s'assurer que les condensateurs ne soient pas chargés, déchargez-les en faisant un court-circuit entre les électrodes à l'aide d'un tournevis isolé, comme décrit dans l'encadré « **Quelques précautions** ». Espacez les électrodes à une distance d'environ 4 mm l'une de l'autre. Retirez le cavalier P1 pour activer le fonctionnement du bouton.

Le circuit doit être alimenté avec, de préférence, une alimentation stabilisée de laboratoire, qui doit fournir une tension continue comprise entre 10 VDC et 12 VDC et dont la puissance doit être d'au moins 40 W (soit 4 A sous 12 V).

Nous ne recommandons pas l'utilisation de blocs secteurs avec la prise 230 V intégrée, car ils ne présentent généralement pas une bonne stabilisation de la tension de sortie et produiront donc des décharges avec moins d'efficacité.Insérez la prise jack de l'alimentation 12 V dans le connecteur CN4.

Ionisation de l'air

L'air que nous respirons est composé d'un mélange de différents types de gaz tels que l'oxygène, l'azote, le dioxyde de carbone.

Les atomes qui composent les molécules présentes dans l'air peuvent, pour différentes raisons, y compris la haute tension, perdre leur état d'équilibre du point de vue de leur charge électrique et devenir des ions positifs s'ils perdent des électrons ou des ions négatifs s'ils acquièrent des électrons.

Les ions positifs, prédominants dans l'air pollué des grandes villes, n'ont pas un bon effet sur notre santé car ils diminuent les défenses immunitaires et ne favorisent pas l'assimilation de l'oxygène, sans compter qu'ils agissent négativement sur le système nerveux.

Les ions négatifs, à l'inverse, doivent être considérés comme de véritables germicides, car ils peuvent éradiquer les bactéries et atténuer les effets produits par les allergies et l'asthme. De plus, ils facilitent l'assimilation de l'oxygène.

L'air que nous respirons en mer ou à la montagne est plus sain car, surtout en montagne, l'air a une forte concentration en ions négatifs (plus de 1000 par cm^3).

Notre circuit « MiniHV », avec une adaptation, peut également être utilisé pour la production d'ions négatifs.

En effet, si la polarité des diodes D1 à D32 est inversée, c'est-à-dire en montant les diodes avec leur cathode du côté opposé à la sérigraphie, une zone avec un potentiel négatif est créée sur les électrodes qui sont alors capables de produire des électrons aux molécules de l'air ambiant pour ensuite créer des ions négatifs.

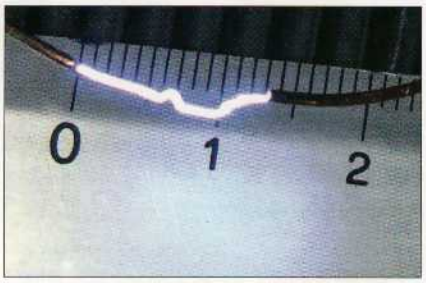


Figure 4 : mesure de la longueur de l'arc électrique.

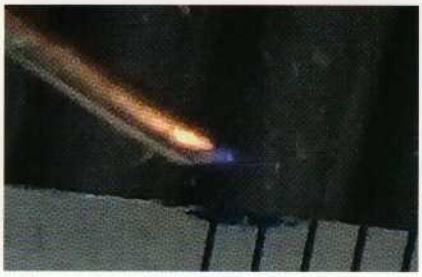


Figure 5 : l'effet corona visible sur une électrode.

Appuyez sur le bouton et vérifiez qu'un arc électrique continu est établi entre les électrodes. Remarque importante : pendant l'utilisation, ne maintenez pas l'arc pendant plus de 10 secondes afin d'éviter une surchauffe du transistor IGBT.

Maintenant, espacez les électrodes d'environ 10 mm et appuyez sur le bouton.

Observez comment l'arc change de forme, d'intensité et de fréquence de déclenchement, en raison de la plus grande distance entre les électrodes (voir la figure 4).

Comment obtenir l'effet corona

Assurez-vous que les condensateurs ne soient pas chargés, pour cela déchargez les capacités en court-circuitant les électrodes comme décrit dans l'encadré intitulé « Quelques précautions ».

Espacez les électrodes à une distance d'environ 20 mm. À cette distance, la tension produite par le « MiniHV » n'est pas suffisante pour produire un arc électrique.

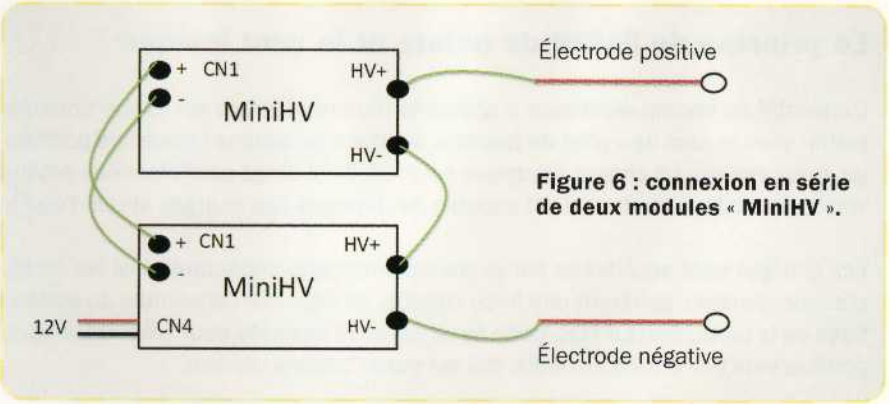


Figure 6 : connexion en série de deux modules « MiniHV ».

Si vous regardez attentivement la zone autour des extrémités des électrodes, vous apercevez l'effet corona représenté par une zone violette autour des extrémités et vous devez entendre un crépitement (voir la figure 5).

Connectez plusieurs modules « MiniHV » en série

Pour ceux qui souhaitent augmenter la tension de sortie, il est possible de connecter plusieurs modules « MiniHV » en série. Pour cela, il est nécessaire d'effectuer les connexions indiquées en figure 6.

Une attention particulière doit être portée à la distance d'isolement entre les deux modules « MiniHV », ils doivent être espacés d'au moins 5 cm et montés sur des entretoises de préférence en plastique

Pour la distribution de l'alimentation 12 V au second module, le connecteur CN1 peut être utilisé comme indiqué en figure .

Il faudra verifier que l'alimentation soit capable d'alimenter les 2 modules, c'est-à-dire qu'elle devra fournir une tension de 12 V avec un courant d'au moins 7 A (80 W). ■

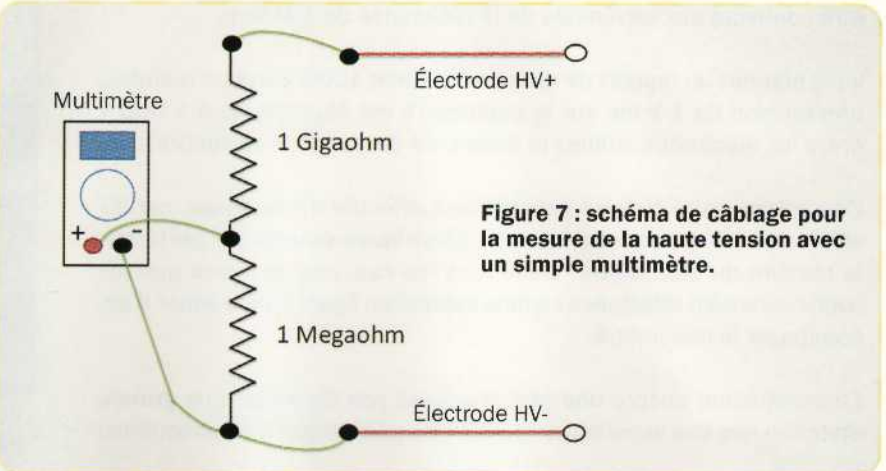


Figure 7 : schéma de câblage pour la mesure de la haute tension avec un simple multimètre.

Le principe de l'effet de pointe et le vent ionique

L'intensité du champ électrique à proximité d'un conducteur est proportionnelle à la courbure de sa surface, phénomène connu sous le nom de « effet de pointe ». Si nous connectons l'électrode positive à un corps pointu et l'électrode négative à un corps arrondi, un champ électrique est créé. Il est dirigé par l'électrode pointue, où les lignes de force sont plus intenses, vers l'électrode arrondie qui est capable de déplacer des charges libres d'où l'ionisation de l'air à proximité.

Les charges sont accélérées par le champ électrique impactant ainsi les molécules neutres de l'air et, grâce au principe d'action-réaction, génèrent une force capable de déplacer l'ensemble du système « pointe-sphère ». Ce phénomène est à la base de la propulsion LIFTER. Cette force est alors associée pour créer un véritable mouvement des particules de l'électrode pointue vers l'électrode arrondie, qui est perçu comme un vent.

Afin d'expérimenter ce phénomène, deux électrodes peuvent être réalisées comme indiqué en figure 8. Pour cela, il suffit d'utiliser un étrier, une vis pour la pointe et un écrou borgne pour la surface arrondie. Pour visualiser le vent ionique, une flamme de bougie placée entre les deux électrodes peut être utilisée afin de vérifier que la flamme se plie lorsque le bouton est enfoncé (voir la figure 8), ou une bande de papier légère (voir la figure 9).

Faites particulièrement attention, en évitant tout contact direct avec les parties métalliques, telles que les supports.

Mesure de la haute tension

Dans le cas d'un champ électrique uniforme et d'un air sec à la pression atmosphérique de référence (au niveau de la mer), la rigidité diélectrique du milieu isolant représenté par l'air vaut 3 kV/mm. En fonction de la longueur de l'arc produit, la tension entre les électrodes peut être déterminée empiriquement.

Par exemple, si l'arc a une longueur de 15 mm, la différence de potentiel théorique est de 45 kV.

Pour mesurer la haute tension en pratique, un diviseur de tension doit être fabriqué avec une résistance de 1 gigaohm (GΩ) et une résistance de 1 mégaohm (MΩ) connectées en série, comme indiqué en figure 7.

Les résistances doivent avoir un corps assez grand (distance entre les broches) pour éviter la formation d'arcs électriques au niveau des broches, car elles sont soumises à une différence de potentiel élevée. La résistance de 1 gigaohm doit avoir un corps d'une distance d'au moins 3 cm, pour la résistance de 1 mégaohm, il faut un corps d'au moins 1 cm.

Réglez une distance entre les électrodes d'au moins 20 mm, un multimètre classique (calibré sur la plus haute gamme de tension) peut être connecté aux extrémités de la résistance de 1 Mohm.

Vous obtenez un rapport de division d'environ 1000. De cette manière, une tension de 1 V lue sur le multimètre est équivalente à 1 000 V entre les électrodes. Affinez la lecture en descendant de calibre.

Cependant, cette mesure pourrait être affectée d'une erreur, car les effets corona et les forts champs électriques pourraient perturber la mesure du multimètre. Dans tous les cas, assurez-vous que les connexions sont effectuées comme indiqué en figure 7 pour éviter d'endommager le multimètre.

En conclusion, encore une fois, n'oubliez pas de prêter une grande attention lors des expériences pour éviter tout danger d'électrocution !

Figure 8 : Effet du vent ionique sur la flamme.

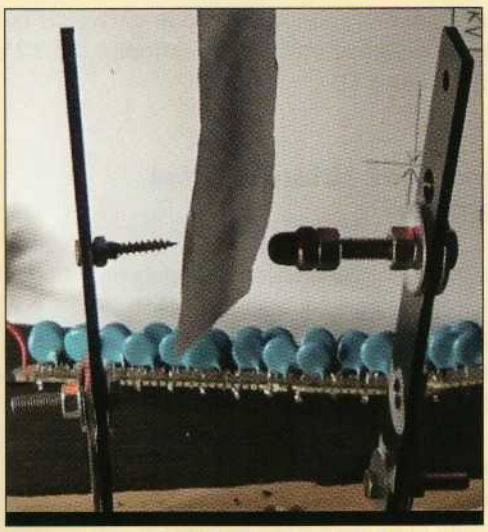


Figure 9 : Effet du vent ionique sur une bande de papier légère.

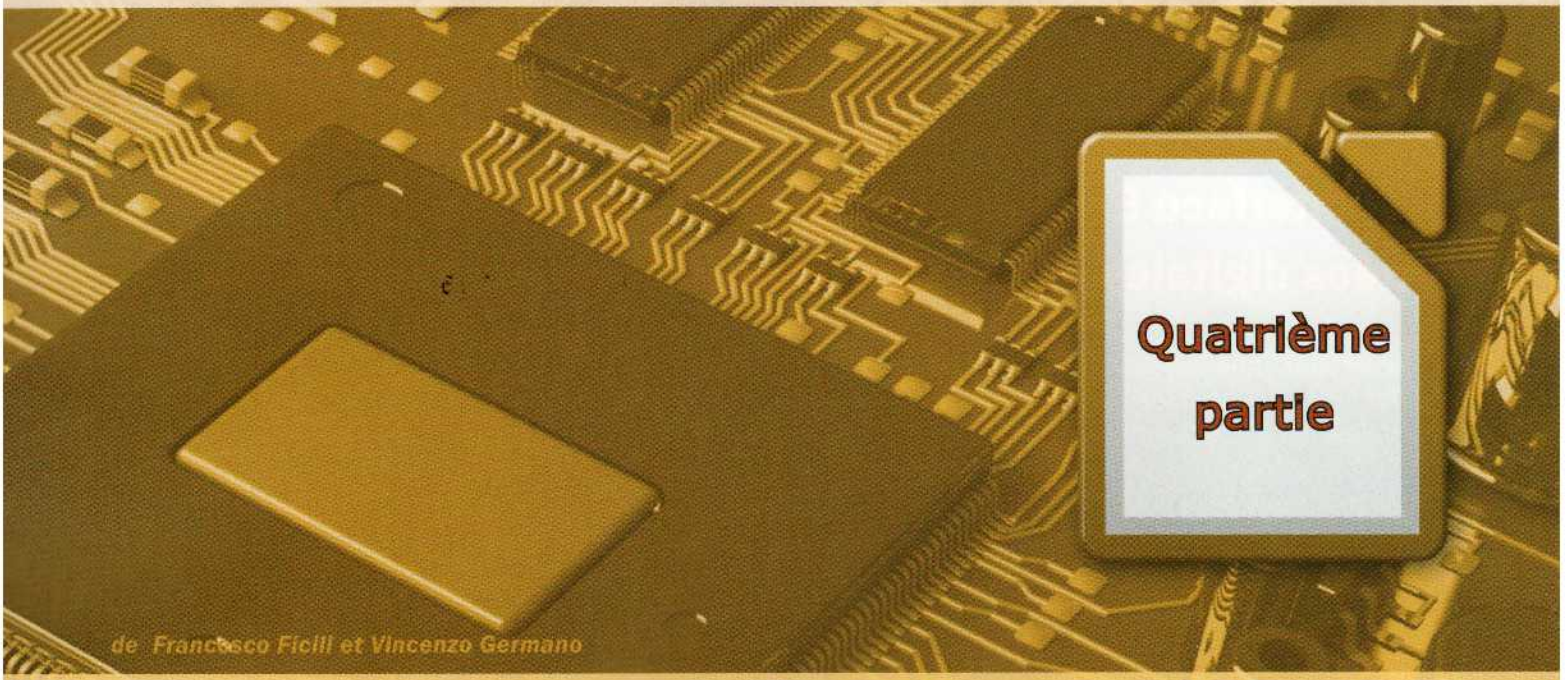
Quelques précautions

Pour tirer le maximum de satisfaction de l'utilisation du circuit, n'oubliez pas qu'il fonctionne à très haute tension, il est donc essentiel de prêter la plus grande attention aux points suivants :

- **assurez-vous qu'il n'y a pas de substances inflammables à proximité du générateur « MiniHV »**, car il dégage de la chaleur au niveau du transistor et produit des arcs électriques capables de créer des flammes ;
- **il ne faut pas toucher avec les mains nues la partie haute tension** se trouvant du côté du secondaire du transformateur, comme les diodes et/ou les condensateurs. **Ne touchez pas non plus les broches de sortie**, même après avoir mis le circuit hors tension, car la charge dans les condensateurs est conservée pendant plusieurs secondes ;
- **après l'utilisation du circuit, déconnecter l'alimentation et déchargez les condensateurs** en court-circuitant les électrodes de sortie à l'aide d'un tournevis isolé, la partie métallique du tournevis doit établir un contact entre les deux électrodes.

Malgré les précautions énumérées, **il est fortement déconseillé d'utiliser le circuit pour les personnes cardiaques ou disposant d'un pacemaker**. Ces personnes doivent cependant rester à distance de sécurité, car les interférences hautes fréquences dues à la présence de la haute tension, ou à la formation d'un arc électrique, peuvent produire des effets imprévisibles sur un pacemaker. À éviter !

Apprenez à maîtriser kicad EDA



de Francesco Ficili et Vincenzo Germano

Nous sommes maintenant familiarisés avec la suite de conception électronique KiCad, qui fait l'objet de ce cours. Nous allons enrichir nos connaissances en commençant par l'analyse de Pcbnew, l'éditeur de circuit imprimé (layout). Pour cela, nous allons poursuivre le développement de notre projet pratique commencé dans le précédent numéro. Quatrième partie.

Dans la publication précédente du numéro 143, nous avons approfondi nos connaissances d'Eeschema, l'éditeur de schémas de KiCad, qui permet de créer des composants personnalisés ou d'éditer des bibliothèques existantes.

Nous avons aussi appris à associer des empreintes avec des schémas de composants à l'aide de l'outil Cvpcb. Nous avons également continué à développer notre demoboard pour le microcontrôleur PIC18F4550, que nous avons pris comme exemple pour la conception d'un circuit imprimé.

Nous avons complété le schéma électrique de la carte de développement et nous avons étudié la fonction d'association des composants.

Dans cette quatrième partie, nous consolidons nos connaissances sur l'utilisation de Cvpcb (l'un des composants fondamentaux de KiCad), en associant les composants de la demoboard.

Nous introduisons également un autre outil, Pcbnew, qui est l'interface de routage du circuit imprimé.

Conception d'un circuit imprimé avec KiCad : l'outil Pcbnew

L'une des phases les plus critiques du développement d'une carte électronique est la réalisation du tracé du circuit imprimé, c'est-à-dire le tracé des pistes et les interconnexions entre les différents composants, mais aussi entre les différentes couches du circuit imprimé.

Ce processus est généralement appelé « **routage** » de la carte. Le résultat final de cette phase est la génération des fichiers « **gerber** » qui seront utilisés pour la réalisation physique du circuit imprimé à l'aide de machines professionnelles à commande numérique.

Dans KiCad, l'outil utilisé pour réaliser le routage du circuit imprimé est l'outil Pcbnew. Cet outil puissant intègre :

- un **éditeur de traçage** de circuit imprimé ;
- un **éditeur d'empreintes** des composants ;
- une **visualisation en 3D** de la carte ;
- un **générateur** des fichiers « **gerber** » de sortie.

C'est donc un outil très complet, qui peut être utilisé pendant toute la phase de routage de la carte électronique, jusqu'à la création des fichiers de sortie « gerber ».

Pour le lancer, vous pouvez cliquer sur l'icône correspondante dans la barre d'outils KiCad, comme illustré en figure 1.

Vous pouvez aussi utiliser le bouton de lancement de Pcbnew à partir d'Eeschema (voir la figure 1a).

Une fois l'outil lancé, l'interface s'ouvre avec une fenêtre qui ressemble à celle de la figure 2. Comme vous pouvez le voir sur la figure, l'interface comprend les éléments suivants :

- « **Project bar** » ou barre du projet : c'est la barre de gestion du projet à partir de laquelle vous pouvez ouvrir un projet, enregistrer le projet en cours, importer une « netlist », définir et exécuter le contrôle des règles de routage des pistes, générer les fichiers gerber de sortie et les fichiers de perçage ;
- « **Tool bar** » ou barre d'outils : comme son nom l'indique elle contient les outils nécessaires pour la réalisation du routage du circuit imprimé. Grâce aux outils de cette barre, il est possible d'effectuer les opérations de routage, d'insertion de pistes, de remplissage de zone et ou encore d'insertion de texte et/ou d'éléments graphiques ;
- « **Workspace** » : correspond à l'espace de travail. Dans cette zone, le circuit imprimé est routé ;
- « **Option bar** » : il s'agit de la barre des options de l'affichage. Elle contient de nombreuses options d'affichages rapides, telles que

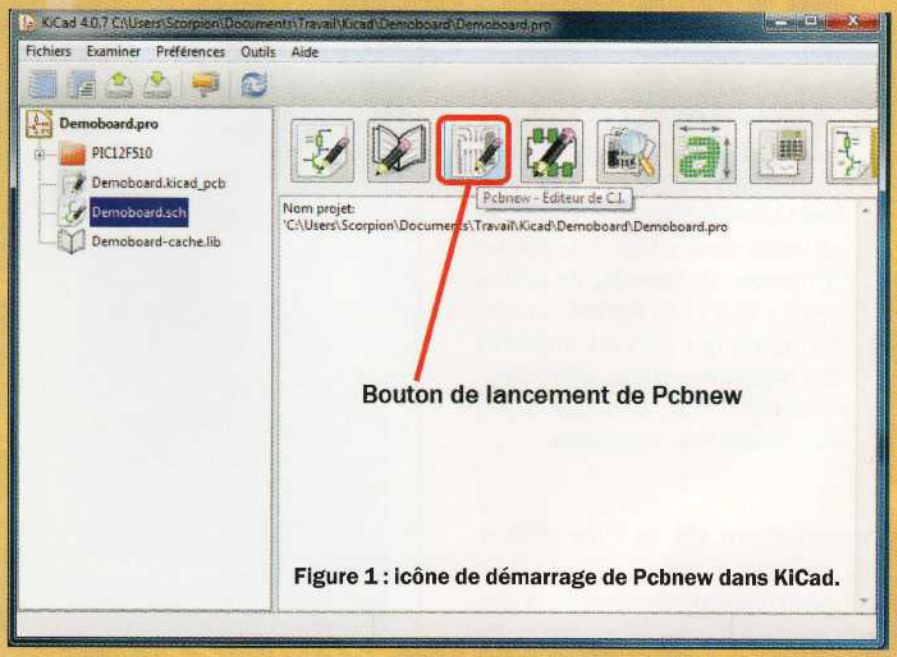


Figure 1 : icône de démarrage de Pcbnew dans KiCad.

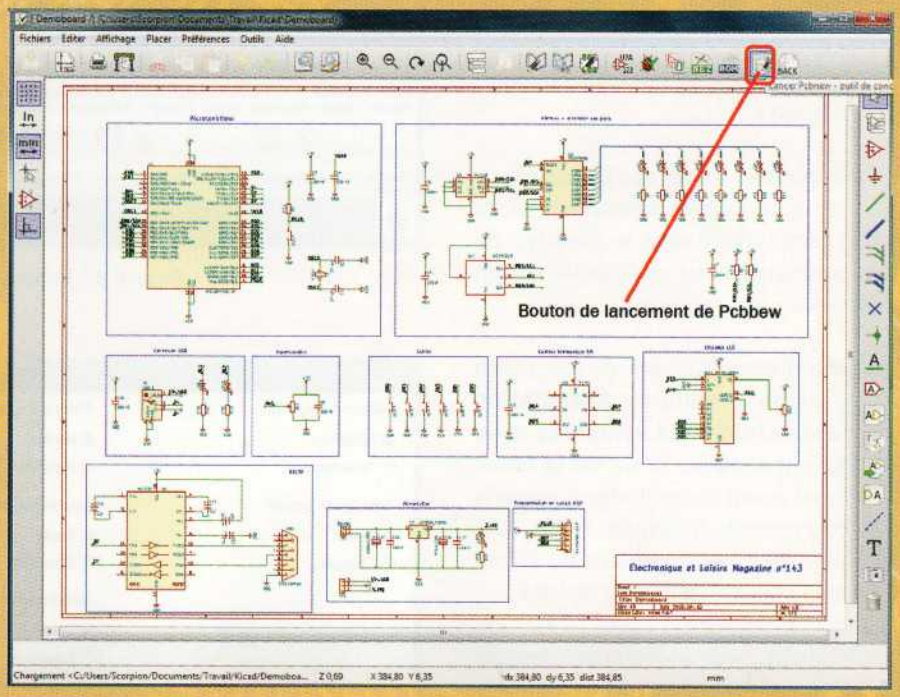


Figure 1a : icône de démarrage de Pcbnew à partir d'Eeschema.

les zones d'affichages, les liens entre les modules, la grille, etc. ;

- « **Design status bar** » ou barre d'état de conception : elle contient les indicateurs qui résument l'état du circuit imprimé en cours de conception. Elle affiche le nombre de nœuds, de pistes et de « vias » (un via est un trou métallisé qui permet d'établir une liaison électrique entre deux couches du PCB), ainsi que la liste des nœuds non connectés.

Importation de la « netlist »

Pour cela, nous allons travailler à partir du schéma du « projet exemple » de la figure 7 (page 90) de la revue 143.

Le schéma contient un PIC12F510, 4 résistances, 2 condensateurs non polarisés, 2 LED, 2 boutons poussoir et un quartz.

Peu importe la valeur des composants, le but de ce « projet exemple » est de vous familiariser avec le passage de la schématique au routage du circuit imprimé.

Eventuellement, vous devez relire le paragraphe intitulé « **Association des empreintes aux symboles avec CvPcb** » (pages 91 et 92 de la revue 143), en ayant correctement associé les composants indiqués, sinon vous n'obtiendrez pas le chevelu du circuit imprimé comme visible en figure 8.

Maintenant, nous pouvons commencer à faire les premiers pas dans l'interface de Pcbnew, en profitant du « projet exemple » contenant le PIC12F510.

Dans l'article précédent, nous avions terminé l'association « composants/empreintes » de ce projet, nous allons maintenant importer la « netlist » dans Pcbnew.

Pour importer une « netlist » dans Pcbnew, sélectionnez l'icône portant l'inscription « **NET** » dans la barre du projet, comme indiqué en figure 3. Une fois cela fait, une boîte de dialogue représentée en figure 4 s'ouvre. Elle comporte de nombreuses options.

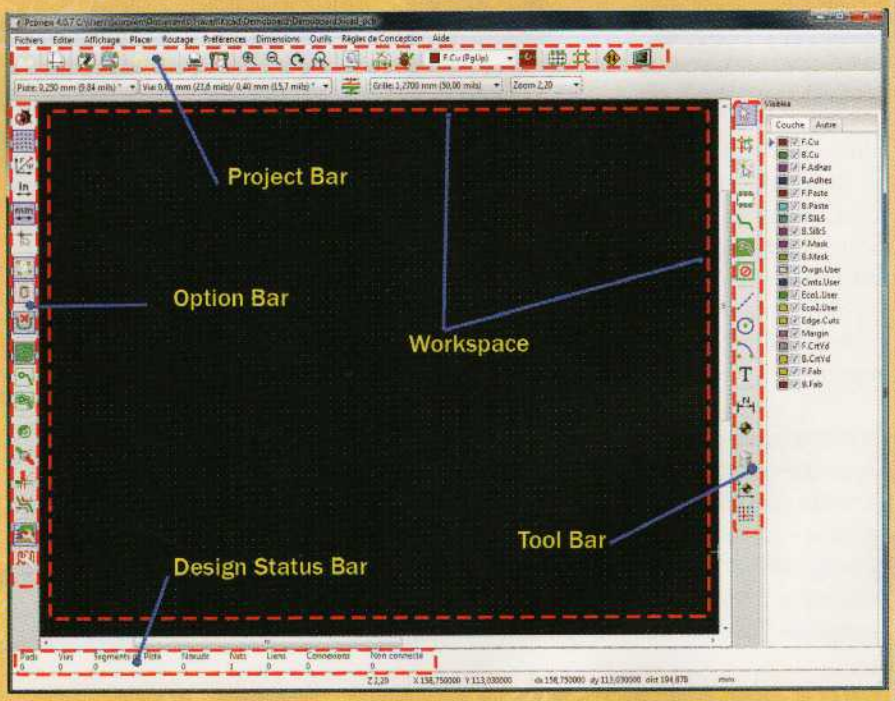


Figure 2 : l'interface de Pcbnew.

