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

ELECTRONIQUE ET LOISIRS

# ELECTRONIQUE magazine

www.electroniquemagazine.com

n°129 HIVER 2014

# Afficheur LCD pour RaspberryPi

- Lecteur MP3
- Générateur de fonction DDS
- Les commandes électroniques de vol
- Récepteur de télécommande infrarouge




## Connectez la 3DRAG au RaspberryPi

## COURS ANDROID 2ème partie

N° 129 Décembre 2014

M 04662 - 129 - F: 7,50 € - RD

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

# Sommaire

## ARTICLES

**Numéro 129**
**Hiver 2014**
**100 pages**

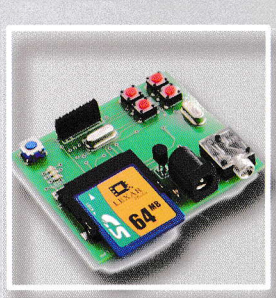




05 AUDIO

**UN LECTEUR MP3 UNIVERSEL**

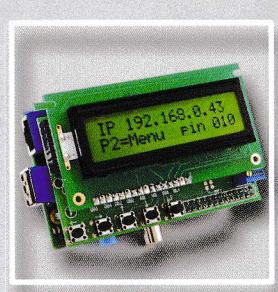
Indispensable et polyvalent, ce lecteur MP3 reproduit de la musique à partir d'une carte SD en fonction de vos besoins, il peut aussi être utilisé pour la reproduction de spots publicitaires ou de messages vocaux. Il s'agit d'un module conçu pour être intégré dans d'autres systèmes où il peut être nécessaire de reproduire un message préenregistré, par exemple une annonce sous forme de fichiers MP3.




19 INFORMATIQUE

**AFFICHEUR LCD POUR RASPBERRYPI**

Nous allons décrire dans cet article une carte d'extension avec un afficheur LCD pour le RaspberryPi. Elle vous permettra de réaliser une interface de contrôle externe pour vos applications sans la nécessité de connecter en permanence un moniteur, un clavier et une souris. Dans les prochains articles nous examinerons plus en détail le fonctionnement et l'utilisation de ces deux cartes d'extensions.




36 LABORATOIRE

**GÉNÉRATEUR DE FONCTION DDS DE 0,1 HZ À 5 MHZ**

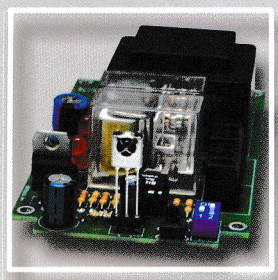
Nous vous proposons dans cet article un générateur de fonction à synthétiseur numérique DDS qui peut générer un signal sinusoïdal, carré et triangulaire jusqu'à une fréquence de 5 MHz. Il dispose d'un contrôle d'amplitude et d'offset (décalage) pour tous les types de signaux. Il peut être géré par le port série d'un PC via un programme de contrôle simple fonctionnant sous Windows.



48 TÉLÉCOMMANDE

**RÉCEPTEUR UNIVERSEL 2 CANAUX POUR TÉLÉCOMMANDE INFRAROUGE**

Ce récepteur infrarouge peut être commandé avec les télécommandes infrarouges des téléviseurs, des appareils Hi-Fi, des lecteurs de DVD et des magnétoscopes. Il dispose de 2 sorties sur relais qui peuvent commuter des dispositifs fonctionnant sous 230 VAC. Ce récepteur infrarouge à auto-apprentissage est capable d'interpréter les codes des télécommandes infrarouges.




Les typons des circuits imprimés et les programmes **lorsqu'ils sont libres de droits** sont téléchargeables à l'adresse : **www.electroniquemagazine.com** dans le sommaire de la revue 129 et à l'onglet « Télécharger »

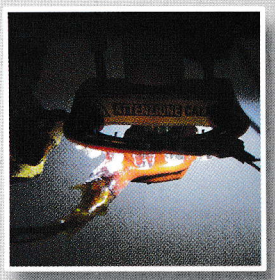
54 TECHNOLOGIE
**EN VOL AVEC L'ÉLECTRONIQUE**

Depuis les frères Wright, qui ont effectué en 1903 le premier vol motorisé d'un avion, jusqu'à nos jours nous allons découvrir comment les systèmes électroniques embarqués ont évolué, aidant les pilotes à être plus à l'aise et confiants, mais aussi plus respectueux de l'environnement. Le premier automatisme important qui a été adopté dans l'aviation civile fut le « pilote automatique ». Il a été développé en 1914 par Lawrence Sperry.



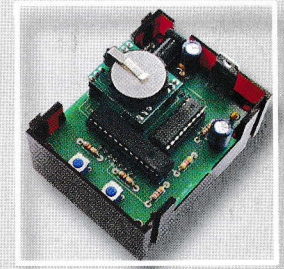
70 IMPRIMANTE 3D
**FAISONS DE LA LUMIÈRE POUR LA 3DRAG**

La Rédaction tient à remercier les nombreux lecteurs qui ont félicité les deux articles consacrés à la fabrication des circuits imprimés à partir de l'imprimante 3DRAG. Dans cet article nous vous proposons encore d'améliorer la 3DRAG en lui ajoutant un anneau d'éclairage à LED au niveau de la tête d'extrusion.



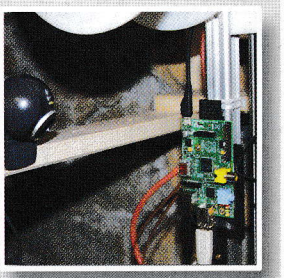
76 HI-TECH
**HOROLOGE BINAIRE**

Au lieu d'utiliser de classiques horloges à cadrans traditionnels avec des aiguilles ou avec un affichage numérique, cette horloge binaire affiche l'heure en utilisant des LED, dont chacune est associée à une valeur binaire. Les « Heures », « Minutes » et « Secondes » sont chacune visualisées par deux rangées de six LED.



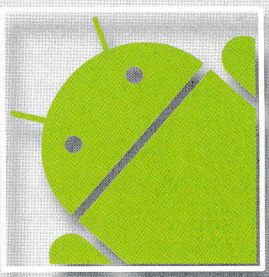
82 IMPRIMANTE 3D
**CONTRÔLEZ À DISTANCE LA 3DRAG AVEC LE RASPBERRYPI**

Dans cet article nous vous proposons d'imprimer à distance et de contrôler la 3DRAG à l'aide d'une solution LINUX embarquée qui n'est autre que le RaspberryPi, sans accaparer un ordinateur. Il nous est arrivé d'être lassés de devoir laisser notre ordinateur pendant des heures sans pouvoir s'en servir pour d'autres tâches.



87 COURS
**PROGRAMMEZ AVEC ANDROID DEUXIÈME PARTIE**

Dans cette deuxième partie, nous continuons le cours sur Android et plus particulièrement sur l'environnement de développement « Eclipse ». Nous allons émuler notre première application et l'exécuter sur un périphérique réel. Le projet que nous vous proposons a une valeur didactique et vous permettra de comprendre le processus complet de programmation.




## L'EDITO HIVER 2014

Chères lectrices, chers lecteurs,

La Rédaction d'Electronique et Loisirs Magazine vous souhaite de Joyeuses Fêtes de Noël ainsi qu'une Bonne et Heureuse Année 2015.

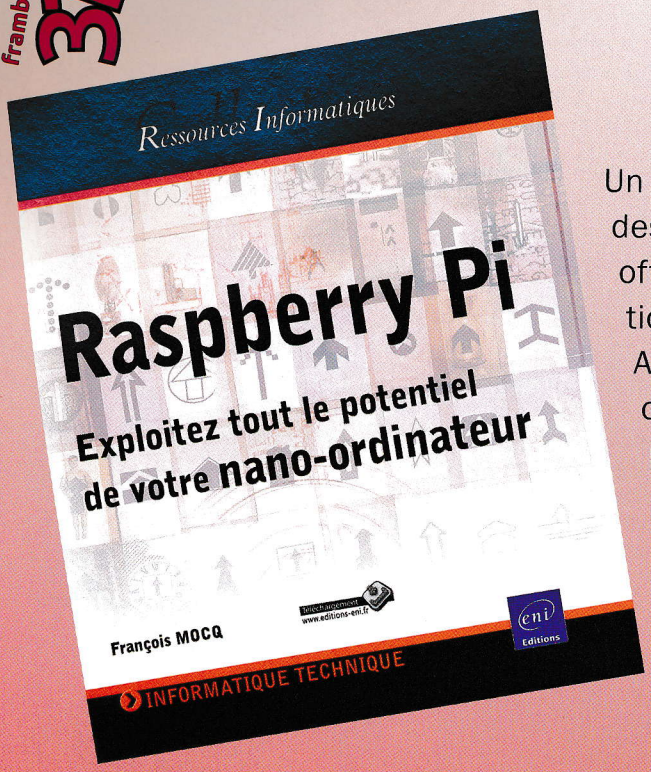
Dans ce numéro nous continuons la description d'appareils pour le laboratoire avec un générateur de fonction DDS. Dans le prochain numéro nous décrirons un fréquencemètre digital. En ce qui concerne le RaspberryPi nous vous présentons dans ce numéro une carte d'extension à afficheur LCD compatible avec la nouvelle version B+ du RaspberryPi. Celle-ci vous permettra de vous passer de l'écran habituel et d'utiliser le RaspberryPi dans des endroits où l'espace est limité. D'autre part dans les prochains numéros nous reviendrons sur l'aspect programmation des cartes d'extensions et plus particulièrement dans le numéro 130 où nous aborderons les applications de la carte A/D 16 bits pour RaspberryPi compatible Arduino décrite dans le numéro 128.

Dans notre laboratoire nous sommes en train d'étudier la nouvelle version du RaspberryPi B+ dont vous pouvez voir ci-dessous l'aspect avec ses 4 ports USB et ses 4 trous de fixation, nous vous proposerons des articles prochainement. Nous continuons aussi le développement de l'imprimante 3DRAG et nous aborderons les aspects pratiques à savoir par exemple comment fabriquer un boitier sur mesure pour vos réalisations.



# Framboise 314

**Le Raspberry Pi à la sauce française**




Un livre de référence destiné à fournir au lecteur des bases solides pour explorer les ressources offertes par le Raspberry Pi (système d'exploitation, développement et interfaçage physique). Aucun pré-requis en Linux, en programmation ou en électronique n'est nécessaire. La compatibilité du GPIO du Raspberry Pi B+ avec le modèle B permet d'utiliser cet ouvrage comme base avec le B+.

## Mais aussi :

### Le blog

L'actualité du Raspberry Pi
Des informations
Des tutoriels
Le Raspberry Pi dans la presse

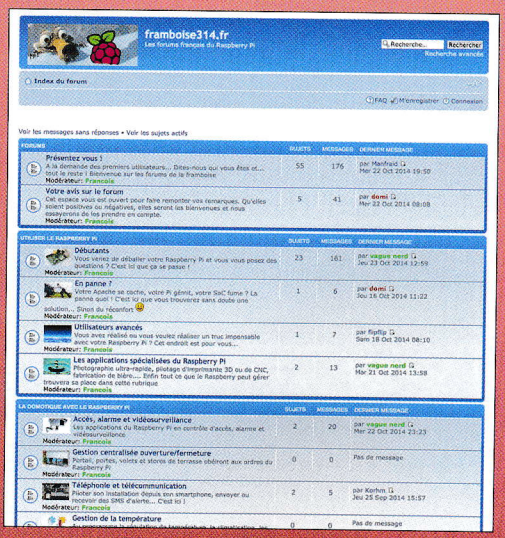






### Le forum

Une communauté
Des experts
De l'entraide




@framboise314

Framboise314

**www.framboise314.fr**

# UN LECTEUR MP3 UNIVERSEL




**Indispensable et polyvalent, ce lecteur MP3 reproduit de la musique à partir d'une carte SD en fonction de vos besoins, il peut aussi être utilisé pour la reproduction de spots publicitaires ou de messages vocaux.**

*de Luca Pascarella*

De nos jours, nous connaissons tous le « **lecteur MP3** », en particulier les portables pour une écoute individuelle au casque. Ils sont très répandus grâce à la grande disponibilité des produits Chinois, qui ont un coût abordable et des capacités pratiquement illimitées. Ils disposent d'écrans couleur et de boutons permettant d'écouter de la musique dans toutes les situations possibles, en bus, à vélo ou en faisant du jogging.

