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

ELECTRONIQUE

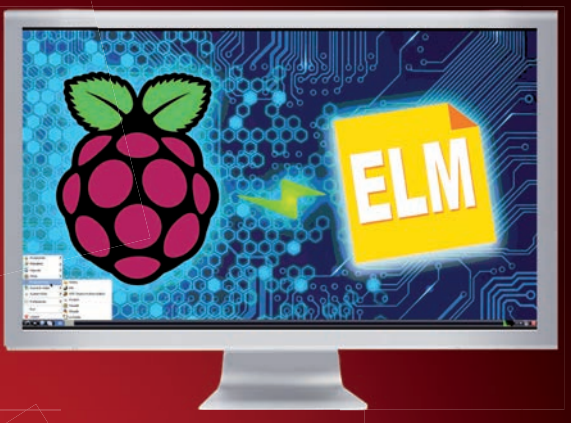
ELECTRONIQUE

ET LOISIRS magazine

http://www.electroniquemagazine.com

n°126 PRINTEMPS 2014

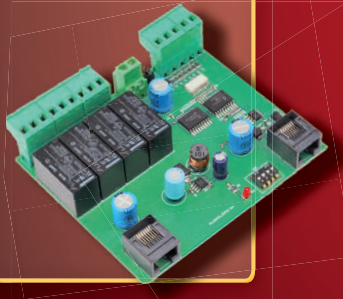
Syntèse vocale pour RASPBERRYPI



CAPACIMÈTRE de 10 pF à 10 000 µF

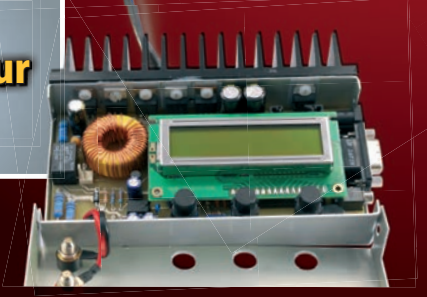
POUR L'AUTOMATISATION

MINI BUS



- 3DRAG : les logiciels
- Alimentation symétrique ± 1,25 V à ± 18 V
- Amplificateur pour MP3 & Smartphones

Chargeur d'accumulateur universel



N° 126 Mars 2014

M 04662 - 126 - F: 7,50 € - RD

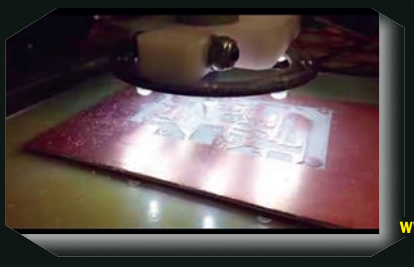
France 7,50 € – DOM 7,50 € – CE 8,00 € – NC (S)900FP – Poly.fr 1200CFP – Canada 11,50 $C

Une imprimante 3D

Cette imprimante 3D permet de créer des objets en plastique de formes diverses d'une taille maximum de 20 x 20 x 20 cm à l'aide de fil en ABS ou PLA de 3mm. L'impression est extrêmement rapide et précise, même à vitesse élevée. Destinée au grand public, elle a été conçue et fabriquée en profilés d'aluminium pour offrir maniabilité, légèreté et rigidité et supprimer les vibrations et résonances indésirables. Elle est compatible avec tous les softwares et firmwares RepRap (disponible gratuitement). L'imprimante utilise les axes X et Y pour le plan d'impression et Z pour la tête d'extrusion. Cette configuration particulière permet de simplifier le système d'extrusion qui n'a plus à se déplacer sur un axe horizontal. Il est simplement fixé sur la structure qui se déplace dans l'axe Z. Les dimensions ont été étudiées pour rendre l'imprimante compacte avec un centre de gravité bas aligné sur les deux couroies. Le support d'impression est conçu pour accueillir une plaque de vetronite qui offre une bonne adhérence au PLA avec une remarquable stabilité dimensionnelle .

CARACTÉRISTIQUES :

- Structure : Profilé en aluminium rigide et léger.
- Montage simplifié avec joints métalliques et inserts en polyoxyméthylène (usiné avec un pantographe cnc (delrin™)
- Plateau d'impression sur axes X / Y
- Extrudeuse (moteurs pas à pas) sur axe Z
- Taille d'impression maximum 20 x 20 x 20 cm
- Résolution X et Y : 0,015 mm - Z 0,39 micron
- Buse de 0,5 mm compatible PLA et ABS
- Vitesse d'impression typique 120 mm / sec
- Vitesse d'impression maximale : 150 à 300 mm / sec (en fonction de l'objet à imprimer)
- Dimensions plaque de support d'impression: 21,5 x 21,5 cm
- Largeur 60 cm • Profondeur : 43 cm • Hauteur : 59 cm • Poids : 8,7 kg
- Alimentation : 12V 3A max (fournie)



Nouveau ! Réalisez vos circuits imprimés avec la 3DRAG

Vidéo disponible sur :

www.3dprint.electroniquemagazine.com

Lot de 5 feuilles adhésives pour imprimante 3D

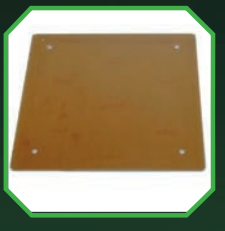
Réf. A3D-FEUILLES3D **10,90 €**

Tube de pate thermique - 5grammes

Réf. PASTADISSIP **3,40 €**

Ensemble plateau pour imprimante 3D

Réf. A3D-33 **12,90 €**



Ruban KAPTON 8mm longueur 33m

Réf. A3D-KAPTON833 **4,90 €**

Plateau chauffant pour imprimante 3D

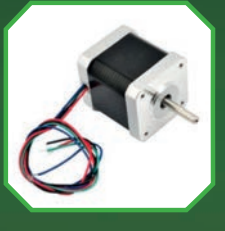
Réf. A3D-3DHEATERPLATE **28,90 €**

Ruban KAPTON 55mm longueur 33m

Réf. A3D-KAPTON5533 **17,96 €**

Moteur pas à pas bipolaire NEMA 17 – 2.5A

Réf. A3D-STEPMOT **21,60 €**



Extrudeur à section chauffante pour le 3DRAG

Réf. A3D-31 **55,80 €**

Thermistance NTC 100K pour plateau chauffant de la 3DRAG

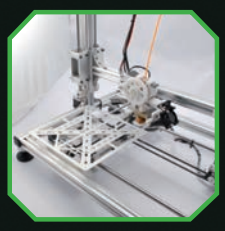
Réf. A3D-NTC100K **5,40 €**

Comelec 03 / 2014 - Prix TTC

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

à la portée de tous

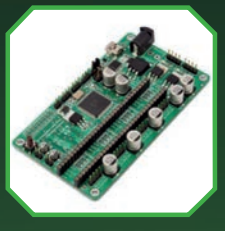




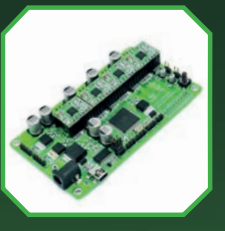

3DRAG
(Version montée) 879,00 €

3DRAG
(en Kit) 689,00 €

Carte contrôleur sans driver pour la 3DRAG
Réf. 3DCONTROLLER 42,90 €



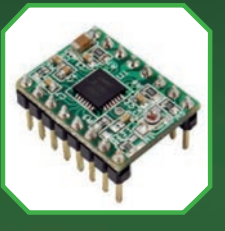
Carte contrôleur complète avec driver pour la 3DRAG
Réf. 3DCONTR-DRIVER 78,60 €



Bobines ABS

- 1kg- 3mm de diamètre - Noire **31,50 €**
- 1kg- 3mm de diamètre - Blanche **27,00 €**
- 2Kg- 3mm de diamètre : noire, jaune, verte **55,90 €**

Driver pour carte contrôleur de la 3DRAG
Réf. 3DDRIVER 9,90 €



Bobines de PLA

- 1KG – 3mm de diamètre : Naturelle, or, blanche, Noire **25,50 €**
- 1KG – 3mm de diamètre : Jaune, orange, rouge, rose, vert pomme, turquoise, bleue **27,00 €**
- 2.3KG – 3mm de diamètre : Noire, naturelle, jaune, orange, rouge, rose, vert pomme, turquoise, bleue **52,20 €**

Sommaire

ARTICLES

Été 2014
n°126

Les typons des circuits imprimés et les programmes **lorsqu'ils sont libres de droits** sont téléchargeables à l'adresse suivante :

www.electroniquemagazine.com

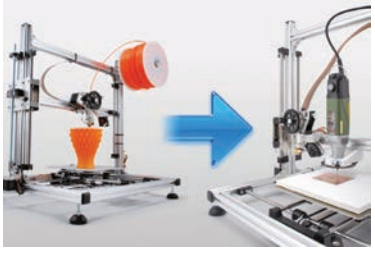
Allez dans le sommaire de la revue 126 et à l'onglet « Télécharger»

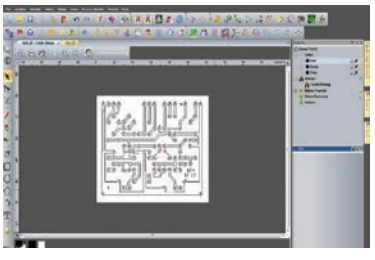
L'EDITO PRINTEMPS 2014

Chères lectrices, chers lecteurs, La Rédaction d'Electronique et Loisirs Magazine remercie les lectrices et les lecteurs qui contribuent au succès de la revue ainsi que de nos sites internet :

electroniquemagazine.com, **3dprint.electroniquemagazine.com**, **raspberrypi.electroniquemagazine.com.**

Avec l'arrivée du printemps nous travaillons ardemment pour transformer notre imprimante 3DRAG en graveuse de circuits imprimés (voir la vidéo sur notre site). Nous avons terminé la première étape qui consiste à graver un circuit imprimé directement à partir d'un fichier GERBER réalisé sous EAGLE. Nous vous proposerons l'article dans le prochain numéro. La deuxième étape permettra de fabriquer directement un circuit avec la 3DRAG à partir d'un fichier image au format très répandu BMP. Nous espérons que cela rendra d'immenses services aux passionnés d'électronique qui sont souvent freinés par la fabrication des circuits ainsi qu'aux petites entreprises (voir la page 96).





La 3DRAG sans secrets

Dans ce numéro nous allons aborder les logiciels et les aspects matériels de l'imprimante 3DRAG.

de Gabriele Daghetta
testé par Simone Majocchi

Parmi les avantages que représentent le matériel et les logiciels **Open Source**, le plus remarquable est la possibilité de connaître tous les détails et d'accéder à tous les réglages des paramètres afin de modifier le fonctionnement de l'appareil. Combien de fois, en se concentrant sur un objet, il nous est venu à l'esprit que nous pourrions améliorer son fonctionnement en changeant ceci ou cela ?

Avec les produits commerciaux, une modification annule la garantie et demande un peu « d'ingénierie inverse » pour comprendre le fonctionnement du produit et comment le modifier. En plus cette action peut même enfreindre la loi sur la « protection de la propriété intellectuelle » (dite IP en anglais).

Par contre avec les produits **Open Source**, nous avons déjà à disposition toutes les informations nécessaires et les codes sources des programmes, ce qui permet d'étudier et de mettre en œuvre nos modifications de la meilleure façon possible. Ce qui nous est exigé en retour est un engagement moral de repartager notre travail avec la communauté, de façon à maintenir le cercle vertueux du développement et de l'amélioration du monde libre.

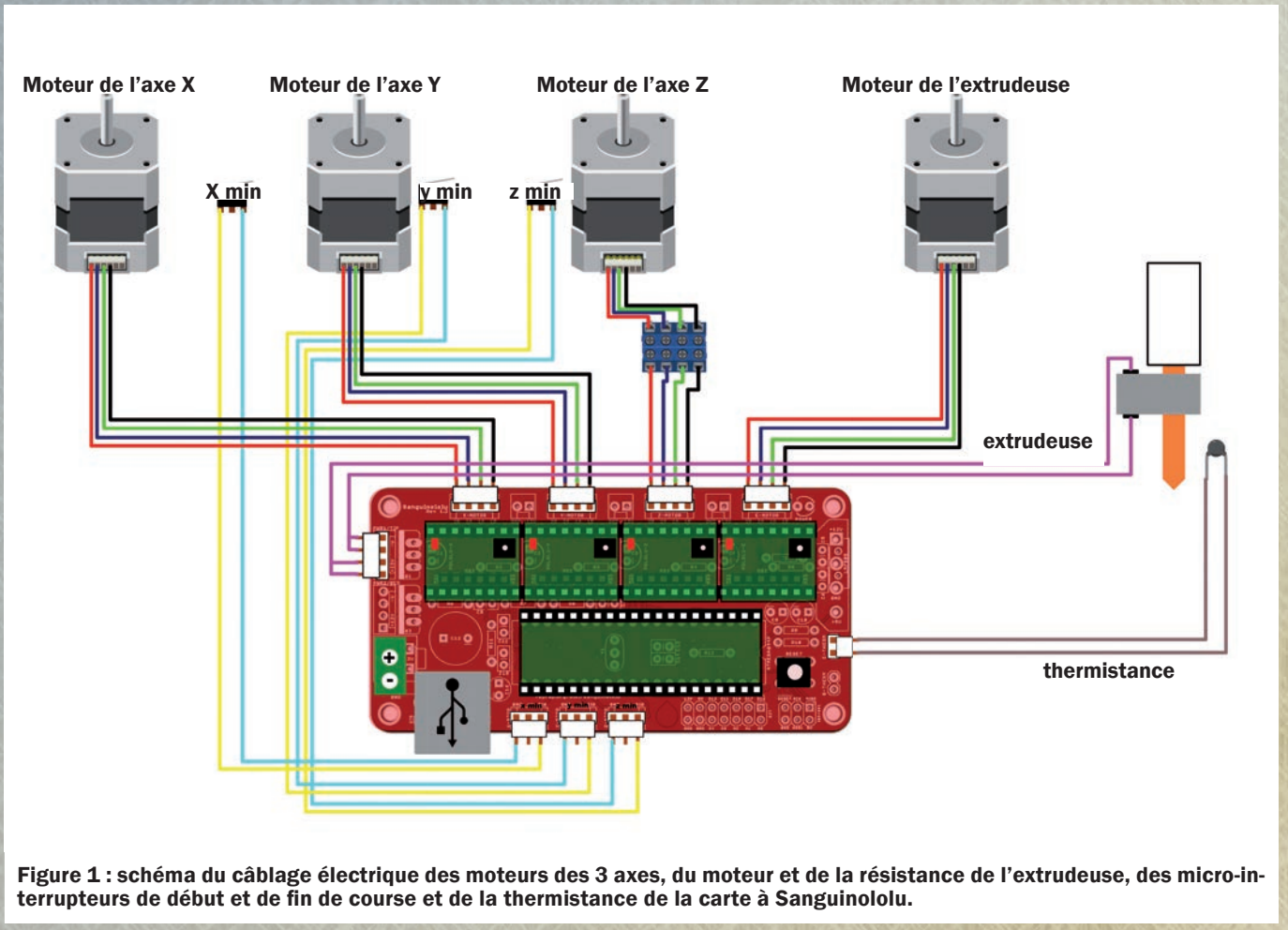


Figure 1 : schéma du câblage électrique des moteurs des 3 axes, du moteur et de la résistance de l'extrudeuse, des micro-interrupteurs de début et de fin de course et de la thermistance de la carte à Sanguinololu.

La **3DRAG** appartient à la famille **Open Source** et est donc conçue pour être en mesure de profiter des nouveaux développements logiciels et d'autre part pour être une source d'inspiration pour d'autres développements d'imprimantes **RepRap**. Par rapport au concept original, la **3DRAG** ne réplique pas les pièces pour se reproduire, cela est une conséquence de l'approche « industrielle » pour la réalisation des pièces, cependant toutes les informations relatives aux pièces mécaniques sont disponibles pour aider ceux qui veulent utiliser une CNC et leur matériel afin de réaliser indépendamment une imprimante 3D.

Le Firmware

Le logiciel résidant dans la carte électronique **Sanguinololu** de la **3DRAG** est chargé de **gérer les fonctions de base de l'imprimante et d'interpréter les différentes commandes du G-Code pour les transformer en mouvements effectués par les moteurs et l'extrudeuse**. Ce code est disponible en tant que source pour toutes les versions publiées en **Open Source** et également disponible sur notre site **www.3dprint.electroniquemagazine.com**. Ce programme a été réalisé par plusieurs auteurs en respectant certaines conventions du projet **RepRap**, avec la capacité de définir à travers un fichier de configuration les caractéristiques mécaniques et électroniques de l'imprimante.

Chaque programme est conçu pour gérer trois moteurs, un pour chaque axe de déplacement, un moteur pour le déplacement de l'extrudeuse, un dispositif de chauffage ainsi que la commande de la thermistance, une plaque chauffante, un ventilateur de refroidissement, et les micro-interrupteurs pour les positions de début et de fin de course. Ensuite les auteurs ajoutent des fonctionnalités spécifiques au programme telles que la gestion d'un lecteur de carte SD, la gestion de boutons avec un écran de contrôle LCD, ou développent des algorithmes pour l'optimisation des déplacements et des mouvements afin d'améliorer l'impression d'une voûte, ou de figures géométriques complexes.

Il est important de se rappeler que le microcontrôleur dispose d'un espace mémoire limité et que l'on ne peut donc pas ajouter d'algorithmes et/ou de fonctions complexes sans tenir compte de cette limite. La solution à ce problème est l'utilisation de microcontrôleurs avec plus d'espace mémoire, comme par exemple l'utilisation d'un **ATmega 1284P** à la place d'un **ATmega 644P**. Ils sont compatibles broche à broche, mais la taille de la mémoire passe de 64K à 128K pour la Flash, de 4K à 16K pour la RAM et de 2K à 4K pour l'**EEPROM**. Ce microcontrôleur peut travailler à une fréquence de 20 MHz contre 16 MHz pour le 664P, ce qui permet une exécution plus rapide du code. Retournons à la carte électronique et plus précisément à l'interface des moteurs et des micro-interrupteurs.

Le mode actuel de fonctionnement de chacune de ces parties dépend de leurs caractéristiques électriques et mécaniques et pour cette raison, le fichier de configuration vous permet de spécifier des variables afin de les adapter.

La configuration de la 3DRAG

Dans cet article, nous allons vous expliquer tous les éléments spécifiques au matériel et au fonctionnement de la **3DRAG** se référant aux variables utilisées dans les fichiers de configuration du firmware. Nous avons choisi comme référence le firmware **Marlin**, les variables seront celles de ce firmware, mais leurs noms et leurs descriptions vous aideront à trouver des variables homologues dans d'autres firmwares, dans le cas où vous vous décidiez à les essayer. Il en est de même pour la mise à jour de **Marlin** à partir d'une version ultérieure. Le processus de mise à jour sera expliqué dans cet article.

Commençons par dire que chaque firmware a besoin de connaître le type de carte utilisée dans la **3DRAG**. Dans notre cas c'est une **Sanguinololu 1.2**, généralement identifiée par le code 62 dans la liste :

```
/ Sanguinololu 1.2 and above = 62
#define MOTHERBOARD 62
```

Avec cette information, le firmware est compilé avec la configuration correcte des broches qui servent à attribuer les différentes ressources matérielles numériques et analogiques, tels que les modules de puissance pour les moteurs pas à pas et les micro-interrupteurs de début et de fin de course.

Une **mauvaise configuration de ces paramètres** provoque un **dysfonctionnement** de l'imprimante. Voici le schéma (voir la figure 1) de câblage de cette carte afin que vous puissiez avoir une référence au niveau des connexions électriques des moteurs, des micro-interrupteurs et de l'extrudeuse.

Le paramètre suivant détermine le type de **CTN** utilisé, les thermistances sont caractérisées par des paramètres spécifiques (les **coefficients A**, **B** et **C**) utilisés dans l'équation de **Steinhart-Hart** qui permet de déterminer avec précision la relation entre la **valeur exprimée en ohms** et la **température mesurée**. La relation de **Steinhart–Hart** modélise l'**évolution de la résistance électrique d'un semi-conducteur selon sa température**.

Cette loi peut s'écrire :

$$\frac{1}{T} = A + B\ln(R) + C(\ln(R))^3$$

T : température en **kelvins**
R : résistance électrique en Ω
A, **B**, **C** : les **coefficients de Steinhart-Hart** qui caractérisent chaque **thermistance**.

Note : le terme **$(\ln R)^2$** est **négligeable** devant les autres coefficients.

Si les coefficients ne sont pas corrects, la température peut s'écarter sensiblement de la valeur réelle, et donc la fusion du filament ne sera pas correcte, ce qui est à éviter. Sur la **3DRAG** nous utilisons une **CTN** de **100 kΩ** qui n'est pas **jumelée avec celle de la plaque chauffante** qui utilise un **autre type de CTN**.

Pour cette raison, la zone allouée aux réglages des capteurs doit être de **type 5** pour **Temp_Sensor_0** et nous devons également veiller à ce qu'il n'y ait pas de capteur associé avec le plaque chauffante (**Bed_Sensor_0**), sinon le firmware attendra un **seuil de température qui ne sera jamais atteint et l'impression ne commencera pas**. Les paramètres du capteur sont généralement contenus dans une section spécifique, comme celle ci-dessous.

```
//===========================================
//=============Thermal Settings ==============
//===========================================
// 5 is 100K thermistor - ATC Semitec
// 104GT-2 (4.7k pullup)
#define TEMP_SENSOR_0 5
...
#define BED_SENSOR_0 0
```

La section suivante contrôle la logique des micro-interrupteurs de début de course, ces trois micro-interrupteurs permettent au firmware de **détecter individuellement le début de la zone utile** des deux **axes X** et **Y**, avec le point à partir duquel doit commencer le dépôt de matériau sur la plaque d'impression (**axe Z**). La flexibilité du firmware dans ce domaine est très grande. Les micro-interrupteurs peuvent également être placés à la fin de la course et non pas au début, pour ce faire le firmware définit la position au **point 0, 0, 0**.

De même, les micro-interrupteurs peuvent être définis comme normalement ouverts ou fermés. Nous avons choisi d'utiliser le mode « **Normalement Fermé** » (**NC**) afin d'arrêter immédiatement le moteur si la connexion est interrompue ou si le micro-interrupteur est défectueux.

Ce mode nécessite la **valeur booléenne** « **false** » (**faux**) pour les quatre constantes qui indiquent au firmware si la logique est inversée ou pas (dans notre cas elle ne l'est pas, et les micro-interrupteurs ouvrent le circuit quand ils sont actionnés). En général, si un interrupteur a un rôle important, comme l'arrêt d'un moteur au début ou en fin de course, ou un verrou de sécurité lorsqu'il est actionné, il est de **bonne pratique de l'utiliser en mode NC** (**normalement fermé**) de manière à avoir une **protection par rapport à son dysfonctionnement** et/ou à l'interruption des connexions. Au contraire en **mode NO**, lors d'un défaut ou lorsqu'un interrupteur est déconnecté, **on ne peut ni arrêter le moteur** à la fin de la course ni l'ensemble du dispositif en cas d'urgence.

Un autre élément choisi en option est l'utilisation d'interrupteurs mécaniques. Certains modèles utilisent des capteurs optiques alimentés, tandis que pour les interrupteurs mécaniques de par leur nature ils ne nécessitent que d'une

résistance de pull-up pour éliminer tout bruit sur les entrées haute impédance. Les constantes qui activent les résistances sont généralement déjà mises à la valeur correcte.

```
const bool X_ENDSTOPS_INVERTING = false;
// set to true to invert the logic of the endstops.
const bool Y_ENDSTOPS_INVERTING = false;
const bool Z_ENDSTOPS_INVERTING = false;
```

Une fois la logique définie, nous devons maintenant transmettre au firmware la position de l'interrupteur par rapport à la position 0 de chaque axe qui peut-être soit en début de course ou en fin de course. Si elle se trouve au **début** (valeur « **-1** » ou **MIN**), le firmware déplacera la plaque d'impression vers la position du début de l'axe (0) et s'arrêtera lorsque l'interrupteur sera actionné. Si l'interrupteur a été fixé et positionné sur la valeur maximale de l'axe (à la fin), il va transmettre la valeur « **1** » et le firmware déplacera le plateau vers la fin de course pour activer l'interrupteur, puis retournera à sa position initiale spécifique à l'axe.

Pour des raisons de commodité de réglage des coordonnées **X = 0**, **Y = 0** et de l'**étalonnage de fin de course**, les interrupteurs sont placés au **début de la course des axes**, et ensuite les valeurs des trois constantes sont mises à « **-1** » :

```
// ENDSTOP SETTINGS:
// Sets direction of endstops when homing;
// 1=MAX, -1=MIN
#define X_HOME_DIR -1
#define Y_HOME_DIR -1
#define Z_HOME_DIR -1
```

Comme nous l'avons indiqué auparavant, lors de la recherche de la position « **home** » qui correspond au « **0** » pour chaque axe, le firmware tient compte des **mesures exprimées en millimètres**, sur chacun des axes, afin d'éviter des **dommages** dus au dépassement de la **limite physique mécanique**.

Etant donné que nous n'utilisons les interrupteurs qu'en « début de course » et non pas un couple « début / fin de course » pour chaque axe, le firmware requiert les valeurs minimales et maximales au-delà desquelles, une fois la remise à zéro effectuée, les moteurs s'arrêtent. Dans le cas de la **3DRAG**, la limite est pratiquement égale au volume de l'impression et la « marge » mécanique de chaque côté est de l'ordre d'environ un centimètre. Définir les valeurs minimales et maximales de la surface d'impression évite ainsi le risque d'atteindre les limites physiques de déplacement. Avec d'autres imprimantes **3D**, ces valeurs peuvent être plus grandes de quelques centimètres que la zone d'impression utilisable, ce qui justifie la nécessité d'insérer des valeurs appropriées dans le firmware pour un matériel spécifique.

Notez qu'avant d'informer le firmware de la position réelle du plateau et de l'extrudeuse, il faut bloquer le programme pour le mouvement au-delà de ces valeurs minimales et maximales, car il n'y a pas de références fiables. **Évitez de déplacer manuellement la mécanique avant d'utiliser la commande « home »** pour chaque axe (la commande « home » réinitialise les axes X, Y et Z). Pour la **3DRAG** les valeurs des axes X et Y vont de 0 à 200 mm, tandis que pour l'axe Z, elle peut aller jusqu'à 220 mm.

```
// Travel limits after homing
#define X_MAX_POS 200
#define X_MIN_POS 0
#define Y_MAX_POS 200
#define Y_MIN_POS 0
#define Z_MAX_POS 220
#define Z_MIN_POS 0
```

Voyons maintenant les poulies et les courroies qui, selon la manière dont elles sont montées, déterminent le déplacement du plateau d'impression et de l'extrudeuse dans un sens ou dans un autre, selon le sens horaire ou anti-horaire de rotation de chaque moteur. Il suffit de monter un moteur dans une partie de la structure plutôt que dans une autre pour inverser le sens de déplacement, comme il suffit de fixer la courroie au-dessus ou au-dessous pour inverser la direction.

Dans la pratique, le plateau se déplace dans un sens ou dans l'autre en fonction de la combinaison de tous ces éléments et bien sûr le concepteur du logiciel a fait en sorte que le processus puisse être géré avec facilité, moteur par moteur. Le paramètre est défini par « **Invert_ <axe> _ Dir** » pour plus de simplicité et peut prendre la valeur « **vrai** » ou « **faux** ». Avec la **3DRAG** la combinaison du positionnement du moteur, de la poulie d'accouplement et de l'agencement de la courroie implique que les **trois axes X**, **Y** et **Z** doivent être définis comme « **faux** », ce qui signifie qu'ils ne doivent pas être inversés en tant que mouvement, tandis que l'extrudeuse doit être inversée, car il y a un petit pignon qui entraîne la grande roue dentée fixée sur le mécanisme et détermine ainsi l'inversion du sens de rotation. Il s'agit d'une situation courante pour la plupart des extrudeuses de type **Wade** avec le moteur sur le côté gauche.

```
#define INVERT_X_DIR false   //
#define INVERT_Y_DIR false   //
#define INVERT_Z_DIR false   //
#define INVERT_E0_DIR true   //
#define INVERT_E1_DIR true   //
#define INVERT_E2_DIR true   //
```

Le dernier paramètre lié à la mécanique sur lequel il convient de s'arrêter est celui qui détermine le comportement du moteur lorsqu'il n'est pas utilisé pour le mouvement tout au long du processus d'impression.

Les moteurs pas à pas, lorsqu'ils sont alimentés, sont bloqués sur une position et exercent une **forte résistance au mouvement**, ce qui a pour effet secondaire une **consommation importante de courant** et par conséquent l'**échauffement du moteur** et de l'**électronique de commande**. Lors de l'impression, il est donc logique de garder les deux axes **X** et **Y actifs** (alimentés) tandis que l'axe **Z** peut être **désactivé** (non alimenté) pendant tout le temps où la couche est imprimée.

Même l'extrudeuse doit être maintenue alimentée pour éviter que la pression de la matière plastique fasse reculer le filament pendant les déplacements sans impression. Cependant, après une certaine période d'inactivité lorsque l'impression est terminée, les moteurs sont éteints pour ne pas surchauffer, cet intervalle fait partie des paramètres d'impression de l'application.

```
// Disables axis when it's not being used.
#define DISABLE_X false
#define DISABLE_Y false
#define DISABLE_Z true
#define DISABLE_E false // For all extruders
```

L'ultime paramètre spécifique à la mécanique de la **3DRAG** est le **nombre de pas de chaque moteur dont le mouvement doit correspondre à une unité de mesure** (dans notre cas c'est **1 mm**). Le calcul prend en compte le diamètre de la poulie, le pas de vis, le rapport d'engrenage et le diamètre de la roue dentée de l'extrudeuse. Tout doit être élaboré en fonction des paramètres de la carte et des pilotes qui peuvent gérer 4, 8 ou 16 micropas. Tous les moteurs de la **3DRAG** fonctionnent à **16 micropas**, une révolution complète compte **3200 micropas**. Sans trop compliquer, nous pouvons dire que le nombre de micropas est égal à **64,25** pour les **axes X** et **Y**, **2560** pour l'**axe Z** et **654** pour l'**extrudeuse**. Pour ce dernier, le comptage a été effectué sur la base effective de la longueur de fil utilisée par rapport à celle prévue dans le logiciel de slicing, nous trouvons ainsi une valeur proche de la réalité.

```
// default settings
#define DEFAULT_AXIS_STEPS_PER_UNIT {64.25,64.25,2560,654} //
```

Maintenant passons à la section vitesse et accélération : ces valeurs indiquent le maximum qui peut être appliqué par le firmware et sont utilisées pour faire fonctionner la mécanique aussi efficacement et rapidement que possible, sans mettre la structure en résonance ou atteindre la limite physique, ce qui pourrait introduire des erreurs et des distorsions au niveau de l'impression. Le premier ensemble de valeurs définit la vitesse maximale des axes linéaires et de l'extrudeuse. Dans le cas de **3DRAG**, la valeur peut être augmentée jusqu'à **500 mm/sec** (c'est à dire un déplacement d'un demi-mètre par seconde, ce qui équivaut à **30 mètres par minute**) sur les deux axes du plan d'impression et sur l'extrudeuse, ces valeurs ne peuvent pas être appliquées sur n'importe quelle imprimante. L'**axe Z** est limité à **50 mm/s** car il est très démultiplié (2560 pas pour effectuer 1 mm) et une vitesse supérieure à 5 cm par seconde exigerait une rotation du moteur au-delà de ses caractéristiques de fonctionnement normal.

```
#define DEFAULT_MAX_FEEDRATE {500, 500, 50, 500}
// (mm/sec)
```

L'augmentation de la vitesse est définie par l'accélération exprimée ici en **mm/seconde x seconde** (**mm.s^{-2}**) ce qui correspond au départ du moteur arrêté jusqu'à la vitesse requise.

Évidemment, il n'est pas pratique d'avoir un moteur qui démarre et s'arrête instantanément, car il provoquerait des vibrations considérables à l'ensemble de la structure mécanique de l'imprimante (pensez à ce qui se passe lorsque vous lâchez d'un coup la pédale d'embrayage d'une voiture), de la même manière les impressions sont souvent faites par de petits déplacements, si nous avons une faible accélération, le moteur peut ne pas atteindre la vitesse requise. De même des accélérations trop grandes engendreraient des vibrations de la structure et des mouvements saccadés, par exemple lors d'un remplissage d'une partie étroite, il faut donc définir une valeur qui soit en conformité avec les types d'objets que l'on souhaite imprimer.

```
// X, Y, Z, E maximum start speed for accelerated moves.
#define DEFAULT_MAX_ACCELERATION {9000,9000,100,10000}
```

Les paramètres d'accélération pour l'impression standard et la rétraction du fil sont définis dans deux lignes spécifiques et lors de nos essais nous avons adopté la valeur **1 000** qui semble bien adaptée à la structure de l'imprimante.

```
// X, Y, Z and E max acceleration in mm/s^2 for printing moves
#define DEFAULT_ACCELERATION          1000
// X, Y, Z and E max acceleration in mm/s^2 for r retracts
#define DEFAULT_RETRACT_ACCELERATION  1000
```

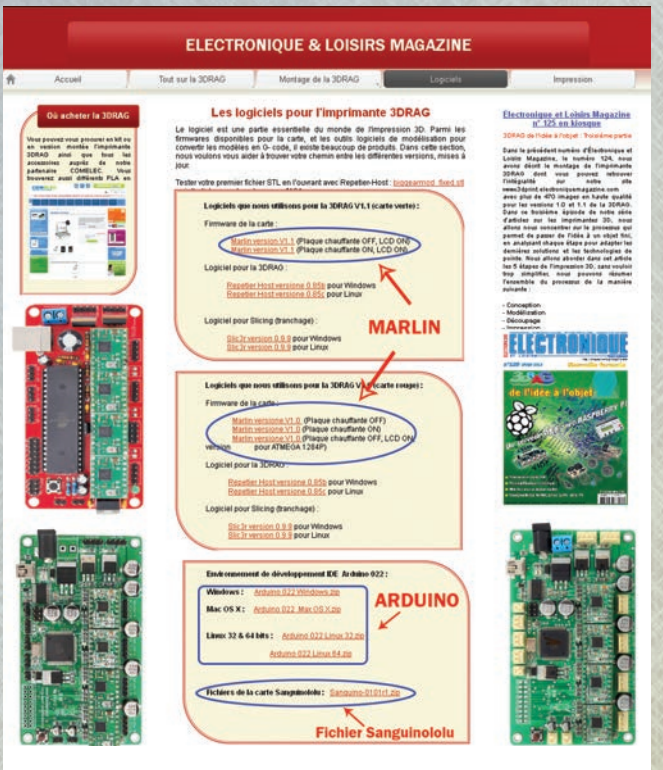


Figure 2 : sur le site www.3dprint.electroniquemagazine.com, vous pouvez télécharger le firmware Marlin en allant dans l'onglet « Logiciels » ainsi que l'environnement de développement IDE de l'Arduino et le programme de la carte Sanguinololu comme illustré dans l'image.

La programmation du firmware

L'imprimante **3DRAG** est livrée avec le firmware déjà programmé sur la carte, si vous l'avez achetée en kit, la carte est déjà assemblée et testée. Nous avons choisi le firmware **Marlin** que vous pouvez télécharger sur notre site **www.3dprint.electroniquemagazine.com** à l'onglet « **Logiciel** », c'est la version actuellement utilisée dans notre laboratoire, bien évidement nous vous proposerons des versions plus récentes en téléchargement sur notre site au cours de l'évolution du firmware. La programmation de la carte s'effectue via la même connexion **USB utilisée pour piloter l'imprimante** et cette compatibilité nous est donnée par **Arduino**. **Marlin**, comme les autres firmwares, a été développé à partir d'un code écrit sous l'environnement de développement **IDE** de l'**Arduino**, il doit être mis à jour en utilisant simplement l'environnement **IDE**. Lorsque vous chargez le code source, tous les modules sont entièrement accessibles et commentés.

Au moment d'écrire ces lignes (fin 2013), pour des raisons pratiques, vous devez utiliser la version **022** de l'**IDE**, disponible sur notre site à l'adresse suivante : **www.3dprint.electroniquemagazine.com /logiciels.html**. Evitez les versions « bêta » qui n'ont pas été testées, de même que la version 023. Nous avons mis à jour le firmware **Marlin** sur notre site avec le fichier source contenant la configuration de la **3DRAG** ainsi que toute une série d'addons sur l'environnement **IDE** de l'**Arduino**.

Le deuxième onglet est appelé « **configuration.h** » et contient les divers paramètres que nous avons indiqués dans les pages précédentes. Si vous apportez des modifications, il est important de se rappeler que lorsque vous enregistrerez, vous aurez une version dans laquelle il n'y aura plus les valeurs que nous recommandons. Conservez le fichier d'origine qui est zippé ou faites une copie de l'intégralité du dossier **Marlin** et donnez-lui un nouveau nom et ouvrez le nouveau dossier **Marlin.pde**.

Pour programmer la carte **Sanguinololu** la première fois à partir de l'**IDE Arduino**, vous devez ajouter les fichiers liés au matériel.

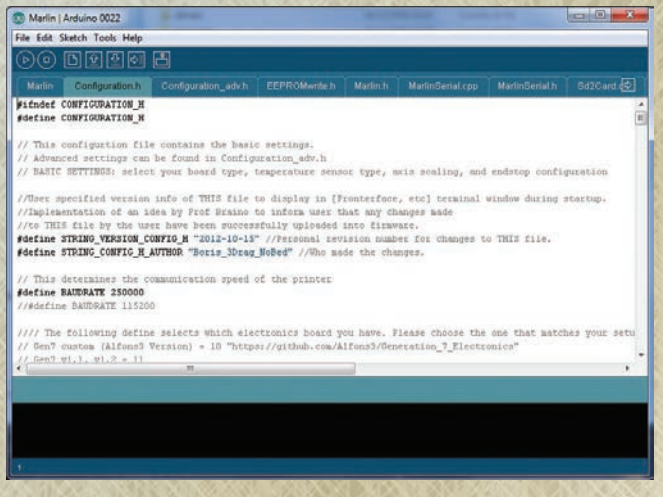


Figure 3 : L'IDE 022 d'Arduino vous permet de reprogrammer la carte avec le firmware. Elle doit être configurée pour être compatible avec les caractéristiques de la 3DRAG en modifiant le fichier « configuration.h ».

Téléchargez sur **www.3dprint.electroniquemagazine.com /logiciels.html** le fichier **Sanguino-0101r1.zip** et insérez le dans le dossier « **hardware** » de l'**IDE Arduino**. Si vous avez placé les fichiers dans le dossier approprié, comme indiqué sur la figure 4, vous trouverez dans le menu « **Tools** » d'**Arduino** une nouvelle série de cartes **Sanguinololu**, avec diverses options pour le processeur (CPU) et l'horloge (Clock) (voir la figure 5). La version actuelle montée sur la **3DRAG** est l'**ATmega 644P** et doit être en haut de la liste, si vous ne la trouvez pas, vous avez mis les fichiers du zip (Sanguino-0101r1.zip) dans un mauvais dossier.

Si vous choisissez un type de carte qui ne correspond pas à celle montée sur l'imprimante, lors de la programmation, vous recevrez un message d'erreur sans conséquence sur le matériel. Bien sûr, avant de programmer, vous devez choisir le **port COM** à utiliser du « **Port Série** », en s'assurant que rien ne dialogue avec l'imprimante (tel qu'un logiciel pour imprimante) et que le « **jumper** » (cavalier) sur la carte

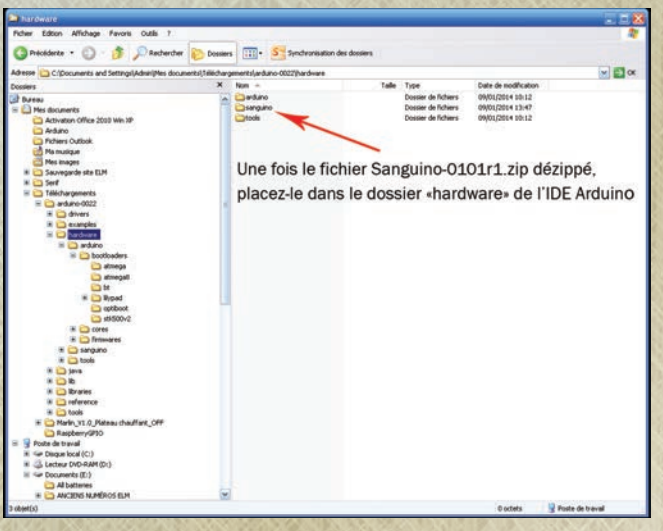


Figure 4 : Une fois le fichier Sanguino-0101r1.zip dézippé, vous devez le placer dans le dossier « hardware » de l'IDE Arduino afin que la carte Sanguinololu apparaisse dans le menu « Tools » de l'IDE.

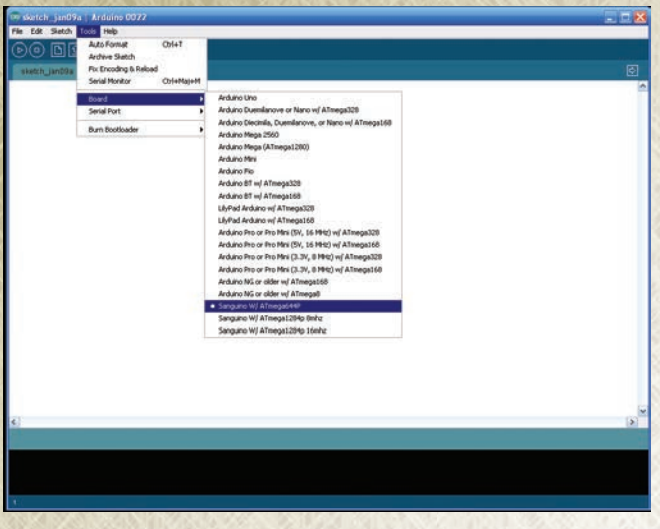


Figure 5 : le dossier « sanguino » met à disposition de nouvelles cartes dans la liste du menu « Tools ». Pour la 3DRAG sélectionnez « Sanguino W/ATmega644P ».

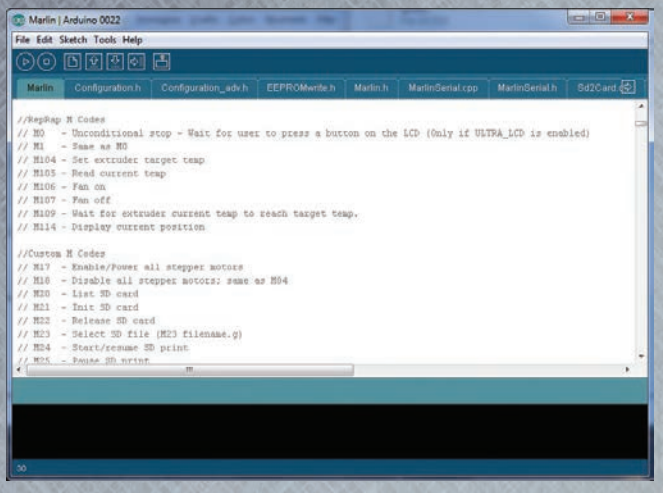


Figure 6 : Les commandes les plus récentes prises en charge par le firmware sont présentes dans la partie initiale du fichier Marlin.pde.

Sanguinololu est en position programmation (il se trouve au milieu proche du microcontrôleur du côté des drivers). Toutes les cartes que nous avons utilisées, ont été programmées avec un **bootloader** compatible **Arduino** et testées avec un **IDE Arduino 022**. Enfin, nous soulignons que dans la première partie du fichier **Marlin.pde**, on trouve la liste des commandes gérées par le firmware **G-Code** standard, en tant que commandes propriétaires avec toute la documentation nécessaire pour leur utilisation.

Définir les paramètres via le logiciel

Une des raisons pour lesquelles nous avons choisi d'adapter le firmware **Marlin** est sa capacité d'accéder, à travers une

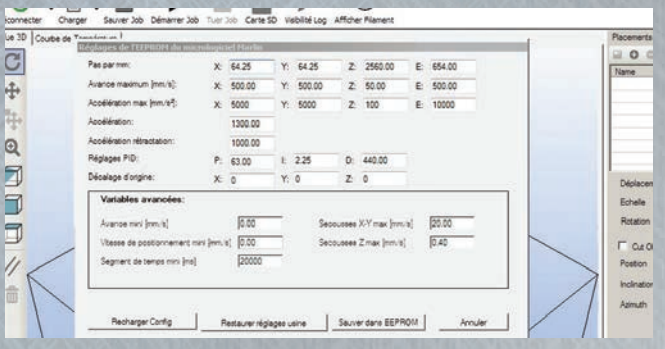


Figure 7 : Cette fenêtre apparaît en choisissant dans « Repetier Host » l'option « Configuration de l'EEPROM » dans le menu de configuration.

série de commandes qui sont spécifiques, aux différents paramètres de fonctionnement. Dans la pratique le firmware **Marlin**, non seulement gère les valeurs en les lisant dans le code « chargé », mais fait aussi une copie de « travail » dans l'**EEPROM** - et toujours à l'aide de commandes spéciales - permet la lecture, la modification, l'usage temporaire et la mémorisation permanente dans l'**EEPROM**. Le logiciel de l'imprimante « **Repetier** » dispose d'une fonction spéciale qui vous permet de gérer vos propres données dans l'**EEPROM** en interrogeant le programme **Marlin** et de gérer les commandes appropriées pour lire et écrire. Bien évidemment lors de ces opérations, l'imprimante doit être connectée.

De cette façon vous pouvez faire des changements importants juste en rallumant la **3DRAG** sans avoir à reprogrammer la carte à l'aide de l'**IDE Arduino** pour modifier les paramètres. L'interface ressemble à celle de **<REPETIER_EEPROM>** et les données présentées sont celles de notre imprimante en fonctionnement.

Voici les commandes spécifiques pour interagir avec les valeurs contenues dans la mémoire **EEPROM** et le firmware utilisé pour l'opération.

M500 - mémorise les paramètres dans l'**EEPROM**, et ceux-ci sont :
- axis_steps_per_unit, (M92)
- max_feedrate, (M203)
- max_acceleration, (M201)
- acceleration, (M204)
- retract_acceleration,
- minimumfeedrate, (M205 S T B X Y)
- mintravelfeedrate,
- minsegmenttime,
- jerk velocities,
- PID (M301)

M501 - lit les paramètres de l'**EEPROM** (par exemple, pour revenir aux réglages initiaux après avoir fait des modifications temporaires).

M502 – retour aux paramètres d'usine, ou ceux qui sont écrits dans le firmware. Doivent être stockés dans la mémoire **EEPROM** et utilisés à la place des paramètres précédemment stockés dans l'**EEPROM**.

M503 - restitue les valeurs courantes enregistrées dans la mémoire et non pas dans la mémoire **EEPROM**.

Toutes les valeurs ne correspondent pas avec celles indiquées jusqu'à présent, car nous sommes constamment en expérimentation afin de trouver les meilleurs réglages possibles pour différents emplois. De plus, les différents matériaux lors de l'extrusion peuvent exiger une vitesse et une accélération spécifique, alors que lorsqu'une impression n'est pas satisfaisante, il ne peut s'agir que d'une question de paramètres.

Par rapport à ce qui est affiché à l'écran, les paramètres qui peuvent affecter la qualité d'impression, la vitesse d'impression et de déplacement sont ceux qui se situent **de la 2eme à la 4eme ligne**. **Ne touchez pas la 1ere ligne** qui concerne le nombre de pas (micropas) que chaque moteur doit effectuer pour un déplacement de 1 mm. Le seul cas où il peut être **utile de modifier le nombre de pas est celui relatif à l'axe Z**. Nous avons choisi de gérer cet axe avec 16 micropas, alors qu'en réalité la précision n'a pas d'incidence sur la qualité d'impression.

Les drivers des moteurs

La carte **Sanguinololu** dispose, en dessous de chaque driver (**Pololu** ou **StepStick**), de **trois** « **jumpers** » (pontages, cavalier) qui déterminent la **résolution des micropas** selon la **table de vérité** représentée dans le **Tableau 1**. En observant le schéma du driver de puissance, vous pouvez remarquer que toutes les broches dédiées au pilotage de la carte sont d'un côté, tandis que celles qui vont aux enroulements des moteurs sont de l'autre côté. Cela découle d'une approche de la conception initiale dans laquelle les drivers étaient situés entre la carte mère et les bornes des moteurs.

Aujourd'hui, la situation est très différente et - avec une seule carte - l'organisation de la **Sanguinololu** est celle que vous voyez, avec toutes les connexions ordonnées grâce à une conception rationnelle de la carte de commande qui demeure inchangée au niveau de la taille et du brochage.

Les drivers fournissent le courant nécessaire aux moteurs et donc il est **impératif de doter la puce CMS du driver d'un dissipateur thermique pour éviter une surchauffe**. Dans certaines versions, la carte contient 4 couches de cuivre et non uniquement deux, la dissipation se fait à travers le circuit imprimé. Un petit ventilateur, avec un support approprié, peut rendre la situation optimale. A cet égard nous vous invitons à laisser libre cours à votre imagination pour concevoir et imprimer cette partie qui doit être montée sur votre **3DRAG**.

Le plateau d'impression

Après avoir consacré une grande attention à la configuration, il est temps de passer un peu de temps sur le plateau d'impression, qui par sa nature, est responsable du résultat final. Une bonne adhérence, la stabilité de l'imprimante et une facilité à retirer les objets sont toutes des caractéristiques liées à l'impression, comme vous le savez, la **3DRAG** a une plaque en **vetronite** qui n'est pas chauffée.

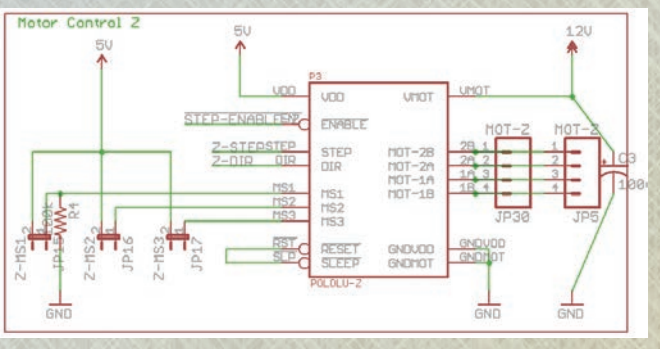


Figure 8 : Schéma de la carte Sanguinololu qui s'interface avec le module driver Pololu ou Stepstick.

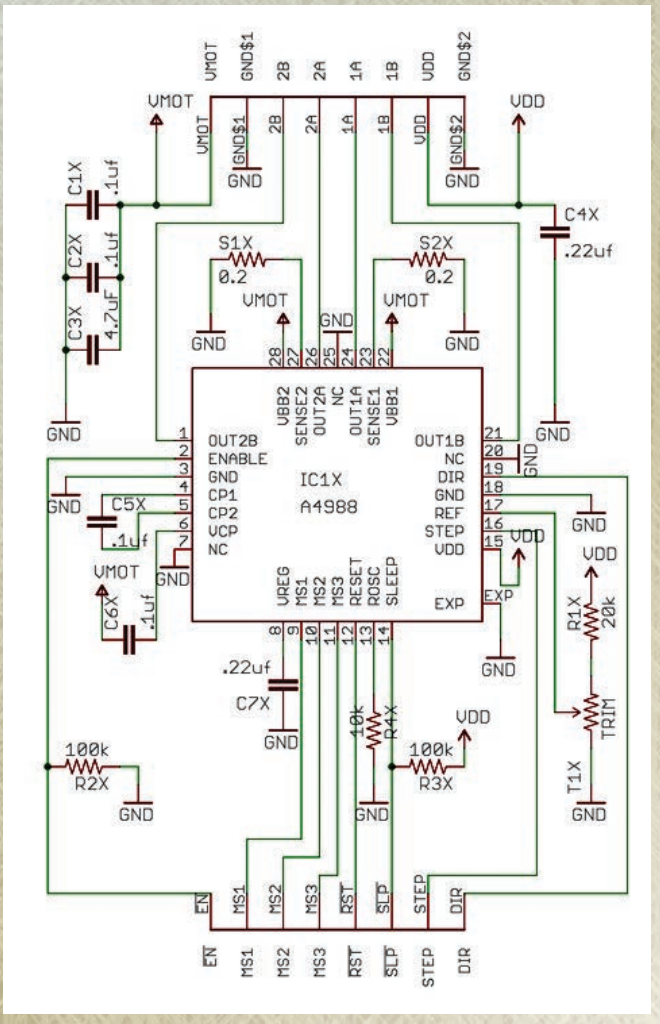


Figure 9 : Schéma de driver Stepstick basé sur une puce de type A4988.

Table 1. Microstepping Resolution Truth Table

MS1	MS2	MS3	Microstep Resolution	Excitation Mode
L	L	L	Full Step	2 Phase
H	L	L	Half Step	1-2 Phase
L	H	L	Quarter Step	W1-2 Phase
H	H	L	Eighth Step	2W1-2 Phase
H	H	H	Sixteenth Step	4W1-2 Phase

Tableau 1 : table de vérité pour le réglage des trois jumpers situés en dessous de chaque driver de puissance. Les jumpers sont classés suivant l'ordre MS1 (à gauche) à MS3 (à droite), si vous tenez le microcontrôleur vers vous.

Afin de maintenir une efficacité maximale du plateau d'impression, vous devez le traiter comme il convient en fonction du matériau que vous voulez imprimer. Si vous prévoyez d'utiliser du **PLA**, vous devez rendre la surface légèrement poreuse et avec les fibres de la **vetronite** exposées. Cela ressemble à un concept complexe, mais il suffit d'utiliser un tampon abrasif, tel que celui de la figure 10 que vous pouvez vous procurer dans les magasins de bricolage, pour transformer la surface polie en une surface opaque qui est idéale pour l'impression avec du **PLA**.

Avec une éponge de type moyen ou fin, commencez par gratter uniformément jusqu'à ce que vous voyiez la texture de

Figure 10 : Vous devez passer périodiquement l'éponge abrasive sur le plateau d'impression en vetronite afin de permettre une meilleure adhésion du PLA sur la surface.

la **vetronite** (une série de zones plus claires). Avec un chiffon antistatique, enlevez la poussière créée et à ce stade essayez d'imprimer quelque chose.

Vous remarquerez que le **PLA** fondu adhère à la **vetronite** en-dessous comme si c'était du vernis. Faites également attention à la distance du plateau et de la tête d'impression : il ne doit y avoir **aucun effort pour atteindre les quatre coins du plateau et également le centre**. A la fin de l'impression test, avec un couteau ou une lame large, soulevez l'objet imprimé à partir d'un coin. Si l'objet est mince, il sera plus facile à enlever, par contre s'il est épais ou solide, vous devez être patient et le détacher délicatement en plusieurs endroits. Une fois que l'objet commence à se décoller, essayez de pousser autant que possible en tenant le plat de la lame sur le plateau d'impression, afin de ne pas exercer trop de pression sur la lame et l'objet.

Il convient de noter que la **vetronite** a tendance à ne pas trop adhérer au **PLA**, mais avec un peu de pratique et l'utilisation de l'éponge abrasive, vous aurez une surface progressivement plus poreuse qui aura une bonne tenue pour l'impression de grands objets. Le **PLA** est beaucoup moins sujet à un rétrécissement ou un détachement du plateau que l'**ABS**. Lorsque l'objet a tendance à se décoller trop facilement, le plateau a encore besoin d'un traitement avec l'éponge abrasive, mais il se peut aussi que cela provienne d'un problème de graisse et de saleté.

Chaque fois que vous touchez le plateau, vous laissez des traces qui peuvent contenir des graisses, et rendre beaucoup plus difficile la tenue du **PLA**. Si vous remarquez des traces de doigts, utilisez un dégraissant à l'aide d'un chiffon et attendez qu'il soit complètement évaporé après le passage du chiffon avant de reprendre l'impression. **NE PAS UTILISER des produits qui laissent des résidus**, car leurs effets auront tendance à réduire l'adhésion du **PLA** sur le plateau.

Lors de notre expérience, la température de la première couche est importante. Si le **PLA** est très liquide, il a tendance à adhérer de mieux en mieux lors du remplissage et s'accroche aux microrugosités de la base, tandis que s'il est plus visqueux, il ne pénètrera pas profondément et même si la couche est bien « diffusée », il adhèrera moins.

Lutter contre l'écart avec le « BRIM »

Ces derniers mois, nous avons testé la solution du « **Raft** » (radeau), qui est une surface créée à l'aide d'un fil torsadé reposant sur le plateau d'impression de façon à ce qu'il soit léger et facile à enlever, et qui peut être utilisée afin de réduire la pression lors du refroidissement. Cependant la communauté des utilisateurs a adopté la méthode du « **BRIM** » (bord) qui permet d'accomplir les impressions les plus exigeantes et qui permet de maitriser la **rétraction thermique**, même sur des plateaux non chauffés. L'idée de cette fonction est de créer une plateforme qui suit l'ensemble du **périmètre** (contour) de l'objet à l'extérieur, avec une largeur définie par l'utilisateur.

Avec une seule couche, la rétraction thermique est contrebalancée et atténuée grâce à la grande surface de contact avec le plateau. La plateforme est construite avec une extrusion abondante de matériel juste pour s'assurer de la prise, ensuite normalement le périmètre s'aligne sur le bord extérieur. Après l'impression, le bord permet de retirer plus facilement l'objet lorsque la lame glisse sous la pièce entraînée par la fine plateforme qui, grâce à sa flexibilité, est facile à détacher. Avec un couteau, vous retirez délicatement ce bord qui, comme vous le remarquez, est bien attaché au reste de la pièce. La **largeur** (**width**) du « **BRIM** » (bord) doit être définie dans le paramétrage de l'imprimante et se trouve dans le même onglet « **skirt** » du logiciel **Slic3r**.

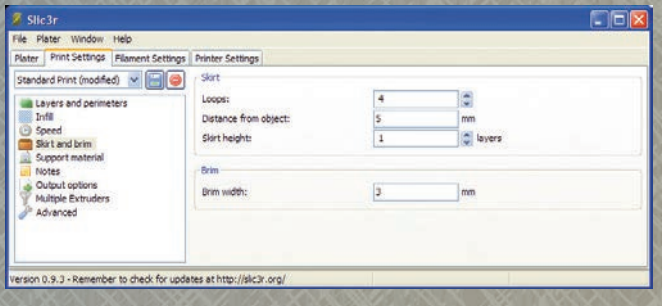


Figure 11 : Sélectionnez dans l'onglet « Print setting » (réglages de l'imprimante) à gauche l'élément « Skirt and brim », du côté droit vous voyez apparaître les paramètres, ici vous modifiez « Brim width » (largeur du bord).

Selon la taille et la densité de l'objet que vous voulez imprimer, vous pouvez donner une valeur plus ou moins grande à la largeur du bord (« **brim** »).

N'oubliez pas d'enregistrer le profil avec les nouveaux paramètres dans « **Repetier Host** », car ceux-ci sont utilisés pour découper le modèle chargé. Vous devez d'abord sélectionner un profil différent de celui que vous avez enregistré, de sorte que vous puissiez sélectionner les nouveaux paramètres. Si au contraire vous faites le découpage directement à partir de **Slic3r**, les paramètres modifiés sont utilisables immédiatement. Il est à noter, cependant, que toutes les impressions n'ont pas nécessairement besoin d'un bord, de même que tous les modèles ne nécessitent pas de support.

Dans le modèle de l'avion que vous pouvez voir sur la figure 13, l'utilisation du bord (« **brim** ») ne procure pas un avantage, mais au contraire peut compromettre le résultat final car vous devez nettoyer l'objet qui comporte de nombreuses difficultés en raison de sa structure faite de seulement quelques couches.

Le site de référence :

www. 3dprint.electroniquemagazine.com

Afin d'apporter un support à ceux qui ont commencé l'aventure ou qui sont sur le point d'acheter une imprimante **3DRAG**, mais aussi pour tous les utilisateurs de la communauté **RepRap**, nous avons créé un site internet dédié uniquement à l'imprimante **3D**.

Ce site est un espace où tous ceux qui souhaitent entreprendre la construction de notre imprimante peuvent trouver les informations nécessaires à la construction, au choix des programmes et les configurations optimales.

Dans les prochains mois, nous améliorerons et développerons le site avec notamment la **nouvelle version 1.2** de la **3DRAG**, tout en offrant un certain nombre de modèles prêts à être « imprimés » en téléchargement gratuit, et des détails relatifs à la configuration de votre imprimante.

Les modèles que vous pouvez trouver sur Internet, en fait, sont au format **STL** et doivent être « découpés » par des programmes appropriés.

En gérant les paramètres et les caractéristiques de l'imprimante, il suffit de créer un fichier contenant les instructions **G-Code** qui, couche par couche, impriment l'objet.

Sur notre site vous trouverez des fichiers **G-Code de modèles testés** et basés sur notre expérience ainsi que sur les tests d'impression effectués avec la **3DRAG**. Les fichiers **G-Code** sont prêts à être imprimés avec les meilleurs réglages possibles pour obtenir un résultat parfait.

En utilisant les fichiers **G-Code**, vous pouvez lancer immédiatement l'impression, épargner le temps de découpage, et aussi éviter d'avoir de mauvaises surprises à cause de certains paramètres qui ne sont pas définis de manière optimale.