Figure 3 : importation de la « netlist ».

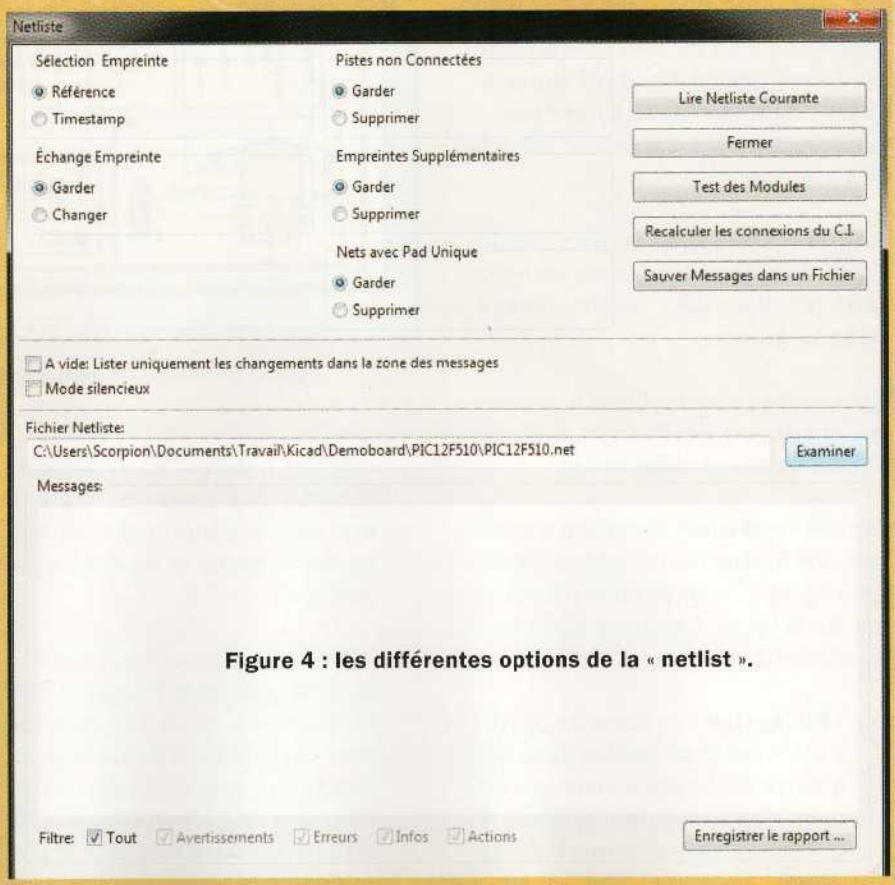


Figure 4 : les différentes options de la « netlist ».

Pour le moment, nous conservons les paramètres par défaut du programme et nous cliquons sur le bouton « **Examiner** » pour sélectionner le fichier « netlist » à importer.

Dans le précédent numéro, nous avions nommé la « netlist » du projet « PIC12F510.net ». Sélectionnez alors ce fichier comme visible en figure 5, puis cliquez sur ouvrir.

À ce stade, vous revenez à la figure précédente avec le chemin correct du fichier indiqué dans le champ « **Fichier Netlist** ».

Cliquez alors sur le bouton « **Lire Netlist Courante** », une série de messages s'affiche dans la zone inférieure.

Si des erreurs dans la « netlist » importée sont détectées, par exemple une discordance entre la schématique d'un composant et l'empreinte correspondant, l'outil le signalera dans la zone « **Messages** » afin que vous puissiez apporter les corrections nécessaires.

Si tout est correct, la « netlist » sera importée dans Pcbnew et l'outil retournera des messages similaires à ceux visibles en figure 6.

À ce stade, les composants sont insérés dans l'espace de travail, mais ils sont tous superposés.

La **première opération** à effectuer est donc de les **distribuer**, c'est-à-dire de les **répartir dans l'espace de travail**.

KiCad dispose d'un mode de positionnement automatique qui évite d'avoir à le faire manuellement.

Cette opération peut être extrêmement longue et fastidieuse si les composants sont nombreux.

Pour effectuer une distribution automatique, activez d'abord l'option « **Mode empreintes** » en cliquant sur l'icône correspondant dans la barre du projet, à côté de la sélection du calque courant (voir la figure 7).

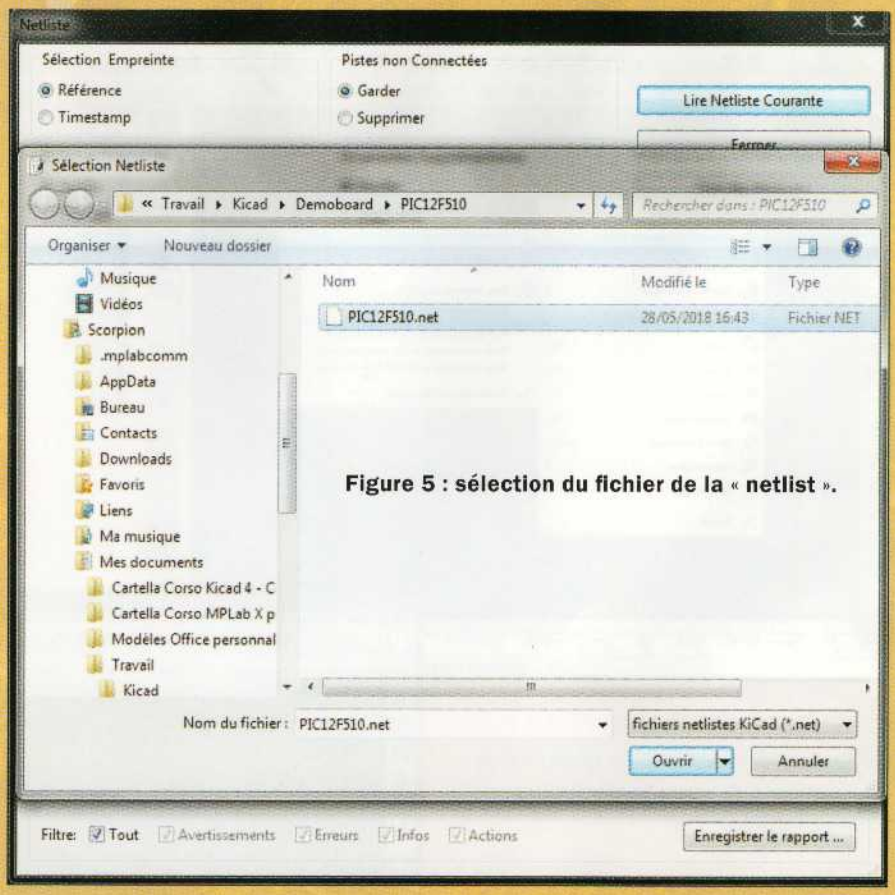


Figure 5 : sélection du fichier de la « netlist ».

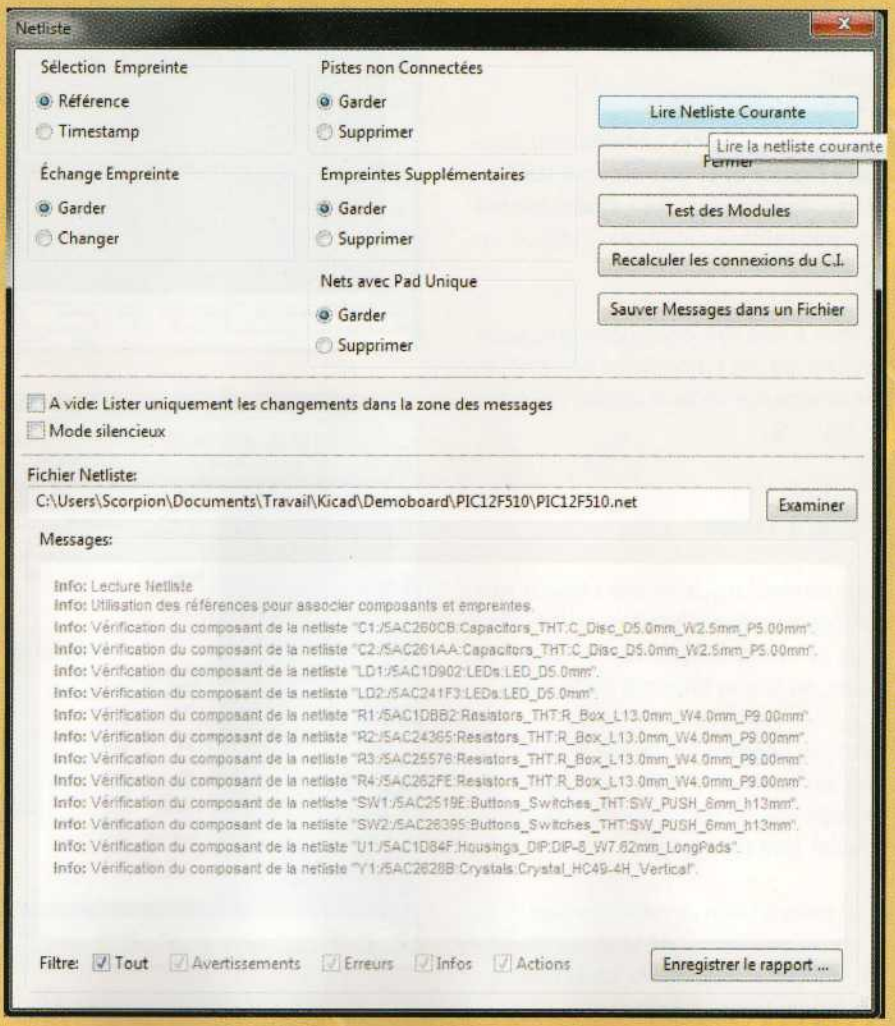


Figure 6 : la « netlist » correctement importée.

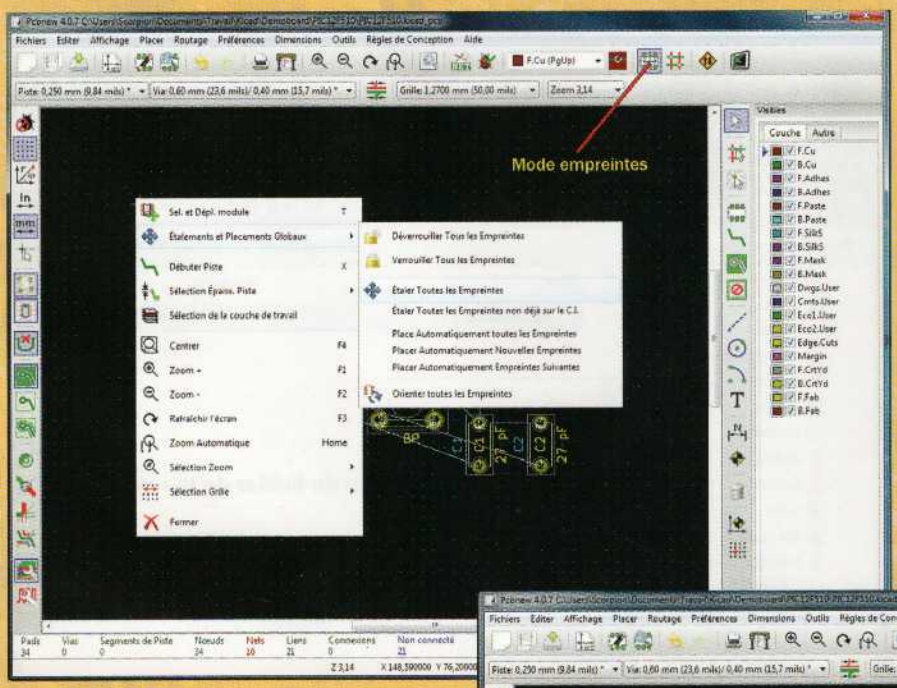


Figure 7 : cliquez sur l'icône d'activation du « Mode empreintes » (en haut), puis ensuite faites un clic droit dans l'espace de travail pour afficher les boîtes de dialogues visibles au centre de cette figure. Vous effectuez ainsi une répartition automatique des empreintes.

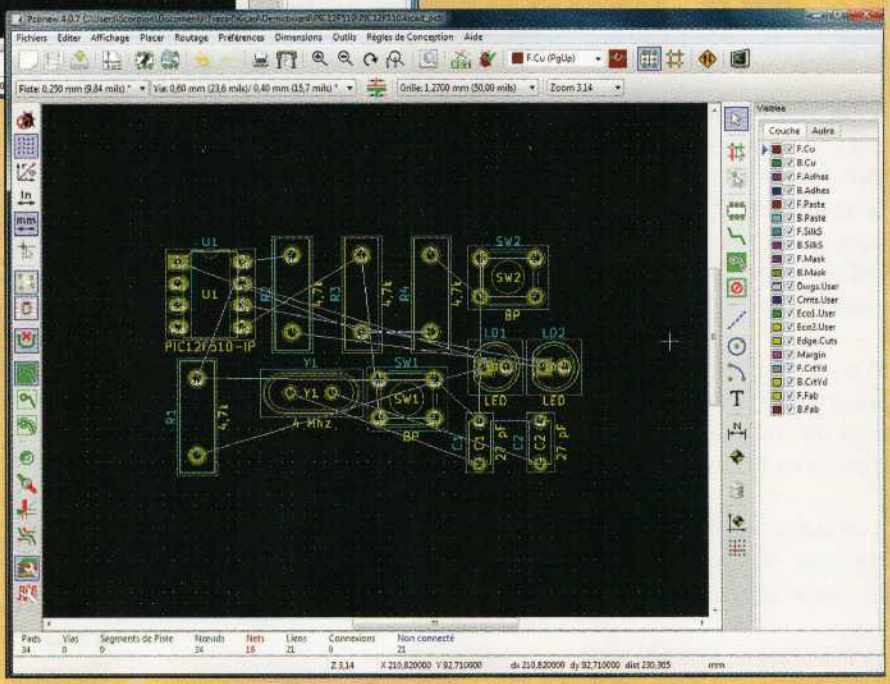


Figure 8 : les empreintes des composants réparties dans l'espace de travail, les fils représentent les connexions physiques entre les composants, ces fils sont appelés « chevelu » du circuit imprimé, ce dernier n'étant pas encore routé.

Cette dernière permet de positionner manuellement ou automatiquement les empreintes.

Une fois l'option activée (clic sur l'icône), il suffit de cliquer avec le bouton droit de la souris dans l'espace de travail.

Une fenêtre s'ouvre, sélectionnez l'option « **Etalements et Placements Globaux** », puis cliquez sur « **Etaler toutes les empreintes** » comme indiqué en figure 7.

Si tout a été fait correctement, vous devriez voir les empreintes des composants répartis de la manière indiquée en figure 8.

Design rules

Maintenant, nous avons réparti nos composants dans l'espace de travail et nous pouvons commencer à router les pistes du circuit imprimé de la carte.

Avant de démarrer la phase de conception proprement dite, les règles de routage (ou de conception) des pistes doivent être définies.

Pour cela, il suffit de sélectionner dans le menu au-dessus de la barre du projet l'option « **Règles de conception** » (à côté du menu « Aide »).

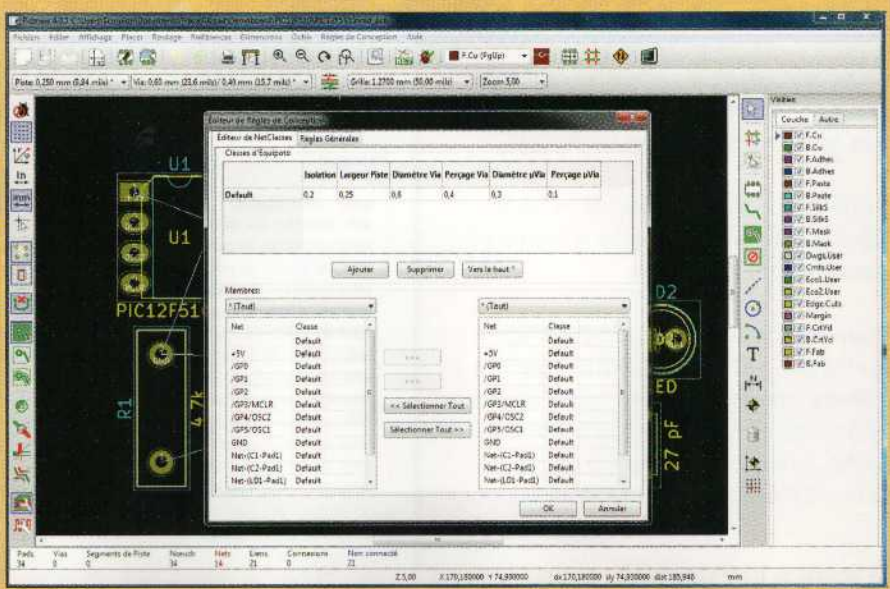


Figure 9 : la fenêtre « Règles de conception » (Design rules).

Cliquez sur l'icône représentée par un « marteau », la fenêtre de la figure 9 apparaît. Celle-ci permet de définir les règles de conception du projet.

Il est ainsi possible de définir des groupes de règles auxquelles un groupe de « nets » peut être associé.

De cette manière, par exemple, il est possible de **définir un groupe de règles** pour les « nets » de l'alimentation et un autre pour les « nets » des signaux qui peuvent avoir des largeurs de pistes différentes et des diamètres de vias différents.

Par exemple, à l'intérieur de la fenêtre précédemment ouverte, nous définissons deux nouveaux groupes de règles que nous appelons respectivement « **Alimentation** » et « **Signaux** » avec les règles définies comme visible en figure 10.

Pour créer un nouveau groupe, cliquez sur le bouton « **Ajouter** » puis dans la zone de texte **tapez le nom du groupe** et cliquez sur le bouton « **OK** ». Le nouveau groupe apparaît sous le groupe par défaut (partie supérieure de la figure 10). Faites de même pour le groupe « Signaux ».

Ensuite, nous devons associer les « nets » « +5V » et « GND » au groupe « Alimentation » et toutes les autres au groupe « Signaux ».

Pour cela, dans la partie inférieure de la fenêtre et à droite, sélectionnons dans la **liste déroulante** où l'inscription « **(*Tout)** » apparaît le groupe « Alimentation » que nous venons de créer.

Dans la liste située à gauche, sélectionnez la « net » « +5V » et cliquez sur le bouton représenté par les 3 symboles « **>>>** » de manière à ajouter cette « net » au groupe alimentation. Faites de même pour la « net » « GND ».

Pour toutes les autres « nets », ajoutez-les au groupe « Signaux » en appliquant la même procédure.

Pour plus de clarté, nous avons résumé les différentes règles dans le tableau 1 qui reprend la partie supérieure en bleue de la figure 10.

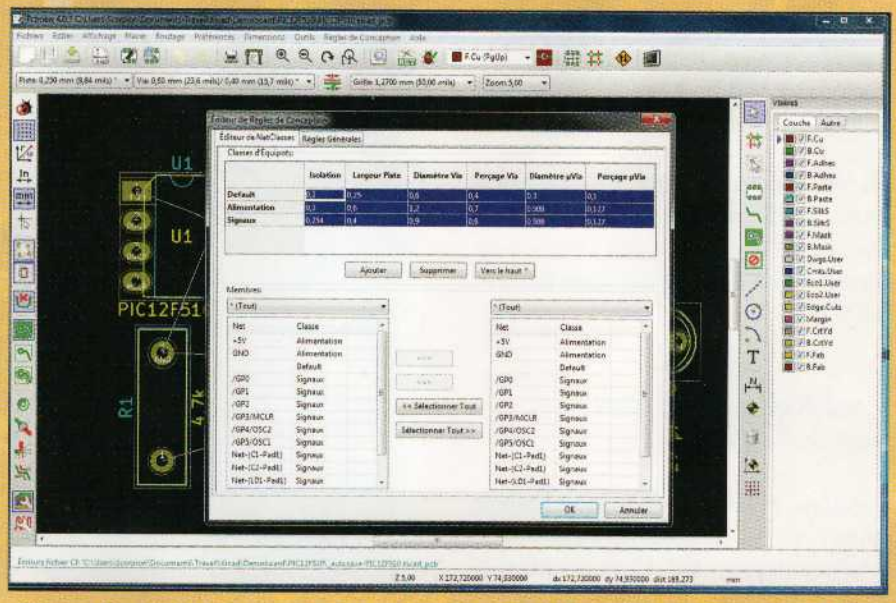


Figure 10 : règles des groupes « Alimentation » et « Signaux ». Reportez-vous au tableau 1 pour les valeurs.

Tableau 1 – Règles de conception

	Isolation	Largeur piste	Diamètre Via	Perçage Via	Diamètre µVia	Perçage µVia
Défaut	0,2	0,25	0,6	0,4	0,3	0,1
Alimentation	0,3	0,6	1,2	0,7	0,508	0,127
Signaux	0,254	0,4	0,9	0,6	0,508	0,127

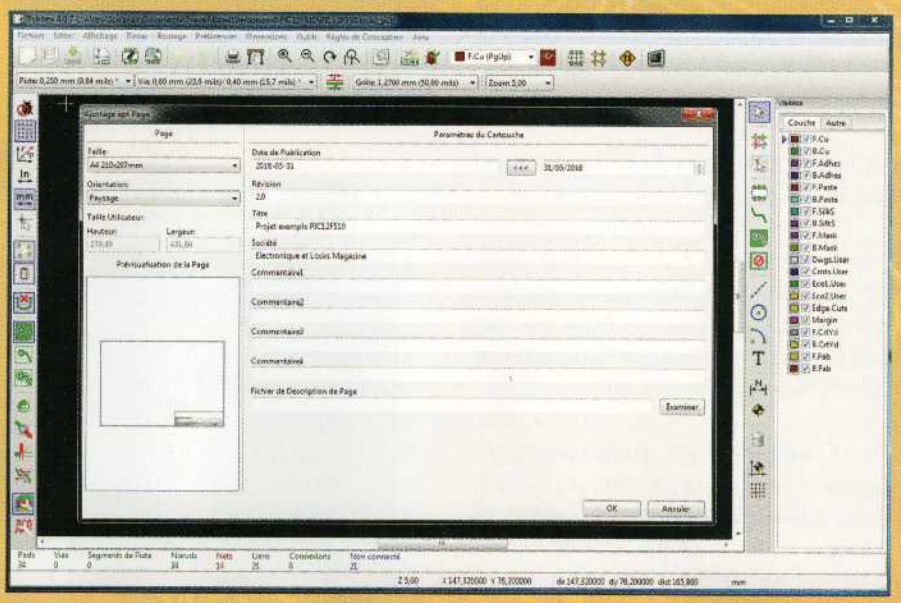


Figure 11 : les différents réglages de la mise en page ainsi que l'insertion des données dans le cartouche.

De cette façon, lorsque nous effectuerons le routage du circuit, en fonction des « nets » sur lesquelles nous travaillerons, l'ensemble de règles définies sera appliqué.

Paramétrage de la mise en page

Comme dans Eeschema, il est possible dans Pcbnew d'ajuster la mise en page.

Pour ce faire, sélectionnez simplement le menu « **Fichier** » → « **Ajustage Page** » (ou la 4ème icône en partant de la gauche dans la barre du projet).

La fenêtre de la figure 11 s'ouvre, à partir de laquelle vous pouvez sélectionner les dimensions de l'espace de travail et insérer des données dans le cartouche.

Positionnement des composants

Passons maintenant à la première phase de la réalisation du circuit imprimé (layout) qui est le positionnement des composants sur le circuit.

Tout d'abord, nous devons définir les limites du contour de la carte électronique que nous voulons réaliser.

Pour cela, sélectionnez l'élément « **Edge.Cuts** » dans le menu déroulant de sélection des couches (ou des calques courants) situé en haut de la barre du projet, comme indiqué en figure 12.

À ce stade, il est possible de dessiner le contour du circuit imprimé de la carte électronique à l'aide des outils de dessin présents dans la barre d'outils.

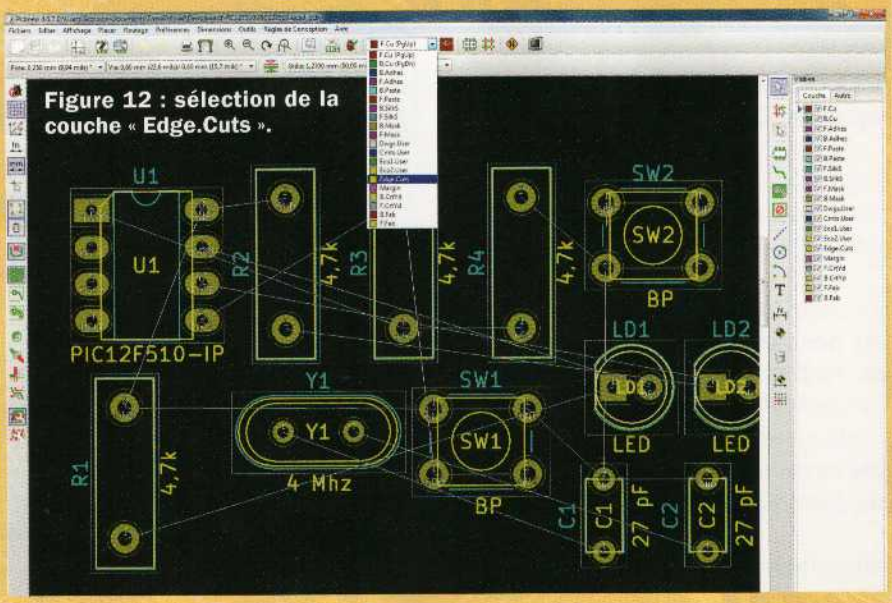
La figure 13 illustre tous les outils de dessin disponibles pour l'opération.

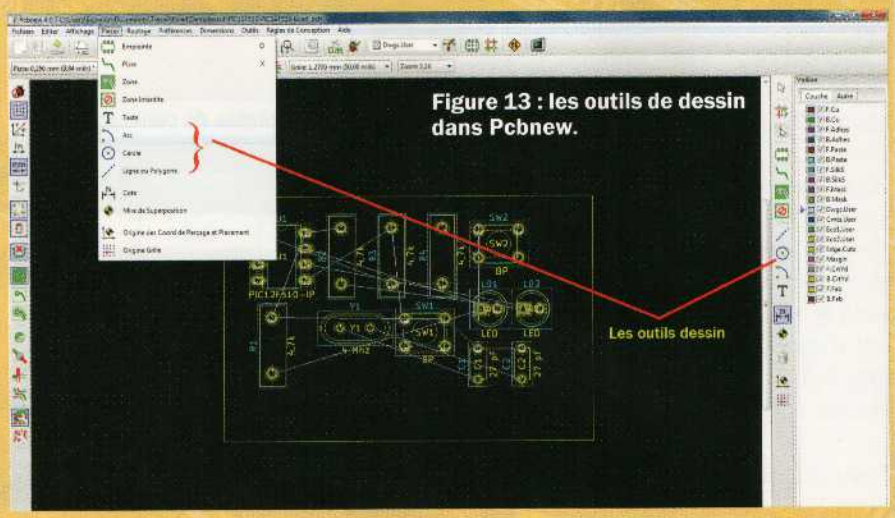
Nous utilisons l'outil de dessin de lignes et nous dessinons un contour rectangulaire d'environ 60 x 40 mm (les dimensions n'ont pas d'importance car il s'agit d'un exercice).

Pour mesurer les dimensions du contour que nous venons de tracer, nous pouvons utiliser l'outil « **Ajout de côtes** » représenté par un « **N** » entre **2 lignes verticales** et qui se trouve en dessous du groupe d'outils de dessin.

Si vous avez suivi correctement toutes ces étapes, le résultat devrait être similaire à l'image de la figure 14.

À ce stade, les composants doivent être positionnés à l'intérieur du contour du circuit imprimé, en les déplaçant éventuellement un par un.