C'est peut-être pour cette raison qu'en regardant le projet que nous vous proposons, vous serez un peu dérouté, car le lecteur **MP3** que nous avons réalisé **est un peu différent des modèles disponibles dans le commerce**.

En effet il ne possède pas de batterie pour l'alimenter, ni d'un afficheur pour interagir avec l'utilisateur, sa mémoire est externe et amovible pour stocker les morceaux à reproduire. En résumé, notre lecteur MP3 est simple, vous ne pouvez pas le transporter ou y stocker de la musique, alors il est naturel de se poser la question suivante : à quoi sert-il ?

Et bien, notre projet est très simple, car **il s'agit d'un module conçu pour être intégré dans d'autres systèmes**, en somme il s'agit d'un lecteur générique à utiliser dans des contextes où il peut être nécessaire de reproduire un message préenregistré, par exemple une annonce sous forme de **fichiers MP3**.

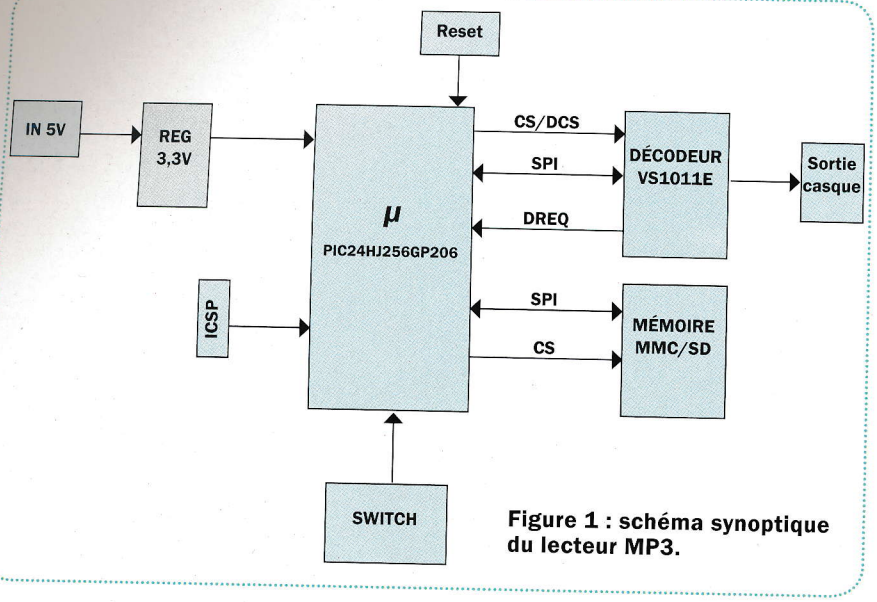



**Figure 1 : schéma synoptique du lecteur MP3.**

Il peut servir pour réaliser une « **box** » contenant des messages publicitaires qui attirent l'attention de la clientèle sur des articles dans un magasin ou aussi pour concevoir des gadgets tels qu'un coffret cadeau « sonore » avec un haut-parleur (parlant ou musical). Les voix peuvent être accompagnées par des jingles musicaux pour réaliser des spots publicitaires, des messages ou de la musique de fond, etc.

Le **mode de reproduction peut être aléatoire** (« random ») grâce à la programmation via les boutons. Ce mode peut être combiné avec des **capteurs de proximité**, des barrières infrarouges ou des mécanismes automatiques **afin de lire le fichier « MP3 » lorsqu'une personne se trouve dans le voisinage immédiat d'un rayon**, ou pour contrôler la lecture au moyen de microcontrôleurs.

Même s'il a été conçu pour concurrencer les produits commerciaux, **notre lecteur MP3 a quelques fonctionnalités intéressantes difficiles à trouver ailleurs**. En outre, il est compact, puisque le circuit imprimé a été entièrement conçu pour accueillir des composants CMS qui permettent une réduction considérable de la taille (6 cm x 6 cm).

## Schéma électrique

Le schéma électrique paraît à première vue très simple, il comporte peu de composants et **la majeure partie des fonctionnalités sont réalisées par le programme contenu** dans le **microcontrôleur PIC qui est le cœur du lecteur MP3**. Le **décodage** des **fichiers MP3** est confié à un composant **spécifique à haute intégration VLSI**, capable d'exécuter diverses fonctions.

L'intégration à très grande échelle **VLSI** (**Very Large Scale Integration**) est une technologie de circuit intégré dont la densité d'intégration permet de supporter plus de 100 000 composants électroniques sur une même puce.

Le **dernier composant fondamental du lecteur** est un « slot » (support) pour **carte mémoire externe SD** ou **MMC**. Le reste des composants, en technologie discrète, sert à égaliser le signal audio et adapter l'interface de communication.

Pour bien comprendre le fonctionnement du lecteur MP3, nous avons besoin d'analyser en détail le schéma électrique. Celui-ci se compose du microcontrôleur **U1** (**PIC24HJ256GP206**) qui représente le **cœur du système**, du circuit **U2** (**VS1011E**) qui est un **décodeur** nécessaire pour **convertir correctement le flux de données** provenant du **fichier** en un **signal analogique** contenant des ondes sonores parfaitement audibles, et de la **carte mémoire qui permet de stocker les fichiers MP3**. Enfin le régulateur de tension (3,3 V) permet d'alimenter correctement les 3 composants de base.

L'apparente simplicité du schéma électrique a été rendue possible grâce à l'utilisation du **décodeur MP3 VS1011** à haute intégration « **VLSI** », capable de réaliser les fonctions suivantes :

- **sample rate converter** : convertisseur de fréquence d'échantillonnage ;
- **bass/treble control** : contrôle de tonalité graves/aigues ;
- **headphone driver** : sortie casque audio ;
- **playback time counter** ; compteur temps réel ;
- **stereo DAC** : convertisseur digital-analogique stéréo ;
- **tolerance of bit errors** : correcteur d'erreurs afin de fiabiliser la reproduction analogique du signal. En cas d'erreur dans le fichier, le circuit reconstitue partiellement la partie manquante ou altérée.

La **complexité** ne se situe plus au niveau matériel (hardware) **mais plutôt au niveau logiciel** (software) contenu dans les composants.

En étudiant le schéma électrique il est possible de comprendre le parcours du flux de données provenant du fichier MP3, mais en regardant le schéma de principe représenté sur la figure 1 cela est beaucoup plus évident.

Le **PIC** utilise **2 modules internes de communication série avec la mémoire** et le **décodeur U2**, il adopte pour tous les 2 le **protocole SPI** (**Serial Peripheral Interface**), qui est un protocole série synchrone avec un **MASTER** (MAITRE) et un **SLAVE** (ESCLAVE) ; le **PIC est défini comme MASTER**.

La **mémoire MMC** ou **SD** supporte un double protocole pour l'échange de données ; par défaut elle utilise un **protocole propriétaire de 4 bits en parallèle**, mais avec une initialisation correcte **il est possible d'utiliser l'interface série SPI** qui peut être utilisée librement sans licence.

Comme nous l'avons décrit précédemment, le décodeur MP3 dispose d'une interface de communication série avec un protocole SPI. Dans ce cas, la **communication est assistée par une ligne de demande** (requête) **DREQ** qui est

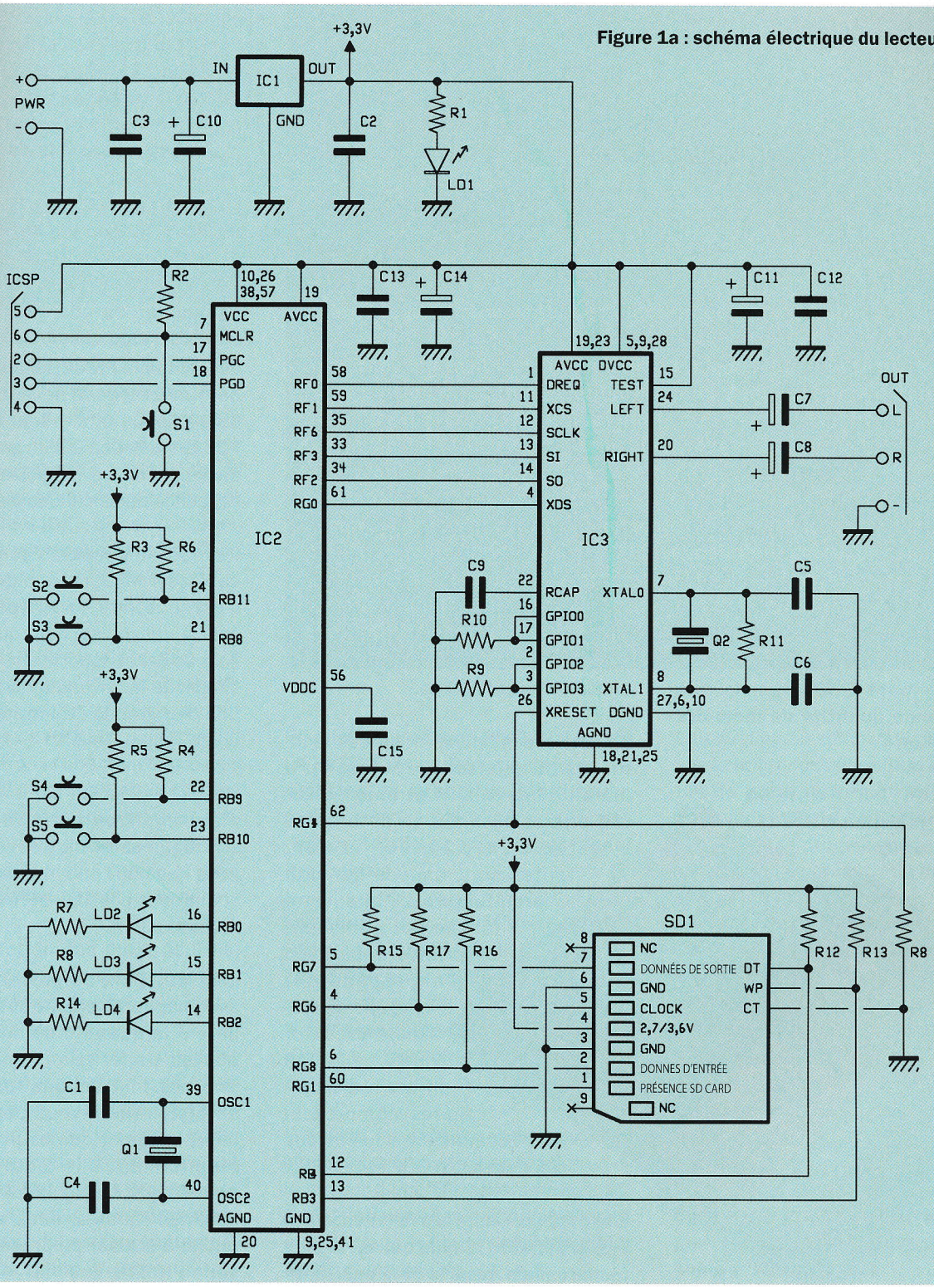
placée à un niveau logique **1** lorsque la file d'attente « **FIFO** » (**First In First Out**) est remplie.

Ce « **buffer** » (tampon) permet au **microcontrôleur de disposer du temps nécessaire pour recharger son propre « buffer » interne** afin qu'il puisse **lire la mémoire externe et/ou gérer des événements externes** tels que l'appui sur un bouton, ou encore de communiquer avec le décodeur lors d'une commande (ce qui est nécessaire pour échanger les informations et ajuster les paramètres demandés par l'utilisateur).

Les **3 LED** connectées aux 3 broches (pins) du **PIC24HJ256GP206** visualisent la **communication** avec le **décodeur** et les **commandes de l'utilisateur**.