Le site comprend également une section consacrée au firmware de l'imprimante et au logiciel pour l'impression

Figure 12 : vous pouvez voir la page d'accueil du site www.3dprint.electroniquemagazine.com, les programmes sont disponibles à l'onglet « Logiciels », les objets prêts à imprimés se trouvent à l'onglet « Impression ». Notez que le site vous propose plus de 470 photos en haute définition (en cliquant sur chaque image pour les agrandir) pour la séquence de montage.

3D, au fur et à mesure nous améliorerons les programmes en proposant des mises à jour.

NB : Attention la **3DRAG** est **vendue uniquement par la société COMELEC**, bien qu'elle utilise une mécanique commune avec la K8200 de Velleman pour des raisons de coûts de fabrication, **elle est très différente au niveau du firmware**. Dans le prochain **numéro 127 d'Electronique et Loisirs Magazine** nous vous proposerons de modifier la **3DRAG** pour **fabriquer directement des circuits imprimés sans produits chimiques à partir d'un fichier GERBER** et nous développerons l'exemple d'un circuit réalisé à l'aide d'**EAGLE CAD 6.5**, **ce qu'aucune autre imprimante 3D du marché sur ce segment de prix est capable de réaliser.**

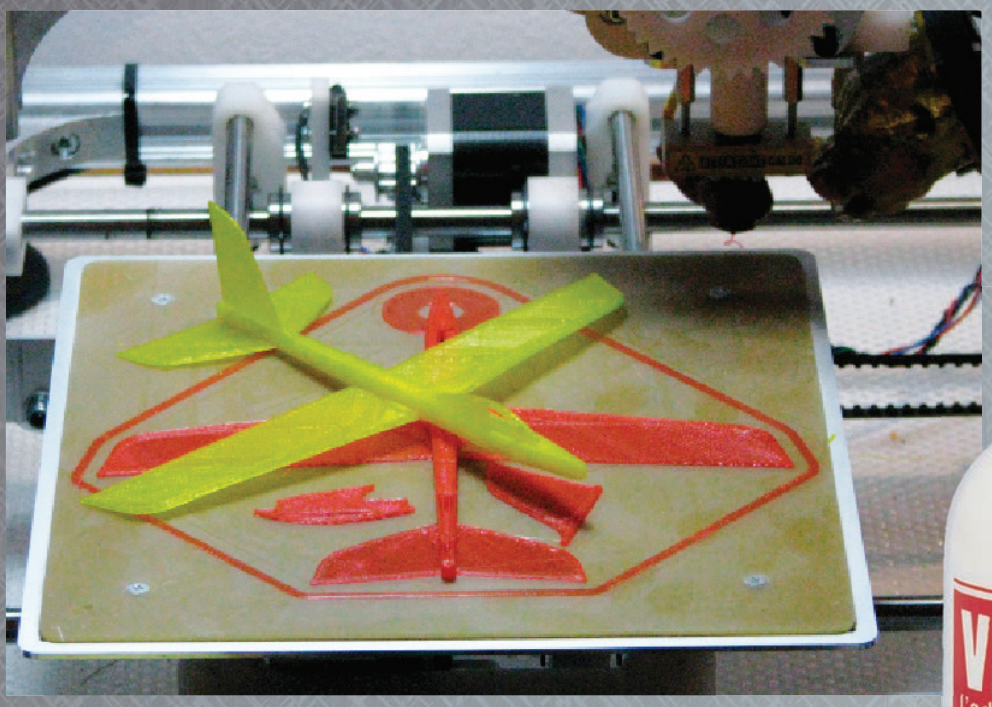


Figure 13 : Ce modèle d'avion est constitué par seulement deux couches pour les ailes, et dispose d'un corps d'une hauteur d'environ 4 mm. De par sa nature, il adhère parfaitement à la surface d'impression et ne nécessite pas l'utilisation d'un « BRIM » (bord).

Ajouter une couche d'adhésif

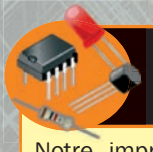
Il existe des cas pour lesquels l'impression nécessite une prise très solide sur le plateau d'impression, comme par exemple pour les vases qui ont une base très petite et un certain poids à supporter pendant une longue période avec des déplacements et des vibrations, ou des sujets qui ont un petit socle et qui doivent ensuite se développer en hauteur pour réaliser des bustes ou des figurines. Pour ceux-ci, le type de prise qu'offre la **vetronite** ne suffit pas. Il existe des solutions : par exemple, utiliser une colle spéciale, connue sous le nom de **VINAVIL** (www.vinavil.com/EN/default.asp). Cette colle qui se mélange avec l'eau à parts égales (1 dose de colle et 1 dose d'eau), doit être étalée sur la plaque d'impression. Lorsque la couche de colle est uniforme, vous devez laisser sécher. La couche de colle doit être lisse au toucher et il ne doit pas y avoir de points collants. Ce n'est que lorsque vous êtes sûr que vous pouvez imprimer. Contrairement à la **vetronite**, la couche de colle est vulnérable aux passages trop bas de la buse et peut même être arrachée par celle-ci, ce qui provoquerait une rainure. C'est pour cette raison que l'étalonnage du « 0 » de l'axe Z et l'homogénéité de la couche de colle doivent être contrôlées soigneusement.

La couche de colle peut être retirée à tout moment avec un couteau ou en lavant à l'eau chaude le plateau (débranchez l'imprimante auparavant). Gardez à l'esprit que la prise de la colle sur le **PLA** peut être très forte et dépend de la température à laquelle vous imprimez, si le **PLA** et la colle fondent entre eux, le détachement pourrait devenir très problématique. Si vous utilisez de la colle, vous pouvez généralement imprimer sans le « **BRIM** », bien que celui-ci puisse aider à guider la pièce sous la lame du couteau. Par rapport à la **vetronite**, la colle a tendance à rendre moins lisse et moins uniforme la surface inférieure imprimée de l'objet.

Figure 14 : Ici la bouteille classique rouge et blanche de la colle VINAVIL, elle vous permet d'augmenter considérablement la prise sur la plaque d'impression (www.vinavil.com/EN/default.asp).

Au fur et à mesure des impressions successives, la colle partira à certains endroits en laissant apparaître la **vetronite**. Les parties restantes de la colle garantissent une prise, mais quand il reste peu de colle, vous devez nettoyer et repasser une nouvelle couche de colle.

Dans le prochain numéro 127 d'**Electronique et Loisirs Magazine** nous vous proposerons de modifier la **3DRAG** pour **fabriquer directement des circuits imprimés sans produits chimiques à partir d'un fichier GERBER** réaliser à partir **EAGLE CAD 6.5**.



Pour le Matériel

Notre imprimante 3D est disponible auprès de notre partenaire COMELEC (voir les publicités) sous forme de kit complet comprenant tous les équipements mécaniques, électriques et électroniques (bloc d'alimentation inclus) nécessaires pour l'imprimante, à l'exclusion des fils PLA et ABS qui peuvent être commandés séparément.

COMELEC CD 908 13720 Belcodène
Tél. : 04 42 70 63 90 www.comelec.fr

SYNTHÈSE VOCALE POUR RASPBERRYPI

Après avoir transformé, dans le précédent numéro 125 d'Electronique et Loisirs Magazine, le RaspberryPi en un serveur web contrôlant à distance des entrées et des sorties, nous allons « faire parler » le RasperryPi localement à l'aide de messages vocaux, si quelqu'un envoie des commandes à distance. . . . . . d MarcdM ag g g n

Après, une longue maturation dans l'environnement exclusif des ingénieurs systèmes, des « hackers » et des professionnels de l'informatique, le système d'exploitation **LINUX** (**GNU-Linux** pour être précis) connaît une période de croissance importante parmi les habitués des systèmes d'exploitation de firmes **Microsoft** et **Apple**. Les premiers signes de ce phénomène sont apparus lorsque certains utilisateurs institutionnels, tels que les gouvernements des Etats et les administrations locales, notamment à l'étranger, mais aussi les grandes et petites entreprises, ont décidé d'utiliser le système d'exploitation **LINUX,** et plus généralement les logiciels **Open Source**, afin de développer une architecture informatique spécifique à leurs besoins. La première motivation de ce choix est dictée par la nécessité pratique de la protection de l'investissement dans les institutions.

En effet le code source du système est disponible et personnalisable, ce qui rend plus fiable le développement de systèmes informatiques spécifiques. Ces caractéristiques ne font pas partie des « **systèmes propriétaires** » et qui par conséquent lient le fabricant et l'utilisateur.

Pour bien comprendre le **concept de protection de l'investissement**, prenons le cas d'une installation industrielle comme une raffinerie, construite avec des centaines de pièces mécaniques, hydrauliques, thermiques, et de plomberie, etc. Tous les plans, dessins et documents

eSpeak

eSpeak est un logiciel de synthèse vocale qui vous permet de générer, à partir de textes multilingues, des messages vocaux sur la sortie audio de votre ordinateur, la même que celle pour la lecture d'un fichier audio. La syntaxe de base de la commande est :

espeak [options] [<texte>]

options	
-h	affiche sur l'écran la liste des options disponibles.
-f<fichier texte>	permet de spécifier le nom du fichier texte qui contient le message à synthétiser.
--stdin	lit le message à synthétiser du stdin au lieu d'un fichier, par exemple pour concaténer directement la sortie au format texte d'un programme à l'entrée du synthétiseur vocal.
-q	mode silence, il ne produit pas de son sur la sortie audio, il peut être utile avec le paramètre --**x.**
-a <entier>	niveau de sortie audio de 0 à 20, la valeur prédéfinie est 10.
-p <entier>	ajuste l'intonation de la voix de 0 à 99, la valeur par défaut est 50.
-s <entier>	vitesse de diction en nombre de mots par minute, la valeur par défaut est 160
-v <fichier vocal de référence>	le paramètre requiert l'utilisation d'un fichier vocal de configuration, par exemple, la liste des « voix » disponibles est obtenue avec la commande espeak **–voices**
-m	indique que le texte à synthétiser contient des balises de type SSML, XML ou **HTML** comme dans le cas où vous souhaitez lire des pages web automatiquement. Le synthétiseur adapte la diction appropriée en fonctions des balises rencontrées.
-w <nom du fichier wave>	écrit le résultat de la synthèse dans un fichier audio, au lieu de la prononcer directement.
-x	génère sur la sortie stdout des phonèmes mnémoniques liés au texte à synthétiser*
--stdout	génère le message sur la sortie stdout synthétisé
--punct= "<caractère"	les caractères de ponctuation sont spécifiquement prononcés par leur nom, par exemple : "virgule». Si vous ne spécifiez pas le paramètre = < caractère >, tous les caractères de ponctuation sont prononcés par leur nom.
-k <valeur>	indique la prononciation des majuscules : 1 = normal, 2 = avec le mot Majuscule. Les valeurs supérieures à 2 sont incrémentées de la valeur d'intonation (essayez -k20).
--voices[=<code de langue>]	produit une liste de tous les « codes voix » disponibles. Si le paramètre **= < code de langue >** est présent, le code de la voix pour la langue spécifiée est uniquement disponible.

*** Phonème**

En phonologie, un **phonème est la plus petite unité discrète ou distinctive** (c'est-à-dire permettant de distinguer des mots les uns des autres) **que l'on puisse isoler par segmentation dans la chaîne parlée**. Un phonème est en réalité une entité abstraite, qui peut correspondre à plusieurs sons. Il est en effet susceptible d'être prononcé de façon différente selon les locuteurs ou selon sa position et son environnement au sein du mot. **Les phones sont d'ailleurs les différentes réalisations d'un phonème**. Par exemple [ɡ̊] dans croc [kʁo], et [ʁ] dans gros [gʁo] sont deux phones différents du même phonème /ʁ/. On transcrit traditionnellement les phonèmes par des lettres placées entre des barres obliques: /a/, /t/, /ʁ/, etc., selon la règle un phonème = un symbole.

Une **mnémonique est une méthode permettant de mémoriser par association d'idées une liste de mots**. Les mnémoniques sont souvent verbales, par exemple de courts poèmes ou des mots sans signification particulière, lesquels permettent surtout de se rappeler les listes. Elles complémentent la répétition en favorisant les associations d'idées entre les concepts à l'aide de constructions faciles à mémoriser, soit parce qu'elles sont absurdes, soit parce qu'elles sont familières.

Ces techniques s'appuient sur le fait qu'une personne a plus de facilité à se souvenir de données rattachées à l'espace (exemple, un triangle), à sa personne (ex., date anniversaire) ou à toute autre information significative. Dans toutes les constructions servant de mnémonique, il faut qu'un sens se dégage, sinon l'information est difficile à mémoriser. L'hypothèse principale qui sous-tend la méthode est qu'il existe deux types de mémoires chez l'humain : la mémoire « naturelle » et l'« artificielle ». La première est innée et est utilisée sur une base quotidienne. La deuxième s'obtient par un entraînement régulier à l'apprentissage et en pratiquant différentes méthodes mnémotechniques.

sont en possession de la société, de sorte que lors d'un défaut ou lors d'un entretien périodique, des opérateurs qualifiés, qui n'appartiennent pas nécessairement à l'entreprise qui a construit l'usine, peuvent réaliser une réparation de remplacement, et si nécessaire, même reconstruire une partie de l'installation.

Supposons maintenant que l'ensemble soit géré par un processus automatisé basé sur une architecture matérielle et logicielle. Si le logiciel est de type « **propriétaire** », et que l'entreprise veut effectuer une modification du système, il faut aussi modifier les parties « **propriétaire** » du logiciel, et la seule option possible pour elle est de contacter le fabricant, en espérant qu'il soit toujours en activité et possède toujours les compétences et les connaissances pour modifier le système. Si ces conditions se produisent, notre entreprise serait confrontée à un problème qui pourrait compromettre son efficacité, la sécurité, sa compétitivité au niveau du marché, et donc, avoir des conséquences plus extrêmes sur la survie de l'entreprise.

Dans le cas de l'utilisation de systèmes **Open Source**, c'est comme dans le reste de l'installation, en possédant le code source du logiciel, les schémas, la documentation, l'entreprise peut compter sur une large communauté de professionnels indépendants pour l'entretien et les modifications du système. Il existe une loi qui impose aux autorités publiques de donner la priorité à des solutions basées sur l'**Open Source** dans leurs choix d'architectures informatiques, de systèmes et de logiciels d'applications.

En partant de ce postulat, nous allons vous faire découvrir et approfondir l'environnement d'exploitation **LINUX** et les applications disponibles qui peuvent être intégrées dans la configuration actuelle du **RaspberryPi**, ou même comme système d'exploitation pour votre PC. Si vous ne voulez pas consacrer un PC à **LINUX**, vous pouvez commencer à utiliser une distribution telle que Debian, la même que celle du **RaspberryPi**, sur une clé **USB** à partir de laquelle vous pouvez démarrer directement et utiliser **LINUX** sans toucher votre configuration d'origine.

De cette façon, vous pouvez utiliser et configurer des packages d'applications **LINUX** déjà disponibles et en installer de nouveaux. Si vous voulez passer maintenant à une expérience de **LINUX** sur votre PC, vous pouvez vous orienter vers la distribution Ubuntu, une distribution qui est appréciée dans le monde des utilisateurs **LINUX**, et qui est une alternative possible aux systèmes d'exploitation de **Microsoft** et d'**Apple**.

Du côté du **RaspberryPi** nous vous proposons une nouvelle distribution sur notre site **www. raspberrypi.electroniquemagazine.com**, avec une mise à jour importante de **Raspbian**, qui permet d'utiliser toute la puissance du **RaspberryPi**, en exploitant de manière automatique « l'**overclocking** » du processeur lorsque cela est nécessaire.

Nous verrons plus tard comment mettre à niveau cette nouvelle version. Une autre distribution intéressante est **Raspbmc**, c'est une version déclinée de **Raspbian** qui permet d'utiliser le **RaspberryPi** comme un centre multimédia (media center) afin de lire des fichiers multimédias (image, son, vidéo), de diffuser ces fichiers, et d'écouter des émissions.

Mettons à jour le RaspberryPi

Nous allons, de manière concrète, étendre les fonctionnalités du **RaspberryPi** en lui donnant la possibilité de s'exprimer. Nous adopterons la fonction « **text-to-speech** » (texte vers la parole), qui est la capacité à synthétiser une chaîne de texte en un message vocal généré par le haut-parleur connecté à la sortie audio. Nous allons utiliser cette fonctionnalité supplémentaire pour enrichir l'application qui nous permet de gérer à distance les entrées et les sorties du port **GPIO.**

Ainsi, lorsque vous serez proche de la carte, vous aurez la possibilité d'être informé par un message vocal, tout en conservant les mains libres, qu'une commande provenant d'un utilisateur distant modifie l'état d'une (ou plusieurs) broche du port **GPIO**. Avant de commencer, nous tenons à rappeler que le **RaspberryPi** est un produit en constante évolution, à la fois en termes de matériel et de kit de développement logiciel. D'un point de vue matériel, la carte dans la **version B**, celle avec tous les connecteurs externes déjà montés, est livrée avec 512 Mo de RAM au lieu de 256 Mo.

Au niveau du système d'exploitation plusieurs modules du noyau ont été optimisés pour exploiter pleinement les caractéristiques matérielles du processeur et de la mémoire, avec une nouvelle gestion de « l'**overclocking** » configurable par l'utilisateur et optimisé pour éviter des dommages à la carte.

L'**overclocking**, ou parfois appelé sur-cadencement, est une manipulation ayant pour but d'augmenter la fréquence du signal d'horloge du processeur au-delà de la fréquence nominale afin d'augmenter les performances. Le processeur overclocké peut exécuter plus d'instructions par seconde, ce qui permet de diminuer le temps d'exécution de certains programmes. En contrepartie, il chauffe plus, et peut mal réaliser certaines opérations du fait d'une trop haute température interne, ou d'une tension d'alimentation trop faible par rapport à sa fréquence, il sera donc instable.

C'est certainement plus rentable de mettre à niveau le système d'exploitation avec la dernière version sur les cartes disposant de 256 Mo de mémoire.

Pour tous ceux qui ont encore la version **Debian Squeeze**, vous devez télécharger la nouvelle version et suivre les instructions décrites dans les numéros précédents. Pour ceux qui ont déjà installé une version de Raspbian, il suffit de taper les commandes habituelles dans la fenêtre :

sudo apt-get clean

pour libérer de l'espace sur la carte SD occupé par les versions précédentes

sudo apt-get update

met à jour la liste des paquets (package) avec les versions actuelles et enfin

sudo apt-get upgrade

pour effectuer la mise à niveau réelle, ce processus prend environ dix minutes. Après la mise à niveau, nous appelerons l'outil de configuration avec la commande suivante :

raspi-config

Sélectionnez le menu « **update** » (dernière ligne en bas de la figure 1), et ensuite dans la nouvelle fenêtre de configuration (voir la figure 2) selectionnez le menu « **overclock** » qui permet de configurer l'**overclocking**. Une fois le choix « **overclocker** » sélectionné, un premier message d'alerte apparaît du type de celui représenté en figure 3. Cliquez sur « **Ok** » et vous pouvez voir les options d'**overclocking** possibles avec lequel vous pouvez expérimenter différentes configurations (voir la figure 4).

Les méthodes d'**overclocking** ont été mises en œuvre de manière à être activées uniquement lorsque les applications en cours d'exécution l'exigent, pour améliorer les performances, et même dans ce cas seulement si le **CPU** n'est pas en surchauffe ou endommagé. Après avoir mis à niveau le système d'exploitation et effectué une configuration correcte, installez le logiciel nécessaire pour reproduire les messages vocaux synthétisés à partir de chaînes de texte. Encore une fois, il s'agit de choisir les logiciels disponibles dans les répertoires de la distribution **Raspian** et de les installer avec la commande ***apt-get***.

Nous avons choisi le package **eSpeak** car il est le plus facile à installer et à intégrer dans notre application, en plus d'être moins lourd en termes de ressources, il possède une qualité de diction acceptable pour nos besoins. Le package **eSpeak** est capable de synthétiser des phrases issues de textes écrits sous forme de voix dans des langues différentes en adaptant les mots à la prononciation des différentes langues.

Si vous préférez une voix de haute qualité, vous pouvez intégrer le synthétiseur « **eSpeak** » avec le package « **mbrola** », ce qui vous permet d'étendre la gamme des voix qui peuvent être reproduites, à la fois en termes de langues tonales, ou en tant que voix féminine ou masculine.

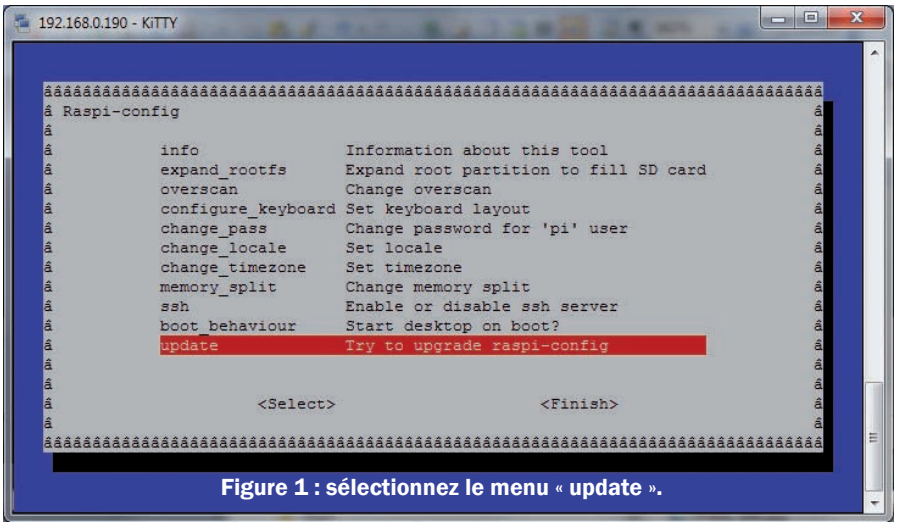


Figure 1 : sélectionnez le menu « update ».

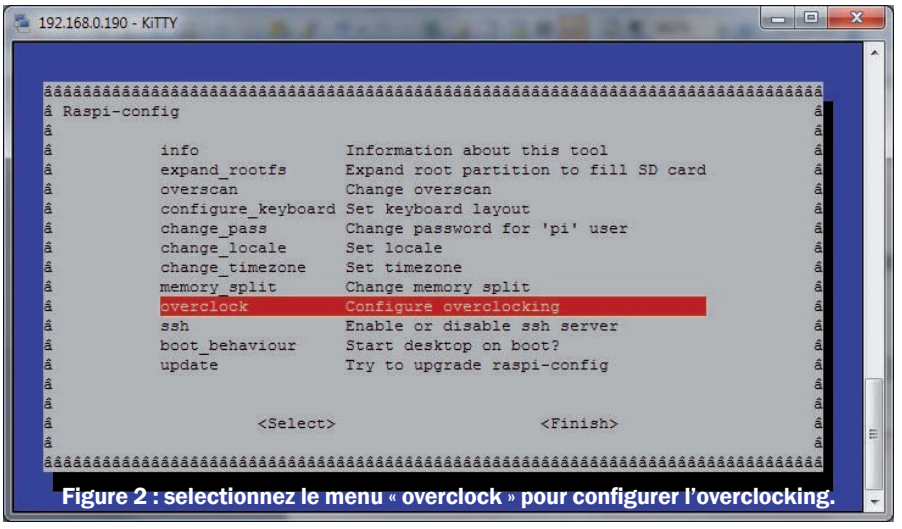


Figure 2 : selectionnez le menu « overclock » pour configurer l'overclocking.

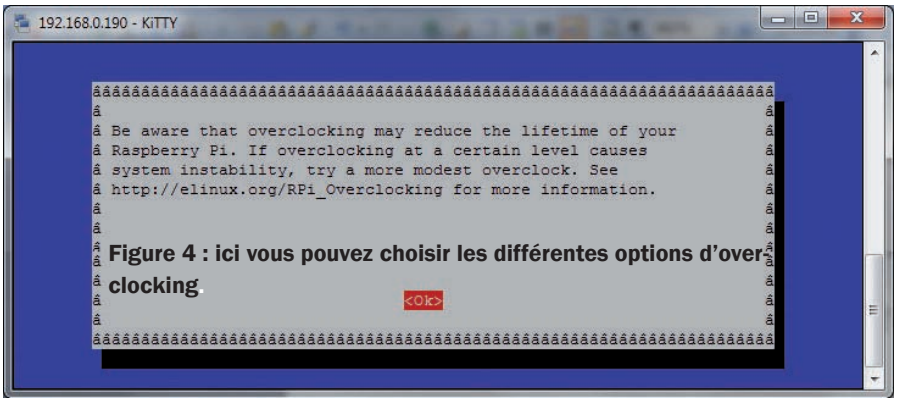


Figure 4 : ici vous pouvez choisir les différentes options d'overclocking.

Une langue à tons, ou **langue tonale**, est une langue dans laquelle **la prononciation des syllabes d'un mot est soumise à un ton** (hauteur et modulation) précis. Une modification de ce ton pouvant alors prononcer un autre mot et donc avoir un autre sens.

Par exemple il ne faut pas confondre les tons d'une langue tonale avec :

- l'accent tonique d'un mot, qui aide à distinguer les mots successifs dans le discours ;

- l'intonation prosodique qui, elle, exprime l'humeur, l'état d'esprit du locuteur, et le type d'assertion (affirmation, interrogation, exclamation) mais ne permet d'opposer des paires minimales.

En principe, les tons d'une langue tonale n'excluent pas l'intonation prosodique, dans la mesure où cette dernière s'applique davantage à une phrase, ou un ensemble de mots, plutôt qu'à une seule syllabe. Les tons sont des unités discrètes au même titre que les phonèmes mais ils ne sont pas segmentables. L'unité tonale, dite tonème, ne peut être perçue sans le support des phonèmes et n'existe pas sans eux.

Installons maintenant notre package **eSpeak** avec les commandes suivantes (voir la figure 5) :

apt-get update
apt-get install espeak

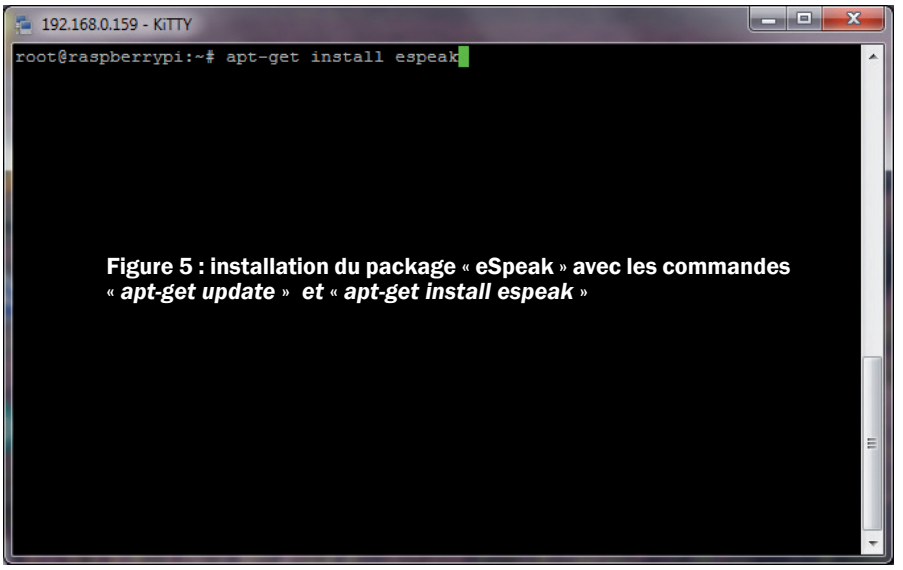


Figure 5 : installation du package « eSpeak » avec les commandes « *apt-get update* » et « *apt-get install espeak* »

Répondons par « **Y** » à la question « **Do you want to continue [Y/n] ?** », lorsque nous sommes invités à continuer après les informations sur l'espace occupé sur la carte SD avant l'installation (voir la figure 6).

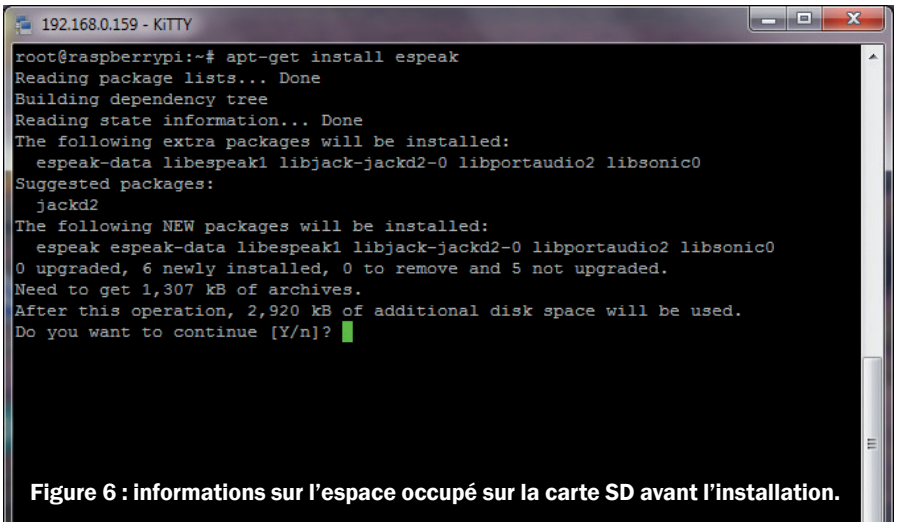


Figure 6 : informations sur l'espace occupé sur la carte SD avant l'installation.

Pour piloter la sortie audio, le synthétiseur « **eSpeak** » s'appuie sur le package « **ALSA** » (Advanced LINUX Sound Architecture) qui est le composant du noyau **LINUX** qui gère de manière automatique les pilotes pour les cartes son les plus communes, le matériel **MIDI** et le mixage entre plusieurs canaux.

Une fois l'installation terminée, assurons nous que la sortie audio que nous voulons utiliser soit configurée par « **ALSA** » pour forcer la sortie du mixage audio vers la sortie casque du **RaspberryPi**, avec la commande :

amixer cset numid=3 1

la valeur du dernier paramètre a la signification :

0 = auto, 1 = analog, 2 = hdmi

Connectons la sortie casque du **RaspberryPi** à un haut-parleur amplifié de type de celui que l'on branche sur un lecteur mp3, un smartphone ou une tablette.

Sans cela, nous ne pouvons vérifier le fonctionnement de l'ensemble, et contrôler les commandes (voir la figure 7 ci-contre) :

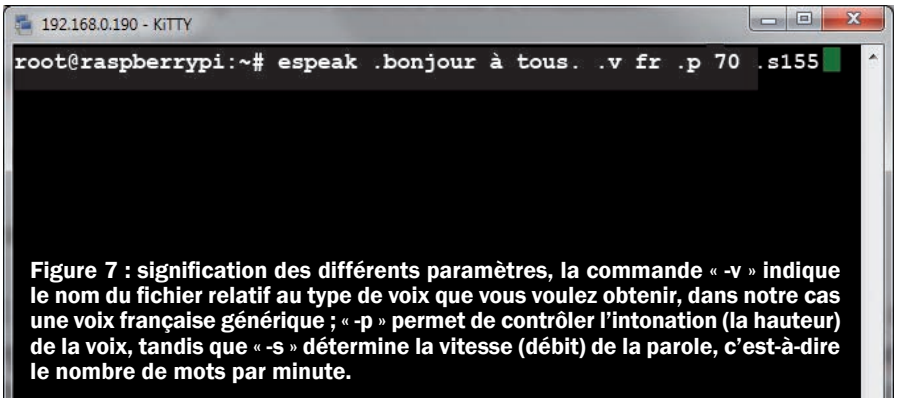


Figure 7 : signification des différents paramètres, la commande « -v » indique le nom du fichier relatif au type de voix que vous voulez obtenir, dans notre cas une voix française générique ; « -p » permet de contrôler l'intonation (la hauteur) de la voix, tandis que « -s » détermine la vitesse (débit) de la parole, c'est-à-dire le nombre de mots par minute.

espeak 'bonjour à tous' –v fr –p 70 –s 155

Dans l'encadré dédié à « **eSpeak** », nous montrons la syntaxe des commandes et la signification des différents paramètres, la commande « **-v** » indique le nom du fichier relatif au type de voix que vous voulez obtenir, dans notre cas une voix française générique. Le paramètre « **-p** » permet de contrôler l'intonation (la hauteur) de la voix, tandis que le paramètre « **-s** » détermine la vitesse (débit) de la parole, c'est-à-dire le nombre de mots par minute. Vous pouvez essayer de modifier les paramètres et de voir les effets que vous obtiendrez.

Le contrôle à distance du RaspberryPi

Nous allons maintenant essayer d'intégrer la synthèse vocale dans l'application qui gère à distance les entrées et les sorties du port **GPIO** via le Web. Le résultat que nous souhaitons obtenir est d'être prévenu par un message vocal lorsque l'état d'une sortie numérique du port **GPIO** est modifié via le web par un utilisateur distant.

Comme autre spécification, nous voulons que le message vocal ne soit émis que pour une partie seulement des sorties sans avoir à modifier le code d'un programme.

Pour ce faire, nous adaptons la base de données **GPIO** pour supporter les données nécessaires aux nouvelles fonctionnalités et les programmes pour être gérés en **PHP** et en **Python**. En ce qui concerne les modifications à apporter aux programmes, nous préférons faire une copie avant les modifications, afin de garder les deux versions. Les programmes à modifier pour la gestion de la synthèse vocale sont nommés « **control2.php** » pour la gestion des pages web et « **GPIOServer2.py** » pour la gestion physique du port **GPIO**.

Schéma de référence de l'application

Dans le diagramme de la figure 8, nous pouvons voir le fonctionnement dynamique de l'application. Comme nous l'avons décrit dans le **numéro 125 d'Electronique et Loisirs Magazine**, après authentification de l'utilisateur avec le nom d'utilisateur et mot de passe (« **admin** » et « **gpio** »), la page Web présente la liste des broches du port **GPIO** qui peuvent être gérées à distance, leur description, leur état en cours indiqué par l'image d'un interrupteur, et des boutons virtuels pour modifier l'état de chaque broche (pin) et, éventuellement, sa description.

Lorsqu'un utilisateur modifie l'état d'une broche, l'action est mise en œuvre par le programme « **control2.php** » qui change l'état de la broche dans le tableau « **pinStatus** » de la base de données **gpio** (gérée par MySQL), se référant à la page du navigateur avec la nouvelle configuration.

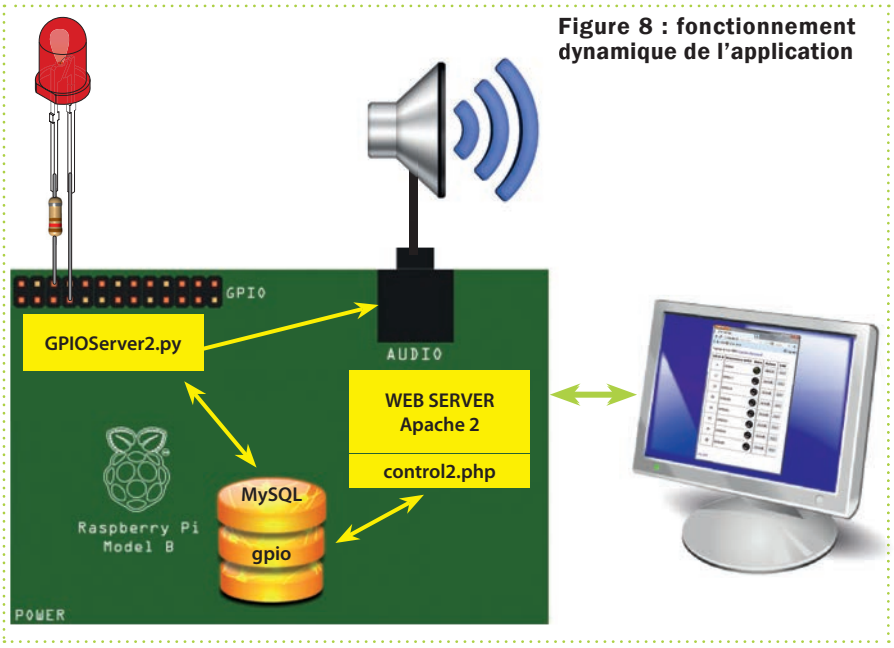


Figure 8 : fonctionnement dynamique de l'application

Le programme **Python** « **GPIOServer.py** », qui tourne en permanence dans la mémoire, interroge périodiquement l'état des broches dans la table « **pinStatus** » et envoie au driver du port **GPIO** les instructions pour configurer l'état physique de la broche avec celui correspondant dans le tableau. De cette façon, il allume et éteint physiquement la LED connectée au port **GPIO** du **RaspberryPi**.

Maintenant, nous devons modifier la logique de fonctionnement de l'application, présentée dans le précédent numéro d'Electronique et Loisirs Magazine, et introduire la fonction de message vocal associée à des changements d'état d'une broche commandée à distance.

Le programme chargé de produire le message vocal est dénommé « **GPIOServer2.py** » qui est une copie de « **GPIOServer.php** », il doit « s'apercevoir » que l'état d'une broche a été modifié et en plus du changement d'état physique du port **GPIO,** il doit aussi chercher le message spécifique à synthétiser et à transmettre à la sortie casque via le synthétiseur « **eSpeak** ».

Nous avons décidé de transmettre au programme **PHP** de gestion des pages web, la tâche de notifier à **GPIOServer** le changement d'état d'une broche, de le modifier de sorte que la variation d'un pin (broche) s'accompagne également de l'activation d'un indicateur dans la base de données, qui notifie le changement à **GPIOServer**, ainsi que l'inscription dans la table « **pinStatus** » du nouvel état de la broche.

Cette solution utilise une sorte d'indicateur de statut et est couramment utilisée dans les systèmes informatiques pour communiquer les changements d'état entre divers processus qui s'exécutent de manière asynchrone et indépendamment les uns des autres.

Lorsque « **GPIOServer2.py** » effectue le contrôle de l'état physique des broches du port **GPIO** dans la table « **pinStatus** », si il détecte un changement d'état d'une broche (valeur = "1"), il produira le message vocal et désactivera l'indicateur, de manière à produire une seule fois le message d'avertissement.

Notez que le système fonctionne également dans le cas où le **RaspberryPi** doit être redémarré alors que le processus est toujours en cours d'exécution.

Examinons d'abord les modifications à apporter à la base de données et aux programmes de gestion.

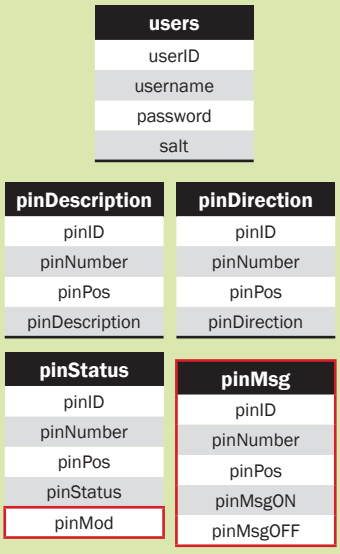


Figure 9 : diagramme de référence de la base de données gpio.	
users	contient le nom et le mot de passe (crypté) de l'utilisateur habilité pour l'application.
pinDescription	contient la description des broches (pin) du port GPIO qui apparaissent dans les pages web.
pinDirection	contient la direction (sens) actuelle de chaque broche du port GPIO (entrée (IN)/sortie(OUT)).
pinStatus	contient l'état, activé ou désactivé, pour chaque broche du port GPIO (1 pour activé et 0 pour désactivé)
pinMod	contient les messages d'avertissement qui doivent être envoyés au synthétiseur vocal, de la broche pour laquelle vous souhaitez annoncer le changement d'état.

Figure 9 : structure des tables de la base de données gpio à gauche , ci-dessus le diagramme de référence de la base de données gpio.

La base de données gpio

Commençons par analyser la structure de la base de données **gpio**, dont nous avons décrit dans le numéro 125 d'Electronique et Loisirs Magazine les opérations d'installation et de configuration. La base de données **gpio** est constituée de tables qui contiennent des relations entre les données nécessaires à la gestion du système, structurées comme représenté sur le diagramme de la figure 9.

La table est définie comme une relation entre une entité et ses attributs (informations) qui la caractérisent.

A titre d'exemple, nous pouvons imaginer une feuille de calcul brut, composée de plusieurs lignes où certaines colonnes sont utilisées comme champs de recherche pour sélectionner une ligne spécifique, et l'autre contiendra les informations descriptives. Un autre exemple, nous pouvons imager une entité « Client » et les attributs sont : « Numéro du Client » ; « Nom du Client » ; « Adresse du Client ».

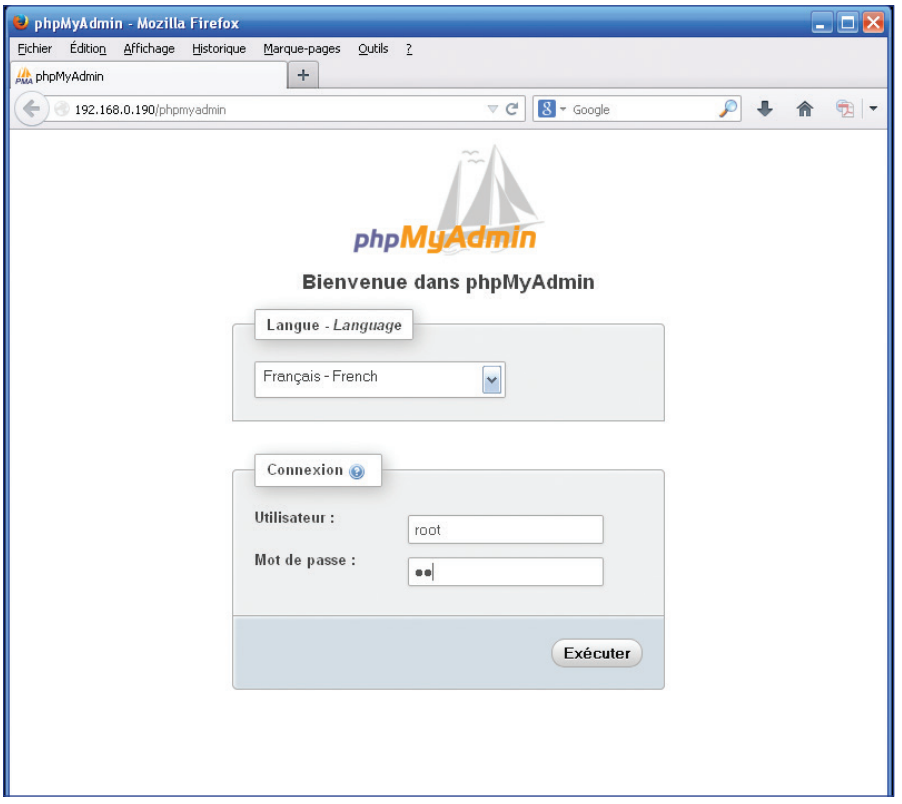


Figure 10 : entrons dans phpMyAdmin en tapant l'adresse IP du RaspberryPi http://192.168.0.190/phpmyadmin et ensuite tapons le nom d'utilisateur et le mot de passe et cliquons sur « Exécuter ».

Chaque table est caractérisée par une clé primaire, qui dans notre cas est un identifiant numérique. Certains attributs sont particulièrement importants dans la recherche d'un seul élément dans la table, et sont définis comme étant uniques (ne peuvent pas occuper deux lignes de la table avec la même valeur) et non-nuls (qui doivent contenir des valeurs significatives).

Ces attributs sont construits à l'aide d'indices pour accélérer la recherche et mettre à jour les lignes de la table par des programmes ou des applications de gestion de base de données. En fait, ils constituent des clés secondaires de recherche.

Pour bien comprendre le résonnement voici un exemple : une clé sert à identifier de manière unique un enregistrement. On parle alors de **Clé primaire**. Le numéro de sécurité sociale

est un bon exemple de clé primaire car il identifie de manière unique un individu. Même si 2 individus ont le même nom et le même prénom, ils ne peuvent avoir le même numéro de sécurité social. Prenons les cas des albums et des auteurs.

D'un côté nous avons une liste d'albums sans auteurs et de l'autre une liste d'auteurs sans album. Il faudrait pouvoir associer un auteur à un album. Ajoutons une clé primaire dans la table « auteurs » et dans la table « albums » que l'on nommera « N° Auteur » et « N° Album ». L'insertion d'une clé primaire permet en plus d'éviter la redondance des informations. Dans ce cas, on n'a qu'une fois le nom et prénom de l'auteur pour tous ces albums.

Mais nous ne savons pas encore quel auteur a fait quel album. Il faut rajouter dans la table « Album » une clé secondaire qui est la copie de la clé primaire créée dans la table « Auteur ». Grâce à cette clé secondaire, nous savons quel auteur a fait quel album, nous avons donc les relations entre les tables grâce à la clé primaire et à la clé secondaire. Nous venons de créer une relation entre les deux tables, et cela va permettre de réunir les informations réparties entre les deux tables.

Il existe 3 types de relations :

- La relation de type « un à plusieurs » est un enregistrement d'une table qui est en relation avec plusieurs enregistrement d'une autre table. C'est le cas pour l'exemple précédent, un auteur a fait plusieurs albums.

- La relation de type « plusieurs à plusieurs », plusieurs enregistrements d'une table correspondent à plusieurs enregistrements d'une autre table.

Dans le cas précédent, on aurait pu avoir plusieurs albums pour plusieurs auteurs. (les reprises de chanson).

ATTENTION : Dans ce cas seulement, il faut créer une autre table dont la clé primaire est la concaténation (le rassemblement) des 2 clés primaires des 2 tables.

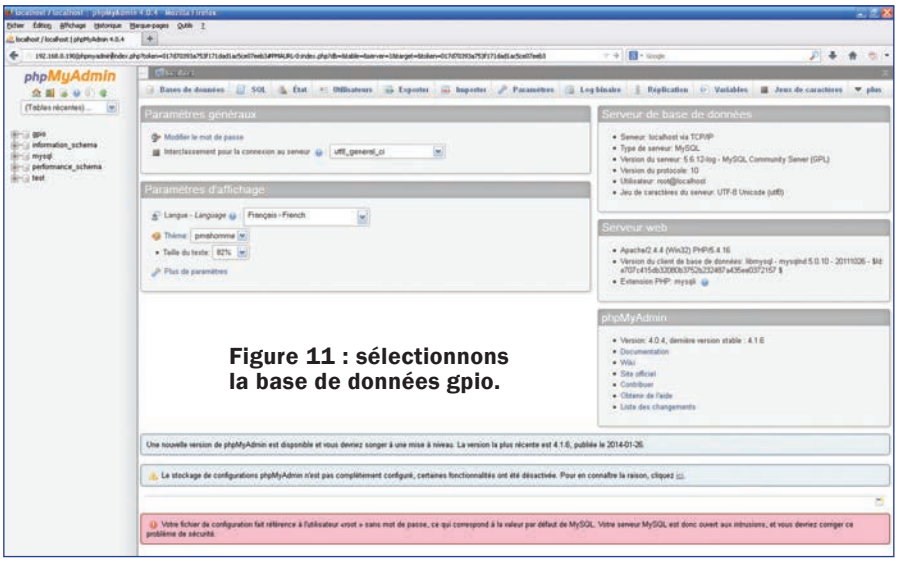


Figure 11 : sélectionnons la base de données gpio.

Figure 12 : ici vous voyez les tables présentes dans la base de données gpio.

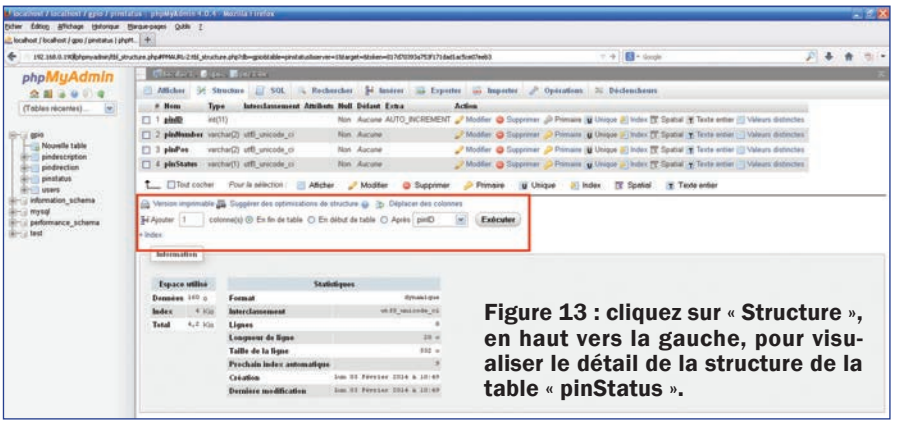


Figure 13 : cliquez sur « Structure », en haut vers la gauche, pour visualiser le détail de la structure de la table « pinStatus ».

- La relation de type « un à un » est un enregistrement d'une table, et un seul, qui est en relation avec un enregistrement, et un seul, d'une autre table. Ce type de relation n'est jamais utilisé car on pourrait ajouter les enregistrements des 2 tables dans une même table, ce qui viendrait à faire une autre table.

Pour la table « users », qui contient la liste des utilisateurs autorisés à utiliser l'application et le mot de passe relatif, la clé de recherche est l'attribut (champ) « username », alors que pour toutes les autres tables, les clés de recherche sont les attributs « **pinNumber** » et « **pinPos** ».

Cette double approche est due à la différence de mode pour reconnaître chaque broche **GPIO** entre les langages **PHP** et **Python**. Le langage **PHP**, pour adresser une broche du port **GPIO**, nécessite l'identificateur du port **GPIO**.

Par exemple, dans le cas de notre LED 4 (GPIO4), le langage **Python** adresse directement les broches (pin) en fonction de leur position physique sur le connecteur (broche 7 dans notre cas).

Par contre le langage **PHP**, qui gère les pages WEB de l'application, utilise un champ de recherche dans les tables dont l'attribut est « **pinNumber** », alors que le langage **Python** qui gère physiquement le port **GPIO** utilise l'attribut « **pinPos** ».

Nous voyons également sur la figure 9 la structure des tables de la base de données **gpio**, avec une description du contenu de chacune, les modifications qui doivent être apportées sont mises en évidence par les rectangles rouges.

Mise à jour de la base de données

En ce référent à la figure 9, nous devons intégrer dans la structure de la base de données la nouvelle configuration qui inclut le nouveau champ et la nouvelle table, mis en évidence par les rectangles rouges.

Dans la pratique, cela consiste à ajouter à la table « **pinStatus** » un nouveau champ nommé « **pinMod** », qui sera utilisé comme un indicateur des variations de l'état de la broche, de la manière décrite précédemment, et une nouvelle table nommé « **pinMsg** » qui va contenir le texte des messages vocaux à émettre en cas de changement d'état d'une des broches, en différenciant l'état de la broche en fonction de l'état « allumé » ou « éteint ».

Nous réalisons la mise à jour de la base de données **gpio**, en utilisant l'application de gestion de base de données « **phpMyAdmin** ». Entrons dans **phpMyAdmin** en tapant l'adresse IP du **RaspberryPi** suivie de « /phpmyadmin » soit **http://192.168.0.190/phpmyadmin**.

Ensuite tapons le nom d'utilisateur et le mot de passe et cliquons sur « Exécuter » comme le montre la figure 10. Nous obtenons la page web de la figure 11, sur laquelle nous sélectionnons la base de données **gpio** en effectuant un double clic. Nous visualisons la page web de la figure 12, qui répertorie les tables présentes dans la base de données.

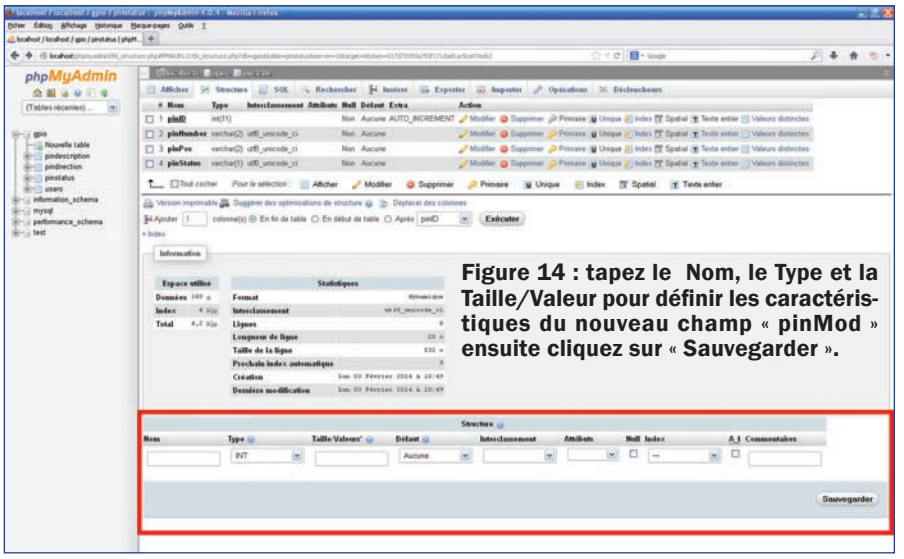


Figure 14 : tapez le Nom, le Type et la Taille/Valeur pour définir les caractéristiques du nouveau champ « pinMod » ensuite cliquez sur « Sauvegarder ».

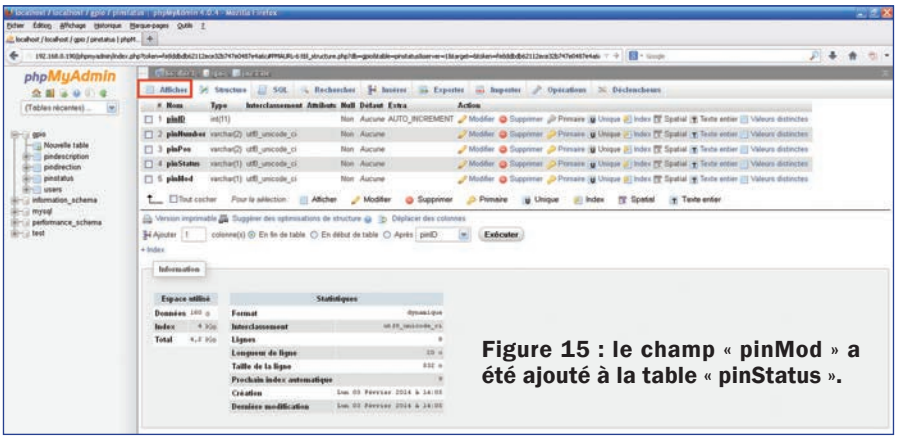


Figure 15 : le champ « pinMod » a été ajouté à la table « pinStatus ».

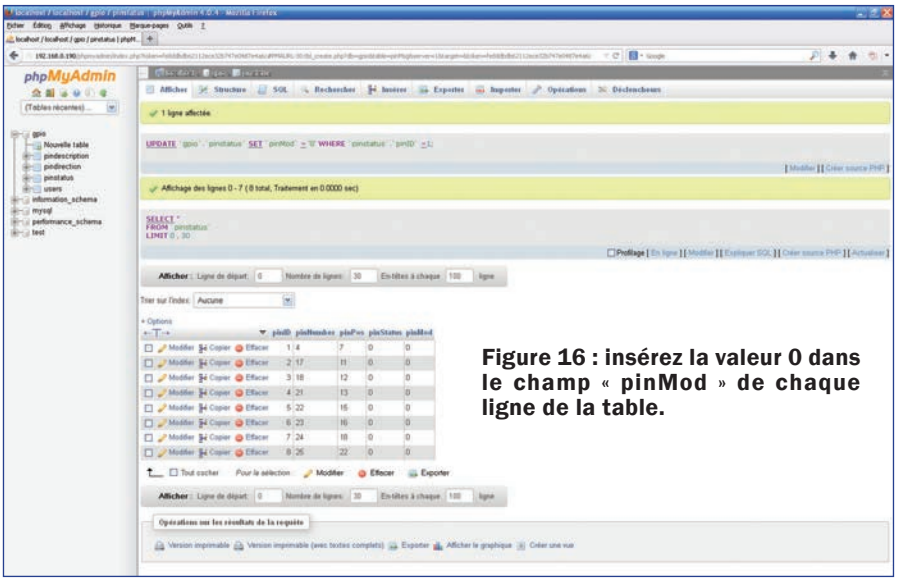


Figure 16 : insérez la valeur 0 dans le champ « pinMod » de chaque ligne de la table.

Cliquons sur la table « **pinStatus** », une nouvelle page s'ouvre, cliquez sur « Structure » (en haut vers la gauche), nous voyons le détail de la structure de la table (figure 13). Nous devons insérer le nouveau champ « **pinMod** » dans la table, puis compléter les champs. Vérifiez comme indiqué dans le rectangle rouge que vous avez une

colonne en fin de table et cliquez sur « Exécuter », ensuite vous obtenez la figure 14, vous tapez alors :

Nom : pinMod
Type : VARCHAR
Taille/Valeur : 1

pour définir les caractéristiques du nouveau champ « **pinMod** » qui sera utilisé comme un indicateur de la variation de l'état de la broche correspondante, et enfin vous cliquez sur « Sauvegarder » (en haut vers la droite).

Normalement vous devez obtenir le résultat de la figure 15, qui confirme que le champ « **pinMod** » a été ajouté à la table « **pinStatus** ». Maintenant, cliquez sur l'onglet « Afficher » (en haut à gauche) pour obtenir le tableau qui contient les 5 champs de la table « **pinStatus** » (figure 16) et dans lequel vous pouvez modifier le contenu des champs de la table. Puisque nous avons spécifié lors de la création du champ une valeur « **NON NULL** », insérez la valeur 0 dans le champ « **pinMod** » de chaque ligne de la table, nous obtenons le résultat illustré sur la figure 16.

Créons maintenant une table qui contient les messages vocaux de changement d'état des broches. Sur le côté gauche de la page, cliquez sur la base de données « **gpio** », dans le champ « Nouvelle table » tapez « **pinMsg** » et au « Nombre de colonnes » tapez 5 (voir la figure 17a) et cliquez sur « Exécuter ». Dans la nouvelle page (voir la figure 17), vous voyez apparaitre en haut le nom de la table « **pinMsg** » et une série de 5 colonnes contenant des champs à remplir de la manière correspondante au Tableau 1. Vous devez obtenir le résultat visible sur la figure 18a, cliquez ensuite sur « Sauvegarder » en bas vers la droite. La nouvelle table a été créée avec les 5 champs comme indiqué sur la figure 18 ci-contre.

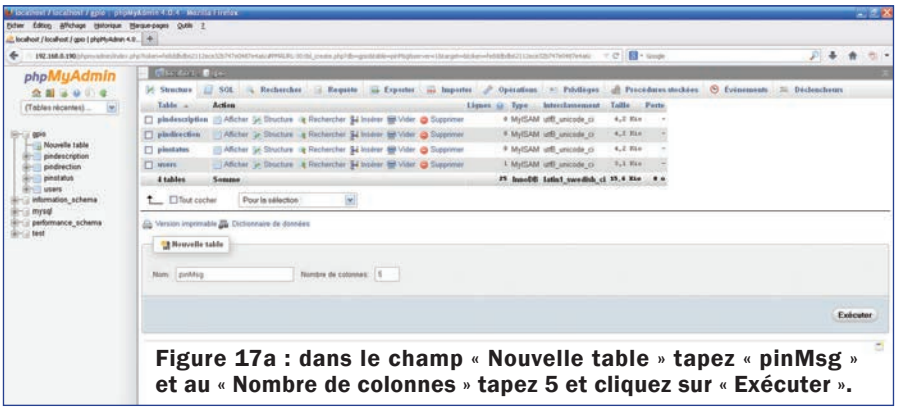


Figure 17a : dans le champ « Nouvelle table » tapez « pinMsg » et au « Nombre de colonnes » tapez 5 et cliquez sur « Exécuter ».

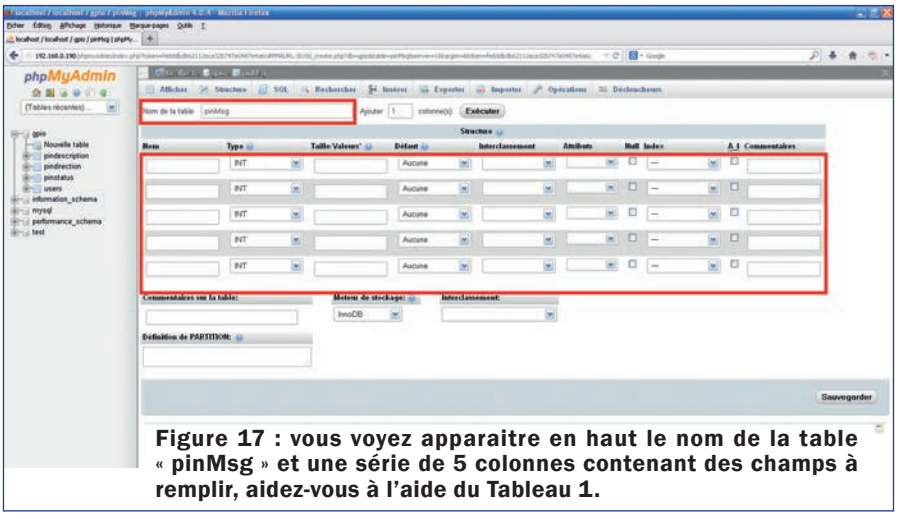


Figure 17 : vous voyez apparaitre en haut le nom de la table « pinMsg » et une série de 5 colonnes contenant des champs à remplir, aidez-vous à l'aide du Tableau 1.