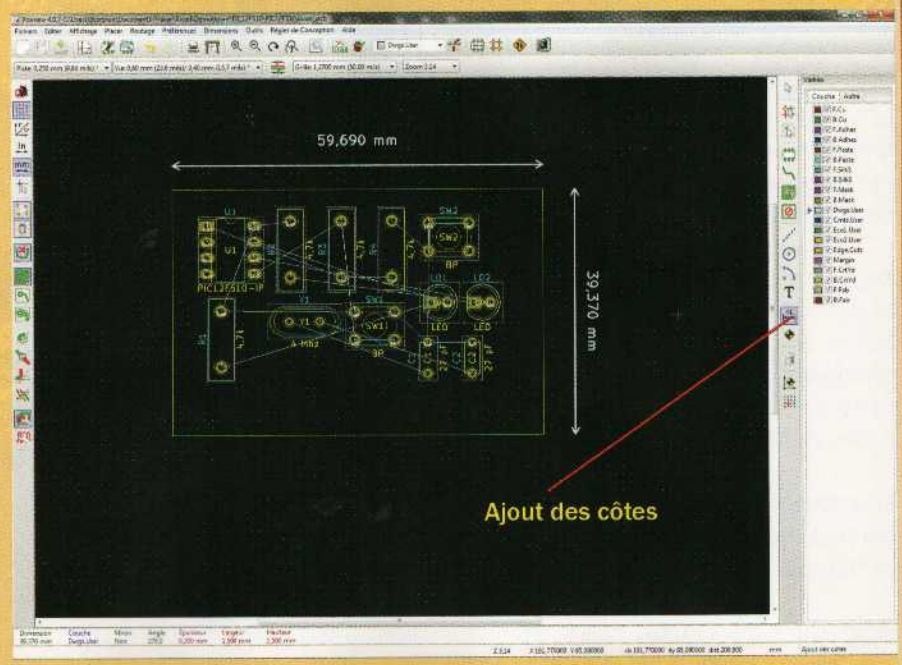


Figure 14 : dessin du contour du circuit imprimé et ajout des côtes (dimensions du circuit).

Pour déplacer un composant, par exemple U1, il suffit de cliquer avec le bouton droit de la souris sur l'empreinte de U1 et de sélectionner l'option « **Empreinte U1 sur F.Cu** » → « **Déplacer** » (voir la figure 15).

Il existe un ensemble infini de variantes de positionnement, il est clair que la suggestion générale est de positionner les composants de manière intelligente, afin de simplifier au maximum la phase de routage mais aussi de tenir compte des contraintes mécaniques (mise en boîtier).

Une solution possible est celle représentée en figure 16.

NB : pour plus de clarté, il est judicieux de ne pas afficher le chevelu. Pour cela, cliquez sur la 7ème icône (« Ne pas montrer le chevelu général ») de la barre des options de l'affichage (côté gauche) en partant du haut.

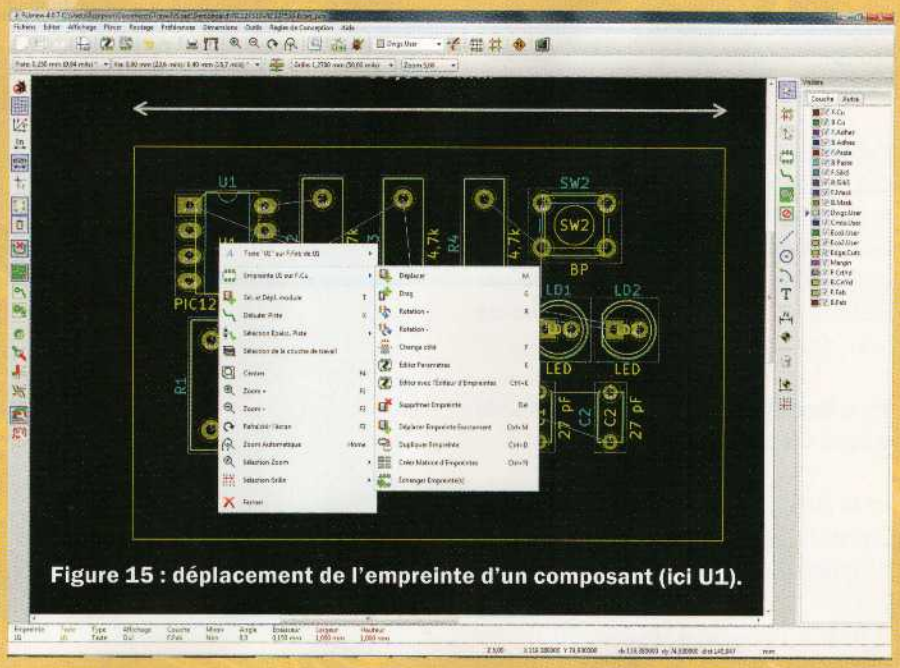


Figure 15 : déplacement de l'empreinte d'un composant (ici U1).

Routage des pistes et des vias

Passons maintenant à la véritable phase de routage du circuit imprimé de la carte. Avant de commencer le routage, il est conseillé de sauvegarder le projet, afin de pouvoir revenir facilement au niveau du chevelu.

Nous devons d'abord sélectionner une couche de cuivre du circuit imprimé.

Nous avons l'intention de fabriquer un circuit imprimé double face, c'est-à-dire constitué de pistes de cuivre sur la **partie inférieure** (couche « **bottom** ») et sur la **partie supérieure** (couche « **top** ») du circuit imprimé.

Pour cela, nous sélectionnons la couche désirée dans le menu de sélection des couches (calques) en haut à droite de la barre du projet.

Sélectionnons la couche « **B.Cu (PgDn)** » (de couleur verte) et commençons à tracer la piste de l'alimentation, en joignant tous les nœuds aux points GND et 5V (en ayant préalablement défini les règles de conception, les pistes appartenant à ces nœuds auront automatiquement une largeur

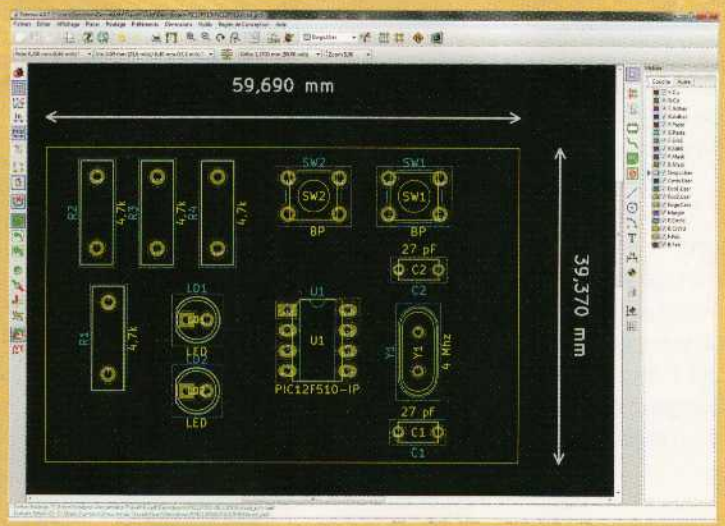


Figure 16 : exemple de positionnement des composants, vous pouvez les disposer autrement selon votre convenance.

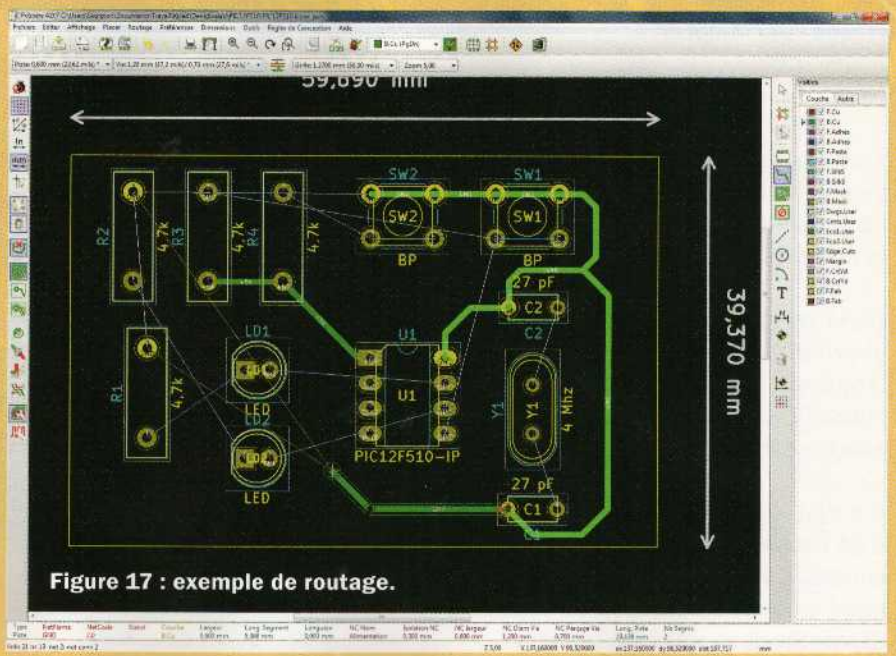


Figure 17 : exemple de routage.

de 0.6 mm car elles appartiennent au groupe « Alimentation » défini dans le tableau 1).

Pcbnew nous aide dans le routage des pistes entre les différents nœuds, il nous empêche de connecter les nœuds de manière différente de celle définie dans la « netlist » générée par Eeschema. Il est donc impossible de faire des erreurs.

Vous devriez obtenir un circuit similaire à celui visible en figure 17.

Ici le circuit est partiellement routé, en bas de la figure vous pouvez voir la piste en cours de routage au niveau de C1.

Il existe plusieurs façons de router un circuit, vous le ferez certainement d'une autre manière.

Continuons le routage avec les nets du groupe « Signal », dont les largeurs sont inférieures à celles du groupe précédent.

Effectuons le routage des boutons poussoir et de l'oscillateur, en changeant éventuellement de couche, c'est-à-dire en passant sur la couche supérieure (« **top** ou « **F.Cu (PgUp)** » de couleur rouge), de manière à ce que les pistes ne se croisent pas.

Comme nous avons défini les règles de conception, nous remarquons que la largeur des pistes du groupe « Signal » est bien inférieure à celle des pistes du groupe « Alimentation ».

Le résultat est illustré en figure 18.

À ce stade, nous avons légèrement modifié le routage au niveau de la LED LD2 (voir la figure 20) afin d'introduire la notion de **via**.

Dans ce cas, le substrat du circuit imprimé est percé et le trou ainsi formé est métallisé pour garantir une connexion électrique entre les deux couches (inférieure et supérieure).

Il s'agit d'un élément fondamental d'un circuit imprimé ayant plus d'une couche, car il augmente considérablement la facilité d'effectuer le routage des pistes de la carte électronique.

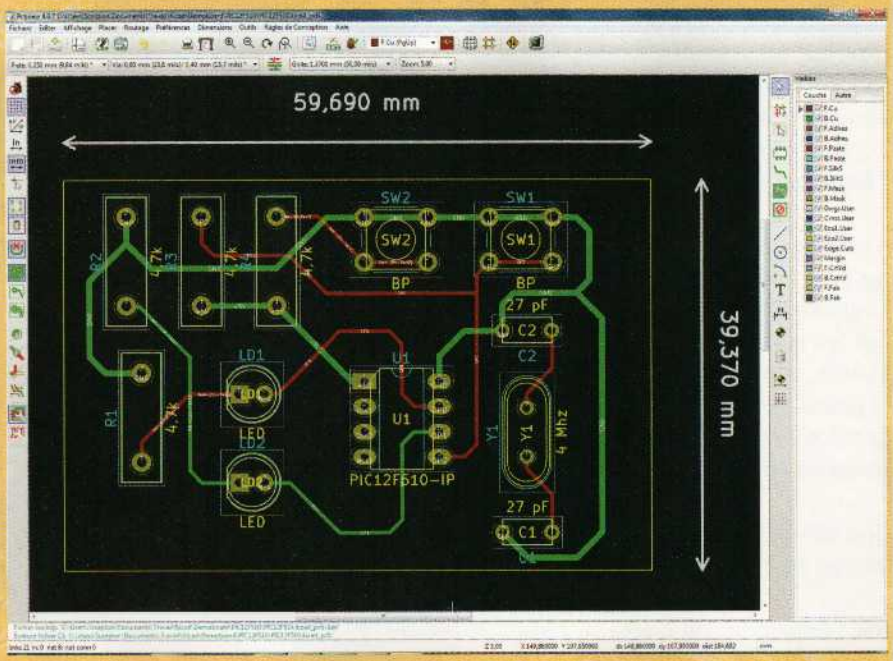


Figure 18 : traçage des pistes du circuit sur les 2 couches (en vert la couche inférieure et en rouge la couche supérieure).

Figure 19 : insertion d'un via pour l'interconnexion entre les deux couches.

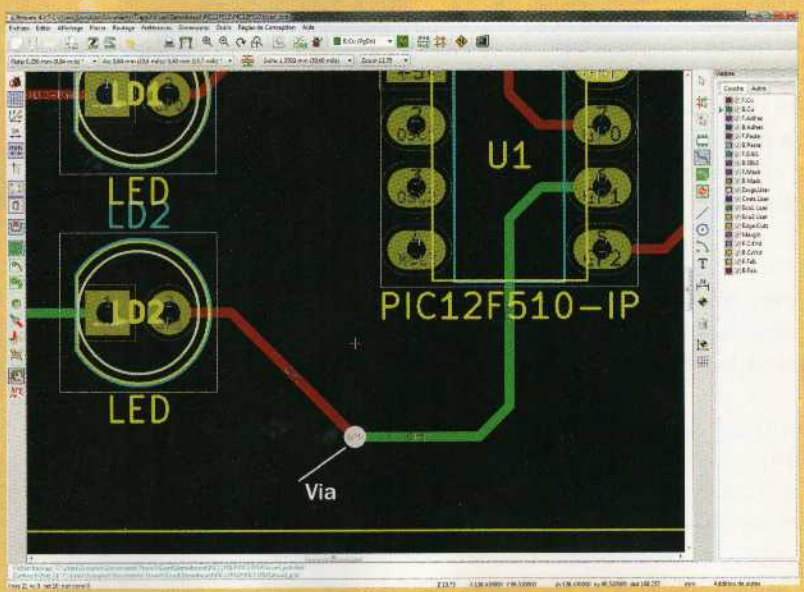


Figure 20 : le via permet la continuité électrique entre la couche inférieure et la couche supérieure du cuivre.

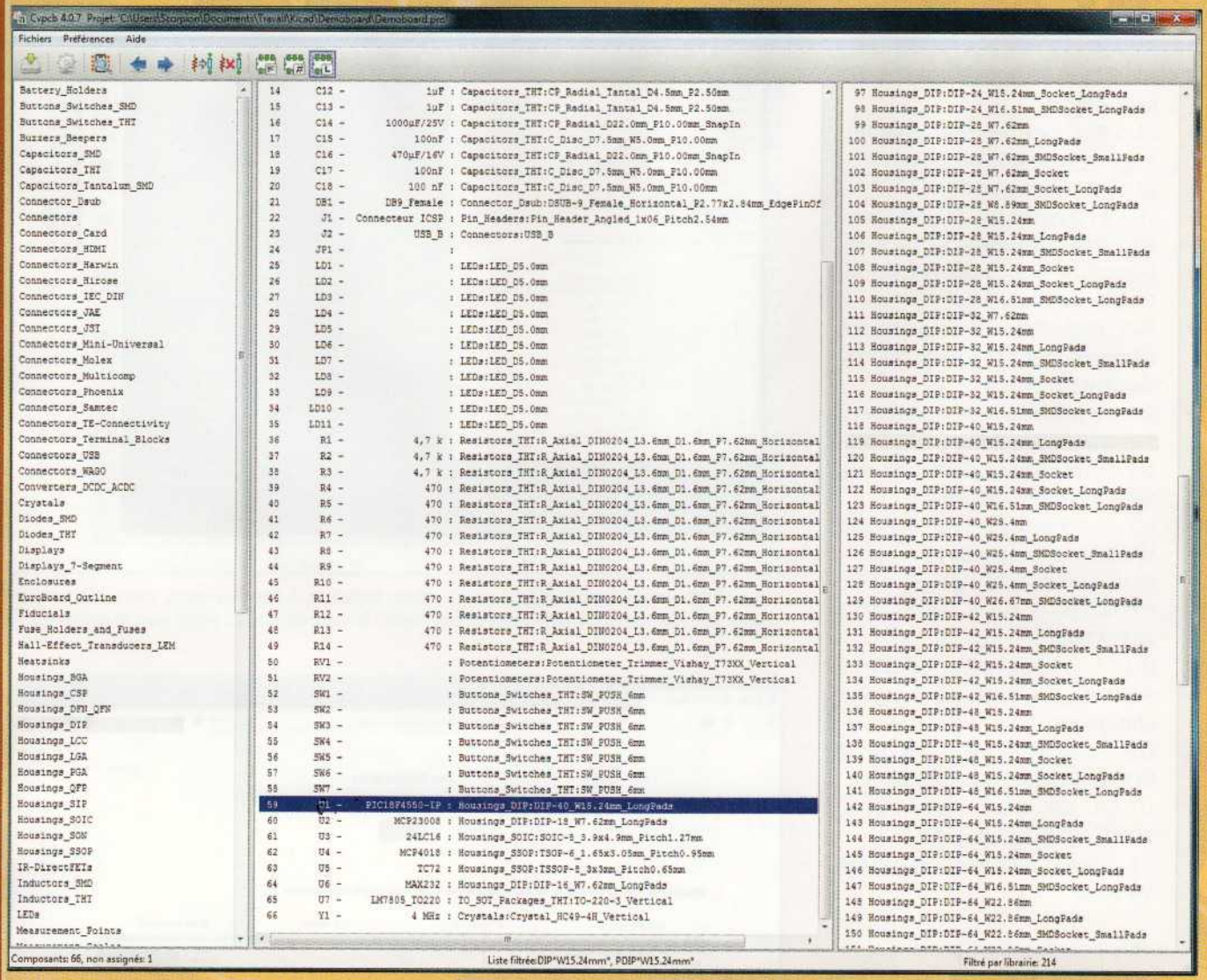


Figure 21 : fenêtre d'association des composants avec leurs empreintes du projet Demoboard.

Pour insérer un via dans Pcbnew, procédez de la manière suivante :

- commencez par router un segment de piste à partir de la cathode de LD2 en sélectionnant la couche « F. Cu (PgUp) » ;
- lorsque vous atteignez le point où vous voulez placer le via sur le circuit, cliquez sur le bouton droit de la souris ;
- à partir du menu déroulant qui s'ouvre, cliquez sur l'option « **Placer Via Traversant** », comme illustré en figure 19 ;
- le via est placé automatiquement et le reste du segment de piste à router change automatique de couche (« B.Cu (PgDn) »).

Nous venons donc de relier la cathode de la LED LD2 à la broche 6 du PIC12F510 en partant de la couche supérieure du cuivre et en passant sur la couche inférieure à travers le via qui sert d'interconnexion entre les deux couches. Le résultat de cette opération est visible en figure 20, le via est représenté par un cercle plein de couleur grise.

Notez que l'utilisation du via dans notre cas n'est pas obligatoire, car nous aurions pu router directement la piste en restant sur la couche supérieure (en rouge).

Projet pratique : association des composants et importation

Revenons maintenant à notre projet pratique à base de PIC18F4550, en procédant à l'association des composants et, par conséquent, à l'importation de la « netlist » sous Pcbnew.

Tout d'abord, passons à l'association des composants. La fenêtre Cvpcb relative au projet pratique est représentée en figure 21.

Pour l'association des composants, nous utilisons les règles suivantes :

- AFF1 : Displays:WC1602A ;
- B1 : TerminalBlocks_Phoenix_MKDS1.5-2pol ;
- C1, C2 : Capacitors_THT:C_Disc_D4.7mm_W2.5mm_P5.00mm ;
- C3, C4, C5, C6, C7, C8, C9, C15, C17, C18 : Capacitors_THT:C_Disc_D7.5mm_W5.0mm_P10.00mm ;
- C10, C11, C12, C13 : Capacitors_THT:CP_Radial_Tantal_D4.5mm_P2.50mm ;
- C14, C16 : Capacitors_THT:CP_Radial_D22.0mm_P10.00mm_SnapIn ;

- DB1 : Connector_Dsub:D-SUB-9_Female_Horizontal_P2.77x2.84mm_EdgePinOffset9.40mm ;
- J1 : Pin_Headers:Pin_Header_Angled_1x06_Pitch2.54mm ;
- J2 : Connectors:USB_B ;
- LD1 à LD11 : LEDs:LED_D5.0mm ;
- R1 à R14 : Resistors_THT:R_Axial_DIN0204_L3.6mm_D1.6mm_P7.62mm_Horizontal ;
- RV1, RV2 : Potentiometers:Potentiometer_Trimmer_Vishay_T73XX_Vertical ;
- SW1 à SW2 : Buttons_Switches_THT:SW_PUSH_6mm ;
- U1 : Housings_DIP:DIP-40_W15.24mm_LongPads ;
- U2 : Housings_DIP:DIP-18_W7.62mm_LongPads ;
- U3 : Housings_SOIC:-SOIC-8_3.9x4.9mm_Pitch1.27mm ;
- U4 : Housings_SSOP:T-SOP-6_1.65x3.05mm_Pitch0.95mm ;
- U5 : Housings_SSOP:T-SOP-8_3x3mm_Pitch0.65mm ;
- U7 : TO_SOT_Packages_THT:TO-220-3_Vertical ;
- Y1 : Crystals:Crystal_HC49-4H_Vertical.

Une fois l'association terminée, cliquez sur la 1ère icône en haut à gauche nommée « **Sauve l'association des empreintes** » et représentée par une flèche verte. Ensuite, vous devez générer la « netlist » dans Eeschema » et enfin vous pouvez lancer « Pcbnew ».

À ce stade, les schémas des composants sont correctement associés à leurs empreintes. Il est alors possible d'importer la « netlist » dans Pcbnew afin d'effectuer le routage de la carte.

Ouvrez Pcbnew et utilisez l'icône d'importation de la « netlist » (voir la figure 3).

Si tout a été fait correctement, vous devez obtenir un résultat semblable à celui de la figure 22 avec les composants et le chevelu concentrés dans la zone centrale de l'espace de travail.

Il se peut que vous n'ayez pas toutes les empreintes des composants. Pour cela nous vous conseillons de télécharger le fichier suivant : « kicad-footprints-master.zip » à l'adresse suivante :

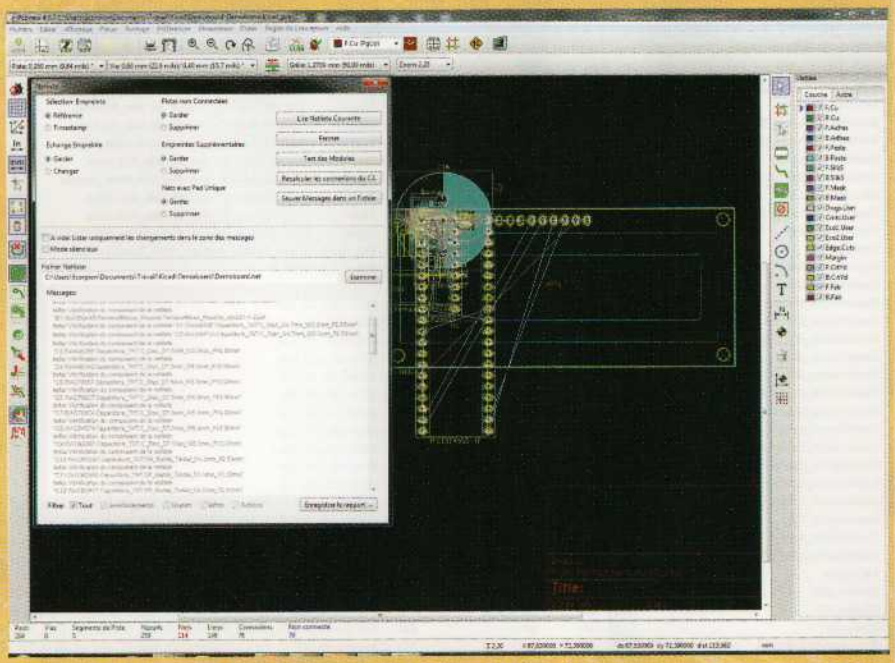


Figure 22 : importation de la « netlist » dans Pcbnew. À l'arrière-plan, vous apercevez dans la zone centrale de l'espace de travail les composants ainsi que le chevelu.

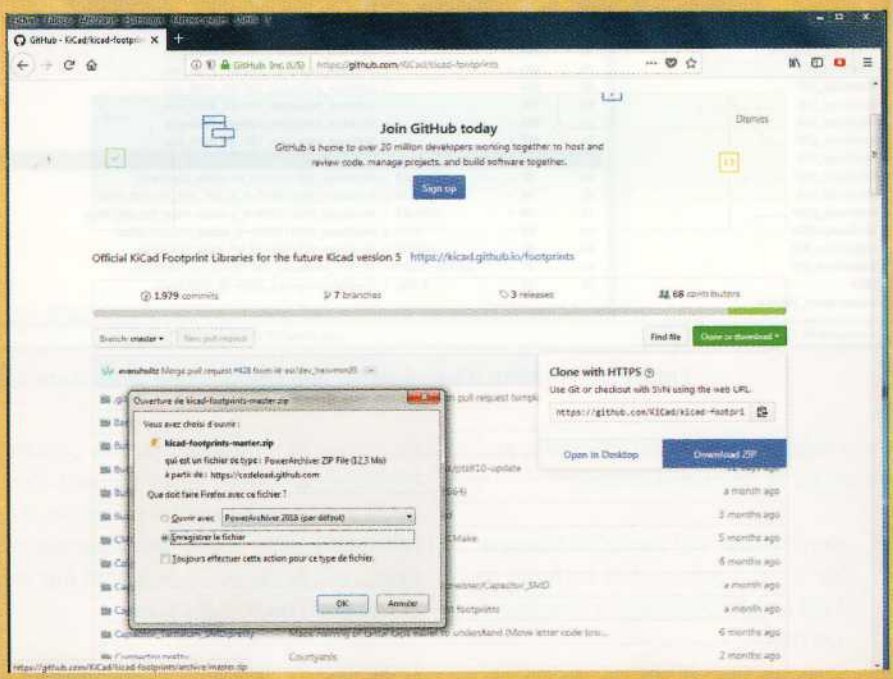


Figure 23 : téléchargement du fichier « kicad-footprints-master.zip ».

https://github.com/KiCad/kicad-footprints.

Pour télécharger le fichier sur la page web, cliquez sur le bouton vert « **Clone or download** ». Le bouton bleu « **Download ZIP** » apparaît alors, cliquez dessus. Une fenêtre apparaît vous invitant à enregistrer le dossier sur votre ordinateur (voir la figure 23).

Une fois le fichier téléchargé, dézippez-le dans un dossier de votre choix.

Maintenant, vous allez intégrer la librairie des connecteurs « Dsub » (pour le connecteur Dsub-9), qui ne se trouve pas par défaut dans Kicad. Ouvrez le schéma et lancez l'outil « CvPcb ».

Dans le menu supérieur, cliquez sur « **Préférences** » → « **Librairies d'Empreintes** ».

Une fenêtre s'ouvre, semblable à celle de la figure 24. Cliquez sur le bouton « **Ajouter avec l'Assistant** ». Une seconde fenêtre s'ouvre, cochez l'option « **Fichiers sur mon ordinateur** » puis cliquez sur le bouton « **Next** ».

Cherchez alors votre dossier décompressé qui doit s'appeler « **kicad-footprints-master** », et sélectionnez-le pour visualiser son contenu (le dossier sélectionné apparaît en bleu sur la figure 25).

Sélectionnez alors le dossier « **Connector_Dsub.pretty** » puis cliquez sur le bouton « Next » (voir la figure 26).

La fenêtre de la figure 27 apparaît vous invitant à confirmer l'intégration de la nouvelle librairie dans KiCad.