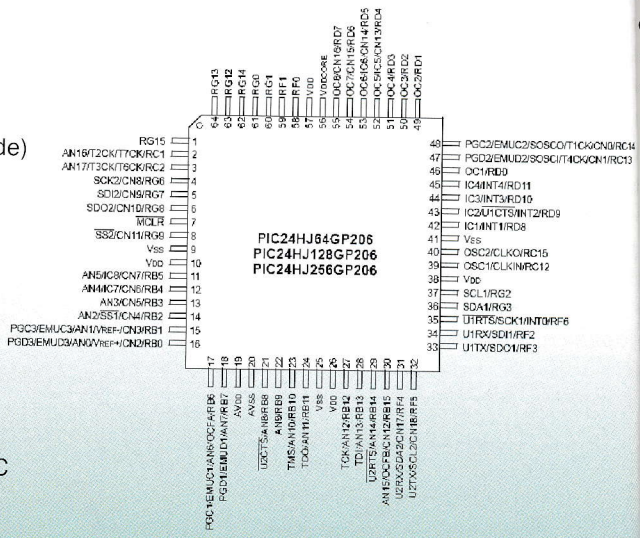
Le **clignotement irrégulier** de la **LED jaune** indique un **fonctionnement correct**, alors que la **LED rouge** et la **LED verte** sont utilisées comme signalisation pour l'utilisateur selon les indications du **Tableau 1**.

**Figure 1a : schéma électrique du lecteur MP3.**



## Le microcontrôleur PIC24HJ256GP206

- Architecture : 16-bit
- Vitesse du CPU : 40 MIPS (million d'instructions par seconde)
- Type de mémoire : FLASH
- Mémoire programmable : 256 Ko
- Mémoire RAM : 16384 octets
- Tension d'alimentation : 3V à 3,6V
- Pin I/O : 53
- Pin total : 64
- System Management Features : POR
- Oscillateur interne : 7,37 MHz et 512 kHz
- Fast Wake/Fast Control
- Périphériques de communication : 2 UART, 2 SPI, 2 BUS I²C
- Convertisseur A/D : 18 de 12 bits chacun
- Modules de capture/comparateur : 8/8 PWM 16 bits
- 16 Timer (9 de 16 bits et 4 de 32 bits)
- Port parallèle
- GPIO
- DMA : 8

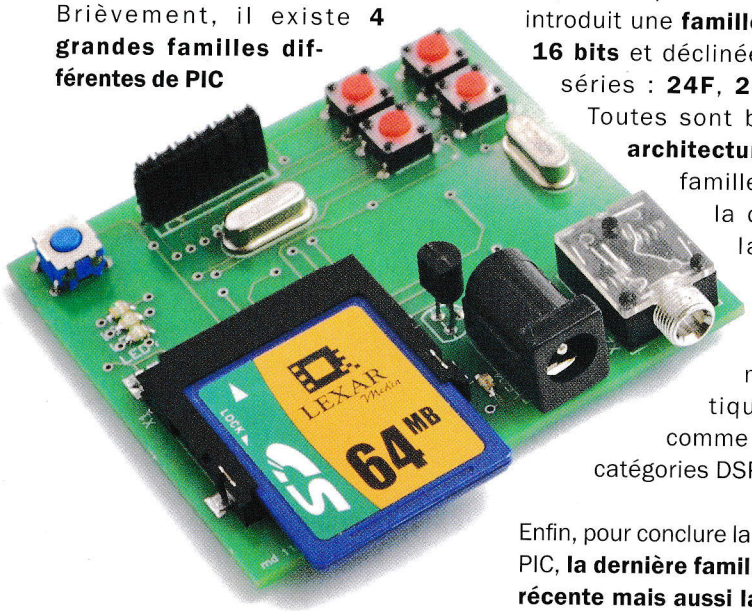



### Le CPU du lecteur

Dans le schéma synoptique vous pouvez remarquer la présence d'un microcontrôleur **PIC24HJ256GP206** de chez **Microchip** comme **unité centrale de traitement** du **lecteur MP3**.

En effet ce composant paraît le plus approprié à cet usage, car il dispose d'une **puissance de calcul élevée** et d'une mémoire « **FLASH** » **importante** pour le firmware (256 Ko), ainsi que d'une **grande quantité de mémoire RAM** (16 Ko).

Brièvement, il existe **4 grandes familles différentes de PIC** chez **Microchip** en production, la plus connue est la famille PIC micro déclinée en différentes catégories en fonction de leurs caractéristiques ; la plus répandue et la plus connue est la **famille 16F** dotée d'une architecture 8 bits très simple et économique. Elle a été complétée par la suite pour répondre aux limites de conception par la **famille 18F** qui préserve l'architecture 8 bits mais introduit de nouvelles fonctionnalités et une puissance de calcul accrue.




Ensuite avec l'amélioration des techniques de production, Microchip a introduit une **famille structurée sur 16 bits** et déclinée en différentes séries : **24F**, **24**, **30F** et **33F**. Toutes sont basées sur une **architecture 16 bits**, cette famille fait nettement la différence avec la précédente, et peut également exécuter des fonctions mathématiques complexes, comme dans le cas des catégories DSP30F et DSP33F.

Enfin, pour conclure la présentation des PIC, **la dernière famille qui est la plus récente mais aussi la plus puissante est la famille PIC32**, fonctionnant avec des « mots » (word) de **32 bits** (comme un classique processeur Pentium de PC). Elle est dotée d'un grand nombre de périphériques intégrés et dispose de capacités de calcul très élevées.

Pendant la phase de conception de ce lecteur MP3, notre bureau d'étude s'est concentré sur le type de processeur à utiliser, sachant qu'un processeur puissant apporte une plus grande simplicité au niveau du programme et une plus grande évolutivité, mais avec un coût plus élevé.

Un processeur un peu moins performant implique de développer des stratagèmes et de trouver des solutions « ad-hoc » pour trouver le meilleur compromis possible et réduire le coût de production.

Lors du choix du processeur, nous avons apporté une attention particulière à la quantité de données qu'il pouvait traiter et au temps d'exécution des instructions (par exemple en temps réel). Vous pouvez facilement imaginer que pour un système fonctionnant sur toute la gamme audio, il y a des limites structurelles infranchissables, **nous avons donc choisi d'utiliser un processeur de la série PIC24H**.

Le **PIC24HJ**
contrôleur
« **16 bits** » a
« **FLASH** » ré
de **16384 o**
Sa **fréquen**
est de **80 M**
est réglable

Il dispose d
de **16 bits**,
**ties** « **PWM**
de commun
tels que l'**U**
unités) et
intègre aus
et **DMA**. I
« **TQFP** » de
réservées a
réalisé en t
chez **Micro**
une **conso**

Le **PIC pre**
**assembleu**
celui de la
**utilisation**
remédier
propose u
« **compila**
l'environn
gratuit « **M**
lent **outil**
avec l'ass
temps de

Une atter
accordée
**haut nive**
**cette fam**
dès le dé
un **langa**
matérielle
miser les
gage et fo
sant à dis

Avec l'ut
vous pou
fonctionn
dre dans

La dern
pouvons
cerne l'a
intégran
l'anglais
qu'on po
**de signa**
**numériq**

**ELECTRO**

Le **PIC24HJ256GP206** est un microcontrôleur doté d'une architecture « **16 bits** » avec **256 Ko** de mémoire « **FLASH** » réservés au programme et de **16384 octets** de mémoire **RAM**. Sa **fréquence d'horloge maximale** est de **80 MHz**, l'oscillateur interne est réglable.

Il dispose de **18 convertisseurs A/D** de **16 bits**, de **9** « **Timers** », de **8 sorties** « **PWM** ». Il est doté des protocoles de communication les plus courants tels que l'**UART** (2 unités), le **SPI** (2 unités) et le **BUS I2C** (2 unités). Il intègre aussi les **périphériques CAN** et **DMA**. Il se présente en boîtier « **TQFP** » de **64 broches** (dont 53 sont réservées aux entrées/sorties). Il est réalisé en technologie « **nanoWatt** » de chez **Microchip** qui permet d'obtenir une **consommation réduite**.

Le **PIC prend en charge un langage assembleur beaucoup plus riche** que celui de la **famille 16F**. De ce fait, son **utilisation est assez complexe**. Pour remédier à ce problème **Microchip** propose une version « **Etudiante** » du « **compilateur C** » qui, combiné avec l'environnement de développement gratuit « **MPLAB IDE** », devient un excellent **outil de développement** interfacé avec l'assembleur afin de réduire le temps de conception.

Une attention particulière doit être accordée à l'emploi d'un **langage de haut niveau** que nous utilisons pour **cette famille** de **PIC**. Elle a été conçue dès le début pour une utilisation avec un **langage C** et des améliorations matérielles ont été réalisées pour optimiser les performances avec ce langage et fournir un outil simple et puissant à disposition de tout le monde.

Avec l'utilisation des outils fournis, vous pouvez vous concentrer sur les fonctionnalités réelles sans vous perdre dans la mise en œuvre.

La dernière observation que nous pouvons faire sur la **famille 24H** concerne l'analogie avec des composants intégrant la technologie « **DSP** » (de l'anglais « **Digital Signal Processor** », qu'on pourrait traduire par « **processeur de signal numérique** » ou « **traitement numérique de signal** »).




## Le MP3 ne connait pas la crise

**Après des années de croissance continue, le marché Européen des produits électroniques grand public semble connaître une phase de stagnation, la situation a été constatée lors de l'IFA de 2008 à Berlin, un des salons les plus importants de l'électronique grand public.**

**Le marché mondial des produits électroniques grand public est toujours en expansion, avec une augmentation de 9 points l'année dernière, et un chiffre d'affaires de 462 milliards d'euros. Le marché en plus forte expansion est l'Asie (+ 17%), principalement en raison du marché Chinois. Cependant la situation Européenne est inquiétante, la période des 6 premiers mois montre une légère hausse (voir une stagnation) de 0,5 % dans le secteur de la vidéo, grâce à la bonne performance des téléviseurs LCD (+ 5 %). Le marché des lecteurs MP3 se maintient grâce, principalement, aux ventes des systèmes de navigation (+ 1,5 %) dans le secteur automobile. Cependant, l'industrie de l'audio est en régression (- 4 %) ainsi que le marché des portables (- 11 %).**

En effet **pour chaque PIC24H il existe un PIC33F ayant des caractéristiques ainsi qu'un brochage identiques**, mais avec en plus des unités spéciales de calcul dédiées au « **Digital Signal Processing** ».

La lecture des caractéristiques du PIC utilisé dans ce projet peut laisser croire que nous exagérons au niveau des avantages qu'il procure. En fait, le but est simplement de prélever un octet de la mémoire et de l'envoyer au décodeur.

En réalité, les **choses ne sont pas aussi simples**, car si vous choisissez d'équiper le lecteur d'une mémoire amovible **SD CARD**, les **données** présentes doivent être dans un **format reconnaissable**, **lisible** et **modifiable** par un **PC**.

La situation se complique car dans ce cas **le microcontrôleur doit être capable de gérer le format MP3 stocké dans un système de fichiers de type standard**, ce qui nécessite un potentiel plus élevé par rapport à une gestion propriétaire.

En outre, étant donné que les PC utilisent plusieurs systèmes de fichiers (**FAT16**, **FAT32**, ...) la gestion de la mémoire **SD CARD** implique l'utilisation d'un **CPU puissant**.

Pendant les **phases de tests** du lecteur, nous avons constaté quelques problèmes dans certaines circonstances car **le PIC n'était pas optimisé** et devenait « lent » en **entraînant** des « **retards** » **dans la reproduction sonore**.

## Comment fonctionne un MP3 ?

Nous sommes tous habitués à utiliser le mot « **MP3** » pour identifier un **fichier audio de petite taille**, mais savons-nous réellement ce qu'est un fichier MP3 ? Pour avoir une idée, il est bon de parler de l'histoire de l'enregistrement audio, qui a connu une révolution avec l'introduction du premier disque compact audio ou CD Audio. Le **disque compact CD Audio** fut inventé conjointement par les firmes **Philips** et **Sony Corporation** en **1978**.

Le format des données, connu sous le nom de norme « **Red Book** » a été développé par **Philips**. En termes techniques, il s'agit d'une **piste stéréo encodée en PCM** avec une **résolution de 16 bits** (linéaire en amplitude, sans compression logarithmique des amplitudes hautes) avec une **fréquence d'échantillonnage** de **44,1 kHz**.

Donc le signal audio analogique est enregistré après avoir été échantillonné à une fréquence de 44100 Hz, les échantillons du signal sont capturés par un convertisseur analogique/digital (ADC) spécifique, avec une profondeur d'au moins 16 bits, l'opération étant effectuée à la fois sur les canaux gauche et droit.

Pour augmenter la qualité de l'enregistrement, la technique de **suréchantillonnage** a été mise au point. Elle consiste en un **échantillonnage 4 fois plus important** (**four times oversampling**) mais entraîne une **augmentation de l'espace requis pour la même quantité de temps écoulé**.