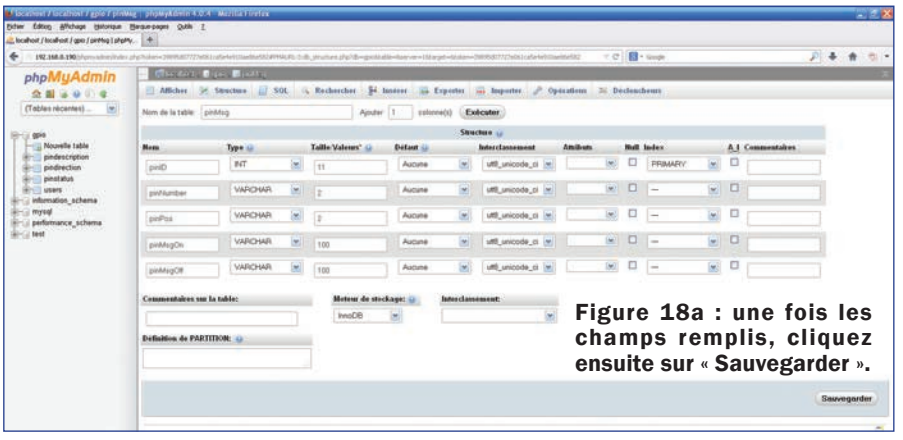


Figure 18a : une fois les champs remplis, cliquez ensuite sur « Sauvegarder ».

Cliquez sur « Structure » (rectangle rouge) de la table « **pinMsg** » pour accéder à la page web de la figure 19. Nous allons ajouter des index aux champs, ce qui nous aidera à accélérer les recherches dans la table.

Nous ajoutons un index sur le champ « **pinNumber** » et un autre sur « **pinPos** ». Cochez les cases « **pinNumber** » et « **pinPos** » dans la structure de la table et cliquez sur « index » en dessous de la ligne « **pinMsgOff** » vers

Nom	Type	Taille/Valeurs	Défaut	Interclassement	Attributs	Null	Index
pinID	INT	11	Aucune	utf8_unicode_ci		PRIMARY	
pinNumber	VARCHAR	2	Aucune	utf8_unicode_ci			
pinPos	VARCHAR	2	Aucune	utf8_unicode_ci			
pinMsgOn	VARCHAR	100	Aucune	utf8_unicode_ci			
pinMsgOff	VARCHAR	100	Aucune	utf8_unicode_ci			

la droite comme indiqué sur la figure 19. Vous devez obtenir le résultat de la figure 20 visible dans le rectangle rouge.

Maintenant que vous avez créé la table pour contenir le texte des messages vocaux qui doivent signaler les changements d'états des broches du port **GPIO**, il est temps de les remplir.

Cliquez sur l'onglet « Insérer » dans le coin supérieur droit de la page web et vous voyez apparaître la figure 21.

Remplissez les champs du haut de la première colonne avec les valeurs suivantes : **pinNumber** =4, **pinPos**=7, **pinMsgON** = LED rouge allumée et **pinMsgOff** = LED rouge éteinte. Ces champs contiennent le texte des messages qui seront synthétisés lors d'un événement. Bien sûr, vous pouvez entrer ce que vous voulez.

Cliquez sur le bouton « Exécuter » pour confirmer les entrées dans la table « **pinMsg** », s'il n'y a pas d'erreur vous devez normalement obtenir la page web de la figure 22. À cet égard, parmi les caractéristiques du projet, nous devons rendre obligatoire la diffusion d'un message lorsqu'une des broches est modifiée à distance, et donc il faut insérer une ligne dans la table « **pinMsg** » pour chaque broche, ou laisser l'utilisateur décider s'il faut générer un message spécifique ou non lors d'un changement d'état d'une broche.

Nous avons opté pour la deuxième option, c'est à dire insérer seulement les lignes relatives aux broches pour lesquelles nous aimerions avoir un message, en ignorant les autres. Il sera alors nécessaire de tenir compte de la présence ou non d'un message en adaptant le comportement du logiciel de gestion.

Cliquez sur l'onglet « Afficher » (en haut à gauche) sur la page et vous obtenez l'affichage des lignes (ici une seule) de la table « **pinMsg** » (voir la figure 23). Pour le moment nous avons terminé les modifications à apporter à la base de données et nous allons analyser et modifier les programmes de gestion des pages web et de l'état physique des broches du port **GPIO**.

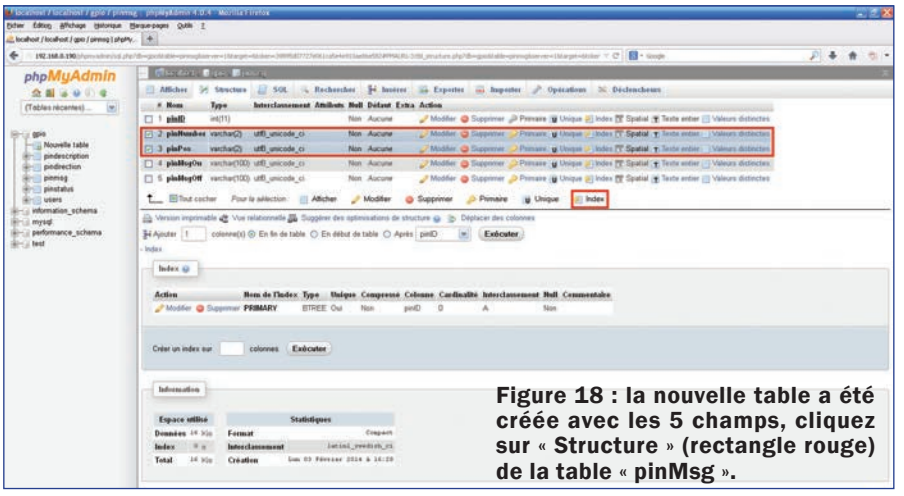


Figure 18 : la nouvelle table a été créée avec les 5 champs, cliquez sur « Structure » (rectangle rouge) de la table « pinMsg ».

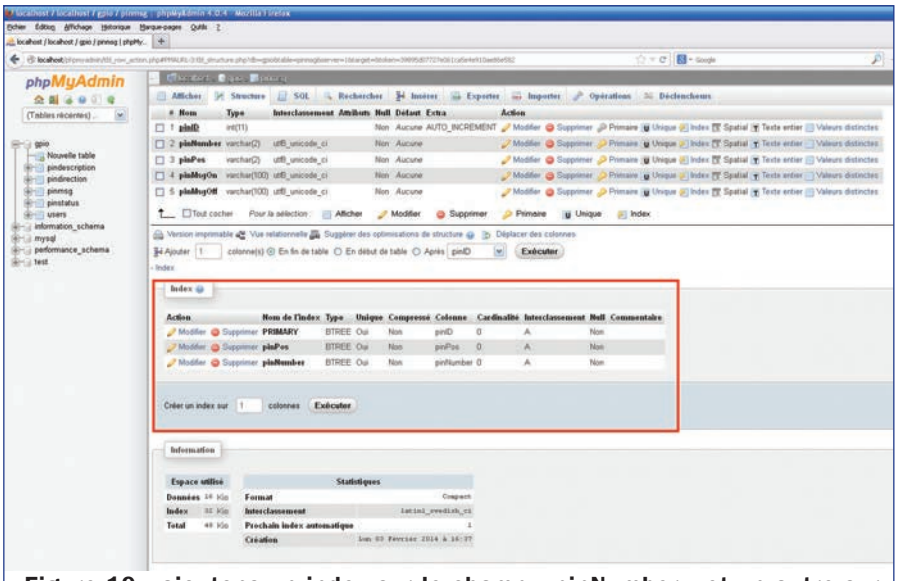


Figure 19 : ajoutons un index sur le champ « pinNumber » et un autre sur « pinPos » en cochant les cases « pinNumber » et « pinPos » dans la structure de la table et cliquez sur « index » en dessous de la ligne « pinMsgOff ».

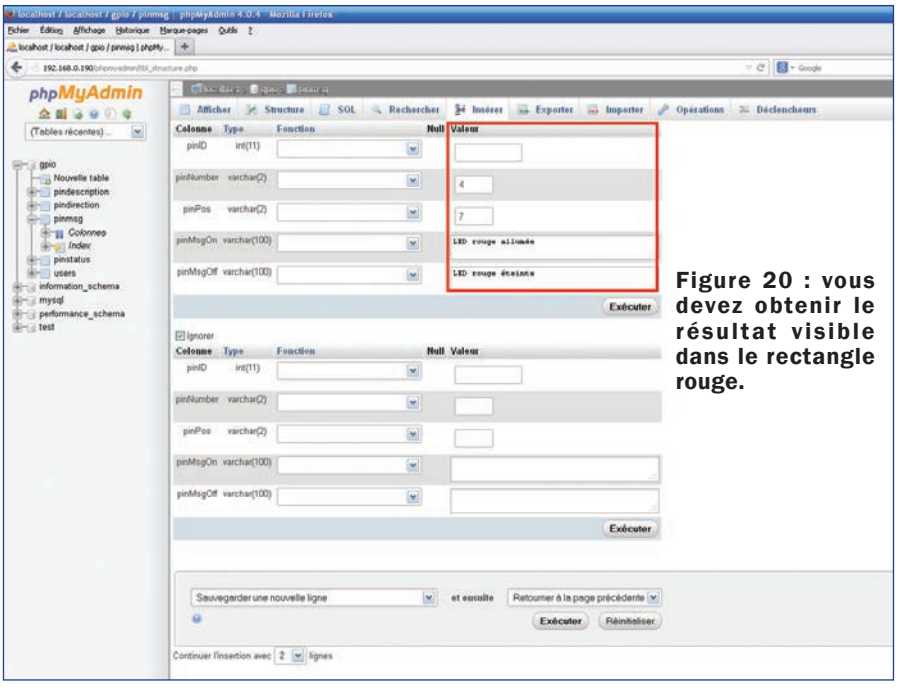


Figure 20 : vous devez obtenir le résultat visible dans le rectangle rouge.

Programme de gestion des pages Web

Comme nous l'avons décrit dans le numéro 125 d'Electronique et Loisirs Magazine, pour la gestion des pages web de l'application nous utilisons le serveur web Apache 2, qui est universellement adopté pour cette fonction. Pour la gestion des pages web proprement dites, qui dans notre cas sont dynamiques et générées de nouveau à chaque requête de l'utilisateur, nous avons choisi le langage **PHP**, largement utilisé pour ce type d'utilisation, facile à apprendre et avec une valeur pédagogique considérable.

Historique de PHP

Le langage **PHP** fut créé en 1994 par Rasmus Lerdorf pour son site web. C'était à l'origine une bibliothèque logicielle en Perl dont il se servait pour conserver une trace des visiteurs qui venaient consulter son CV. Au fur et à mesure qu'il ajoutait de nouvelles fonctionnalités, Rasmus a transformé la bibliothèque en une implémentation en langage C, capable de communiquer avec des bases de données et de créer des applications dynamiques et simples pour le Web.

Rasmus décida alors en 1995 de publier son code, pour que tout le monde puisse l'utiliser et en profiter. **PHP** s'appelait alors PHP/FI pour Personal Home Page Tools/Form Interpreter. En 1997, deux étudiants, Andi Gutmans et Zeev Suraski, redéveloppèrent le cœur de PHP/FI. Ce travail aboutit un an plus tard à la version 3 de **PHP**, devenu alors **PHP Hypertext Preprocessor**. Peu de temps après, **Andi Gutmans** et **Zeev Suraski** commencèrent la réécriture du moteur interne de PHP afin de développer de nouvelles versions. En 2002, **PHP** est utilisé par plusieurs millions de sites Web à travers le monde et en 2013 par plus de 244 millions.

La version actuelle est la version 5, sortie le 13 juillet 2004. Elle utilise Zend Engine 2 et introduit une modélisation objet plus performante, une gestion des erreurs fondée sur le modèle des exceptions, ainsi que des fonctionnalités de gestion pour les entreprises.

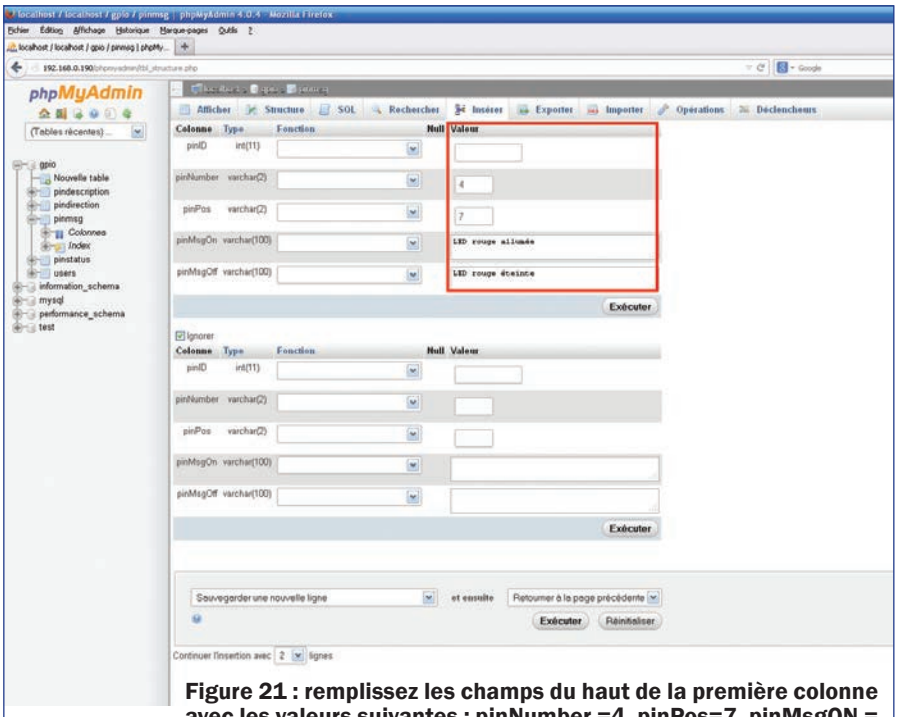


Figure 21 : remplissez les champs du haut de la première colonne avec les valeurs suivantes : pinNumber =4, pinPos=7, pinMsgON = LED rouge allumée et pinMsgOFF = LED rouge éteinte.

PHP 5 apporte beaucoup de nouveautés, telles que le support de SQLite ainsi que des moyens de manipuler des fichiers et des structures XML. A ce jour (13/01/2014), la dernière mise à jour est la 5.5.8

Présentation de PHP

PHP est un langage de script utilisé le plus souvent côté serveur : dans cette architecture, le serveur interprète le code **PHP** des pages web demandées et génère du code (HTML, XHTML, CSS par exemple) et des données (JPEG, GIF, PNG par exemple) pouvant être interprétées et rendues par un navigateur.

PHP peut également générer d'autres formats comme le WML, le SVG, le PDF. Il a été conçu pour permettre la création d'applications dynamiques, le plus souvent développées pour le Web. **PHP** est le plus souvent couplé à un serveur Apache bien qu'il puisse être installé sur la plupart des serveurs HTTP tel que IIS de Microsoft. Ce couplage permet de récupérer des informations issues d'une base de données, d'un système de fichiers (contenu de fichiers et de l'arborescence) ou plus simplement des données envoyées par le navigateur afin d'être interprétées ou stockées pour une utilisation ultérieure.

L'accès aux bases de données est simplifié une fois l'installation des modules correspondant effectuée sur le serveur. La force de ce langage est qu'il a permis au fil du temps la résolution de problèmes autrefois compliqués et est devenu par conséquent un composant incontournable des offres d'hébergements.

Il est multiplateformes : autant sur **LINUX** qu'avec Windows il permet aisément de réutiliser le même code sur un environnement à peu près semblable (prendre en compte les règles d'arborescences de répertoires qui peuvent changer). Libre, gratuit, simple d'utilisation et d'installation, ce langage nécessite comme tout langage de programmation une bonne compréhension des principales fonctions usuelles ainsi qu'une connaissance aiguë des problèmes de sécurité liés à ce langage.

Fonctionnement de PHP

PHP appartient à la grande famille des descendants du langage C, dont la syntaxe est très proche. En particulier, sa syntaxe et sa construction ressemblent à celles des langages Java et Perl, à la différence que du code **PHP** peut facilement être mélangé avec du code **HTML** au sein d'un fichier **PHP**.

Dans une utilisation Web, l'exécution du code **PHP** se déroule ainsi : lorsqu'un visiteur demande à consulter une page Web, son navigateur envoie une requête au serveur HTTP correspondant. Si la page est identifiée comme un script **PHP** (généralement grâce à l'extension .php), le serveur appelle l'interprète **PHP** qui va traiter et générer le code final de la page (constitué généralement d'**HTML** ou de XHTML, mais aussi souvent de CSS et de JS). Ce contenu est renvoyé au serveur **HTTP**, qui l'envoie finalement au client (voir la figure A).

Une étape supplémentaire est souvent ajoutée, celle du dialogue entre **PHP** et la base de données. Classiquement, **PHP** ouvre une connexion au serveur de SGBD (système de gestion de base de données) voulu, lui transmet des requêtes et en récupère le résultat, avant de fermer la connexion.

L'utilisation de **PHP** en tant que générateur de pages Web dynamiques est la plus répandue, mais il peut aussi être utilisé comme langage de programmation ou de script en ligne de commande sans utiliser de serveur **HTTP** ni de navigateur. Il permet alors d'utiliser de nombreuses fonctions du langage C et plusieurs autres sans nécessiter de compilation à chaque changement du code source. Pour réaliser en **Linux/UNIX** un script **PHP** exécutable en ligne de commande, il suffit comme en **Perl** ou en **Bash** d'insérer dans le code en première ligne le **shebang** : « **#! /usr/bin/php** ».

Sous un éditeur de développement comme SciTE, même sous Windows, une première ligne « **<?php** » suffit, si le fichier possède un type « **.php** ». **PHP** possède un grand nombre de fonctions permettant des opérations sur le système de fichiers, la gestion des bases de données, des fonctions de tri et hachage, le traitement de chaînes de caractères, la génération et la modification d'images, des algorithmes de compression.

Comment installer PHP et phpMyAdmin sur votre ordinateur ?

Il existe plusieurs types d'installations sont possibles, vous pouvez installer

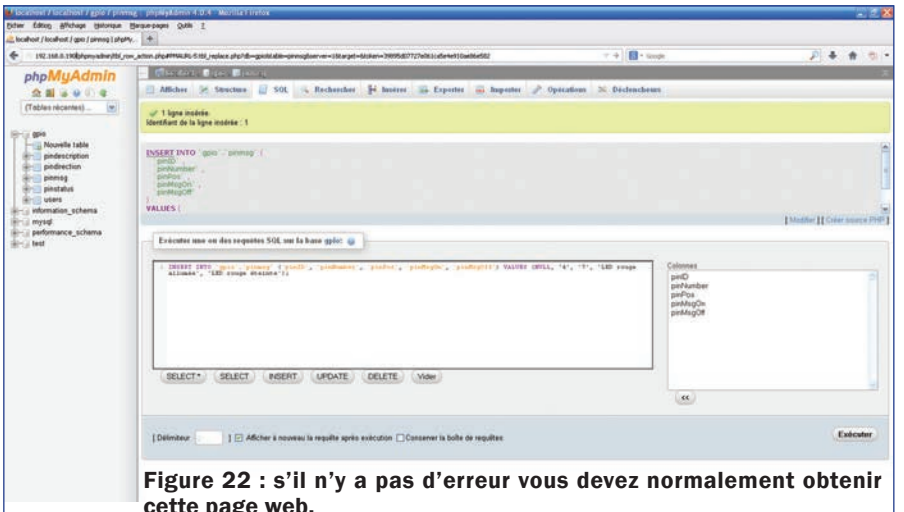


Figure 22 : s'il n'y a pas d'erreur vous devez normalement obtenir cette page web.

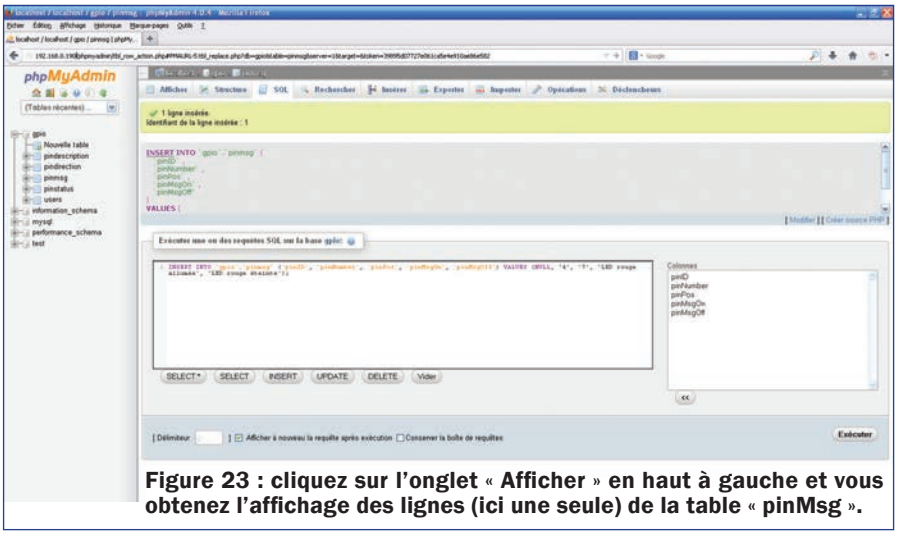


Figure 23 : cliquez sur l'onglet « Afficher » en haut à gauche et vous obtenez l'affichage des lignes (ici une seule) de la table « pinMsg ».

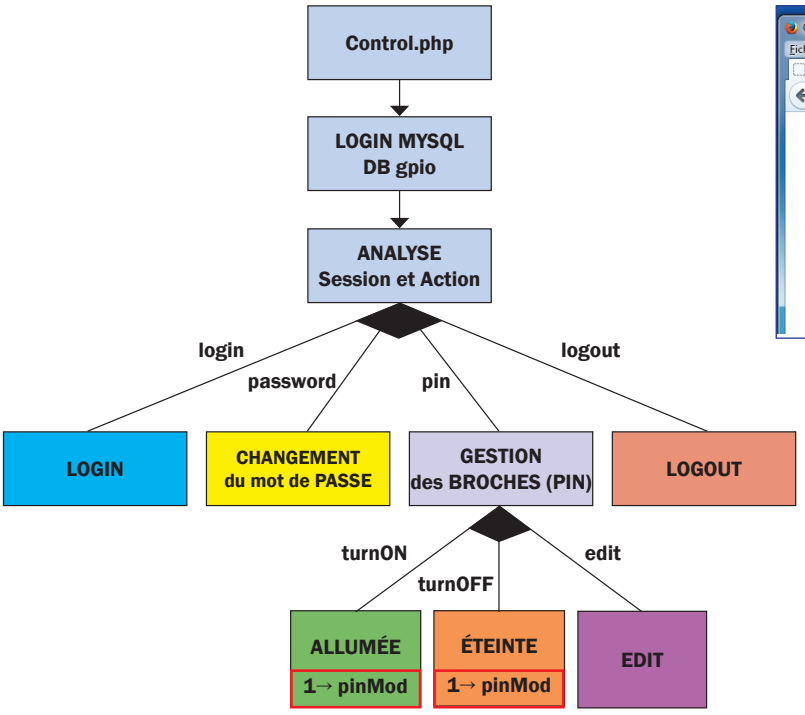
phpMyAdmin séparément si vous disposez déjà d'un serveur Apache et d'un environnement **PHP**. Selon les différentes versions de ces logiciels, vous aurez à adapter finement l'installation de **phpMyAdmin**. Le plus simple est d'utiliser une solution sous la forme d'un package complet comme les **serveurs WAMP** ou **EasyPHP** que vous pouvez télécharger aux adresses suivantes.

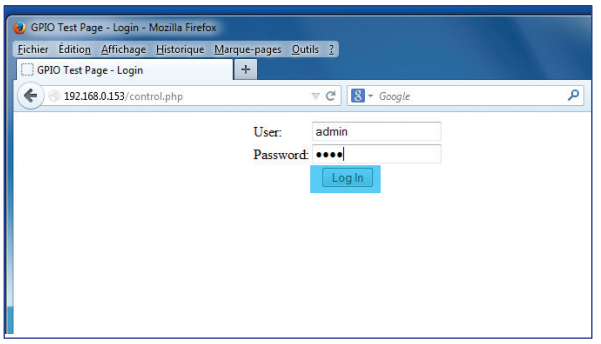
WAMP : www.wampserver.com
EasyPHP : www.easyphp.org

L'interface **phpMyAdmin** est écrite en **PHP**, elle est gratuite et sous licence publique générale GNU. Elle fonctionne sur toute plateforme Windows, Mac OS X et **LINUX**, et nécessite un environnement **PHP** (de la version 4 à 5) et une base **MySQL** (de la version 3 à 5).

Une fois installé, le programme **phpMyAdmin** offre de nombreuses fonctionnalités, notamment administrer plusieurs serveurs, créer, modifier, supprimer des bases de données, tables, index, utilisateurs et leurs privilèges,

Figure A : principe de fonctionnement du langage PHP.





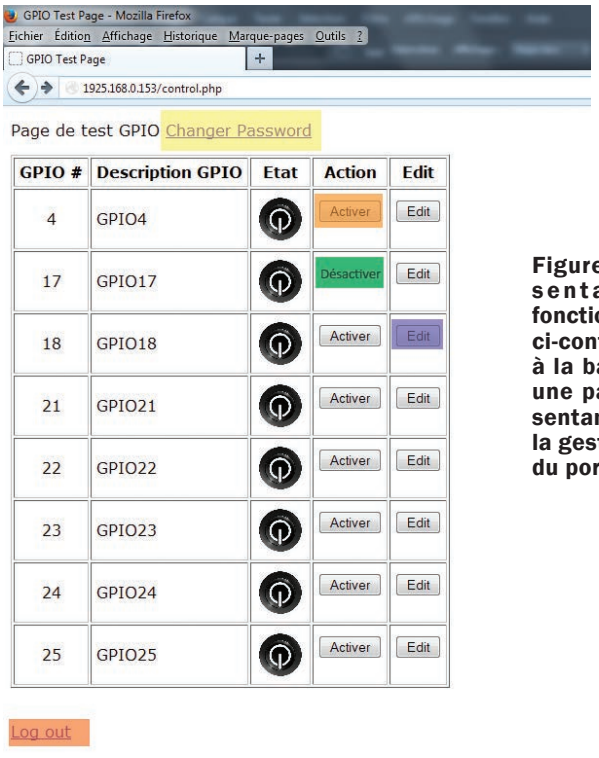


Figure 24 : diagramme représentant la structure de fonctionnement du programme, ci-contre une fois la connexion à la base de données réussie, une page web apparaît représentant sous forme de tableau la gestion et l'état des broches du port GPIO.

exécuter toute instruction SQL, événements, requêtes, importer et exporter des données sous différentes formes (texte, CSV, XML, PDF, Latex, etc.), créer des graphiques PDF du schéma de votre base de données.

Revenons maintenant à notre application, toujours dans le fichier **control.php** dans le numéro 125 d'Electronique et Loisirs Magazine, nous avons présenté le programme de gestion des pages web de notre projet. Dans ce programme, nous devons insérer les instructions nécessaires pour que le champ « **pinMod** » de la table « **pinStatus** » ait la valeur « 1 » quand une broche change d'état. Sans entrer dans les détails du langage **PHP** et de la logique de génération des pages **HTML** dynamiques, nous vous proposons le diagramme de la figure 24 qui représente la structure de fonctionnement du programme.

Dans le diagramme de la structure du programme, les losanges indiquent les blocs exécutés de manière sélective selon les valeurs contenues dans les variables de contrôle. Dans la figure 24, les rectangles colorés représentent les caractéristiques individuelles dans les pages web. Ces couleurs correspondent à celles des blocs dans le diagramme de la structure qui traitent

Listing 1

```
$action = $_GET['action'];
                $pin = mysql_real_escape_string($_GET['pin']);
                $setMod = "1";
            if ($action == "turnOn"){
                    $setting = "1";
                    mysql_query("UPDATE pinStatus SET pinStatus='$setting', pinMod='1' WHERE pinNumber='$pin';");
                mysql_close();
                    header('Location: control2.php');
                } else If ($action == "turnOff"){
                    $setting = "0";
                    mysql_query("UPDATE pinStatus SET pinStatus='$setting', pinMod='1' WHERE pinNumber='$pin';");
                    mysql_close();
                    header('Location: control2.php');
```

des mêmes fonctionnalités, et qui se retrouvent dans les sections correspondantes du listing.

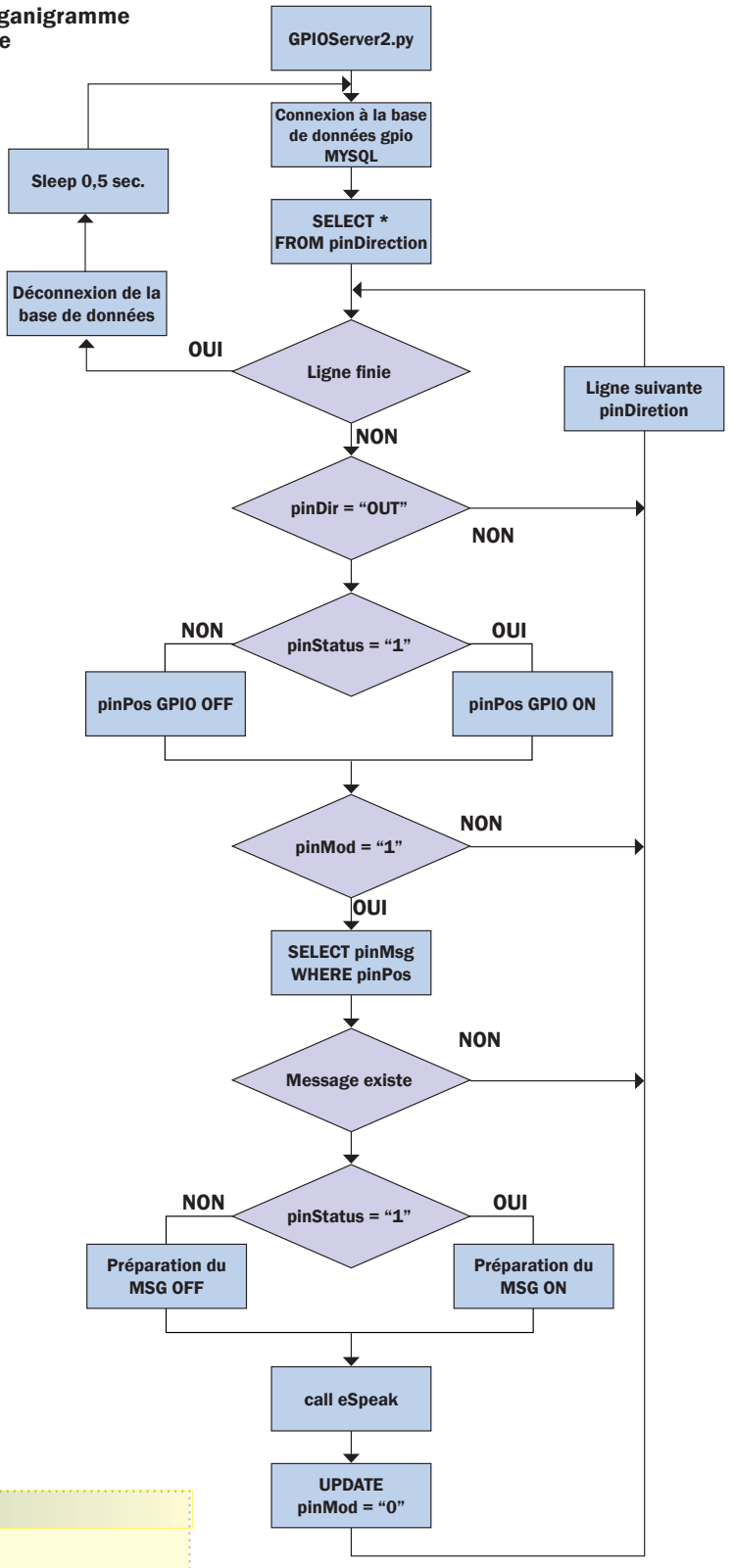
Lorsqu'un utilisateur externe demande la page, la première opération qu'effectue le programme est de se connecter à la base de données **gpio** hébergée sur le serveur **MySQL**. Ensuite, il vérifie si l'utilisateur se connecte pour la première fois (voir la figure 24a), dans ce cas, il renvoie la page de connexion et demande le Nom d'utilisateur (User) et le mot de passe (Password).

Une fois la connexion réussie, une page web représente sous forme de tableau la gestion et l'état des broches (voir la figure 24b). Cette page permet une modification de l'état d'une broche, la modification de la description d'une broche, la possibilité de changer de mot de passe et de quitter l'application en se déconnectant. **Chaque ligne fait référence à une seule broche**, et il est possible de faire varier l'état soit « **Allumée** » soit « **Eteinte** » à l'aide des boutons « **Activer** » ou « **Désactiver** », ou de modifier la description de la broche à l'aide du bouton « Edit ».

En se référant au diagramme, nous pouvons améliorer le comportement des blocs « ALLUMÉE » (branche « turnON ») et « ETEINTE » (branche « turnOff») de sorte que dans la table « **pinStatus** », en plus du champ « **pinStatus** » soit modifié également le champ « **pinMod** » en le mettant à la valeur 1.

Dans le **listing 1**, vous pouvez voir les instructions, mises en évidence par le rectangle rouge, pour obtenir le fichier **control2.php** qui est une copie du fichier **control.php**.

Figure 25 : organigramme du programme



Listing 2

```
$MySQLHost = "localhost";
$MySQLDB = "gpio";
$MySQLUsername = "gpio";
$MySQLPassword = "pi";

$dbConnection = mysql_connect($MySQLHost, $MySQLUsername, $MySQLPassword);
mysql_select_db($MySQLDB, $dbConnection);
...
$query = mysql_query("SELECT pinDescription FROM pinDescription WHERE pin-
Number='$pin';");
...
mysql_query("UPDATE pinStatus SET pinStatus='$setting', pinMod='1' WHERE
pinNumber='$pin';");
...
mysql_close();
```

À l'intérieur du listing du fichier **control2.php**, nous devons modifier toutes les occurrences (nombre de répétitions d'un mot ou d'une expression dans un texte) de type « control.php » en « **control2.php** ». Le **listing 2** montre le code nécessaire pour se connecter à une base de données **MySQL** à partir d'un programme de **PHP**.

Pour pouvoir vous connecter depuis une page **PHP** à votre base de données **MySQL**, il faudra spécifier plusieurs paramètres :

- l'hôte (le serveur sur lequel MySQL est installé).
- le login utilisateur.
- le mot de passe.
- le nom de la base de données.

Par défaut, les paramètres mis en place sont :

$MySQLHost = "localhost"; (le serveur sur lequel MySQL est installé).
$MySQLDB = "gpio"; (le nom de la base de donnée).
$MySQLUsername = "gpio"; (utilisateur).
$MySQLPassword = "pi"; (mot de passe)

La connexion au serveur **MySQL** s'effectue par la fonction msql_connect(), sa syntaxe est :

msql_connect ('hôte', 'login', 'mot de passe')

La connexion à la base de données s'effectue par msql_select_db (nom de la base, identifieur de connexion). La fonction retourne aussi TRUE en cas de succès et FALSE en cas d'erreur, sa syntaxe est :

mysql_select_db($MySQLDB, $dbConnection)

Programme de gestion physique du port GPIO

Toujours dans le numéro **125 d'Electronique et Loisirs Magazine**, nous avons anticipé le choix du langage **Python** pour la création d'un programme de gestion côté serveur destiné à être exécuté en tant que processus permanent ou processus secondaire afin d'exécuter des fonctions spécifiques lorsqu'un événement particulier survient.

Il existe bien sûr d'autres langages disponibles pour l'écriture d'un processus côté serveur tels que les langages **C**, **Perl**, **Assembleur**, et bien d'autres, mais le langage **Python** se prête particulièrement bien à la situation, car il est relativement facile à apprendre et simple d'utilisation pour l'écriture de programmes. Par exemple, il n'est pas nécessaire de déclarer les variables avant de les utiliser et les terminaisons ne sont pas nécessaires pour les instructions et les blocs.

Listing 3

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

import time
import RPi.GPIO as GPIO
import MySQLdb as mdb
#import sys
from subprocess import call

DbServer = "localhost"
DbUser = "gpio"
DbPasswd = "pi"

X_pinPos  = 99
X_pinStatus = 0
X_pinDirection = "zzz"
X_pinMod = 0
X_pinZero = 0

while True:

 # con = mdb.connect('localhost', 'gpio', 'pi', 'gpio');
   con = mdb.connect(DbServer, DbUser, DbPasswd, 'gpio');

   with con:

       cur1 = con.cursor(mdb.cursors.DictCursor)
       cur1.execute("SELECT * FROM pinStatus")

       rows = cur1.fetchall()
       for row in rows:

         X_pinPos = row["pinPos"]
         X_pinStatus = row ["pinStatus"]
         X_pinMod = row["pinMod"]

         cur2 = con.cursor(mdb.cursors.DictCursor)
         cur2.execute("SELECT * FROM pinDirection WHERE pinPos = %s", (X_pinPos))
         rows2 = cur2.fetchall()
         for row2 in rows2:
             X_pinDirection = row2["pinDirection"]

             if X_pinDirection == "out":
                GPIO.setup(X_pinPos, GPIO.OUT)
                if X_pinStatus == "1":
                   GPIO.output(X_pinPos, True)
                else:
                   GPIO.output(X_pinPos, False)

                if X_pinMod == "1":
                   cur3 = con.cursor(mdb.cursors.DictCursor)
                   cur3.execute("SELECT * FROM pinMsg WHERE pinPos = %s", (X_pinPos))

                   if cur3.rowcount > 0:
                      rows3 = cur3.fetchall()

                      for row3 in rows3:

                         if X_pinStatus == "1":
                            parle=row3["pinMsgON"]
                         else:
                            parle=row3["pinMsgOFF"]

                         call(["espeak", parle, "-v", "fr", "-p", "70", "-s", "155"])

                         cur4 = con.cursor(mdb.cursors.DictCursor)
                           cur4.execute ("""UPDATE pinStatus SET pinMod=%s
WHERE pinPos=%s""",(X_pinZero, X_pinPos))

   if con:
           con.close()

   time.sleep (0.5)
```

La structure logique du programme s'établit en gérant correctement les indentations pour identifier les blocs d'instructions soit en instructions individuelles ou en blocs structurés. En **Python** nous devons prêter une attention particulière à l'indentation des blocs d'instruction afin de répondre à la logique du programme.

Voyons maintenant comment adapter le fichier **GPIOServer**.py que nous avons présenté dans le dernier numéro afin qu'il puisse reconnaître le changement d'état d'une broche et permettre le transfert au synthétiseur vocal pour produire le bon message. Dans ce cas, nous vous proposons également le listing du programme (voir le Listing 3), afin de se familiariser avec le langage **Python**. Les commentaires sont signalés par le caractère « **#** ».

Les déclarations « **import** » servent à inclure les librairies nécessaires au fonctionnement du programme ; dans notre cas la librairie « **time** » permet la gestion du retard dans un cycle d'exécution du programme et le suivant ; la librairie « **RPi.GPIO** » permet la gestion des broches (pin) du port **GPIO** ; la librairie « **MySQLdb** » réalise l'interface avec la base de données **MySQL**.

L'appel de la fonction d'importation de la bibliothèque est un sous-processus qui est nécessaire parce que nous avons décidé d'appeler le synthétiseur vocal « **eSpeak** » en tant que processus externe. Cela signifie que chaque fois que vous devez exécuter « **eSpeak** » pour synthétiser un message vocal, le programme **Python** nécessite l'ouverture d'un nouveau processus qui génère la commande pour exécuter « **eSpeak** » avec la même syntaxe qu'une ligne de commande :

call([“espeak”, parle, “-v”, “fr”, “-p”, “70”, “-s”, “155”])

L'utilisation des guillemets vous permet de transférer correctement les variables et les constantes de l'argument lors de l'appel de la ligne de commande du processus externe. Cela rend plus facile l'exécution des tests à partir de la ligne de commande avant de modifier les instructions dans le programme ou insérer des messages dans la table « **pinMsg** » dans la base de données.

Sur la figure 25 est représenté l'organigramme du programme qui nous permettra de mieux comprendre le listing. Maintenant, examinons quelques particularités concernant l'interface avec la base de données **MySQL**. Le langage de référence pour la manipulation des bases de données relationnelles est le **langage SQL** (Structured Query Language), qui peut être utilisé comme un langage de commande ou intégré dans un autre langage hôte, tel que **Python**, dans notre cas.

Pour interroger les données dans une table, nous utilisons la commande « **SELECT** » avec sa syntaxe et ses paramètres.

Le résultat de la commande « **SELECT** » est une liste de lignes qui répondent aux critères de sélection de la commande « **SELECT** » elle-même. La liste peut être vide, si aucun n-uplet satisfait aux critères de recherche, ou composée d'un ou plusieurs n-uplet satisfaisant aux critères de recherche. Avec le langage **Python**, il est nécessaire de traiter le résultat de la sélection obtenue une ligne à la fois.

Vous devez utiliser une construction intermédiaire qui vous permette d'indexer individuellement les lignes de la liste obtenue. La construction en question est le curseur qui fournit une zone de données structurées de manière à contenir le résultat de la sélection afin que le mécanisme de pointage fasse défiler la zone de données une ligne à la fois.

Un curseur doit être utilisé conformément à une séquence correcte d'instructions qui est illustré ci-dessous, provenant du listing 2. Comme pour le langage **PHP**, la première des choses à faire est de se connecter à la base de données :

DbServer = “localhost”
DbUser = “gpio”
DbPasswd = “pi”

con = mdb.connect(DbServer, DbUser, DbPasswd, ‘gpio’);

Nous devons ensuite définir un curseur associé à la connexion en lui donnant un nom unique dans le programme :

cur1 = con.cursor(mdb.cursors.DictCursor)

La méthode « execute » permet de remplir la zone structurée avec le résultat de la commande « **SELECT** » :

cur1.execute(“SELECT * FROM pinStatus”)

Si vous devez vérifier que le nombre de lignes obtenues par la recherche est supérieur à zéro, vous pouvez utiliser l'instruction suivante :

if cur1.rowcount > 0

Pour affiner la recherche dans le programme vous pouvez utiliser l'instruction :

rows = cur1.fetchall()

Avec l'instruction « **fetchall()** », vous obtenez les lignes qui répondent aux critères de recherche de l'instruction « **SELECT** », avec l'instruction « **fetchone()** » seule la première des lignes correspondant aux critères de recherche est récupérée avec l'instruction « **fetchmany(num)** », vous obtenez un certain nombre de lignes correspondant à la recherche et dont le nombre est égal à « (num) ».

for row in rows

L'instruction « **for** » vous permet de faire défiler une ligne d'un sous-ensemble d'une recherche effectuée avec « **fetch** », les champs de chaque ligne peuvent être lus (et ensuite affichés) à l'aide de l'instruction suivante :

X_pinPos = row[“pinPos”]

A la fin du traitement, vous devez fermer la connexion à la base de données, ce qui libère toutes les ressources impliquées, avec l'instruction suivante :

con.close()

Le curseur est également utilisé pour exécuter des instructions SQL différentes de l'instruction « **SELECT** »,

par exemple dans le cas où nous voulons réinitialiser le champ « **pinMod** » de la table « **pinStatus** » afin d'éviter de répéter le message plusieurs fois :

```
cur4.execute ("""UPDATE pinStatus SET pinMod=%s WHERE pinPos=%s""",(X_pinZero, X_pinPos))
```

Les modifications à apporter au programme d'origine, pour y inclure la fonctionnalité de la synthèse vocale sont mises en évidence par les rectangles rouges du Listing 3.

Si vous ne voulez pas effectuer manuellement les modifications décrites dans cet article, vous pourrez télécharger le listing sur notre site web :

www.raspberrypi.electroniquemagazine.com.

Nous avons également ajouté un nouveau fichier « **gpio.sql** » pour créer à partir de zéro une nouvelle base de données **gpio** avec la nouvelle configuration.

Pour essayer les **nouveaux programmes** vous devez vous référer aux **instructions** décrites dans le numéro **125 d'Electronique et Loisirs Magazine**, en accordant une attention particulière sur le fait que nous avons changé les noms des programmes, afin de maintenir les deux versions opérationnelles en même temps sur le **RaspberryPi**.

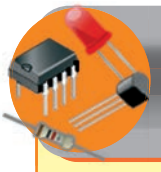
Donc à partir du web, vous devez appeler le fichier « **control2.php** » de la manière suivante :

http://adresse-ip/control2.php

et comme programme côté serveur, vous lancez « **GPIOServer2.py** » par l'utilisateur « root » de la manière suivante :