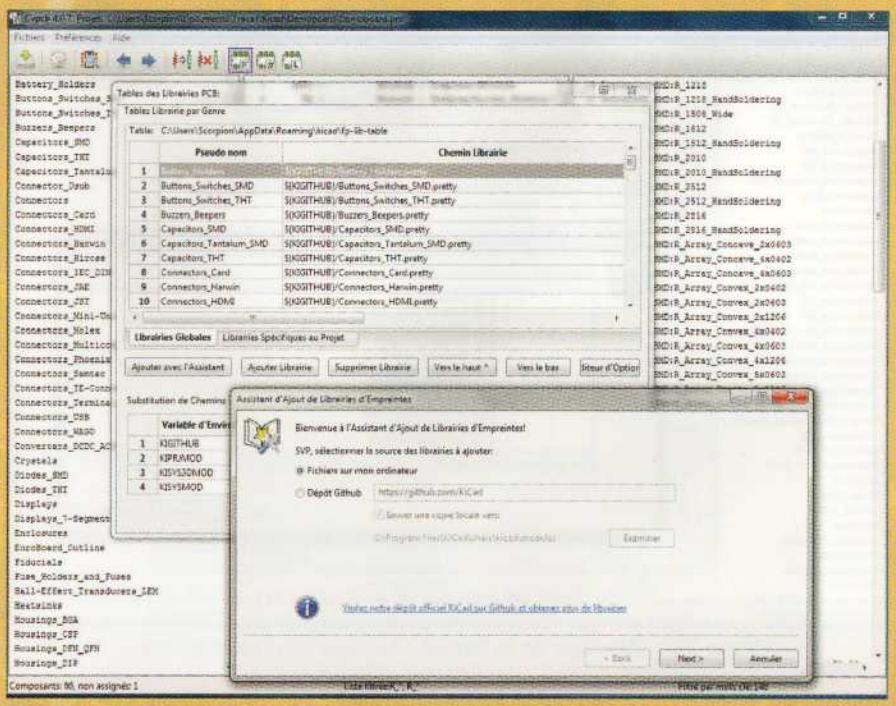


Figure 24 : ajout d'une librairie dans KiCad à l'aide de l'assistant.

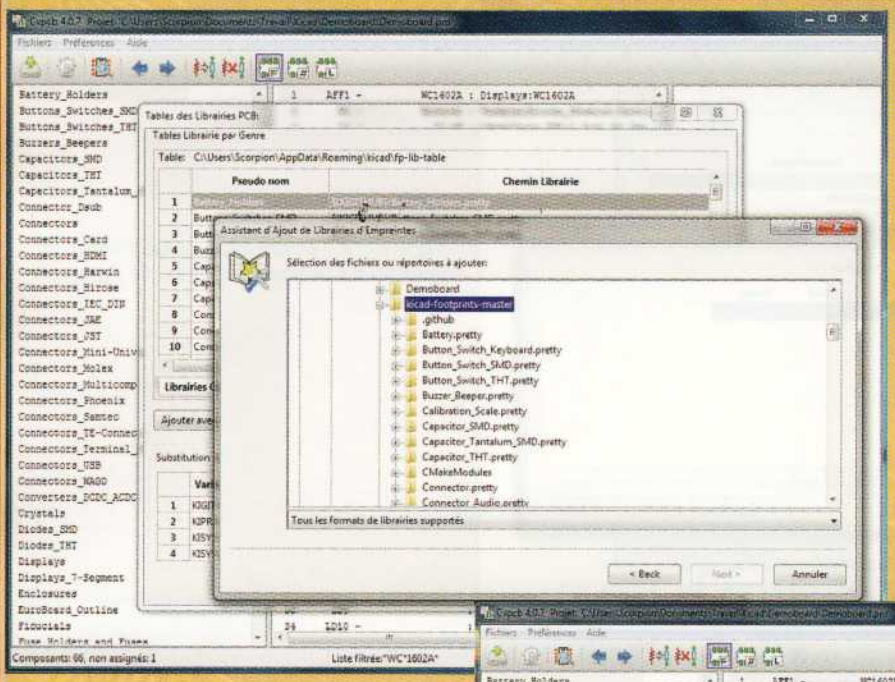


Figure 25 : sélection du dossier « kicad-footprints-master ».

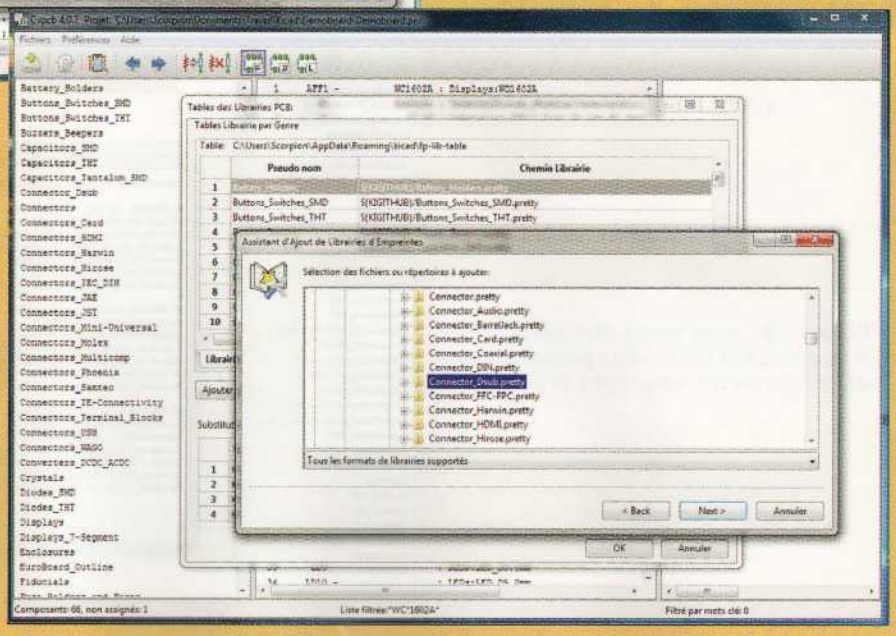


Figure 26 : sélection de la librairie « Connector_Dsub.pretty ».

Cliquez sur le bouton « **Next** » et dans la fenêtre suivante cochez l'option « **A la configuration globale des librairies** », de manière à pouvoir **utiliser la nouvelle librairie pour d'autres projets**. Cliquez sur le bouton « **Finish** » pour terminer l'opération.

Vous devez voir apparaître le chemin de la nouvelle librairie dans la fenêtre « **Table des librairies PCB :** ».

Maintenant, vous pouvez associer le connecteur DB1 à l'empreinte « Connector_Dsub:DSUB-9 ».

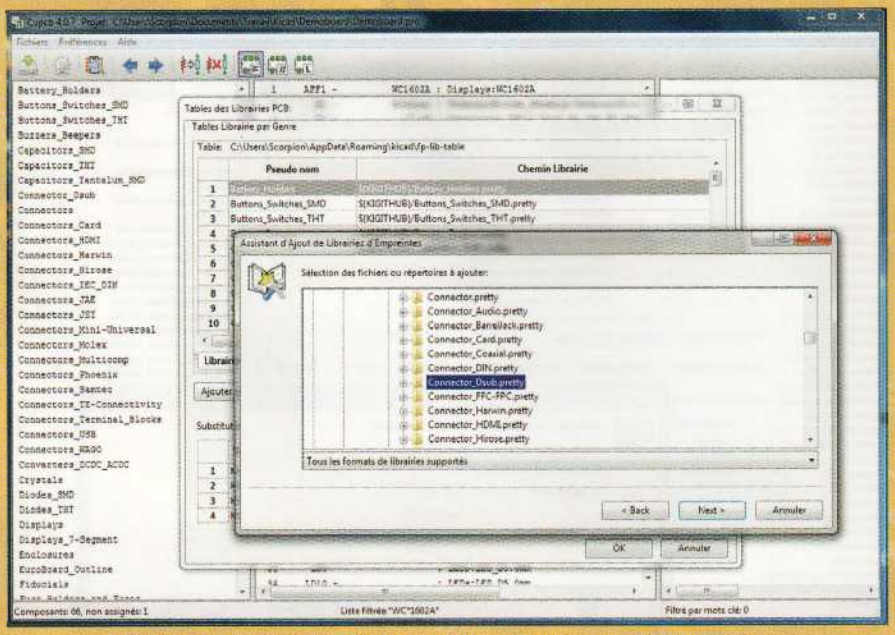


Figure 26 : sélection de la librairie « Connector_Dsub.pretty ».

Procédez de la même manière dans le cas où des empreintes d'autres composants ne soient pas disponibles par défaut dans KiCad.

Vous pouvez aussi intégrer toutes les librairies du dossier « kicad-footprints-master » en sélectionnant tous les dossiers portant l'extension « **XXXX.pretty** ». Attention l'opération peut prendre un certain temps.

NB : lorsque vous lancez CvPcb, **vous devez être connecté à internet** car sinon vous pouvez avoir un message d'erreur comme visible en figure 28.

Dans la prochaine leçon, nous aborderons l'éditeur d'empreinte intégré dans Pcbnew qui nous permettra de créer nos propres empreintes de composants. Nous commencerons aussi le routage de notre Demoboard.

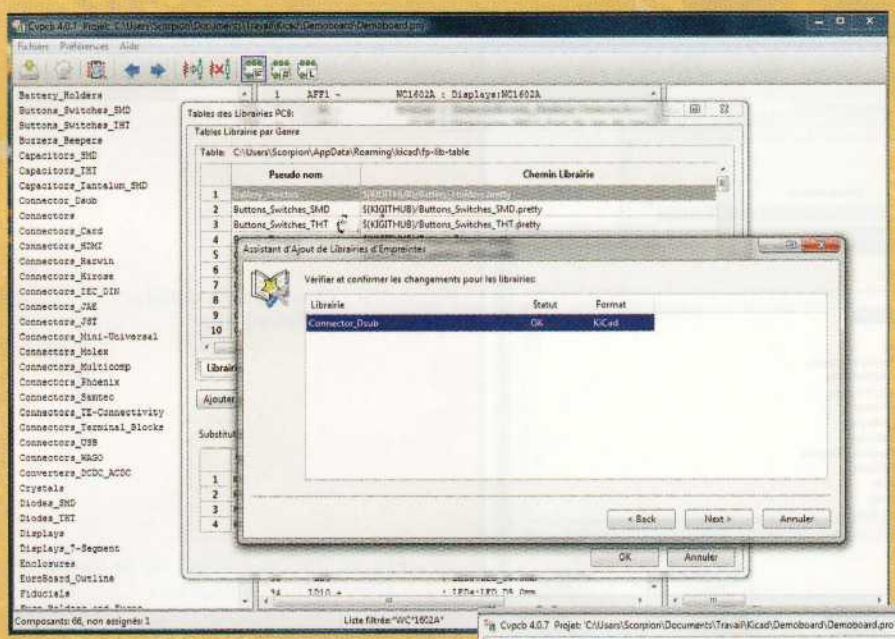


Figure 27 : confirmation de l'intégration de la nouvelle librairie dans KiCad.

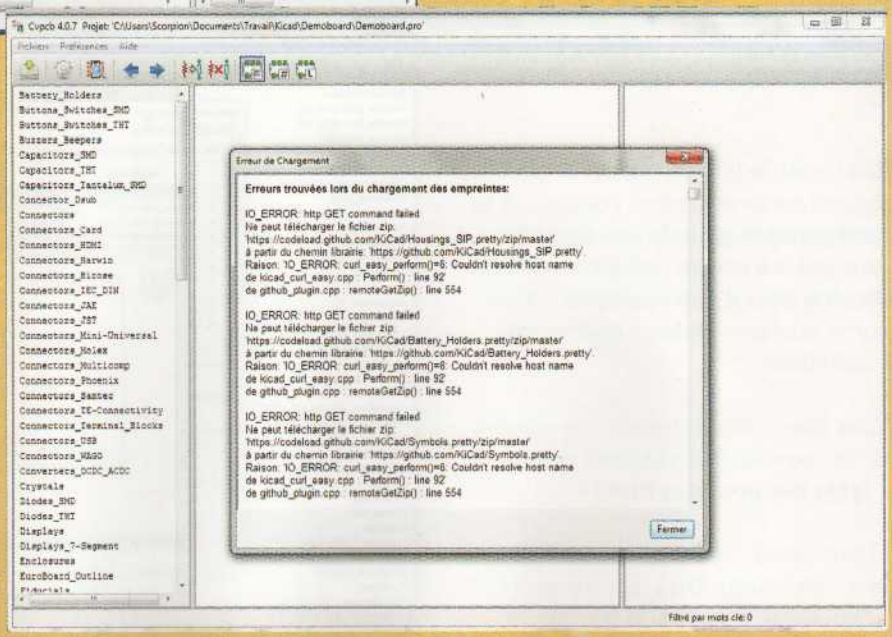
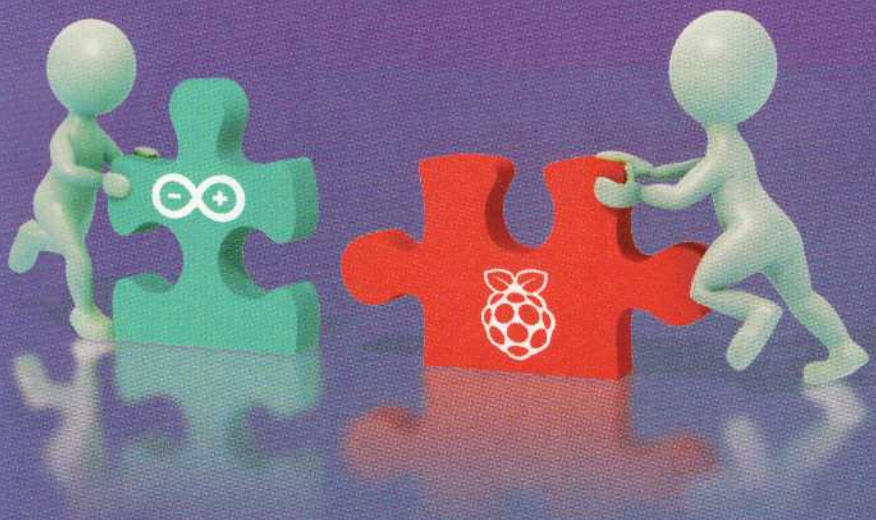


Figure 28 : vous devez être connecté à internet, car sinon vous pouvez avoir un ou plusieurs messages d'erreurs.



Dans cet article nous allons installer un logiciel spécifique, vérifier le bon fonctionnement de la carte et créer un serveur Web. Nous terminerons par un exercice de programmation. Troisième partie.

Apprenons à utiliser RANDA

Troisième partie

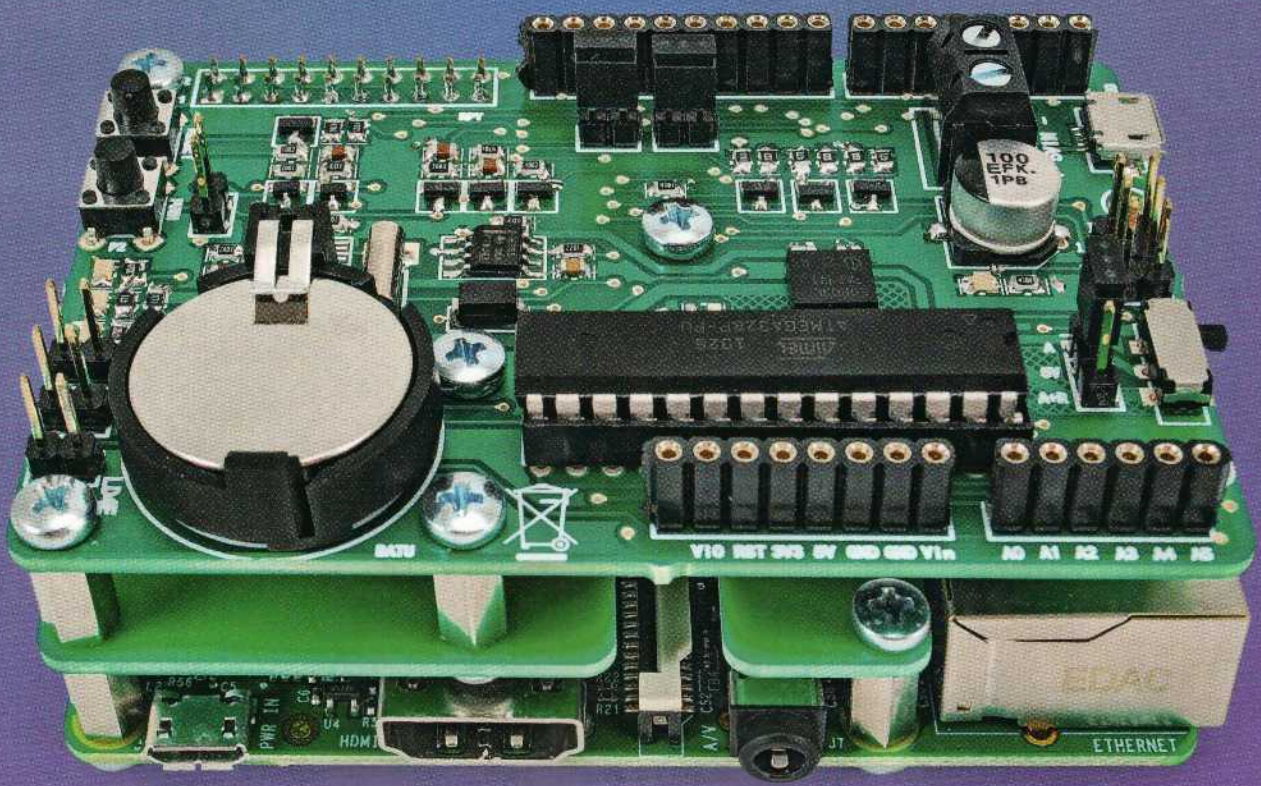
de Daniele DENARO

Si vous avez manqué les articles précédents, nous résumons brièvement les caractéristiques de la carte « **RandA** ». Il s'agit d'une carte qui permet l'**intégration physique et fonctionnelle d'un RaspberryPi avec une carte Arduino**, permettant ainsi d'utiliser la large gamme de cartes Arduino en combinant l'énorme potentiel du RaspberryPi.

En outre, « RandA » comprend également une gestion intelligente de l'alimentation et une horloge temps réel (RTC).

En figure 1a, vous pouvez voir le schéma synoptique de la carte « RandA », tandis qu'en figure 1b sont décrites les commandes et les connexions.

Afin de gérer convenablement la carte, nous mettons à disposition un ensemble de logiciels pouvant être copiés sur une carte SD afin d'être insérée dans un RaspberryPi. Par conséquent, la première chose à faire après avoir réalisé la carte « RandA » et l'avoir couplée au RaspberryPi, est d'installer



sur la carte SD les programmes nécessaires au fonctionnement du système.

En fait, sans cela, le hardware (matériel), l'horloge RTC et même la carte Arduino ne pourraient pas communiquer avec le RaspberryPi.

Nous avons inclus dans le package logiciel le programme à installer sur votre PC pour modifier l'IDE d'Arduino afin d'utiliser les ports distants et pouvoir téléverser le sketch dans « RandA » connectée au réseau.

L'archive contenant les fichiers d'installation se trouve dans le répertoire : « **/home/pi/RandA** ».

Le programme est téléchargeable sur notre site dans le sommaire détaillé de la revue (en bas de la page web). Il contient les fichiers du projet et se présente sous la forme de 2 programmes au format compressé (.zip).

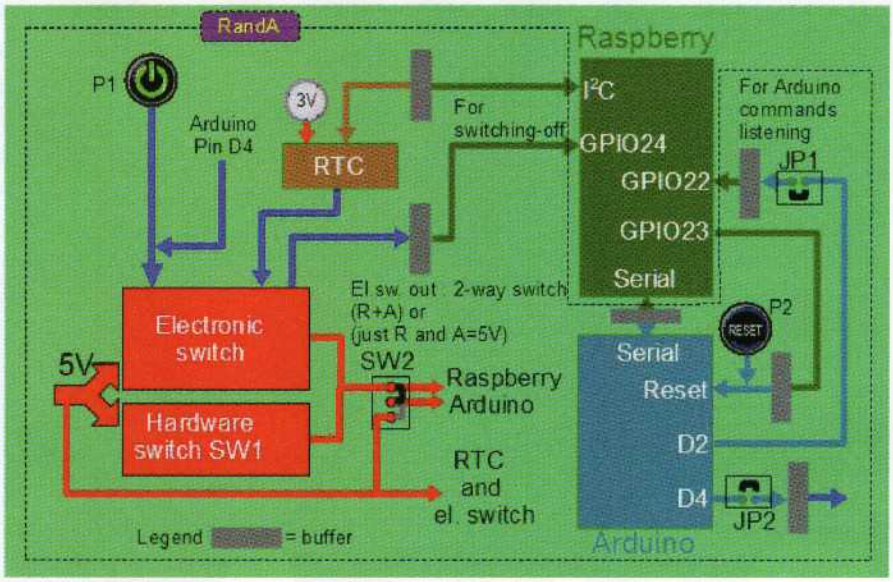


Figure 1a : schéma synoptique de la carte « RandA ».

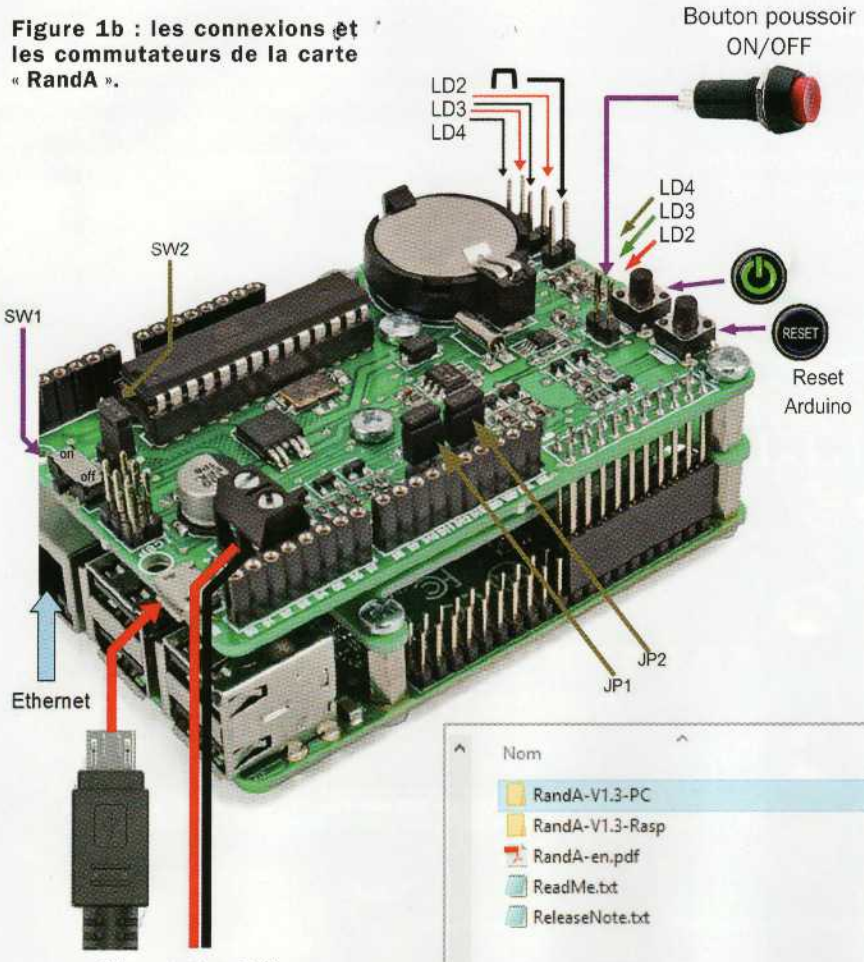
L'un est pour votre PC et l'autre pour le RaspberryPi sur lequel la carte « RandA » est accouplée. Vous y trouverez un manuel en anglais (voir la figure 2).

La procédure à suivre pour installer le logiciel est la suivante :

1. **extraire l'archive** du programme pour le PC. Une fois extraits, les fichiers doivent être placés dans le dossier « lib » de l'IDE d'Arduino (utiliser de préférence la version 1.0.5) ;
2. pour des raisons de sécurité, **renommer les 3 fichiers** que vous remplacez. L'IDE Arduino va maintenant être modifié. Terminez l'opération en extrayant la librairie « RAComm » contenue dans le dossier « libraries » et en la plaçant dans le dossier « libraries » de l'IDE modifié ;
3. **extrayez l'archive « RandAinstall. tar.gz »** et le fichier « **RandAinstall.sh** » de l'archive pour le RaspberryPi.

À ce stade, vous devez utiliser le RaspberryPi. Puisque, lors de l'installation de Raspbian (le système d'exploitation

Figure 1b : les connexions et les commutateurs de la carte « RandA ».



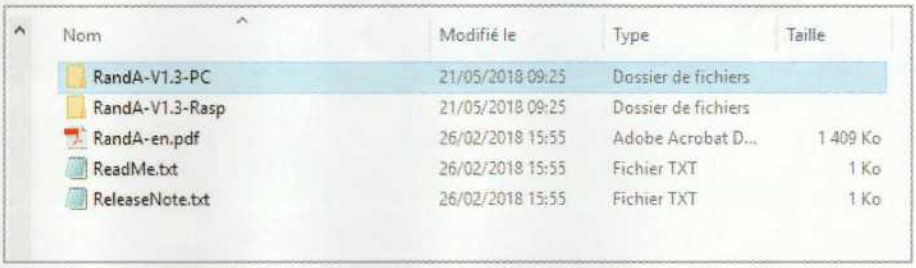


Figure 2 : programmes d'installation de « RandA-VI.3.zip ».

du RaspberryPi) le **serveur SSH est configuré et activé par défaut**, nous pouvons nous connecter au RaspberryPi via le réseau LAN avec un programme qui gère ce protocole.

Dans cet article, nous supposons que vous utilisez « **MobaXterm** », qui contient également un serveur « **X window** » et une **fenêtre FTP plus pratique** que celle de la console Linux (reportez-vous aux articles précédents).

Pour cette raison, nous devons identifier l'adresse réseau d'Arduino. Nous pouvons l'identifier via le routeur ou via un utilitaire qui analyse le réseau comme par exemple « Advanced Ip scanner » (vous pouvez le trouver sur le web).

Ouvrons une session SSH avec ce dernier (voir la figure 3).

Nous apercevons immédiatement qu'à côté de la console se trouve la fenêtre FTP.

À ce stade, nous devons juste **transférer les deux fichiers sur le RaspberryPi**. Nous pouvons les mettre dans le dossier « **/home/pi** ».

Avant de lancer le script « **RandAinstall.sh** », nous devons le rendre exécutable à l'aide de la commande :

```
sudo chmod 777 RandAinstall.sh
```

Maintenant, tout ce qui nous reste à faire est d'exécuter le script en utilisant la commande suivante (il est supposé être dans le dossier « /home/pi ») :

```
./RandAinstall.sh
```

Rappelez-vous que, puisque le script installe l'IDE Arduino pour Linux et le programme « codesblocks » pour Raspbian, **vous devez avoir une connexion Internet active**.

Dans le cas où vous ne disposez pas de connexion Internet, vous pouvez installer « codesblocks » et l'IDE Arduino dans un deuxième temps, à l'aide des commandes :

```
sudo get-apt install codeblocks
sudo get-apt install arduino
```

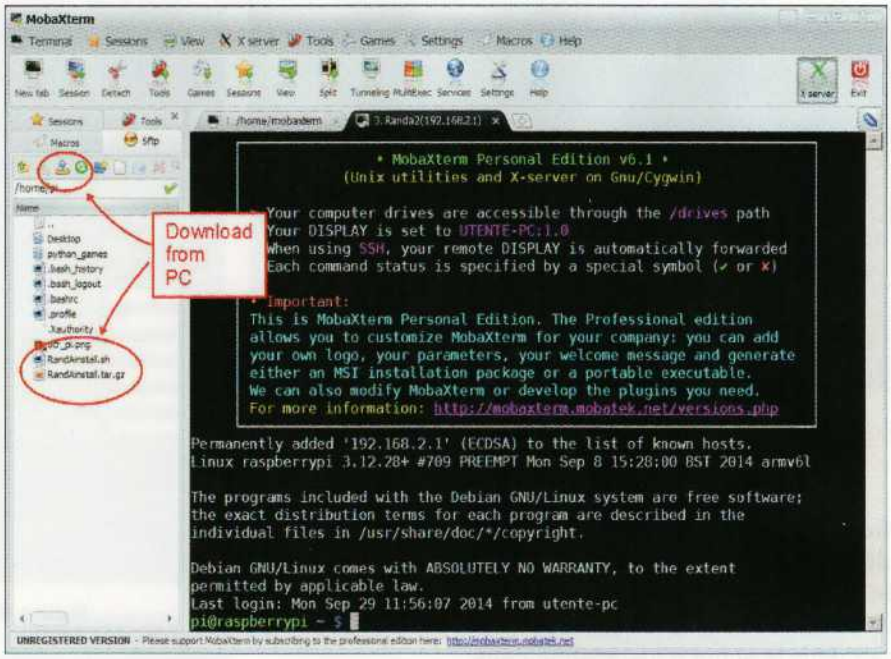


Figure 3 : la console « MobaXterm ».