Le **suréchantillonnage consiste à échantillonner un signal à une fréquence très élevée**, beaucoup plus que ne l'exigerait le « **Théorème de Shannon** ». Cela permet de simplifier la conception du filtre « anticrènelage » (ou « antirepliement », ou encore « anti-aliasing ») et de diminuer le bruit présent dans la bande utile en augmentant le rapport signal sur bruit.

La technique de suréchantillonnage permet de minimiser les pertes lors des transitoires (notamment harmoniques) du signal audio analogique. Ces transitoires sont situés entre deux paliers numériques successifs créés par la fréquence d'échantillonnage du convertisseur A/N.

Les besoins des utilisateurs d'aujourd'hui et le **désir d'avoir à portée de main des dizaines d'albums**, a fait que la **technique du CD n'est plus adaptée au monde d'aujourd'hui**. Nous savons qu'un **CD ordinaire ne peut pas contenir plus de 80 minutes d'enregistrement audio**. En outre, la nécessité de transférer des fichiers musicaux sur Internet (streaming et P2P pour Peer To Peer) a imposé la recherche de formats audio capables de limiter la taille des fichiers à l'aide d'une compression importante, afin de réduire le temps de téléchargement.

Ceci a été réalisé, **mais au détriment de la qualité de la reproduction sonore**, avec des **méthodes de compression** qui sont essentiellement du « **MP3** » pour l'**audio** et du « **DivX** » (MPEG4) pour la **vidéo**.

La nécessité de distribuer l'audio sur Internet et la disponibilité des processeurs « DSP » qui peuvent décoder l'audio en temps réel, combinées avec l'étude des techniques de compression, ont permis la naissance du **format « MP3 »**.

Ouvrons une parenthèse sur les techniques de compression, il existe 2 types possibles : la « **compression sans perte d'information** » et la « **compression avec perte d'information** ». Le premier cas consiste en l'élimination de la redondance, un exemple typique est le fichier texte, où la récurrence des caractères est asymétrique. Par exemple en analysant un texte nous pouvons compter le nombre de caractères « a » et « z », nous constatons que le caractère « a » est présent un plus grand nombre de fois que le caractère « z », et ainsi de suite, nous pouvons alors établir une liste de tous les caractères utilisés.

Sur la base de cette information, il est possible d'utiliser des codes d'identification pour chaque caractère dont la longueur est variable et non pas fixe comme dans le cas du codage ASCII. L'affectation doit garantir des codes courts (quelques bits) pour les caractères les plus nombreux et des codes longs pour les caractères les moins nombreux. De cette façon, nous obtenons un fichier résultant compressé qui est plus petit.

Cette technique est valable dans les deux sens, car elle permet le passage « inverse » en appliquant un algorithme de décodage. De ce fait nous pouvons toujours revenir au fichier original avec les codes ASCII classiques.

Pour éviter ces problèmes, nous avons dû étudier et comprendre les implémentations du système de fichiers (file-system) et les exigences requises afin de pouvoir **gérer une partition de type FAT**.

| Répartition de l'espace en fonction de la FAT | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0000 | 0006 | 0000 | 0000 | 0008 | 0000 | FFFF | 0000 |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

**Exemple de répartition des données en fonction du type de FAT. Le fichier est divisé en secteurs, dont le dernier contient le code qui indique la fin du fichier.**

### Le système de fichiers (file-system)

Le **système de fichiers** est un **mécanisme destiné à conserver de manière ordonnée les informations contenues dans un espace de stockage** dudit **fichier**. Le **système de fichiers** ainsi que **les fichiers sont deux abstractions qui vous permettent**

**Cette technique appliquée à un fichier audio n'apporte pas de grands avantages, car les échantillons d'octets convertis au cours de la phase d'acquisition n'ont pas véritablement de valeurs communes**, mais plutôt des valeurs disparates.

Ainsi, l'unique façon de **réduire la taille des fichiers est de mémoriser moins d'informations.** Cependant, nous savons que si nous mémorisons moins d'informations cela signifie une perte de qualité audio. Or nous sommes dans le champ de la **Haute-Fidélité**, et nous ne voulons pas une reproduction qui présente des **distorsions** importantes ou du **bruit**.

L'idée est venue en étudiant le « **comportement** » de notre **oreille.** Naturellement, cette dernière **n'est pas en mesure de percevoir distinctement les harmoniques pairs qui sont reproduits dans des situations particulières**. En acoustique, un **harmonique est une composante d'un son périodique** (et plus généralement d'une onde), qui possède une **fréquence multiple** de la **fréquence fondamentale**.

Par exemple, si on appelle « ***f*0** » la **fréquence fondamentale**, les **harmoniques** auront des fréquences égales à : **2*f*0**, **3*f*0**, **4*f*0**, **5*f*0**, etc. En audio, **un harmonique est un composant à part entière d'un son musical**. En prenant comme note fondamentale le « **LA** » (440 Hz) du piano, les **harmoniques sont toutes les notes ayant pour fréquence un multiple de 440**. Les harmoniques d'une note sont donc forcément **plus aigus que cette note** et s'appellent harmoniques supérieurs.

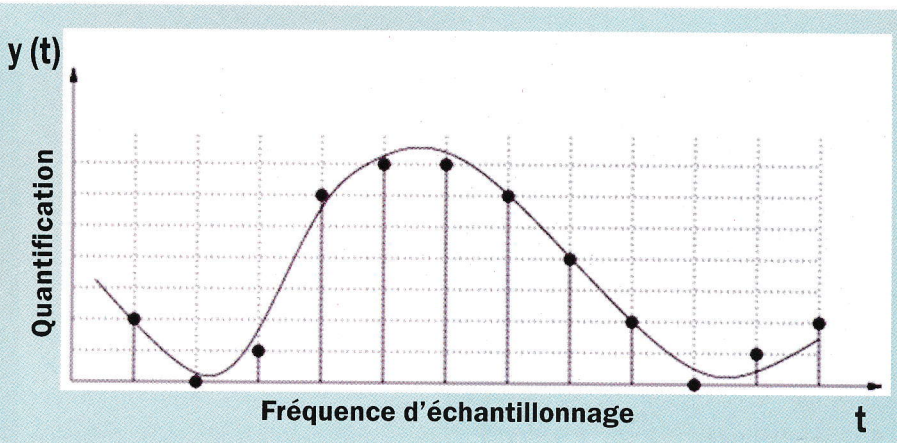



**Figure A : évolution temporelle d'un signal continu échantillonné à une fréquence « *f* » donnée. Chaque échantillon du signal est quantifié par un convertisseur ADC avec une grande précision.**

En pratique l'**oreille humaine entend la note dont l'harmonique a la plus grande intensité et c'est souvent la plus grave**. Grossièrement, nous pouvons dire que l'oreille ne distingue pas 2 sons successifs rapides.

L'**encodage** du « **MP3** » est basé sur cette « **incapacité** » de notre système auditif à interpréter correctement le son, si nous ne pouvons pas percevoir ces informations, pourquoi perdre de l'espace pour les stocker ? En réalité, le problème est beaucoup plus complexe car dans le cas d'une reproduction Hi-Fi de nombreux facteurs entrent en jeu, dont l'un est la capacité différente de chaque personne à percevoir un son.

Des études ont abouti à la définition de la norme de compression des fichiers audio au format MP3 afin de se conformer à une qualité qui n'est pas toujours appréciable, mais dont la taille des fichiers est réellement inférieure. Essentiellement un fichier MP3 est une séquence de trames stockées avec un codage défini par des algorithmes de compression qui traitent le signal initial, et éliminent les composantes imperceptibles. La compression applique des techniques supplémentaires dans les « **graves** » pour **stocker les deux canaux gauche et droite sous la forme d'un seul canal**, car l'**oreille ne perçoit pas l'effet stéréo dans les basses fréquences**. Le **fichier MP3** obtenu se présente sous la **forme de plusieurs blocs d'octets regroupés en trames d'une durée chacune de quelques millièmes de secondes**.

Le processus de décodage doit reconnaître et lire le fichier de manière séquentielle, ensuite il applique le décodage approprié pour chaque trame. Sans entrer dans les détails, l'information est identifiée par une séquence particulière de bits présents au début de chaque trame. Une fois décodé, le flux d'octets peut être transmis au convertisseur DAC qui prendra soin de reproduire correctement le signal audio afin d'être amplifié et reproduit par un haut-parleur, avec une qualité acceptable.

**de voir en un seul espace continu un nombre d'octets contenant un document**. Le **système de fichiers est divisé en répertoires**, **dossiers**, **sous-dossiers** et **fichiers**, des concepts que nous retrouvons dans la vie quotidienne. Grâce à ce mécanisme, les fichiers peuvent être stockés dans des dispositifs tels qu'une **SD CARD**, une **clé USB** ou un **disque dur**.

Différentes méthodes permettent d'associer un nom de fichier à son contenu. Dans le cas du **système de fichiers FAT**, ancien système de fichiers de **MS-DOS** et de **Windows** encore largement utilisé sur les supports amovibles comme les SD CARD et clés USB, **chaque répertoire contient une table associant les noms des fichiers à leur taille** et un **index pointant vers la table d'allocation de fichiers**, une **zone réservée** du disque indiquant **pour chaque bloc de données l'index du bloc suivant** d'un même fichier.

**MS-DOS** (et compatibles) et les **premières versions de Windows 95** utilisaient les systèmes de fichiers **FAT16** et **FAT12** (pour les supports de moins de 16 Mo).

## Plan de montage

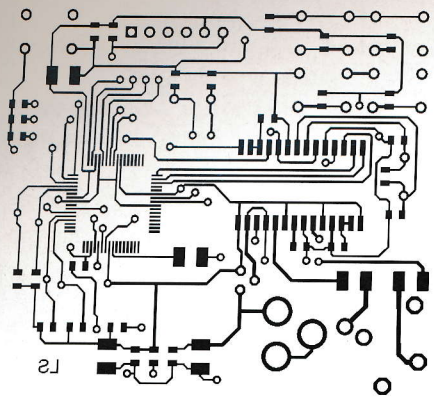


**Figure 1b : dessin du circuit imprimé à l'échelle 1 : 1 côté soudures.**

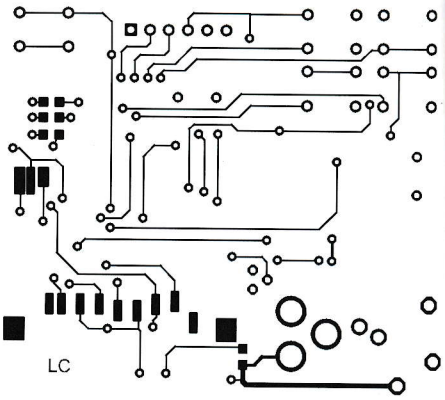



**Figure 1c : dessin du circuit imprimé à l'échelle 1 : 1 côté composants.**

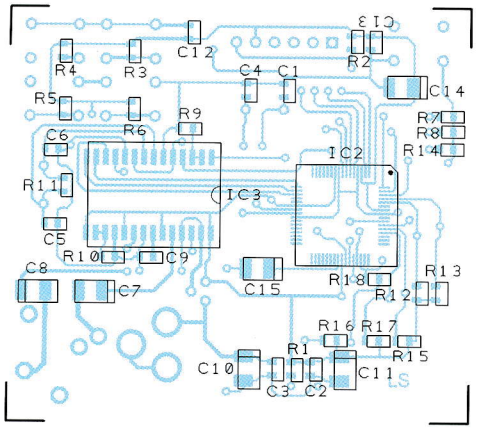



**Figure 1d : implantation des composants CMS côté soudures du circuit imprimé.**

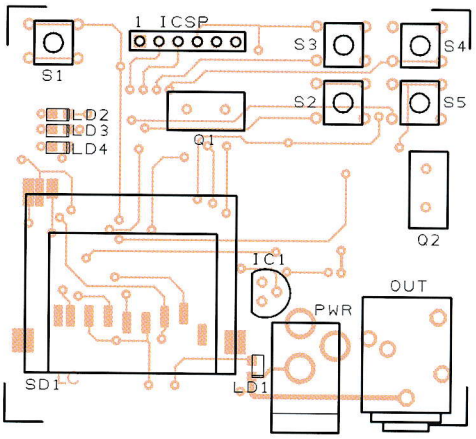



**Figure 1e : implantation des composants traditionnels (poussoirs, connecteurs .....) côté composants du circuit imprimé.**

À partir de **Windows 95 OSR2**, le choix entre les systèmes de fichiers a commencé à s'élargir. **FAT16** et **FAT32** pouvaient tous les deux être utilisés, et à partir d'une certaine taille de partition, le choix du système **FAT32** était alors plus judicieux. Sur les premières versions de **Windows NT** (NT3.x et NT4), il y avait le choix entre le système **FAT16** et **NTFS**. Ce système d'exploitation ne supporte pas le FAT32. Généralement, le **système NTFS est conseillé**, car il procure une **sécurité plus grande** ainsi que des **performances accrues** par rapport au **FAT**. Contrairement aux précédentes versions de Windows NT, Windows NT5 c'est-à-dire **Windows 2000** et les versions suivantes jusqu'à **Windows 8.1.1** acceptent des partitions de type **FAT16**, **FAT32** et **NTFS**.