```
cd /home
python GPIOServer2.php
```

Dans le prochain numéro, nous approfondirons encore les possibilités d'intégration des applications à partir du périphérique **GPIO** en réalisant une station météo connectée au web à l'aide du **RaspberryPi**.

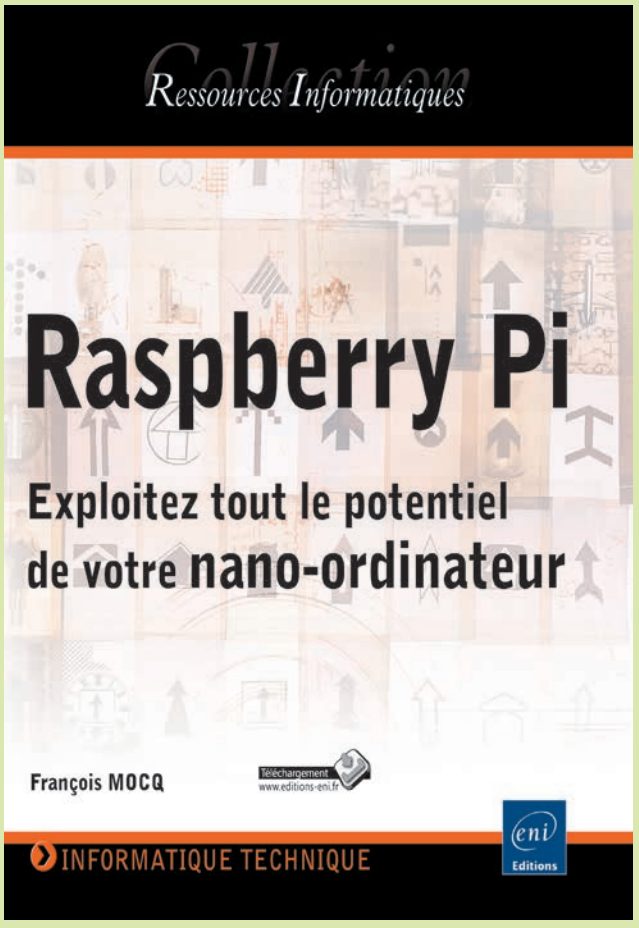


Le RaspberryPi et ses accessoires

Le **pack complet** pour le **RaspberryPi** comprenant un boitier, une carte SD, une alimentation et les câble est disponible auprès de l'annonceur **COMELEC**.

Sur notre site internet dédié au **RasperryPi,** vous pouvez télécharger gratuitement tous les programmes et applications : **www.raspberrypi.electroniquemagazine.com**

COMELEC
Tél. : 04 42 70 63 90
www.comelec.fr



Le livre de référence du RaspberryPi de François MOCQ Editions ENI

L'objectif de ce livre est de fournir au lecteur des bases solides pour explorer les ressources offertes par le **Raspberry Pi**. Ces connaissances serviront de socle pour un futur approfondissement des différentes voies offertes : Système d'exploitation, développement, interfaçage physique... Aucun prérequis en Linux, en programmation ou en électronique n'est nécessaire.

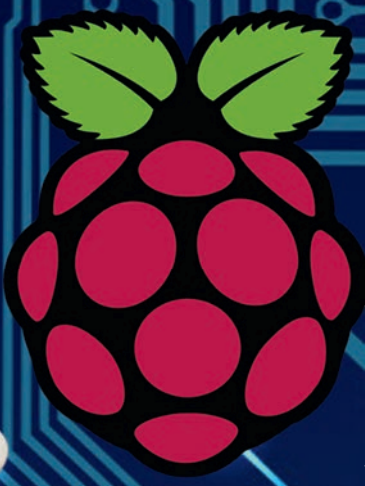
Les chapitres du livre

Avant-propos – Le Raspberry Pi – Description technique – Systèmes d'exploitation disponibles – Préparer la carte SD – Démarrer Raspbian – Utiliser la ligne de commande – Utiliser le mode graphique – Utiliser une mémoire de masse externe – Démarrer sur un disque externe – Que faire avec le Raspberry Pi ? – Programmer en Scratch – Programmer en Python – Le GPIO du Raspberry Pi – Les périphériques – Dépanner le Raspberry Pi

Date de parution : 12/03/14
Editeur : ENI (Editions)
Collection : Ressources informatiques
ISBN : 978-2-7460-8777-4
EAN : 9782746087774
Présentation : Broché
Nb. de pages : 560 pages

http://www.framboise314.fr/livre-eni/

PACK DE DÉMARRAGE RASPBERRY PI



Réf. RASPKIT-PACK

81,00 €

Le Raspberry Pi est un ordinateur monocarte à processeur ARM pas plus gros qu'une carte de crédit. Il réalise de nombreuses fonctions comme se connecter à une télévision, une souris ou un clavier.
Il permet également l'exécution de plusieurs variantes du système d'exploitation libre GNU/Linux et des logiciels compatibles.

Le pack comprend:

Le Raspberry Pi modèle B, (512 Mo de RAM),
Le boîtier pour Raspberry PI,
Une carte micro SDCard HC de 4GB avec le logiciel nécessaire pour exécuter les expérimentations.
Une alimention à découpage ultras compacte, (68 x 35 x 14 mm) avec sortie USB 5VDC/1A,
Un câble HDMI d'une longueur de 1,5 mètre,
Un câble USB M(A, / Micro(B, de longueur 0,7 mètres et un câble FTP CAT5E de longueur 0,75 mètre.
Contenu de la carte SD
Système opérationnel Raspbian wheezy 2-9-2013 mis à jours le 18 mai 2013
Server SSH activé
Server web apache2 installé et configuré
Server MySQL installé et configuré
Utilisateur «root» activé avec mot de passe «root»
Server emoncms installé et configuré
Les bases de données RaspiBase et emoncms chargées et configurées
Paquet espeak installé
Inclus des langages de développement et des librairies de support

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

CPDV-1

Capacimètre numérique de 10 pF à 10 000 µF

Première partie

de Michele Menniti

Dans cet article, nous vous proposons de réaliser un capacimètre digital sans microcontrôleur et donc facilement reproductible par nos lecteurs néophytes, en utilisant seulement des circuits intégrés TTL et CMOS. Le but est de vous proposer un cours didactique en abordant la théorie et la pratique de fonctionnement de la logique numérique et de la faire connaître à ceux qui ne l'ont jamais utilisée.

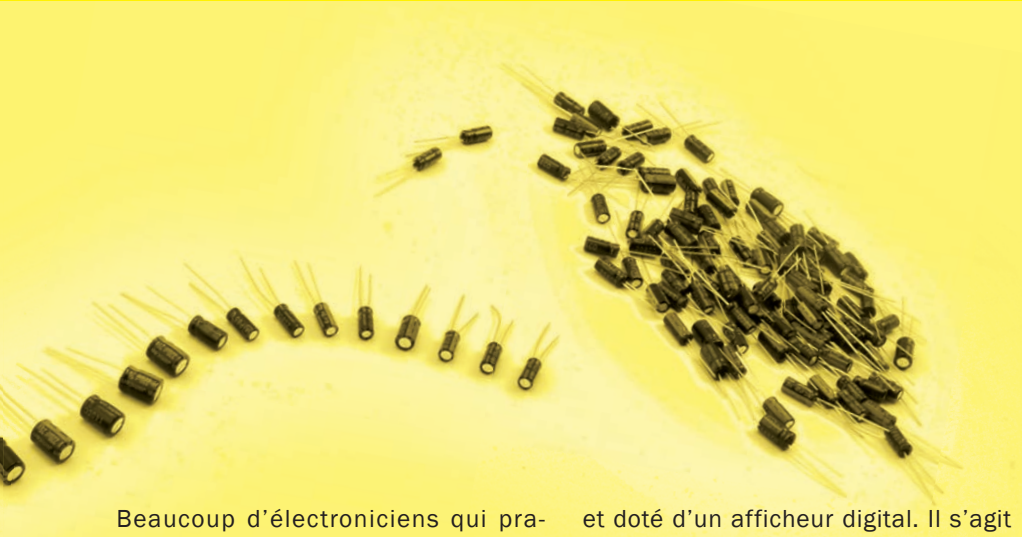
Sans aucun doute, l'apparition des microcontrôleurs a provoqué un bouleversement dans le monde de l'électronique numérique. Parfois de très petite taille, ces composants sont désormais en mesure d'exécuter des taches (des fonctions) qu'il y a quelques décennies demandaient une conception extrêmement complexe et une mise en œuvre de circuits imprimés de grande taille à cause du nombre de composants nécessaires; sans parler de la consommation énergétique et de la nécessité de gérer la dissipation thermique considérable due au grand nombre de composants utilisés impliquant des coûts de réalisation.

Afin de mieux comprendre ces questions, prenons l'exemple simple très bien connu du clignotement d'une LED à intervalles précis de 2 secondes (une seconde pour l'allumage et une seconde pour l'extinction), un petit microcontrôleur comme l'**ATMEL ATtiny25** est en mesure de contrôler la LED avec moins de dix lignes de code. Il suffit alors que d'un petit microcontrôleur, d'une résistance de limitation pour le courant circulant dans la LED, et d'une pile de 3 V, la consommation de l'ensemble du circuit sera inférieure à celle de la LED.

Si dans les années 70, nous avions voulu réaliser quelque chose de similaire avec la même précision de fonctionnement, nous aurions dû penser à un circuit intégré oscillateur avec un quartz pour générer une fréquence de 1 Hz, afin de réaliser la base de temps. Ensuite il aurait fallu un autre circuit intégré pour réduire de moitié la fréquence afin d'allumer la LED pendant la moitié positive et l'éteindre pendant la moitié négative.

De plus la pile de 3V n'aurait pas convenu car les circuits intégrés **TTL** nécessitent une tension d'alimentation de 5 V. Cela aurait pu devenir possible avec l'arrivée de la famille des circuits intégrés **CMOS** dont la tension d'alimentation s'étend de **3 V** à **15 V**, mais nous n'aurions pas pu diminuer la complexité du circuit ainsi que la consommation d'énergie, qui bien que plus faible que celle des circuits **TTL**, reste néanmoins plus élevée que celle de la version moderne à microcontrôleur.

Si à l'époque nous avions voulu faire clignoter deux (ou plusieurs) LED à des fréquences différentes, le problème aurait augmenté de façon exponentielle, alors qu'aujourd'hui il suffit d'ajouter une ou plusieurs résistances en fonction du nombre de LED et quelques lignes de code supplémentaires ! Mais comme dans toute évolution historique (oui, parce que c'est l'histoire de l'électronique) c'est toujours fascinant d'étudier, ce qui pour aujourd'hui est défini par le terme « vintage » (identique en français et en anglais), quelque chose qui remonte à au moins deux décennies, et qui a encore sa valeur ou au moins son propre charme.



Beaucoup d'électroniciens qui pratiquent l'électronique depuis les années 70 et 80, reconnaissent les avantages des microcontrôleurs, mais n'ont pas hésité à continuer à réaliser des circuits avec les technologies **TTL**/**CMOS**, appartenant à des familles de dernière génération qui prennent moins de place (elles existent maintenant en version CMS) et consomment moins d'énergie.

En outre, ces circuits sont encore largement utilisés dans les montages à microcontrôleurs pour exécuter des fonctions auxiliaires, qui prennent beaucoup de ressources pour un microcontrôleur (pour traiter un signal de 8 bits il faut au moins trois broches d'un microcontrôleur). Par exemple prenons le cas du « **SHIFT REGISTER** » ou « **registre à décalage** ».

Un registre à décalage est un ensemble de bascules synchrones reliées l'une à l'autre, à l'exception de deux bascules qui ne sont pas forcément reliées. À chaque cycle d'horloge, le nombre représenté par ces bascules est mis à jour, c'est-à-dire qu'à chaque cycle d'horloge, on ne peut écrire que dans une bascule ou forcer la valeur de toutes les bascules (tampons, temporisateur, division par une puissance 2).

Cet article, placé dans le contexte de l'enseignement didactique de notre revue, vise à introduire le monde fascinant de la logique numérique pour ceux qui ne l'ont jamais utilisé ou qui ont besoin de rappels. Comme l'étude théorique n'est pas très attractive pour nos lecteurs, nous vous proposons de **réaliser un outil professionnel performant capable de mesurer tout type de condensateur** dont la valeur s'étend de **10 pF** à **10 000 µF**, soit la plupart des condensateurs utilisés en électronique et doté d'un afficheur digital. Il s'agit donc d'un outil extrêmement utile dans un laboratoire, notamment en raison de l'avènement des composants CMS, dont la valeur des condensateurs non polarisés n'est parfois pas indiquée sur le boîtier et donc impossibles à identifier.

Bien sûr, nous savons que n'importe quel microcontrôleur avec un circuit externe minimal peu convenir pour réaliser un outil similaire, mais nous pouvons vous assurer qu'il n'a rien à envier à une version « microcontrôlée », et présente l'avantage de vous permettre de bien comprendre la fonction d'une douzaine de circuits intégrés logiques. Commençons par un peu d'histoire et de théorie.

Les familles logiques et leur évolution technologique

Les familles logiques, au fil du temps, ont été construites dans de nombreuses variantes, chacune caractérisée par des spécifications particulières, et cela a créé beaucoup de confusion, aggravée par le fait que, dans les années 70 et 80 trouver des fiches techniques (datasheet) de n'importe quel composant se résumait à l'achat de livres coûteux et devenant obsolètes lorsque de nouvelles familles apparaissaient. Eh oui ! Internet n'existait pas ! Aujourd'hui, il suffit de faire une recherche sur Google et en quelques secondes apparait un joli fichier PDF avec toutes les spécifications d'un composant particulier.

Même aujourd'hui, il est difficile de comprendre tous les acronymes (mots formés d'initiales provenant de plusieurs mots éventuellement composés), néanmoins, nous allons essayer de vous donner quelques indications. Ces informations vous seront très utiles si vous décidez de tenter votre chance dans le projet que nous allons vous présenter dans ces pages, ou même plus simplement, lors de l'utilisation de composants appartenant à ces familles.

La caractéristique électronique de ces circuits intégrés appartenant à ces familles réside dans le fait qu'ils fonctionnent grâce à la transmission et la reconnaissance de signaux caractérisés par deux niveaux logiques : H (valeur proche de l'alimentation positive) et L (valeur proche de 0 V).

La famille que nous pouvons définir comme historique est la famille **TTL** (**Transistor-Transistor Logic**) et identifiée par les caractères « **74NNN** », où « **NNN** » est un **nombre indiquant la fonction du circuit**.

Par exemple « **00** » (7400) correspond à un circuit comportant **4 portes logiques** à **2 entrées NON-ET** (**NAND**). La famille **TTL** a été inventée dans les années 60, elle est réalisée avec la technologie du transistor bipolaire et tend à disparaître du fait de sa consommation électrique élevée comparativement aux circuits **CMOS**.

Plus tard, est apparu la famille **CMOS** (Complementary Metal Oxide Semiconductor), qui a été identifiée par les caractères « **40NNN** », où « **NNN** » représente un nombre indiquant la fonction du circuit. Par exemple « **01** » (**4001**) correspond à un circuit comportant 4 portes logiques indépendantes **NON-OU** (**NOR**) à 2 entrées.

Dans ces circuits, l'étage de sortie est composé d'un couple de transistors **MOSFET N** et **P** placés de manière symétrique et réalisant chacun la même fonction. Du fait de leur caractéristique de fonctionnement inversée, un transistor est passant alors que l'autre est bloqué (ils sont donc complémentaires, d'où l'appellation « Complementary »).

La famille **CMOS** est caractérisée par une consommation inférieure à la famille **TTL** ainsi que la possibilité d'être alimentée avec une tension comprise entre **3 V** et **15 V**, ce qui

a permis de mettre en œuvre de la logique numérique dans les circuits analogiques qui sont typiquement alimentés en 12 V.

Malgré cela, **il n'était pas possible d'interfacer directement les deux familles TTL** et **CMOS**, même toutes les deux alimentées en **5 V**, sans avoir recours à des techniques appropriées d'adaptation des niveaux logiques. Afin d'éviter toute erreur d'interversion des circuits intégrés **TTL** et **CMOS** ayant les mêmes caractéristiques logiques, ils ont été nommés différemment, avec un brochage interne différent.

Par exemple, le « **00** » du **74** correspondant à la famille **TTL** contient **4 portes NAND**, tandis que le « **00** » du **40** de la famille **CMOS** contient 2 à 3 portes **NOR** et une porte **NOT**. Le **CMOS** équivalent du **TTL 7400** est le **4011**, mais le brochage est différent.

Dans le **Tableau 1**, nous avons résumé les caractéristiques générales des familles standards **TTL** et **CMOS** et les principales familles dérivées au fil du temps, dont certaines sont récentes.

Voici les abréviations utilisées dans le **Tableau 1** :

- V_{CC} : tension d'alimentation ;
- V_{IL} : niveau de tension maximal nécessaire pour être reconnu comme signal d'entrée **LOW** (niveau bas) ;
- V_{IH} : niveau de tension minimum nécessaire pour être reconnu comme signal d'entrée **HIGH** (niveau haut) ;
- V_{OL} : niveau de tension maximal présent à la sortie pour un signal **LOW** (niveau bas) ;
- I_{OL} : courant maximum circulant dans la sortie à l'état bas (**LOW**) ;
- V_{OH} : niveau de tension minimum présent à la sortie pour un signal **HIGH** (niveau haut) ;
- I_{OH} : courant maximum disponible en sortie à l'état haut (**HIGH**) ;
- F_{max} : fréquence maximale de fonctionnement de la famille (la valeur peut varier considérablement d'un modèle à l'autre au sein de la même famille).

Pour simplifier le tableau récapitulatif, nous avons indiqué pour V_{IH} et V_{OH} les valeurs minimales, et pour V_{IL} et V_{OL} les valeurs maximales, ces valeurs sont toujours référencées au 5V.

Dans le cas où la tension **Vcc** est inférieure à 5 V (généralement de 3,3 à 3,6 V), la référence est prise à la valeur de **Vcc**. On peut noter que, quelle que soit le type de famille, il existe entre le niveau maximal reconnu comme bas (**LOW**) et le niveau minimum reconnu comme haut (**HIGH**) aussi bien en entrée qu'en sortie, une zone de séparation définie comme une « zone indéterminée », les valeurs

Sigle	Signification	Niveau logique	Description	V_{CC}	V_{IL}	V_{IH}	V_{OL}	I_{OL}	V_{OH}	I_{OH}	F_{max} (MHz)
TTL	**Transistor-Transistor Logic**		**Transistors en entrée et en sortie**								
54	standard (militaire)	TTL	-55 à +125 °C	5V	0,8V	2V	0,4V	16 mA	2,4V	400 µA	35
74	standard (commercial)	TTL	0 à 70 °C	5V	0,8V	2V	0,4V	16 mA	2,4V	400 µA	35
L	Low power	TTL	basse consommation	5V	0,8V	2V	0,5V	3,6 mA	2,7V	200 µA	3
S	Schottky	TTL	rapide (TR Schottky)	5V	0,8V	2V	0,5V	20 mA	2,7V	1 mA	125
F	Fast	TTL	très rapide et haute impédance (remplace la famille S)	5V	0,8V	2V	0,5V	20 mA	2,7V	1 mA	125
LS	Low power Schottky	TTL	famille S à basse consommation et un peu moins rapide	5V	0,8V	2V	0,5V	8 mA	2,7V	400 µA	45
AS	Advanced Schottky	TTL	famille S avancée	5V	0,8V	2V	0,5V	20 mA	2,7V	2 mA	200
ALS	Advanced Low power Schottky	TTL	famille LS avancée	5V	0,8V	2V	0,5V	8 mA	2,7V	400 µA	50
CMOS	**Complementary MOS**		**MOSFET**								
40/45XX	Cmos	CMOS	brochage non compatible TTL	3V à 15V	1,5V	3,5V	0V	1 mA	5V	1 mA	4
74C	Cmos	CMOS	compatible broche à broche TTL mais niveau CMOS	3V à 15V	0,7V	3,15V	0,5V	3,3 mA	5V	3,6 mA	4
74AC	Advanced Cmos	CMOS	famille C avancée	2V à 6V	1,5V	3,15V	0,1V	24 mA	5V	24 mA	160
74ACT	Advanced Cmos TTL	TTL	famille C avancée avec niveau TTL	5V	0,8V	2V	0,1V	24 mA	5V	24 mA	160
74HC	High Cmos	CMOS	série C avec temps de propagation réduit (niveau CMOS)	2V à 6V	1,35V	3,15V	0,4V	4 mA	3,7V	7,5 mA	80
74HCT	High Cmos TTL	TTL	série C avec temps de propagation réduit (niveau TTL)	5V	0,8V	2V	0,4V	4 mA	3,7V	4 mA	80
74AHC	Advanced High Cmos	CMOS	série HC avancée (niveau CMOS)	2V à 5,5V	1,5V	3,5V	0V	8 mA	5V	8 mA	160
74AHCT	Advanced High Cmos TTL	TTL	série HCT avancée (niveau TTL-LS)	5V	0,8V	2V	0,8V	8 mA	5V	8 mA	160
74LV	Low voltage	CMOS	basse tension	1V à 5,5V	1,5V	3,5V	0,35 V	35 mA	2,2V	35 mA	60
74LVC	Low voltage CMOS	CMOS	basse tension (niveau CMOS)	1,2V à 3,6V	0,8V	2V	0V	24 mA	2,2V	24 mA	500
74LVT	Low voltage TTL	TTL	logique BICMOS basse tension (niveau TTL)	2,7V à 3,6V	0,8V	2V	0,5V	64 mA	2V	32 mA	500
74ALVC	Advanced Low voltage CMOS	CMOS	série LVC avancée (niveau CMOS)	1,65V à 3,6V	0,8V	2V	0,5V	24 mA	2V	24 mA	300
74ALVT	Advanced Low voltage TTL	TTL	série LVT avancée (niveau TTL)	2,5V à 3,3V	0,7V	2V	0,8V	12 mA	2V	8 mA	300

Tableau 1 : les caractéristiques techniques des différentes familles TTL et CMOS

des tensions qui se situent dans cette zone ne sont pas reconnues comme valides et sont donc ignorées.

Cela garantit **que la reconnaissance du niveau logique est sûre** mais, en même temps cela implique que les circuits intégrés doivent être dans un état de fonctionnement correct pour générer des tensions valides, c'est à dire avec des valeurs qui se situent au-dessous ou au-dessus de la zone indéterminée.

Ce tableau a seulement une valeur indicative, en particulier pour le courant de sortie et la fréquence de travail, étant donné que pour chaque circuit il existe une valeur maximale absolue différente de celles des valeurs standards, mais qui n'est pas conseillée de prendre comme référence. Le courant de sortie fait toujours référence au sens positif dans l'état « **HIGH** », c'est-à-dire que le courant sort de la broche, tandis que dans le cas « **LOW** » le courant entre.

La dénomination classique des familles logiques se présente en général sous la forme « **AAYYBBBXXXXC** ». Par exemple **SN74LS00N**, dans laquelle :

- **AA** est le préfixe qui indique le fabricant de la puce (SN = Texas, MD = National, CD = RCA) ;

- **YY** sont des symboles qui représentent la famille logique **TTL** et **CMOS**, par exemple 74 représente une famille **TTL** pour un usage commercial et 54 pour une utilisation militaire. Pour la famille **CMOS** c'est 40 ou 45 ;

- **BBBB** sont les initiales de la famille originelle, elles peuvent être absentes pour une famille standard ou représentées par un « N » (Normal). Elles sont utilisées pour identifier la sous-famille et la technologie de construction dont voici les principales :

- **TTL** : série standard ;

- **TTL-L** (low power) : série à faible consommation ;

- **TTL-S** (shottky) : série rapide (utilisation de diodes schottky) ;

- **TTL-AS** (advanced shottky) : version améliorée de la série S ;

- **TTL-LS** (low power shottky) : combinaison des technologies L et S, c'est la famille la plus répandue ;

- **TTL-ALS** (advanced low power shottky) : version améliorée de la série LS ;

- **TTL-F** (FAST : Fairchild Advanced Schottky Technology);

- **TTL-AF** (advanced FAST) : version améliorée de la série F ;

- **TTL-HC** (high speed C-MOS) : circuit TTL fabriqué en technologie C-MOS dans un boîtier TTL (tension compatible TTL, mais pas l'emplacement des broches) ;

- **TTL-HCT** (high speed C-MOS transposed) : série HC dotée de niveaux logiques compatibles TTL (100 % compatible TTL, car le brochage TTL est conservé) ;

- **XXXX** est le nombre qui identifie le type de circuit logique contenu dans le circuit dont voici une principale liste :

- **7400** : 4 portes logiques NON-ET à 2 entrées ;

- **7402** : 4 portes logiques NON-OU à 2 entrées ;

- **7404** : 6 portes logiques inverseuses NON ;

- **7407** : 6 tampons avec sortie à collecteur ouvert d'une protection de 30 V ;

- **7408** : 4 portes logiques ET à 2 entrées ;

- **7410** : 3 portes logiques NON-ET à 3 entrées ;

- **7411** : 3 portes logiques ET à 3 entrées ;

- **7413** : 2 portes logiques NON-ET à 4 entrées avec trigger de Schmitt ;

- **7414** : 6 portes logiques inverseuses NON avec trigger de Schmitt ;

- **7415** : 3 portes logiques ET à 3 entrées avec sortie à collecteur ouvert ;

- **7418** : 2 portes logiques NON-ET à 4 entrées avec trigger de Schmitt ;

- **7419** : 6 portes logiques inverseuses NON avec trigger de Schmitt ;

- **7420** : 2 portes logiques NON-ET à 4 entrées ;

- **7421** : 2 portes logiques ET à 4 entrées ;

- **7423** : double porte NON-OU à 4 entrées avec validation ;

- **7424** : 4 portes logiques NON-ET à 4 entrées avec trigger de Schmitt ;

- **7427** : 3 portes logiques NON-OU à 3 entrées ;

- **7430** : une porte NON-ET à huit entrées ;

- **7432** : 4 portes logiques OU à 2 entrées ;

- **7436** : 4 portes logiques NON-OU à 2 entrées (brochage différente du 7402) ;

- **7441** : décodeur BCD vers décimal ;

- **7443** : décodeur 4 bits vers décimal incrémenté de 3 (3 à 12) ;

- **7449** : décodeur BCD à 7 segments avec sortie à collecteur ouvert

- **74123** : double monostable ;

- **74132** : 4 portes logiques NON-ET à 2 entrées avec trigger de Schmitt ;

- **74163** : compteur binaire de 4 bits ;

- **74184** : convertisseur BCD vers binaire.

- **C** : lettre indiquant le type de boîtier qui, pour les modèles au format DIP, peut être en céramique (D, F, J, L) ou en plastique (N, P).

Il existe une particularité au niveau de la dénomination, les nouvelles familles **CMOS**, dans la grande majorité des cas, sont représentées par des familles

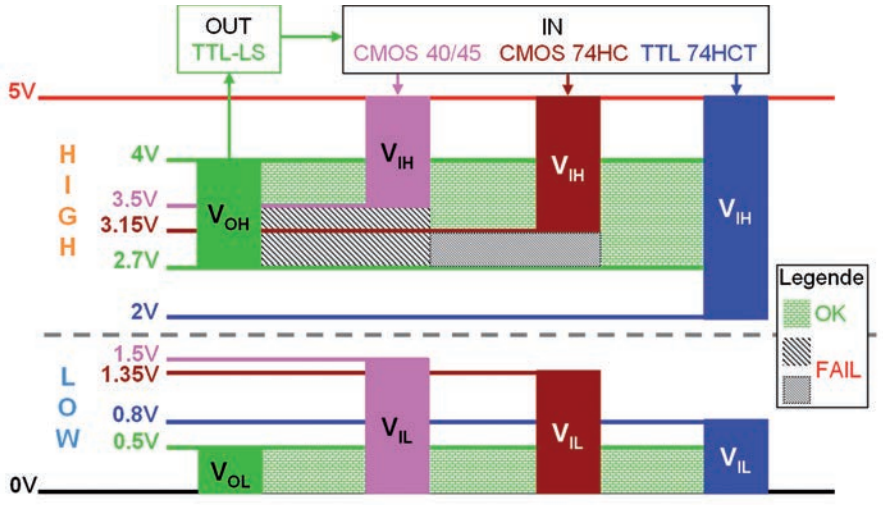


Figure 1 : niveaux logiques des familles TTL et CMOS.

TTL standards construites avec les nouvelles technologies, mais dans le respect du brochage original, si bien que, dans de nombreux cas, un ancien standard **TTL** peut se substituer, sans modification du circuit, à une plus récente ayant des niveaux logiques **TTL**.

Comme le montre le **Tableau 1**, les nouvelles familles **CMOS** se divisent entre celles qui ont gardé les niveaux logiques initiaux et celles qui se sont adaptées à des niveaux logiques **TTL**.

A l'intérieur d'une même famille nous trouvons, par exemple, les **SN74HCT90** et **SN74HCT4511** qui sont des répliques des circuits originaux **SN7490** au standard **TTL** et **CD4511** au standard **CMOS**, et qui sont parfaitement compatibles avec les niveaux logiques **TTL**. De même pour le **74HC90** et le **74HC4511** qui ont les mêmes niveaux logiques **CMOS**. Cependant, il reste le problème de l'interfaçage entre les niveaux logiques **TTL** et **CMOS** que nous allons traiter de manière exhaustive.

L'interfaçage logique entre différentes familles

Le problème de l'interfaçage entre les familles **TTL** et **CMOS** est lié à la différence des niveaux logiques entre les entrées et les sorties. Il est évident que pour communiquer correctement, les deux circuits logiques doivent être compatibles pour les tensions **VO** et **VI** des signaux respectifs. Examinons les problèmes et évoquons les solutions.

L'interfaçage entre une sortie CMOS et une entrée TTL ne devrait en théorie poser aucun problème parce que les valeurs $\mathbf{V_{OL}}$ **= 0 V** et $\mathbf{V_{OH}}$ **= 5 V** du **CMOS** semblent correspondre à la gamme des valeurs des niveaux **TTL** $\mathbf{V_{IL}}$ (max 0,8 V) et $\mathbf{V_{IH}}$ (min 2 V). En fait des problèmes peuvent survenir lors de la reconnaissance du niveau « **LOW** » (bas) du signal. En effet un courant peut circuler vers la sortie **CMOS** en provenance de l'entrée du circuit **TTL**, par l'intermédiaire du transistor **MOSFET N** de sortie du circuit **CMOS**.

Dans ce cas, il est possible que se produise une augmentation de la tension du niveau « **LOW** » (bas) à une valeur supérieure à **0,8 V**, ce qui, par conséquent, se situerait dans la zone indéterminée dont la plage est comprise entre **0,8 V** et **2 V** et que le circuit **TTL** ne reconnaît ni comme niveau « **LOW** » (bas) et ni comme niveau « **HIGH** » (haut).

Lorsque le signal de sortie du **CMOS** est à un niveau « **HIGH** », le courant provenant du circuit **TTL** passe à travers la sortie du **MOSFET P** du **CMOS**, dans ce cas, le changement de niveau ne pose pas de problème. Plus tard, nous verrons comment interfacer une sortie **CMOS** et une entrée **TTL**, pour surmonter le problème du niveau « **LOW** » (bas) du signal.

L'interfaçage entre une sortie **TTL** et une entrée **CMOS** dans le cas d'un niveau « **LOW** » (bas) à la sortie **TTL**, la valeur $\mathbf{V_{OL}}$ d'un circuit **TTL** (max 0,4 V) correspond à la gamme des valeurs $\mathbf{V_{IL}}$ du circuit **CMOS** (max 1,5 V).

Le problème se pose pour la reconnaissance d'un signal de niveau « **HIGH** » (haut), en effet la valeur $\mathbf{V_{OH}}$ d'un **TTL** peut commencer à partir d'un minimum de **2,7 V** (2,4 V même dans les versions de base), tandis que la valeur $\mathbf{V_{IH}}$ d'un **CMOS** est reconnue comme telle qu'à partir de **3,15 V**.

Lorsque l'on connecte directement des sorties **TTL** à des entrées de type **CMOS**, le niveau « **LOW** » (bas) est régulièrement reconnu, tandis que le niveau « **HIGH** » (haut) ne peut pas être reconnu, en particulier dans les situations de circulation de courants élevés, ce qui tend à réduire les niveaux de tension « **HIGH** » (haut) d'un circuit **TTL**. Pour mieux comprendre ce que nous venons de vous expliquer, observez attentivement la figure 1.

Elle se compose de deux parties séparées par une ligne en pointillés grise. Dans la partie inférieure sont représentés les niveaux **TTL** de la tension $\mathbf{V_{OL}}$ de la famille « **LS** » (en vert) et les niveaux **TTL** de la tension $\mathbf{V_{IL}}$ des familles **CMOS 40/45** (en violet), **TTL 74HC** (en marron) et **74HCT** (en bleu, cette famille a des niveaux compatibles **TTL**, comme indiqué dans le Tableau 1) et dans la partie supérieure sont représentés les niveaux $\mathbf{V_{OH}}$ des familles **TTL** et les niveaux $\mathbf{V_{IH}}$ des familles **CMOS**.

Commençons par étudier le niveau « **LOW** » (bas), le signal d'un niveau « **LOW** » d'une sortie **TTL** d'un circuit de type « **LS** » a un niveau de tension de **0,5 V.** Les entrées des trois familles **CMOS** acceptent des signaux de niveau « **LOW** » de **0 V** à **1,5 V** pour la famille **40/45**, **1,35 V** pour la famille **74HC** et **0,8 V** pour la famille **74HCT**. La valeur de **0,5 V** fait partie des gammes des familles **CMOS**, et est bien reconnue. Comme vous pouvez le voir dans la partie inférieure de la figure 1, le niveau $\mathbf{V_{OL}}$ de la famille « **LS** » (en vert) est toujours contenu dans les zones des niveaux $\mathbf{V_{IL}}$ des familles **CMOS**.

Voyons ce qui se passe dans la zone de niveau « **HIGH** » (haut). Le niveau de la tension du signal de sortie d'un circuit **TTL** de type « **LS** » peut varier dans une plage comprise entre **2,7 V** et **4 V**, mais cette plage n'est pas totalement

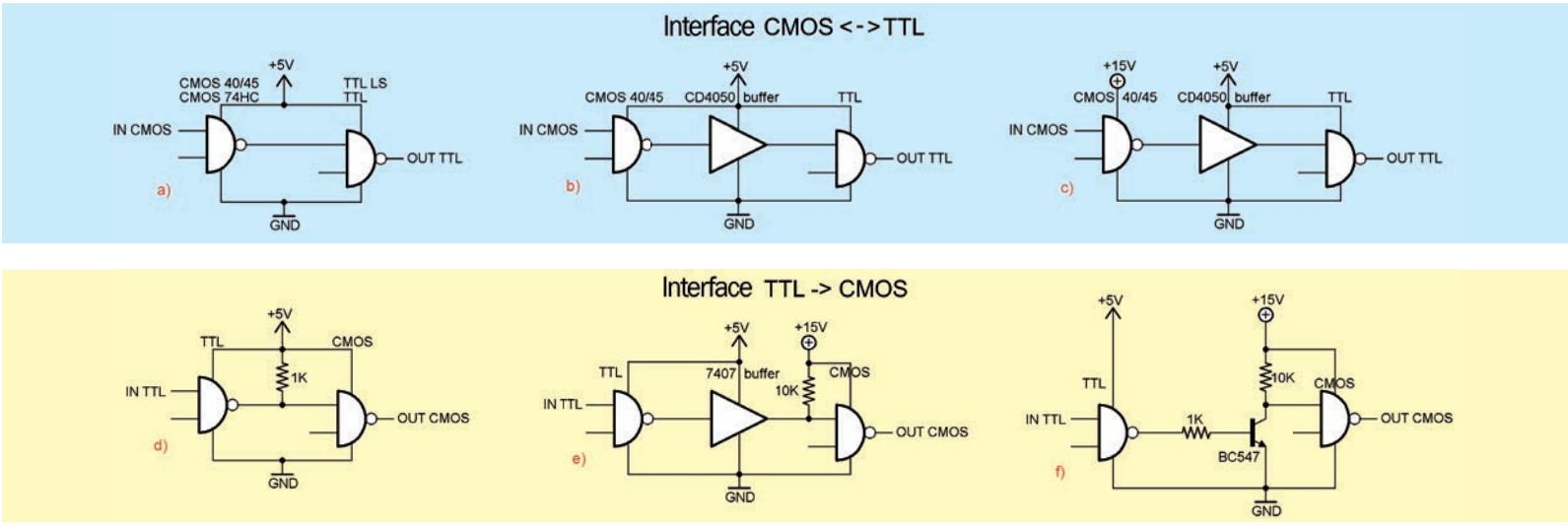


Figure 2 : solutions pour l'interfaçage entre les circuits TTL et CMOS

comprise dans les niveaux des deux familles **CMOS 40/45** et **74HC**, alors qu'elle est comprise dans les niveaux de la famille **74HCT** qui, comme nous l'avons déjà dit, sont compatibles **TTL**.

En regardant le graphique, il est facile de constater qu'un signal compris entre **2,7 V** et **3,15 V** est reconnu comme un niveau « **HIGH** » (haut) seulement sur l'entrée d'un **74HCT**, et un signal compris entre **3,15 V** et **3,5 V** est reconnu sur l'entrée d'un **74HC** et c'est seulement à partir de **3,5 V** jusqu'à **4 V maximum** que le signal serait reconnu par un **CMOS 40/45**.

Les deux bandes grises sur la figure 1 représentent précisément les zones critiques de la compatibilité entre la sortie d'un **TTL** « **LS** » et une entrée **CMOS**.

Il est nécessaire de préciser que des problèmes peuvent se produire même dans les zones de compatibilité car les chiffres que nous avons donnés sont purement théoriques. Il faut en effet considérer que la tension $\mathbf{V_{OH}}$ d'un **TTL** « **LS** » (mais plus généralement pour tous les TTL) tend à diminuer avec l'augmentation du courant à fournir, alors dans les conditions de faibles courants, la tension pourrait être suffisamment élevée pour se situer dans la zone de fonctionnement des **CMOS**.

Dans notre projet, nous avons essayé d'utiliser certains circuits **CMOS HC**, à la place normalement des versions **HCT**, et nous n'avons pas rencontré d'anomalie.

Toutefois, au stade de la conception, nous ne pouvons pas nous appuyer sur ce cas, les règles doivent être respectées. Nous ne pouvons pas négliger le fait que les familles **CMOS**, surtout les familles **40/45** et **74C**, peuvent être alimentées avec des tensions **supérieures** à **5 V** qui correspond aux circuits **TTL**, et il n'est pas du tout rare que nous ayons besoin d'interfacer un circuit **TTL** alimenté en **5V** avec un **CMOS** alimenté avec une tension beaucoup plus élevée de l'ordre de **10 V** à **12 V**.

En figure 2, nous montrons un certain nombre de solutions pour l'interfaçage des circuits **TTL** et **CMOS** et vice versa. Nous devons dire clairement que, quand nous parlons de familles **CMOS** et **TTL**, nous nous référons au type de niveau logique.

Dans le **Tableau 1**, vous aurez certainement remarqué que, bien que toutes les familles **TTL** ont des niveaux logiques **TTL**, différentes familles **CMOS** ont des niveaux logiques compatibles **TTL** (normalement, elle est représentée par les références qui ont la lettre « T » à la fin, par exemple ACT, HCT, etc.) et donc dans ce cas il n'y a aucun problème d'interfaçage.

Dans les solutions que nous décrivons, nous employons des portes **NAND** seulement pour raison de « commodité graphique » :

a) dans le schéma « a » (partie supérieure) de la figure 2 vous voyez que la sortie d'un circuit **CMOS** est parfaitement compatible avec un circuit **TTL** « **LS** ». En ce qui concerne la compatibilité avec un circuit **TTL** standard, que vous pouvez voir dans la partie inférieure, il faut que le circuit **CMOS** soit de type « **HC** » ou « **AHC** » ;

b) dans le cas de la circulation d'un courant provenant de l'entrée du circuit **TTL** (schéma « b »), à travers le **MOSFET** N de sortie du circuit **CMOS**, il est possible qu'il y ait une augmentation de la tension du niveau « **LOW** » (bas) à une valeur supérieure à **0,8 V**. Dans ce cas, il est possible d'appliquer le schéma qui consiste à insérer entre la sortie du circuit **CMOS** et l'entrée du circuit **TTL** un « **buffer** » (tampon) non-inverseur tel que le circuit **CMOS CD4050**, qui est un convertisseur compatible **TTL**. Avec ce type de solution, bien sûr n'importe quelle famille **CMOS** peut être interfacée avec n'importe quelle famille **TTL** ;

c) si vous avez besoin d'interfacer (schéma « c »)la sortie d'un circuit **CMOS** fonctionnant à des tensions supérieures à 5V (typiquement 10 V à 15 V) avec l'entrée d'un circuit **TTL** dont l'alimentation est de 5 V, il est toujours possible de recourir à un **CD4050**. Notez que cette solution implique que le circuit soit alimenté en 5 V et doit être capable de tolérer un signal d'entrée dont la tension est supérieure

L'anti-rebond (debounce)

L'un des problèmes les plus difficiles de la logique numérique a toujours été l'**anti-rebond** (debounce), qui consiste à l'**élimination ou le filtrage d'une série d'impulsions parasites qui ont eu lieu à chaque fois que l'on manipule un bouton**, un **interrupteur** ou un **commutateur mécanique**. Le problème a été aggravé par le fait que les boutons étaient exclusivement en métal et les contacts à glissière provoquaient de vrais feux d'artifice autres que des rebonds ! En fait, dans les systèmes professionnels, on avait souvent recours à des commutateurs à glissière ou des rotacteurs pour effectuer un changement de calibre ou autre.

Aujourd'hui, il est facile de mettre en œuvre une application spécifique intégrée, par exemple le **Maxim MAX6816-7-8** est respectivement un **commutateur simple**, **double** et **8 canaux anti-rebond**. Il sert d'interface entre les commutateurs mécaniques et les systèmes numériques. Il accepte une ou plusieurs entrées avec un « rebond » d'un interrupteur mécanique et produit une sortie numérique, les « rebonds » d'ouverture et de fermeture de l'interrupteur sont supprimés.

Le **MC14490** est identique mais avec **6 commutateurs anti-rebond**. On peut aussi régler le problème à l'aide d'un microcontrôleur correctement programmé en introduisant un retard de 100 ms lors de la manœuvre de l'interrupteur ce qui a pour effet d'élimer le « rebond ». Pour comprendre la somme de travail que doit accomplir un de ces circuits intégrés spécifiques, il suffit de regarder la figure A qui représente le schéma synoptique de l'interrupteur anti-rebond MC14490 à 6 canaux, c'est quelque chose de vraiment génial ! En figure B, nous avons représenté les signaux de l'oscillateur interne du circuit (en haut), au milieu les signaux

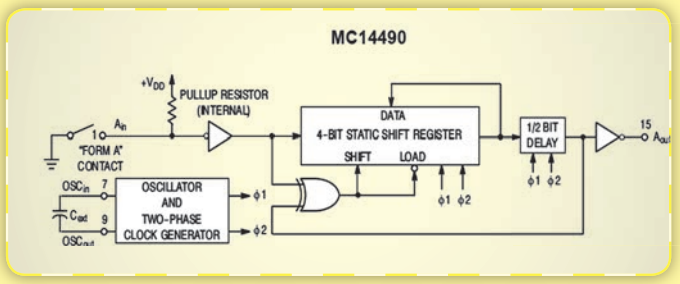


Figure A : schéma synoptique du circuit anti-rebond (debounce) MC14490.

qui comprennent les « rebonds » sur l'entrée, dans la partie inférieure la sortie contrôlée et dépourvue de toute perturbation.

Initialement, l'interrupteur n'est pas pressé, l'entrée (**INPUT**) et la sortie (**OUTPUT**) se trouvent à un état haut (**HIGH**). Dès que l'on appuie sur l'interrupteur (vers un niveau bas **LOW**), une série de rebonds apparaît sur l'entrée avec une séquence irrégulière de signaux de niveaux hauts et bas. Au bout d'un moment pendant cette phase les rebonds disparaissent et la sortie n'est pas affectée et reste à un niveau haut (**HIGH**). Seulement après un temps prédéfini, le niveau bas (**LOW**) de l'entrée **INPUT** est interprété comme valide et la sortie (**OUTPUT**) se porte à un niveau bas (**LOW**). Au moment où l'interrupteur est relâché, des rebonds réapparaissent, mais ceux-ci sont ignorés et la sortie (**OUTPUT**) reste à un niveau bas (**LOW**). Ce n'est que lorsque l'entrée (**INPUT**) se stabilise à un niveau haut (**HIGH**) au bout d'une période de temps déterminée, que l'état de la sortie (**OUTPUT**) passe un niveau haut (**HIGH**) et termine la fin du cycle de « lecture » de l'interrupteur avec une grande efficacité.

Dans les années 70 et 80, les circuits intégrés anti-rebond n'existaient pas, et les microcontrôleurs étaient réservés aux professionnels, à l'époque on utilisait

à celle de l'alimentation, en sortie les signaux auront un niveau logique **TTL**.

En effet, toujours d'après les spécifications techniques, le **CD4050** est un **convertisseur CMOS** de « **haut à bas niveau logique** », il a la capacité de diminuer le niveau de la tension présente sur l'entrée ;

d) maintenant, voyons la situation inverse (schéma « d »), c'est à dire l'interface d'une sortie **TTL** avec une entrée **CMOS**. Nous avons vu que le problème peut se produire lorsque un niveau logique « **HIGH** » (haut) se présente, parce qu'un courant de sortie élevé peut abaisser la tension de **2,7 V** à **2,4 V**, valeur qui n'est pas reconnue sur l'entrée du **CMOS** comme un niveau « **HIGH** » (haut). Dans ce cas, la solution simple est d'ajouter une résistance de pull-up (tirage) de 1 kΩ, qui permet d'élever le niveau à une valeur supérieure à **3,5 V**, et donc reconnue par le **CMOS**. Cet ajout n'est pas nécessaire lorsque le circuit **TTL** n'a pas à fournir en sortie un courant élevé, le niveau « **HIGH** » (haut) en sortie reste aux alentours des 4 V, reconnu sans problèmes ;

e) le schéma « e » du bas de la figure 2 est l'équivalent du schéma « c » du haut, mais dans des positions inversées. Dans ce cas également, nous avons recours à un circuit « **buffer** » (tampon) **TTL**, un **SN7407** dont les spécifications techniques correspondent à un convertisseur de niveaux **TTL** à des niveaux **CMOS**. Le « **buffer** » (tampon) doit être alimenté obligatoirement en 5 V, c'est un **TTL**, mais dispose de sorties à « collecteur ouvert » qui supportent des tensions plus élevées provenant de la **résistance de pull-up** 10 kΩ ;

f) ce schéma est une variante du précédent (e), parce que le **buffer** est remplacé par un transistor NPN

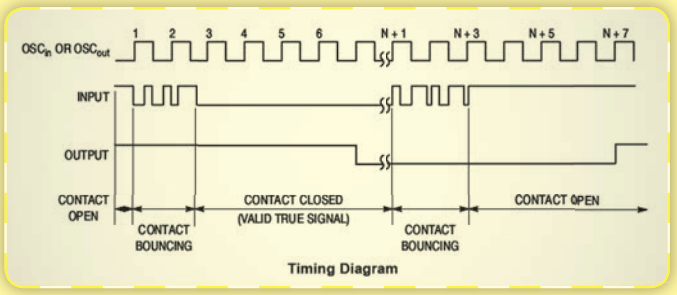


Figure B : comportement d'un circuit anti-rebond (debounce).

différentes techniques pour résoudre le problème d'une manière acceptable, parce que la perfection est atteinte que très rarement et que le vieillissement et l'oxydation progressive des contacts des interrupteurs ne faisaient qu'augmenter les problèmes. Regardons quelques solutions possibles en figure C :

- l'une des plus utilisées (voir la figure C - a)) est basée sur une bascule asynchrone de type S-R (**SET-RESET** généralement réalisée avec deux portes NAND), qui est une configuration qui change d'état la sortie si un niveau bas (**LOW**) arrive sur l'entrée « **SET** » ignorant tous les autres, et change à nouveau d'état si un niveau bas arrive sur « **RESET** ».

- Mais cela signifie que l'on doit piloter deux broches (pin) différentes, donc on doit utiliser un interrupteur à deux positions. Dans la position de repos, il maintient un niveau bas (**LOW**) sur « **RESET** », et lorsqu'on appuie sur l'interrupteur il porte à un niveau bas (**LOW**) l'entrée « **SET** », provoquant le changement d'état de la sortie. En le relâchant il met « **RESET** » à un niveau bas (**LOW**) et ainsi de suite. Le procédé est tout à fait fiable, mais il n'est pas toujours possible d'utiliser un inverseur à la place d'un bouton poussoir par exemple.

- une variante, la plus efficace de loin (voir la figure C -b)), est de relier un poussoir à l'entrée « **SET** » et, en même temps, ajouter un multivibrateur astable (tels qu'un **SN74121** ou un **NE555**) utilisé comme temporisateur, dont la sortie est reliée à l'entrée « **RESET** » de la bascule. Dans la pratique, en appuyant sur le poussoir, la sortie est fixée et débute le décompte d'une période de temps suffisamment longue pour dépasser la phase du rebond (typiquement de l'ordre de 100 ms), à la fin de laquelle le multivibrateur donne l'impulsion à l'entrée « **RESET** ».

- une solution simple et très efficace, est d'utiliser deux portes NON de type **Trigger de Schmitt**, avec une résistance de pull-up, un condensateur relié à la masse et le bouton en parallèle sur le condensateur. Le circuit RC ainsi réalisé détermine la durée de la période au cours de laquelle les impulsions parasites sont ignorées, la formule du calcul de la période est la suivante (voir la figure C – c)) :

T(msec) = R(kΩ) x C(µF) x 0,9.

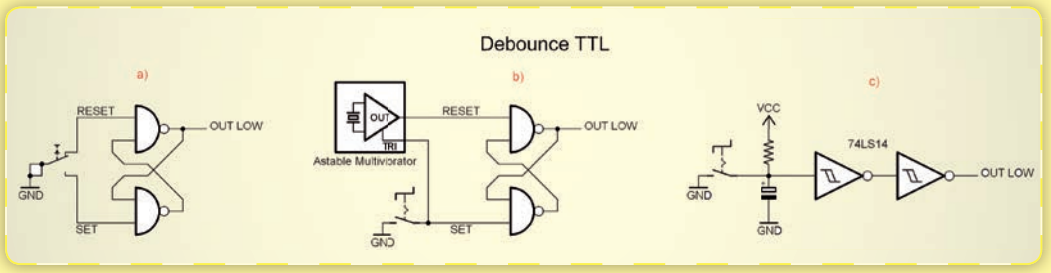


Figure C : les différentes solutions anti-rebond (debounce) à base de circuits TTL utilsées dans les années 70 - 80.

monté en configuration « émetteur commun », le transistor fonctionne alors en régime de saturation (passant)/blocage (non passant).

Dans le cas où le transistor est saturé (passant), un niveau de tension proche de 0 V, c'est-à-dire un niveau bas (**LOW**) se trouve sur son collecteur et donc à l'entrée du **CMOS.** Dans le cas où le transistor est bloqué, la tension présente sur son collecteur est proche de celle de l'alimentation, un niveau haut (**HIGH**) se présente sur l'entrée du **CMOS** en raison de la résistance de 10kΩ de pull-up. Cependant les signaux sont inversés sur l'entrée du **CMOS**.

Les caractéristiques techniques des circuits intégrés de notre projet

A ce stade, nous pouvons dire que nous avons montré une image globale, d'un point de vue théorie et pratique, du monde des familles logiques **TTL** et **CMOS**. Puisque cet article et ce projet ont pour but d'être éducatifs, nous devons vous mettre en condition pour bien comprendre le schéma complexe que nous allons vous présenter. Nous avons donc pensé que le meilleur moyen était de vous expliquer la fonctionnalité de chaque circuit intégré utilisé dans notre projet, et de voir ensuite les applications dans le schéma électrique du circuit.

C'est pour cette raison que nous avons préparé deux extraits des documentations techniques qui résument les caractéristiques essentielles de ces circuits intégrés (voir les figures 3 et 4). Comme vous le verrez, nous avons signalé pour chaque circuit intégré les brochages avec les symboles logiques et les tables de vérité. Il est temps de comprendre ces documentations. Commençons par la figure 3, où nous avons représenté les caractéristiques de 4 circuits **TTL** de la famille « **LS** » :

- **SN74LS00** : le symbole montre qu'il s'agit d'un circuit contenant **4 portes NAND** (**NON-ET**) à **2 entrées**, par exemple la première porte comporte

sur les broches 1 et 2 les entrées et sur la broche 3 la sortie. La caractéristique principale de la porte NAND (NON-ET) est de maintenir la sortie à l'état haut (**HIGH**) lorsqu'au moins une des deux entrées est à un niveau bas (**LOW**), indépendamment de l'état de l'autre entrée.

Cette caractéristique est facilement comprise par la lecture de la formule de la porte : **Y = (A x B)**. Notez que sur la figure 3 **vous voyez la formule avec une barre horizontale au-dessus du produit A x B, c'est le symbole nommé « barre » et qui signifie que le résultat de la multiplication doit être inversé**. Dans notre cas, nous représentons la même chose sous le symbole « **'** » qui est placé en dehors des parenthèses pour indiquer précisément que cela s'applique à l'ensemble du résultat.

Pour donner un exemple, si l'on applique à l'entrée **A** un **signal** de **niveau bas** (**LOW** = **0**) et à l'entrée **B** un **signal** de **niveau haut** (**HIGH** = **1**), nous obtenons **Y = 0 x 1 = 0**, mais le résultat doit être inversé donc **Y = 1**. Cela est très bien représenté dans la « **Table de vérité** » (dans la documentation elle se nomme aussi « Table de fonction »).

Dans la « **Table de vérité** » le symbole « **H** » représente le niveau haut « **HIGH** » ou **1**, le symbole « **L** » représente le niveau bas « **LOW** » ou **0**, le symbole « **X** » représente un état indéfini qui n'a pas d'influence, ce qui signifie que l'état de cette entrée peut être soit haut ou bas, sans qu'elle influe en aucune façon sur la sortie.

En effet, en partant de ce que nous avons défini comme caractéristique particulière, à savoir qu'il suffit d'avoir une seule entrée à un niveau bas « **LOW** » pour avoir toujours la sortie à un niveau haut « **HIGH** », il est évident que l'état de l'autre entrée ne peut avoir aucune influence, et c'est dans ce sens que nous avons la deuxième et la troisième ligne dans la « **Table de vérité** ».

Voyons maintenant la première ligne de la « **Table de vérité** », dans ce cas les deux entrées sont à un niveau haut « **HIGH** » et la sortie se trouve à un niveau bas « **LOW** », d'où **Y = 1 x 1 = 1** qui doit être inversé pour devenir **0**.

De tout cela, nous pouvons en déduire le comportement intéressant de la porte **NAND** (NON-ET) : si l'une des entrées est à un niveau bas « **LOW** » alors la sortie est fixée à un niveau haut « **HIGH** », mais si l'une des entrées est à un niveau haut « **HIGH** », la sortie prend un état opposé à l'entrée, indépendamment de l'état des deux entrées. Par exemple, si un signal à une certaine fréquence se présente sur une des deux entrées, tandis que l'autre entrée est fixée à un niveau haut « **HIGH** », la sortie prendra le niveau logique inverse « **LOW** », cela sera approfondi lorsque nous étudierons le schéma électrique.

- **SN74LS14** : ce circuit se compose de **six portes NON** (on dit aussi « **NOT** ») également appelé **INVERSEUR** (on dit aussi « **INVERTER** ») de type « **Trigger de Schmitt** ». La **porte NON inverse l'état du signal d'entrée logique**, si nous appliquons à l'entrée un niveau bas « **LOW** », en sortie nous trouverons un niveau haut « **HIGH** », et vice versa. Une porte de type « **Trigger de Schmitt** » a la particularité d'être insensible aux perturbations, car si une porte normale reconnaît tous les signaux de niveau haut « **HIGH** » dépassant le seuil $\mathbf{V_{IH}}$, ce type de porte le reconnaît comme tel qu'après une transition du signal en-dessous du seuil $\mathbf{V_{IL}}$.

L'exemple suivant vous aidera à mieux comprendre ce mécanisme. Imaginez que sur une porte normale **TTL** « **LS** » (n'oubliez pas que les valeurs de seuil sont pour **LOW ≤ 0,8 V** et pour **HIGH ≥ 2,0 V**) arrivent dans l'ordre les tensions suivantes : 3 V - 0.5 V - 3 V - 1.5 V - 3 V. Ces tensions seront traduites en niveaux logiques par la séquence « **HIGH** », « **LOW** », « **HIGH** », « **zone indéterminée** », « **HIGH** ».

Dans la pratique, vous aurez 3 signaux de niveau haut et un signal de niveau bas et une zone indéterminée qui se situe entre les deux tensions de 3 V. Par conséquent, elles seront vues et comptabilisées comme deux impulsions séparées de niveau haut « **HIGH** ».

Dans le cas d'une porte de type « **Trigger de Schmitt** », les mêmes tensions seront traduites par la séquence: « **HIGH** », « **LOW** », « **HIGH** ». La tension de la « zone indéterminée » étant inférieure au seuil $\mathbf{V_{IL}}$, elle sera ignorée, le groupe 3 V - 1,5 V - 3 V sera vu et comptabilisé comme un seul niveau haut « **HIGH** ». Ce type de problème peut se produire lorsqu'on interface une entrée d'une porte logique avec des composants discrets.

Dans notre projet, nous avons utilisé un bouton poussoir qui, comme tout élément mécanique, tend à provoquer un « **rebond** » (« bounce » en anglais). Si vous regardez à l'oscilloscope l'entrée de la porte logique qui est reliée au poussoir, à la fermeture du contact vous verrez une série de pics parasites que la porte logique interprète comme une série de signaux haut (« **HIGH** ») et bas (« **LOW** »), au lieu d'un seul.

Une porte « **Trigger de Schmitt** » permet de résoudre ce problème, contrairement à une porte standard, une porte « **Trigger de Schmitt** » n'a pas 1 seuil de basculement, mais 2 seuils de basculement.

La fonction « **Trigger de Schmitt** » est utilisée pour traiter les signaux à fronts lents, et permet une mise en forme de ces signaux. La qualité des signaux à traiter par les circuits logiques est souvent médiocre. Leur forme les rend inexploitables par les circuits intégrés.

Le « **Trigger de Schmitt** » permet d'obtenir des créneaux de forme régulière, aisément exploitables. Les entrées de nombreux circuits logiques comportent un trigger pour effectuer la mise en forme des signaux qui leur sont appliqués. Cela sera également approfondi lors de l'étude du schéma électrique.

- **SN74LS42** : ce circuit est un **décodeur BCD** (« **Binary Coded Decimal** » qui peut se traduire en français par « **décimal codé binaire** » dont les nombres sont représentés en chiffres décimaux et chacun de ces chiffres est codé sur quatre bits) qui accepte sur les **4 entrées une combinaison d'autant de bits et selon laquelle il active une seule de ses 10 sorties**, en la positionnant sur « **LOW** ».

Les entrées sont dénommées **A0** à **A3**, tandis que les sorties sont dénommées **0'** à **9'**, le symbole « **'** » indique l'état **inversé** de la **sortie**, à savoir que lorsque l'état actif est bas (« **LOW** »),

toutes les sorties à l'exception de celle qui est active, sont à un état haut (« **HIGH** »).

En regardant la « **Table de vérité** », vous pouvez facilement comprendre que c'est en fait une **conversion binaire → décimale**. En considérant que **A0** est le bit le moins significatif et que **A3** est le bit le plus significatif, vous pouvez voir immédiatement que chaque ligne, en fonction de la valeur binaire, active la sortie décimale correspondante.

Par exemple, la troisième ligne présente un niveau haut « **H** » en position **A1**, en le considérant comme second bit à partir du bit le moins significatif, ce bit vaut « 2 », en effet l'unique sortie active « **LOW** » sur la même ligne correspond à **2** « **barre** » ou **2'**.

Un autre exemple, prenons la huitième ligne et nous voyons que les trois premiers bits, toujours en commençant par le bit le moins significatif, sont tous à un niveau haut « **H** », le quatrième est à un niveau bas « **L** », cela signifie d'un point de vu binaire 1 + 2 + 4 = 7, nous observons que la sortie active « **LOW** » correspond à 7 « barre » ou 7'.

Notez que pour gérer les 10 sorties, le circuit fait appel à 4 bits qui permettent d'obtenir 16 combinaisons. Les combinaisons 10 à 15 n'activent pas de sortie, elles sont ignorées.

- **SN74LS74** : ce circuit est un « **Flip-Flop** » ou **bascule synchrone avec deux éléments de type** « **D** », un « **Preset** » et un « **Clear** ». Traiter de la théorie des **bascules asynchrones** (verrous ou en anglais latch) et des **bascules synchrones** (bascule en anglais Flip-Flop) serait extrêmement intéressant, mais il faudrait trop de place et nous serions hors sujet.

Nous allons vous expliquer l'usage de ces deux types de circuits intégrés dans le cadre de notre projet. Chaque circuit dispose de **4 entrées** (**PR**, **CLR**, **CLK**, **D**) et **2 sorties** (**Q**, **Q'** ou **Q** « barre »), pris individuellement, chacun d'eux peut fonctionner comme un « **Latch synchrone D** » ou comme « **Latch asynchrone Set-Reset** ». Pour comprendre ces deux circuits, vous devez vous référer attentivement à la « **Table de vérité** » :

Photo de l'un de nos prototypes du capacimètre décrit dans cet article, vu de l'intérieur et de la face avant décrit dans cet article. Vous pouvez voir les circuits logiques, il ne nécessite pas de microcontrôleur, ni de programme spécifique.

- pour mettre en place le mode « **Latch synchrone D** », vous devez mettre les broches d'entrée « **PR** » (« Preset ») et **CLR** (« Clear ») à un niveau haut « **HIGH** », la partie de la « **Table de vérité** » qui nous intéresse dans ce cas comprend les 3 dernières lignes, les signaux en « **front montant** » sur l'entrée **CLK** (Horloge) et les niveaux présents sur la broche « D ».

Dans la quatrième ligne le symbole « ↑ » représente un « **front montant** », c'est à dire le moment où le niveau sur l'entrée **CLK** passe d'un état bas (**LOW**) en un état haut « **HIGH** ». S'il y a un état haut « **H** » sur la broche D, la sortie Q prendra un état haut « **H** » et la sortie Q' ou Q « barre » présentera un état logique opposé c'est-à-dire bas « **L** ».

Si par contre (ligne 5) au moment où se produit le « **front montant** » (« ↑ »), un niveau logique bas « **L** » se présente sur la broche D, la sortie Q sera également à un état bas, et la sortie Q' ou Q « barre » présentera un état logique opposé c'est-à-dire haut « **H** ».

Au moment où l'état logique de la broche **CLK** passe à un niveau bas « **L** » (ligne 6), les états des sorties Q et Q' (Q « barre ») ne changent pas et restent fixés par rapport au moment précédent. En résumé, lorsque l'entrée **CLK** passe d'un état bas « **L** » à un état haut « **H** », l'état logique qui se trouve sur l'entrée D est reporté identiquement sur la sortie Q et inversé sur la sortie Q'.

- en ce qui concerne le mode « **Latch asynchrone Set-Reset** », l'état des 2 entrées « **CLK** » et « **D** » importe peu (état « X » qui veut dire « indéfini »), les lignes qui nous intéressent sont les trois premières.

A la première ligne, nous voyons que si l'état de l'entrée « **PR** » est à un niveau bas « **L** », et que l'entrée « **CLR** » est à un niveau haut « **H** », alors la sortie Q est à un état haut « **H** » et Q' (Q « barre ») est à un état bas « **L** ». Inversement, si « **PR** » est à un état haut « **H** » et « **CLR** » à un état bas « **L** », alors Q = L et Q ' = H. Mais que se passe-t-il si « **PR** » et « **CLR** » se trouvent à un même niveau logique ? Si le même niveau logique est haut « **H** », nous avons déjà vu ce cas, c'est le mode « **Latch synchrone D** » dont le comportement de la bascule dépend des deux entrées « **CLK** » et « **D** ».

D'autre part, si le même niveau logique est bas « **L** », la documentation du constructeur parle alors d'une configuration instable, ce cas est à éviter.

Passons maintenant à la figure 4, dans laquelle nous avons décrit 5 circuits **TTL**, 3 « **LS** », un « **standard** » et un « **HCT** ».

- **SN74LS90** : ce circuit est un **compteur BCD décade ou compteur BCD « MODULO 10 »**, c'est-à-dire qu'il compte en base 10. Ce circuit est constitué de 2 sections, une section « **diviseur par 2** » et une section « **diviseur par 5** ». La première section « diviseur par 2 » comporte une bascule dont l'entrée d'horloge **CKA** se trouve sur la broche 14 et la sortie **QA** sur la broche 12. La deuxième section du circuit comporte 3 bascules dont l'entrée d'horloge **CKB** est sur la broche 1 et les trois sorties **QB**, **QC**, **QD** sur les broches 9, 8, 11. La troisième sortie **QD** délivre un **signal divisé par 5** par rapport au **signal d'horloge appliqué à l'entrée de l'horloge CKB**.

Ces sections peuvent être utilisées séparément pour obtenir une division de la fréquence d'entrée par 2 ou 5, ou être branchées en cascade (sortie QA reliée à l'entrée CKB) pour obtenir sur la sortie QD une fréquence divisée par 10 présente à l'entrée de CKA. Dans la pratique, l'application d'un signal de 10 MHz sur l'entrée CKA, avec les étages connectés en cascade, se traduit sur la sortie QD par un signal d'une fréquence de 1MHz.

De plus, ce circuit peut également effectuer un codage binaire d'impulsions comptabilisées sur l'entrée, que vous pouvez voir dans la « **Table de vérité** » appelée « BCD Count Sequence » ou séquence de comptage BCD. Pour chaque impulsion présente sur l''entrée, l'état des 4 sorties change de « LLLL » (0) à « HLLH » (9), avec QA représentant le bit le moins significatif et QD le plus significatif.

Dans ce cas, les sorties sont au début à « LLLL », la première impulsion fait passer les sorties à « LLLH », la seconde impulsion à « LLHL » et ainsi de suite, lorsque l'entrée atteint la dixième impulsion, les sorties reviennent à « LLLL ».

Précisons que le symbole, représenté par la barre horizontale et le rond visible dans le schéma logique (figure 4) sur la broche 14 et 1, signifie qu'il s'agit d'entrées actives à l'état bas (« **LOW active** »), c'est à dire que le comptage s'effectue au moment où le signal passe de l'état haut « **H** » vers l'état bas « **L** ».