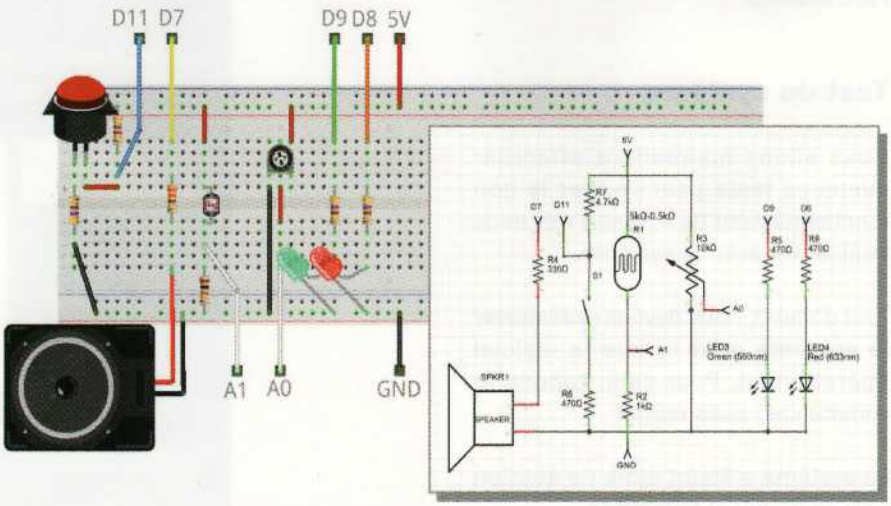


Figure 4a : tests avec la plaque d'essai.

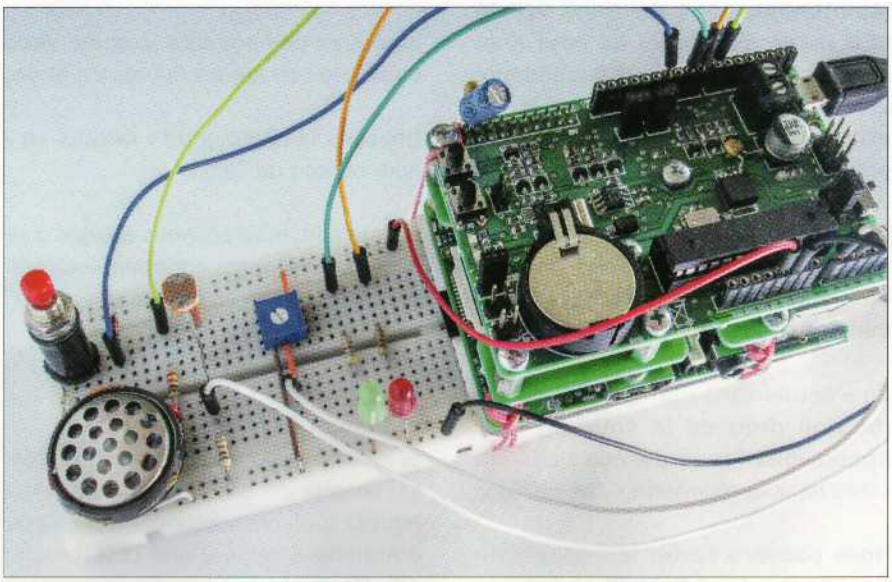


Figure 4b : photo de la plaque d'essai reliée à la carte « RandA ».

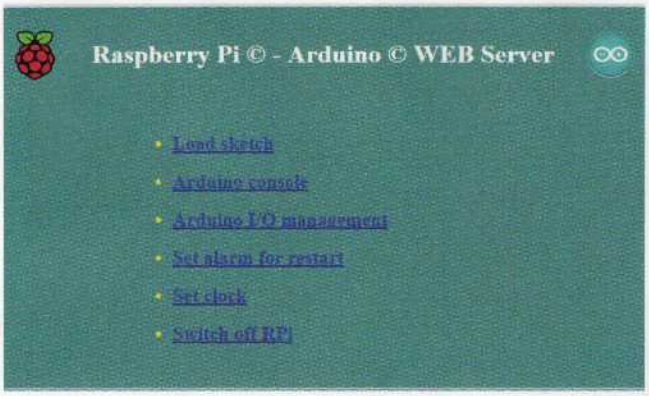


Figure 5 : menu général du serveur Web Arduino.

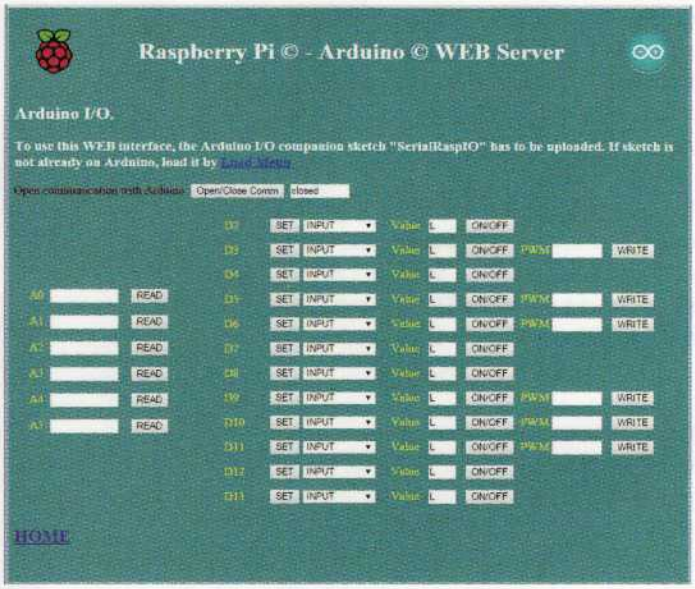


Figure 6 : l'écran des entrées/sorties.

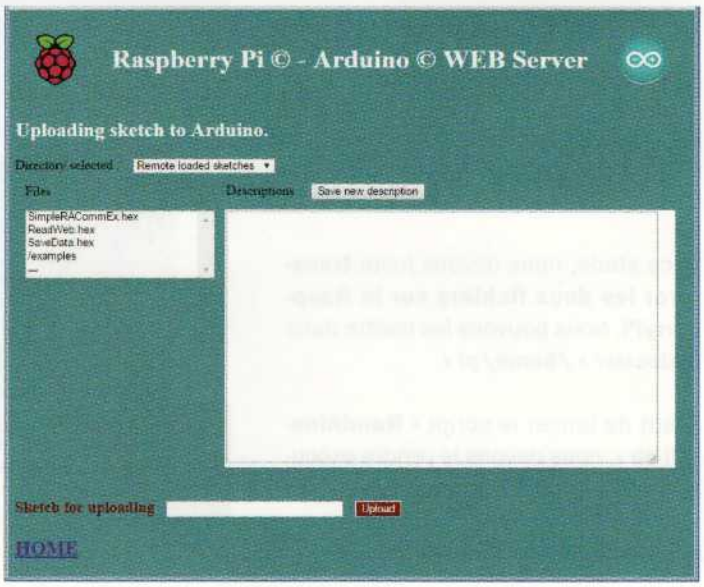


Figure 7 : chargement du sketch.

Cependant, après l'installation d'Arduino vous devez modifier manuellement l'IDE du RaspberryPi en vous référant aux trois lignes que vous trouvez dans le script d'installation dans la section relative à la modification de l'IDE Arduino.

Test du système

Nous allons maintenant effectuer quelques tests pour vérifier le bon fonctionnement de « RandA » et nous familiariser avec ce système.

Tout d'abord, nous devons **redémarrer le système** pour rendre le logiciel opérationnel. Pour cela, tapons la commande : **sudo reboot**

Le système s'éteint alors (la session SSH se ferme) et redémarre.

Maintenant, nous devrions voir la **LED jaune s'éteindre au bout d'un moment**. En fait, elle reste allumée pendant la phase de démarrage et son extinction signifie que le système est prêt à fonctionner.

Ouvrons de nouveau la session SSH, nous devrions voir le nouveau message de bienvenue, avec la liste des commandes spécifiques pour « RandA ».

En « actualisant » la fenêtre « **sftp** » (bouton droit de la souris), nous apercevons les nouveaux dossiers créés dans le répertoire « **/home/pi** ».

Nous pouvons tester la « collaboration » entre RaspberryPi et Arduino en utilisant une plaque d'essai et quelques composants, comme vous pouvez le voir sur les figures 4a et 4b.

Utilisons la commande « **ArduIO –h** » pour obtenir de l'aide.

Maintenant, nous pouvons essayer d'allumer la LED rouge en tapant les commandes selon la séquence suivante :

```
ArduIO -set 8 out
ArduIO -wrd 8 1
```

Si vous ne disposez pas de plaque d'essai comme celle de la figure 4a, vous pouvez toujours tester l'entrée/sortie d'Arduino à l'aide d'une LED (broche 13). Dans ce cas, vous devez taper les commandes suivantes :

```
ArduIO -set 13 out
ArduIO -wrd 13 1
```

ArduIO utilise le sketch « SerialRasp ». S'il n'est pas sur Arduino, installez-le, mais vous devez redémarrer à nouveau.

Le sketch utilisé se trouve aussi sous la forme d'un code source dans le dossier « **schetch4cmd** » dans le répertoire « **/home/pi/bin** ».

Pour le désactiver, vous devez exécuter la commande : **ArduIO -wrd 8 0**

Nous pouvons procéder de la même manière pour la LED verte (broche 9). Nous pouvons l'allumer avec une intensité variable grâce à la sortie

PWM (seulement avec la broche 9). Dans ce cas la commande, pour avoir 50/255 soit 1/5 de l'intensité, se présente sous la forme : **ArduIO -wra 9 50**

Nous pouvons lire la valeur de la photorésistance sur A1 (ou via le diviseur de tension sur A0) avec la commande : **ArduIO -rda 1**

Nous envoyons un signal carré à la fréquence de 600 Hz sur D7 avec la commande : **ArduIO -pou 7 600**

Pour arrêter le signal, nous utilisons la commande : **ArduIO -puo 7 0**

Bien évidemment, la puissance sonore est très faible, étant donné que le HP est piloté directement par Arduino.

Si vous désirez un son plus puissant, vous aurez besoin d'un amplificateur BF.

Pour lire la durée d'une impulsion (de la valeur 1 vers la valeur 0) sur D11, nous utilisons la commande :

ArduIO -pin 11 0

Si nous voulons mesurer la durée de la valeur 0, nous devons taper la commande : **ArduIO -pui 11 1**

Si nous utilisons le bouton, nous devons tenir compte du délai d'attente qui est d'une seconde.

L'utilisation du bouton en tant qu'entrée digitale s'effectue grâce à la commande : **ArduIO -rdd 11**

En réalité, nous pouvons dialoguer directement avec Arduino en utilisant le port série :

echo "WD13=1" > /dev/ttyS0

Cette commande allume la LED d'Arduino (broche 13). Si, sur Arduino, le sketch « SerialRasp » est chargé (celui utilisé par la commande « ArduIO »), la commande précédente (du port série) correspond à la commande :

ArduIO -wrd 13 1

Le protocole utilisé par « SerialRasp » est très simple et est décrit dans le source du sketch.

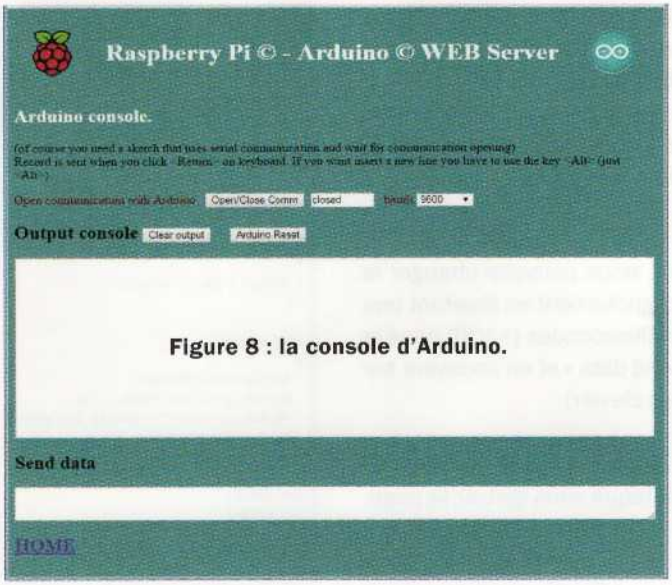


Figure 8 : la console d'Arduino.

Le serveur Web

Essayons maintenant l'accès via le navigateur web. Si vous travaillez en réseau local, entrez l'adresse IP de « RandA » dans votre navigateur et la page d'accueil du serveur apparaîtra.

Si vous venez d'allumer la carte « RandA », soyez patient avant d'accéder à l'adresse. Attendez environ une dizaine de secondes après que le voyant de démarrage jaune se soit éteint afin de permettre au serveur Web de s'initialiser.

Cliquez sur « **RPi&Arduino management** » et le menu général visible en figure 5 apparaîtra.

Vous pouvez utiliser la plaque d'essai précédente pour tester la communication via le web. Pour cela, cliquez sur « **Arduino IO management** » et l'écran de la figure 6 apparaîtra.

Si vous avez déjà testé la plaque d'essai avec les commandes « ArduIO » sur Arduino via le sketch de gestion des broches, vous vous apercevrez que cette application est fondamentalement identique.

Vous devez d'abord **charger l'application** en utilisant la page « **Load sketch** ».

Dans cet écran, la première chose à faire est d'**ouvrir la communication via le port série** (attendez jusqu'à ce qu'elle apparaisse ouverte).

Vous pouvez ensuite utiliser les différents boutons pour configurer, modifier ou lire les valeurs.

Par exemple, pour lire la valeur de la luminosité détectée par la photorésistance, cliquez sur le bouton « READ » correspondant à « A1 ».

Revenons au menu et passons à la page « Load sketch » (voir la figure 7). Dans celle-ci, vous pouvez voir qu'il y a deux boîtes de dialogues principales, une contenant une liste de sketchs et l'autre contenant les descriptions que vous pouvez insérer ou modifier directement.

En réalité, il y a deux listes qui peuvent être sélectionnées via la liste déroulante. En fait, un dossier contient les sketchs créés avec l'IDE distant et l'autre ceux créés avec l'IDE local (sur le RaspberryPi).

Sélectionnons le sketch « **TestSerial.hex** » dans le dossier « **/examples** », la description apparaît.

Maintenant, cliquons sur le bouton « Upload », le message de téléchargement (uploading) réussi devrait apparaître (si quelque chose ne va pas, réinitialisez ou téléchargez de nouveau). À ce stade, nous pouvons l'utiliser simplement.

Revenons maintenant au menu général et sélectionnons le lien « Arduino console » (voir la figure 8).

Nous vérifions d'abord que la **vitesse de communication sélectionnée est de 9600 bauds**, ensuite nous ouvrons le port série. Attendons qu'il soit « ouvert » (open) et cliquons sur le bouton « **Arduino Reset** » pour redémarrer le sketch.

Maintenant, nous pouvons changer la durée de clignotement en insérant une valeur en millisecondes (>100) dans le champ « Send data » et en appuyant sur « Enter » (au clavier).

N'oubliez pas de fermer la communication série lorsque vous quittez la page.

Ensuite, nous pouvons explorer le lien « Set clock » afin d'entrer la valeur correcte de l'heure. Nous pouvons la régler automatiquement à partir du système. En fait, si le RaspberryPi est connecté à Internet, il devrait avoir la date et l'heure à jour.

Pour définir un horaire de redémarrage automatique, nous utilisons la page « Set alarm for restart » puis nous éteignons « RandA » via la page « Switch off RPi » qui lance un « shutdown » (arrêt) et arrête l'alimentation.

« RandA » devient un Arduino amélioré

Supposons que vous ayez acheté la carte SD déjà configurée, l'installation est alors réduite à la modification de l'IDE d'Arduino déjà utilisé sur votre PC. Cette opération a été décrite au début de l'article.

L'archive d'installation, en tant que sauvegarde, se trouve dans le dossier : « **/home/pi/RandA** ». Il suffit de copier l'archive sur votre PC, par exemple en utilisant la fenêtre FTP de la session « MobaXterm », et de procéder comme indiqué ci-dessus, mais uniquement pour la partie relative au logiciel sur le PC.

Une fois les fichiers remplacés, démarrez l'IDE Arduino sur votre PC.

Il se peut que cela mette du temps car le réseau doit être analysé pour trouver le port distant de « RandA ». Vous pouvez vérifier que, parmi les ports disponibles, il y en a un distant qui contient une adresse réseau du style « //192.168.1.8/Arduino ». Sélectionnez-la. Pour effectuer un test, vous pouvez utiliser le sketch « Blink » ou le vôtre.

Gardez à l'esprit que tout sketch que vous chargez dans « RandA » est conservé en copie dans le dossier dédié au développement de l'IDE distant.

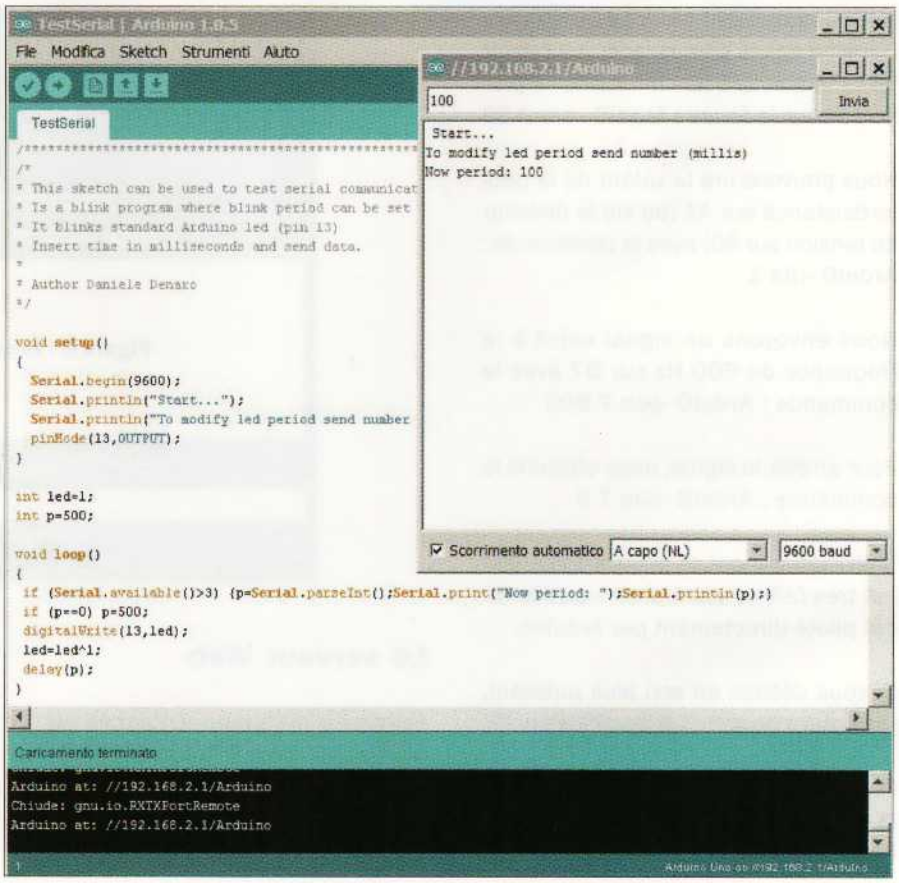


Figure 9 : le sketch « TestSerial » dans l'IDE Arduino.

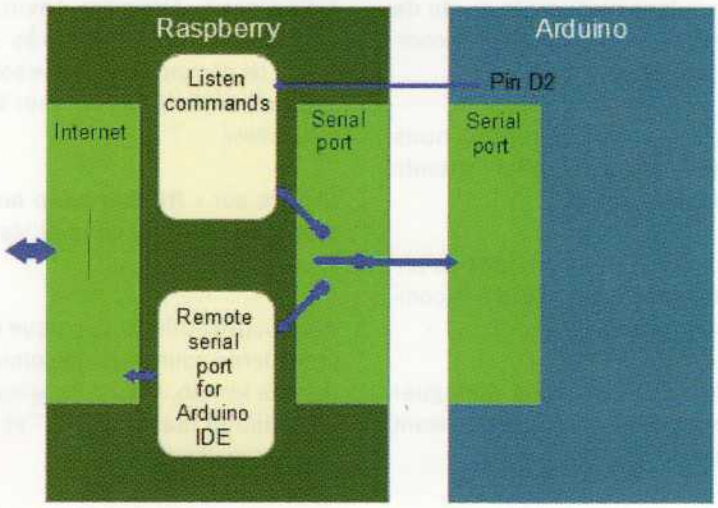


Figure 10 : le port série au service de différents programmes.

Cependant, les sketchs sont uniquement sous la forme d'un fichier exécutable (.hex), mais ils ont le même nom que le fichier source.

Essayons d'utiliser le sketch « TestSerial » que nous avons déjà utilisé avec le serveur Web. Le fichier source se trouve dans le dossier « **examples-notRAComm** » de la librairie « **RAComm** ».

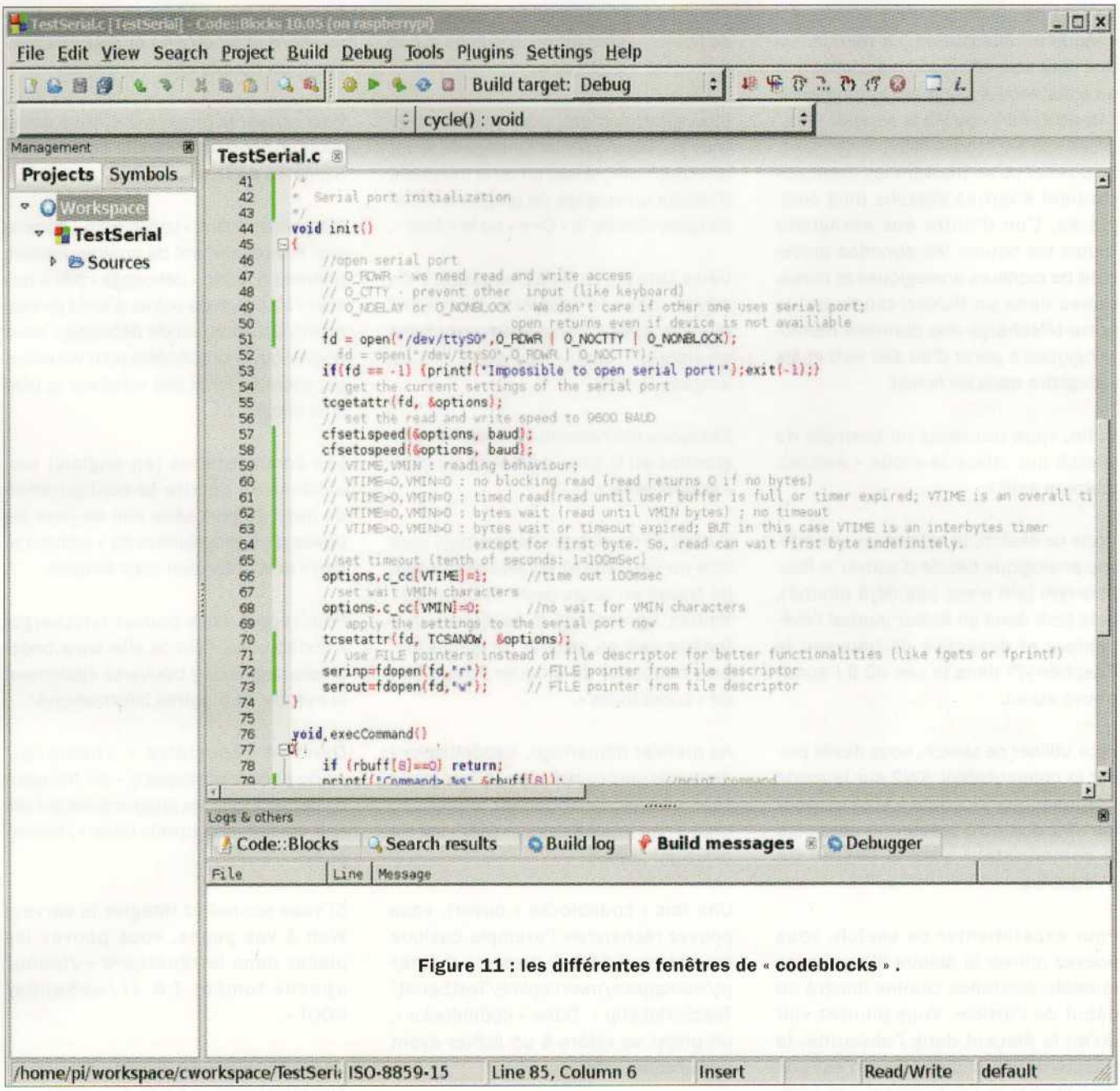


Figure 11 : les différentes fenêtres de « codeblocks » .

Dans ce dossier, nous avons ajouté des exemples à utiliser avec « RandA » qui n'exploitent pas correctement la librairie. Téléchargeons le sketch via le port distant.

Maintenant, pour l'utiliser et interagir via le port série, il suffit d'ouvrir la console de l'IDE, comme si Arduino était connecté via un port USB. Dans tous les cas, il est conseillé d'effectuer une réinitialisation, car il est important de se rappeler qu'il n'y a plus de reset automatique lors de l'ouverture de la console.

La réinitialisation peut être effectuée manuellement ou à distance à l'aide de la commande Linux « ResetRandA » ou via le serveur Web à partir de la page « Arduino console » (voir la figure 9). Dans ce cas, il n'est pas nécessaire d'ouvrir la connexion série.

Essayons d'utiliser un exemple à l'aide de la librairie qui communique avec le RaspberryPi. Le sketch le plus simple parmi les exemples de la librairie est « BasicRACommExample ».

Avant de le télécharger, il est préférable d'**ouvrir une session** « MobaXterm » (si elle n'est pas déjà ouverte), car **la console de l'IDE est inutilisable du fait que le port série est géré par le programme Linux** qui communique avec Arduino pour exécuter les commandes.

Par conséquent, pour afficher le contenu, il est nécessaire de le visualiser sur une fenêtre « Xterminal » ouverte pour l'occasion (voir la figure 10).

Lorsque le sketch est chargé, cela doit ouvrir une fenêtre « Xterminal » et afficher l'horodatage demandé au RaspberryPi.

Après quelques secondes, la fenêtre se ferme et le sketch fait clignoter la LED autant de fois que l'heure actuelle.

La lecture de l'horloge s'effectue à chaque réinitialisation. La réinitialisation peut être effectuée manuellement ou à distance à l'aide de la commande « ResetRandA » ou via le serveur Web.

Le dossier où se trouvent les exemples contient d'autres sketchs plus complexes. L'un d'entre eux enregistre toutes les heures les données provenant de capteurs analogiques et numériques dans un fichier, tandis qu'un autre télécharge des données météorologiques à partir d'un site web et les enregistre dans un fichier.

Enfin, vous trouverez un exemple de sketch qui utilise le mode « Arduino toujours actif ».

Dans ce sketch, la valeur sur une broche analogique décide d'activer le RaspberryPi (s'il n'est pas déjà allumé), puis écrit dans un fichier journal l'événement et désactive de nouveau le RaspberryPi dans le cas où il l'aurait trouvé éteint.

Pour utiliser ce sketch, vous devez placer le commutateur SW2 sur le mode « Arduino toujours actif » et le cavalier sur JP2 doit être présent (il connecte le commutateur avec la broche D4 d'Arduino).

Pour expérimenter ce sketch, vous pouvez utiliser la plaque d'essai avec la photorésistance comme illustré au début de l'article. Vous pourrez voir qu'en la plaçant dans l'obscurité, le RaspberryPi s'allume (si ce n'est pas déjà le cas) et l'événement est enregistré.