Ainsi, le système de fichiers le plus récent (NTFS 5.1 et +) est conseillé puisqu'il offre de plus nombreuses fonctionnalités que les systèmes FAT.

Vous remarquez donc qu'il existe plusieurs systèmes de gestion des fichiers en fonction du système d'exploitation qui assurent plusieurs fonctions :

- **manipulation des fichiers** : des opérations sont définies pour permettre la manipulation des fichiers par les programmes d'application, à savoir : créer/détruire des fichiers, insérer, supprimer et modifier une partie d'un fichier ;
- **allocation de la place dans la mémoire** : les fichiers étant de taille différente, celle-ci pouvant varier, alors le système alloue à chaque fichier un nombre variable d'unités de mémoire (clusters) de taille fixe ;
- **localisation des fichiers** : il est nécessaire de pouvoir identifier et retrouver les données, pour cela chaque fichier possède un ensemble d'informations descriptives (nom, adresse...) regroupées dans un « inode » (nœud d'index) ;
- **sécurité et contrôle des fichiers** : le système permet le partage des fichiers par différents programmes d'applications tout en assurant la sécurité et la confidentialité des données. En effet, un nom et une clé de protection sont associés à chaque fichier afin de le protéger contre tout accès non autorisé ou

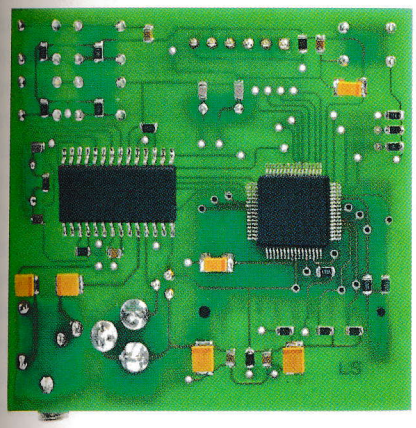


Figure 1f : photo de l'un de nos prototypes, vous pouvez voir les composants CMS soudés du côté soudures du circuit imprimé.

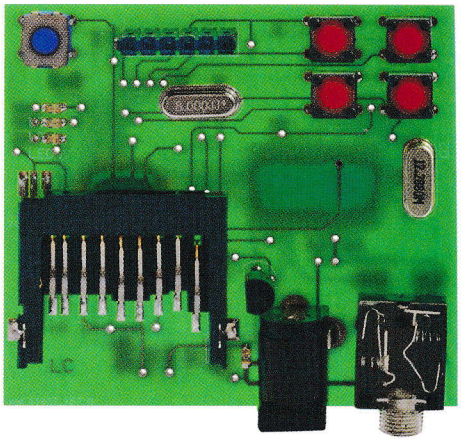


Figure 1g : photo de l'un de nos prototypes, vous pouvez voir les composants traditionnels placés sur le côté composants du circuit imprimé.

## Liste des composants du lecteur MP3

| | | |
|---|---|---|
| R1 | 150 Ω | Boitier CMS 0805 |
| R2 | 10 kΩ | Boitier CMS 0805 |
| R3 | 100 kΩ | Boitier CMS 0805 |
| R4 | 100 kΩ | Boitier CMS 0805 |
| R5 | 100 kΩ | Boitier CMS 0805 |
| R6 | 100 kΩ | Boitier CMS 0805 |
| R7 | 820 Ω | Boitier CMS 0805 |
| R8 | 1,3 kΩ | Boitier CMS 0805 |
| R9 | 100 kΩ | Boitier CMS 0805 |
| R10 | 100 kΩ | Boitier CMS 0805 |
| R11 | 1 MΩ | Boitier CMS 0805 |
| R12 | 100 kΩ | Boitier CMS 0805 |
| R13 | 100 kΩ | Boitier CMS 0805 |
| R14 | 560 Ω | Boitier CMS 0805 |
| R15 | 47 k Ω | Boitier CMS 0805 |
| R16 | 47 k Ω | Boitier CMS 0805 |
| R17 | 47 k Ω | Boitier CMS 0805 |
| R18 | 100 kΩ | Boitier CMS 0805 |
| C1 | 22 pF céramique | Boitier CMS 0805 |
| C2 | 100 nF multicouche | Boitier CMS 0805 |
| C3 | 100 nF multicouche | Boitier CMS 0805 |
| C4 | 22 pF céramique | Boitier CMS 0805 |
| C5 | 33 pF céramique | Boitier CMS 0805 |
| C6 | 33 pF céramique | Boitier CMS 0805 |
| C7 | 100 µF/16 V tantale | Boitier CMS type B |
| C8 | 100 µF/16 V tantale | Boitier CMS type B |
| C9 | 100 nF multicouche | Boitier CMS 0805 |
| C10 | 100 µF/16 V tantale | Boitier CMS type B |
| C11 | 100 µF/16 V tantale | Boitier CMS type B |
| C12 | 100 nF multicouche | Boitier CMS 0805 |
| C13 | 10 nF 63 V polyester | Boitier CMS 0805 |
| C14 | 10 µF/16 V tantale | Boitier CMS type B |
| C15 | 10 µF/16 V tantale | Boitier CMS type B |
| LD1 | LED verte | Boitier CMS 0805 |
| LD2 | LED verte | Boitier CMS 0805 |
| LD3 | LED jaune | Boitier CMS 0805 |
| LD4 | LED rouge | Boitier CMS 0805 |
| IC1 | LP2950CZ-3.3 | TO92 |
| IC2 | PIC24HJ256GP206 | TQFP |
| IC3 | VS1011E | SOIC28 10 x 17.5 x 2.5 mm |
| Q1 | 8 MHz | HC49 |
| Q2 | 12.288 MHz | HC49 |
| S1 | Microswitch | |
| S2 | Microswitch | |
| S3 | Microswitch | |
| S4 | Microswitch | |
| S5 | Microswitch | |

Divers
Fiche alimentation pour circuit imprimé
Connecteur femelle jack stéréo 3,5 mm pour circuit imprimé
Connecteur SD CARD pour circuit imprimé
Barette femelle 6 pôles

mal intentionné lors du partage des fichiers. Le système doit aussi garantir la conservation des fichiers en cas de panne du matériel ou du logiciel.

La SD CARD utilisée dans notre projet est identique à une clé USB ou au disque dur de notre ordinateur, qui contiennent de longues séquences d'octets accessibles individuellement grâce à des adresses ordonnées par ordre croissant. Vous pouvez facilement imaginer qu'il serait difficile de l'utiliser sans système de fichiers.

Cependant, d'un autre côté, vous percevez la difficulté à mettre en œuvre la compatibilité avec les différents systèmes de fichiers vus précédemment pour reconnaître correctement et cartographier les séquences d'octets dans la mémoire.

Cela donne la possibilité d'accéder non pas à l'adresse mémoire, mais par les noms et positions.

Voici un exemple pour vous aider à comprendre la mise en œuvre et le fonctionnement du système de fichiers. Si dans un périphérique de stockage **à un moment donné sont créés 2 documents au format texte nommés** « **A.txt** » et « **B.txt** », il y a de fortes chances pour qu'ils soient alloués à des **adresses consécutives** dans la mémoire.

Pour simplifier, imaginons que « **A.txt** » précède « **B.txt** ».

Si à une **date ultérieure le fichier** « **A.txt** » est mis à jour par l'**ajout d'un élément** supplémentaire dans le texte, sa **taille finale augmente** et alors il est nécessaire de prévoir un **mécanisme qui permette de stocker les nouvelles informations dans le fichier** « **A.txt** » **sans écraser les fichiers qui sont contigus** (par exemple « B.txt »). Le **système de fichiers effectue également cette tâche, il suit les modifications et assure l'intégrité des fichiers** « **A.txt** » et « **B.txt** ».

Pour éviter qu'un **fichier envahisse l'espace consécutif**, il est **fragmenté** c'est à dire que la **partie qui n'est pas dans l'espace d'origine est située dans une autre zone du support de stockage** (le fichier est placé dans un endroit de la mémoire où il y a un espace libre).

Dans le cas où nous devons stocker de nombreuses données (plusieurs mégaoctets) ), ces dernières sont disponibles uniquement sous la forme de centaines ou de **milliers de fragments** d'une **taille** de **4096 octets chacun** (c'est la taille par défaut de la plus petite **unité d'allocation** ou « **cluster** »).

Il devient évident que l'**opération de fragmentation devient très lourde pour un disque dur**, car celui-ci a un temps d'accès aux données supérieur à une mémoire de type SD ou une clé USB à cause de la mécanique présente dans le disque dur. D'ailleurs **de nos jours les disques durs sont de plus en plus de type SSD** (mémoire FLASH) **sans mécanique**.

Parmi les effets produits par la **fragmentation**, un **bruit mécanique bien connu est généré par le disque dur** de l'ordinateur lorsque nous accédons à un fichier très volumineux, ceci est dû au mouvement continu des têtes qui recherchent sur les plateaux du disque dur toutes les parties qui composent le fichier.

Maintenant, le **concept de défragmentation d'un fichier devient plus clair**, il est **stocké de manière traditionnelle en le plaçant dans une zone continue**. S'il **augmente de taille**, il est possible de **déplacer des données proches qui appartiennent à un autre fichier plus petit vers d'autres zones** de la mémoire, par contre **si sa taille diminue l'espace reste inutilisé**. En résumé la **défragmentation réorganise physiquement le contenu du disque pour mettre chaque morceau de fichier ensemble et dans l'ordre**, dans le but d'**augmenter la vitesse de lecture**. Elle essaye également de **créer une grande région d'espace libre pour retarder la fragmentation**.

En utilisant le système de fichiers, chaque fichier peut être réparti là où il y a de l'espace disponible. Le fichier est identifié par le système de fichiers qui collecte toutes les données avec les positions où elles se trouvent et l'ordre correct de lecture. Cela offre l'avantage de placer les fichiers de manière dynamique afin d'optimiser l'espace, les données ne se trouvent pas toujours dans la même position car elles sont déplacées en fonction des besoins.

**Cette gestion explique aussi pourquoi un fichier n'est pas identifié par un emplacement dans la mémoire, mais par un nom dans le système de fichiers**, car les données ne sont pas toujours dans la même position, de sorte que nous ne pouvons pas les définir avec une adresse. Quand nous voulons ouvrir les fichiers, il nous suffit d'écrire le nom du fichier dans la barre de recherche et le système indique dans quel répertoire il se trouve.

La gestion des fichiers a été pensée pour des utilisateurs communs, car il est plus facile de se rappeler d'un nom que d'une « adresse mémoire ». Aujourd'hui, il existe de nombreux types de systèmes de fichiers, plus ou moins optimisés pour la facilité d'utilisation Certains utilisent les caractéristiques de latence du disque, d'autres la réécriture de la mémoire flash. Sans minimiser l'histoire des systèmes de fichiers nous allons considérer le système le plus adapté à notre projet.

Le **système de fichiers** utilisé par notre **lecteur MP3 est l'un des premiers mécanismes standard de subdivision** (fractionnement) des fichiers. Il n'est pas le meilleur, mais pour des raisons diverses il a affirmé sa prédominance et aujourd'hui, bien qu'il soit dépassé techniquement, il est encore très utilisé en raison de sa simplicité de mise en œuvre. Il s'agit de la **FAT** (**File Allocation Table** ou **Table d'allocation des fichiers**) qui consiste en une **table centrale spécialisée dans le stockage des informations relatives aux fichiers** : la longueur, la position, le nom, les attributs.