Un rôle fondamental, dans l'utilisation de ce circuit, est joué par les 2 unités de commande R0 et R9. Ce sont deux portes **NAND** qui contrôlent la remise à zéro (**RESET**) des deux étages diviseurs (voir « **Table de vérité** » nommée « **Reset/Count Truth Table** »). En particulier, la porte **NAND R0**, dont les entrées sont sur les broches 2 et 3, contrôle le « **RESET** » de l'étage « **diviseur par 5** ».

Si ces deux entrées sont placées sur un état haut « **H** », les sorties du circuit sont mises à la valeur 0 (« LLLL »), d'où le nom de **NAND R0** (**R0 = RESET 0**). Toutefois il est nécessaire que la porte NAND R9 soit désactivée (sortie sur « **H** »), pour atteindre cet état. Au moins une des broches 6 et 7 doit être à un niveau « **L** ».

Par contre la porte NAND R9 contrôle le « **RESET** » de l'étage « diviseur par 2 », lorsque ses deux broches d'entrée 6 et 7 sont mises à un niveau « **H** », la porte NAND porte la sortie à un niveau « **L** » ce qui active le « **RESET** » de l'étage diviseur en fixant la sortie du circuit à la valeur 9 (« HLLH »), dans ce cas, l'état de l'entrée R0 de l'autre porte **NAND** n'a pas d'influence. Ces deux commandes de « **RESET** » sont utilisées pour faire fonctionner le circuit comme un codeur binaire. Généralement on utilise une seule des deux commandes « **RESET** », selon la façon dont on veut régler la condition de départ (« LLLL » ou « HLLH »), activant le moment où l'on souhaite réinitialiser le compteur.

- **SN74LS390** : nous pouvons dire globalement que ce circuit intégré est en réalité tout simplement un double SN74LS90, car il intègre deux paires d'étages diviseurs par 2 et par 5. Il peut être utilisé de la même manière que le SN74LS90. Le brochage du circuit est différent, mais la signification est la même : **CP0' = CKA**, **CP1' = CKB**, **Q0** à **Q3 = QA** à **QD**.

Le symbole « ' » appliqué à CP0 et CP1 a le même sens que la barre horizontale et le rond vu précédemment pour CKA et CKB, c'est à dire que ces entrées sont actives sur le « front descendant » du signal (passage de « **H** » à « **L** »).

Pour identifier le brochage des deux paires d'étages diviseurs, reportez-vous au diagramme fonctionnel du **SN74LS390** de la figure 4. La première paire est reliée aux broches 1 à 7 et la seconde aux broches 9 à 15 (chiffres entre parenthèses). L'unique différence par rapport au **SN74LS90** se situe dans la gestion du « **RESET** ».

Dans ce cas il existe, pour chaque paire d'étage une seule broche nommée MR (Master Reset) qui va sur une porte NOT. Lorsque la broche est placée sur un état « **H** », les étages sont remis à zéro (« **RESET** ») et les sorties correspondantes verrouillées sur « LLLL ». La « **Table de vérité** » nous montre que la broche **M** doit être à un niveau « **L** » pour que les étages fonctionnent.

A la première ligne, nous voyons que le « **front montant** » (↑) n'a pas d'influence sur le compteur, la deuxième ligne indique que le « front descendant » (↓) est pris en compte, et enfin la troisième ligne confirme que, lorsque la broche **MR** est à un niveau « **H** » (condition de « **RESET** » général des étages) l'état des entrées n'a pas d'influence (X = indéfini).

- **SN74LS151** : c'est un **multiplexeur 8 vers 1 fonctionnant en logique binaire**. Ce circuit dispose de 8 entrées de données D0 à D7 (broches 4, 3, 2, 1, 15, 14, 13, 12), deux sorties complémentaires Q et Q « barre » (une directe et l'autre inversée) dénommée Y et W' (broches 5 et 6), trois entrées de sélection A, B et C (broches 11, 10, 9) et une entrée de validation dénommée « **Strobe G'** » (broche 7).

Le fonctionnement de ce circuit est très simple et bien illustré dans la « **Table de vérité** ». Lorsque l'entrée de validation « Strobe G' » (active sur « **L** ») est à un niveau « **H** » (1), la sortie Y passe à un état « **L** » et la sortie W' à un état opposé « **H** », tandis que l'état des entrées A, B, C n'a pas d'influence. Si au contraire l'entrée de validation « Strobe G' » est fixée à un état « L » (0), alors le multiplexeur

fonctionne conformément à la combinaison des entrées de sélection C, B, A (l'entrée A représente le bit le moins significatif). Par exemple, si la combinaison logique des entrées de sélection est « HLH » (décimale 5), l'état du signal présent sur l'entrée D5 est reporté de manière identique sur la sortie Y et inversé sur la sortie W'.

- **SN74121** : c'est un **multivibrateur monostable** qui appartient à la famille originale **TTL** « **standard** », **il n'existe pas dans la version** « **LS** », ou plutôt sous le modèle **SN74LS122**, avec des différences mineures par rapport à la version 121. Nous pourrions utiliser en toute sécurité ce dernier, mais nous avons préféré vous montrer comment interfacer sans problèmes des sous-familles différentes.

Cette approche a été largement utilisée dans les circuits **TTL**, dont la plupart sont toujours disponibles auprès des distributeurs de composants électroniques. Il peut être piloté aussi bien avec des « fronts montants » qu'avec des « fronts descendants », en fonction de la configuration des entrées A1, A2, B (broche 3, 4, 5), qui sont insensibles aux perturbations car ces entrées sont de type « **Trigger de Schmitt** ». Il possède deux sorties complémentaires Q et Q' (Q « barre »), et est capable de générer des fréquences allant de **0,035Hz** à **35MHz**, donc une impulsion à partir de **28 ns** (35MHz) jusqu'à **28 s** (0,035Hz). La fréquence générée, et par conséquent la durée de l'impulsion en sortie, dépend des valeurs des composants R-C (résistance et condensateur) connectés aux broches Rext/Cext, Cext et Rint (broches 11, 10, 9).

Dans la pratique, vous pouvez choisir d'utiliser la résistance interne Rint de 2 kΩ (elle est à l'intérieur du circuit) ou une résistance externe Rext dont la valeur doit être calculée en fonction de la capacité du condensateur qui est connecté à la broche Cext (broches 10 et 11). La largeur de l'impulsion générée en sortie est calculée selon la formule suivante :

Tw = K x R x C

où **K** est une constante égale à 0,7, **R** est exprimée en ohms et **C** est exprimée en farads,

Tw est le résultat exprimé en secondes. Le principe de fonctionnement est très simple, en l'absence de circuit RC les sorties sont verrouillées sur Q = L et Q' = H. En présence du circuit RC, et avec les entrées A1, A2, B placées dans l'une des combinaisons prévues, la sortie Q présente une impulsion positive de durée égale à la période de charge du condensateur et la sortie Q' présente une impulsion négative de durée égale.

- **SN74HCT4511** : c'est le seul circuit intégré **CMOS** utilisé dans notre projet dans la version « **HCT** ». Comme le montre le **Tableau 1**, ce circuit a des niveaux logiques compatibles **TTL**, et il est parfaitement interfaçable avec un circuit **TTL** de type « **LS** ». Il s'agit d'un **décodeur BCD à 7 segments** composé d'entrées « latch » pour mémoriser des nombres et de circuits de commande pour le pilotage des afficheurs à LED 7 segments.

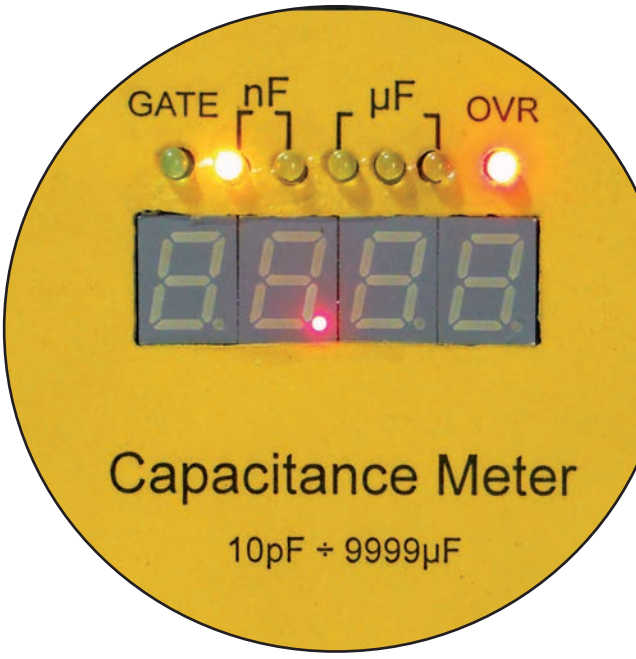
En pratique, en fonction de la combinaison binaire présente sur les entrées, les sorties sont activées pour afficher la représentation du nombre décimal correspondant. Par exemple, si sur les 4 entrées la combinaison suivante « LLHL » se présente, cela correspondra au nombre décimale « 2 », le circuit active les sorties « a-b-d-e-g» pour afficher le nombre « 2 ».

Puisque les 7 sorties de ce circuit sont actives sur un niveau « **HIGH** », il peut contrôler un affichage de type « cathode commune ». En plus des 4 entrées D3 à D0 (broches 6, 2, 1, 7) parmi lesquelles D0 correspond au bit le moins significatif, il est doté de 7 sorties nommées a à f (broches 9 à 15), et de 3 broches de contrôle :

- « **LT'** » (Lamp test, broche 3) permet d'allumer tous les segments de l'afficheur pour les tester ;

- « **BL'** » (BLanking, broche 4) permet d'éteindre entièrement l'afficheur ;

- « **LE'** » (« Latch Enable » c'est-à-dire « verrou actif », broche 5) permet de mémoriser la valeur présente sur les



entrées et de « geler » ainsi un chiffre sur l'afficheur même si les entrées sont modifiées.

L'entrée « **LT'** » (test de la lampe) est utilisée pour la vérification extemporanée (immédiate) de l'afficheur en la reliant à la masse (GND) de l'alimentation ou en la plaçant sur un état logique bas « **L** ». Instantanément toutes les sorties du circuit intégré sont activées provoquant l'allumage des 7 segments de l'afficheur (dans la pratique le nombre « 8 » s'affiche). Dans les conditions d'utilisation normale, la broche de l'entrée « **LT'** » doit être reliée en permanence au positif de l'alimentation.

Inversement, si l'entrée « **BL** » (blanking) est mise à un état bas « **L** », elle désactive toutes les sorties du circuit et éteint l'afficheur à 7 segments. Pour un fonctionnement normal du circuit, cette broche doit également être reliée à l'alimentation positive mais, comme nous le verrons dans notre projet, nous avons également prévu de la piloter via des niveaux logiques.

Si l'entrée « **Latch Enable** » est mise à un état bas « **L** », elle est inactive et n'a aucun effet sur l'affichage. Si par contre elle est placée à un état haut « **H** », elle bloque (fige) l'affichage dans une « position » à ce moment, la logique de commande de cette broche est essentielle pour un bon fonctionnement de notre circuit.

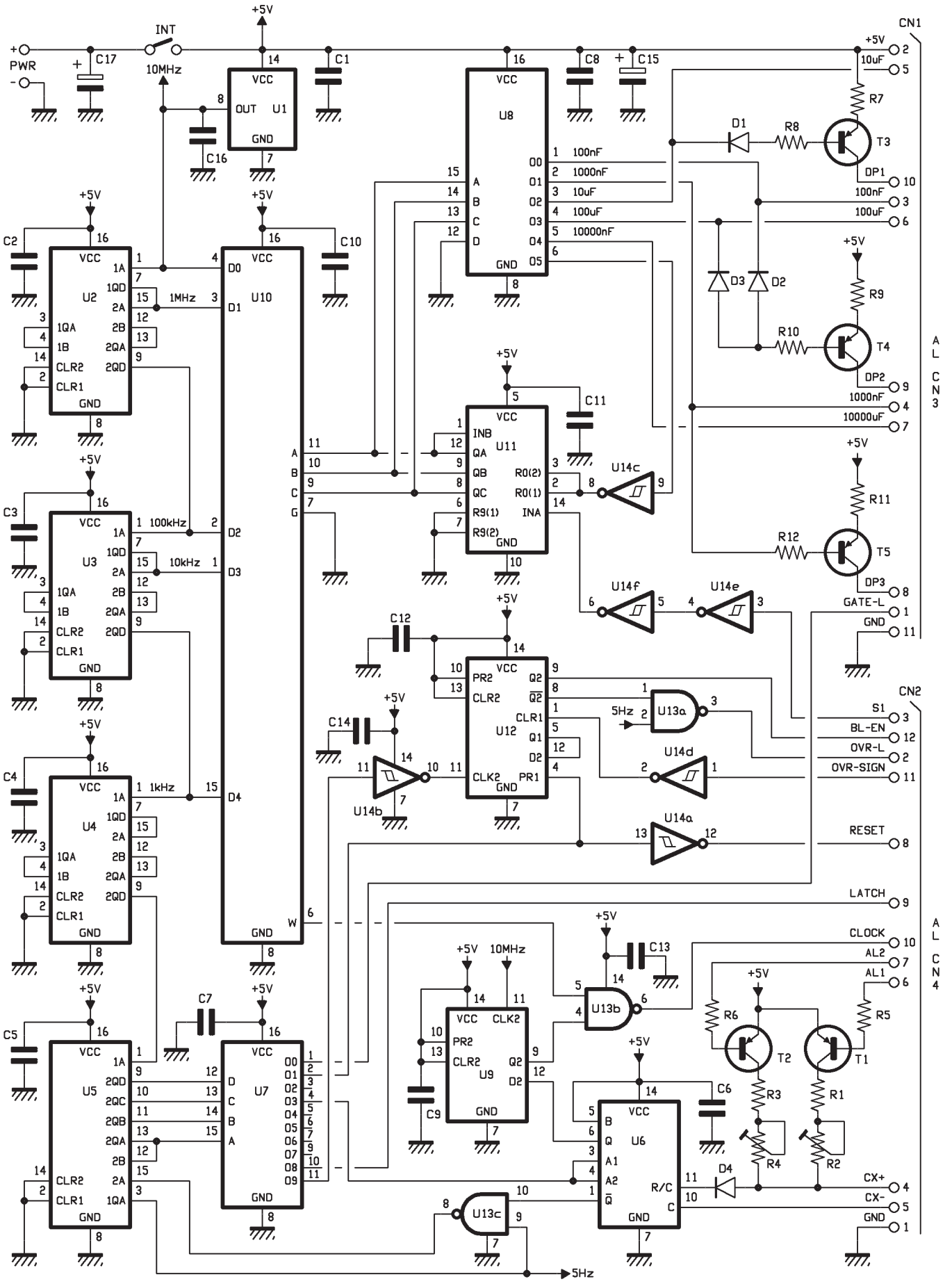


Figure 5 : schéma électrique de l'étage de la base de temps et de la logique.

Liste des composants de la base de temps et de la logique.

R1..... 1 kΩ 1%
R2..... Trimmer multitours 1 kΩ
R3..... 10 kΩ 1%
R4..... Trimmer multitours 10 kΩ
R5..... 1 kΩ
R6..... 1 kΩ
R7..... 220 Ω
R8..... 1 kΩ
R9..... 220 Ω
R10.... 1 kΩ
R11 ... 220 Ω
R12 ... 1 kΩ
C1..... 100 nF multicouche
C2..... 100 nF multicouche
C3..... 100 nF multicouche
C4..... 100 nF multicouche
C5..... 100 nF multicouche
C6..... 100 nF multicouche
C7..... 100 nF multicouche
C8..... 100 nF multicouche
C9..... 100 nF multicouche
C10.... 100 nF multicouche
C11.... 100 nF multicouche
C12.... 100 nF multicouche
C13.... 100 nF multicouche
C14.... 100 nF multicouche
C15.... 470 µF 16 V électrolytique
C16.... 15 pF céramique
C17.... 470 µF 16 V électrolytique
D1..... 1N4148
D2..... 1N4148
D3..... 1N4148
D4..... 1N4148
T1 ..... BC557
T2 ..... BC557
T3 ..... BC557
T4 ..... BC557
T5 ..... BC557
U1..... Oscillateur 10 Mhz TC46C
U2..... SN74LS390N
U3..... SN74LS390N
U4..... SN74LS390N
U5..... SN74LS390N
U5..... SN74LS390N
U6..... SN74121N
U7..... SN74LS42N
U8..... SN74LS42N
U9..... SN74LS74N
U10.... SN74LS151N
U11 ... SN74LS90N
U12 ... SN74LS74N
U13 ... SN74LS00N
U14.... SN74LS14N

INT..... interrupteur coudé 90°
PWR... prise alimentation pour ci
Barrette sécable femelle 11 et 12 pôles
Support circuits 2 x 8 (7 pièces) et 2 x 7 (6 pièces)

Caractéristique techniques

- Gamme de mesure : de 10 pF à 9999 µF
- Calibres : 5 sélectionnables par bouton poussoir :

1er calibre : de 0,01 nF (10 pF) à 99,99 nF
2eme calibre : de 0,1 nF (100 pF) à 999,9 nF
3eme calibre : de 0,001 µF (1 nF) à 9,999 µF
4eme calibre : de 0,01 µF (10 nF) à 99,99 µF
5eme calibre : de 1 µF à 9999 µF

- Visualisation de la mesure : affichage à 4 chiffres avec indication du calibre par LED en nF ou µF
- Indication par LED de la mesure, du calibre et de l'over-range (dépassement d'échelle)
- Alimentation 5 VDC externe stabilisée
- Courant consommé 600 mA.

Jusqu'à présent nous avons traité tous les aspects théoriques de l'utilisation des circuits logique **TTL** et **CMOS**, nous avons également étudié les fonctionnalités des différents types de circuits intégrés, appartenant aux familles **TTL Standard**, **TTL LS** et **CMOS HCT** que nous allons employer dans notre capacimètre numérique.

Pour bien comprendre les différentes parties du schéma électrique, nous vous conseillons de reprendre les explications du début de l'article.

Entrons dans le vif du sujet en vous reportant aux caractéristiques techniques de notre capacimètre qui sont très respectables, celui-ci vous permet de mesurer pratiquement tous les condensateurs disponibles sur le marché.

Pour ce projet, nous avons prévu deux sections différentes, chacune correspondant à des cartes disposées l'une sur l'autre, et chaque carte ayant son propre schéma électrique, nous pouvons par conséquent les étudier séparément.

Pour des raisons de simplification dans tout ce qui suit nous adoptons les abréviations suivantes :

- **H = HIGH (haut)**
- **L = LOW (bas)**
- **BTL = Base de temps et logique**
- **C&D = Compteurs et affichage.**

La base de temps

Une base de temps est un circuit qui, à partir d'une fréquence précise, généralement de grande valeur, effectue des divisions successives de manière à mettre à disposition du circuit une série de plusieurs fréquences multiples entre elles (mais pas nécessairement).

Sur la figure 5, vous pouvez voir le schéma électrique de la base de temps et de la logique. Dans notre cas, nous commençons avec une fréquence de 10 MHz qui est prélevée d'un oscillateur spécifique à quartz intégré dans un seul boîtier (U1). En fait il faudrait utiliser deux ou trois portes logiques, un circuit RC et un oscillateur à quartz pour recréer un montage style des années « 70 ».

Cela vous obligerait à effectuer un calibrage par compensation à l'aide d'un fréquencemètre ou d'un oscilloscope, nous avons opté pour ce composant qui assure une grande précision et qui ne nécessite aucune calibration.

Nous prévoyons que dans le circuit « C&D » nous faisons usage de certains composants de nouvelle génération, tout en respectant l'objectif du projet, mais indispensables pour obtenir certaines caractéristiques sans trop compliquer la réalisation pratique.

En sortie de la broche 8 (OUT) de U1, nous avons une onde carrée de

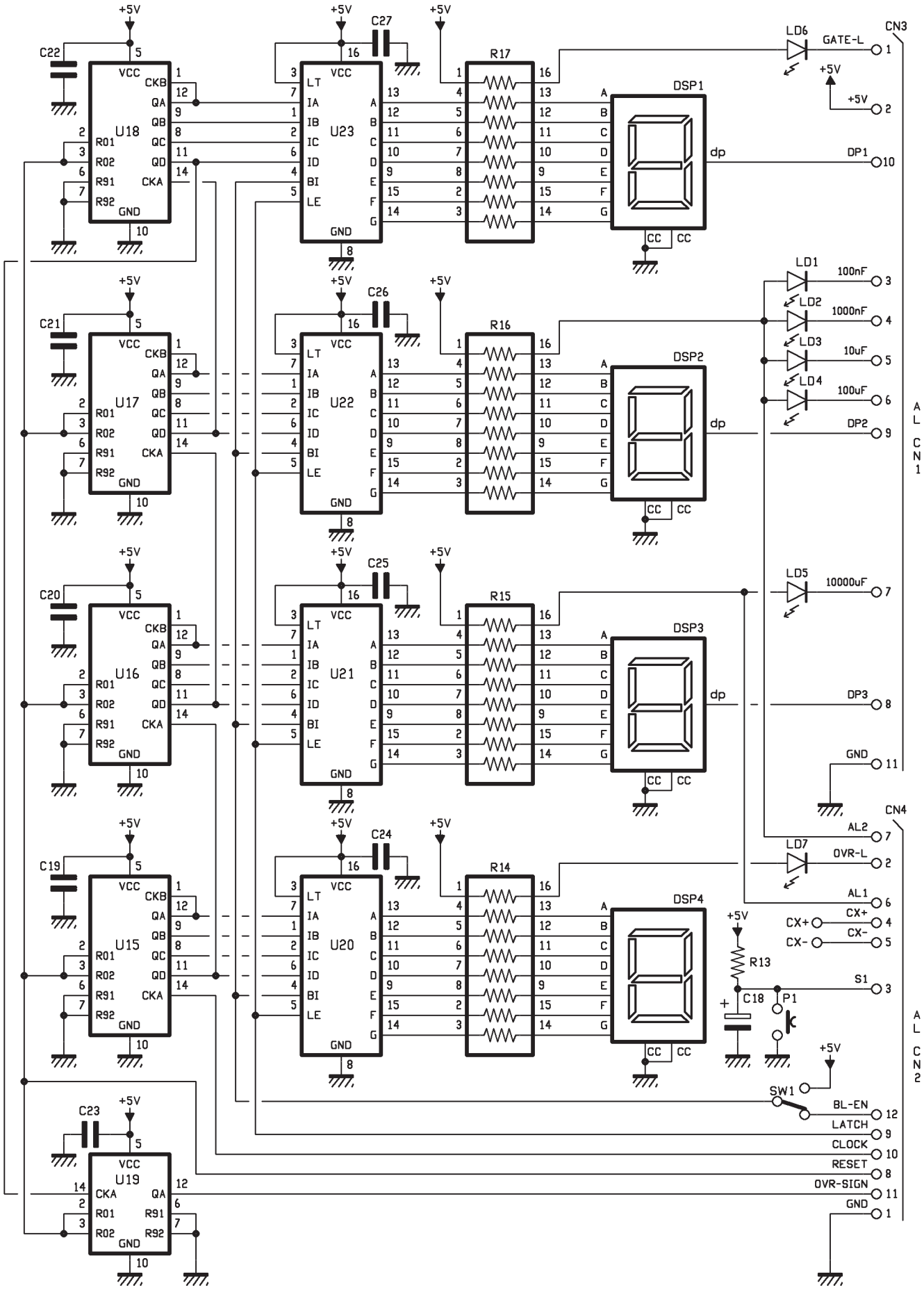


Figure 6 : schéma électrique de l'étage compteur et des afficheurs 7 segments.

Liste des composants du compteur et des afficheurs.

R13 ... 100 Ω
R14.... réseau de résistances 8 x 180 Ω SOIC
R15 ... réseau de résistances 8 x 180 Ω SOIC
R16.... réseau de résistances 8 x 180 Ω SOIC
R17.... réseau de résistances 8 x 180 Ω SOIC

C18.... 1µF 16 V tantale
C19.... 100 nF céramique
C20.... 100 nF céramique
C21.... 100 nF céramique
C22.... 100 nF céramique
C23.... 100 nF céramique
C24.... 100 nF céramique
C25.... 100 nF céramique
C26.... 100 nF céramique
C27.... 100 nF céramique

LD1.... LED 3mm jaune
LD2.... LED 3mm jaune
LD3.... LED 3mm jaune
LD4.... LED 3mm jaune
LD5.... LED 3mm jaune
LD6.... LED 3mm verte
LD7.... LED 3mm rouge

DSP1. Afficheur 7 segments à cathode commune (SC39-11SRWA)
DSP2. Afficheur 7 segments à cathode commune (SC39-11SRWA)
DSP3. Afficheur 7 segments à cathode commune (SC39-11SRWA)
DSP4. Afficheur 7 segments à cathode commune (SC39-11SRWA)

U15 ... SN74LS90N
U16.... SN74LS90N
U17.... SN74LS90N
U18 ... SN74LS90N
U19 ... SN74LS90N
U20 ... SN74HCT4511
U21.... SN74HCT4511
U22 ... SN74HCT4511
U23 ... SN74HCT4511

P1..... poussoir

SW1... interrupteur coudé 90°

Barrette sécable mâle 11 et 12 pôles

fréquence 10 MHz, de niveau logique compatible **TTL**. Les 7 étages suivants ont été réalisés à l'aide de 4 **SN74LS390**.

Chacune des deux paires d'étages diviseurs qu'il contient a été configurée comme un diviseur par 10, la fréquence à diviser arrive sur l'entrée A du premier étage, et sa sortie QA est connectée à l'entrée B du deuxième étage, la fréquence divisée par 10 est obtenue sur la sortie QD de cet étage.

Donc la fréquence de 10 MHz générée par U1 est appliquée à l'entrée A du premier étage contenu dans le **SN74LS390** (U2A).

A la sortie QD de U2A nous trouvons donc un signal dont la fréquence vaut 1MHz, qui à son tour est appliqué à l'entrée A du deuxième étage diviseur (U2B), à la sortie QD de celui-ci, nous obtenons une fréquence de 100 kHz.

Donc un seul **SN74LS390** est capable de réaliser une division par 100, avec une division intermédiaire par 10 utilisable.

De même, le circuit U3 effectue le même type de division avec un signal de 10 KHz sur sa première sortie QD (U3A) et de 1 KHz sur sa seconde sortie QD (U3B). U4 effectue à son tour deux divisions par 10 supplémentaires, la première (100Hz sur la sortie QD de U4A) et la deuxième (10Hz sur la sortie QD de U4B).

La situation est un peu différente pour le dernier circuit (U5) **SN74LS390** utilisé dans notre montage. Dans ce cas, seul le premier étage (U5A) est utilisé pour la base de temps, cet étage est configuré en diviseur par 2, et uniquement la sortie QA fournit une fréquence de 5 Hz. Il n'y a pas de connexion vers l'entrée B du deuxième étage, une division supplémentaire est inutile pour notre projet.

Cependant comme les étages du circuit **SN74LS390** sont indépendants l'un de l'autre, nous avons utilisé le deuxième étage (U5B) pour effectuer une fonction différente, comme nous le verrons plus tard.

Veuillez noter que tous les circuits **SN74LS390** ont la broche CLR (MR) reliée à la masse, c'est-à-dire qu'ils fonctionnent tout le temps et ne sont jamais remis à zéro.

A ce stade du schéma, nous avons un ensemble de fréquences : 10 MHz, 1 MHz, 100 kHz, 10 kHz, 1 kHz, 100 Hz, 10 Hz, 5 Hz, les fréquences 10 Hz et 100 Hz ne sont pas utilisées, mais ces divisions ont été nécessaires pour fabriquer la fréquence de 5 Hz, qui sera utilisée dans deux situations différentes.

Le multivibrateur astable, le cœur du circuit

Après la description de la base de temps, passons à la Logique de la mesure et du contrôle, réalisée autour du circuit multivibrateur astable U6, un SN74121, qui peut être considéré comme le cœur de notre capacimètre numérique.

Commençons par la description des entrées : B est directement reliée à l'alimentation donc à un niveau logique H, pendant que A1 et A2 ont un niveau logique dépendant de la sortie O3' de U7 (broche 4 du SN74LS42), et les sorties Q et Q' sont toutes les deux utilisées, nous verrons comment plus tard.

Les broches **R/C** et **C** (respectivement 11 et 10 de U6) méritent une attention particulière, ce sont les deux broches sur lesquelles sera connecté, par l'intermédiaire de douilles appropriées, le condensateur à mesurer dont la capacité est évidemment théoriquement inconnue.

Alors que la broche **C** correspond seulement à la connexion du pôle négatif dans le cas de condensateurs à mesurer de types polarisés, la broche **R/C** présente pour le raccordement de l'autre extrémité du condensateur (pôle positif si polarisé) une résistance qui est connectée à l'alimentation positive, et qui est utilisée pour charger le condensateur à mesurer.

En réalité, la broche **R/C** n'est pas directement reliée au circuit RC, mais par l'intermédiaire de la diode **D4**, tel que préconisé dans la documentation technique du **SN74121**, en cas de mesure de condensateurs électrolytiques (D4 évite

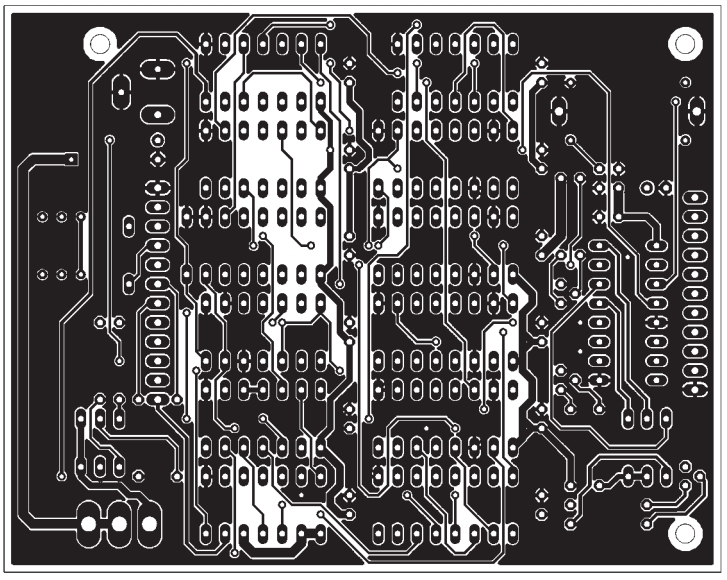


Figure 7 : circuit imprimé à l'échelle 1 : 1 de l'étage base de temps et logique du capacimètre côté soudures.

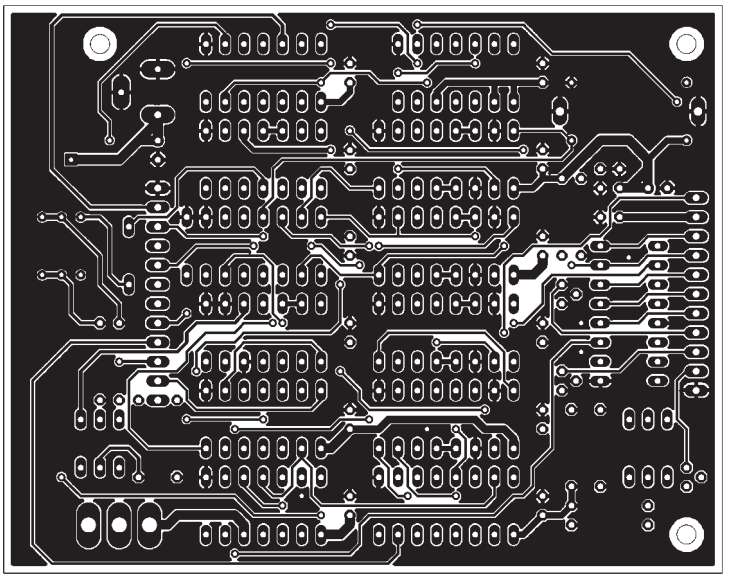


Figure 8 : circuit imprimé à l'échelle 1 : 1 de l'étage base de temps et logique du capacimètre côté composants.

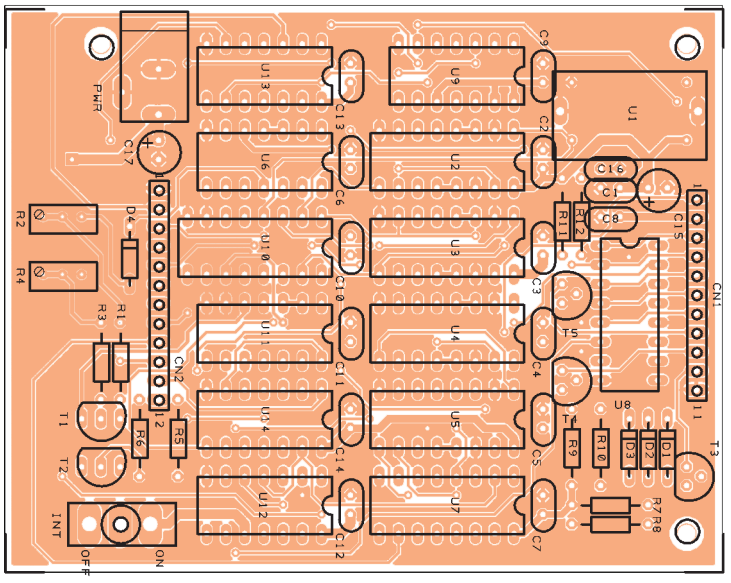


Figure 9 : schéma de câblage de l'étage base de temps et logique du capacimètre.

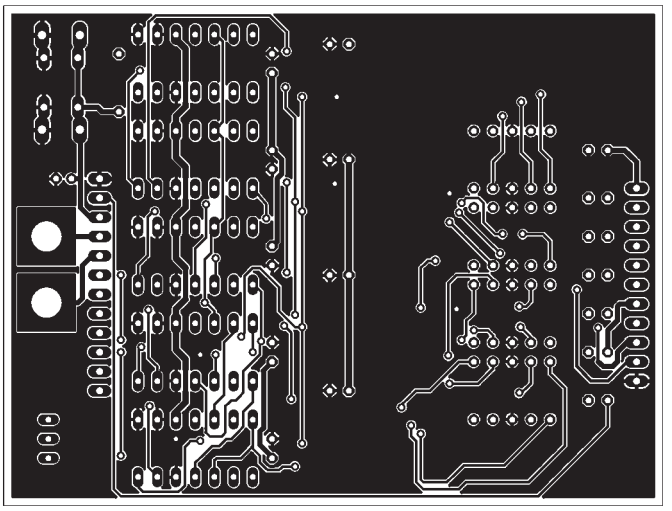


Figure 10 : circuit imprimé à l'échelle 1 : 1 de l'étage compteur et des afficheurs 7 segments capacimètre côté soudures.

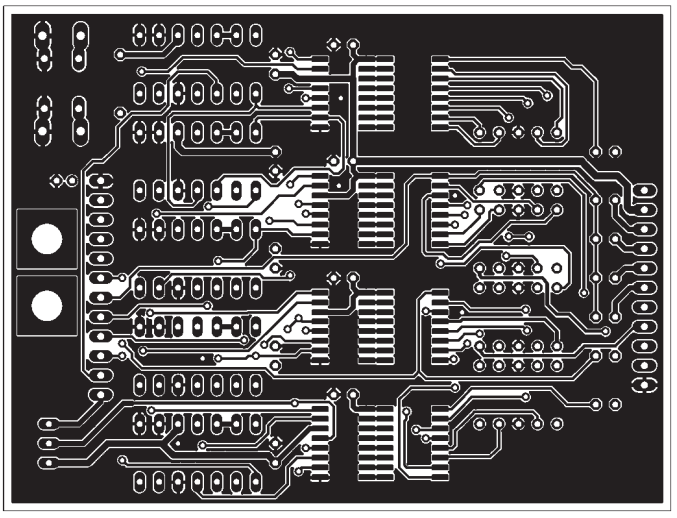


Figure 11 : circuit imprimé à l'échelle 1 : 1 de l'étage compteur et des afficheurs 7 segments capacimètre côté composants.

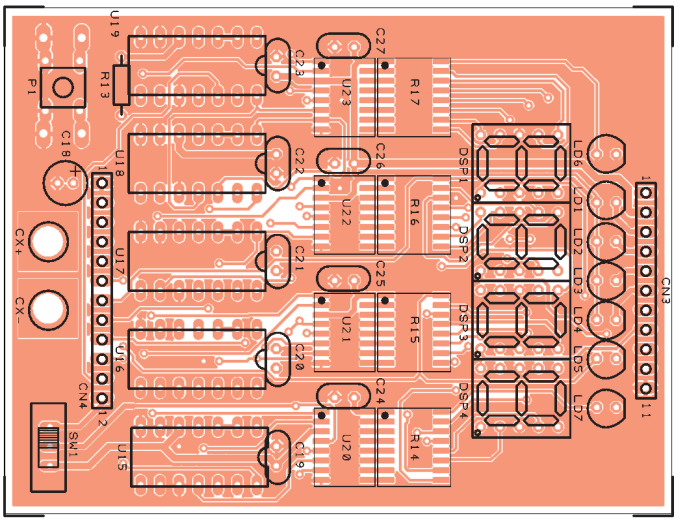


Figure 12 : schéma de câblage de l'étage compteur et des afficheurs 7 segments du capacimètre.

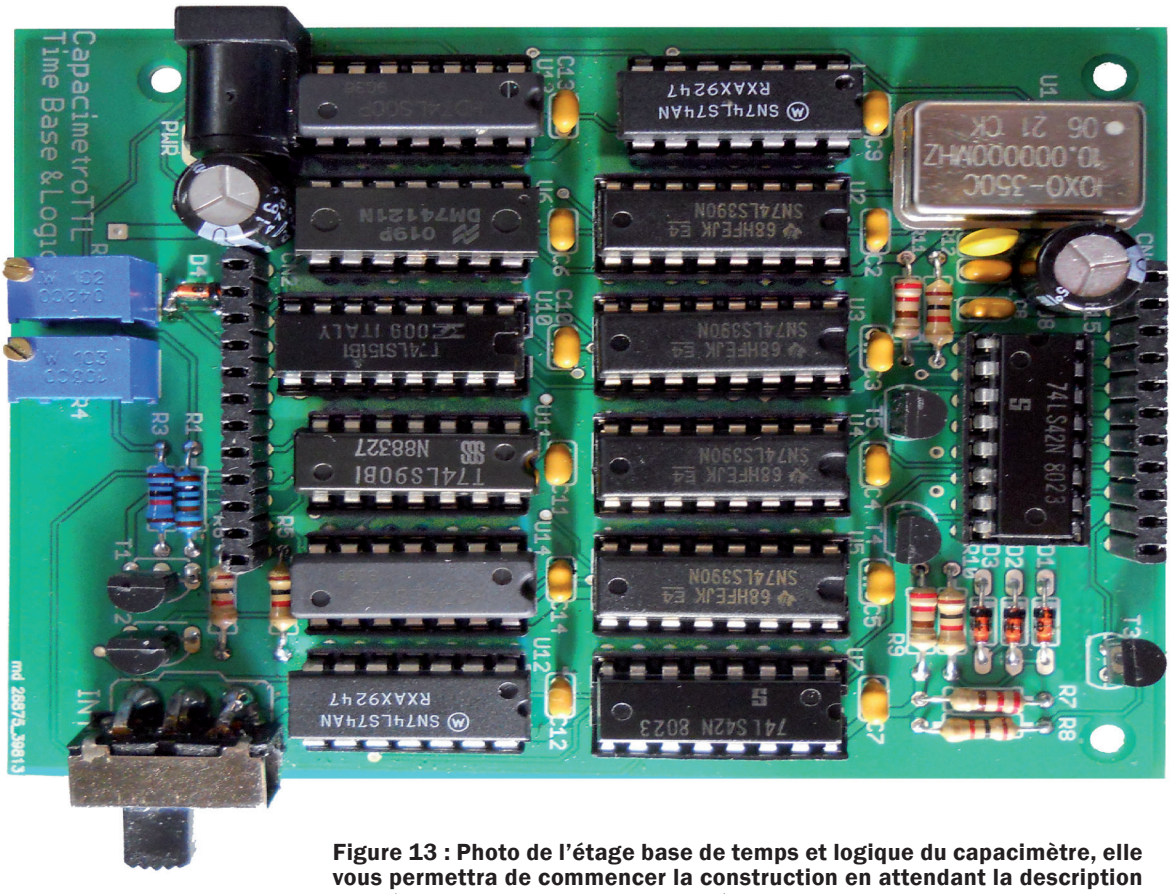


Figure 13 : Photo de l'étage base de temps et logique du capacimètre, elle vous permettra de commencer la construction en attendant la description des réglages dans le prochain numéro .

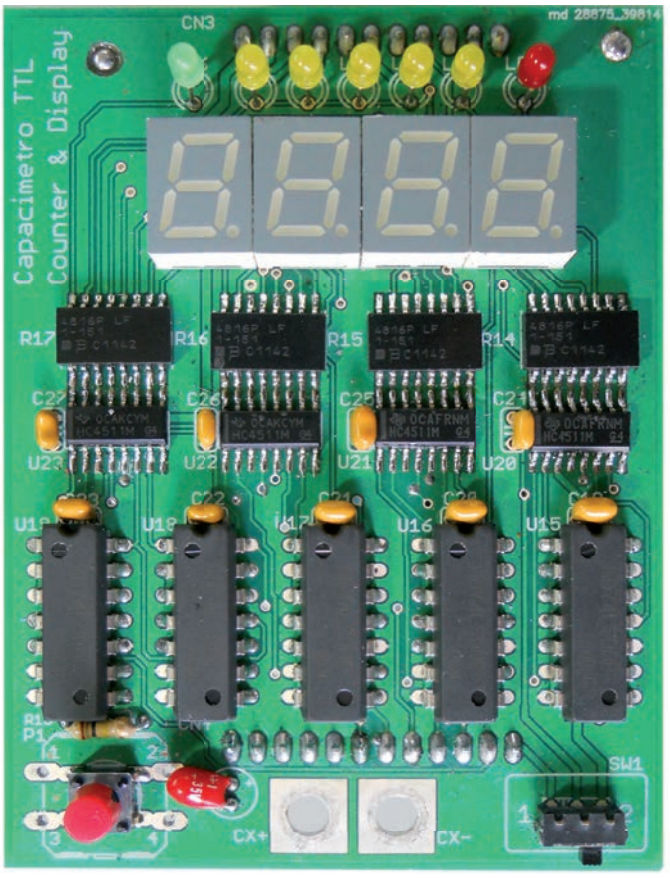


Figure 14 : Photo de l'étage compteur et des afficheurs 7 segments du capacimètre, elle vous permettra de commencer la construction en attendant la description des réglages dans le prochain numéro .

les courants inverses de l'entrée **TTL** vers le condenseur à mesurer, ce qui fausserait la lecture).

La diode **D4** ne pose pas de problèmes particuliers pour les autres types de condensateurs.

Dans notre cas le circuit est un peu plus compliqué, puisque nous utilisons deux valeurs de résistances différentes. Pour les 4 premiers calibres (mesures de condensateurs de 10pF à 100µF), nous utilisons une résistance d'environ 15 kΩ (R3 de 10 kΩ et un trimmer R4 de 10 kΩ en série) pour charger le condensateur.

Cette valeur devient excessive pour les condensateurs de haute capacité, le temps de charge va augmenter considérablement empêchant ainsi une mesure précise.

Pour surmonter ce problème, nous utilisons une résistance d'une valeur 10 fois moindre, soit 1,5 kΩ (R1 de valeur 1 kΩ et le trimmer R2 de valeur 1 kΩ en série). A ce stade se pose le problème de la façon de commuter la résistance utile, selon le calibre choisi.

Comme vous pouvez le voir sur le schéma électrique en figure 5, la solution que nous avons trouvée est assez simple. Chacune des deux résistances (pour plus de commodité nous ne parlerons que de R1 et R3) est reliée d'un côté à la broche R/C et l'autre côté au collecteur d'un transistor PNP (R1 à T1 et R3 à T2).

Dans la pratique, lorsque l'une des sorties Q0 à Q3 de U8 est active, la LED correspondante (LD1 à LD4 voir la figure 6 schéma électrique C&D) se retrouve avec sa cathode autour de 0 V, l'anode étant connectée au point AL2 qui, à travers R6 est reliée à la base de T2.

Sur la base de T2 se trouve un potentiel d'environ 0,7 V inférieur au 5V présent sur l'émetteur de T2, ce qui le rend conducteur et met la résistance R3 à 5 V.

Simultanément, la résistance entre les broches 1 et 16 du réseau de résistance R16 (voir la figure 6 schéma électrique C&D) de 180 Ω, fait circuler à travers la LED un courant d'environ 14 mA, et provoque l'allumage.

De cette façon, lorsque vous sélectionnez l'un des 4 premiers calibres, à travers les sorties de U8 se trouve alimenté la LED correspondante et la broche R/C de U6 (figure 5) est alors connectée avec une résistance d'environ 15 kΩ.

La même chose est vraie lorsque la sortie Q4 de U8 se trouve à un niveau bas L (broche 7 du connecteur AL CN3 figure 5), dans ce cas la diode LD5 reliée à **AL CN1** (figure 6), et rend conducteur le transistor T1. La broche **R/C** de U6 se retrouve avec une résistance d'une valeur de 1,5 kΩ. La LED LD5 est allumée en raison du courant circulant dans la résistance entre les broches 1 et 16 du réseau de résistance R15, toujours de 180 Ω.

Nous reviendrons plus tard sur cette partie lorsque nous expliquerons la mesure de la capacité.

La sélection du calibre et de l'horloge

Cette fonction importante du montage est réalisée à l'aide des circuits **U14** (SN74LS14, portes U14c, U14e, U14f), **U11** (SN74LS90), **U8** (SN74LS42) **U10** (SN74LS151) et également à l'aide des composants **P1**, **R13** et **C18** visibles sur la figure 6 (schéma électrique C&D).

Avant de commencer à expliquer le fonctionnement de cette partie, vous devriez relire le passage sur l'anti-rebond (debounce), en début d'article.

Nous avons réalisé notre système de **changement de calibre au moyen de la résistance de pull-up R13 (100 kΩ), du consensateur C18 connecté à la masse (1µF au tantale) et du bouton poussoir P1 en parallèle avec le consensateur** (schéma C&D en figure 6).

Le point dénommé **S1** va directement à l'entrée de la porte **NOT U14e.** Le fonctionnement du circuit anti-rebond est assez simple.

Lorsque le poussoir est relâché (bouton ouvert), la résistance charge le condensateur et sur l'entrée **NOT** se trouve un état logique **H**. Sur la sortie de cette porte évidemment l'état logique est inversé (**L**) et appliqué à l'entrée de la second porte **NOT U14f**, à la sortie de celle-ci nous trouvons de nouveau un niveau **H** qui arrive sur l'entrée **CKA** de **U11**. Cette condition ne provoque pas d'activité du circuit puisque, comme vous le savez, il nécessite un « **front descendant** » pour compter les impulsions.

Notez la configuration des 4 entrées « **RESET** » de U11. **R9-1** et **R9-2** (broches 6 et 7) sont reliées à la masse, donc désactivées (n'oubliez pas que le circuit de « **RESET** » du **SN74LS90** nécessite un signal de **niveau logique H pour réinitialiser les sorties**).

Les lignes **R0-1** et **R0-2** (broches 2 et 3) sont reliées à la sortie de la porte **NOT U14c**, l'entrée de cette porte est reliée à la sortie **O5'** de U8, étant normalement à un niveau **H**, porte à un niveau logique **L** ces deux broches pour les maintenir inactives.

Puisque nous sommes dans la condition initiale, les sorties QA à QC de U11 et, par conséquent, les entrées A, B, C de U8 et U10, sont sur des niveaux « LLL ». Dans ce cas sur U8, vous avez la première sortie (O0') qui se trouve à un niveau L et toutes les autres à un niveau H. L'entrée D0 de U10 est activée, et inversée sur la sortie W'.

Voyons ce qui se passe après les deux circuits intégrés. La sortie **O0'** de **U8**, nommée « 100nF » sur le schéma de la figure 5 (c'est-à-dire le calibre) **porte la base de T4 à travers R10 et D2 à un niveau de tension inférieur d'au moins 0,7 V par rapport à l'émetteur**, ainsi **le transistor PNP commence à conduire**, mettant à 5V le « point décimal » de l'afficheur DP2 qui s'allume.

La résistance **R9 limite le courant** à environ 15 mA. Notez aussi que la même sortie **O0' provoque également la liaison de R3 avec U6** et l'allumage de la LED LD1. A ce stade l'afficheur indique « **00.00** » et **LD1 s'allume**, ce qui veut dire que le capacimètre se trouve sur le calibre « 100 nF », ou mieux « **99,99 nF** ».

Par contre la sortie **W'** de **U10** transmet un signal d'une fréquence égale à 10 MHz (entrée D0) à l'une des entrées de la porte **NAND U13b** (broche 5 du **SN74LS00**).

Maintenant, revenons à l'entrée **U14e** et imaginons que nous appuyons sur P1 pour passer au calibre suivant. Cette action provoque la décharge immédiate du condensateur C18 et relie simultanément l'entrée de la porte **NOT** à la masse (GND).

En réalité, comme vous le savez, le bouton poussoir P1 provoque un certain nombre de rebonds (« bounce ») qui sont partiellement ignorés par le circuit « **Trigger de Schmitt** » de la porte **NOT**, mais certains sont assez grands pour être reconnus comme des pressions successives et peuvent tromper le compteur.

En présence de **C18**, complètement **déchargé**, ces impulsions sont absorbées pour recharger le condensateur en faisant en sorte que le niveau logique à l'entrée de la porte NOT soit maintenu pendant un temps suffisamment long, en-dessous du seuil de 0,8 V qui est la limite de l'état logique bas L.

Ce temps est déterminé par le circuit RC et est calculé selon la formule :

T (ms) = R (kΩ) x C (µF) x 0,9

Dans notre cas, nous avons **T = 100 x 1 x 0,9 = 90 millisecondes**, un temps suffisamment long pour supprimer les rebonds de P1. (NB : lors de nos tests, nous avons trouvé que la meilleure réponse est donnée par les condensateurs au tantale, nous vous recommandons fortement d'utiliser ce type de condensateur pour C18).

Finalement, le circuit **anti-rebond garantit que toute pression sur le bouton P1 soit correctement interprétée comme un unique niveau logique bas L sur l'entrée de U14e**. Simultanément la sortie de **U14f** passe d'un état **H** à **L** et génère un « **front descendant** » sur l'entrée **CKA** de **U11** qui comptabilise alors comme une impulsion.

Les sorties **QA** à **QC** passent alors à « **LLH** », **U10** commute sur la sortie **W'** la fréquence de 1 MHz et **U8** met la sortie **01'** sur **L** (calibre 1000 nF). À la suite de l'explication précédente, il sera maintenant facile de comprendre le comportement des autres composants impliqués dans le projet.

Il est intéressant de voir ce qui se passe à la quatrième pression de P1. Les sorties QA à QC prennent respectivement les états « HLL » et la sortie W' de U10 est maintenant à une fréquence de 1 kHz, tandis que U8 porte la sortie 04' à un niveau bas L (calibre 10 000µF).

Comme déjà décrit précédemment, cette sortie 04' provoque la liaison de R1 avec U6 et l'allumage de la LED LD5.

Enfin, nous allons voir ce qui se passe lorsque nous appuyons une cinquième fois sur le bouton P1. Les sorties de U11 passent respectivement sur les états « HLH », et sur la sortie W' de U10 théoriquement il manquerait un signal d'horloge (l'entrée D5 est libre).

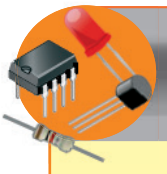
En fait, cette condition ne persiste pas car U8 porte à un niveau bas L sa sortie 05', la porte NOT **U14c** inverse sa sortie à un niveau H que l'on retrouve sur les deux broches R0 -1 et R0-2, qui, dans cet état, réinitialisent le compteur U11 en mettant simultanément les sorties à « LLL », ce qui ramène le calibre sur « 100 nF » et la fréquence d'horloge à 10 MHz.

Ainsi, lorsque vous atteignez le dernier **calibre « 10 000µF », une pression supplémentaire sur P1 ramène le capacimètre sur le premier calibre « 100 nF ».**

Nous arrivons au terme de ce premier article dans lequel nous avons étudié une partie de la théorie des circuits logiques **TTL** et **CMOS** ainsi que le début de l'analyse schématique de notre capacimètre. Afin de **permettre aux lecteurs d'entreprendre la construction du capacimètre en attendant la suite de l'article dans le prochain numéro du mois de juin**, nous avons publié les circuits imprimés des deux platines (voir les figures 7 à 12) ainsi que des photographies en haute qualité du prototype (figures 13 et 14) afin de vous aider dans la fabrication.

Bien que nous n'ayons pas encore abordé le chapitre de la réalisation pratique et celui des réglages, la plupart des lecteurs mêmes néophytes peuvent se laisser dès à présent dans l'aventure car ce projet ne nécessite pas de composant à programmer. Cependant un soin particulier est à prendre dans la fabrication des circuits imprimés en évitant tout excès de soudure qui pourrait provoquer des courts-circuits.

Dans le prochain numéro nous étudierons la logique de contrôle, la mesure de la capacité et enfin la réalisation pratique des deux platines de notre capacimètre.



Comment construire ce montage

Tous les composants du capacimètre se trouvent facilement dans le commerce. Les typons des circuits imprimés sont disponibles en téléchargement gratuit sur notre site **www.electroniquemagazine.com** dans la section « Sommaire détaillé » du numéro 126 à l'onglet « Télécharger ».

Transformez votre TV en SMART TV

Mini PC Android MK802IV

Ce PC miniature relié à une télévision en HDMI la transforme en une smart TV avec laquelle vous pouvez naviguer sur Internet, regarder des vidéos en streaming en HD, films en 3D et jouer en ligne. Il se connecte en wireless à votre réseau Wi-Fi (essentiel pour permettre au dispositif de fonctionner). Le port USB placé sur le côté est capable de gérer aussi bien des périphériques tels que souris, claviers et webcams, que des lecteurs de disques USB. Livré avec Android 4.2 Jelly Bean pré-installé et également Google Play boutique. Pour profiter de toutes les fonctionnalités du MK802IV, il est recommandé d'utiliser le clavier sans fil Mini 8 vendu séparément.

réf. MK802IV

€ 116,99

réf. MINI8

€ 37,70

Clavier Wireless Rii Mini i8

Clavier sans fil capable de contrôler à distance toutes les fonctionnalités du PC miniature MK802IV, permettant d'utiliser tous les services disponibles sur le téléviseur. Il peut également être utilisé pour de nombreuses autres applications.

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

Comelec 03 / 2014

CBAT2550 - 2

Chargeur d'accumulateur universel

Deuxième partie

de **Mirco Segatello**

Nous continuons dans ces pages la deuxième partie de la description de notre chargeur/déchargeur professionnel d'accumulateur de type : Ni-Cd (nickel-cadmium), Ni-MH (nickel-hydrure métallique), Li-Po (lithium-ion polymère), Li-Io (lithium-ion), Li-Fe (lithium fer phosphate) et Pb (plomb). Nous aborderons la construction mécanique et l'étude du firmware (programme).

Nous débutons cette seconde partie **par un erratum concernant la liste des composants parue dans le numéro 125 d'Electronique et Loisirs Magazine, la** Rédaction présente ses excuses à ses fidèles lecteurs. En effet une erreur s'est introduite au niveau de l'imprimerie dans la liste des composants du numéro 125 à la page 30, vous pouvez lire ceci en début de liste :

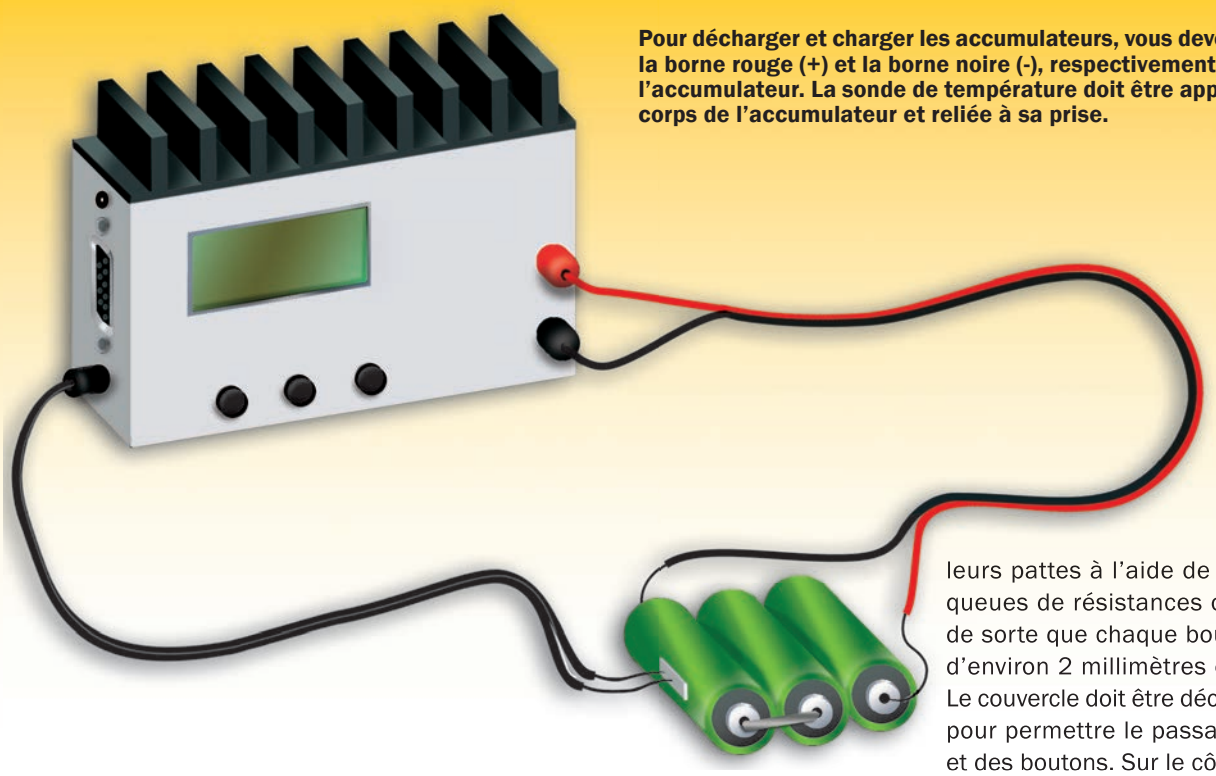
~~R1..... 56 k 2 W~~
R1..... 10 kΩ 1%

La vraie valeur de R1 est : 10 kΩ 1% (**R1..56 k 2 W n'existe pas**).

Cela étant dit, dans le précèdent numéro, nous avons commencé la description du chargeur géré par un microcontrôleur, et qui peut charger et décharger **tout type d'accumulateur présent sur le marché de manière optimale**, tout en garantissant la plus grande capacité possible et une prolongation (même au-delà des attentes) de la durée vie.

La polyvalence et les performances supérieures que ce chargeur vous permet d'obtenir des accumulateurs sont essentiellement dues à un montage complexe qui utilise un **convertisseur bimodal** (**mode buck + boost**) et une **conception avancée du firmware du microcontrôleur** qui permet une surveillance constante des paramètres de base (température, courbe de variation de la tension et du courant de charge), mais aussi par le contrôle précis de la charge qui s'effectue en fonction des courbes prédéfinies et optimisées par les constructeurs de chaque type d'accumulateur.

Après avoir exposé les spécificités du projet et décrit le schéma électrique du circuit avec un accent particulier sur les solutions matérielles adoptées, le moment est venu de vous décrire la construction de l'appareil et l'étude du firmware du microcontrôleur.



Pour décharger et charger les accumulateurs, vous devez connecter la borne rouge (+) et la borne noire (-), respectivement au + et – de l'accumulateur. La sonde de température doit être appliquée sur le corps de l'accumulateur et reliée à sa prise.

Réalisation pratique

Le typon du circuit imprimé à double face est disponible en téléchargement gratuit sur notre site **www.electroniquemagazine.com** dans la section « **Sommaire détaillé** » du **numéro 126** à l'onglet « **Télécharger** ». Une fois que vous avez gravé le circuit imprimé, percez les trous correspondant aux composants et puis commencez par souder les résistances, les diodes, les supports de circuits intégrés, le buzzer, les condensateurs céramiques et les condensateurs électrolytiques en respectant la polarité (le « - » est indiqué sur le boîtier).

Enfin, il vous reste à souder les composants en boîtier TO-220 (U1, T2, T4, T3, D4, D5) et l'inductance. **Portez une attention particulière aux résistances R31** et **R32**, qui doivent avoir une tolérance la plus faible possible, soudez-les en gardant un espace de quelques millimètres entre le corps des résistances et le circuit imprimé afin de faciliter la dissipation thermique.

L'inductance est déjà prête, mais si vous voulez la construire vous-même, vous pouvez utiliser un tore de dimensions de 27 x 15 x 11 mm et enrouler **50 spires de fil de cuivre émaillé d'un mm** de diamètre.

Les extrémités de l'enroulement doivent être grattées pour permettre le soudage. Il est recommandé également d'insérer une rondelle en plastique entre l'inductance et le circuit imprimé, pour éviter tout court-circuit qui pourrait se produire avec l'enroulement.

Faites attention lorsque vous soudez les composantes en boîtier TO-220, il faut les mettre à la même hauteur et les insérer jusqu'au fond dans le circuit imprimé. Pour comprendre le montage, aidez-vous des photos.

Les dimensions du circuit imprimé sont prévues pour une installation dans un boîtier en métal modèle **AL BOX 4/A.1** de **Teko** (www.teko.it/fr/prodotti/famiglia/AL/serie/6) facilement disponible à un coût modeste. Une fois terminé, le circuit imprimé peut être placé dans le boîtier à l'aide d'entretoises et de rondelles en plastique en laissant un espace de 2 mm avec le fond et en le fixant avec des écrous. Sur un des côtés il faut prévoir le montage des boîtier s TO-220 et le dissipateur. Du côté du couvercle, positionnez l'écran LCD de telle manière qu'une fois le boîtier fermé, l'écran arrive au niveau de la découpe du couvercle.

Les boutons doivent être soudés sur le circuit imprimé après avoir prolongé leurs pattes à l'aide de morceaux de queues de résistances ou de diodes, de sorte que chaque bouton dépasse d'environ 2 millimètres du couvercle. Le couvercle doit être découpé et percé pour permettre le passage de l'écran et des boutons. Sur le côté vous devez percer les trous de la fiche d'alimentation, du connecteur DB-9 et du connecteur du capteur de température.

La **fixation des composants en boîtier TO-220 nécessite beaucoup d'attention**, tout d'abord vous devez vous procurer un dissipateur compatible avec la taille du coffret, nous avons utilisé pour notre prototype un dissipateur de dimensions 25 x 140 x 25 mm. Les vis doivent **être fixées en même temps dans les composants TO-220, dans le côté du boîtier et dans le dissipateur**, comme vous pouvez le voir sur les photos. Évidemment, le dissipateur doit présenter les trous appropriés pour accueillir les vis de fixation. La solution proposée n'est qu'un exemple possible visant à obtenir un boîtier compact et esthétique, mais vous pouvez adopter d'autres solutions en fonction de vos besoins.

Lorsque vous insérez la carte électronique, il est souhaitable de prévoir **une liaison entre la masse de l'électronique et la partie métallique du boîtier** , afin d'éviter que des décharges électrostatiques créent des problèmes de fonctionnement du chargeur. Notez que pour les essais, le courant maximal, que vous pouvez utiliser sans boîtier et sans dissipateur, ne doit pas dépasser pour le réglage du courant de sortie 200 à 300 mA.

Tous les composants en boîtier TO-220 doivent être isolés électriquement du coffret et du dissipateur, sinon cela provoquerait des courts-circuits. Il est nécessaire d'intercaler entre la surface métallique des TO-220 et la surface de contact du dissipateur des **feuilles de mica ou de Teflon isolantes**, en prenant soin d'isoler les vis de fixation au moyen de rondelles en plastique appropriées.

Comme alternative aux rondelles, vous pouvez utiliser des **vis en Teflon**. Une fois les composants montés, vous devez tester l'isolation électrique entre le dissipateur, le coffret et les boîtiers TO-220.

A l'aide d'un **multimètre positionné sur la fonction ohmmètre** (échelle x 1), **mesurez la résistance entre le métal des TO-220, le coffret**, et le dissipateur en plusieurs endroits ; il doit y avoir une **résistance infinie** ou presque.

Le capteur de température interne du chargeur, dont la valeur de la **CTN** est de **10 kΩ**, doit être fixé dans le boîtier proche du dissipateur afin de détecter rapidement la température. Il serait préférable d'enduire de pâte thermique silicone le capteur et le dissipateur. **Contrôler la température du circuit est également important**, car la limite de fonctionnement à courant maximal n'est pas un problème pour l'étage final, **mais plutôt** pour les **condensateurs C5**, **C10** et l'**inductance**.

Pour comprendre ce phénomène, il est nécessaire de rappeler que les condensateurs ont une **résistance interne ESR** (résistance série équivalente) et que lorsqu'ils doivent fournir des pics de courant élevés, comme dans le cas de la commutation dans une alimentation, leur **résistance de fuite interne provoque un effet Joule qui est dissipé sous forme de chaleur**.

Dans le cas de notre chargeur, ce phénomène est d'autant plus gênant que le courant de sortie est élevé. Pour cette raison, nous limitons la puissance de sortie maximale à 50 W, ce qui correspond à un courant maximal de 2 A, dans le cas où nous avons le nombre maximal de cellules en charge, avec des tensions de sortie inférieures à 14 V, le courant peut atteindre 5 A.

Comme il n'y a aucune limitation au niveau du programme, nous pouvons améliorer les performances en adoptant pour **C5** et **C10** des condensateurs à **faible résistance série** (type Low ESR). Dans notre montage, nous avons utilisé pour C5 et C10 des condensateurs **Panasonic** de type **EEUFM1C102** et pour **C23** un de type **EEUFM1E471L**.

De cette façon nous pouvons atteindre un courant de 3 A pour la charge de 6 cellules au lithium polymère (Li-Po). Une autre possibilité pour les condensateurs « **Low ESR** » est d'utiliser un coffret plus haut (TEKO 4/B.1) de sorte que l'on soude du côté soudures du circuit imprimé, 2 autres condensateurs en parallèle avec ceux déjà existants (C5, C10), ce qui a pour effet de diminuer la résistance ESR (mise en parallèle).

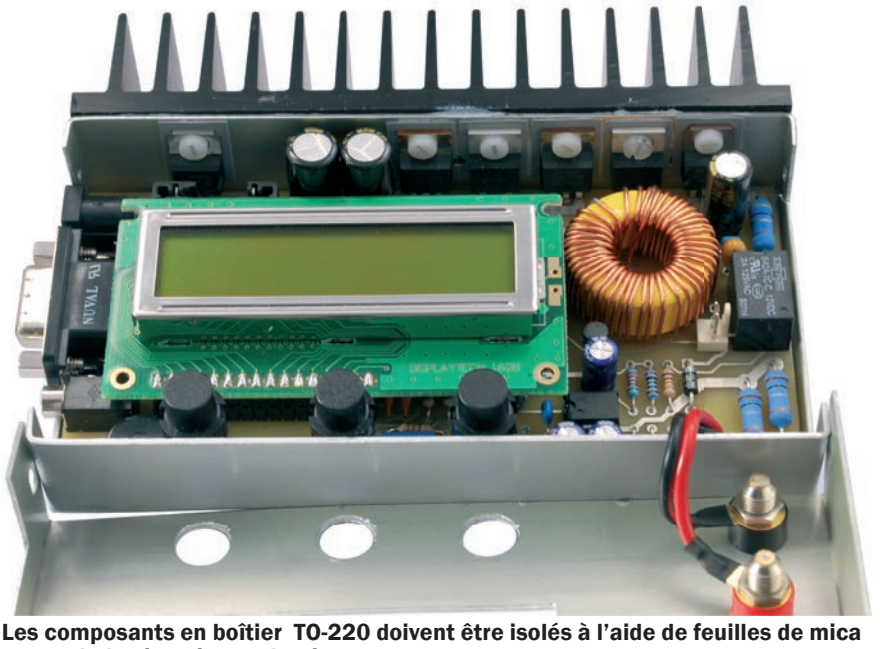
L'espace intérieur disponible vous permet d'insérer un petit ventilateur de refroidissement (12 V DC - 100 mA) relié au connecteur approprié sur le côté droit de l'inductance. Si vous optez pour la version 10A de l'inductance, vous pouvez la fixer en dessous du circuit imprimé.

Toujours dans un souci d'efficacité de la dissipation thermique, même si les composants TO-220 sont installés très proches l'un de l'autre, ils ne fonctionnent pas tous en même temps et donc il n'y a pas de problèmes liés à la dissipation thermique.

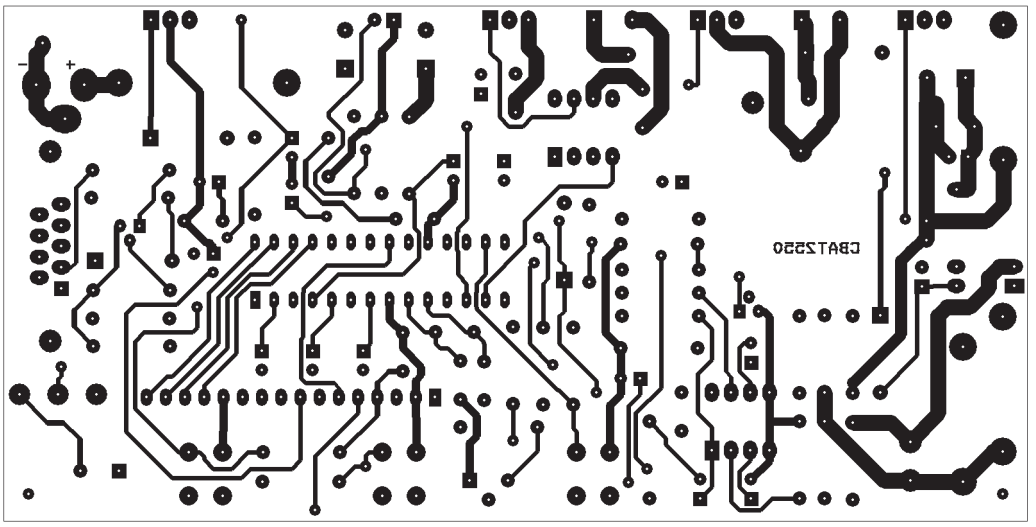
En ce qui concerne les diodes **Schottky**, si vous ne trouvez pas celles prescrites dans la liste des composants, vous pouvez les remplacer par un modèle équivalent ayant un courant d'au moins 8 A, de même pour les **MOSFET** vous devez utiliser des modèles dont la tension **Vds** est d'au moins **30 V** avec un courant **Id** d'au moins **15 A**.

Un point particulier à respecter est la valeur de la **RdsON** (c'est la résistance du MOSFET à l'état passant), elle doit être **aussi faible que possible**. Les **MOSFET** utilisés dans notre montage sont des **IRF9Z34N** (canal P) avec un courant de drain **Id** de **19 A**, une tension **Vds** de **55 V** et une résistance **RdsON** de **0,1 Ω**, et des **IRF3205** (canal N) avec un **Vds** de **55 V**, un **Id** de **110 A** et une **RdsON** de **0,008 Ω**.

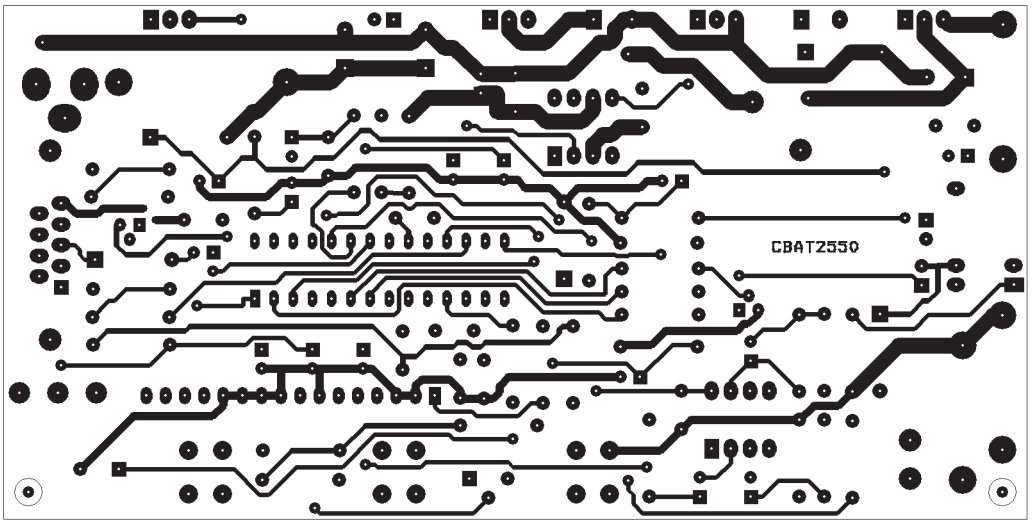
En bas à droite, soudez directement 2 fils (un rouge et un noir) sur le circuit imprimé, et de l'autre côté de chaque fil soudez 2 douilles de 4 mm, une rouge et une noire, qui seront fixées sur le couvercle. Les douilles servent de connexion pour l'(es) accumulateur(s). Ensuite vous devez **percer le couvercle du coffret selon le plan de la figure 1**. Maintenant il ne vous reste plus qu'à mettre l'ensemble du circuit dans le boîtier et de fixer les 4 vis du couvercle.



Les composants en boîtier TO-220 doivent être isolés à l'aide de feuilles de mica et par de la visserie en plastique.



Circuit imprimé à l'échelle 1 : 1 de notre chargeur/déchargeur côté soudures.



Circuit imprimé à l'échelle 1 : 1 de notre chargeur/déchargeur côté composants.

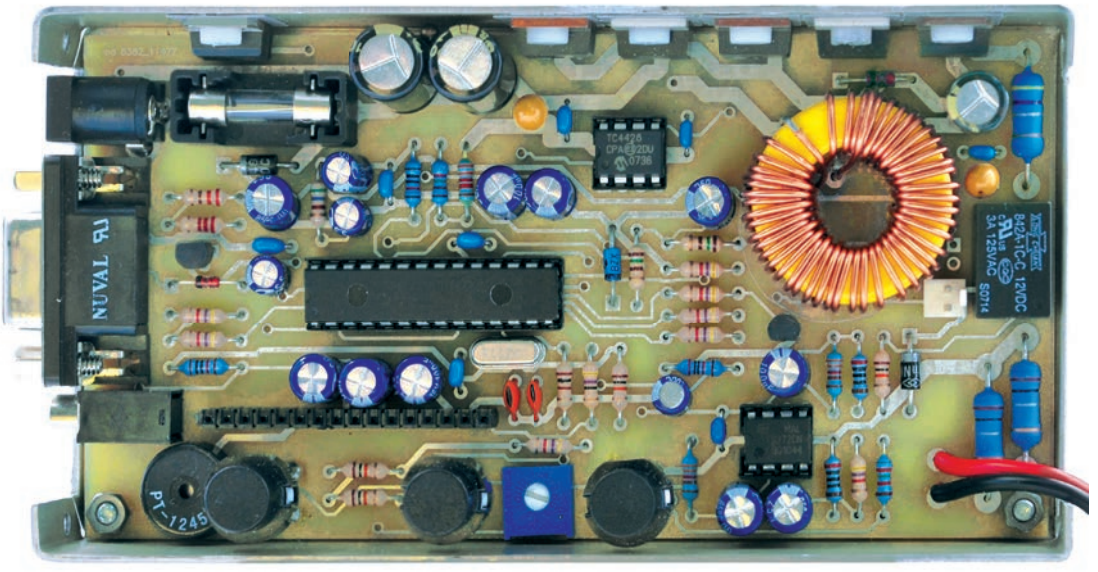


Photo de l'un de nos prototypes de notre chargeur/déchargeur.

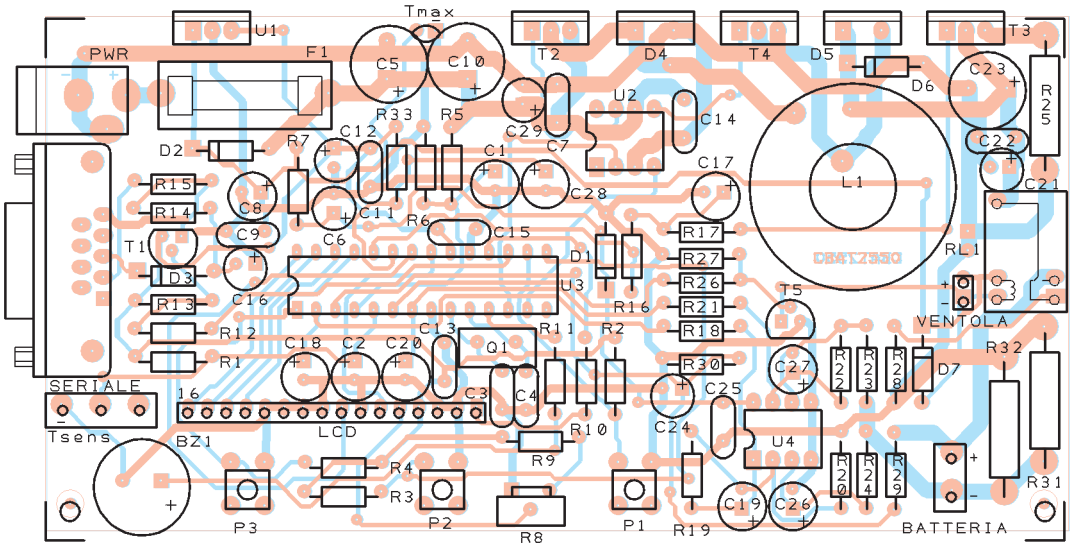


Schéma de câblage de notre chargeur/déchargeur.

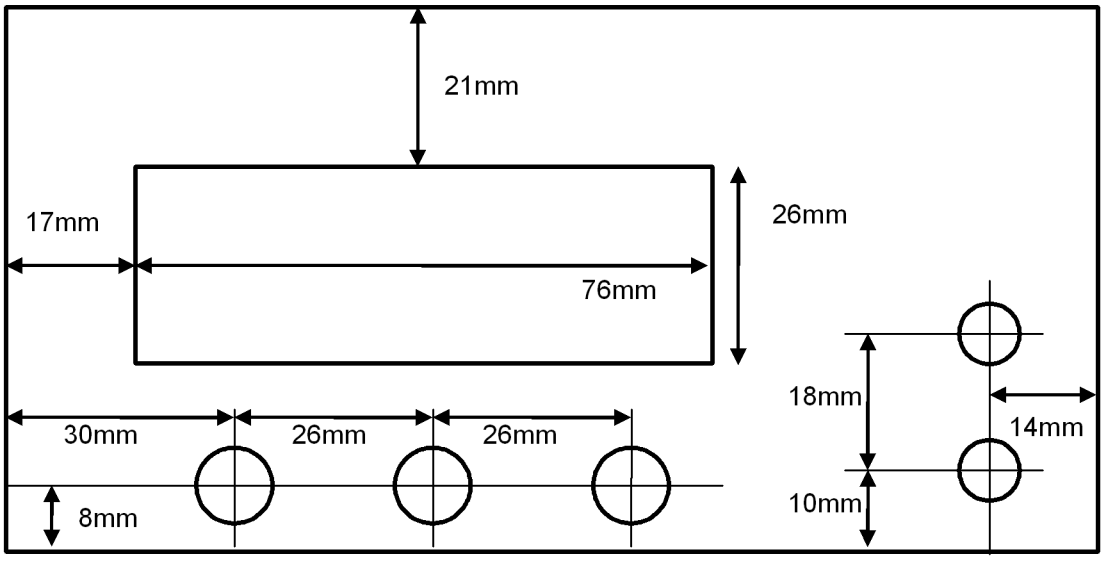


Figure 1 : plan de perçage du couvercle de notre chargeur/déchargeur.

Pour construire le capteur de température extérieur (mesure de la température du ou des accumulateurs), en plus de la thermistance CTN de 10 kΩ, vous devrez vous procurer un morceau de câble à 2 conducteurs et un connecteur mâle compatible avec la prise présente sur le circuit imprimé. Soudez la CTN à une extrémité du câble et de l'autre côté du câble, soudez les 2 fils au connecteur.

Adaptez un petit aimant sur le capteur de sorte qu'il puisse être facilement apposé sur l'accumulateur pendant la charge. Si vous prévoyez d'utiliser le chargeur dans la voiture, vous pouvez souder à la place de la fiche d'alimentation une prise « allume cigare ».

Enfin n'oubliez pas de programmer le PIC avec le programme .hex disponible gratuitement sur notre site.

Tableau 1

Tension d'alimentation	Tension de sortie	Courant de sortie	courant absorbé
13,8 V	7,8 V	1 A	0,6 A
13,8 V	8,8 V	3 A	2,2 A
13,8 V	8,2 V	5 A	3,7 A
13,8 V	11,7 V	2 A	1,8 A
13,8 V	11,7 V	5 A	4,6 A
13,8 V	24,2 V	2 A	4,4 A
13,8 V	24,1 V	3 A	6,0 A

Les essais

Bien qu'il ait été conçu pour fonctionner à partir d'une batterie de voiture, le chargeur, convenablement alimenté par une alimentation électrique appropriée, peut être utilisé pour des applications domestiques.

Un simple transformateur 50 Hz, un redresseur et un condensateur peuvent suffire, mais afin de limiter l'espace et d'améliorer le rendement, nous vous recommandons l'utilisation d'une **alimentation à découpage** avec une sortie réglable. Elle doit fournir au moins 13,8 V pour charger des batteries de 12 V, (15 V ou plus) pour les packs d'accumulateurs à tension plus élevée.

En général, les alimentations à découpage assurent une tension constante en fonction des pics de courant absorbés. A cet égard, dans le tableau 1, nous montrons la valeur du courant absorbé en fonction du courant de sortie.

Pour un premier test, vous pouvez alimenter le chargeur avec une petite alimentation de 12 V, 5 W. Vérifiez maintenant le fonctionnement de l'écran LCD et des boutons, **ajustez si nécessaire le contraste de l'écran à l'aide du potentiomètre**. Pour le bon fonctionnement du chargeur, il est nécessaire de vérifier que la valeur du courant et la valeur de la tension de sortie sont correctement indiquées sur l'afficheur. Branchez un accumulateur en série avec un ampèremètre aux bornes de la charge et vérifiez que la valeur du courant affichée sur l'écran LCD corresponde à celle que l'instrument indique.

Maintenant, connectez un voltmètre sur l'accumulateur et vérifiez que la tension mesurée coïncidence avec celle indiquée par le chargeur. Répétez l'opération pendant les 2 premières minutes de charge, avec un courant fixé à 100 mA environ.

La fiabilité de la mesure de la tension dépend, en plus de la tolérance des résistances, de la précision du régulateur 7805 qui alimente la référence de tension du convertisseur A/N du PIC. Bien sûr, vous pouvez utiliser un régulateur de précision pour la partie « référence de tension du convertisseur », cependant cela complique encore le circuit, mais la mesure de la tension doit être très précise lors de la charge d'accumulateurs au lithium.

Terminez le câblage du chargeur, pensez à la connexion des câbles qui vont à la charge, ils doivent être dimensionnés de manière appropriée et aussi courts que possible, afin d'avoir le minimum de chute de tension lors de charges à forts courants.

Le programme (firmware)

Passons maintenant à la description de la partie la plus intéressante du projet, qui est le programme qui « tourne » dans le microcontrôleur. Comme nous l'avons déjà indiqué précédemment, c'est un **PIC 18F2550** qui est utilisé ici, son choix est dû à sa haute fréquence d'horloge (48 MHz environ) et son interface USB intégrée. Les 3 sorties **PWM** (modulation de largeur d'impulsions) du PIC sont utilisées pour gérer la commande du convertisseur bimodal et l'étage de décharge à transistors. Avec une résolution de 10 bits de l'étage **PWM**, la fréquence de découpage est fixée à 48 kHz, valeur suffisamment élevée pour permettre l'utilisation d'une inductance relativement faible. La gestion des boutons et de l'afficheur LCD a été largement décrite dans de nombreux articles d'Electronique et Loisirs Magazine, de sorte que nous ne nous y attarderons pas davantage. Cinq entrées analogiques sont utilisées pour mesurer la tension de la charge, le courant de charge et de décharge, la température interne du chargeur ainsi que la température de l'(es) accumulateur(s).

Dans tous les cas, le convertisseur est configuré pour une acquisition de 10 bits. Cependant, grâce à une moyenne algébrique exploitée sur plusieurs lectures, la résolution dans la pratique atteint 13 bits, ce qui assure une excellente précision. La gestion de la commutation du convertisseur bimodal est obtenue en utilisant 2 sorties **PWM** disponibles sur les broches **RC1** et **RC2**, tandis que la 3eme sortie **PWM** qui correspondant à la broche **RB3** est utilisée uniquement lors de la phase de décharge de l'(es) accumulateur(s).

Comme nous l'avons déjà mentionné lors de la description du convertisseur bimodal, le signal **PWM** de la broche **RC1** est utilisé pour commander le **MOSFET T2** lorsque la **tension de sortie doit être inférieure à la tension d'alimentation**, tandis que le signal **PWM** de la broche **RC2** contrôle le **MOSFET T4** dans le cas où la **tension de sortie doit être supérieure à la tension d'alimentation**. Le microcontrôleur génère les signaux **PWM** de manière à maintenir constantes les valeurs de la tension de sortie, en fonction du programme actif à ce moment, grâce à la mesure de la tension et du courant de l'(es) accumulateur(s).

Le programme détermine la fin de la charge en utilisant les méthodes suivantes :

- **Delta-Peak** : la fin de la charge est caractérisée par un pic de tension, il est adapté seulement pour la charge des accumulateurs **Ni-Cd** et **Ni-MH** en mode rapide avec un courant de charge supérieur à **0,5 C**.

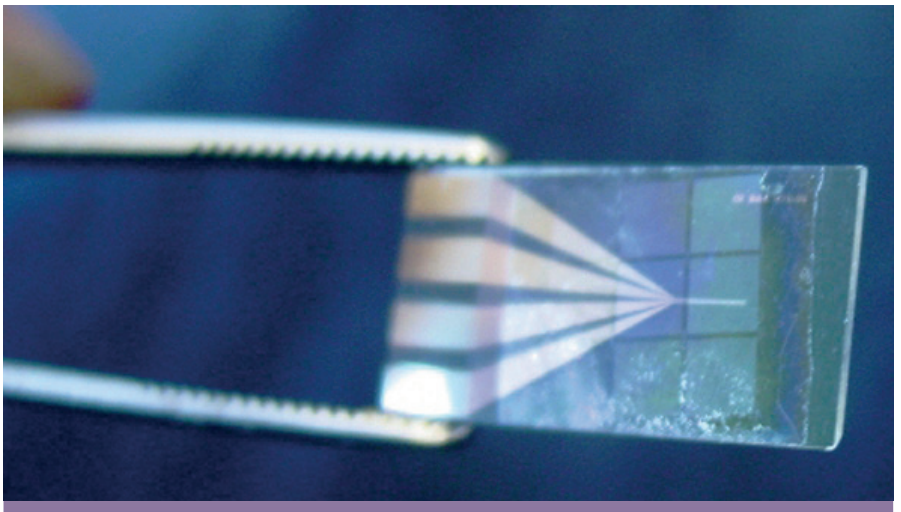
- **Charge en mode CC/CV** : ces modes correspondent à la méthode standard pour charger des accumulateurs au **lithium** et au **plomb**.

- **Durée** : la charge s'effectue à courant constant pour un temps déterminé. Cette méthode est utile pour charger les accumulateurs **Ni-Cd** et **Ni-MH** à faible courant et dans le cas où la détermination du **Delta Peak** ne peut pas être effectuée correctement.

En ce qui concerne les protections, les systèmes suivants ont été mis en œuvre :

- **Délai de sécurité** (Safety Time Limit) : cette fonction est activée avec l'une des méthodes décrites ci-dessus et permet d'arrêter la charge lorsque la durée du temps fixée est atteinte, ce qui assure une protection dans le cas où, pour diverses raisons (telles que des accumulateurs très vieux, ou partiellement endommagés ou dans le cas de courants de charge trop faibles), le **Delta Peak** n'est pas détecté.

- **Arrêt de la charge** (Capacity cut-off) : cette fonction est également activée avec l'une des méthodes décrites ci-dessus et vous permet d'arrêter la charge à un certain seuil (de même que la décharge). La valeur est exprimée en **mA/h** (milliampères par heure).



L'énergie tirée d'un virus

Cela ressemble à de la science-fiction, mais c'est une expérience bien réelle ; en 2006 des chercheurs du MIT (Massachusetts Institute of Technology à Boston) ont modifié l'ADN du virus M13 pour produire une protéine qui se mélange avec les ions cobalt en solution, donnant de l'oxyde de cobalt, un matériau possédant une assez haute capacité de stockage de l'énergie, plus élevée que celle des batteries lithium-ion actuelles, basées sur le carbone. Ils ont construit un prototype de la batterie dont la taille est extrêmement réduite, adaptée pour les téléphones mobiles et les lecteurs MP3.

Dans une batterie classique, les ions lithium s'écoulent entre l'anode (-) généralement en graphite, et la cathode (+) généralement de l'oxyde de lithium et de cobalt. Dans la nouvelle pile, on utilise des virus qui sont recouverts de phosphate de fer agrégé en nanotubes de carbone afin de créer un réseau de matériaux de haute conductivité. Etant donné que les virus reconnaissent et se lient spécifiquement à certains matériaux, chaque élément nanoscopique en phosphate de fer peut être soumis à un champ électrique afin de déplacer les électrons le long du réseau de nanotubes de carbone, s'infiltrant à travers les électrodes du phosphate de fer et ainsi de transférer rapidement l'énergie. Les batteries conçues à l'aide de virus ont, pour un même volume, une densité de charge 3 fois supérieure à celle d'une batterie au lithium-ion.

Par exemple, vous pouvez fixer une valeur de 1000 mA/h pour être sûr que l'(es) accumulateur(es) n'aura pas une charge supérieure à 1000 mA/h (de même pour la décharge, la valeur ne dépassera pas 1000 mA/h). Cette fonction est utile dans la phase de décharge car elle permet d'éliminer une quantité précise de la charge et de mettre l'(es) accumulateur(es) à un certain niveau. Par exemple lorsque l'on veut stocker sur une longue période des accumulateurs au lithium, on les décharge à environ 30% de la capacité maximale.

- **Protection thermique « Delta-T »** : un cycle de charge ou de décharge est arrêté lorsque la température de l'accumulateur atteint une valeur de consigne. Ce mode est automatiquement activé en présence du capteur de température externe, il est essentiel d'éviter d'endommager l'accumulateur à cause d'une surchauffe pendant le cycle de charge ou de décharge.

Normalement cette protection intervient lorsque vous rechargez des accumulateurs qui ne sont pas prévus pour le mode rapide, ou des accumulateurs de types inconnus dont vous ne connaissez pas les caractéristiques et que vous ne voulez pas endommager.

Lorsque vous allumez l'appareil, l'écran LCD affiche le nom du projet « **CBAT2550** » et la version du firmware.

Au bout de 2 secondes le chargeur est prêt à fonctionner. Initialement, l'affichage reprend les valeurs de la dernière utilisation, donc si vous utilisez toujours les mêmes types d'accumulateurs, cela vous fera gagner du temps en évitant à chaque fois de régler l'appareil.

Le premier paramètre, sélectionnable avec les boutons « → » et « ← » gère la sélection du mode de fonctionnement :