Programmons « RandA »

Si dans le monde d'Arduino, l'écriture de programmes s'effectue via l'environnement IDE bien connu, dans le monde du RaspberryPi le choix est plus large. Il est possible d'utiliser des commandes déjà prêtes et référencées dans le répertoire « /home/pi/bin ».

Rappelez-vous cependant que pour vous déplacer dans le système de fichiers du RaspberryPi, vous pouvez utiliser la fenêtre « Sftp » de la session « MobaXterm », ou mieux encore, lancer l'interface graphique du gestionnaire de fichiers « pcmanfm » que nous avons renommé « explorer ».

Plus généralement, pour programmer, vous pouvez utiliser un langage de script si vous en connaissez un ou le mieux est d'utiliser un langage de programmation complet comme le « C++ » ou le « Java ».

Vous trouverez deux exemples de scripts dans le dossier « /home/pi/bin/examples », l'un d'entre eux utilise un fichier « bash » et l'autre est en langage Python.

Essayons maintenant de créer un programme en C, pour cela nous lançons « **codeblocks** ».

Gardez à l'esprit que le démarrage peut être un peu long et surtout sans signe de travail en cours (work in progress). Veillez également à ce qu'aucune fenêtre soit en attente de validation, cela risquerait de bloquer l'exécution de « codeblocks ».

Au premier démarrage, « codeblocks » demande quel compilateur utiliser parmi ceux trouvés. Sélectionnez le premier, c'est-à-dire le compilateur « GCC » qui est défini par défaut.

Une fois « codeblocks » ouvert, vous pouvez rechercher l'exemple basique qui se trouve dans le dossier « /home/pi/workspace/cworkspace/TestSerial/TestSerial.cbp ». Dans « codeblocks », un projet se réfère à un fichier ayant l'extension « .cbp ».

La fenêtre principale de « codeblocks » se compose de plusieurs fenêtres, dont deux importantes contenant la liste des projets ouverts avec leur structure et l'éditeur de code source. La sortie de la console est affichée en ouvrant automatiquement une fenêtre « Xterminal ».

Le programme « TestSerial.c » (à ne pas confondre avec le sketch Arduino « TestSerial » vu plus haut) est le fichier source du projet. Il représente un exemple d'ouverture d'un port série pour communiquer avec Arduino.

Ce programme ouvre le port série, lit les enregistrements envoyés par Arduino et les affiche sur la console Linux.

Pour l'afficher, il suffit de le sélectionner dans la fenêtre de projets en effectuant un double clic.

Pour utiliser le programme, vous devez avoir un sketch qui envoie des enregistrements au RaspberryPi.

Bien évidemment, « codeblocks », comme tout environnement de programmation, permet un mode « débogage » pas à pas avec l'insertion de points d'arrêt (breakpoint). Grâce au mode débogage, vous pouvez ouvrir une fenêtre pour visualiser par exemple l'état des variables et bien plus encore.

Les commentaires (en anglais) permettent de décrire la configuration de base du port série afin de jeter les bases d'un programme de « collaboration » et d'utilisation avec Arduino.

Pour rappel, vous pouvez télécharger « codeblocks » sur le site www.codeblocks.org, vous y trouverez également le manuel et d'autres informations.

Dans le répertoire « /home/pi/workspace/cworkspace » se trouvent également tous les programmes qui ont leur version exécutable dans « /home/pi/bin ».

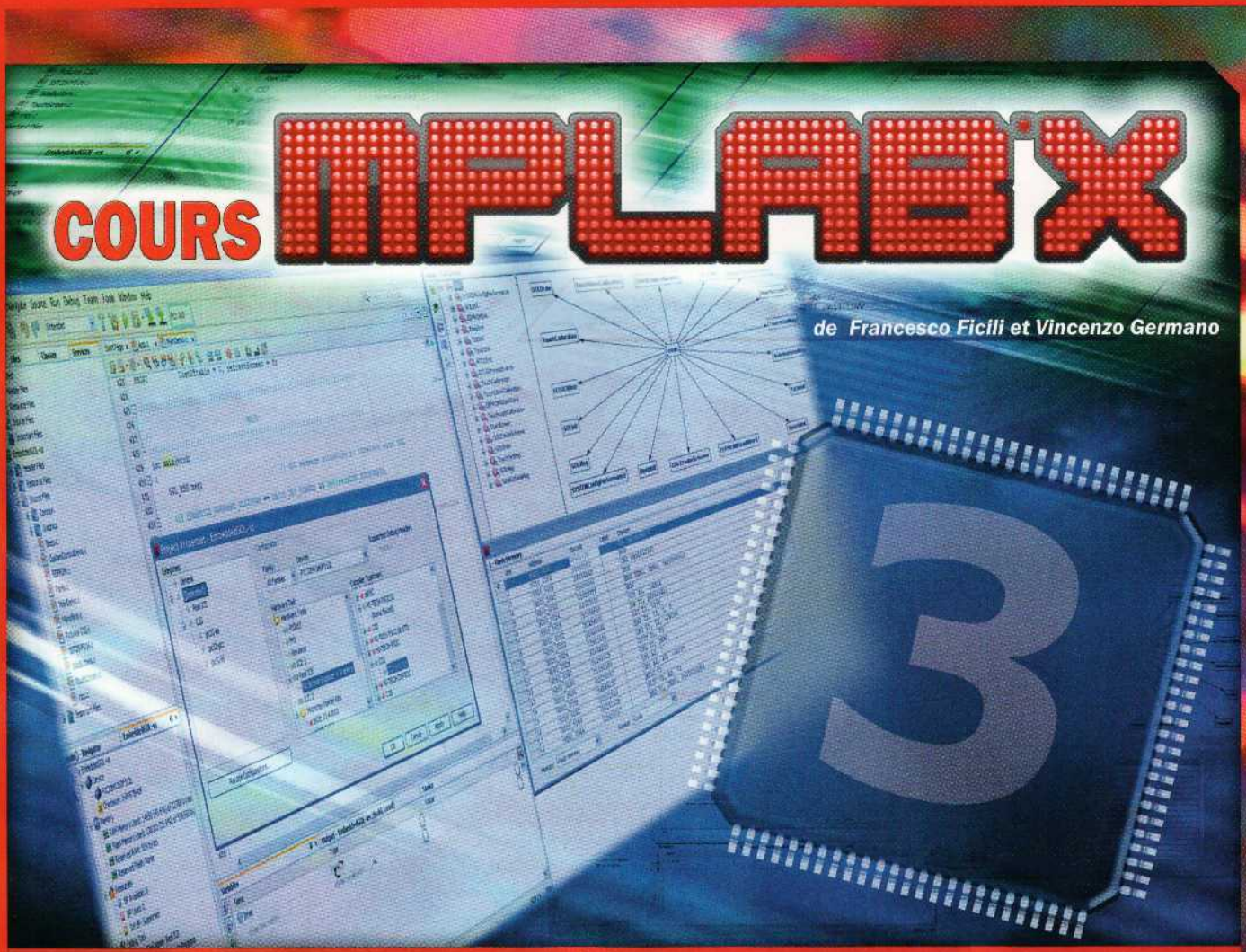
Si vous souhaitez intégrer le serveur Web à vos pages, vous pouvez les placer dans le répertoire « /home/apache-tomcat-7.0.47/webapps/ROOT ».

Une application web (fichier « .war ») peut être téléchargée à l'aide du « manager » de Tomcat (nom d'utilisateur et mot de passe : tomcat).

Enfin, si vous voulez utiliser le script CGI, vous devez placer vos pages dans le répertoire « /home/apache-tomcat-7.0.47/webapps/ROOT/WEB-INF/cgi ».

Cependant, n'oubliez pas de les appeler de la manière suivante : « http://...../cgi-bin/.... ». Vous trouverez quelques exemples de scripts dans le dossier.

Dans le prochain numéro, nous aborderons la mise en œuvre pratique d'un système anti-intrusion sophistiqué à l'aide de la carte « RandA ». ■



de Francesco Ficili et Vincenzo Germano

Cours MPLAB X

Quatrième partie

Nous continuons notre voyage à la découverte de MPLAB-X, le nouvel environnement de développement intégré produit et distribué par Microchip Technology. Il remplace l'ancien IDE MPLAB. Dans cet article, nous allons étudier la façon de mettre en œuvre des applications multitâches embarquées avec un PIC32 en utilisant le périphérique USB comme protocole de communication.

Dans les leçons précédentes, nous avons décrit les caractéristiques et le fonctionnement des microcontrôleurs PIC32 de Microchip et proposé des exemples pratiques pour utiliser divers périphériques, tels que le bus SPI et l'UART. Par la suite, nous avons analysé et implémenté un « scheduler » afin de gérer la programmation en mode multitâche coopératif.

À ce stade, nous avons tous les outils et tous les éléments pour aborder dans ce cours le fonctionnement et l'utilisation de l'un des périphériques les plus courants et les plus utilisés. Il s'agit du périphérique USB.

La pile (stack) USB de Microchip

Le standard de communication série USB (Universal Serial Bus) a été conçu pour permettre de connecter des périphériques hétérogènes à un ordinateur personnel (PC) en utilisant une seule interface standardisée et un seul type de connecteur.

Version: v2013-02-15			
v2013-02-15	Windows	Microchip Libraries for Applications	
v2013-02-15	Mac OS X	Microchip Libraries for Applications	
v2013-02-15	Linux	Microchip Libraries for Applications	
v2013-02-15		Help Files	
v2013-02-15		Release Notes ⓘ	

Figure 1 : téléchargement des fichiers MLA.

Ce standard a été développé pour permettre aussi l'implémentation de la fonctionnalité « **plug and play** » afin de connecter/déconnecter les périphériques « à chaud », sans devoir à chaque fois redémarrer l'ordinateur. Au fil du temps, l'USB s'est imposé comme l'un des standards de communication les plus courants et les plus répandus.

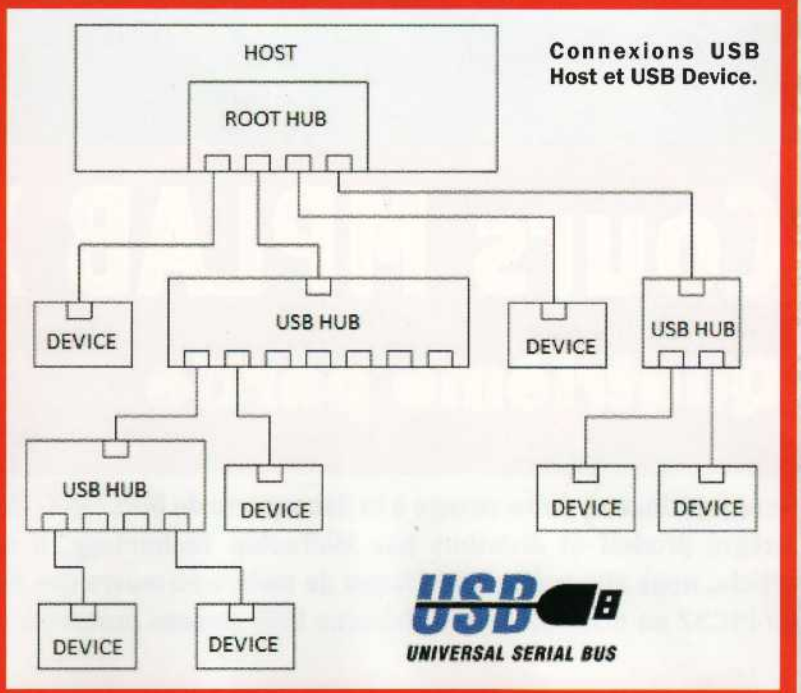
Aujourd'hui, il est présent dans la plupart des périphériques électroniques tels que les souris, les claviers, les mémoires de masse et les disques durs, les scanners, les appareils photo numériques, les imprimantes, les GPS, les téléviseurs, les smartphones, etc. L'une des principales caractéristiques de l'**USB** est son **architecture asymétrique** qui permet de connecter **un seul hôte** (**HOST**) et **plusieurs périphériques** (**DEVICE**) via une structure arborescente. Cela est rendu possible grâce à des dispositifs appelés « **hubs** » (concentrateurs).

Dans cet article, nous aborderons uniquement la partie concernant la configuration du périphérique USB DEVICE.

USB HOST & USB DEVICE

La première version de l'USB a été introduite en janvier 1996 et supportait des débits de seulement 1,5 Mbit/s (vitesses adaptées aux souris, claviers et autres appareils « lents ») avec une longueur de câble maximale de 3 mètres. Au fil du temps, son évolution a connu diverses mises à jour, jusqu'à l'USB 3.1 actuel, qui est capable de transférer des données avec un taux de transfert de 10 Gbit/s (soit environ 1,2 Go/s).

Le bus USB a été conçu pour permettre la connexion de plusieurs périphériques à l'aide d'une seule interface standardisée et d'un seul type de connecteur, mais également pour améliorer la fonctionnalité « plug and play » permettant de connecter ou déconnecter les périphériques à chaud sans devoir redémarrer l'ordinateur.



Connexions USB Host et USB Device.

L'une des principales caractéristiques de l'USB est son architecture asymétrique composée d'un seul gestionnaire (identifié comme « Host ») et de plusieurs périphériques (identifiés comme « Devices ») connectés à un arbre à l'aide de dispositifs appelés hubs (concentrateurs), comme le montre la figure ci-contre. Par conséquent, deux entités peuvent être identifiées dans le protocole de communication : « Host » et « Device ». La première communique avec le dispositif et reçoit les données de celui-ci par rapport aux actions effectuées par l'utilisateur, tandis que la seconde est l'entité qui interagit directement avec l'utilisateur et peut être, par exemple, un clavier ou une souris.

Dans les ordinateurs, le pilote analyse les données et permet une association dynamique des données d'entrées/sorties avec l'application. Ces caractéristiques ont permis une innovation et un développement rapide de diverses et nouveaux périphériques.

Rappelez-vous que l'**USB comporte deux types de périphériques** : « **HOST** » et « **DEVICE** ».

Leur fonctionnement est expliqué de manière succincte dans l'encadré intitulé « USB HOST et DEVICE », en analysant la **pile** (**stack**) mise à disposition par Microchip et en l'intégrant dans notre application. Dans la prochaine leçon, nous aborderons le dialogue et la configuration avec l'USB « HOST ».

Concentrons-nous maintenant sur la pile (stack) de Microchip. La première étape consiste à télécharger sur le site officiel le package d'installation des librairies « **Microchip Libraries for Applications** » ou (MLA).

Ce sont des librairies de référence à partir desquelles nous pourrons exploiter tout ce dont nous avons besoin dans notre projet. Elles améliorent l'interopérabilité entre les applications qui utilisent généralement plusieurs librairies et contiennent les outils de base pour le développement de la plupart des applications (du code source aux pilotes, documentation et utilitaires).

Pour cet article et les futurs, nous avons préféré **ne pas utiliser la version actuelle** (v2017-03-06) du MLA car elle ne comporte pas le framework dédié au PIC32 (il se trouve maintenant dans la suite logicielle : « MPLAB Harmony software suite »).

Nous avons opté pour une version précédente visible en figure 1, il s'agit de la version « **v2013-02-15** » qui nous a causé le moins de problèmes.

Vous pouvez la télécharger sur cette page : https://www.microchip.com/mplab/microchip-libraries-for-applications.

Cliquez sur l'onglet « **Downloads Archive** » qui se trouve vers le bas de la page et ensuite cherchez la version (toujours en allant vers le bas de la page).

Nous vous conseillons également de télécharger les fichiers d'aide (Help Files), ils sont très utiles pour mieux comprendre la structure de l'ensemble MLA mais malheureusement ils sont en anglais.

Dans tous les cas, une fois l'exécutable téléchargé et installé, le dossier de destination se présente comme en figure 2 (à gauche), montrant l'organisation principale des librairies de la MLA.

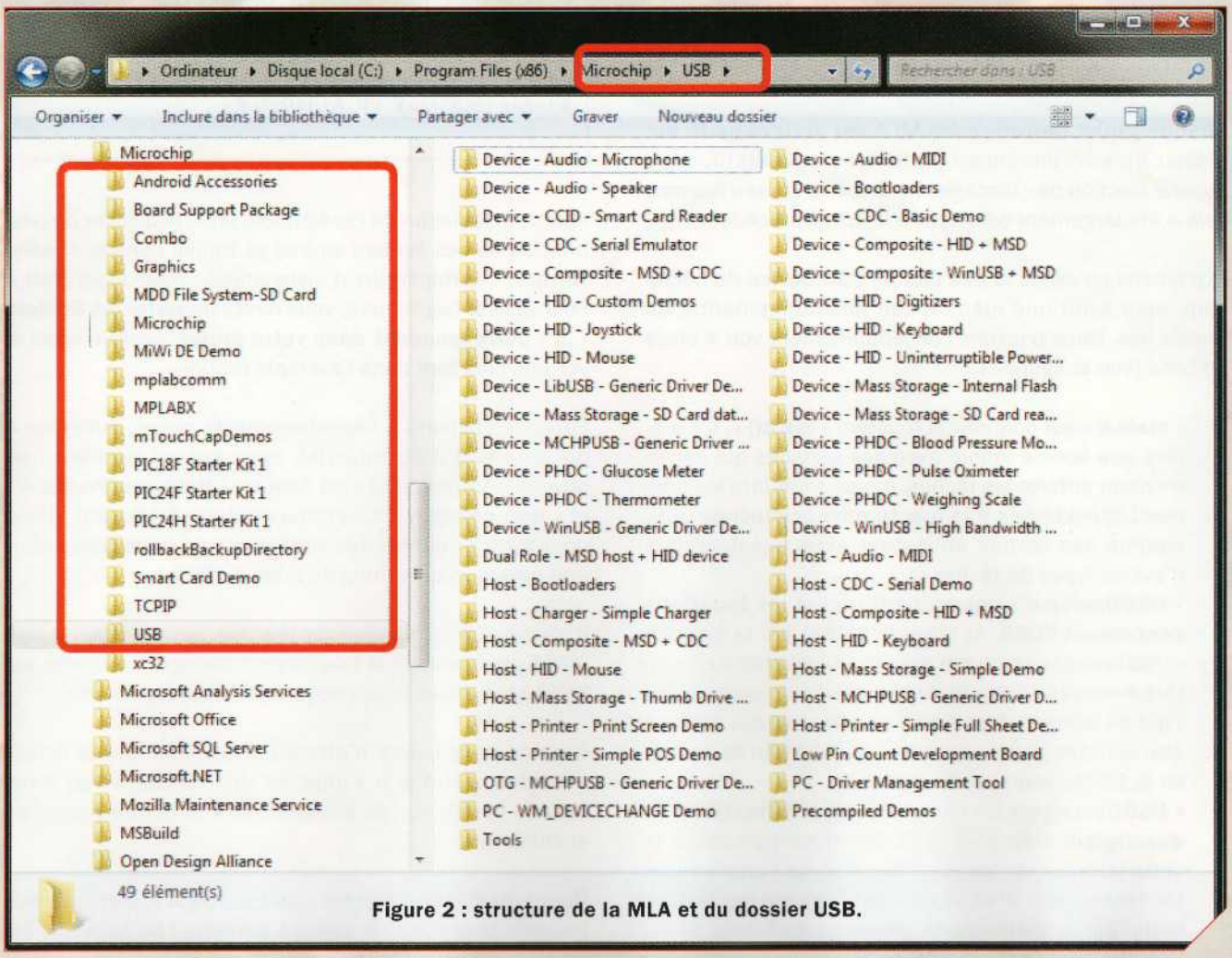


Figure 2 : structure de la MLA et du dossier USB.

Vous pouvez voir qu'il existe des dossiers liés aux graphiques, au protocole TCPIP, au protocole USB, etc. Dans chaque dossier, vous pouvez trouver des librairies, des codes source, des pilotes et des utilitaires, prêts à être intégrés dans vos projets.

Notez que, bien que l'utilisation de ces exemples de codes mis à disposition par Microchip puisse sembler immédiate, il n'en est rien. Vous devez toujours avoir une bonne connaissance de ce que vous voulez utiliser.

Le dossier qui nous intéresse pour cette partie du cours est bien évidemment l'USB, visible en figure 2 (à droite). Il contient les fichiers de configuration de ce périphérique et permet une création assez rapide d'applications dans lesquelles le périphérique USB est nécessaire. À l'intérieur, nous trouvons d'autres sous-dossiers de projets de référence qui couvrent une grande variété d'applications et de cas d'utilisation.

Configurer un projet de périphérique USB

Pour le projet pratique que nous vous proposons, nous faisons référence au contenu du dossier « **USB/Device - HID – Joystick** ». Dans ce dernier se trouve un ensemble d'interfaces qui sont des programmes modulaires gérant la plupart des communications USB. La figure 3 montre un organigramme typique d'un programme USB.

Chaque projet de référence MLA est écrit dans le but d'avoir un environnement multitâche coopératif, donc aucune fonction de « blocage » ne doit être utilisée (comme cela a été largement décrit dans le cours précédent).

Examinons en détail la pile (stack) USB Device de Microchip, pour avoir une idée de son fonctionnement à un niveau bas. Nous pouvons conceptuellement voir 4 blocs de base (voir la figure 3) :

- « ***main.c*** » qui contient la fonction « **main()** », c'est-à-dire une boucle infinie avec des services qui implémentent différentes tâches, qui peuvent être logiquement considérées à la fois comme des tâches USB, comme des tâches utilisateur mais aussi comme d'autres types de tâches ;
- « ***USBDevice.c*** » qui contient toutes les **fonctions concernant l'USB**, et qui sont gérées par la fonction « USBTasks() » en phase avec la philosophie du multitâche coopératif expliquée dans le cours précédent ;
- « ***hid.c*** » contenant les fonctions (**macros**) qui peuvent être utilisées par un utilisateur humain afin de contrôler le fonctionnement du système ;
- « ***USBDescriptor.c*** » contenant les **informations du descripteur USB** pour le périphérique particulier et cette information varie en fonction de l'application. Le descripteur d'un fichier est un enregistrement indiquant la méthode de stockage du fichier et/ou la structure de son contenu. Le descripteur peut aussi être son sélecteur, aussi appelé « handle ». Au sens le plus général, il s'agit d'un mot caractérisant l'information contenue dans un élément afin d'en faciliter l'utilisation. Il est à noter que l'HOST n'a pas besoin de comprendre intrinsèquement ou d'analyser le descripteur, généralement il existe un « descripteur de chaîne » qui fournit une description textuelle du périphérique.

Nous pouvons dire que l'« **USBDescriptor.c** » est un **tableau d'octets de données** qui décrivent les **paquets du périphérique**, y compris le nombre de paquets supportés par le périphérique, la taille des paquets et la fonction de chaque octet/bit dans le paquet.

Listing 1

```
// Configuration matérielle
#define USB_PING_PONG_MODE  USB_PING_PONG__
FULL_PING_PONG

// Configuration du périphérique
#define MY_VID              0x04D8
#define MY_PID              0x0000
#define USB_POLLING
#define USB_PULLUP_OPTION     USB_PULLUP_ENABLE
#define USB_TRANSCEIVER_OPTION USB_INTERNAL_
TRANSCEIVER
#define USB_EP0_BUFF_SIZE    8
#define USB_MAX_NUM_INT      (0+1)
#define USB_MAX_EP_NUMBER 6
```

Après l'installation de l'exécutable MLA, le dossier de destination de ces fichiers source se trouve dans le chemin suivant : « \<Répertoire d'installation\>\Microchip\USB ». Pour utiliser ces fichiers, vous devez **importer les fichiers « .h » correspondant dans votre projet**, comme nous le verrons plus tard dans l'exemple pratique.

Parmi les fichiers à importer pour le projet, outre les 4 qui viennent d'être décrits, deux autres méritent une attention particulière : les fichiers « **HardwareProfile.h** » et « **usb_config.h** ». Le premier est généralement utilisé pour mapper différentes configurations matérielles pour une définition commune du code à utiliser.

Il contient divers paramètres tels que, par exemple, le port utilisé pour les LED, la fréquence d'horloge de la carte, les broches utilisées pour l'acquisition d'un signal, etc.

D'après ce qui vient d'être dit, il est clair que le fichier « HardwareProfile.h » joue un rôle fondamental dans chaque projet, afin de maintenir une structure ordonnée et cohérente.

Quant au fichier d'en-tête « usb_config.h », comme nous pouvons le voir dans le listing 1 (un extrait du fichier), c'est un élément clé pour configurer la pile USB.

En fait, il définit différents paramètres à l'intérieur de la pile qui déterminent le fonctionnement de celle-ci ainsi que les caractéristiques optionnelles à y inclure.

Évidemment, la modification de ces options peut avoir des répercussions sur la taille du code, l'utilisation de la RAM et le débit des données.

Comme vous pouvez l'imaginer, sans une définition appropriée de ce fichier, la pile USB ne fonctionnera pas comme vous l'auriez désiré pour votre application.

La classe USB HID

Après avoir mentionné les parties fondamentales de la pile Microchip et examiné comment elles fonctionnent ensemble, abordons maintenant la classe « **Human Interface Device** » ou « **HID** » liée à l'« Universal Serial Bus » (USB).

Le standard « HID » a été adopté entre autre pour permettre l'innovation dans les périphériques d'entrée des ordinateurs, mais aussi pour simplifier le processus d'installation de ces derniers.

En fait, de nos jours, la plupart des périphériques « HID » sont dotés de paquets d'auto-installation qui peuvent contenir divers types et formats de données.

Par exemple, il n'y a plus besoin de CDROM d'installation lorsque vous connectez un disque dur nomade sur un port USB, en branchant la prise USB vous voyez apparaître dans la barre des tâches d'un PC en bas à droite les messages « installation du pilote XXX » puis « Votre périphérique est prêt à l'emploi », sans avoir installé le moindre driver ou pilote.

La classe « HID » est composée de périphérique qu'un humain peut utiliser pour contrôler le fonctionnement des systèmes, c'est-à-dire une souris, un clavier, un joystick, etc.

Il est important de savoir que, en plus de fournir des informations à partir d'interfaces humaines, elle fournit également des données de sortie (output) pour indiquer au système d'exploitation de l'ordinateur les actions nécessaires à effectuer.

Pour avoir une idée générale de la façon dont une communication d'un périphérique USB fonctionne, nous pouvons affirmer qu'elle repose sur des tuyaux, c'est-à-dire des connexions logiques du contrôleur hôte (HOST) à une entité logique située sur un périphérique et qui est identifiée avec le terme « **endpoint** ».

Puisque ces canaux correspondent 1 à 1 aux « endpoint », les terminaisons sont parfois utilisées de manière interchangeable. Un périphérique USB peut avoir jusqu'à **32** « **endpoint** » (16 IN et 16 OUT), mais il est rare d'en avoir autant.

Un « endpoint » est défini et numéroté par le périphérique pendant la phase d'initialisation (la période immédiatement après la connexion physique appelée « énumération ») et est donc permanent, tandis qu'un canal peut être ouvert et fermé après une communication.

En utilisant le firmware de Microchip que nous venons de présenter, la classe « HID » implémente les fonctions spécifiques suivantes :

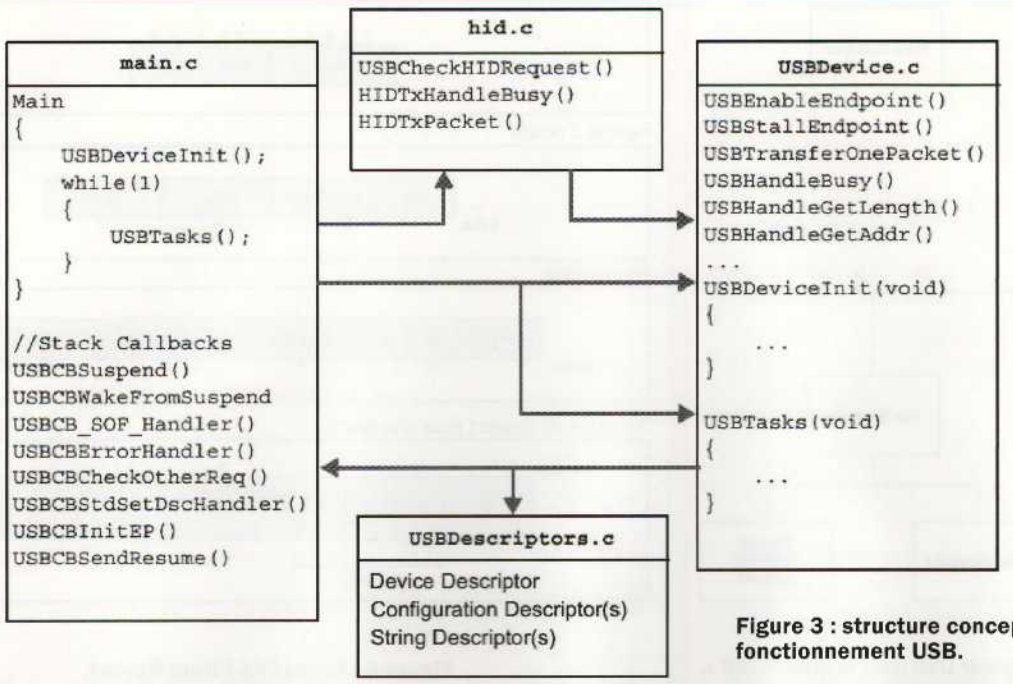


Figure 3 : structure conceptuelle du fonctionnement USB.