Le contenu réel du fichier est réparti sur l'espace disponible du disque dur.

Ce système de fichiers est devenu un standard de l'industrie. Il est simple et robuste, offre de bonnes performances dans des environnements simples, mais n'atteint pas la fiabilité et l'évolutivité de certains systèmes de fichiers modernes. Il est cependant pris en charge par la plupart des systèmes d'exploitation actuels ainsi que de très nombreux appareils mobiles ou systèmes embarqués. Il est donc bien adapté aux échanges de fichiers entre les appareils de tous types et ce depuis les années 1980.

Le **système FAT** a été conçu en **1977** par **Bill Gates** et **Marc McDonald** pour le **Microsoft Disk BASIC**, puis réutilisé dans **QDOS** dont c'est le seul élément qui ne soit pas inspiré par **CP/M** (QDOS est l'ancêtre de MS-DOS, écrit par **Tim Paterson**). Il fut utilisé à partir de **MS-DOS** jusqu'aux **versions 9x de Windows** en évoluant avec une capacité croissante des supports en 3 variantes **FAT12**, **FAT16** puis **FAT32**.

Les systèmes **Windows suivants**, dérivés de la branche NT, utilisent le **système NTFS**, mais reconnaissent toujours les systèmes FAT. Le système FAT reste toujours employé sur les clefs

USB, les cartes SD ainsi que dans de très nombreux équipements mobiles qui utilisent ces supports.

La reconstitution des fichiers s'effectue selon une séquence définie dans la table centrale contenant l'adresse du premier bloc de données, typiquement un bloc de quelques kilooctets (ko) appelé secteur. Le bloc contient également l'octet représentant le contenu du fichier et, si le fichier est plus grand que le secteur, la dernière information représente l'adresse du secteur suivant.

Plusieurs secteurs sont regroupés en « clusters », ceux-ci servent d'unité d'allocation aux fichiers. Chaque « cluster » stocke donc les données d'un fichier.

Pour un fichier de 9 000 octets par exemple, sur un disque utilisant des clusters de 8 192 octets (soit 16 secteurs de 512 octets), 2 clusters sont utilisés, le dernier n'utilisant que 808 octets (9 000 - 8 192).

Comme nous l'avons dit précédemment, dans l'histoire du PC, différentes versions de « **FAT** » ont été conçues pour gérer des unités de disques de plus en plus grandes. La première était la **FAT12** utilisée par la disquette et caractérisée par une capacité d'adressage de **32 Mo au maximum**.

Ensuite l'évolution exponentielle de la capacité des disques durs a poussé à développer la **FAT16** pour une capacité maximale de **2 Go** et enfin la **FAT32** pour des capacités maximales de **2 To**.

La **FAT classique pouvait seulement enregistrer des noms de fichiers d'une longueur maximale de 8 caractères plus 3 pour l'extension** (pour ceux d'entre vous qui ont eu l'occasion d'utiliser le MS-DOS), ce qui permettait à MS-DOS d'identifier le type de fichier. Par la suite la possibilité d'utiliser **des noms plus « longs »** a été introduite avec l'avènement de **Windows 95**, la structure de la **FAT a été mise à jour** à l'aide d'un correctif. Le nouveau système de fichiers était nommé « **VFAT** » et pouvait supporter des **noms de fichiers d'une longueur de 255 caractères**.

Comme le système « **VFAT** » **est toujours la propriété de Microsoft**, pour utiliser notre lecteur sans restriction, nous avons dû utiliser le **système de FAT classique** avec des noms de fichiers de « **8 + 3** » **caractères**, bien que tout système moderne de fichiers utilise des noms longs.

Dans un système de type FAT, les fichiers sont fragmentés et doivent être reconstitués en un seul tenant, on dit aussi que l'on défragmente les fichiers. C'est pour cette raison que le microcontrôleur doit effectuer une lourde tâche, car lorsqu'il doit lire un fichier il se peut que celui-ci se trouve physiquement en plusieurs endroits. Il doit donc chercher les différentes parties qui composent le fichier. Environ 60 % du programme écrit pour ce projet est réservé à la gestion du système de fichiers.

## Fonctionnement du lecteur

A ce stade le fonctionnement de l'appareil est assez simple à comprendre. En ce qui concerne le schéma synoptique de la figure 1, le **PIC initialise la mémoire externe** en conformité avec le système de fichiers **FAT16** ou **FAT32**, et **recherche les fichiers disponibles** sur la carte SD. A la **fin de la recherche**, il envoie un **signal de remise à zéro au décodeur** afin de le préparer à la lecture des trames d'octets via le port série.

Pour **lire un fichier audio**, le système de fichiers exécute une analyse et **pour chaque morceau de musique il réassemble les données fragmentées afin de les mettre dans le bon ordre et ensuite il les transmet au décodeur**. Après la phase de configuration initiale, le PIC se place en mode d'attente des commandes provenant de l'utilisateur et démarre la lecture des octets dans la mémoire pour les envoyer au décodeur.

Pendant **les phases où le tampon de réception du décodeur est saturé**, le **PIC contrôle l'état des boutons** et **détecte** d'éventuelles **commandes de l'utilisateur**. Le contrôle des boutons est réalisé à une fréquence très élevée ce qui garantit toujours la reconnaissance d'une touche pressée.

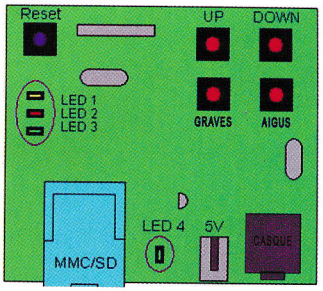



**Figure 3 : disposition des composants sur le circuit imprimé vu de dessus.**

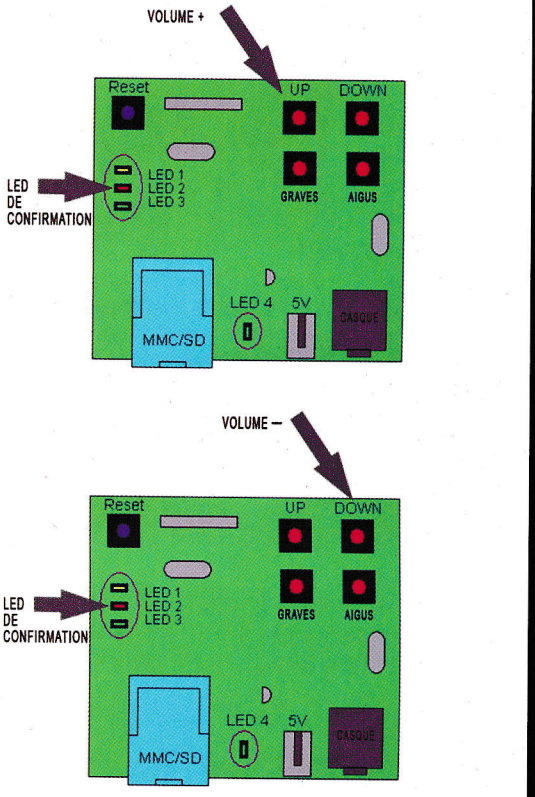



**Figure 4 : réglage du volume +/- : appuyez sur le bouton correspondant à votre choix UP ou DOWN pour augmenter ou diminuer le volume. Une simple pression fait clignoter rapidement 2 fois la LED rouge. Une pression longue permet de modifier continuellement le volume (dans ce cas la LED rouge clignote jusqu'au relâchement). Dans le cas où la limite est atteinte, la LED verte clignote.**

## Réalisation pratique

Le circuit imprimé du lecteur MP3, disponible sur notre site **www.electroniquemagazine.com**, est entièrement réalisé en technologie CMS pour être de petites dimensions. Il est vrai que pour un amateur souder des composants CMS n'est pas facile,

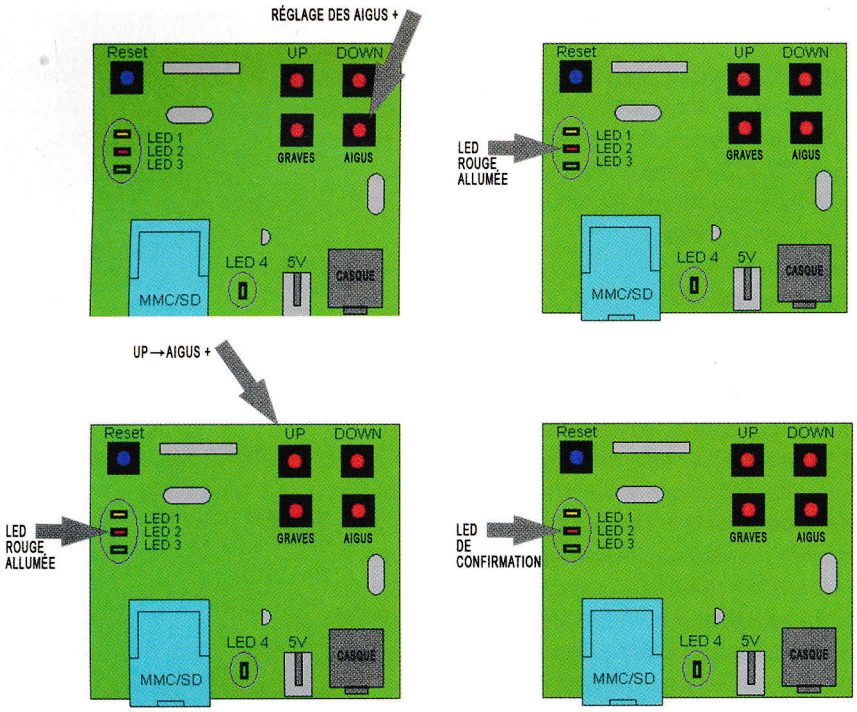



**Figure 5 : Réglage de tonalité des graves/aigus. Il s'effectue en 4 étapes : appuyez sur l'une des 2 touches GRAVES ou AIGUS (la LED rouge reste allumée), ensuite appuyez brièvement sur l'une des 2 touches UP (+) et DOWN (-) pour augmenter ou diminuer le gain. La LED rouge clignote pour indiquer le changement. Dans le cas où la limite est atteinte, la LED verte clignote.**

cependant en utilisant un fer à souder à pointe fine et une soudure fine le travail est réalisable.

Eventuellement il faudra utiliser une tresse à dessouder pour traiter l'excédent d'étain, il faut cependant beaucoup de patience. A certains égards, les composants CMS sont plus faciles à souder car il suffit de les placer avec un peu de colle sans avoir à faire des trous comme dans le cas de composants traditionnels et couper les longueurs des pattes.

Malheureusement, **le point sensible est la difficulté lors de la soudure des broches du PIC et du décodeur**, il faut éviter autant que possible de faire des courts-circuits au niveau des broches.

Les règles d'assemblage restent fondamentalement les mêmes que pour un circuit classique. Il faut commencer par souder les composants les plus petits, c'est-à-dire les résistances et les condensateurs non polarisés.

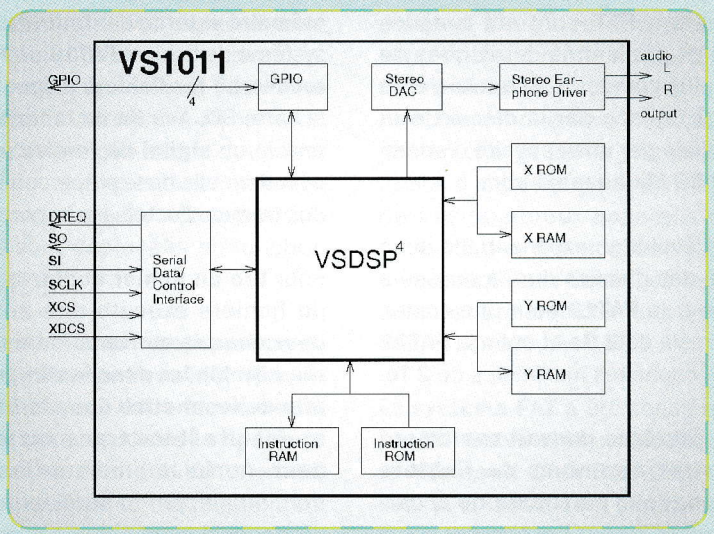
## Le décodeur MP3

Dans notre lecteur, le **décodage des fichiers audio MP3 a été confié au décodeur VS1011** fabriqué par la société **VLSI**. Il s'agit d'un décodeur audio complet contenu dans une seule puce (single-chip), il est capable de réaliser les 4 fonctions suivantes : « **buffering** », **décodage**, **conversion digitale/analogique** et **amplification stéréo**.