```
Select function
Charge Ni-Cd
```

Ici le chargeur peut être utilisé pour recharger des accumulateurs **Ni-Cd** (nickel-cadmium). La fin de la charge s'effectue par la méthode « **Delta-Peak** », dont la valeur peut être réglée dans le menu de configuration à l'aide du paramètre « **Peak NiCd** ».

```
Select function
Charge Ni-Mh
```

Cette fonction est utilisée pour recharger des accumulateurs **Ni-Mh** (nickel-hydrure métallique). La fin de la charge s'effectue par la méthode « **Delta-Peak** » dont la valeur est définie dans le menu de configuration avec le paramètre « **Peak NiMh** ».

```
Select function
Charge Li-Po
```

Cette fonction est utilisée pour recharger des accumulateurs **Li-Po** (lithium-polymère). La charge s'effectue selon la méthode **CC/CV**.

```
Select function
Charge Li-Io
```

Cette fonction est utilisée pour recharger des accumulateurs **Li-Io** (lithium-ion). La charge s'effectue selon la méthode **CC/CV**.

```
Select function
Charge Li-Fe
```

Cette fonction est utilisée pour recharger des accumulateurs **Li-Fe** (lithium

fer phosphate). La charge s'effectue selon la méthode **CC/CV**.

```
Select function
Charge Pb
```

Cette fonction est utilisée pour recharger des batteries au **plomb** (Pb).

```
Select function
Charge Time
```

Dans ce mode, la **fin de la charge se produit après un temps prédéfini**, indépendamment de la valeur de la tension de l'accumulateur. Ce mode est **adapté pour la charge lente** (1/10 C) des accumulateurs de types **Ni-Cd** et **Ni-MH**, dans tous les cas où la détermination du « **Delta-Peak** » présente une difficulté en raison de la présence d'un important « **effet mémoire** » dû à des accumulateurs très vieux ou partiellement endommagés.

```
Select function
DisChg Ni-xx
```

Ce mode est utilisé pour la **décharge** des accumulateurs **Ni-Cd** et **Ni-MH**. Dans ce cas, la tension de fin de décharge est réglable via le paramètre « **Ni-xx Volt DsChg** » dans le menu de configuration.

```
Select function
DisChg Ni-xx
```

Ce mode est utilisé pour la **décharge** des accumulateurs **Li-Po**, **Li-Io**, **Li-Fe**. Dans ce cas, la tension de fin de décharge est réglable via le paramètre « **Li-xx Volt DsChg** » dans le menu de configuration.

```
Select function
Setup
```

Ce menu est utilisé pour définir certains paramètres de fonctionnement (voir ci-après). La confirmation de la fonction souhaitée est obtenue en appuyant sur la touche « **Enter** ».

```
IChg=2400mA
```

L'écran suivant permet le **réglage** du **courant de charge** ou **décharge** (IDisChg) en utilisant les boutons « + » et « - ». Le réglage s'incrémente par pas de 100 mA pour une échelle comprise entre 0 et 2 A. Au-delà de 2 A, le pas est de 200 mA. La confirmation du courant est toujours validée avec la touche « **Enter** ».

```
Check: 15s  Adj
Tok CCOf IDLimit
```

À ce moment, le chargeur effectue un test de (ou des) l'accumulateur(es) « **Check Battery** » pour vérifier la présence de (des) l'élément(s), la connexion correcte ainsi que la polarité et vérifie la compatibilité de la tension de (ou des) l'accumulateur(es) avec ses propres caractéristiques. Il détecte le nombre de cellules connectées en série (à partir du nombre indiqué avant la lettre « **s** », c'est le nombre de cellules) et la présence d'un capteur de température extérieure, dans ce cas il va lire la valeur de la température ambiante et affiche à l'écran « **Tok** » (température ok). Si la fonction « **arrêt de la charge** » (capacity cut-off) est activée, il apparaît à l'écran « **CCOf** ».

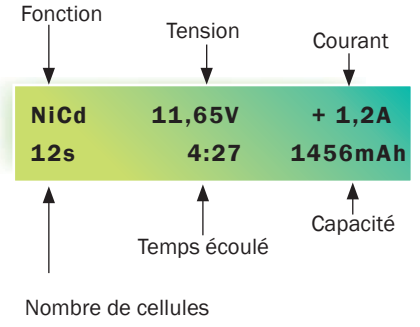
Si la fonction de décharge est activée, il vérifie la compatibilité du courant imposé avec le maximum autorisé affiché par le message « **IDLimit** ». En appuyant sur les boutons « + » et « - », vous pouvez corriger le nombre de cellules, qui plus particulièrement lors de la décharge, peut provoquer une erreur. Dans le cas d'accumulateurs de types **Li-xx**, vous devez être très prudent dans le paramétrage du nombre exact de cellules, sinon vous pouvez les endommager pendant la charge.