- **USB Descriptor Table** ou table des descripteurs USB ;
- **Endpoint Configuration Table** ou table de configuration des « endpoint » (points de terminaison) ;
- **Function Driver Table** ou table des fonctions des pilotes.

En ce qui concerne l'« USB Descriptor Table », chaque périphérique USB doit fournir une série de descripteurs (structures de données) qui définissent le périphérique et informent l'hôte USB du type de classe du pilote (driver) à utiliser.

Les descripteurs des périphériques USB peuvent être organisés en 3 groupes : Device, Configuration et Strings.

- « ***Device*** » identifie le **type de périphérique** et fournit le nombre de configurations possibles ;
- « ***Configuration*** » décrit les **types d'interfaces** et les « **endpoint** » utilisés (y compris les descripteurs spécifiques des classes) ;
- « ***Strings*** », généralement facultatives, fournissent des **informations lisibles** par l'utilisateur que l'HOST peut visualiser.

La « **Endpoint Configuration Table** » est utilisée par la pile (stack) USB afin de **configurer correctement** tous les « **endpoint** » d'interface et les paramètres alternatifs, tels que définis par la table des descripteurs. Elle identifie également les fonctions à utiliser pour gérer l'événement qui se produit sur chaque « endpoint ».

Enfin, en ce qui concerne la « **Function Driver Table** », étant donné qu'un dispositif peut implémenter plus d'une classe/driver d'un périphérique USB d'un fournisseur spécifique, la pile Microchip utilise une table pour **gérer l'accès** et **supporter la fonction du driver**.

Chaque entrée de la table contient les informations nécessaires pour gérer une seule fonction du driver.

Chaque périphérique USB est associé à une structure de descripteur et peut contenir plusieurs classes définies dans la couche d'interface, comme par exemple pour le « HID ».

Une structure de descripteur arborescente est représentée en figure 4, elle décrit un périphérique de classe « HID ». Comme vous pouvez le voir, le descripteur « HID » indique combien d'autres descripteurs spécifiques suivent.

Notez qu'**au moins un « Report Descriptor » doit être présent**, alors que les « physiques » sont facultatifs.

Un « Report Descriptor » **décrit le format et la signification des données générées par le périphérique**, il est chargé par la classe du driver « HID » de l'HOST à l'aide d'une requête spécifique.

Après l'initialisation, le périphérique génère des rapports pour indiquer si un utilisateur interagit avec un périphérique et à quel moment. Il se compose d'**éléments d'informations** appelés « **Items** » (éléments) et chacun décrit un aspect des données du rapport.

Généralement un rapport d'un « **Item** » suit un format **composé d'un préfixe d'un octet** et d'une « **payload** » (information utile).

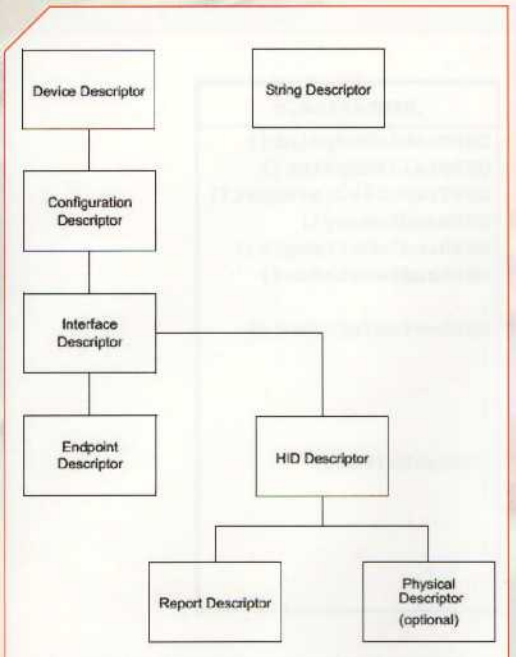


Figure 4 : descripteur USB pour le driver « HID ».

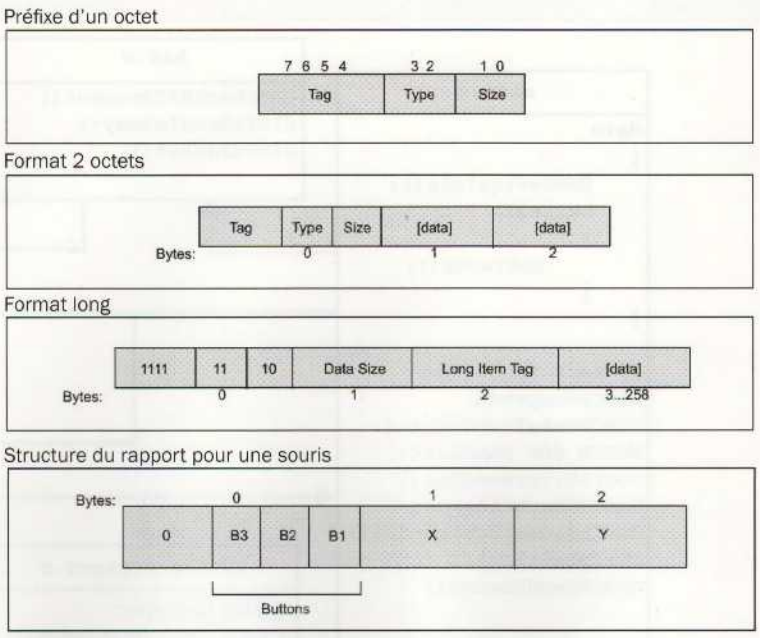


Figure 5 : format de l'Item Report.

Le premier contient l'étiquette (tag), le type et la longueur de la « payload », comme le montre la figure 5 (préfixe d'un octet). Cette figure montre également les deux formats qu'un rapport peut prendre : court et long.

Dans l'exemple pratique qui sera abordé plus loin, le premier format sera utilisé. Grâce à celui-ci, il sera possible d'envoyer à l'ordinateur les mouvements de la souris en interagissant avec un joystick (dans le format indiqué par « Structure du rapport pour une souris »).

Il est facile d'imaginer que les rapports d'entrée sont utilisés pour envoyer les données des interactions de l'utilisateur avec le périphérique vers l'HOST, tandis que les rapports de sortie sont utilisés par l'HOST pour envoyer des données au contrôleur, par exemple l'allumage d'une LED.

Enfin, les rapports sont utilisés par l'HOST pour configurer correctement le périphérique.

La macro USB

Avec l'« **Application Programming Interface** » (API ou Interface de Programmation d'Applications), est généralement disponible pour le programmeur un ensemble de procédures regroupées pour former un ensemble d'outils spécifiques pour une tâche donnée dans un programme.

Le but étant d'obtenir une abstraction de plus haut niveau entre les logiciels de haut et de bas niveau, simplifiant ainsi le travail de programmation, et permettant aux programmeurs d'éviter de réécrire à chaque fois toutes les fonctions nécessaires au programme à partir du bas niveau.

Il existe **deux niveaux** d'**API liés à l'USB** « **HID** » : le niveau USB et le niveau du système d'exploitation.

Le premier est un protocole utilisé par les périphériques, il permet d'envoyer (dialoguer) les caractéristiques de leurs fonctionnalités/capacités au système d'exploitation afin que ce dernier puisse les gérer correctement.

Le second niveau du système d'exploitation offre une vue de haut niveau aux applications, qui n'ont plus besoin de prendre en charge les périphériques de manière individuelle.

Dans notre cas, la fonction la plus importante que nous utiliserons dans l'exemple pratique sera la macro « **HIDTxPacket** » contenue dans le fichier « **usb_function_hid.h** », qui a par définition la structure suivante :

```
#define HIDTxPacket USBTxOnePacket
```

Grâce à cette macro, il est possible d'envoyer des données spécifiques à partir d'un « endpoint » particulier et donc d'être en mesure de transférer des informations au système d'exploitation.

La macro accepte les paramètres suivants :

- « **BYTE ep** » : il s'agit d'un « endpoint » vers lequel envoyer les données ;
- « **BYTE* data** » : c'est un pointeur vers les données que vous voulez envoyer ;
- « **WORLD len** » : correspond à la longueur des données à transférer.

La macro fournit en retour une valeur qui permet de comprendre l'état du transfert (« USB_HANDLE »). Nous voyons ci-après une application.

Listing 2 : Configuration des ports du microcontrôleur connectés au Joystick.

```
/*********************************************
* Utilisateur Joystick
*********************************************/
/* Registre de direction des données */
#define JOY_UP_TRIS      TRISBbits.TRISB10
#define JOY_RIGHT_TRIS   TRISBbits.TRISB9
#define JOY_LEFT_TRIS    TRISBbits.TRISB12
#define JOY_DOWN_TRIS    TRISBbits.TRISB11
#define JOY_FIRE_TRIS    TRISBbits.TRISB13

/* Ressources des Alias */
#define JOY_UP   PORTBbits.RB10
#define JOY_RIGHT        PORTBbits.RB9
#define JOY_LEFT         PORTBbits.RB12
#define JOY_DOWN         PORTBbits.RB11
#define JOY_FIRE         PORTBbits.RB13
```

Listing 3 : insertion d'une tâche USB.

```
/* Tableau des tâches à appeler */
void (*TaskArray[ACTIVE_TASK_NUMBER]) (void) =
{
   LedTask,
   UsbTask,
};

/* Tableau des délais d'expiration des périodes des tâches */
UINT16 TaskPeriodTimeoutMs[ACTIVE_TASK_NUMBER] =
{
   LED_TASK_PERIOD_MS,
   USB_TASK_PERIOD_MS,
};
```

Listing 4 : implémentation de la tâche USB.

```
/*****************************************************
* Fonction:        UsbTask
* Input:           None
* Output:          None
* Auteur:          F.Ficili
* Description:     Gestions des tâches USB et des machines à états
* Date:            28/02/18
*****************************************************/
void UsbTask (void)
{
   switch (SystemState)
   {
      /* Phase d'initialisation du système */
      case InitializationState:

         /* Initialise le port en digital */
         AD1PCFG = 0xFFFF;
         /* Initialise les broches du JOYSTICK */
         JOY_UP_TRIS    =  LINE_DIRECTION_INPUT;
         JOY_RIGHT_TRIS =  LINE_DIRECTION_INPUT;
         JOY_LEFT_TRIS  =  LINE_DIRECTION_INPUT;
         JOY_DOWN_TRIS  =  LINE_DIRECTION_INPUT;
         JOY_FIRE_TRIS  =  LINE_DIRECTION_INPUT;

         /* Initialise le périphérique USB*/
         USBDeviceInit();
         break;

      /* Phase normale du système d'exploitation */
      case RunningState:
         /* Tâche de la pile USB */
         USBDeviceTasks();
         /* Émulation du joystick */
         JoystickTask();
         /* Gestion du joystick */
         JoystickHidSendTask();
         break;

      /* Défaut */
      default:
         break;
   }
}
```

La structure du projet reste celle du « scheduler » présenté dans l'article précédent, cependant l'architecture correspond à celle de la figure 7. Elle montre la fonctionnalité du « scheduler » en arrière-plan, le blocage des pilotes qui communiquent à un niveau bas avec le matériel et enfin les deux tâches qui seront exécutées en fonctionnement (dans ce cas la LED et l'USB).

L'organisation globale du projet Joystick USB est illustrée en figure 8, où vous pouvez voir les fichiers « header » dans la partie de gauche et les fichiers source dans la partie de droite. Il est conseillé d'avoir une structure de projet bien organisée afin de mieux gérer la maintenance.

Le dossier « **Bsp** » (fichier header ou d'en-tête) contient les fichiers de configuration, tels que les fichiers de configuration des bits et le fichier « hw_profile.h », précédemment expliqué, qui concerne la configuration de notre matériel. Par contre, dans le dossier équivalent du code source, nous trouvons la gestion des interruptions.

En ce qui concerne le dossier « **Stack** », dans la partie liée aux « header », nous trouvons les fichiers de la configuration de l'USB, y compris les fonctions « HID » (usb_function_hid.h). L'implémentation de l'USB est contenue dans le dossier « Stack » du dossier source, en fait nous pouvons y reconnaître les fichiers précédemment traités. Les dossiers « Sched » et « Task » ont été largement expliqués dans l'article précédent.

Après avoir étudié la structure du projet et son fonctionnement, passons aux détails d'implémentation à partir de la configuration des ports de la Demoboard PIC32.

Exemple pratique : Mini Joystick USB

Maintenant, utilisons les informations que nous avons vues jusqu'à présent en présentant un exemple pratique de joystick USB. Le code a été écrit et testé sur la Demoboard PIC32 spécialement conçue pour ce cours.

Les parties qui nous intéressent dans notre cas sont encadrées par un rectangle bleu en figure 6. Il s'agit du joystick et du connecteur USB.

Évidemment, grâce au connecteur USB, nous allons nous connecter à l'ordinateur. En utilisant le joystick nous pourrons déplacer la souris comme un « trackpad » avec l'ajout du bouton « Confirmation » (bouton gauche de la souris).

En analysant le data-sheet du PIC32MX et en regardant les interconnexions de celui-ci avec le joystick, nous pouvons insérer dans le fichier « header » de configuration (hw_profile.h) les définitions visibles dans le listing 2.

Nous configurons les ports à partir desquels nous pouvons lire les signaux provenant du joystick. Nous ajoutons une nouvelle tâche liée à l'USB dans le fichier « scheduler.c » (en plus de la LED déjà présente), comme indiqué précédemment dans l'architecture et visible dans le listing 3.

À ce stade, nous créons un fichier appelé « **usb_task.c** » qui initialise le périphérique, puis le gère.

Listing 5 : implémentation de la tâche USB

```
void JoystickTask (void)
{
   /* Variable d'état du joystick */
   static JoystickStateType JoystickState = DoNothing;

      if (JOY_UP == 1)
      {
         /* Va à MoveUP */
         JoystickState = MoveUp;
      }
      else if (JOY_RIGHT == 1)
      {
         /* Va à MoveRight */
         JoystickState = MoveRight;
      }
      else if (JOY_LEFT == 1)
      {
         /* Va à MoveLeft */
         JoystickState = MoveLeft;
      }
      else if (JOY_DOWN == 1)
      {
         /* Va à MoveDown */
         JoystickState = MoveDown;
      }
      else if (JOY_FIRE == 1)
      {
         /* Va à MoveDown */
         JoystickState = FireButton;
      }
      else
      {
         JoystickState = DoNothing;
      }

      switch (JoystickState)
      {
         case MoveUp:
            /* déplace le joystick vers le haut */
            HidBuffer[HID_REPORT_BUTTON] = 0;
            HidBuffer[HID_REPORT_X_AXIS] = 0;
            HidBuffer[HID_REPORT_Y_AXIS] = JOY_SPEED*MOVE_UP;
            break;

         case MoveDown:
            /* déplace le joystick vers le bas */
            HidBuffer[HID_REPORT_BUTTON] = 0;
            HidBuffer[HID_REPORT_X_AXIS] = 0;
            HidBuffer[HID_REPORT_Y_AXIS] = JOY_SPEED*MOVE_DOWN;
            break;

         case MoveLeft:
            /* déplace le joystick vers le bas */
            HidBuffer[HID_REPORT_BUTTON] = 0;
            HidBuffer[HID_REPORT_X_AXIS] = JOY_SPEED*MOVE_LEFT;
            HidBuffer[HID_REPORT_Y_AXIS] = 0;
            break;

         case MoveRight:
            /* déplace le joystick vers le bas */
            HidBuffer[HID_REPORT_BUTTON] = 0;
           HidBuffer[HID_REPORT_X_AXIS] = JOY_SPEED*MOVE_RIGHT;
```

En se référant au Listing 4, nous pouvons voir dans ce cas également un état différent de celui dans lequel le périphérique a été initialisé (case InitializationState) et une phase de fonctionnement normal (case RunningState).

Dans la phase d'initialisation, les ports ont été définis en tant que « digitaux » (étant donné que, par défaut, ils sont définis comme analogiques), puis initialisés comme des entrées pour permettre la lecture des états du joystick. Enfin, la fonction d'initialisation du périphérique USB est appelée.

Encore une fois en se référant au Listing 4, après la phase d'initialisation, le code qui est exécuté est celui du fonctionnement normal (case RunningState), vous pouvez voir que les fonctionnalités suivantes sont présentes :

- lecture de l'état du joystick, qui est obtenue à l'aide de la fonction « «JoystickTask() » ;
- envoi de la commande correspondante à l'ordinateur via l'USB à l'aide de la fonction « JoystickHidSendTask() ».

Le Listing 5 montre l'implémentation de la fonction « **JoystickTask()** » qui, comme prévu, lit l'état du joystick et le mémorise dans la **variable** « **JoystickState** » qui est ensuite utilisée pour connaitre la direction vers laquelle déplacer la souris via le bloc (partie du programme) « switch (JoystickState) ».

Comme vous pouvez le voir, en plus de la définition des 4 mouvements que peut effectuer la souris, deux autres cas ont été insérés. Le premier concerne la pression d'un bouton (bouton gauche de la souris) et le second est lié au statut d'attente d'une action de la part de l'utilisateur.

Ce dernier cas est fondamental car sinon la souris resterait dans le dernier état de mouvement mémorisé et **continuerait à bouger vers l'un des 4 bords de l'écran**.

Enfin, la dernière étape consiste à envoyer les données vers l'ordinateur ainsi que les mouvements que la souris doit effectuer.

```
            HidBuffer[HID_REPORT_Y_AXIS] = 0;
            break;


        case FireButton:
            /* déplace le joystick vers le bas */
            HidBuffer[HID_REPORT_BUTTON] = FIRE_ON;
            HidBuffer[HID_REPORT_X_AXIS] = 0;
            HidBuffer[HID_REPORT_Y_AXIS] = 0;
            break;

        case DoNothing:
            /* déplace le joystick vers le bas */
            HidBuffer[HID_REPORT_BUTTON] = 0;
            HidBuffer[HID_REPORT_X_AXIS] = 0;
            HidBuffer[HID_REPORT_Y_AXIS] = 0;
            break;

        default:
            break;
    }
}
```

Pour cela, nous avons défini la fonction « **JoystickHidSendTask()** » visible dans le listing 6.

Comme vous pouvez le voir, cette fonction ne fait rien d'autre que copier les données de la souris dans le buffer de l'HID, puis les transmettre à l'ordinateur en utilisant la macro analysée précédemment.

À ce stade, le programme est terminé. Tout ce que vous avez à faire est de le compiler, puis de programmer le PIC32 de la Demoboard.

Ensuite, vous devez la connecter à votre ordinateur afin d'utiliser votre propre « trackpad » fait maison.

La plupart des systèmes d'exploitation reconnaissent les périphériques standard HID USB tels que les claviers/souris, sans avoir besoin de drivers (pilotes) spécifiques.

En fait, une fois connecté, l'ordinateur les installe et un message apparaît à l'écran pour confirmer que le périphérique a été correctement reconnu et installé. Il peut ainsi être utilisé, comme indiqué dans la séquence des messages d'installation visibles en figure 9.

Il est à noter que ce qui vient d'être présenté dans ce projet pratique est compatible avec les versions de Windows XP, Vista et 7, **mais pas avec les dernières versions du système d'exploitation de Microsoft**, c'est-à-dire avec Windows 8 et Windows 10.

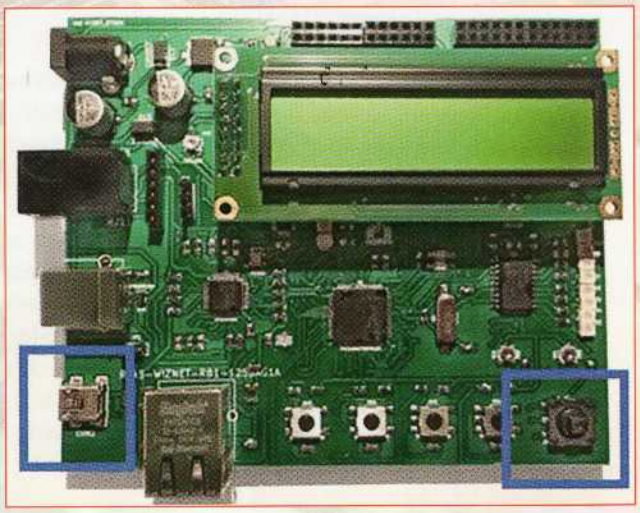


Figure 6 : la prise USB et le joystick sur la Demoboard PIC32.

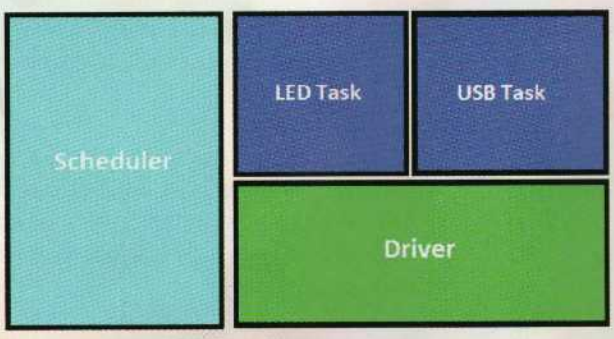


Figure 7 : architecture logicielle.

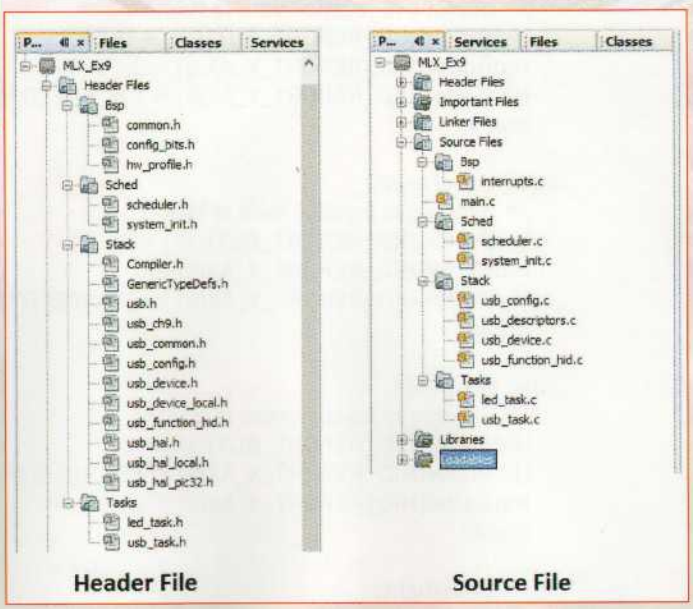


Figure 8 : structure du programme Joystick USB.

Listing 6 : fonction de transmission des données via l'USB.

```
/*********************************************************
* Fonction:    JoystickHidSendTask
* Input:       void
* Output:      void
* Auteur:      F.Ficili
* Description :  envoi de données HID à l'HOST
* Date:            07/03/18
*********************************************************/
void JoystickHidSendTask (void)
{
   /* Copie les données dans le buffer de l'HID */
   HidReportIn[HID_REPORT_BUTTON] = HidBuffer[HID_REPORT_BUTTON];
   HidReportIn[HID_REPORT_X_AXIS] = HidBuffer[HID_REPORT_X_AXIS];
   HidReportIn[HID_REPORT_Y_AXIS] = HidBuffer[HID_REPORT_Y_AXIS];

   /* Envoie le paquet de 3 octets via l'USB à l'HOST */
   LastTransmission = HIDTxPacket(HID_EP, (BYTE*)HidReportIn, 0x03);
}
```

Listing 7 :

```
// Descripteur de chaîne de produit

ROM struct{BYTE bLength;BYTE bDscType;WORD string[32];}sd002={
sizeof(sd002),USB_DESCRIPTOR_STRING,
{'J','o','y','s','t','i','c','k',' ','D','e','m','o',' ',
     'E','l','e','c','t','r','o','n','i','q','u','e',' ','L','o','i','s','i','r','s'
}};
```

Figure 9 : séquence des messages d'installation sous Windows XP.

En effet, dans ce dernier cas, le périphérique ne sera pas reconnu et il ne sera pas possible de l'utiliser car il y a une incompatibilité des drivers.

Comme vous pouvez le voir sur l'image de gauche de la figure 9, le descripteur peut être personnalisé de manière à pouvoir afficher le texte souhaité pendant l'installation.

Dans notre cas, nous l'avons configuré comme ceci : « Joystick Demo Electronique Loisirs ». La personnalisation du descripteur s'effectue en modifiant, dans le dossier « Stack » du projet, le fichier « **usb_descriptors.c** » comme indiqué dans le listing 7.

Faites attention au fait que vous devez également **modifier la longueur de la chaîne à envoyer**, en plus de son contenu, sinon l'opération n'est pas possible. ■

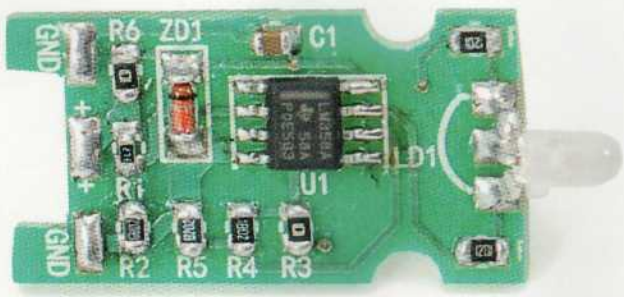
INDICATEUR D'ÉTAT DE LA BATTERIE

Nous vous proposons dans cet article un petit montage qui, une fois inséré dans la prise allume-cigare d'un véhicule, indique le niveau de tension de la batterie au moyen d'une LED bicolore.

de Davide Scullino

Une batterie excessivement déchargée ne permet pas le démarrage du moteur, a fortiori en hiver où le froid réduit considérablement son efficacité. En connaissant la tension de la batterie lorsque le moteur est éteint et donc son état, cela permet d'éviter des problèmes désagréables le matin pour se rendre sur son lieu de travail par exemple.

En effet, la tension en dit long sur les conditions de la batterie, c'est-à-dire sur sa charge résiduelle et sur sa capacité de charge ainsi que sur ses possibilités à fournir le courant d'appel requis par le démarrage du moteur, surtout dans le cas d'un diesel. L'augmentation du taux de compression des moteurs (de l'ordre de 15/1 pour un turbo diesel moderne à rampe commune contre 22/1 pour un vieux diesel atmosphérique) implique une puissance nécessaire plus importante pour le démarreur.



La tension du système électrique de la voiture reflète le fonctionnement du moteur, car elle nous permet de comprendre si l'alternateur charge correctement la batterie ou si un problème modifie la tension.

Pour ces raisons, un voltmètre placé sur le tableau de bord indiquait cette tension sur les anciennes voitures, aujourd'hui ce dernier a disparu. Certainement pour réaliser des économies sur les coûts de production et aussi parce que l'unité de contrôle du véhicule mesure en permanence la tension et met le moteur en mode dégradé si un problème survient.

La durée de vie d'une batterie est généralement estimée à environ 4 ou 5 ans.