De plus ce circuit intégré est caractérisé par une **faible consommation**, une facilité de gestion, une évolutivité et un faible coût. Il est donc parfaitement adapté pour des applications où la portabilité, la consommation, les dimensions et les fonctionnalités sont cruciales.

Le principe de fonctionnement est très simple, après une initialisation le circuit reste en veille et scrute l'entrée de l'interface série sur laquelle arrivent les données (octets) en mode SLAVE (ESCLAVE). La première trame d'octets reçue active les routines de

**Figure A : schéma synoptique du décodeur VS1011E.**




**Caractéristiques techniques du VS1011**

| Paramètres | Valeurs |
|---|---|
| Codecs supportés | 1.0 & 2.0 Couches Audio III (CBR, VBR, ABR); couches I et II en option; WAV (PCM + IMA ADPCM) |
| Débit binaire maximal | 320 kbit/s pour un fichier MP3 avec un oscillateur à 12 Mhz. |
| Streaming pris en charge | MP1/2/3 et WAV |
| Options de tonalité | Graves et aigus séparés |
| Charge en sortie | Casque stéréo avec une impédance de 30 Ω |
| Convertisseur DAC stéréo | Convertisseur Sigma Delta 18 bits |
| Possibilité de codage de type DSP | 5,5 Kb |
| Tension d'alimentation | De 2,5 V à 3,6 V (analogique et numérique séparées) |

Ensuite passez aux circuits intégrés et aux condensateurs polarisés (attention au repère) puis passez aux connecteurs jack, d'alimentation et le support de carte. Même les composants CMS ont une polarité, n'oubliez pas que les condensateurs au tantale ont une bande de couleur pour identifier le « + » et les circuits intégrés ont un point ou une encoche sur leur boitier indiquant la broche 1.

Une **attention particulière doit être accordée aux LED car leurs repères ne sont pas très visibles**. Pour identifier la cathode et l'anode vous devez effectuer un test pratique en prenant une pile 9 V (ou une source d'alimentation) et en alimentant chaque LED à travers une résistance de 1 kΩ en série.

Lorsque la LED s'allume, le « + » de la pile indique la broche correspondant à l'anode de la LED. En ce qui concerne l'utilisation du lecteur, le circuit doit être alimenté avec une tension continue ne dépassant pas 6 VDC.

**Tableau 1 : les modes de fonctionnement du lecteur.**

| Touches | Description du fonctionnement |
|---|---|
| UPDOWN | Lecture aléatoire de tous les fichiers présents dans le dossier racine, ne vérifie pas si un fichier a déjà été lu. |
| DOWN | Le lecteur se met en pause jusqu'à une pression sur « UP », ensuite il lit le fichier et repasse en pause. |
| GRAVES | Lecture par ordre alphabétique ; lit tous les fichiers dans le dossier racine classés par nom de A à Z. |
| AIGUS | Mode prédéfinit : le lecteur reproduit tous les fichiers dans l'ordre de mémorisation du système de fichiers. |

Ceci à cause du régulateur qui fournit le 3,3 VDC nécessaire au microcontrôleur, au décodeur et à la carte SD. Le régulateur ne dispose pas de radiateur et donc il ne peut pas dissiper beaucoup de puissance. Etant donné que la dissipation est directement proportionnelle à la différence de tension entre l'entrée et la sortie du régulateur, afin de maintenir la température dans des limites acceptables, la source d'alimentation utilisée doit fournir 5 VDC à 6 VDC continus au maximum pour un courant de l'ordre de 100 mA.

## Comment utiliser le lecteur MP3 ?

Sur le circuit imprimé se trouvent 4 boutons qui peuvent être utilisés pour d'autres fonctions. Pour les lecteurs qui maitrisent le langage C, le projet complet avec le **code source en C** sera disponible à la vente dans la rubrique « **TOP PROJETS** » sur notre site **www.electroniquemagazine.com**.

Ils pourront modifier les contrôles en fonction de leurs besoins. Les boutons « UP » et « DOWN » sont utilisés pour

conversion et décodage selon l'algorithme standard de la norme MPEG 1.0 et 2.0 des couches I, II et III.

Les données (octets) traitées sont transmises aux contrôles numériques de tonalité et de volume, section du circuit dans laquelle le signal audio est amplifié, atténué, modulé selon les tonalités graves-médium-aigus de manière numérique.

Enfin la séquence d'octets ainsi « modelée » passe par un **convertisseur D/A** de **18 bits** de type **Sigma Delta** qui génère le signal de sortie analogique. Celui-ci, convenablement amplifié par un étage amplificateur interne, est directement utilisable avec des casques de moyenne et haute impédance.

Enfin le circuit VS1011 permet de réaliser un traitement DSP du signal pour ajouter des effets spéciaux. Cependant cette opération nécessite des connaissances sur les techniques de traitement numérique du signal.

**Brochage du décodeur VS1011.**

| Nom Pin | Pin | Type Pin | Fonction |
|---|---|---|---|
| DREQ | 1 | DO | données requises, entrée bus |
| GPIO2[2] / DCLK[1] | 2 | DIO | entrée série données d'horloge (clock) |
| GPIO3[2] / SDATA[1] | 3 | DI | entrée série données |
| XDCS[4] / CSYNC[1] | 4 | DI | signal de synchronisation d'octet (byte) |
| DVDD1 | 5 | PWR | + de l'alimentation digitale 1 |
| DGND1 | 6 | PWR | masse digitale 1 |
| XTALO | 7 | CLK | sortie quartz |
| XTALI | 8 | CLK | éntrée quartz |
| DVDD2 | 9 | PWR | + de l'alimentation digitale 2 |
| DGND2 | 10 | PWR | masse digitale 2 |
| XCS[4] | 11 | DI | entrée de sélection du circuit (actif à l'état bas) |
| SCLK[2] | 12 | DI | horloge du bus série |
| SI[2] | 13 | DI | entrée série |
| SO | 14 | DO3 | sortie série, active quand XCS=0, quel que soit XRESET |
| TEST | 15 | DI | réservée pour les tests, à connecter à DVDD |
| GPIO0 / SPIBOOT[2,3] | 16 | DIO | I/O 0 à usage général avec résistance pull-down de 100 kΩ |
| GPIO1[2] | 17 | DIO | I/O 1 à usage général |
| AGND0 | 18 | PWR | masse analogique 0 |
| AVDD0 | 19 | PWR | + de l'alimentation analogique 0 |
| RIGHT | 20 | AO | sortie canal droit |
| AGND2 | 21 | PWR | masse analogique 2 |
| RCAP | 22 | AIO | capacité de filtrage pour référence |
| AVDD2 | 23 | PWR | + de l'alimentation analogique 2 |
| LEFT | 24 | AO | sortie canal gauche |
| AGND3 | 25 | PWR | masse analogique 3 |
| XRESET | 26 | DI | reset asynchrone actif à l'état bas |
| DGND0 | 27 | PWR | masse digitale 0 |
| DVDD0 | 28 | PWR | + de l'alimentation digitale 0 |

le contrôle du volume, à cet égard la régulation du volume ne se fait pas de manière passive comme avec un classique potentiomètre, mais en temps réel par le décodeur MP3 afin qu'il ne se produise pas de distorsion lors des phases transitoires.

La touche « UP » augmente le volume et la touche « DOWN » le diminue. Les touches GRAVES et AIGUS ajustent la tonalité, respectivement pour les fréquences basses et hautes. Pour ajuster la tonalité vous devez appuyer sur l'une des touches GRAVES et AIGUS et simultanément sur « UP » ou « DOWN » pour régler le gain.

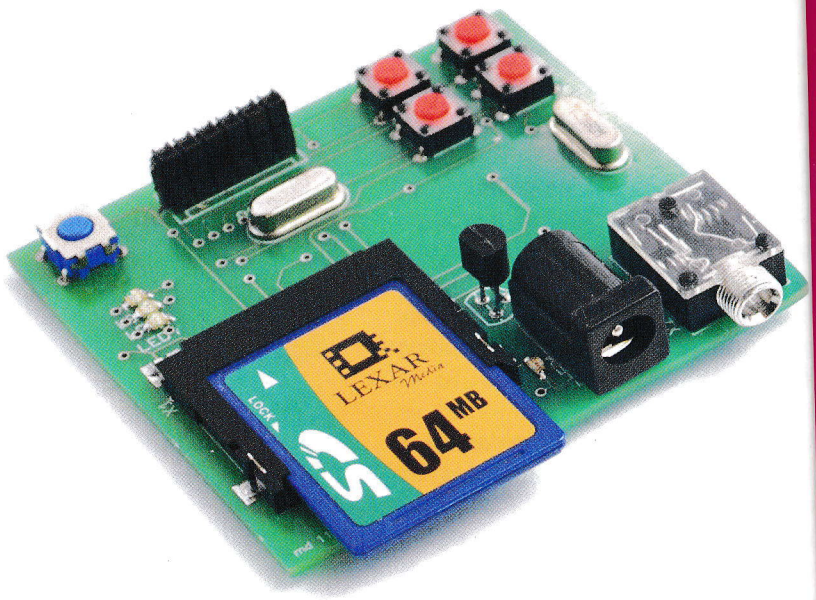
Les modifications du réglage de tonalité sont visualisées par la LED rouge, elle reste allumée jusqu'à ce que vous appuyiez sur l'une des 2 touches « UP » ou « DOWN ». Un clignotement rapide indique la modification du réglage.

Les **réglages du volume et de tonalité sont toujours confirmés par 2 clignotements successifs** de la **LED rouge**. Si l'**ajustement atteint la limite**, la **LED verte clignote** pour avertir que vous êtes soit au **minimum soit au maximum des possibilités de réglage**.

La **LED jaune s'allume lorsque le PIC communique avec le décodeur**. Vous remarquerez que les phases correspondant aux pauses pendant lesquelles le PIC lit les données dans la mémoire SD sont considérablement réduites.

Les 4 touches peuvent être utilisées pour exécuter les 4 modes lors de l'allumage ou d'une réinitialisation (bouton RESET). La LED verte clignote rapidement pendant quelques secondes, ce qui permet à l'utilisateur de choisir l'un des 4 modes. Si au cours de l'intervalle correspondant au clignotement de la LED aucune touche n'est pressée, le lecteur se lance en mode par défaut. Les 4 modes de fonctionnement sont résumés dans le Tableau 1.

Le lecteur est en mesure de reproduire les fichiers présents dans le répertoire « racine », y compris ceux contenus dans les sous-dossiers (ou sous-répertoires), sauf indication contraire.




Par exemple, lors d'une **lecture par ordre alphabétique**, le **lecteur ne lit que les fichiers MP3 présents dans le répertoire « racine »**, les **fichiers dans les sous-répertoires ne sont pas lus**.

Il ne faut pas oublier que le **lecteur ne peut reproduire que des fichiers dont la longueur des noms n'excède pas « 8 + 3 » caractères**, afin de permettre de lire les morceaux de musique par ordre alphabétique.

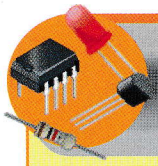
**Si les noms des fichiers sont trop longs, l'ordre alphabétique ne peut pas être garanti** car le « firmware » du microcontrôleur ne peut pas les lire. Dans ce cas cela signifie que la lecture des morceaux s'effectuera plutôt dans un ordre aléatoire que dans un ordre alphabétique.

**Dans tous les autres modes, le lecteur sera en mesure de lire correctement les fichiers même avec des noms longs**, sans aucune distinction de priorité.

Nous faisons cette remarque sur les noms longs des fichiers, car généralement lorsque vous copiez des fichiers à partir du disque dur de votre ordinateur il faut vérifier la compatibilité avec le périphérique cible. Malheureusement les fichiers présents sur un ordinateur utilisent certainement un système de fichiers de type FAT32 ou NTFS, ces systèmes attribuent des noms plus longs aux fichiers que ceux autorisés par une FAT16.