```
SafeTime 00:30
Tok CCOf IDLimit
```

Si les fonctions « **Safety Time** » (délai de sécurité) ou « **charge time** » (temps de charge) sont activées, vous serez invité à entrer le « **temps limite de charge** », et à confirmer en appuyant sur la touche « **Enter** ». Utilisez les boutons « + » et « - » pour augmenter ou diminuer le temps par tranches de 30 minutes dans la limite de **25 heures**.

Si vous utilisez le capteur de température et définissez un temps très long, la détermination de la **fin de charge** aura lieu suite à une **surchauffe des accumulateurs**.

Maintenant, appuyez sur la touche « **Enter** » et les fonctions paramétrées s'exécutent. L'écran LCD affiche en temps réel la fonction (type d'accumulateur), le nombre de cellules, la tension, le courant, le temps et la capacité pendant le fonctionnement.



La fonction représente le type d'accumulateur qui peut être :

- **Ni-Cd** : charge d'accumulateurs de type nickel-cadmium ;
- **Ni-Mh** : charge d'accumulateurs de type nickel-hydrure métallique ;
- **Li-po** : charge d'accumulateurs de type lithium-polymère à courant constant ;
- **Li-Io** : charge d'accumulateurs de type lithium-ion à courant constant ;
- **Li-Fe** : charge d'accumulateurs de type lithium fer phosphate à courant constant ;
- **Li-CV** : charge d'accumulateurs de type **lithium à tension constante** ((Litio Constant Voltage) ;
- **Pb** : charge de batterie au plomb ;
- **Time** : temps de charge (Time) ;
- **DsNi** : décharge (Discharge) d'accumulateurs de type **Ni-xx** ;
- **DsLi** : décharge (Discharge) d'accumulateurs de type **Li-xx** ;

L'écran affiche également le nombre de cellules, la tension aux bornes de (ou des) l'accumulateur(es) et le courant qui y circule.

Le temps total est indiqué en minutes et secondes pour la première heure, puis en heures et minutes pour les heures suivantes.

Le chargeur mesure la valeur de la charge effective reçue par l'(les) accumulateur(es) pendant la phase de charge ou la valeur perdue par l'(les) accumulateur(es) lors de la phase de décharge. Cette valeur est très utile pour connaître, par exemple pendant la phase de décharge, combien d'énergie a été laissée dans l'(les) accumulateur(s).

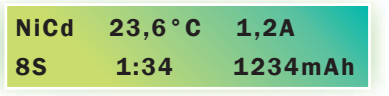
De même, lors de la phase de charge, il est possible de déterminer l'état de (ou des) l'accumulateur(es) avant de commencer la charge.

Par exemple, si vous rechargez complètement un accumulateur au polymère de 2000 mA/h et que la quantité de charge délivrée est de 1200 mA/h, vous pouvez déduire que l'accumulateur avait encore 800 mA/h de la charge précédente, c'est à dire qu'il était chargé à 40 %.

De même, si vous déchargez un accumulateur à 1200 mA/h et que vous obtenez une quantité de charge finale de 600 mA/h, cela signifie que l'accumulateur est encore chargé à 50 %.

Lorsqu'une charge (ou une décharge) est terminée, l'écran LCD affiche la dernière mesure effectuée et indique que la fonction est terminée :

- **DP_E** : charge complète obtenue à la suite d'un « **Delta-Peak** » (« **Delta-Peak End** ») ;
- **Li_E** : charge au lithium complète (« **Lithium End** ») ;
- **Pb_E** : charge au plomb terminée (« **Pb End** ») ;
- **TimE** : charge à durée (temps) déterminée complète (« **Time End** ») ;
- **TmpE** : charge arrêtée en raison d'une surchauffe (« **Temperature End** ») ;
- **DscE** : décharge terminée (« **Discharge End** ») ;
- **CCOf** : charge interrompue en raison de la capacité atteinte (« **Capacity Cut Off** ») ;
- **SafT** : charge arrêtée en raison de la durée atteinte du temps fixé (« **Safety Time** »).



Si la sonde de température externe est présente et appliquée sur l'accumulateur, la **température mesurée est affichée alternativement avec la tension toutes les 20 secondes**, dans ce cas seulement, la **fonction de protection thermique est activée** (« **Delta-T** »). **Il est important que le capteur de température soit bien en contact avec l'accumulateur**, sinon la lecture de la température sera erronée.

Il peut arriver que la tolérance des composants ne permette pas de lire correctement la valeur de la température, avec une erreur de quelques degrés. Cela n'est pas inquiétant, car ce qui compte, c'est la différence de valeur entre la température de l'accumulateur et la température ambiante.

Les réglages de certains paramètres peuvent être effectués à l'aide du menu de configuration (« **setup** »). Avec la touche « **Enter** », faites défiler les menus en utilisant le touches « **+** » et « **-** » pour modifier les paramètres affichés, qui peuvent être l'un des suivants :

- **Peak NiCd** : définit la valeur de la tension relative du « **Delta-Peak** » **d'une seule cellule** pour les accumulateurs de type **Ni-Cd**. Nous vous recommandons d'utiliser une valeur faible, qui pourra être augmentée si une détection prématurée du « **Delta-Peak** » se produit.

- **Peak NiMh** : définit la valeur de la tension relative du « **Delta-Peak** » d'une seule cellule pour les accumulateurs de type **Ni-Mh**. Normalement, vous devez utiliser **une valeur inférieure** à celle typique des accumulateurs de type **Ni-Cd**.

- **Delta-T Cut-Off** : définit le seuil de température qui provoque l'arrêt de la charge. Cette fonction est active uniquement en présence du capteur externe de température.

- **Ni-xx Volt DsChg** : représente la tension de décharge par cellule pour des accumulateurs de type **Ni-Cd** ou **Ni-Mh**.

- **Li-xx Volt DsChg** : représente la tension de décharge par cellule pour des accumulateurs de type **Li-Po**, **Li-Io**, **Li-Fe**.

- **Safety Time** (délai de sécurité) : permet d'activer ou de désactiver la fonction de protection de dépassement de la durée maximale. Si la fonction est activée dans le menu, un réglage vous est proposé et vous devez définir la durée maximale au-delà de laquelle la charge sera interrompue.

- **Capacity Cut-Off** (arrêt de la charge): permet de définir, en appuyant sur les touches « **+** » et « **-** », la valeur de la capacité maximale au-delà de laquelle la charge sera interrompue. Toute valeur non nulle implique que la fonction est active.

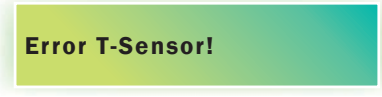
- **Beep sound** (bip) : active ou désactive le bip du buzzer.

Si une anomalie se produit, l'affichage fournit une indication quant à la nature du problème, au moyen de messages d'erreurs spécifiques.

Ce message s'affiche lorsque vous démarrez une charge ou une décharge et que l'accumulateur est **débranché** ou connecté avec une **polarité inversée**.

Ce message peut également apparaître, si l'accumulateur est très déchargé, ou s'il est en **court-circuit**.

Ce message indique que **la tension de sortie dépasse la valeur maximale autorisée** qui est égale à **25,5 V**.



Cette erreur s'affiche lorsque le capteur de température est accidentellement **débranché** pendant la charge ou qu'il fournit des mesures inexactes à cause d'un défaut.

Temp Over!

Ce message apparaît lorsque la **température interne du chargeur dépasse la valeur maximale autorisée**, éteignez le chargeur et laissez-le refroidir.

Short Circuit!

Ce message s'affiche lorsqu'au cours d'une charge ou d'une décharge, un **court-circuit est détecté** au niveau de l'accumulateur.

Va too Low!

Ce message ci-dessus peut apparaître lorsque vous branchez le chargeur à une batterie de voiture et que sa **tension est trop faible**, ou lorsque l'alimentation de puissance du chargeur est insuffisante.

Va too High!

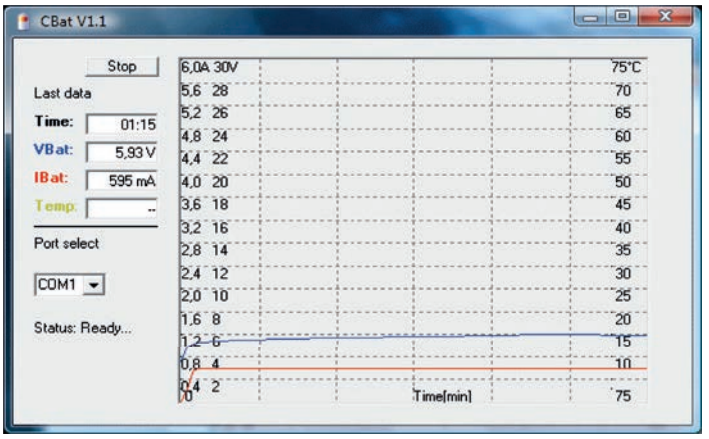
Cet avertissement apparaît si la **tension d'alimentation** du **chargeur** est **trop élevée**. Dans ce cas, vérifiez la tension fournie à la sortie de l'alimentation. Le **rétro-éclairage** de l'afficheur **LCD** est **allumé tout au long de la phase de réglage des paramètres** et **s'éteint automatiquement** au commencement d'une charge ou d'une décharge. Au cours d'une charge ou d'une décharge, en appuyant sur le bouton « + » vous allumez le rétro-éclairage de l'afficheur, tandis qu'en appuyant sur le bouton « - » vous éteignez le rétro-éclairage de l'afficheur. Cette fonction vous permet de visualiser au cours d'une charge ou d'une décharge l'état d'avancement du processus. Pendant une phase de charge ou de décharge, il est possible de quitter et de **retourner au menu principal** en appuyant sur la touche « **Enter** ».

Le logiciel enregistreur de données (data logger)

A chaque minute, le chargeur envoie au port série un paquet de données qui contient les valeurs de la tension, du courant et de la température, ainsi que l'heure selon la trame : **Sheure:minuteTtensionVcourantItemperatureM**. Les lettres sont utilisées pour séparer les valeurs numériques.

Par exemple, la chaîne suivante : **S2:45T1034V678A234M** indique que le temps écoulé est de 2 heures et 45 minutes, la tension est de 10,34 volts, le courant est de 678 mA, et la température est de 23,4 ° C. Les données peuvent être visualisées avec le programme **Hyper Terminal** de **Windows** ou d'autres émulateurs de terminaux pouvant gérer un port de communication **RS232** série à 9600 bps, 8 bits de données, pas de parité et un bit d'arrêt (**9600,8, N, 1**).

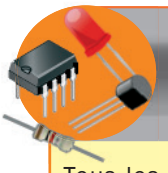
Malheureusement les autres programmes, tel que **Hilgraeve** (qui est le concepteur d'Hyper Terminal de Windows **mais qui n'est plus disponible dans Windows Vista, 7 et 8**) est en version payante. La méthode la plus simple, si vous avez un **Windows XP** sous la main, est de récupérer directement **HyperTerminal** sur le PC. Pour cela il suffit de récupérer les deux fichiers **hypertrm.dll** et **hypertrm.exe**. Copiez simplement ces deux fichiers sur votre machine et cela fonctionnera. Les fichiers se trouvent normalement dans C:\Program Files\Windows NT pour hypertrm.exe dans C:\Windows\System32 pour hypertrm.dll. Nous les mettrons à disposition sur notre site avec les programmes du chargeur. Nous avons aussi développé une application spécifique fonctionnant sous Windows qui se nomme **CABT.msi**. Pour démarrer l'installation, il suffit de cliquer deux fois sur le programme.



Au démarrage du programme « Cbat.exe », l'écran initial apparaît, appuyez sur le bouton « Start » pour démarrer l'acquisition des données.

Une fois l'installation du programme terminée, vous pouvez lancer l'application, l'écran principal apparaît. Branchez le chargeur au port série du PC (via un câble série) et lancez la capture à l'aide du bouton « **Start** » (Démarrer).

Le logiciel permet de visualiser sur un PC un graphique montrant l'évolution des variables représentant les grandeurs mesurées par le chargeur, c'est à dire la tension, le courant et la température. Le graphique est de type « redimensionnement automatique » et permet d'afficher avec précision aussi bien les valeurs acquises en 30 minutes que celles acquises en 10 heures. L'affichage peut être redimensionné pour mieux analyser certains détails et vous pouvez positionner un marqueur à l'aide de la souris, ce qui vous permet de lire des données à n'importe quel point du graphique.



Comment construire ce montage

Tous les composants sont disponibles auprès de nos annonceurs, voir les publicités. Les circuits imprimés, le programme MF810.hex du microcontrôleur PIC, la solution Hyper Terminal et l'application CABT2550.msi sont disponibles en téléchargement gratuit sur notre site www.electroniquemagazine.com dans la section « Sommaire détaillé » du numéro 126 à l'onglet « Télécharger ».

Caméra couleur HD pour enregistrements vidéo

- Caméra portable livrée avec support ventouse
- Enregistrement sur carte SD (pas incl.)
- Enregistrement vidéo HD
- Microphone incorporé
- Fonction d'enregistrement automatique en démarrant le moteur de la voiture
- Facile à configurer à l'aide des boutons-poussoirs
- Livrée avec câble USB, câble AV et support de fixation sur vitre
- Afficheur TFT couleur de 2.5"

68,23 €
réf. CAMCOLVC15

NEW

Revivez vos grands moments d'émotions

Action tournage sans limites!

Idéal pour la prise de vue sur les mouvements en continu, pour des activités sportives, à des fins d'assurance ou de plaisir. Installée sur les tableaux de bord des voitures, casques ou sur le guidon de motos et vélos.

Caméra couleur HD

- Caméra portable livrée avec plusieurs étriers de montage
- Enregistrement sur microSD (32Go, non incl.)
- Enregistrement vidéo et images HD
- Capacité de mémoire interne de 64Mo
- Format d'enregistrement: MPEG-4
- Capteur: CMOS 5 mégapixels
- Microphone incorporé très sensible
- Dimensions: 105 x 45 x 37mm
- Alimentation: 5VDC, câble USB et 2 piles au lithium incl.
- Livrée avec étrier de montage pour vélo, étrier de fixation sur vitre et sangle.

132,44 €
réf. CAMCOLVC13

Caméra vidéo couleur

- Caméra portable et étanche
- Capacité de mémoire interne de 64Mo
- Enregistrement sur carte microSD (max. 8Go non incl.)
- Objectif de la caméra: 8.3mm (F3.0)
- Fonction webcam
- Microphone intégré, haute sensibilité
- Sortie A/V pour connexion à un téléviseur
- Livrée avec câble USB, câble A/V, étrier pour vélo, bracelet
- Dimensions Ø31 x 101 mm
- Alimentation 5VDC via câble USB et pile au lithium 380mAH (interne)

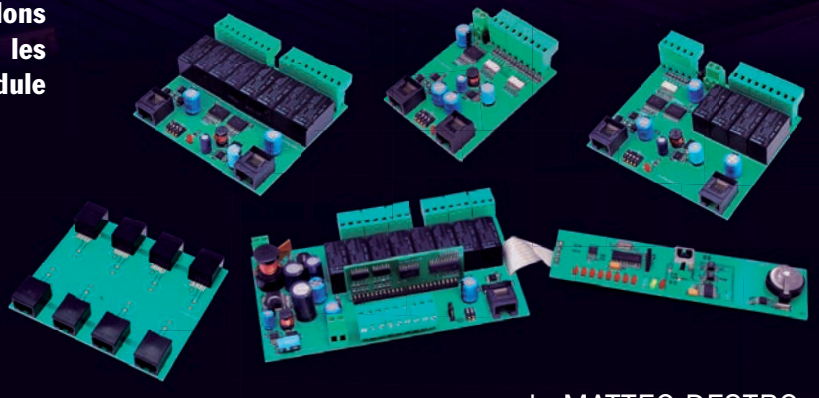
96,00 €
réf. CAMCOLVC14

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

MINIBUS, UN BUS POUR L'AUTOMATISATION

3ème partie

Nous continuons la description du système domotique MINIBUS flexible qui est basé sur le protocole de communication I2C. Nous allons vous décrire dans ce troisième article les modules périphériques et le firmware du module MASTER (maître).



de MATTEO DESTRO

Dans les articles précédents, nous avons décrit en détail les modules **MASTER** (maître) et **SLAVE** (esclave). Pour rappel vous pouvez voir à la figure 1 le schéma de fonctionnement du système **MINIBUS**. Dans cet article, nous complétons la description de la carte à 4 entrées et 4 sorties sur relais et celle de la carte **HUB**. Nous allons introduire quelques concepts de base utilisés pour le développement du firmware présent dans le **PIC18F2620** qui est monté sur l'unité **MASTER**. En outre, le module **MASTER** doit être configuré et adapté pour traiter toutes les entrées et les sorties qui sont connectées directement ou indirectement par l'intermédiaire des unités **SLAVE** (esclaves). Pour configurer le module, vous devez utiliser un programme spécial développé pour la plateforme **Windows**, qui vous permet de personnaliser chaque fonction. Le logiciel en question s'appelle **MiniBusTools** dont nous analyserons chaque fonction dans le prochain et dernier article, afin de donner au lecteur une vue d'ensemble complète du potentiel du système. Maintenant, complétons la description de la dernière carte **SLAVE**.

Schéma de fonctionnement

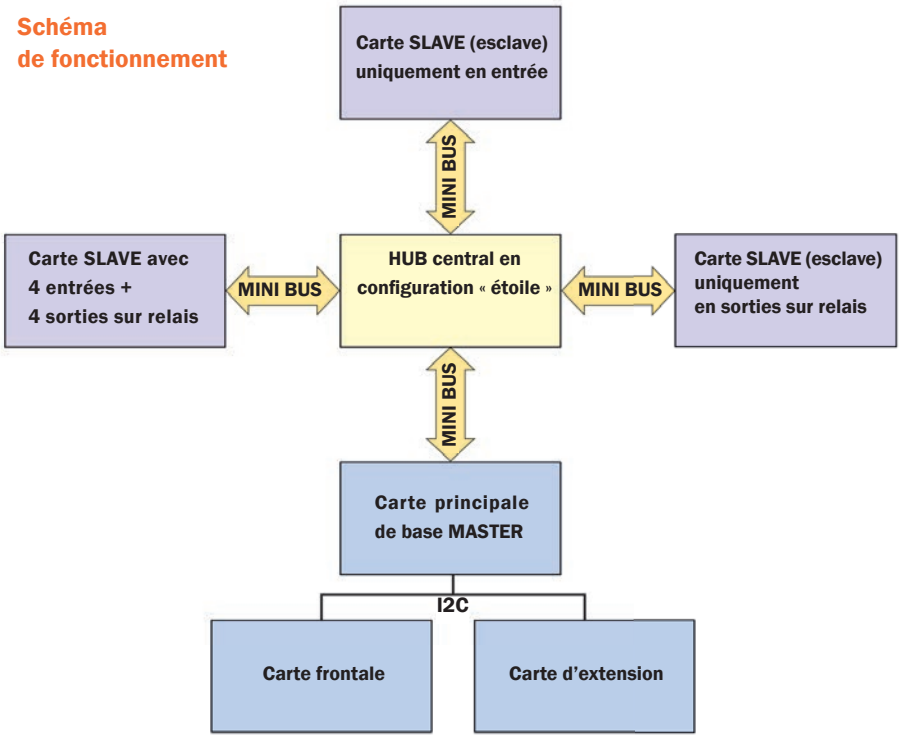


Figure 1 : dans le schéma de fonctionnement ci-dessus, vous pouvez voir les différentes connexions entres les modules du système. L'unité de commande est composée d'un module contenant une carte de base MASTER, dotée d'une carte principale et d'une carte d'extension, reliées entre elles par le protocole I2C. L'unité de commande est connectée à la carte esclave (SLAVE) par l'intermédiaire du HUB central en utilisant le MINIBUS. Les différentes cartes esclaves sont reliées entre elles, ainsi qu'à la carte principale (MASTER) par l'intermédiaire du HUB central. On peut connecter jusqu'à un maximum de 7 cartes esclaves (SLAVE) dans n'importe quelle combinaison entre les différents types disponibles : cartes esclaves en sorties sur relais, cartes esclaves en entrées uniquement, et cartes esclaves avec 4 entrées et 4 sorties sur relais.

Module SLAVE à 4 entrées et 4 sorties sur relais

Il ne reste plus qu'à traiter le dernier type de module **SLAVE** (esclave) qui est une combinaison des deux modules précédents (voir le schéma électrique du module **SLAVE** « **4 IN 4 OUT** » sur la figure 2), c'est-à-dire qu'il dispose de 4 entrées opto-isolées et de 4 sorties sur relais. Sur figures **2a**, **2b** et **2c** vous trouverez les deux faces du circuit imprimé à l'échelle 1 : 1, ainsi que le schéma de câblage des composants du module **SLAVE** « **4 IN 4 OUT** ». Comme pour les modules précédents, vous trouverez un circuit intégré **P82B96** (U1) adaptateur de niveau pour le **bus I2C**, un circuit intégré **MCP23008** (U4) qui est un circuit d'extension des entrées/sorties et la section alimentation autour d'un circuit intégré LM2674M5.0 (U3).

Concernant le circuit d'extension des entrées/sorties (U4), il est configuré de telle manière que la moitié des broches du port sont des « entrées » (GP0 à GP3) et l'autre moitié des broches du port sont des « sorties » (GP4 à GP7).

Les entrées, comme pour les autres modules, disposent d'une isolation galvanique grâce au circuit intégré

Liste des composants du module SLAVE « 4 IN 4 OUT »

R1..... 2,2 kΩ 1% boîtier 0603 CMS
R2..... 2,2 kΩ 1% boîtier 0603 CMS
R3..... 4,7 kΩ 1% boîtier 1206 CMS
R4..... 4,7 kΩ 1% boîtier 0603 CMS
R5..... 3,3 kΩ 1% boîtier 1206 CMS
R6..... 1,5 kΩ 1% boîtier 1206 CMS
R7..... 4,7 kΩ 1% boîtier 1206 CMS
R8..... 4,7 kΩ 1% boîtier 0603 CMS
R9..... 1,5 kΩ 1% boîtier 1206 CMS
R10.... 4,7 kΩ 1% boîtier 1206 CMS
R11 ... 4,7 kΩ 1% boîtier 0603 CMS
R12 ... 1,5 kΩ 1% boîtier 1206 CMS
R13 ... 4,7 kΩ 1% boîtier 1206 CMS
R14.... 4,7 kΩ 1% boîtier 0603 CMS
R15 ... 1,5 kΩ 1% boîtier 1206 CMS
R16.... 4,7 kΩ 1% boîtier 0603 CMS
R17.... 4,7 kΩ 1% boîtier 0603 CMS
R18 ... 4,7 kΩ 1% boîtier 0603 CMS
R19 ... 4,7 kΩ 1% boîtier 0603 CMS
R20 ... 2,2 kΩ 1% boîtier 0603 CMS
R21.... 2,2 kΩ 1% boîtier 0603 CMS
R22 ... 4,7 kΩ 1% boîtier 0603 CMS
R23 ... 4,7 kΩ 1% boîtier 0603 CMS
R24.... 4,7 kΩ 1% boîtier 0603 CMS
C1..... 220 µF 50 V électrolytique
C2..... 100 nF 50 V céramique boîtier 0603 CMS
C3 ..... 220 µF 50 V électrolytique
C4..... 10 nF 50 V céramique boîtier 0603 CMS
C5..... 100 µF 25 V électrolytique
C6..... 100 nF 50 V céramique boîtier 0603 CMS
C7..... 100 µF 25 V électrolytique
C8..... 100 nF 50 V céramique boîtier 0603 CMS
C9..... 100 nF 50 V céramique boîtier 0603 CMS
C10.... 220 µF 50 V électrolytique
C11.... 100 nF 50 V céramique boîtier 0603 CMS
C12.... 100 nF 50 V céramique boîtier 0603 CMS

D1..... SMBJ13CA
D2..... SMBJ13CA
D3..... SS34
LD1.... LED 3 mm rouge

U1..... P82B96
U2 TLP280
U3..... LM2674M_5.0
U4..... MCP23008
U5..... ULN2803ADW
L1..... Inductance 100 µH 800 mA
DS1 ... Dip-Switch 4 contacts
RL1.... Relais 12V JW1FSN
RL2.... Relais 12V JW1FSN
RL3.... Relais 12V JW1FSN
RL4.... Relais 12V JW1FSN
F1..... Fusible réarmable polyswitch 50 mA
F2..... Fusible réarmable polyswitch 50 mA
Divers

CN1 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN2 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
J1..... Barrette 3 contacts + jumper
Bornier 2 pôles au pas de 5,08 mm (x 2)
Bornier 3 pôles au pas de 5,08 mm (x 4)

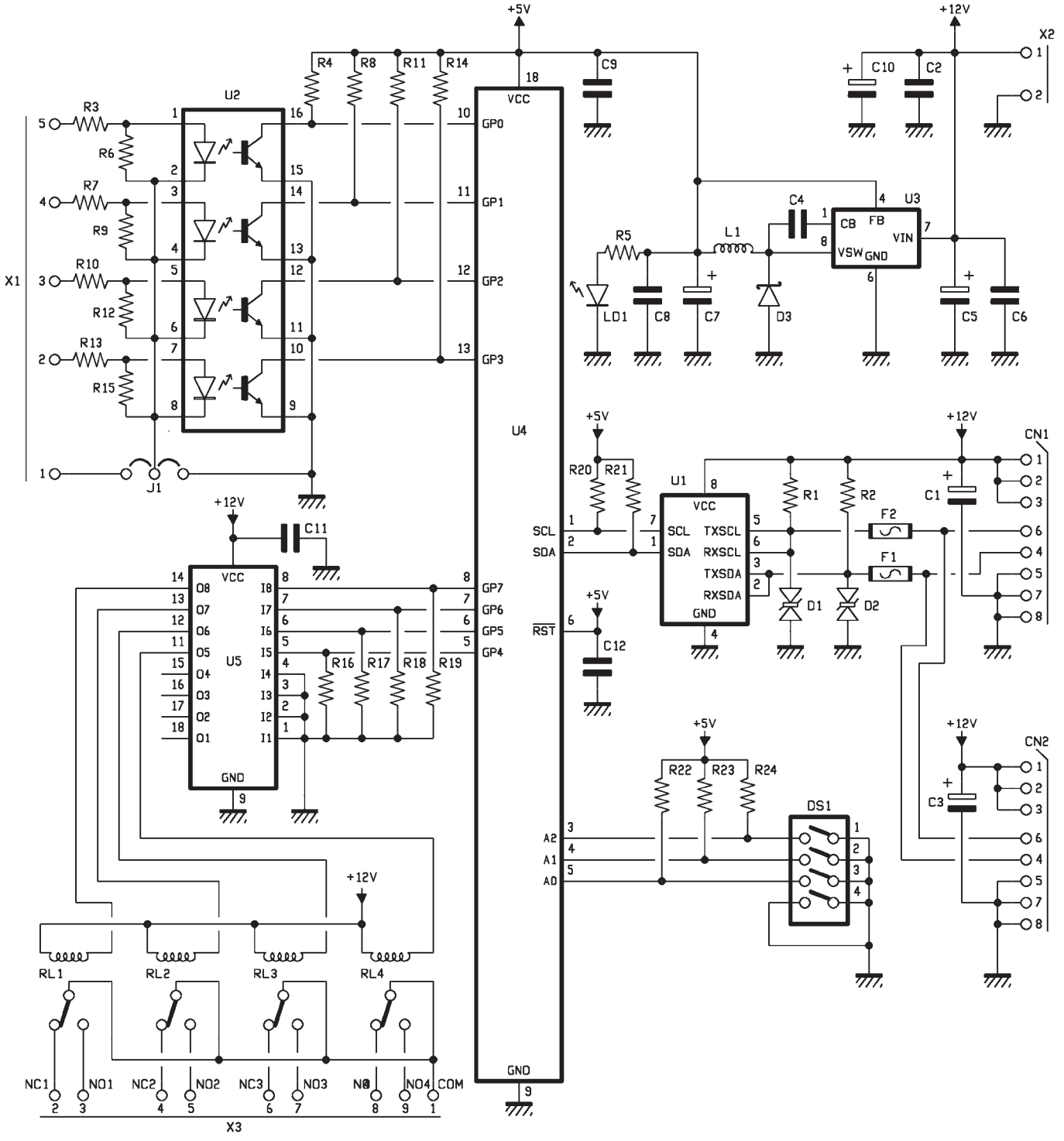


Figure 2 : schéma électrique du module « 4 IN 4 OUT », il dispose de 4 entrées opto-isolées et de 4 sorties sur relais.

TLP2804 (U2), tandis que les sorties sur relais sont gérées par le « driver » de puissance ULN2803ADW (U5).

Il est important de noter que les entrées I1 à I4 sont connectées à la masse car elles ne sont pas utilisées, ces broches ne doivent pas rester flottantes car cela pourrait perturber le fonctionnement du circuit U5.

Par contre les entrées I5 à I8 sont dotées de résistances de tirage (« pull-up ») pour assurer une valeur logique stable dans le cas où le circuit **MCP23008** ne soit pas correctement configuré, ou soit endommagé ou mal monté. De même pour le circuit U4, les entrées GP0 à GP3 sont dotées de résistances de tirage (« pull-up ») pour assurer une valeur logique stable dans le cas où le circuit TLP2804 (U2) ne soit pas monté ou endommagé.

Le bornier X1 est utilisé pour la connexion des entrées aux broches 2 à 5 (interrupteur/bouton poussoir). La broche 1 est utilisée pour relier à la masse les entrées dans le cas où elles sont alimentées par une alimentation externe à la carte, dans ce cas, le « jumper »

(cavalier) J1 doit être positionné sur « 1-2 ». Le bornier X2 alimente le circuit à l'aide d'une tension stabilisée de 12 VDC. Si vous utilisez la tension d'alimentation de la carte pour les entrées, le « jumper » (cavalier) J1 dans ce cas, doit être positionné sur « 2-3 ». Le bornier X3 est utilisé pour connecter les charges aux relais respectifs. Le commun est relié à la broche 1 du bornier.

HUB central en configuration « étoile »

Nous utilisons un module « **HUB** » central (aussi appelé concentrateur, c'est un appareil permettant d'interconnecter électriquement plusieurs appareils, typiquement des ordinateurs ; connexions réseau Ethernet via hub Ethernet ; ou encore des périphériques) en configuration « étoile » qui nous permet de créer un point central pour la connexion des modules utilisés par le système.

L'électronique du module « **HUB** » se limite à la connexion en parallèle les uns avec les autres, de 8 connecteurs RJ45 et de la présence de 8 condensateurs de filtrage de 220 µF/25 V sur la ligne d'alimentation 12 VDC. Vous pouvez voir le schéma électrique du module « **HUB** » sur la figure 3, vous remarquerez sa simplicité. Dans tous les modules **SLAVE** (esclave), comme vous avez pu le constater, nous avons utilisé comme circuit d'extension des entrées/sorties un **MCP23008** qui appartient à la même famille que le **MCP23017** utilisé dans le module **MASTER** (maître).

Dans la pratique un **MCP23008** dispose de la moitié des entrées/sorties disponibles dans un **MCP23017** (8 I/O contre 16 I/O). Dans ce cas, les registres sont mappés à partir de l'adresse 0x00 jusqu'à l'adresse 0x0A et il n'est pas nécessaire de définir sur quel registre travailler comme pour le **MCP23017**.

Nous ne décrirons pas en détail la configuration de chaque registre, mais uniquement les plus pertinents pour notre application. Nous partons du registre « IODIR » qui configure la « direction » des broches (pin) du port en entrées ou en sorties, donc selon le type de module **SLAVE** (esclave) nous aurons :

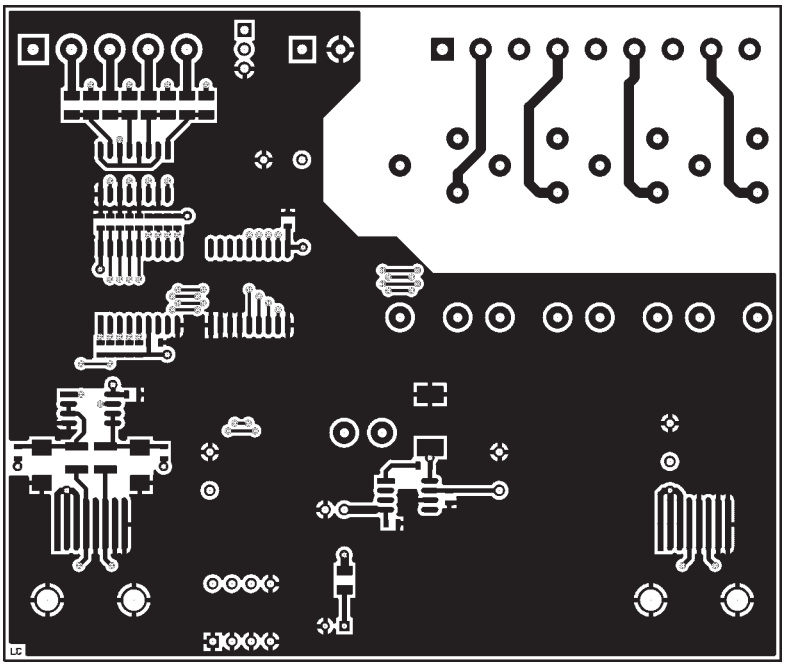


Figure 2a : circuit imprimé à l'échelle 1 : 1 du module SLAVE « 4 IN 4 OUT » côté composants.

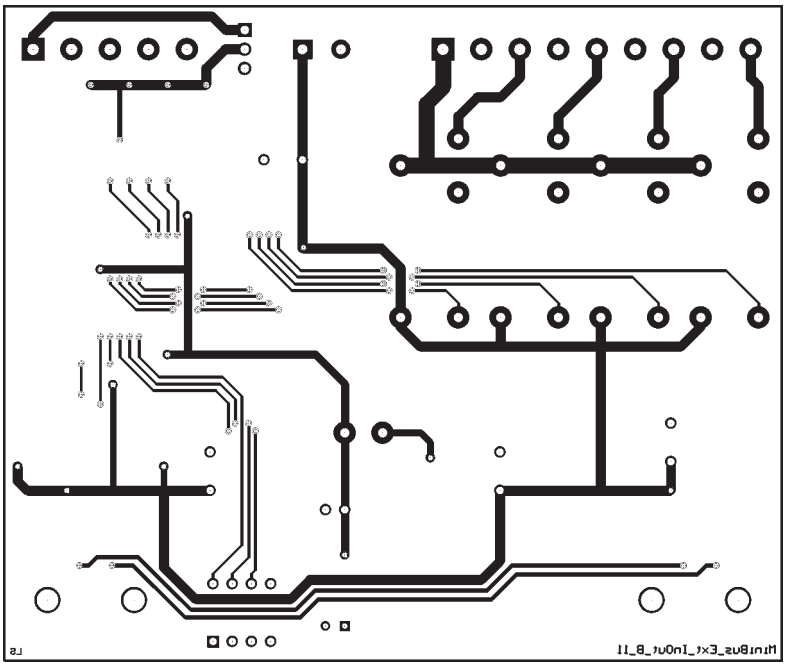


Figure 2b : circuit imprimé à l'échelle 1 : 1 du module SLAVE « 4 IN 4 OUT » côté soudures.

- module **SLAVE** en entrée seulement : **IODIR = 0xFF**
- module **SLAVE** en sortie seulement : **IODIR = 0x00**
- module **SLAVE** en entrée et en sortie : **IODIR = 0x0F**

Le registre « **IPOL** » détermine le « signe » de la valeur lue par le registre **GPIO**, en d'autres termes, si le bit du registre **IPOL** est défini à « 1 » cela signifie que la valeur lue par le port **GPIO** est l'inverse de la valeur réelle qui est présente sur le port, à l'inverse si le bit du registre **IPOL** est à « 0 » le port **GPIO** lit la valeur réelle. Cette méthode est utilisée pour décider comment les « entrées » doivent travailler, c'est à dire si elles doivent être actives sur un front montant ou sur un front descendant. La valeur attribuée au registre **IPOL** est paramétrée par l'utilisateur via le logiciel.

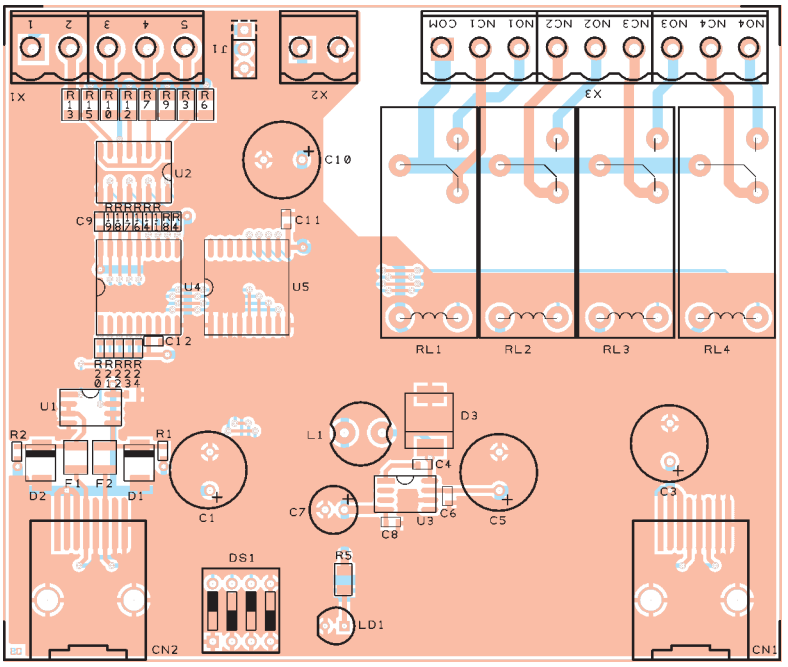


Figure 2c : schéma de câblage des composants du module SLAVE « 4 IN 4 OUT ».

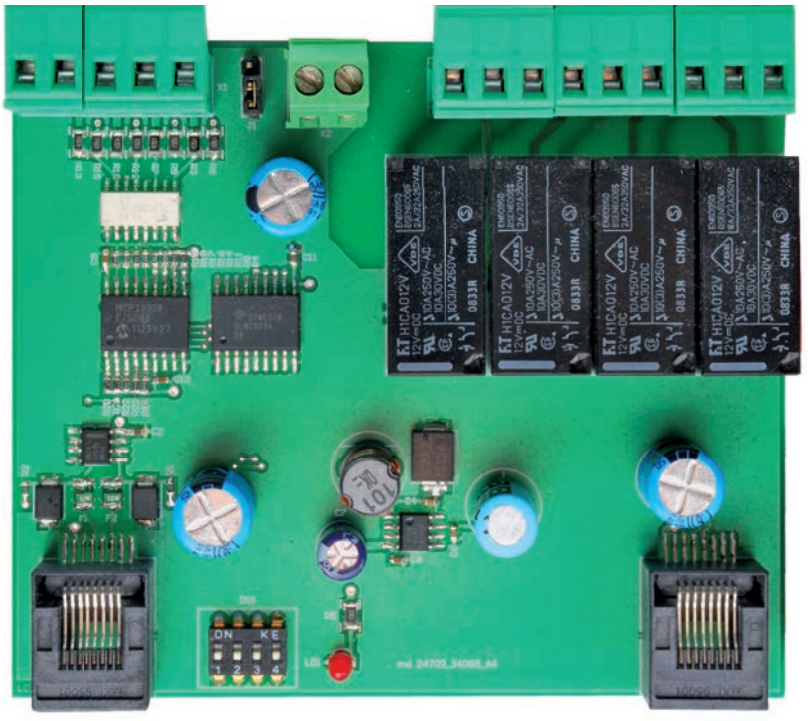
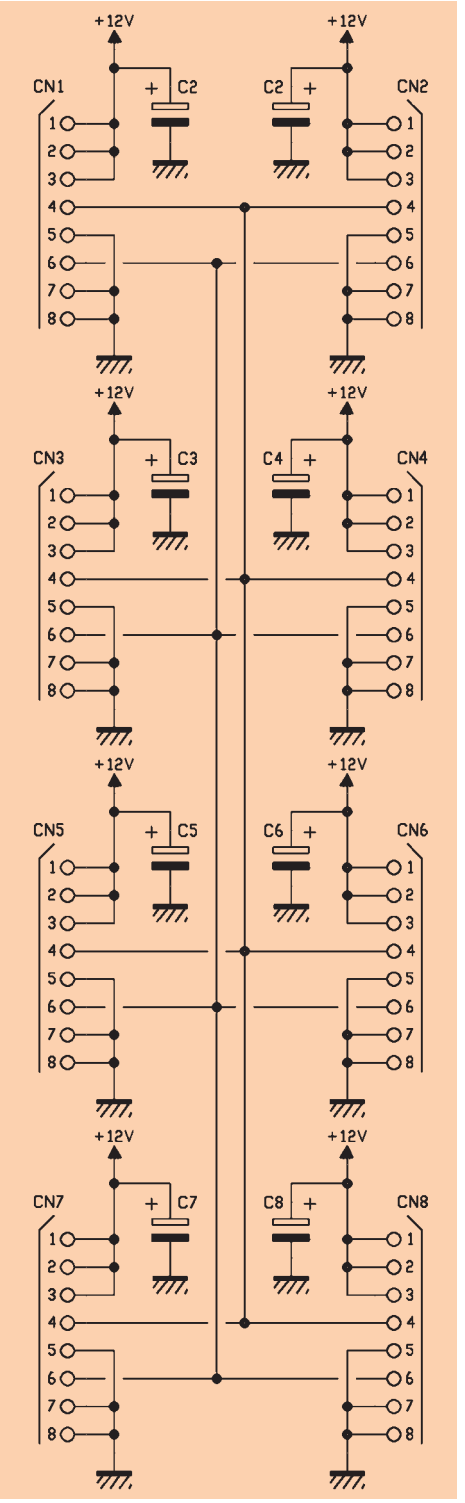


Photo de l'un de nos prototypes du module SLAVE « 4 IN 4 OUT ».

Figure 3 : schéma électrique du module « 4 IN 4 OUT », il dispose de 4 entrées opto-isolées et de 4 sorties sur relais.



Liste des composants du module HUB central

C1..... 220 µF 25 V électrolytique
C2..... 220 µF 25 V électrolytique
C3 ..... 220 µF 25 V électrolytique
C4..... 220 µF 25 V électrolytique
C5..... 220 µF 25 V électrolytique
C6..... 220 µF 25 V électrolytique
C7..... 220 µF 25 V électrolytique
C8..... 220 µF 25 V électrolytique
CN1 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN2 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN3 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN4 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN5 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN6 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN7 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889
CN8 ... connecteur RJ45 pour circuit imprimé Molex 95501-6889

Cette configuration est appliquée sur les modules **SLAVE** n'ayant que des « entrées » ou sur les modules mixtes ayant des « entrées/sorties ». Le registre **GPIO** traduit la valeur actuelle au port lorsqu'il est configuré en « entrée » ou une partie de ce port dans le cas d'un module mixte (« entrées/sorties »).

Enfin, le registre **OLAT** indique la valeur à affecter au port lorsqu'il est configuré en « sortie ». Le contenu du registre dépend de la valeur qui est assignée à la sortie à ce moment-là, ou si la sortie doit être activée ou non. La valeur assignée à la sortie dépend de deux facteurs : le premier est l'état de la sortie (active ou inactive), le second est la configuration attribuée à chaque sortie.

Pour bien comprendre le concept, notez bien que toutes les sorties sont considérées comme de simples relais et l'utilisateur peut décider si elles doivent travailler avec un contact normalement fermé (NC) ou normalement ouvert (NO).

Selon le choix effectué par l'utilisateur à travers le logiciel, un masque (un état, une image) des sorties va être généré et appliqué au registre **OLAT**. Le masque est appliqué à l'aide d'une opération booléenne (XOR) parmi la valeur à affecter au registre et au masque lui-même.

Le module **SLAVE** (esclave) qui comporte uniquement des entrées, est conçu pour être inséré dans un boîtier de type rail DIN référence 4MH53 (http://www.mix-elec.fr/article-7830-4MH53-5.html). Les modules **SLAVE** (esclaves) qui comportent uniquement des sorties ou des entrées/sorties, sont conçus pour être insérés dans un boîtier de type rail DIN référence 6MH53.

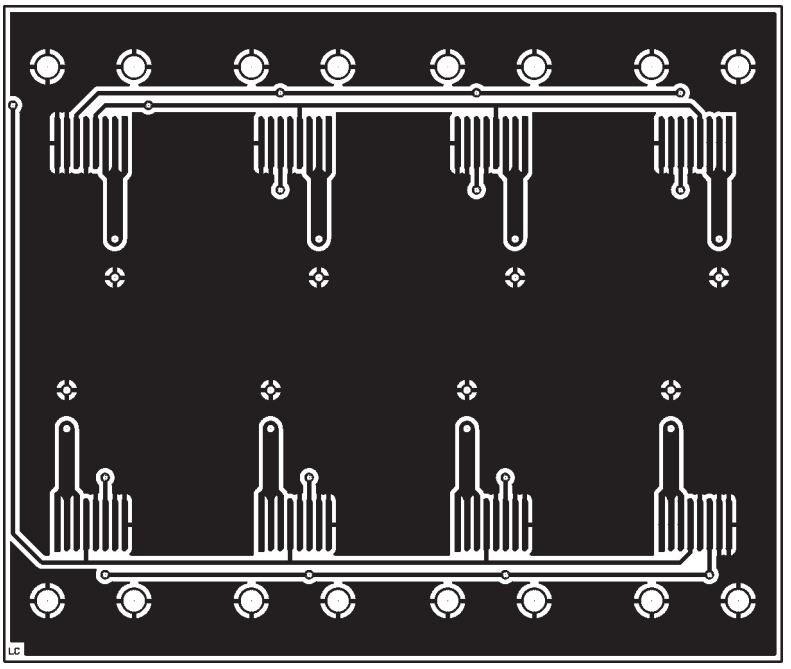


Figure 3a : circuit imprimé à l'échelle 1 : 1 du module HUB central côté composants.

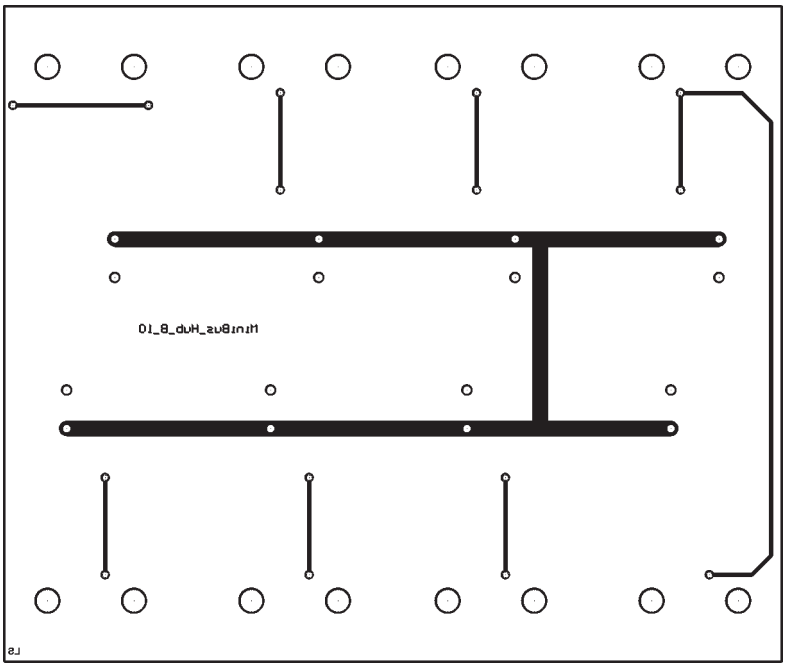


Figure 3b : circuit imprimé à l'échelle 1 : 1 du module HUB central côté soudures.

Description du firmware

Comme d'habitude, faisons un bref panoramique de l'étude du firmware. Compte tenu de la complexité du projet, nous avons décidé d'écrire directement le code en langage **ANSI C**, ce qui permet de mettre en œuvre et de déboguer un code complexe avec beaucoup plus de facilité que le langage assembleur.

Comme avec le langage assembleur, vous avez la possibilité d'interagir directement avec les périphériques du **PIC18F2620**, en fonction des besoins nécessaires pour la configuration et la gestion du projet. De plus, le **langage C** nous permet de mettre en œuvre et de gérer des structures de données complexes et imbriquées avec beaucoup plus de facilité et de rapidité que l'assembleur. La figure 5 montre une structure de données imbriquées utilisée dans le projet pour la gestion d'intervalles de temps.

Actuellement, la quantité de code utilisée pour l'exécution des fonctions du programme occupe environ 1/3 de la totalité de la mémoire **FLASH**, tandis que l'utilisation de la mémoire **RAM** s'élève à environ la moitié de celle disponible (voir la figure 6). Nous pouvons donc facilement ajouter de nouvelles fonctionnalités au système. La figure 4, montre comment le projet a été structuré et comment le code a été divisé en

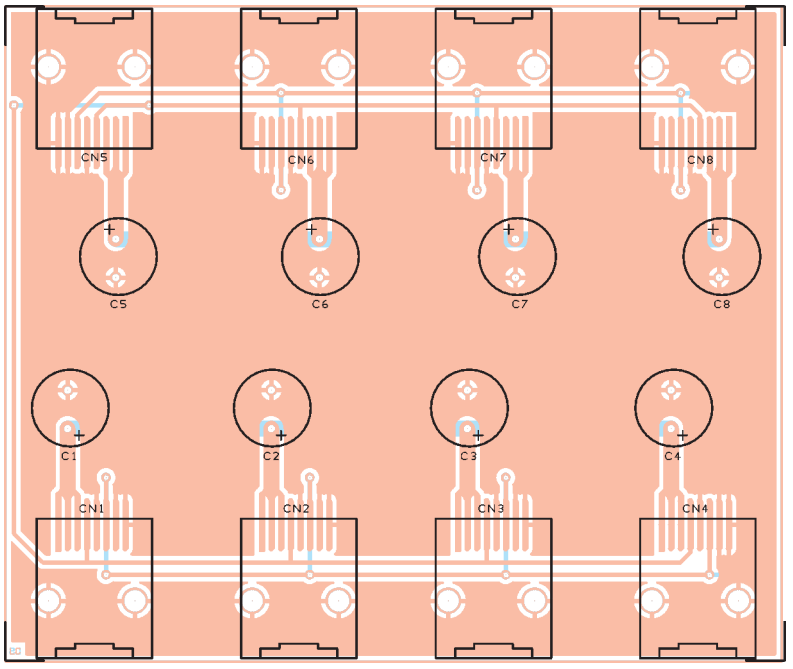


Figure 3c : schéma de câblage des composants du module HUB central.

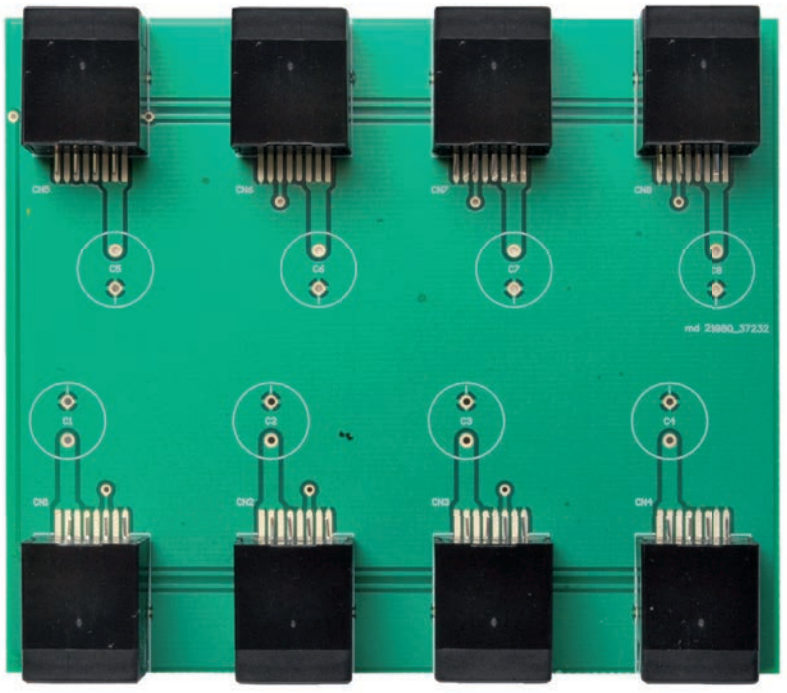


Photo de l'un de nos prototypes du module HUB central.

plusieurs fichiers afin d'avoir une gestion ordonnée de l'ensemble du projet.

Chaque nom attribué à un fichier « **. c** » appelle sans équivoque les fonctions qui sont mises en œuvre. Ainsi chaque fichier « **. c** » dispose d'un fichier équivalent « **. h** » contenant toutes les déclarations des fonctions mises en œuvre et, éventuellement, les déclarations des constantes nécessaires à la mise en œuvre du code.

Voici par ordre d'importance, une description des fichiers utilisés dans le code et qui remplissent les tâches suivantes :

« **main.c** » : contient le code qui appelle et gère les fonctions nécessaires au bon fonctionnement de l'ensemble du système. Après l'initialisation du microcontrôleur, le firmware entre dans une boucle infinie en appelant de manière cyclique toutes les fonctions disponibles.

« **inits.c** » : contient le code qui initialise tous les périphériques du **PIC18F2620** y compris l'initialisation de la mémoire RAM.

« **init_eeprom.c** » : contient le code d'initialisation de la mémoire **EEPROM** interne du **PIC18F2620**. Ce sont les réglages d'usine.

« **init_flash.c** » : contient la fonction de lecture de la mémoire FLASH du **PIC18F2620**, avec en plus une chaîne d'initialisation qui est stockée dans la mémoire FLASH à une adresse bien précise.

« **isr_management.c** » : contient une seule fonction qui est appelée chaque fois qu'une interruption est générée. Quand elle est active, elle appelle différentes sous-fonctions de manière individuelle gérées par les interruptions, qui seront exécutées si et seulement si, l'interruption correspond, sinon elles seront ignorées.

« **isr_timer.c** » : contient des fonctions liées à la gestion de l'interruption « TIMER » mis à la disposition du **PIC18F2620**.

« **isr_i2c.c** » : contient la fonction de gestion de l'interruption liée au périphérique I2C. Dans ce fichier, sont stockées toutes les fonctions nécessaires pour gérer les périphériques I2C utilisés dans notre projet.

« **isr_uart.c** » : contient la fonction de gestion des interruptions liées au périphérique SCI (UART). Dans ce fichier, sont stockées toutes les fonctions nécessaires à la gestion de la communication série entre le PC et le **PIC18F2620**.

« **debouncing.c** » : contient la fonction « deboucing » (anti-rebond) des entrées et la gestion relative. Met en corrélation les entrées avec les sorties correspondantes, si elle est programmée.

« **eeprom.c** » : contient les fonctions de lecture et d'écriture de la mémoire **EEPROM** interne du **PIC18F2620**.

« **led_management.c** » : contient les fonctions de gestion des LED du module **MASTER**.

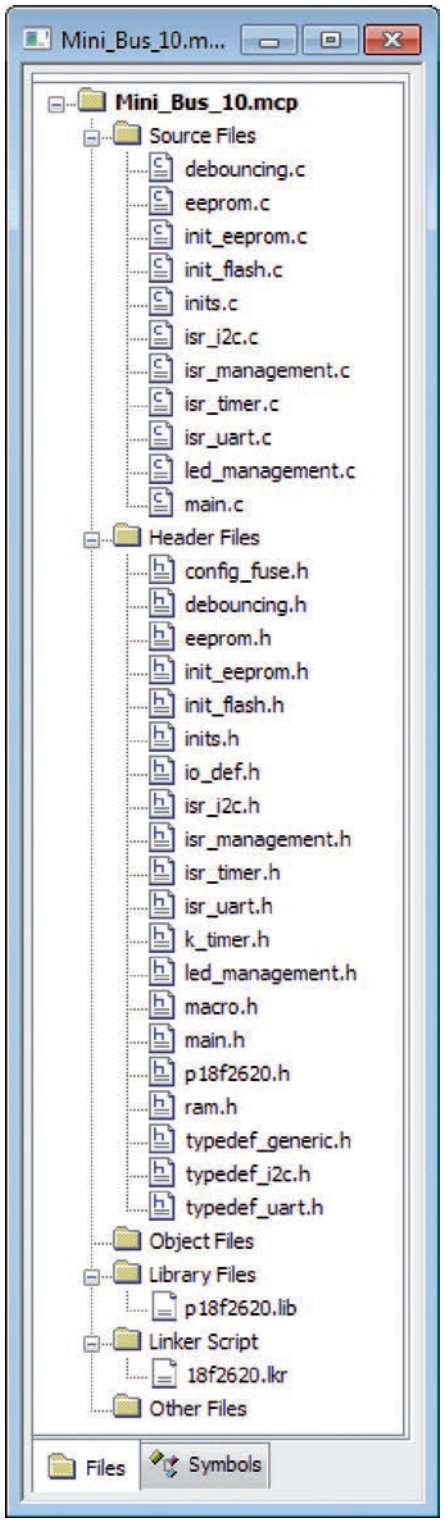


Figure 4 : structure du code de l'ensemble du projet divisée en plusieurs fichiers.

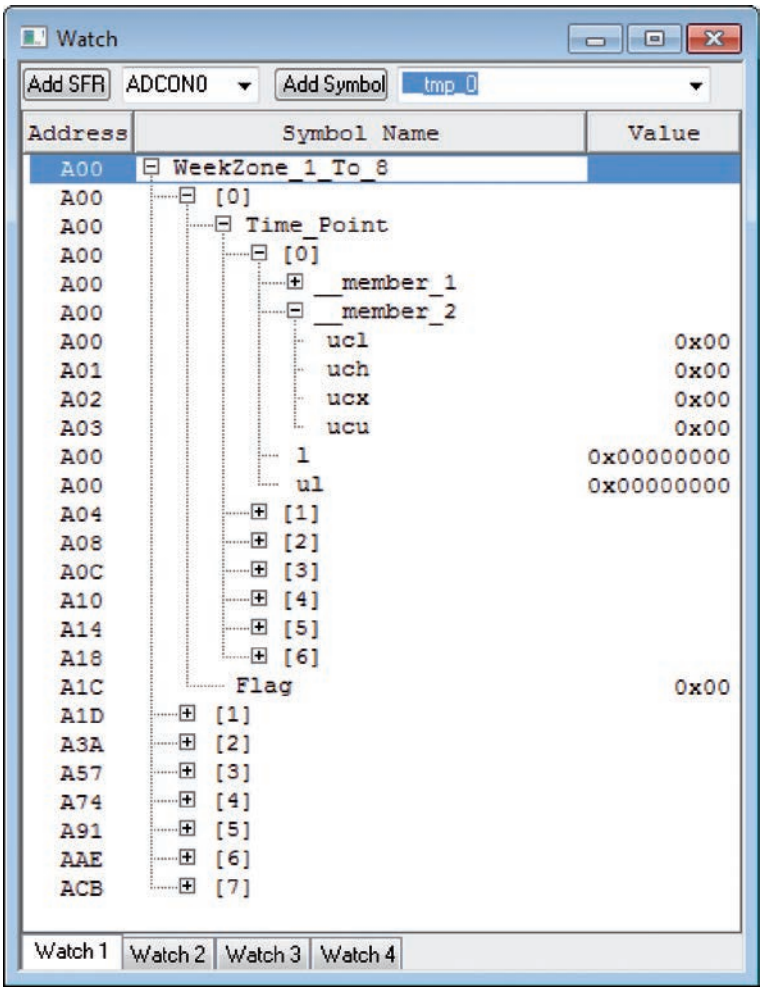


Figure 5 : structure de données imbriquées utilisée dans le projet pour la gestion d'intervalles de temps.

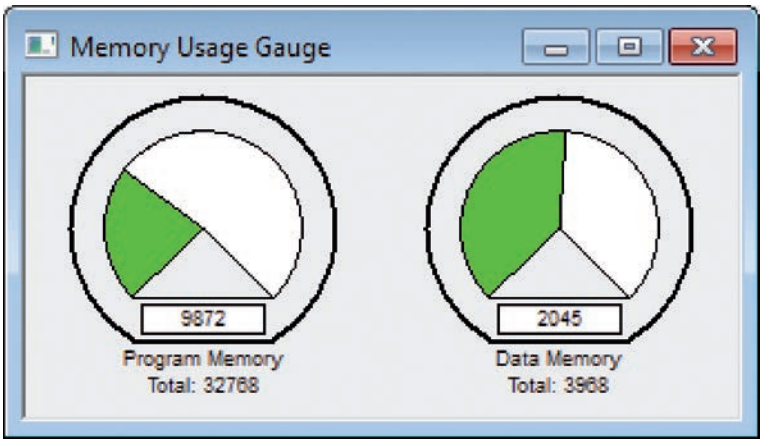


Figure 6 : la quantité de code utilisée pour l'exécution des fonctions du programme occupe environ 1/3 de la totalité de la mémoire FLASH, tandis que l'utilisation de la mémoire RAM s'élève à environ la moitié de celle disponible.

« **ram.h** » : contient la déclaration de toutes les variables présentes dans la mémoire RAM.

« **typedef_generic.h** » : contient la déclaration des structures de données génériques utilisées et les variables associées allouées dans la mémoire RAM.

« **typedef_i2c.h** » : contient la déclaration des structures de données utilisées pour la gestion du **bus I2C** et les variables associées allouées dans la mémoire RAM.

« **typedef_uart.h** » : contient la déclaration des structures de données utilisées pour la gestion du périphérique SCI et les variables associées allouées dans la mémoire RAM.

Compte tenu de l'hypothèse initiale, passons à la décrispation du « **noyau** » qui se trouve dans le fichier « **main.c** » du firmware.

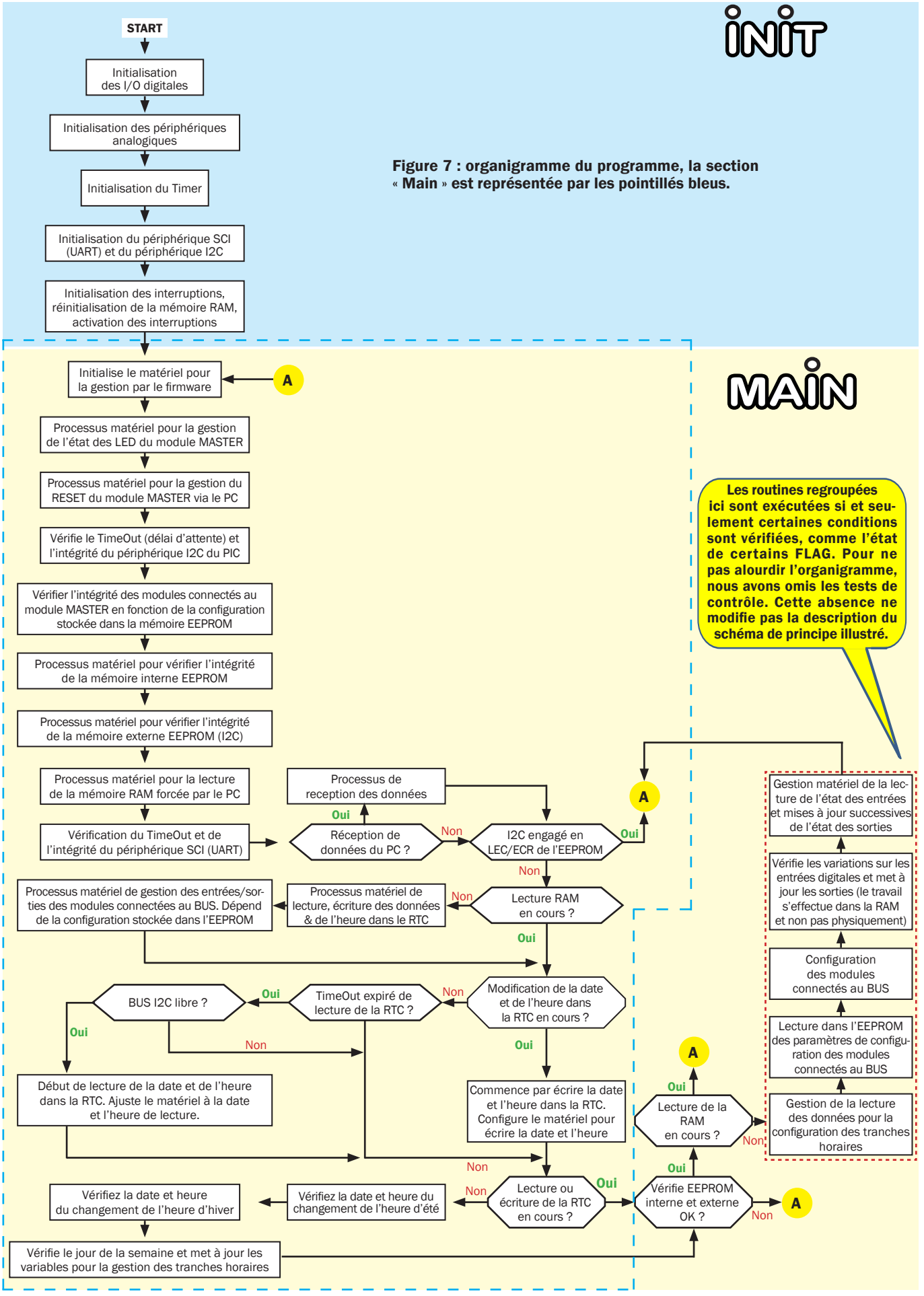


Figure 7 : organigramme du programme, la section « Main » est représentée par les pointillés bleus.

Pour vous aider dans la compréhension de la description, référez-vous à l'organigramme de la figure 7.

L'organigramme est divisé en deux parties distinctes : la première, appelée « **Init** », est utilisée pour initialiser le système c'est-à-dire tous les périphériques du **PIC18F2620** nécessaires au fonctionnement du dispositif. La seconde, appelée « **Main** », est la section dans laquelle sont traités tous les périphériques précédemment configurés, et appelle toutes les fonctions nécessaires au bon fonctionnement du système.

Évidemment, il existe un certain nombre de « **feux de signalisation** » ou **lignes de codes** dans le programme, qui représentent de simples indicateurs et qui doivent être activés ou désactivés en fonction des circonstances. Cela permet d'appeler les procédures et de déterminer l'ordre de traitement des périphériques.

Par exemple, si l'**EEPROM** externe n'est pas programmée, cela signifie que vous n'avez pas entré de programme utilisateur valide et donc certaines fonctions ne peuvent pas être exécutées parce que les données figurant dans l'**EEPROM** ne sont pas cohérentes.

En outre, le test d'intégrité de la mémoire **EEPROM**, qu'elle soit interne ou externe, est nécessaire afin de déterminer s'il est possible de traiter les données qu'elle contient, puis de configurer le module **SLAVE** (esclave) connecté au bus, ensuite de gérer l'état des sorties en fonction de l'état des entrées et enfin de gérer tous les créneaux horaires mis à disposition par le système (la configuration stockée dans la mémoire **EEPROM** concerne également le module **MASTER**).

Les informations stockées dans la mémoire **EEPROM** sont programmées à l'aide du logiciel sur l'ordinateur qui, en exploitant un protocole de communication particulier, envoie les données au microcontrôleur qui, à son tour, programme à la fois la mémoire **EEPROM** interne et l'**EEPROM** externe via le **bus I2C**. En outre, la connaissance des adresses mémoires présente un intérêt, vous pouvez faire un diagnostic en lisant l'état des variables dans la mémoire **RAM**.

Toujours à l'aide du protocole de communication, nous pouvons lire l'horloge temps réel (RTC : Real Time Clock) présente sur le module **MASTER** et, si nécessaire, régler la date et l'heure avec celle du PC ou à une autre valeur désirée.

Le firmware fonctionne dans un mode totalement automatique et tient compte de l'heure d'hiver et de l'heure d'été, il est donc en mesure d'identifier le dernier dimanche du mois de Mars et d'avancer d'une heure l'heure actuelle (de 2h00 à 3h00 passage à l'heure d'été) ou le dernier dimanche du mois d'Octobre et ensuite de reculer d'une heure (de 3h00 à 2h00 passage à l'heure d'hiver).

En regardant l'organigramme de la figure 7, vous pouvez voir la section « **Main** » représentée par des pointillés bleus. Vous remarquerez qu'il existe un certain nombre de fonctions (et de processus connexes) qui sont traitées dans chaque cycle de la boucle principale « **Main** » (notez que la boucle principale « **Main** » est répétée à l'infini par le microcontrôleur).

Ces fonctions servent à initialiser/réinitialiser un certain nombre de « flags » (drapeaux) qui permettent ou non l'exécution des fonctions décrites dans la zone en pointillé rouge (à droite vers le bas de la figure 7). Toutes ces fonctions qui sont regroupées dans la partie rouge sont liées à la configuration des modules **SLAVE** et **MASTER** et aux fonctions de lecture de l'état des entrées et de gestion de l'état des sorties. En outre, ces fonctions sont traitées par les intervalles de temps.

Le plus important est la partie du firmware qui gère la communication du **bus I2C**. La prise en charge est réalisée par un dispositif d'interruption généré par un périphérique matériel du **PIC18F2620**. La gestion par l'intermédiaire d'une interruption vous permet d'écrire un code plus « propre » et compréhensible. Le firmware doit gérer les composants suivants connectés au **bus I2C** :

- le circuit intégré **DS1307** qui est l'horloge temps réel (**RTC** : **Real Time Clock**) ;

- la mémoire **EEPROM** 24FC1025 (1Mbit) ;

- le circuit **MCP23017** qui dispose de deux blocs d'extension de 8 entrées/sorties chacun ;

- le circuit **MCP23008** qui dispose d'un seul bloc d'extension 8 entrées/sorties.

Le périphérique **I2C** présent dans le microcontrôleur **PIC18F2620** est conçu pour travailler en « **MODE MASTER** » (en mode maître). **Attention à ne pas confondre avec le module MASTER, cela fait référence à un mode de fonctionnement d'un périphérique du bus I2C qui peut travailler soit en « mode maître » ou en « mode esclave ».**

Pour communiquer avec le circuit intégré désiré, le firmware doit d'abord vérifier que le **bus I2C** est libre et que par conséquent aucune communication n'est en cours, après quoi il peut engager le bus dans une opération de lecture ou d'écriture vers le circuit.

Quand une communication est en cours, vous pouvez identifier le périphérique qui communique, simplement en vérifiant l'état de certains « flags » (un « flag » par périphérique).

En outre, pour chaque périphérique est alloué un « **buffer** » (une mémoire tampon) pour traiter les données à envoyer ou recevoir, chaque « **buffer** » a une taille différente en fonction des données à gérer pour chaque périphérique qui sont différentes à la fois en quantité et en type. Également, la vitesse de communication entre les divers circuits intégrés est différente, par exemple en ce qui concerne la ligne **SCL** (horloge) de la mémoire **24FC1025**, elle est conçue pour fonctionner à **400 kHz**, tandis que pour le circuit d'horloge **DS1307** la vitesse de fonctionnement de la ligne **SCL** est de seulement **100 kHz** et ne peut pas accepter des vitesses plus élevées.

Sinon, le circuit **MCP23017** est conçu pour travailler à 400 kHz et est monté directement sur le module **MASTER**, tandis que tous les circuits **MCP23008** fonctionnent à 100 KHz et sont montés sur les modules **SLAVE**, qui peuvent être placés à quelques mètres du module **MASTER**. En conséquence, le registre SSPADD qui règle la vitesse (fréquence d'horloge) de la ligne **SCL** sera paramétré en fonction du périphérique utilisé.

Chaque communication vers l'un des périphériques commence toujours par la génération du bit de démarrage par le microcontrôleur **PIC18F2620** (« **Start Bit** »), ce qui engage le **bus I2C** dans un processus de communication qui sera interrompu par la génération d'un bit d'arrêt (« **Stop Bit** »). Entre ces deux « repères », des données subdivisées en octets sont insérées, et envoyées aux différents périphériques (par exemple le DS13017 ou un **MCP23008**) et pour chaque octet envoyé, le **PIC18F2620** attend un signal d'acquittement (ACK) du périphérique concerné.

A chaque génération d'un bit de démarrage ou d'un bit d'arrêt et après l'envoi d'un octet de données, une interruption est générée pour indiquer que le périphérique est libre et que l'on peut interroger un autre périphérique.

Ainsi, dans la routine de gestion de l'interruption, il existe une structure qui détermine dans quel état se trouve la gestion de la communication et qui peut agir en conséquence. Nous devons également distinguer vers quel périphérique envoyer ou recevoir les données lors du changement de la gestion de la communication, le cas échéant.

Par exemple, la figure 8 d'après le datasheet du composant, met en évidence une communication possible du périphérique **MCP23017**. Notez que la communication commence toujours par un bit de démarrage (« **Start Bit** ») et se termine toujours par un bit d'arrêt (« **Stop Bit** »). Le premier octet qui doit être obligatoirement envoyé après un bit de démarrage (« **Start Bit** ») est le « **Control Byte** » ou **octet de contrôle** (voir la figure 9) et doit contenir un code d'identification unique qui correspond à l'adressage du périphérique I2C.

Le « **Control Byte** » contient 8 bits dont les 4 premiers sont définis par le constructeur, dans ce cas nous avons « 0100 ». Les 3 bits suivant (A2, A1, A0) définissent l'adresse physique du périphérique pour communiquer et peuvent être modifiés par l'utilisateur (dans notre cas il y a un seul **MCP23017** et donc nous sélectionnons « 000 ») et, enfin le dernier bit R/W permet d'effectuer une écriture ou une lecture sur le périphérique I2C concerné.

La figure 8 montre à la fois la lecture/écriture d'un octet individuel et la lecture/écriture séquentielle de plusieurs octets, vous devez indiquer à quelle adresse vous souhaitez commencer à écrire ou à lire.

Ce concept s'applique à tous les périphériques I2C. Donc, en résumé, le firmware pour gérer une communication I2C avec l'un des périphériques possibles, doit :

- préparer les données à envoyer au périphérique souhaité. Pour le faire, il doit utiliser un « **buffer** » (mémoire tampon) approprié dans la mémoire;

- s'assurer que le **bus I2C** est libre, c'est à dire qu'il n'y a pas de communication en cours avec d'autres périphériques;

- générer le bit de démarrage (« **Start Bit** ») et attendre l'interruption avant de passer à l'étape suivante;

- envoyer le « **Control Byte** » ou octet de contrôle et attendre l'interruption avant de passer à l'étape suivante. Chaque périphérique dispose de son propre octet de contrôle (voir la figure 9).

Le périphérique est interrogé, pour chaque octet reçu, et doit générer un signal d'acquittement ACK. Si le **PIC18F2620** ne reçoit pas le signal d'acquittement ACK du périphérique, il interrompra la communication par la génération d'un bit d'arrêt (« **Stop Bit** »);

- envoyer l'adresse à partir de laquelle commence la lecture ou l'écriture et attendre l'interruption avant de passer à l'étape suivante. L'adresse peut nécessiter un ou deux octets selon le cas.

Par exemple en ce qui concerne la mémoire **EEPROM 24FC1025**, deux octets sont nécessaires pour l'adressage ;

- envoyer les données dans le cas de l'écriture, ou commencer le processus de lecture. Si l'écriture doit s'effectuer vers un périphérique nécessitant un octet à la fois, le firmware attend de vider le « **buffer** » (mémoire tampon) correspondant, et attend l'interruption pour chaque octet avant d'envoyer le suivant.

Comme précédemment pour chaque octet envoyé, il doit recevoir un signal d'acquittement ACK du périphérique. Par contre dans le cas de la lecture il doit :

- générer un bit de « **Re-Start** » et attendre l'interruption avant de passer à l'étape suivante ;

- renvoyer le « **Control Byte** » ou octet de contrôle en spécifiant qu'il effectue une opération de lecture (voir la figure 9) ;

- lire les données, un octet à la fois sur le périphérique en cours. Dans ce cas, c'est le **PIC18F2620** qui génère le signal d'acquittement ACK pour indiquer que l'octet a bien été reçu.

Avant de passer à l'octet suivant, il est obligatoire d'envoyer l'accusé de réception au périphérique interrogé ;

- terminer l'envoi ou la réception de données en générant le bit d'arrêt (« **Stop Bit** »).

Ce processus est réitéré indéfiniment à chacun des périphériques disponibles (**EEPROM**, DS13017, **MCP23008**, **MCP23017**, ...) et sera traité par le système, permettant ainsi une gestion correcte des entrées et des sorties. En outre, la mémoire **EEPROM** sera contrôlée pour s'assurer qu'elle est intègre et qu'elle contient des données cohérentes avec l'application envisagée. L'horloge et le calendrier sont lus chaque seconde pour être toujours mis à jour avec la date et l'heure afin d'être en mesure de gérer les intervalles de temps appropriés pouvant être implémentés lors de l'application.

Dans le prochain et dernier article du **MINIBUS**, nous décrirons le logiciel qui permet la programmation des fonctions mises à disposition par le système, la programmation des tranches horaires, la configuration des entrées/sorties, les types de modules **SLAVE** reliés au **bus I2C** et l'association des entrées/sorties pour l'exécution des commandes **SET**, **RESET** ou **TOOGLE** de la sortie désirée.

Les typons des circuits imprimés et le **programme .hex** du **MINI BUS** sont téléchargeables sur notre site **www.electroniquemagazine.com** dans le sommaire détaillé de la revue numéro 126 section « télécharger ».

Systèmes d'éclairage à LED Solutions

pour l'économie d'énergie, appropriées à tout environnement.

Panneau lumineux 112 LED

€ 95,00
réf. P3030-W

- Flux lumineux de 880 lm
- Consommation 12 W
- Dimensions 29,5 x 29,5 cm

Disponible en version 416 LED,
3000 lm, 45 W, 59,5 x 59,5 cm
réf. P6060-W € 218,00

Panneau lumineux 36 LED encastrable

€ 64,50
réf. P1717-WW

- Flux lumineux de 880 lm
- Consommation 16 W
- Dimensions 17 x 17 cm

Tube lumineux 56 LED

€ 43,00
réf. T8-HWSMD

- Flux lumineux de 840 lm
- Consommation 10 W
- Dimensions 600 mm x 26 mm (diam)

Spot 12 LED encastrable

€ 86,00
réf. D1202-WW

- Flux lumineux 1410 lm
- Consommation 28 W
- Dimensions 16 cm

Lampe à LED - E27

€ 35,00
réf. G60ME27-9W-WW

- Flux lumineux 900 lm
- Consommation 9 W

Lampe 72 LED - E27

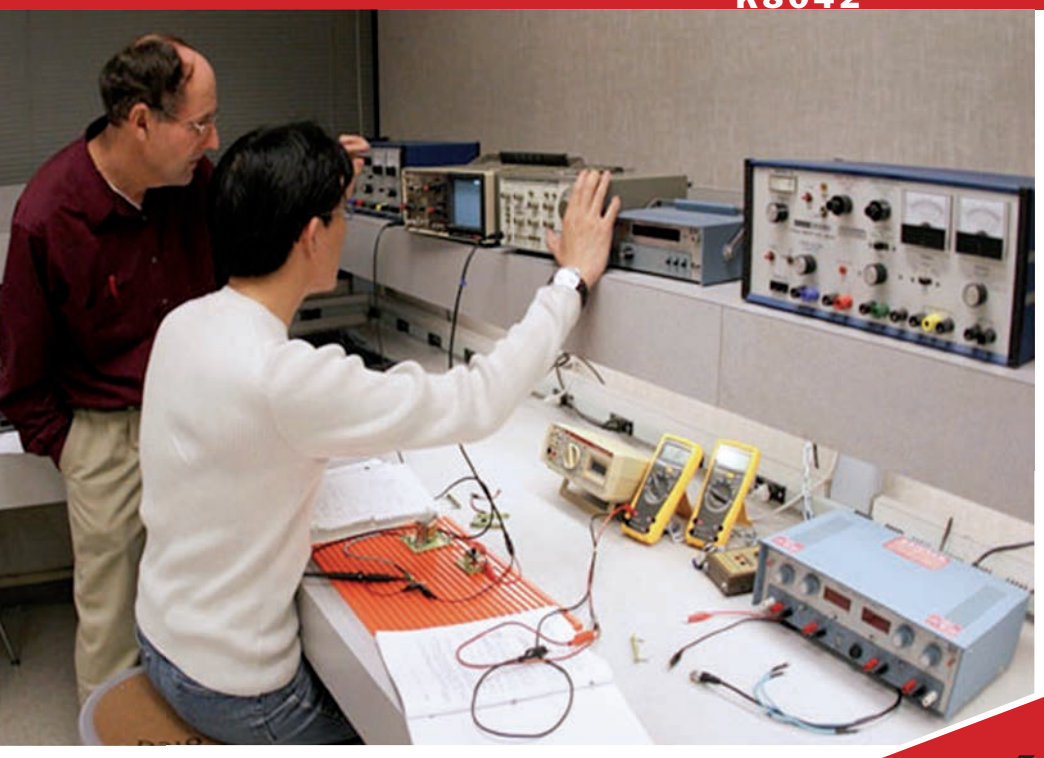
€ 24,20
réf. CORN72SMD-WW

- Flux lumineux 900 lm
- Consommation 9,5 W

Comelec 03 / 2014

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

K8042



Cette alimentation, grâce à un transformateur à point milieu, peut délivrer une tension continue symétrique par rapport à la masse, ou asymétrique, réglable entre ± 1,25 V et ± 18 V avec un courant maximal de ± 1 A. Elle est idéale pour alimenter des amplificateurs audio de moyenne puissance, ou des montages à amplificateurs opérationnels nécessitant une tension symétrique.

ALIMENTATION SYMÉTRIQUE DE ± 1,25 V À ± 18 V / ± 1 A

DAVIDE SCULLINO

Les circuits d'amplification d'un signal audio en général, et ceux des instruments de mesure et des capteurs qui renvoient des signaux analogiques, fonctionnent normalement avec une tension symétrique par rapport à la masse, qui se compose de deux tensions de valeur absolue égale mais de signe opposé entre elles.

En effet les étages amplificateurs et les étages tampons doivent être en mesure de fournir deux tensions positives et négatives par rapport à la masse. C'est le cas, par exemple, pour le traitement d'une composante BF sinusoïdale, oscillant entre des valeurs positives et négatives, et qui nécessite un amplificateur qui soit en mesure de fournir en sortie une tension sinusoïdale avec une distorsion acceptable.

Dans le cas des amplificateurs de puissance audio, le problème devient encore plus important parce que l'étage de sortie doit être capable de fournir un courant sinusoïdal de grande amplitude, afin d'actionner les membranes des haut-parleurs d'avant en arrière. Dans de nombreux cas, l'alimentation symétrique est simulée en plaçant un condensateur en série avec la sortie qui, en se déchargeant, fournit le courant nécessaire à la branche négative de l'amplificateur.

Cette solution est acceptable pour les petits amplificateurs mais pas pour ceux de forte puissance, même si cette technique a beaucoup été utilisée ces dernières décennies. Pour réaliser une alimentation avec deux tensions symétriques, il existe diverses solutions. Ici nous vous proposons une solution basée sur une configuration linéaire et capable de fournir des tensions stabilisées.

La double alimentation linéaire est basée sur un transformateur qui possède un point central sur l'enroulement secondaire (capable de fournir à ses extrémités à chaque instant deux tensions d'amplitude égale mais en opposition de phase par rapport au point central) qui alimente un pont redresseur double alternance et, éventuellement, un régulateur de tension.

L'alimentation décrite ici peut être utilisée à la fois pour alimenter un amplificateur, une table de mixage, un égaliseur, ou aussi un instrument de mesure, un sampler, un convertisseur numérique/analogique ou une interface avec des capteurs d'un appareil médical, mais rien n'interdit de l'utiliser pour construire une alimentation symétrique réglable de laboratoire de petite puissance.

Caractéristiques techniques :

Tension de sortie : **± 1,25 V** à **± 18 V**
Courant nominal de sortie : **± 1 A**
Courant maximal de sortie : **± 1,5 A**
Tension maximale d'entrée : **2 x 24 V**
Régulation de la tension
Protection contre les courts-circuits

Le schéma électrique

Notre alimentation double de puissance est stabilisée dans le sens où en plus de fournir une tension symétrique, elle permet de stabiliser les variations de charge, de fournir un courant maximal de **1 A pour chaque branche** (1 A pour le positif et 1 A pour le négatif), et également d'ajuster les valeurs des tensions de sortie entre un minimum de **1,25 V** et un maximum de **18 V** , soit de **+ 1,25 V** à **+ 18 V** pour la branche positive et de **- 1,25 V** à - **18 V** pour la branche négative. Le circuit visible en **figure 1** est basé sur un pont de diodes en technologie discrète et de deux régulateurs de tension ajustable, c'est à dire que les tensions de sorties sont réglables indépendamment : celle du **LM317** et celle du **LM337**.