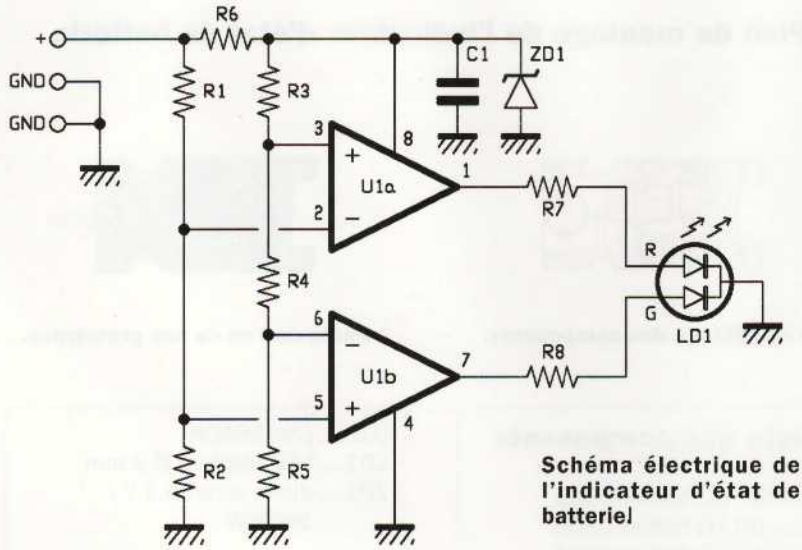
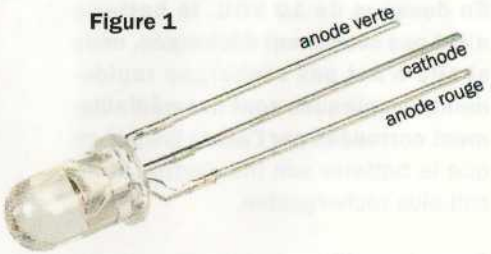


Schéma électrique de l'indicateur d'état de batterie

Figure 1



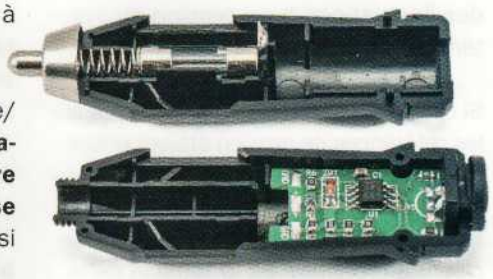
Par conséquent, la couleur rouge indique que la batterie est faible, la couleur jaune indique que la batterie se trouve dans une plage de tension acceptable (mais à surveiller) et la couleur verte indique que la batterie est bien chargée.

Pour comprendre la signification de ces tensions, considérons que la batterie est du type plomb-acide sulfurique, c'est-à-dire un **dispositif électrochimique réversible** dont la **structure élémentaire** (cellule) est **constituée de deux plaques de plomb immergées dans un liquide** appelé **électrolyte**. C'est de l'acide sulfurique dilué (H_2SO_4) dans l'eau.

Dans les **conditions de repos**, la **tension d'une cellule** de la batterie est d'environ **2 VDC**. Cette tension augmente pendant la charge et diminue pendant la décharge. Pour obtenir une batterie de 12 VDC, il est donc nécessaire de relier 6 cellules en série.

En considérant que la tension à vide aux bornes de la batterie reflète son état de charge (la batterie doit être au repos pendant au moins 6 heures et le voltmètre avec lequel la mesure est effectuée doit avoir une très haute impédance), la tension à la **charge complète doit être de 12,6 VDC**, elle chute à **12,2 VDC à 50 % de la charge**.

Si la tension tombe **en dessous de 11,5 VDC**, la batterie est **déchargée**.



Si vous avez un doute, sachez qu'une batterie en bonne santé a une tension comprise entre 12,5 VDC et 12,7 VDC (moteur arrêté). Si elle est inférieure à 11 VDC, il est probable que la batterie doive être remplacée.

Pour prolonger la durée de vie de votre batterie, contrôlez le niveau d'électrolyte (une solution d'eau et d'acide sulfurique qui conduit l'électricité) au moins une fois par an.

Pour ce faire, dévissez le bouchon de remplissage et vérifiez visuellement la position du liquide par rapport au repère. Si nécessaire, faites l'appoint avec de l'eau déminéralisée de manière à ce que les éléments en plomb de la batterie soient intégralement immergés.

À noter que cette opération est impossible sur les batteries « sans entretien » car elles ne s'ouvrent pas.

Des cosses encrassées nuisent également à la conductivité. **Avant de les débrancher** pour les nettoyer, **renseignez-vous sur les conséquences auprès du constructeur de la voiture**.

Sur certains véhicules récents, vous pouvez par exemple **perdre la programmation** de votre autoradio ou, plus ennuyeux, du **calculateur** !

Il existe dans le commerce des chargeurs qui se branchent sur la prise allume-cigare de manière à **maintenir une tension lorsque la batterie est débranchée** et éviter ainsi de gros problèmes.

Une fois ces précautions prises, desserrez les cosses, passez un coup de papier de verre sur les dépôts blanchâtres (sulfates) et remettez l'ensemble en place.

Si votre batterie continue de donner des signes de faiblesse, il existe 2 possibilités : soit le problème vient d'ailleurs, par exemple l'alternateur ne recharge pas correctement la batterie lorsque la voiture roule et n'alimente pas correctement les accessoires, soit la batterie est hors d'usage, et il faut dans ce cas la remplacer.

C'est la raison pour laquelle nous vous proposons un accessoire très simple, qui peut être réalisé sur un circuit imprimé miniature de façon à tenir dans une prise allume-cigare.

Grâce à la lumière produite par une LED bicolore, il indique l'état du système électrique de la voiture et donc de la batterie. Il peut être conservé à proximité dans la boîte à gants.

En utilisant une LED bicolore (rouge/verte), nous obtenons une **signalisation verte** si la tension est **supérieure à 12 VDC**, **jaune** si elle est **comprise entre 10,4 VDC et 12 VDC** et **rouge** si elle est **inférieure à 10,4 VDC**.

En dessous de 10 VDC, la batterie n'est pas seulement déchargée, mais si elle n'est pas rechargée rapidement, les plaques sont irrémédiablement corrodées par l'acide jusqu'à ce que la batterie soit inutilisable et ne soit plus rechargeable.

À noter que l'accumulateur au plomb a été inventé en 1859 par le Français **Gaston Planté**. Il a été en effet le premier à avoir mis au point la batterie rechargeable. À l'origine, les accumulateurs étaient situés dans des cuves en verre.

De nos jours, les batteries sans entretien se généralisent avec des cosses traitées anti-sulfatage, des plaques au plomb-calcium, supprimant le besoin de refaire le niveau de liquide, et donc permettant le scellement.

Le schéma électrique

Eh bien, cela étant dit, examinons maintenant le circuit de l'indicateur de charge. En vous référant au schéma électrique, vous pouvez voir qu'un comparateur d'échelle de tensions fournit en sortie des signaux qui alimentent les deux jonctions d'une LED bicolore rouge/verte à cathode commune.

La tension d'entrée du comparateur est reliée directement à la prise allume-cigare et représente donc la différence de potentiel entre le « + » et « - » du système électrique de la voiture.

Les deux tensions appliquées en référence au comparateur sont obtenues, toujours à partir de la tension du système électrique de la voiture, en dérivant une composante stabilisée au moyen de la **diode zener** de **9,1 V**.

Avoir des références de tension stabilisées permet d'obtenir un fonctionnement du comparateur indépendant des fluctuations de la tension du système électrique.

Si cela n'était pas le cas, c'est-à-dire si les références de tension étaient prises à partir de la tension d'alimentation sans aucune stabilisation, les seuils de commutation des comparateurs varieraient en fonction de la tension de la voiture et donc l'indication fournie par la LED serait erronée.

Si la tension de la batterie augmentait, les seuils augmenteraient également et la LED s'allumerait en jaune même si la batterie était bien chargée. Inversement, la LED pourrait s'allumer en vert alors que la batterie ne serait pas chargée.

Le « + » de l'indicateur d'état de batterie est relié au « +12 V » de la voiture (via la prise allume-cigare) qui alimente à la fois le diviseur de tension formé par les résistances R1 et R2 et la diode zener ZD1 à travers la résistance R6 de limitation du courant.

La diode zener ZD1 stabilise à 9,1 VDC la tension à ses bornes, qui est filtrée par le condenseur C1 afin de minimiser les perturbations électriques présentes dans le système électrique de la voiture (perturbations produites par exemple par le démarreur, les moteurs des essuie-glaces ou des lève-vitres électriques) qui pourraient altérer le fonctionnement des comparateurs.

Ainsi, **grâce à la diode zener qui stabilise la tension**, nous **alimentons correctement le double amplificateur opérationnel**, afin d'obtenir un **fonctionnement stable** des comparateurs.

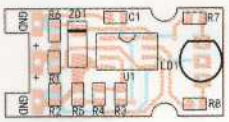
La tension zener alimente un autre diviseur de tension composé par les résistances R3, R4, et R5 qui fournit les tensions de référence à l'entrée non inverseuse de l'amplificateur opérationnel U1a et à l'entrée inverseuse de U1b.

L'amplificateur opérationnel utilisé est un **LM358** en version CMS, nous l'avons préféré au classique TL082, car il est conçu pour fonctionner avec une alimentation unique et sa sortie, avec une alimentation mono tension référencée à la masse, peut descendre pratiquement jusqu'à 0 V.

La configuration des amplificateurs opérationnels est celle d'un **comparateur d'échelle**, ainsi appelé, car les deux comparateurs ont successivement des références de tension. C'est à dire qu'ils déterminent le seuil de commutation des sorties respectives.

Celui de U1b est déterminé par la résistance R5 qui forme un diviseur de tension avec la somme des résistances R3

Plan de montage de l'indicateur d'état de batterie



Plan de câblage des composants.

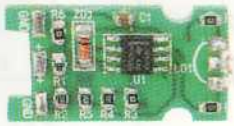


Photo de l'un de nos prototypes.

Liste des composants

R1..... 27 kΩ boîtier 0805
R2..... 68 kΩ boîtier 0805
R3..... 100 Ω boîtier 0805
R4..... 18 kΩ boîtier 0805
R5..... 82 kΩ boîtier 0805
R6..... 100 Ω boîtier 0805
R7..... 1,2 kΩ boîtier 0805
R8..... 1,2 kΩ boîtier 0805

U1..... LM358ADR
LD1.... LED bicolore Ø 3 mm
ZD1.... diode zener 9,1 V / 500mW

Divers :

prise allume-cigare
fusible en verre 5 x 20 mm 250 mA

NB : les typons des circuits imprimés à l'échelle 1 sont disponibles en téléchargement dans le sommaire détaillé de la revue.

et R4 (nous supposons que les amplificateurs opérationnels sont idéaux, et n'absorbent aucun courant sur les entrées).

Le seuil de U1a est déterminé par la somme de R4 et R5 qui forme un diviseur de tension avec R3.

Il en résulte que sur l'entrée inverseuse de U1b une tension de 7,46 V est appliquée, de même l'entrée non inverseuse de U1a est soumise à une tension de 9,09 V.

Les comparateurs ont les entrées en commun (la broche 2 de U1a est connectée avec la broche 5 de U1b) et alimentées par le diviseur de tension constitué par les résistances R1 et R2. Ce dernier réduit d'un facteur de 0,716 la tension du système électrique de la voiture.

En conséquence, le comparateur inférieur (U1b) a la sortie à 0 V lorsque la tension d'entrée du circuit est inférieure à 10,4 V, tandis que la sortie du comparateur supérieur (U1a), dans cette condition, présente un niveau élevé (un peu moins de 9 V).

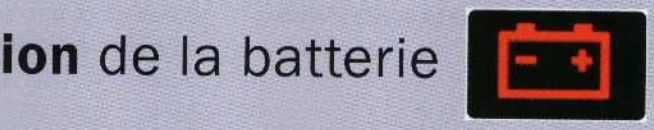
Par conséquent, avec une **tension de batterie inférieure à 10,4 V**, la LED bicolore **n'allume que la jonction** de couleur **rouge**.

Si la tension est plus élevée, mais **comprise entre 10,4 V et 11,6 V**, la sortie du comparateur U1a reste à un niveau élevé mais celle de U1b, puisque l'entrée inverseuse est maintenant moins positive que l'entrée non inverseuse, passe à un niveau haut et **active la jonction verte** de la LED bicolore.

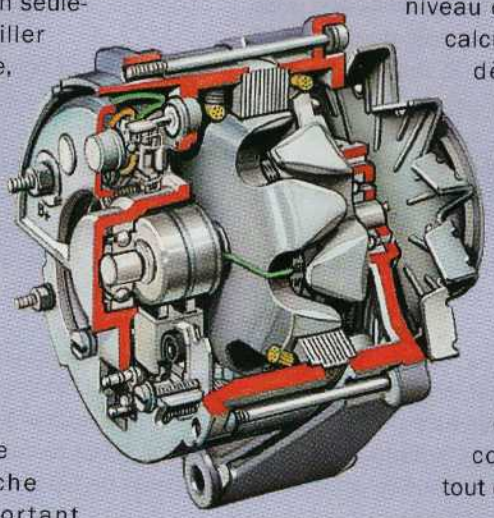
Comme **la lumière rouge est allumée en même temps**, la diode émet une lumière **jaune** (deux lumières, une verte et une rouge qui forment une lumière jaune).

Enfin, si la tension **dépasse 12 V,** ce qui correspond à un dépassement du seuil de commutation du comparateur U1a, la sortie de ce dernier passe à 0 V (car l'entrée inverseuse devient plus positive que l'entrée non inverseuse) et donc la LED **rouge s'éteint**.

La tension de la batterie



Le montage décrit dans ces pages est utile non seulement pour surveiller l'état de la batterie, mais aussi pour vérifier si elle est correctement chargée lorsque le moteur tourne et donc si l'alternateur fonctionne bien.



Le contrôle de la tension avec le moteur en marche est devenu important ces dernières années, car dans beaucoup de voitures modernes l'indicateur de charge de batterie traditionnel a disparu.

À l'époque, c'était une simple ampoule connectée entre la sortie positive de l'alternateur et le « + » du redresseur de l'alternateur dédié au contrôle du voyant (redresseur connecté en parallèle à celui de la puissance).

Lorsque la tension de charge de la batterie chutait parce que l'alternateur ne pouvait pas fournir le courant nécessaire, l'ampoule commençait à s'allumer faiblement jusqu'à ce qu'elle soit complètement allumée lorsque l'alternateur ne chargeait plus.

Aujourd'hui, la tension de charge est mesurée par le calculateur du véhicule via un bus de données, de type CAN BUS.

Dans le contrôleur de charge, se trouve un convertisseur de tension analogique/digital ADC qui échantillonne la tension et la numérise, puis l'envoie au calculateur de la voiture via le bus de données.

En dessous d'un certain niveau de tension, le calculateur considère que l'alternateur ne charge pas et allume une voyant au niveau du tableau de bord.

Au-dessus de ce seuil de tension, il considère que tout est correct.

Il arrive que l'alternateur fournisse une tension trop faible, en raison de l'usure des charbons ou du redresseur, la batterie va alors se charger de moins en moins bien.

Par contre, le calculateur ne le remarquera pas et n'allumera pas l'indicateur de défaut de la batterie. En effet, l'unité de contrôle considère que le seuil de la tension (malgré qu'il ne soit pas correct) est au-dessus du seuil de détection d'un défaut et considère que l'alternateur charge correctement.

Par exemple, si la batterie est maintenue à 12 V bien que l'alternateur lui fournisse 13 V, dans ce cas, il ne la charge pas correctement.

Cependant, l'unité de contrôle considère que tout va bien et n'allume pas le voyant de défaut de charge.

Normalement, le moteur tournant au ralenti, la tension doit être de 14,4 V en sortie de l'alternateur, il recharge donc correctement la batterie. La seule façon de démasquer cette situation est de surveiller la tension lorsque le moteur tourne.

Comme la sortie de U1b reste à un niveau élevé, la **LED bicolore s'allume en vert**.

Réalisation pratique

Le montage doit être **réalisé sur un circuit imprimé de petites dimensions** afin d'être **inséré dans une prise allume-cigare**, ou alors dans un petit boîtier à fixer sur le tableau de bord du véhicule.

Vous pouvez télécharger sur notre site dans le sommaire détaillé de la revue les typons à l'échelle 1 du circuit imprimé double face et/ou les fichiers Gerber pour une fabrication professionnelle.

Une fois le circuit imprimé réalisé par gravure, vous devez percer les trous pour la LED et les cosses, les autres composants sont soudés en surface (CMS).

Comme les composants sont des CMS, vous devez utiliser un fer à souder d'une puissance maximale de 20 W avec une panne de 0,5 mm de Ø (ou moins), de la soudure pour composants CMS (Ø 0,5 mm), une pince à épiler pour positionner les composants et éventuellement de la pâte à braser CMS pour faciliter les soudures.

Pour le câblage des composants référez-vous à l'encadré intitulé « **Plan de montage** ».

Rappelez-vous que la broche centrale de la LED bicolore représente la cathode commune, celle du côté du biseau est l'anode de la jonction rouge et la broche opposée correspond à l'anode de la jonction verte (voir la figure 1).

La cathode est la patte la plus longue, la patte la plus courte est l'anode de la jonction verte et celle restante est l'anode de la jonction rouge.

La LED bicolore doit être montée en pliant ces pattes de manière à former un angle droit (sans les casser) afin qu'elle puisse sortir de la prise allume-cigare, ou du boîtier (au tableau de bord) dans lequel vous avez monté le circuit.

Après avoir terminé le circuit, procurez-vous une prise allume-cigare, ouvrez-la et insérez le circuit en introduisant la LED au niveau du passe câble.

Reliez 2 fils aux extrémités « + » et « - » respectivement au contact du ressort central à travers un fusible et aux contacts latéraux.

Afin d'**éviter tout court-circuit accidentel** avec les contacts latéraux qui sont reliés à la masse, nous vous conseillons d'**isoler le circuit en l'insérant dans une gaine thermorétractable** (à chauffer avec un sèche-cheveux) ou en l'enveloppant dans du ruban isolant.

La même précaution est nécessaire si vous décidez de monter le circuit de manière permanente sur le tableau de bord. Dans ce cas, il est nécessaire de percer un trou pour faire sortir la LED.

Un autre type de montage peut être réalisé à l'intérieur d'un boîtier en plastique relié aux bornes positive et négative du neiman au moyen de 2 fils rouge et noir de 0,5 mm^2.

Si vous optez pour la prise allume-cigare, souvenez-vous de ces deux détails :

- dans le cas de certaines Audi récentes, la prise allume-cigare est dotée d'un régulateur de tension qui fixe la tension disponible à 12 V quelle que soit la tension fournie par l'alternateur lorsque le moteur tourne. Dans ce cas, vous ne pouvez pas insérer le circuit dans la prise allume-cigare, mais vous devez le connecter au système électrique, en localisant la ligne positive au niveau du neiman ;

- faites attention également aux véhicules dont la prise allume-cigare n'est pas commutable par la clé de contact, car dans ce cas, le circuit est toujours alimenté et malgré sa faible consommation, à la longue il contribue à décharger la batterie.

Dans les voitures modernes, cependant, la prise allume-cigare est commutée par un relais qui l'alimente lorsque les portes sont ouvertes et la déconnecte après quelques dizaines de secondes lors de leur fermeture ou immédiatement après la fermeture des portes.

Une fois l'installation terminée, si vous tournez la clé de contact et que vous voyez la LED s'allumer en rouge, il est très probable que vous ne puissiez pas démarrer (batterie déchargée).

Si la LED s'allume en vert, cela veut dire que la batterie est en forme.

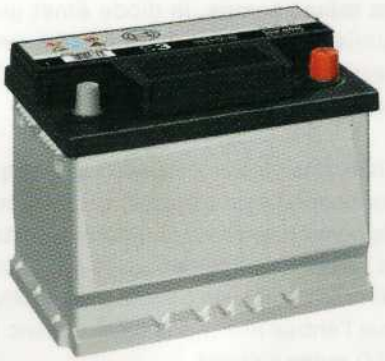
Si la LED s'allume en jaune, cela signifie que la batterie n'est pas bien chargée ou qu'elle commence à se décharger.

Vous devez alors la charger correctement, sinon il faudra envisager son remplacement.

Notez qu'un test plus approfondi peut-être effectué sur les véhicules diesel au début de l'hiver.

En effet, en tournant la clé en position MAR (allumage, c'est-à-dire en position de marche), les bougies de préchauffage sont alimentées pendant quelques secondes (allumage de l'indicateur correspondant au tableau de bord) et la batterie est mise à l'essai, car les bougies de préchauffage consomment au moins 10 ampères chacune.

Si lors de la phase de préchauffage des bougies, la LED s'allume en jaune, la batterie est dans un état acceptable, alors que si elle s'allume en vert cela signifie que la batterie est bien chargée et fournit le courant nécessaire aux bougies pour chauffer les cylindres et démarrer correctement. ■



ABONNEZ-VOUS

OUI, Je m'abonne à ELECTRONIQUE ET LOISIRS magazine LE MENSUEL DE L'ÉLECTRONIQUE POUR TOUS **A PARTIR DU N°** 145 ou supérieur → N°

E0144

Ci-joint mon règlement de ..... € correspondant à un abonnement de 4 revues Annuel

Règlement CB directement sur le site **www.electroniquemagazine.com rubrique Abonnement**

Adresser mon abonnement à :

Nom _____ Prénom _____

Adresse _____

Code postal _____ Ville _____

Tél. _____ e-mail _____

*Date, le*_____

Signature obligatoire ▷

TARIFS FRANCE

❑ **4 numéros** **30€,00**

TARIFS CEE/EUROPE

❑ **4 numéros** **34€,00**

DOM-TOM/HORS CEE OU EUROPE :

CONSULTEZ le site www.electroniquemagazine.com rubrique Abonnement

L'ASSURANCE de ne manquer aucun numéro en recevant votre revue directement dans votre boîte aux lettres près d'une semaine avant sa sortie en kiosques.

BÉNÉFICIER de 50 % de remise sur les CD-ROM.

POUR TOUT CHANGEMENT D'ADRESSE, N'OUBLIEZ PAS DE NOUS INDIQUER VOTRE NUMÉRO D'ABONNÉ (INSCRIT SUR L'EMBALLAGE)

Bulletin à retourner à : JMJ – Abo. ELM

B.P. 20025 - 13720 LA BOUILLADISSE – Tél. +334 427 063 96 – Fax +334 427 063 95

Directeur de Publication
Rédacteur en chef
Jean Marc MOSCATI
CD908
13720 Belcodène

Direction - Administration
JMJ éditions
B.P. 20025
13720 LA BOUILLADISSE
Tél. : +334 427 063 96

Secrétariat - Abonnements
Petites-annonces - Ventes
À la revue

Vente au numéro
À la revue

Publicité
À la revue

Maquette - Illustration
Composition - Photogravure
JMJ Editions SARL

Impression
Rotimpres
C/ Pla de l'Estany sn
17181 Aiguaviva (Girona)
Espagne

Distribution
MLP
55 Boulevard de la Noirée
38070 Saint-Quentin-Fallavier

Hot Line Technique
+334 427 063 96 non surtaxé
du lundi au vendredi de 16 h à 18 h

Web
www.electroniquemagazine.com

E-mail
support@electroniquemagazine.com

JMJ éditions
Sarl au capital social de 7800 €
RCS MARSEILLE : 421 860 925
APE 221E
Commission paritaire: 1221 K 79056
ISSN: 1295-9693
Dépôt légal à parution

EST RÉALISÉ
EN COLLABORATION AVEC :
NUOVA ELETTRONICA
Elettronica In

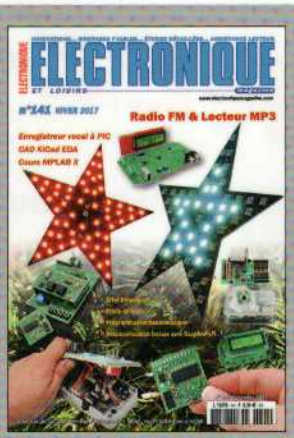
13 ,10 €* la revue frais de port inclus pour la France Métropolitaine

*Frais d'expédition (CEE, les DOM-TOM et autres pays), contactez-nous pour un devis ou bien à calculer directement sur notre site

Articles, Revues et CD téléchargeables au format PDF sur Internet : www.electroniquemagazine.com

Au sommaire : Construisons une tondeuse autonome à l'aide de composants de récupérations - Amplificateur 4 W pour guitare qui dispose d'un réglage de volume, de tonalité et un système de distorsion. Notification push avec RaspberryPi, service web permettant d'envoyer des notifications d'événements avec des smartphones sous iOS et Android.- Whiff, le purificateur/ ventilateur d'air pour la maison. Transformez votre oscilloscope en analyseur de spectre. - Imprimante 3D VERTEX - 5. Variateur de vitesse pour moteur à courant continu, permet de faire varier la vitesse de rotation de n'importe quel moteur électrique à balais de 12 VDC à 35 VDC - Cours Arduino 9 ème partie - Cours de programmation sur iPhone -5

Au sommaire : Sustentation magnétique, suspendre une ampoule sans fil pour la maintenir, l'allumer sans aucune pile ou câble - Compteur d'énergie électrique 220 VAC, doté d'un afficheur LCD qui visualise la consommation instantanée et globale Générateur de bruit blanc et rose, ce montage génère un bruit blanc et rose dont la fréquence de base peut atteindre 1 MHz - Vision numérique avec RaspberryPi et Caméra Pi - Chargeur pour smartphone intelligent - Imprimante 3D VERTEX, le logiciel Repetier-Host - Cours Arduino X - CHIPKIT PI : comment rendre le RaspberryPi plus polyvalent - Veilleuse à LED universelle - Cours de programmation sur iPhone – VI



Au sommaire : Reconnaissance faciale avec RaspberryPi, nous allons utiliser les fonctionnalités de la librairie SimpleCV afin de reconnaître les visages et les autres détails du corps humain dans des images - Programmateur hebdomadaire - Radio FM & lecteur MP3 USB - Enregistreur vocal avec PIC & EEPROM - 1. Apprenez à maîtriser KiCad EDA – 1 . Effet trémolo pour guitare, un effet qui fait vibrer le son des cordes à travers un système optique original de modulation d'amplitude. Cours MPLAB X IDE - 1, C'est l'occasion d'apprendre à connaître les microcontrôleurs de la famille PIC32 de Microchip. Étoile de Noël - Boule de Noël à changement de couleur, en utilisant la technologie « LED Neopixel ».

Au sommaire : Un amplificateur Hi-Fi 200 W à MOSFET, il pourra sonoriser correctement des pièces volumineuses sans la moindre distorsion - Chambre d'écho digitale - RandA : connectons RaspberryPi avec Arduino avec une carte de développement qui permet de créer une passerelle entre le monde du RaspberryPi et le monde d'Arduino - Enregistreur vocal avec PIC & EEPROM - 2 Cours MPLAB X IDE – 2 pour découvrir MPLABX IDE - Apprenez à maîtriser KiCad EDA – 2, nous poursuivons la description du logiciel de CAO Open Source KiCad - Comment ponter un amplificateur - Contrôleur pour moteur DC - Télécommande infrarouge 4 canaux.

Au sommaire : Apprenons à utiliser RandA, programmation et l'utilisation de la carte - STOP au BLACKOUT ! , système de mesure et de limitation automatique de la consommation électrique jusqu'à 6 kW sous 220 VAC. - Chauffage par induction 1 kW, La technologie ZVS permet une régulation de la tension à l'aide d'une « commutation douce» Détecteur de flamme - Enregistreur vocal avec PIC & EEPROM - 3. Cours MPLAB X IDE – 3, le nouvel environnement de développement intégré produit et distribué par Microchip Technology.. Récepteur 1 & 2 canaux 433,92 MHz compatible MM53200, UM86409 et UM3759, Chaque récepteur peut être combiné jusqu'à 10 télécommandes

CD-ROM ENTIÈREMENT IMPRIMABLE

CD 6 Numéros 25,61 €* / CD 12 Numéros 45,61 €*

50%

de remise pour nos abonnés sur tous nos CD

*CD - FRAIS DE PORT INCLUS POUR LA FRANCE METROPOLITAINE (DOM-TOM ET AUTRES PAYS : NOUS CONSULTER).

JMJ EDITIONS - CD 908 - 13720 BELCODÈNE. Règlement par chèque à l'ordre de JMJ ÉDITIONS et/ou règlement par Paypal ou CB sur notre site : www.electroniquemagazine.com - Tél. : 04 42 70 63 96