Ce sont des régulateurs complémentaires, car ils remplissent la même fonction seulement, le premier est prévu pour réguler des tensions positives et le second des tensions négatives.Ces régulateurs offrent une parfaite stabilisation de la tension de sortie dans la limite du courant disponible, et disposent d'une protection contre les courts-circuits et les surcharges, ce qui signifie que dans toutes les conditions ils ne délivreront pas plus de 1,5 A, même si l'utilisateur leur demande plus.

Voyons le fonctionnement de l'alimentation : le transformateur qui abaisse la tension secteur aux environs de **24 VAC** est connecté aux points **VA-0-VB** c'est à dire l'entrée **AC IN**, avec le **point milieu** de l'enroulement secondaire du transformateur relié au **point 0** (masse) du montage, les autres extrémités de l'enroulement sont reliées aux points **VA** et **VB**, la tension présente sur les demi-enroulements secondaires **VA-0** et **0-VB** est de valeur égale mais de sens opposé (par rapport au point milieu 0). Le **pont redresseur** de **Graetz** constitué par les diodes **D1**, **D2**, **D3** et **D4**, redresse les tensions sinusoïdales fournies par le transformateur aux points **VA-0** et **0-VB** en une **tension unidirectionnelle positive** sur les **cathodes** des diodes **D3** et **D4** par rapport à la masse (point 0) et une **tension unidirectionnelle négative** sur les **anodes** des diodes **D1** et **D2**. Les condensateurs **C1** et **C3** filtrent la tension positive présente sur les cathodes de D3 et D4 pour la transformer en une **composante continue** bien **lissée**, les condensateurs C2 et C4 réalisent la même opération pour la branche négative (anodes des diodes D1 et D2).

Par conséquent, nous nous retrouvons avec deux tensions théoriquement identiques, mais de **sens opposé**, entre la borne **IN** du régulateur **VR1** et la **masse** et entre le point **IN** du régulateur **VR2** et la **masse**.

Chacune des tensions, en l'absence de charge, vaut **1,414** fois la **valeur efficace** de la tension alternative fournie par les enroulements secondaires correspondants, le tout diminué de la chute de tension dans les deux diodes du pont. On appelle **Vrms** la tension efficace de chaque demi enroulement secondaire et **Vd** la chute directe dans les diodes (en général 0,6 V par diode), nous voyons que la valeur absolue (en ignorant le signe) de la tension continue **Vcc** présente sur les condensateurs de filtrage vaut :

Vcc = Veff x 1,414 - 2 x 0,6 V

Tableau 1

Tension d'entrée alternative au point AC IN	Tension de sortie stabilisée
de ± 1,25 V à 5 V	2 x 9 VAC
de ± 5 V à 9 V	2 x 12 VAC
de ± 9 V à 12 V	2 x 15 VAC
de ± 12 V à 15 V	2 x 18 VAC
de ± 15 V à 18 V	2 x 22 VAC

Valeur de la tension de chaque enroulement secondaire du transformateur en fonction de la tension de sortie avec un courant maximal de 1 A.

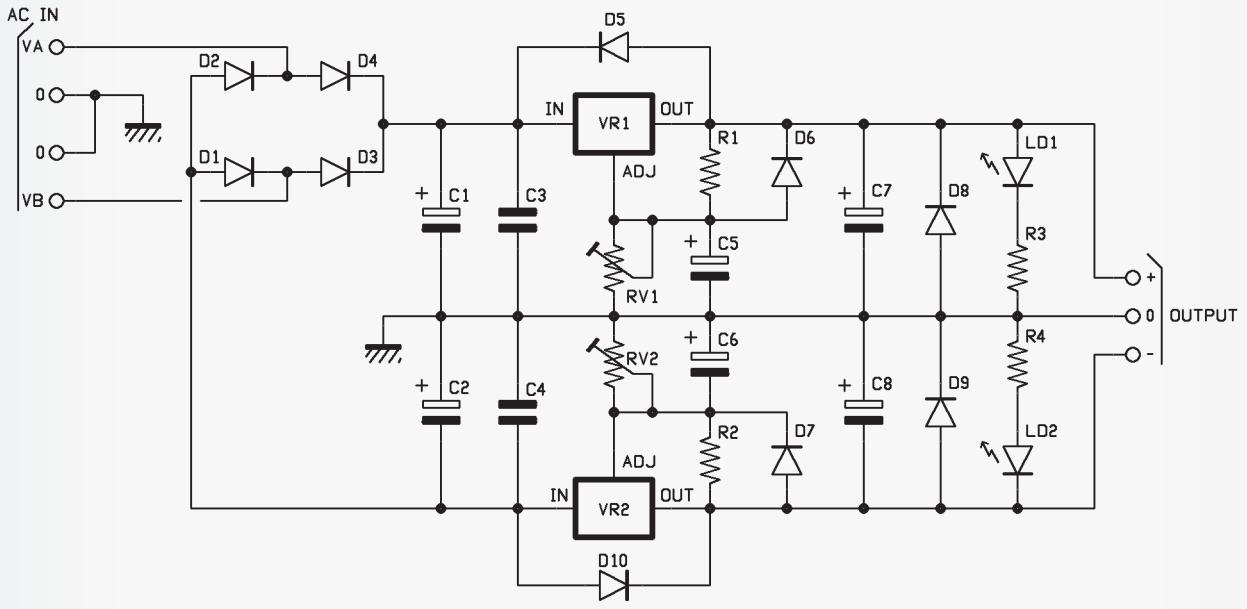


Figure 1 : schéma électrique de notre alimentation symétrique de ± 1,25 V à ± 18 V.

Le nouveau label énergétique

De nouvelles étiquettes énergétiques sont apparues pour les appareils électroménagers, indiquant la classe énergétique conformément à la législation Européenne. Elles introduisent de nouvelles classes supplémentaires énergétique qui sont, **A +**, **A ++** et **A +++**,plus représentatives de la plus grande efficacité énergétique des appareils modernes. Dans tous les pays de l'Union Européenne, l'étiquette se présentera sous forme de flèches uniformes et de couleurs différentes pour distinguer les différentes classes d'efficacité énergétique. Si un produit a la flèche verte foncée, cela signifie que c'est un produit avec une efficacité énergétique élevée, tandis qu'un appareil avec une flèche rouge indique que le produit est de faible efficacité.

Les étiquettes précisent également la **consommation annuelle d'énergie en kWh**. Pour aider les consommateurs à la compréhension de ces nouvelles étiquettes, un site web a été mis en place à l'adresse suivante : www.guidetopten.fr, qui propose un guide comparatif des 10 meilleurs produits ayant les mêmes performances et les mêmes fonctionnalités, mais qui consomment le moins d'électricité.

Supposons que nous alimentons le circuit avec un transformateur ayant une tension de 2 x 12 V efficaces aux secondaires, nous obtenons alors sur chaque entrée des régulateurs, par rapport à la masse la valeur suivante :

Vcc = 12 x 1,414 - 1,2V
= (16,968 - 1,2)V = 15,768V

Il s'agit de la valeur de la tension positive et négative appliquée à l'entrée des régulateurs **LM317** et **LM337**.

Voyons maintenant ce que font les deux régulateurs, à commencer par le positif, soit **VR1**, utilisé pour fournir une tension stabilisée **Vout** en fonction d'une tension **non stabilisée Vin** ; nous obtenons la relation suivante :

Vout = Vref(1 + RV1/R1)+ Iadj x RV1

où **Vref** est la tension de référence présente entre la broche **ADJ** et la **masse**, et s'élève à **1,25 V**. La formule ci-dessus peut être simplifiée de la manière suivante :

Vout = 1,25V(1+RV1/R1)

On néglige le terme **Iadj x RV1**, parce que le courant présent sur la broche **ADJ** du régulateur est **négligeable** par

rapport au **courant de sortie** (de l'ordre de quelques dizaines de microampères). Dans notre cas, grâce au trimmer **RV1**, nous pouvons faire varier la tension de sortie **Vout** du régulateur (par rapport à la masse), de **1,25 V** (correspondant à la position du trimmer en « court-circuit », c'est à dire une résistance nulle) et **18 V** (correspondant à la position du trimmer présentant une résistance maximale entre ses bornes).

La même chose est vraie pour la partie négative, pilotée par le **LM337**, mais les tensions sont de valeurs opposées, nous appliquons la formule suivante simplifiée :

Vout = 1,25V(1+RV2/R2)

Dans ce cas, lorsque le potentiomètre **RV2** est **court-circuité** (résistance nulle), la tension de sortie est égale à **Vref** soit **- 1,25 V**, alors que si le trimmer **RV2** présente une **résistance maximale**, la tension de sortie du régulateur est de **- 18 V**.

Naturellement, cela implique que la tension appliquée à l'entrée des régulateurs soit suffisante, c'est-à-dire égale à celle de la sortie **Vout** augmentée de la chute de tension entre l'entrée et la sortie des régulateur VR1 et VR2. Il faut considérer qu'à pleine charge, la tension filtrée par le condensateur électrolytique est réduite d'environ **20% par rapport à la valeur à vide** (sans charge), cela signifie que la tension d'entrée des régulateurs est considérée en charge. Tout cela, pour s'assurer du calcul de la valeur des diviseurs de tension fixée par les trimmers, sinon les régulateurs ne pourraient pas fournir les tensions stabilisées souhaitées.

Pour nous faire une idée du problème, les régulateurs **LM317** et **LM337** pour fonctionner correctement doivent avoir une **chute de tension** (drop-out) entre l'entrée et la sortie de l'ordre de **3 V**, si nous voulons **1,25 V en sortie**, il est nécessaire d'avoir une **valeur minimale de Vin** de 1,25 V + 3 V = **4,25 V**. Sachant que la tension chute de **20% en charge**, nous devons être un peu plus élevés et appliquer à **Vin** au moins **5,2 V**.

Pour plus de commodité, reportez-vous aux tensions indiquées dans le **Tableau 1** pour l'enroulement secondaire du transformateur afin d'obtenir les valeurs les plus courantes de la tension de sortie stabilisée du circuit.

Gardez à l'esprit que la tension d'entrée de chacun des régulateurs ne doit pas dépasser **40 V**, les condensateurs **C1** et **C2** ne supportent qu'une tension de fonctionnement de **35 V**, et c'est pour cela que vous ne devez pas appliquer aux points **AC IN**, une tension de plus de **2 x 24 VAC efficaces**.

Il est important de respecter les tensions, car un transformateur qui délivre une tension trop faible ne garantira pas une tension stabilisée en sortie, de même qu'un transformateur fournissant une tension trop élevée, avec le même courant de sortie, provoquera une dissipation trop importante des régulateurs. En effet, la puissance dissipée (Pd) par les LM317 et LM337 est égale au produit du courant fourni à la charge (courant en sortie) et de la chute de tension entre les broches IN et OUT des régulateurs.

Par exemple, pour un courant de 1 A en sortie, si la tension aux bornes du condensateur de filtrage est de 20 V (transformateur de 2 x 15 VAC) et que le trimmer est réglé pour obtenir une tension de sortie de 10 V, le régulateur aura à ses bornes une chute de tension de 10 V. Ainsi, la perte de puissance dans chaque régulateur VR1 et VR2 est égale à :

Pd = 1 A x 10 V = 10 W

Cela étant dit, nous complétons la description du circuit en analysant les composants qui restent, nous commençons par les diodes **D5** et **D10** qui empêchent les condensateurs de sortie de se décharger dans les régulateurs de tension, en cas de court-circuit en entrée ou d'arrêt du circuit. Le but de ces diodes est d'empêcher que la tension présente aux bornes de **C1** et **C3** pour la branche positive et de **C2** et **C4** pour la branche négative, ne devienne **inférieure respectivement à la tension présente aux bornes** de **C7** et **C8**, et de maintenir la tension d'entrée des régulateurs supérieure à celle de sortie. Cette situation peut se produire aussi si les sorties se retrouvent sans charge et risquent d'endommager les étages de sortie des régulateurs, du moins lorsque l'on travaille à des tensions relativement élevées.

Les diodes **D6** pour le positif et **D7** pour le négatif ont une fonction similaire, elles **protègent les broches ADJ** des régulateurs d'une décharge des condensateurs électrolytiques **C5** et **C6** en faisant circuler directement le courant vers les condensateurs de sortie et contournant ainsi le circuit **ADJ OUT**.

Notez que les condensateurs **C5** et **C6** filtrent le **bruit** et **l'ondulation résiduelle** (que les condensateurs de filtrage ne parviennent pas à supprimer, et qui pourrait affecter la sortie), de la tension de référence des broches ADJ de chacun des régulateurs. On augmente ainsi la rejection de l'ondulation de l'alimentation. Les diodes **D8** et **D9** protègent des courts-circuits pouvant se produire sur les sorties de l''alimentation, ou en cas de charges inductives. Enfin, les LED **LD1** et **LD2** indiquent la présence des tensions positive et négative, respectivement en sortie d'alimentation.

Réalisation pratique

Passons maintenant à la construction du circuit qui est très simple, car elle ne nécessite que l'utilisation de quelques composants traditionnels à assembler.

La première des choses à faire est de préparer le circuit imprimé (visible à la **figure 3**), après l'avoir téléchargé sur notre site **www.electroniquemagazine.com**, dans la catégorie « revue papier », revue numéro 124 ou de se le procurer tout fait auprès des annonceurs.

Une fois le circuit gravé et percé, reportez-vous au schéma d'implantation de la **figure 4** et aidez-vous à l'aide de la photo du prototype visible en **figure 2**. Commencez par souder les résistances, les diodes (attention à la polarité, respectez bien la position de la bague pour cela reportez-vous au plan du montage).

Soudez ensuite les deux trimmers, puis placez les condensateurs en respectant la polarité pour les électrolytiques, finissez par les deux LED, les méplats doivent être orientés comme indiqué sur le plan de montage, les cathodes (méplats) sont vers les résistances R1 et R3. Maintenant, il suffit de monter debout les régulateurs LM317 et LM337 en plaçant le

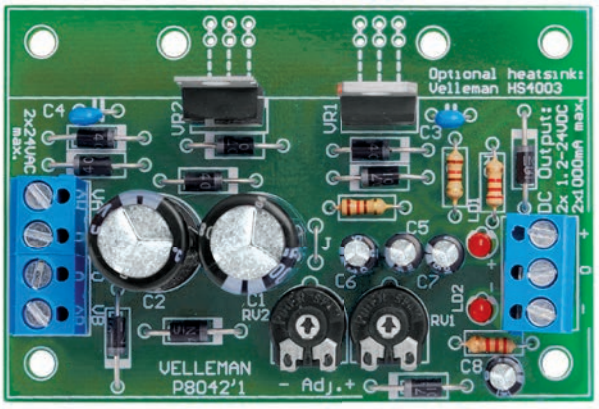


Figure 2 : photo de l'un de nos prototypes de l'alimentation symétrique.

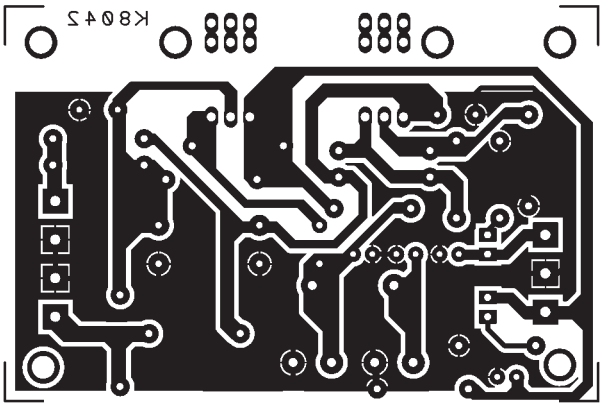


Figure 3 : circuit imprimé côté soudures à l'échelle 1 : 1 de l'alimentation symétrique.

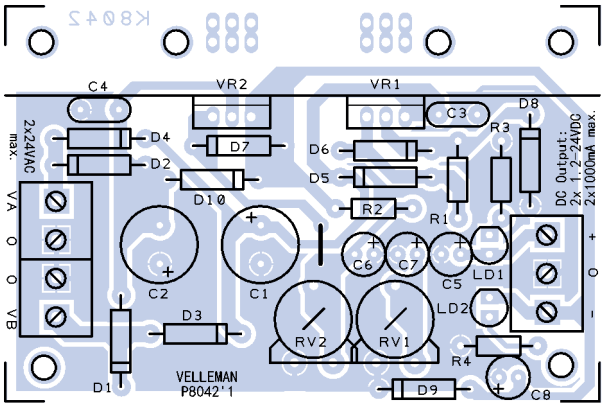


Figure 4 : schéma d'implantation des composants de l'alimentation symétrique.

Liste des composants du K8042

R1.. 120
R2.. 120
R3.. 2,2 k
R4.. 2,2 k
RV1.. trimmer 2,2 k
RV2.. trimmer 2,2 k
C1.. 1000 µF/35 V électrolytique
C2.. 1000 µF/35 V électrolytique
C3.. 100 nF multicouche
C4.. 100 nF multicouche
C5.. 10 µF/63 V électrolytique
C6.. 10 µF/63 V électrolytique
C7.. 10 µF/63 V électrolytique
C8.. 10 µF/63 V électrolytique

D1.. 1N4007
D2.. 1N4007
D3.. 1N4007
D4.. 1N4007
D5.. 1N4007
D6.. 1N4007
D7.. 1N4007
D8.. 1N4007
D9.. 1N4007
D10.. 1N4007
VR1.. LM317
VR2.. LM337
LD1 .. LED 3 mm rouge
LD2 .. LED 3 mm rouge
bornier 2 pôles x 2 pour AC IN
bornier 3 pôles x 1 pour Vout

boîtier à une distance de 7 à 8 mm de la surface du circuit imprimé pour permettre aux régulateurs de dissiper la chaleur pendant le fonctionnement.

Chaque régulateur doit être monté sur un radiateur ayant une résistance thermique de 8 °C/W, avec un boulon de type 3 MA et en insérant de la graisse au silicone pour faciliter le transfert de la chaleur de la semelle métallique du régulateur au radiateur. Vous pouvez aussi utiliser un seul dissipateur pour les régulateurs VR1 et VR2, à condition qu'il n'est pas une résistance thermique supérieure à 4 °C/W.

Dans ce cas vous devrez isoler électriquement les régulateurs du dissipateur, en plaçant (entre les semelles en métal des boîtiers de chaque régulateur et le dissipateur) un isolant de mica pour boîtier TO220 recouvert de graisse au silicone sur les deux faces. Sinon, vous pouvez utiliser des isolants gris en Teflon, qui transfèrent la chaleur sans utilisation de la graisse au silicone. Les dissipateurs de chaleur peuvent être fixés au circuit imprimé en utilisant les trous prévus à cet effet, et en mettant une pointe d'étain sur les tiges traversant les trous de fixation.

Notez que les LM317 et LM337 peuvent fournir jusqu'à un maximum de 1,5 A, cependant pour que cela soit possible, vous devez installer des dissipateurs surdimensionnés ayant une résistance thermique de 5 °C/W pour chaque régulateur ou un seul dissipateur de 2,5 °C/W pour les deux régulateurs VR1 et VR2.

Une fois que vous avez terminé l'installation, vous pouvez souder les borniers correspondant à l'entrée AC IN et celui correspondant à la sortie Vout (voir la figure 5).

Afin de faciliter l'installation, le circuit peut être logé à l'intérieur du boîtier de l'appareil qui doit être alimenté ou dans son propre boîtier, comme dans le cas où vous voudriez réaliser une alimentation de laboratoire.Si vous voulez, vous pouvez remplacer les deux trimmers RV1 et RV2 par un potentiomètre linéaire double en soudant les points milieux à gauche des extrémités et les deux contacts de chaque potentiomètre entre la masse du circuit et les points ADJ de VR1 et VR2 (voir la **figure 6**). Notez qu'il y a deux trimmers distincts, et que chaque régulateur peut être ajusté pour fournir une tension différente de l'autre, ce qui peut être utile dans le cas d'applications où les valeurs des deux tensions positives et négatives ne doivent pas être symétriques.

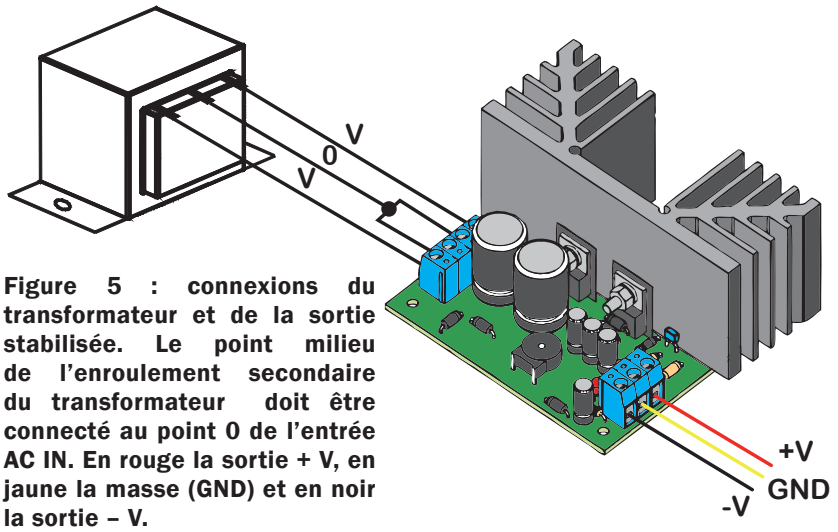


Figure 5 : connexions du transformateur et de la sortie stabilisée. Le point milieu de l'enroulement secondaire du transformateur doit être connecté au point 0 de l'entrée AC IN. En rouge la sortie + V, en jaune la masse (GND) et en noir la sortie – V.

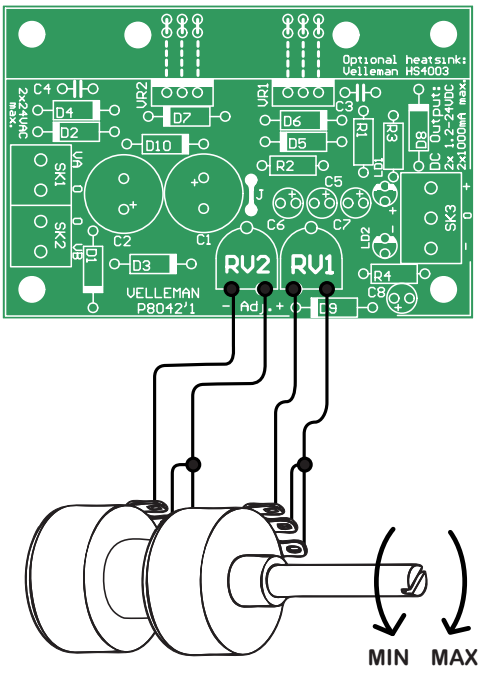
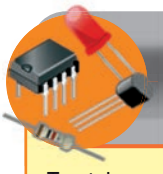


Figure 6 : Pour modifier simultanément (de manière identique) les deux tensions positive et négative, remplacez les deux trimmers par un potentiomètre linéaire double de valeur égale, comme indiqué sur la figure.

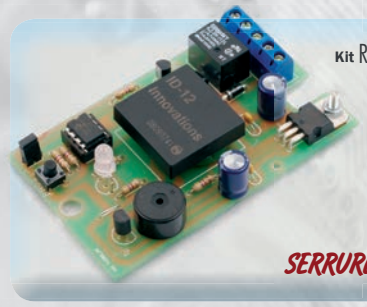


Comment construire ce montage

Tout le matériel nécessaire pour construire cette alimentation symétrique est disponible auprès de nos annonceurs. Voir les publicités dans la revue. Les typons des circuits imprimés sont téléchargeables gratuitement sur notre nouveau site Internet à la nouvelle adresse : **www.electroniquemagazine.com** dans le sommaire détaillé de la revue numéro 126 section « télécharger ».

CULTIVEZ VOTRE HOBBY AVEC NOS MONTAGES de Kits électroniques

Devenez un expert avec nos kits et modules.
Mettez en "pratique" votre connaissance en programmation!



Kit Réf. K6400
30,45 €

SERRURE CODÉE

Kit Réf. VM188
19,70 €

MINUTEUR IMPULSION-PAUSE 1s ~ 60h

Kit Réf. K8004
15,30 €

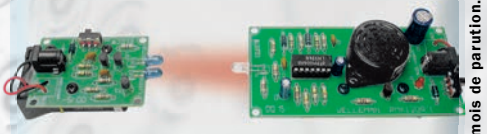
CONVERTISSEUR DE TENSION CONTINUE EN IMPULSIONS

Kit Réf. K4001
9,90 €

UN AMPLIFICATEUR MONO 7 W

Kit Réf. MK125
7,45 €

INTERRUPTEUR CRÉPUSCULAIRE



Kit Réf. MK120
11,00 €

BARRIÈRE INFRAROUGE TX & RX

Kit Réf. MK138
8,90 €

THERMOSTAT

Kit Réf. MK127
12,64 €

MICROBUG COUREUR

Kit Réf. MK182
15,95 €

CHAMBRE D'ÉCHO NUMÉRIQUE

Kit Réf. MK189
6,90 €

MONITEUR BATTERIE DE VOITURE

Réf. ET908M
89,00 €

LOCALISATEUR GSM (Vendu monté)

Réf. ET908M-GPS
114,60 €

LOCALISATEUR GSM / GPS (Vendu monté)

Photos non contractuelles, publicité valable pour les mois de parution. Prix exprimés en euros TTC. sauf erreurs typographiques ou omissions

Comelec 03 / 2014

Large gamme de coffrets d'encastrement et les achats en ligne sur www.comelec.fr

CD 908 13720 BELCODÈNE
Tél. 04 42 70 63 90 • Fax 04 42 70 63 95

COMELEC

K190

AMPLIFICATEUR STÉRÉO 2X5 WATTS

Dans cet article nous vous proposons de réaliser un mini amplificateur de puissance stéréo universel, adapté pour amplifier le signal de sortie d'appareils tels que les lecteurs MP3, les sorties audio des Smartphones ou encore les cartes son des PC.

de SANDRO REIS

L'amplificateur audio est un de ces dispositifs électroniques comme les temporisateurs (timers) ou les alimentations électriques, qui peuvent être considérés comme « essentiels » parce que même si les technologies changent avec l'avènement du numérique, nous en avons toujours besoin. Si vous voulez faire une démonstration avec un PC fixe ou portable en utilisant la carte son, ou avec **un lecteur MP3 portable dont la sortie BF n'est pas amplifiée** et qui peut tout juste fournir quelques centaines de millivolts à une faible impédance pour un casque, vous avez besoin d'un amplificateur audio.

Par exemple pour le **lecteur MP3 décrit dans le numéro 123 d'Electronique et Loisirs Magazine**, avoir un mini amplificateur de puissance qui rende audible un son quelconque sur une paire de petits haut-parleurs est très utile, car il vous permet de créer une base fixe où vous pouvez écouter votre propre musique quand vous arrivez à la maison.

Le projet décrit dans ces pages a été pensé dans le but de réaliser une « **box audio** » que vous pouvez connecter à une variété d'appareils à l'aide d'une paire de haut-parleurs, comme celles d'un ordinateur. Il vous suffit de le brancher sur les deux haut-parleurs et connecter l'entrée à une source BF à l'aide d'un câble blindé et une prise stéréo RCA ou jack, en fonction de la connexion de la source audio.

Le schéma électrique

Le schéma électrique de l'amplificateur (voir la figure 1) a été simplifié au maximum par l'adoption d'un seul intégré monolithique qui contient en interne deux étages de puissance distincts capables de fournir en mode stéréo **2 x 5 W sur une charge de 4 Ω** et sous une tension d'alimentation de **12 V** et jusqu'à **14 V** au maximum, la puissance atteint dans ce cas environ **6 W par canal**.

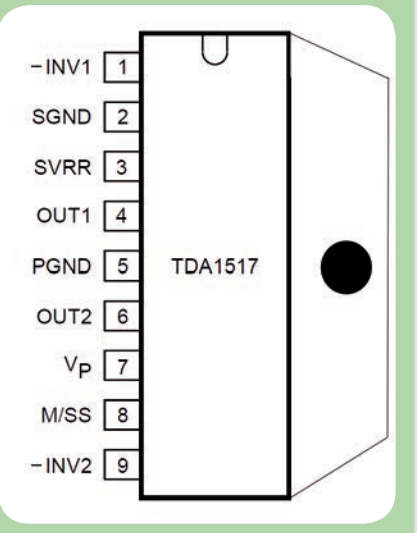
Le circuit intégré que nous utilisons est le **TDA1517** de la firme **Philips/NXP**, constitué de deux étages amplificateurs de puissance fonctionnant en **class B** en configuration asymétrique (entrée du signal BF entre un point chaud et la masse). Il est doté d'une **logique de contrôle qui surveille le fonctionnement du circuit**, d'une **fonction de mise sous tension progressive** (« soft-start ») qui évite le « **cloc** » dans les haut-parleurs et un circuit de « **muting** » (silencieux).

Mais analysons dans l'ordre le circuit. Les **entrées audio sont sur les broches 1 et 9**, respectivement les canaux gauche et droite, vous pouvez toujours les intervertir cela ne change rien. La **sortie de l'étage final de puissance gauche** est sur la **broche 4**, tandis que celle de l'**autre canal** est sur la **broche 6**, les deux sorties fonctionnent par rapport à la masse, de sorte que le côté négatif des haut-parleurs doit être relié à la piste de masse du circuit imprimé, comme illustré sur le schéma.

Etant donné que le circuit intégré est alimenté avec une alimentation simple et que les étages de puissance sont individuels (pas dans de pont) dans

Le circuit intégré TDA1517

L'amplificateur décrit ici est pratiquement entièrement contenu dans le circuit intégré **TDA1517** fabriqué par la firme **NXP** (une division de Philips).



Brochage du circuit TDA1517.

Ce composant est constitué de deux étages amplificateurs audio de puissance fonctionnant en **classe B**, dont chaque canal est capable de développer jusqu'à **6 W_{rms}** dans une charge **4 Ω**, avec une distorsion à pleine puissance de l'ordre de 10 %. Le circuit peut être alimenté avec une tension comprise entre 6 et 14 V, mais tolère **un maximum de près de 18 V**, chaque étage (canal) de l'amplificateur a un **gain fixe de 20 dB** (soit 40 fois la tension présente à l'entrée) ce qui permet d'obtenir 5 W sous 4 Ω avec une amplitude du signal d'entrée égale à environ **110 mV_{eff}**.

En plus des deux étages amplificateurs classe B, le circuit dispose d'une protection intégrée contre la surchauffe et d'une logique qui permet de mettre en veille les étages finaux lorsqu'un niveau de tension est appliqué sur la broche 8.

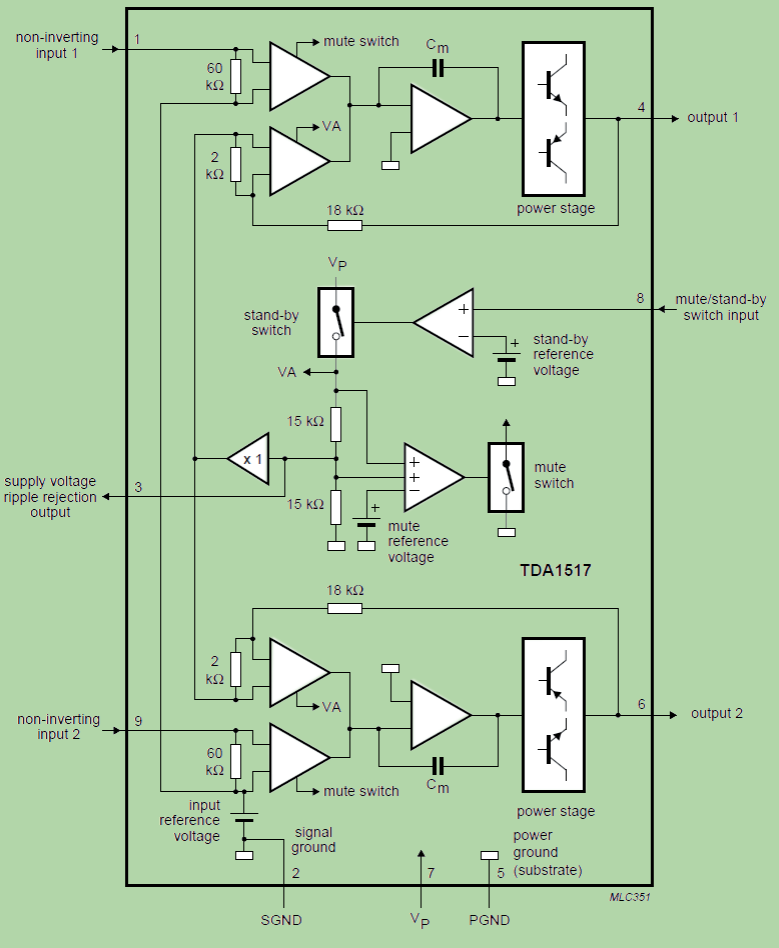


Schéma synoptique du circuit TDA1517.

les conditions de repos (c'est à dire en l'absence du signal) les broches 4 et 6 présentent un potentiel égal à environ la moitié de celui de l'alimentation.

Cette polarisation sert à faire en sorte que chaque sortie peut appliquer des valeurs de tension et de courant à la fois positives et négatives aux haut-parleurs, par rapport à la moitié du potentiel de l'alimentation ; au-dessus le courant est positif, au-dessous le courant est négatif. Pour que cela fonctionne, nous avons besoin de découpler chacun des haut-parleurs à l'aide d'un condensateur, comme vous le voyez sur le schéma électrique.

Au repos les condensateurs **C6** et **C7** empêchent que les **bobines des haut-parleurs soient parcourues par un courant qui pourrait les détruire**, alors qu'en présence d'un signal audio (on dit aussi en « **régime dynamique** »), les **condensateurs se chargent et se déchargent en inversant la polarité de la tension** et donc **le sens du courant traversant les haut-parleurs**.

Les condensateurs **C6** et **C7** sont de type électrolytique et compte tenu de la faible impédance de chaque sortie du circuit **TDA1517** et des haut-parleurs, pour assurer une fréquence de coupure inférieure assez basse pour couvrir le spectre des graves de la gamme audio, **ils ont des valeurs de capacité élevées**. Nous les trouvons dans le commerce seulement sous le forme de condensateurs électrolytiques.

Sachez que la **fréquence limite inférieure** (**fti**) que l'amplificateur peut reproduire sur chaque canal est donnée par la formule suivante :

$$fti = 1/6{,}28 \times Z \times C$$

où **Z** représente l'impédance du haut-parleur exprimée en Ω et **C** la **capacité** de C6 (C7) exprimée en **farads** (1 Farad = 1 000 000 µF).

En effet dans la formule nous devons tenir compte également de l'impédance de sortie de l'amplificateur, qui est de l'ordre de quelques dixièmes d'ohms et est **petite par rapport à Z** (donc nous négligeons son terme dans la formule).

CARACTÉRISTIQUES TECHNIQUES

- **Puissance de sortie : 2 x 5 W_{rms}**
- **Bande passante à – 3dB/8 Ω : 30 Hz à 100 000 Hz**
- **Distorsion harmonique à 2 x 5 W/1 kHz : 10 % sous 4 Ω**
- **Impédance de charge de 4 à 16 Ω**
- **Sensibilité d'entrée à 2 x 5 W sous 4 Ω : 110 mV_{rms}**
- **Impédance d'entrée : 60 kΩ**
- **Tension d'alimentation : 6 à 14 VDC**
- **Courant consommé : 1 A**

Tournons-nous maintenant vers le circuit d'entrée, qui véhicule le signal audio de la prise jack stéréo vers les entrées **IN1** et **IN2** du **TDA1517**.

En plus des **condensateurs de découplage C1** et **C2**, qui sont nécessaires pour éviter que toute **composante continue apparaisse sur broches 1** et **9** du circuit et se retrouver en sortie de l'amplificateur ce qui endommagerait les haut-parleurs (en fait, les condensateurs servent également à éviter que l'impédance de la source BF modifie la polarisation des étages d'entrée du circuit), nous trouvons le **potentiomètre double RV1a/RV1b qui règle le niveau du signal BF de la source audio** à amplifier et crée ainsi un contrôle efficace du volume d'écoute.

Si le **curseur va vers la masse le niveau diminue**, tandis que si le curseur va vers la **direction opposée le volume augmente**. Si vous réalisez le circuit imprimé en suivant le dessin des pistes de la figure 2, le volume augmente en tournant l'axe du potentiomètre vers la droite et baisse vers la gauche.

Continuons l'analyse du schéma, nous constatons que le condensateur **C4** est relié à la masse, il est utilisé pour filtrer les étages « drivers » internes du **TDA1517**. Il évite ainsi la propagation dans le circuit des ondulations résiduelles provenant de l'alimentation. De même sur la broche **7** (**VP**) qui correspond à l'alimentation du circuit, nous trouvons les condensateurs **C5** et **C3**.

C5 sert de réservoir d'énergie pour les étages de puissance notamment à **basse fréquence** lorsque l'ampli doit fournir un courant important aux haut-parleurs.

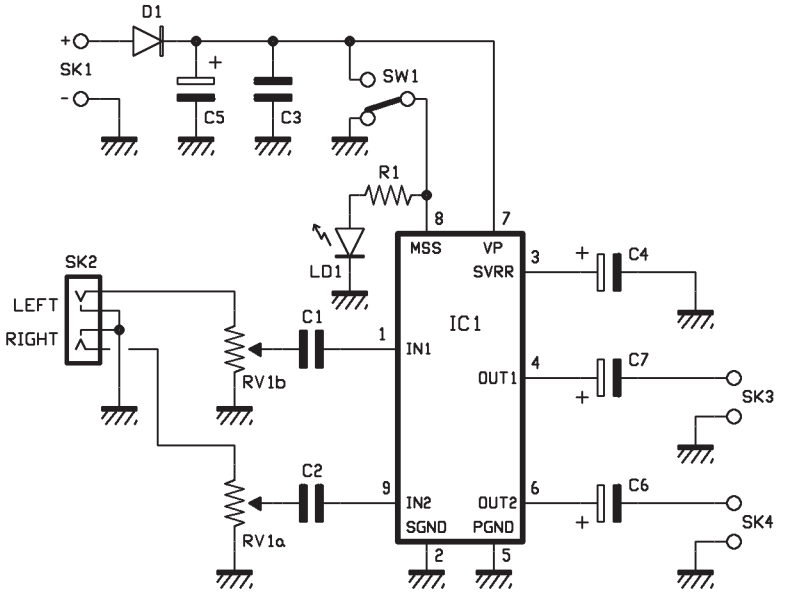


Schéma électrique de l'amplificateur, il a été simplifié au maximum grâce à l'utilisation du circuit TDA1517.

C3 filtre les parasites hautes fréquences présents sur la ligne d'alimentation. La **résistance électrique des pistes** de cuivre qui va de la prise d'alimentation jusqu'à la broche 7 doit être l**a plus faible possible** afin d'éviter une chute de tension, ce qui réduirait le rendement en fonction du signal audio (plus précisément, lorsque le courant dans les haut-parleurs augmente la tension sur la broche VP diminue légèrement).

Les ondulations résiduelles de l'alimentation peuvent se propager à l'intérieur du circuit intégré, et causer des variations dans la polarisation, résultant dans le meilleur des cas d'une distorsion élevée (et donc rendant le signal déformé et compromettant la fidélité) et dans le pire des cas provoquer un phénomène d'auto-oscillation.

Si le phénomène est en phase avec le signal, il augmente d'autant son amplitude en amorçant un cycle qui conduit à l'auto-alimentation du phénomène (le niveau sonore augmente l'amplitude de l'oscillation qui croît à son tour et entraîne une augmentation du niveau de l'audio, etc.). L'auto-oscillation empêche le bon fonctionnement de l'amplificateur en provoquant une surchauffe du **TDA1517**, elle doit donc absolument être évitée. C'est le rôle du condensateur **C4**.

Passons maintenant à la **logique de contrôle** qui est accessible via la **broche 8**. Elle vous permet d'obtenir un **démarrage progressif de l'amplificateur** et une fonction de **mise en sourdine** (silencieux ou « mute »), le démarrage progressif permet d'éviter le « **cloc** » classique dans les haut-parleurs lors de la mise sous tension, provoqué par le pic de courant présent dans les condensateurs déchargés C5, C6 et C7.

La fonction est réalisée par le raccordement d'un circuit RC entre l'alimentation positive du circuit (le condensateur doit être mis à la masse) et la broche 8, de sorte que la tension appliquée au circuit initialement sur cette broche est à un **niveau logique zéro**, puis atteint l'état haut avec un certain retard à partir duquel l'amplificateur est alimenté en évitant ainsi le « cloc » dans les haut-parleurs.

Circuit imprimé côté soudures à l'échelle 1 : 1 de l'amplificateur.

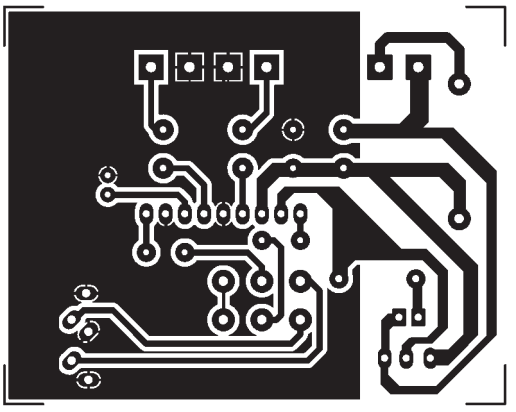


Schéma de câblage de l'amplificateur.

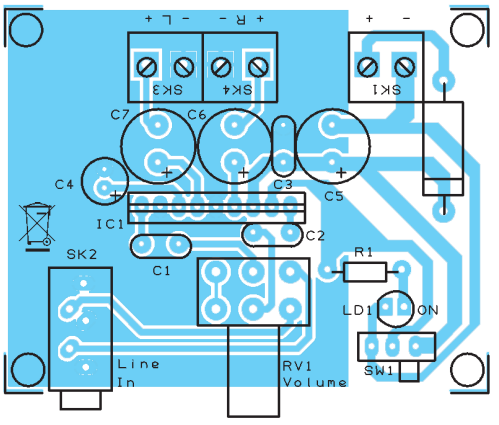
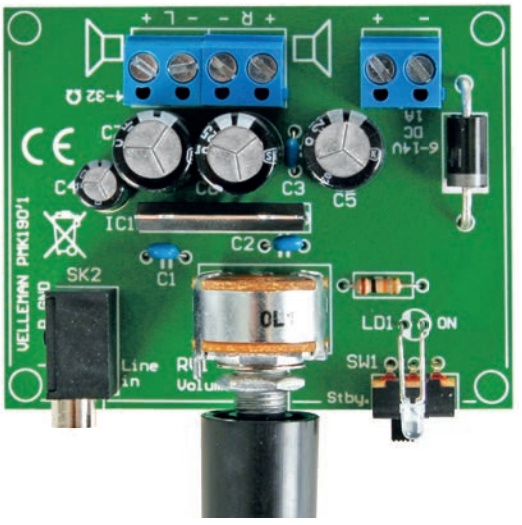


Photo de l'un de nos prototypes de l'amplificateur.



Liste des composants de l'amplificateur

R1..... 4,7 kΩ
RV1.... Potentiomètre double logarithmique 10 kΩ

C1..... 220 nF multicouche
C2..... 220 nF multicouche
C3..... 220 nF multicouche
C4..... 100 µF 25 V électrolytique
C5..... 1000 µF 25 V électrolytique
C6..... 1000 µF 25 V électrolytique
C7..... 1000 µF 25 V électrolytique
D1..... 1N5404
IC1..... TDA1517
LD1.... LED 3mm rouge
SW1... interrupteur coudé à 90°

Divers
connecteur jack stéréo 3,5 mm pour circuit imprimé
bornier 2 pôles au pas de 5 mm x 3
bouton pour potentiomètre

Le même raisonnement permet de s'assurer que, tant que la **broche 8 n'atteint pas un niveau logique haut, l'étage de puissance n'est pas alimenté**, et par conséquent cela permet d'obtenir un silence à la mise en marche.

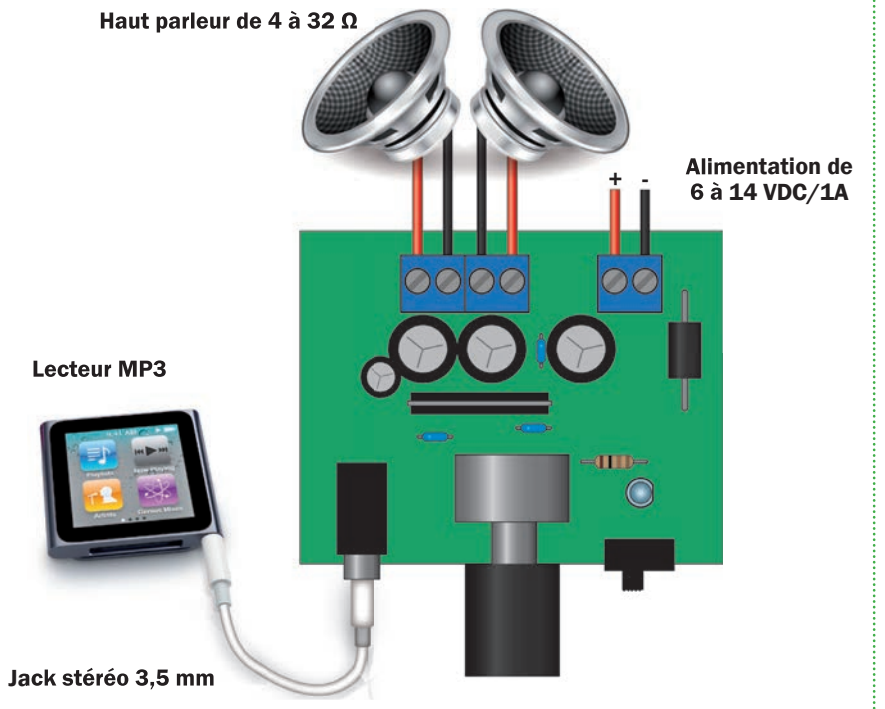
Dans notre cas, comme vous pouvez le voir sur le schéma électrique, nous avons confié le contrôle de la broche 8 (MSS) à un interrupteur **SW1**. Lorsque celui-ci est à la masse, l'amplificateur est coupé, tandis que s'il se trouve au potentiel de l'alimentation, l'amplificateur est en fonctionnement normal et la LED est allumée.

Par conséquent, **pour une utilisation optimale il faut toujours garder le circuit d'alimentation connecté, et allumer et éteindre l'amplificateur par SW1, de sorte que vous éviterez un « cloc » dans les haut-parleurs**. L'amplificateur de puissance consomme très peu en veille, seulement un courant de 100 µA.

Si vous le souhaitez, vous pouvez contrôler à distance l'amplificateur de puissance en connectant la **broche 8** à un niveau logique **TTL** (0/5 V) si la tension d'alimentation est inférieure à 9 V ou **CMOS** (0/12 V) si la tension d'alimentation est supérieure à 9 V, le niveau logique de contrôle doit être fourni par la source du signal audio que vous souhaitez amplifier.

Terminons la description du circuit avec l'alimentation, elle doit fournir une **tension continue non stabilisée de 9 VDC à 14 VDC**, bien que le **TDA1517** fonctionne avec seulement **6 V**. L'alimentation peut varier de 6 V à 14 V, vous pouvez même essayer de 5 V à 12 V. Avec 6 V vous obtiendrez un peu plus de 1 W avec des haut-parleurs de 4 Ω.

Comme expliqué précédemment, les condensateurs C3 et C5 sont utilisés pour filtrer les bruits sur l'alimentation, et lutter contre des ondulations positives et négatives de l'alimentation pouvant commencer à se propager lorsque l'amplificateur fonctionne à puissance maximale, La diode **D1 protège le circuit en cas d'inversion de polarité de l'alimentation**.



Réalisation pratique et tests

La construction de l'amplificateur est à la portée de tous, car il n'y a pas de composants critiques et le circuit peut également être monté sur une plaque d'essai à insertion.

Mais dans tous les cas nous vous conseillons d'utiliser un circuit imprimé que vous pouvez facilement graver en **téléchargeant le typon sur notre site www.electroniquemagazine.com dans le sommaire de la revue 126**.

Le circuit une fois gravé et les trous percés, vous devez d'abord souder la résistance R1 et la diode D1, puis continuer avec les condensateurs, en donnant la priorité aux non-polarisés.

Ensuite vous soudez les condensateurs électrolytiques en prêtant attention au sens d'insertion. Le « - » (moins) est indiqué sur le boîtier (ils doivent tous être du côté des borniers) et la patte la plus longue correspond au « + » (plus). Faites attention au sens de la diode, la bague doit être montée vers SW1 comme indiqué sur le schéma de câblage (la bague indique la cathode).

Ensuite passez à la LED, la cathode est indiquée par un méplat sur le boîtier, montez le commutateur SW1, les deux borniers au pas de 5 mm pour les haut-parleurs et pour l'alimentation, la prise stéréo jack et enfin le circuit intégré **TDA1517**, dont le détrompeur en « U » doit être positionné vers le côté gauche lorsque vous avez le potentiomètre face à vous.

Ce composant doit être soudé en maintenant le boîtier à environ 5 mm au-dessus du circuit imprimé et doit être équipé d'un dissipateur thermique de 15 °C/W, vous pouvez utiliser une plaque en aluminium de 2 mm d'épaisseur ayant une surface de 5 par 5 cm.

A propos du potentiomètre, il peut être soit du type linéaire ou logarithmique, bien que ce dernier soit idéal pour le contrôle du volume, en vertu du fait que la courbe de sensibilité de l'oreille humaine est logarithmique. Ce qui signifie que la **personne perçoit bien l'augmentation du niveau acoustique lorsque la croissance est de type logarithmique**.

Rappelons que les potentiomètres linéaires sont identifiés par le suffixe A après la valeur, tandis qu'un logarithmique par un suffixe B. Un potentiomètre marqué 47 KA est linéaire, tandis qu'un 47KB est logarithmique.

Après avoir terminé le montage, vous pouvez relier la sortie de l'amplificateur à deux haut-parleurs en y connectant un au + et - R (côté droit) et l'autre au + et – L (côté gauche) de 5 W 4 Ω ou deux enceintes acoustiques de mêmes caractéristiques ou plus puissantes (cela n'est que mieux), le câble rouge au positif et le noir au négatif afin d'éviter des erreurs de phase. Appliquez à la prise jack d'entrée le signal provenant de la sortie audio d'un PC ou d'un casque d'un lecteur MP3.

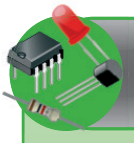
Branchez l'alimentation 12 V au bornier SK1 du circuit en respectant la polarité + et -. Mettez temporairement l'interrupteur SW1 à la masse (vérifiez que le voyant est éteint), maintenant allumé la source du signal audio (PC ou lecteur MP3 ou tout autre chose que vous avez connecté à l'entrée) en réglant au minimum le contrôle du volume.

Positionnez SW1 sur la position ON (vers le positif), vous devez voir la LED s'allumer. Peu de temps après, en tournant progressivement le potentiomètre dans le sens horaire, vous devez entendre le son dans les haut-parleurs.

Une fois que vous avez vérifié que tout fonctionne correctement, éteignez-le et procurez-vous un boîtier en plastique approprié pour l'amplificateur. Vous devez percer les trous pour le passage du cordon d'alimentation, l'axe du potentiomètre (sur lequel vous monterez un bouton) et la prise d'entrée jack. Vous pouvez également créer une « box » contenant les deux haut-parleurs placés chacun d'un côté, ou bien deux petits caissons amplifiés. Dans ce dernier cas, mettez le circuit dans un caisson et reliez l'autre à l'aide d'un câble en ruban rouge et noir, celui-ci sera simplement un haut-parleur passif.

Sortez l'axe du potentiomètre, et rendez accessible la prise jack sur la face avant, placez la prise d'alimentation sur le panneau arrière, de sorte que vous pouvez alimenter l'amplificateur avec une alimentation dotée d'une prise universelle et capable de fournir au moins 9 à 12 VDC avec un courant d'au moins 1 A (de préférence 1,2 A). L'alimentation doit être de préférence de type linéaire, c'est-à-dire composée d'un simple transformateur, d'un pont de diodes pour le redressement et de condensateurs de filtrage.

Vous pouvez également **utiliser une alimentation à découpage, mais dans ce cas, nous vous conseillons de mettre entre la borne positive de l'alimentation et le « + » du bornier un filtre LC composé d'une bobine de 1 ou 2 mH** (capable de fournir 1 A) **en série et deux condensateurs en parallèle entre le « + » et « - », de chaque côté de la bobine de valeur 2200 µF/25 V**. Sans ce filtre, il est probable que des résidus de la commutation de l'alimentation interfèrent avec la musique en sortie.



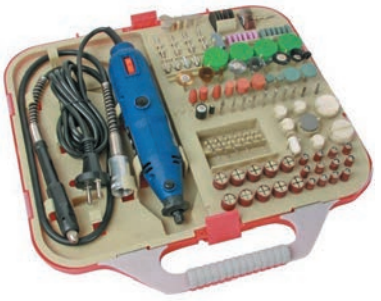
Comment construire ce montage

Tous les composants utilisés dans ce projet sont facilement disponibles et le typon du circuit imprimé est téléchargeable sur le site de la revue. L'amplificateur est disponible sous forme de kit (MK190) ou monté (MK190/KM) auprès de la société COMELEC. Le kit comprend le circuit imprimé percé et sérigraphié ainsi que tous les composants.

OUTILLAGES - ACCESSOIRES - COFFRETS

KIT D'INITATION AU SOUDAGE
EDU03 26,99 €

PERCEUSE DE PRÉCISION + 162 ACCESSOIRES
VTHD05 45,15 €



TROISIÈME MAIN AVEC LOUPE, LAMPE LED ET SUPPORT POUR FER À SOUDER
VTHH3 17,96 €

POMPE À DESSOUDER
VTD4 2,51 €

JEU DE TOURNEVIS DE PRÉCISION 32 EN 1
VTSCRSET10 12,60 €

CHALUMEAU À GAZ PROFESSIONNEL
GASMTPRO 47,16 €

GRAVEUSE VERTICALE AVEC POMPE ET RÉSISTANCE
ET20 107,36 €

JEU DE 4 PINCES BRUCELLES ANTISTATIQUES
VTTWSET2 4,11 €

VALISE D'OUTILS POUR CÂBLES RÉSEAU
VTMUS3 69,90 €

PERCEUSE DE PRÉCISION + 6 ACCESSOIRES
VTHD01 6,82 €

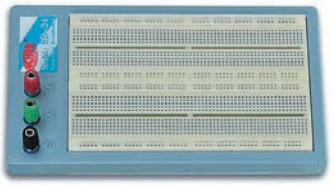
STATION DE RÉPARATION POUR CMS
VTSS100 184,45 €

EXTRACTEUR DE CIRCUIT INTÉGRÉ (FORMAT DIL)
VTIC 1,81 €

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

OUTILLAGES - ACCESSOIRES - COFFRETS

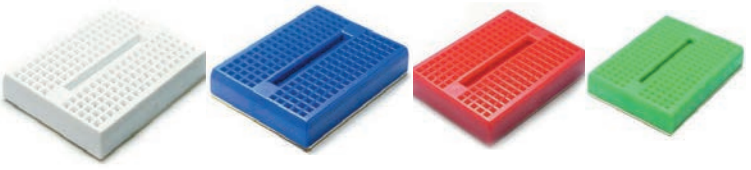
PLAQUE DE CONNEXIONS SANS SOUDURE 1680 TROUS
SD24N 21,07 €



COFFRET EN PLASTIQUE MOULE - GRIS FONCE 150 X 80 X 60MM
G418 6,42 €

LAMPE-LOUPE LED 3 + 12 DIOPTRIES 4~5.5W - 60 PCS - BLANC
VTLLAMP3W 46,96 €

MINI PLAQUE D'ESSAI (COULEUR AU CHOIX)
BBMINI (W) (B) (R) (V) 2,91 €



JEU DE TOURNEVIS DE PRÉCISION 28 EN 1
VTBT15 20,77 €

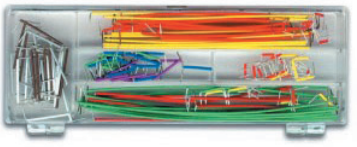
BOÎTE DE 20 FRAISES DIAMENTÉES
VTHDS2N 3,91 €

JEU DE 6 OUTILS D'AIDE AU SOUDAGE.
VTSA 6,50 €

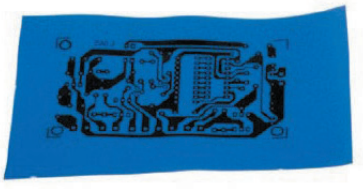
JEU DE 3 PINCES
VT33N 8,63 €

KIT D'INITATION AU SOUDAGE
EDU03 26,99 €

STRAPS POUR PLAQUES DE CONNEXIONS SANS SOUDURE
WJW70 6,42 €



PNP BLUE (LOT DE 5)
PNP 5 10,94 €



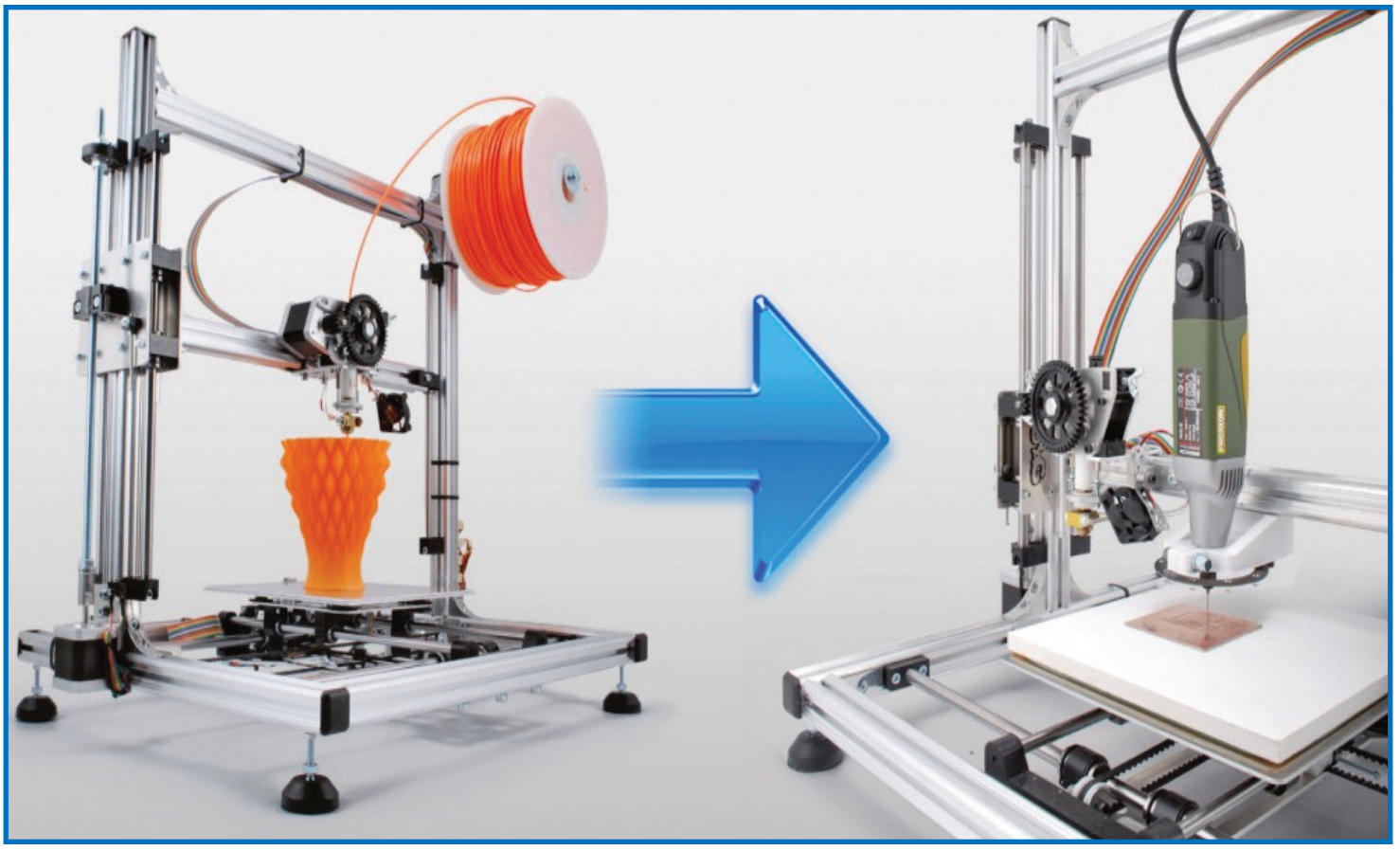
JEU DE 100 EMBOUTS
VTBT11 11,64 €

OUTIL MULTIFONCTIONS
VTMPP5 7,12 €

JEU DE 6 TOURNEVIS DE PRÉCISION ('STAR')
VTSET30 5,92 €

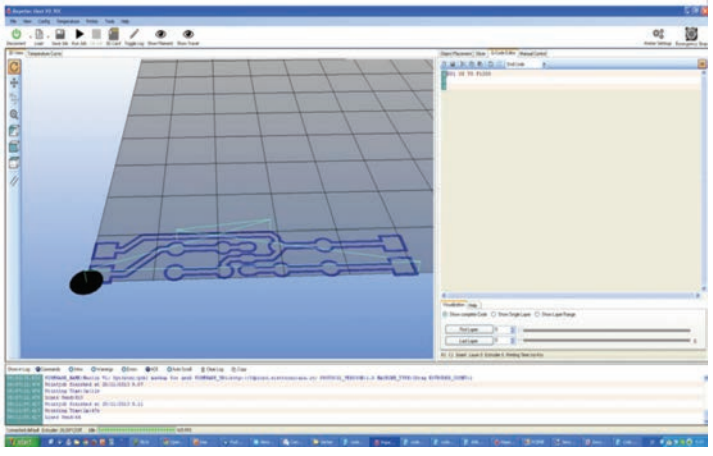
PASSEZ VOS COMMANDES DIRECTEMENT SUR NOTRE SITE ! www.comelec.fr

Transformez la 3DRAG en graveuse pour circuits imprimés

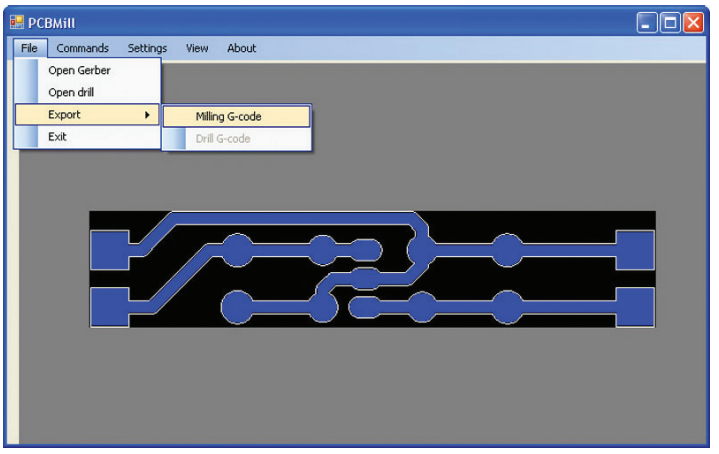


A partir du prochain numéro d'Électronique et Loisirs Magazine, nous allons décrire une série d'articles pour transformer l'imprimante 3DRAG en une fraiseuse mécanique à commande numérique pour la production de circuits imprimés. Nous étudierons dans un premier temps les procédures pour obtenir un fichier G-Code nécessaire à l'imprimante pour graver le circuit à partir de tout fichier GERBER. En particulier nous développerons un exemple concret à l'aide des logiciels PCBMill et EAGLE CAD.

Dans un deuxième temps, nous étudierons les étapes pour passer d'un fichier image au format BMP à un format G-Code, c'est-à-dire qu'à partir d'un fichier image BMP vous pourrez fabriquer un circuit imprimé à l'aide de la 3DRAG. Nous avons étudié ce système particulièrement pour les passionnés d'électronique et les petites entreprises qui veulent réaliser des prototypes dont la fabrication du circuit imprimé est très couteuse.



Passage d'un fichier image BMP à un fichier G-Code sous Repetier-Host à l'aide du programme ArtCam.



Passage d'un fichier GERBER sous EAGLE à un fichier G-Code sous Repetier-Host à l'aide de PCBMill.

ABONNEZ-VOUS

OUI, Je m'abonne à ELECTRONIQUE ET LOISIRS magazine — LE MENSUEL DE L'ÉLECTRONIQUE POUR TOUS

A PARTIR DU N° 127 ou supérieur

E0126

N°

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** **28€,00**

TARIFS CEE/EUROPE

❑ **4 numéros** **32€,00**

DOM-TOM/HORS CEE OU EUROPE :

NOUS CONSULTER SUR 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 des anciens numéros

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. 0820 820 534 – Fax 0820 820 722

Directeur de Publication
Rédacteur en chef
Jean Marc MOSCATI

Directeur Technique
Charles CARDONA

Direction - Administration
JMJ éditions
B.P. 20025
13720 LA BOUILLADISSE
Tél. : 0820 820 534

Secrétariat - Abonnements
Petites-annonces - Ventes
A la revue

Vente au numéro
A la revue

Publicité
A la revue

Maquette - Illustration
Composition - Photogravure
JMJ éditions sarl

Impression
Print Courtage
25 Bd Bouès
13003 Marseille

Distribution
NMPP

Hot Line Technique
0820 820 534 *
du lundi au vendredi de 16 h à 18 h

Web
www.electroniquemagazine.com

e-mail
support@electroniquemagazine.com
* prix d'un appel local

JMJ éditions
Sarl au capital social de 7800 €
RCS MARSEILLE : 421 860 925
APE 221E
Commission paritaire: 1015T79056
ISSN: 1295-9693
Dépôt légal à parution

ELECTRONIQUE ET LOISIRS magazine — LE MENSUEL DE L'ÉLECTRONIQUE POUR TOUS

EST RÉALISÉ EN COLLABORATION AVEC :

NUOVA ELETTRONICA
Elettronica In

9,90 € la revue Frais de port inclus pour la France

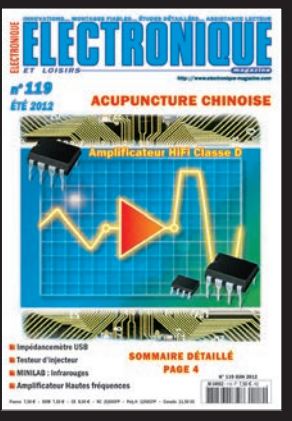
(Calcul des frais de port pour la CEE, les DOM-TOM et autres Pays directement dans le panier sur notre site www.electroniquemagazine.com)

Au sommaire : Chargeur de batterie sans fil, recharger vos batteries au plomb - Détecteur de métaux à impulsions - Taser/dissuadeur anti agression, dispositif délivrant des impulsions à haute tension, portatif - MINILAB : Lumières psychédéliques à LED - Convertisseur d'ultrasons en sons audibles - Doubler la puissance du linéaire RF 88-108 MHz - Un automatisme pour faire face aux coupures de courant, ce montage vous permet de rallumer automatiquement les appareils domestiques mais pas tous en même temps, afin d'éviter le désagrément du black-out à répétition - Luminaire à LED en 230 V réglable par variateur Etc...

Au sommaire : Récepteur DRM s'affranchir des frontières pour l'écoute radio - MINILAB : Lumières psychédéliques Variateur 230 VAC à MOSFET - iTESLA la note aiguë d'une soprano, le chant d'un rossignol ou la totalité d'un morceau de musique peuvent-ils être reproduit fidèlement Signalisation d'alarme multifonction cet automatisme simple se prête à de multiples exigences - Sirène-flash anti agression - Alarme anti inondation capacitive, une fuite d'eau peut rapidement se transformer en une petite catastrophe domestique - Facteur Q, leçon d'approfondissement, examinons le Quality Factor des composants électroniques. condensateurs, selfs et des circuits résonants.

Au sommaire : «Theremin» en version professionnel - Deux alimentations à découpage avec dimensions réduites et à la possibilité d'obtenir une vaste gamme de tensions - Emetteur FM 88-96 MHz à construire sur la plaque d'essais du Minilab - Convertisseur N/A USB, avec ce microscopique convertisseur R2R nous allons transformer notre interface USB EN1741 en convertisseur N/A Numérique/Analogique) - Synthétiseur de 143 MHz à 970 MHz qui, relié au générateur DDS, peut fournir n'importe quelle fréquence comprise entre 143 MHz et 970 MHz avec une résolution de 10 Hz. - Platine universelle pour LM358 - Antenne active pour ondes courtes.. Etc...

Au sommaire : Surveiller les fissures des murs avec l'USB. Stand-by (veille) off réactivable avec la télécommande: Réduisez votre facture d'électricité. Mesurer la distorsion avec un simple multimètre - Un selfmètre pour mesurer l'inductance des selfs - Mesurer la température avec le Minilab - Platine universelle pour LM358 - un amplificateur différentiel avec alimentation simple. Un sommateur inverseur et non inverseur avec alimentation double un convertisseur tension / courant un comparateur trigger de Schmitt - un intégrateur inverseur - un dérivateur inverseur - un amplificateur pour DDS. Etc...



Au sommaire : ÉLECTRORÉFLEX le générateur d'ondes «chinoises» utilisés pour contrer les douleurs aiguës de différentes origines - Amplificateur Hi-Fi stéréo 2x20 W en classe D, amplificateur aux dimensions réduites, que vous pourrez relier à votre Ipod, mp3 - Les rayons infrarouges avec le Minilab, expérimentations qui vous aideront à comprendre comment fonctionnent les dispositifs électroniques utilisant ces invisibles radiations électromagnétiques . Impédancemètre USB pour PC Seconde partie, le logiciel - Testeur d'injecteur pour automobile - Les amplificateurs RF à MMIC. très intéressants pour celui qui opère dans le domaine de la radiofréquence . Etc...

CD-ROM ENTIÈREMENT IMPRIMABLE

50 € Les 3 CD du Cours d'Électronique en Partant de Zéro

COURS Niveau 1,2 ou 3 19 € l'unité

Numéros spéciaux l'unité 5,50 €

CD - FRAIS DE PORT INCLUS POUR LA FRANCE (DOM-TOM ET AUTRES PAYS: NOUS CONSULTER.)

JMJ/ELECTRONIQUE - B.P. 20025 - 13720 LA BOUILLADISSE *règlement par Chèque à l'ordre de* ***JMJ ÉDITIONS***

Règlement par Carte Bancaire sur notre site : **www.electroniquemagazine.com** - Téléphone : 0820 820 534

Articles, Revues et CD téléchargeables au format PDF sur Internet

http://www.electroniquemagazine.com

LISEZ ET IMPRIMEZ VOTRE REVUE SUR VOTRE ORDINATEUR PC OU MACINTOSH

Au sommaire : Analyseur de spectre pour PC - Un récepteur FM à super-réaction avec une série de propositions d'applications pour le Minilab - Une barre lumineuse à LED pour téléviseur - Amplificateur linéaire RF large bande avec une paire de MOSFET PD55015 - Applications XOR et XNOR avec le programmateur CPLD, dédié aux applications pratiques réalisées avec notre programmateur pour dispositifs CPLD EN1685 - Un montage à ultrasons intéressant - Un antivol à ultrasons - Mini alimentation 9-12-15 V 0,4 A, conçue pour alimenter de petits circuits expérimentaux réclamant une tension de 9-12-15 V et un courant ne dépassant pas 0,4 A. Etc...

Au sommaire : Le QR CODE pour accéder rapidement à des contenus internet OPEN SOURCE : logiciel open ou free - La simulation de la 3D, est une technique de réalisation et de visualisation d'images, de dessins, photographies et films - Un micro stéréo préamplifié disposant de deux canaux indépendants - MINILAB : Expérimenter les CMOS - LTSpice pour apprendre à simuler vos circuits - Chargez les NiCd et NiMh avec votre alimentation. Il s'agit d'une manière intelligente et écologique de produire de l'énergie - Magnétothérapie RF professionnelle et portative - Le sismographe... ou ces secondes maudites - COURS: Le bruit des résistances. Etc...

Au sommaire : Localisateur GSM/GPS, combine les techniques de localisation sur le réseau cellulaire GSM et le réseau satellite GPS - L'imprimante 3D qui existe depuis des décennies dans le monde professionnel, mais depuis deux ans environ commence à faire partie intégrante du monde des amateurs - Un lecteur MP3 Juke-Box sur clé usb 16 go - Comment utiliser une LED comme veilleuse et comme détecteur de lumière - Interrupteur crépusculaire miniature - RaspberryPi un PC au format carte de crédit : 1 ere partie, PC de faible coût , fonctionnant sous un système d'exploitation libre GNU/Linux - Amplificateur stéréo 2 x 10 W - Cours : Tutoriel EAGLE CAD V 6

Au sommaire : Antivol pour panneaux solaires - L'imprimante 3D, le montage et les premiers pas, nous avons finalement décidé de vous dévoiler notre interprétation du projet d'imprimante 3D - RaspberryPi la programmation : 'il fonctionne comme un système embarqué, objectif, écrire un premier programme - MINIBUS, la qualité du signal a été améliorée afin de couvrir de longues distances entre les divers équipements. - Transmission audio par LED, utiliser son émission de lumière pour transmettre un signal audio en FM (Modulation de Fréquence). - Récepteur 4 canaux compatible MM53200, UM3750 et UM86409 Etc...

Au sommaire : Un serveur web avec le RaspberryPi, transformer le RaspberryPi en un système complet de gestion du port GPIO à distance (en réseau) - Chargeur d'accumulateur universel - Préamplificateur stéréo avec commande de tonalité numérique - 3DRAG 3 ème partie de l'idée à l'objet , nous allons nous concentrer sur le processus qui permet de passer de l'idée à un objet fini - MINIBUS, un bus pour l'automatisation, cet appareil permet de transmettre un signal audio en utilisant la ligne secteur 230 VAC de sorte qu'il soit disponible partout dans la maison sans avoir à tirer des câbles ou acheter des émetteurs/récepteurs radios Etc...

CD 6 Numéros 25 € / CD 12 Numéros 45€

50% de remise pour nos abonnés sur tous les CD de 6 ou 12 numéros

Contrôle à distance
Utiliser son téléphone GSM
comme télécommande et récepteur d'alarme

Comelec 03 / 2014 - Prix TTC

Installations antivol pour les bâtiments civils et industriels, pour les voitures, contrôle des systèmes de climatisation, chauffage, contrôle de pompes et de systèmes d' irrigation, contrôle d'installations industrielles etc.

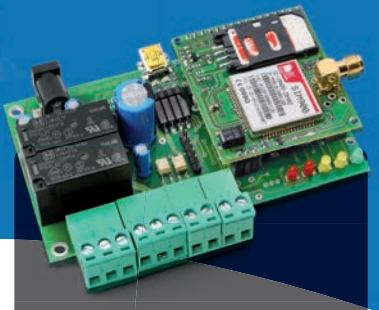
SYSTÈME GSM INTELLIGENT

- 8 numéros pour envoyer des alarmes par appel vocal, SMS ou les deux à la fois
- 2 entrées d'alarme avec messages personnalisables.
- 2 sorties relais pouvant être gérées avec des tonalités DTMF et programmables par commandes SMS.
- Possibilité d'associer des entrées aux événements d'alarme et de contrôle à distance.
- Alimentation 9 à 32Vdc/1A.

réf. TDG135
115,50 €

réf. TDG133 • 87,00 €
TÉLÉCOMMANDE GSM
bidirectionnelle
2 entrées/2 sorties relais

réf. TDG134 • 83,00 €
TÉLÉCOMMANDE GSM
ouvre portail 1 sortie relais



réf. TDG140 • 94,50 €
TÉLÉCOMMANDE GSM
avec commandes DTMF

Toutes les télécommandes sont facilement configurables à l'aide de SMS ou localement via PC.
Pour la connexion au PC, vous avez besoin d'un module d'interface USB. ET782M (Module convertisseur USB / Serie, vendu 14,50 € en option).
Les appareils sont livrés assemblés et testés certifiés CE-R & TTE.

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