Si la carte SD est formatée avec un système de fichiers de type FAT32, les fichiers seront enregistrés sur la carte SD sans problème, mais le lecteur aura du mal à lire les fichiers présents sur la carte SD. Si vous avez l'intention de profiter de la fonction de reproduction par ordre alphabétique, l'idéal est de renommer les fichiers avant de les transférer sur la carte SD ou MMC ou plus commodément de faire des copies sur votre disque dur et de les renommer de manière à sauvegarder les fichiers originaux avec leurs noms complets.

Nous arrivons au terme de la description de ce lecteur MP3, vous pourrez l'intégrer dans une chaîne audio ou dans des installations existantes.



### Comment construire ce lecteur MP3

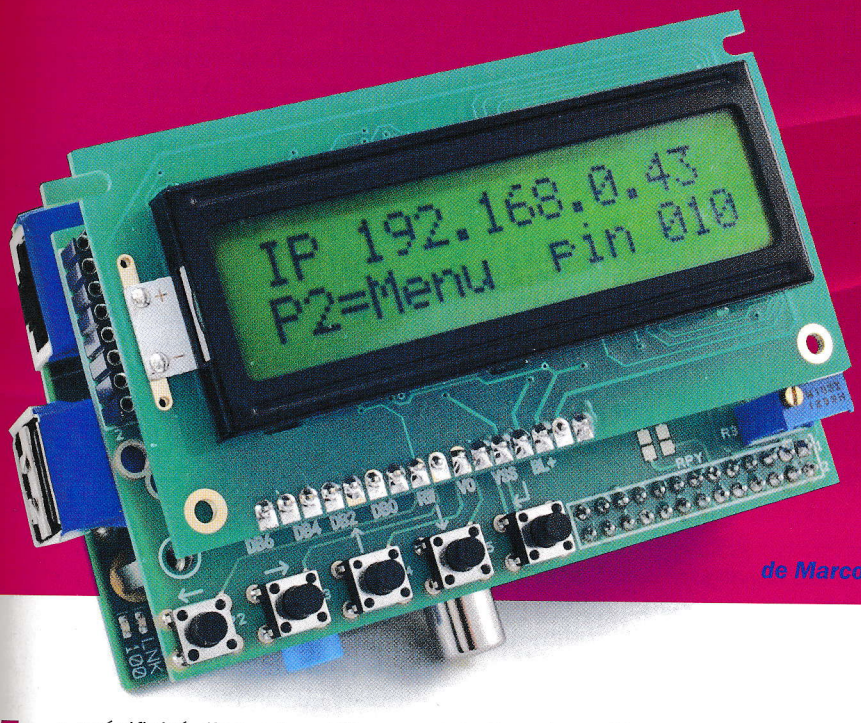
Tous les composants CMS ainsi que le PIC24HJ256GP206 et le décodeur VS1011E sont disponibles auprès de **COMELEC**. Le PIC n'est pas vendu programmé (boitier TQFP). Il se programme obligatoirement après avoir été soudé.

Les typons des circuits imprimés ainsi que le **programme** au format « **.hex** » du PIC sont téléchargeables sur notre site : **www.electroniquemagazine.com** dans le sommaire détaillé de la revue numéro 129 section « Télécharger ».

ET1074

# Afficheur LCD pour RaspberryPi

**Dans cet article, nous continuons la description de cartes d'extensions pour le RasperryPi. Après vous avoir proposé une carte d'extension de type convertisseur analogique-numérique d'une résolution de 16 bits dans le numéro 128 d'Electronique et Loisirs Magazine, nous allons décrire dans cet article une carte d'extension avec un afficheur LCD pour le RaspberryPi. Elle vous permettra de réaliser une interface de contrôle externe pour vos applications sans la nécessité de connecter en permanence un moniteur, un clavier et une souris. Dans les prochains articles nous examinerons plus en détail le fonctionnement et l'utilisation de ces deux cartes d'extensions.**




de Marco Magagnin

La spécificité d'un micro PC comme le **RaspberryPi** par rapport à un microcontrôleur, est qu'il peut effectuer plusieurs tâches à la fois en partageant les ressources. En plus des extensions déjà disponibles sur la carte de base, vous pouvez en ajouter d'autres pratiquement à l'infini, les seules limites résident dans l'utilisation des lignes de communication, dans la capacité de traitement et dans la quantité de mémoire disponible. Cela dépend beaucoup de la manière dont les applications sont conçues et mises en œuvre, en particulier en ce qui concerne la capacité à respecter le partage des ressources disponibles sans perturber les autres applications, c'est-à-dire sans créer un « **interblocage** ».

Un « **interblocage** » (ou « **Deadlock** » en anglais, appelé aussi « étreinte fatale ») est un phénomène qui peut survenir en programmation concurrente. L' « **interblocage** » **se produit lorsque deux processus concurrents s'attendent mutuellement**. Les processus bloqués dans cet état le sont définitivement, il s'agit donc d'une situation catastrophique.

# Le circuit intégré MCP23017

Le circuit intégré **MCP23017** permet de réaliser une extension avec **16 canaux digitaux d'entrées/sorties**, qui peuvent être pilotés par le bus de communication I2C. Il est disponible en différents boitiers : 28-PDIP, 28-SOIC et 28-SSOP, nous avons utilisé pour notre interface la version 28 broches PDIP. Le brochage du circuit est visible sur la figure A.

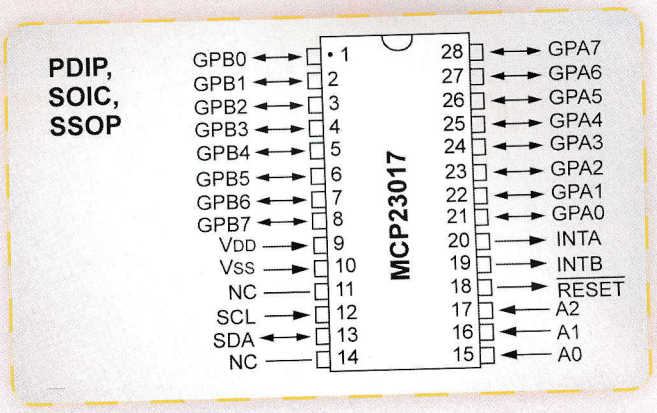



**Figure A : brochage du MCP23017.**

Le système « MASTER » (MAITRE) qui gère le circuit MCP23017 doit paramétrer correctement les registres de configuration, de 8 bits chacun, à partir desquels vous pouvez sélectionner le « sens » de chaque broche (en « entrée » ou en « sortie ») ainsi que la polarité. Sur la figure B vous pouvez voir l'architecture interne du MCP23017.

Les broches des ports GPIO (entrées/sorties) sont réparties en deux ports de 8 broches (8 bits) chacun et dont chaque broche est configurable individuellement. Les 2 ports sont nommés « PORTA » et « PORTB ».

Le registre « IODIRA/B » vous permet de configurer le « sens » de fonctionnement de chaque broche (en « entrée » ou en « sortie »).

La polarité de chaque broche peut être inversée (pour les « entrées ») en intervenant sur le registre d'inversion de polarité (« Polarity Inversion »). Il est évident que le système « MASTER » est capable de lire tous les registres.

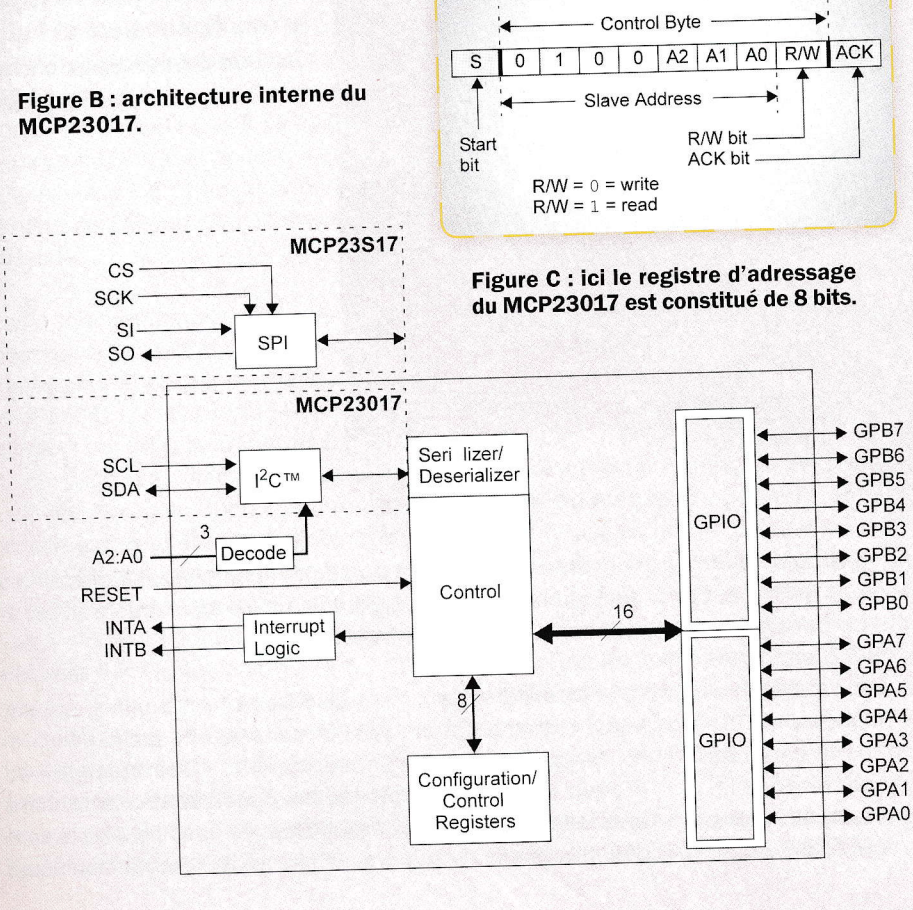



**Figure B : architecture interne du MCP23017.**




**Figure C : ici le registre d'adressage du MCP23017 est constitué de 8 bits.**

### *Registre d'adressage*

Le registre d'adressage du MCP23017 est constitué de 8 bits (voir la figure C). Les 4 bits les plus significatifs (ceux de gauche) sont paramétrés en usine par le fabricant sur la valeur fixe « 0100 » qui permet d'identifier le circuit. Les valeurs des 3 bits restants sont définies en fonction des niveaux des tensions présentes sur les broches A1, A2 et A3 du circuit intégré.

Cette configuration permet d'adresser sur le même bus I2C jusqu'à 8 périphériques MCP23017, soit un total de 128 broches I/O (entrées/sorties). Le dernier bit du registre d'adressage R/W pour READ/WRITE (LECTURE/ECRITURE) est utilisé pour définir le type d'opération que vous souhaitez effectuer, « 0 » pour effectuer une écriture des registres et « 1 » pour effectuer une opération de lecture.

### *Registre de contrôles*

Afin de contrôler le fonctionnement du circuit intégré MCP23017, nous avons besoin d'écrire et de lire dans une série de registres dans une séquence bien définie.

Une vue d'ensemble des registres disponibles est représentée sur la figure D. La première colonne du tableau indique le nom de registre, la deuxième colonne l'adresse hexadécimale à laquelle vous pouvez lire ou écrire le contenu du registre.

| Register Name | Address (hex) |
|---|---|
| IODIRA | 00 |
| IODIRB | 01 |
| IPOLA | 02 |
| IPOLB | 03 |
| GPINTENA | 04 |
| GPINTENB | 05 |
| DEFVALA | 06 |
| DEFVALB | 07 |
| INTCONA | 08 |
| INTCONB | 09 |
| IOCON | 0A |
| IOCON | 0B |
| GPPUA | 0C |
| GPPUB | 0D |
| INTFA | 0E |
| INTFB | 0F |
| INTCAPA | 10 |
| INTCAPB | 11 |
| GPIOA | 12 |
| GPIOB | 13 |
| OLATA | 14 |
| OLATB | 15 |

**Figure D : cart contrôlent le**

Les registres IO longueur chacu de configurer c sortie (I/O) des ou en « sortie ». che en « entrée pondant dans à « 1 », tandis « sortie » le bit

Les registres permettent d de chaque bro ports « PORTA respondant à le registre GP opposé à celu le bit est à « 0 le même état sur la broche

Pour définir configurées saire de para et OLATB de correspondar valeur « 0 » si et la valeur «

Les registres mettent de « entrées ». l'entrée est seront « inte une logique

Si le bit est « interprétés

| Register Name | Address (hex) | bit 7 | bit 6 | bit 5 | bit 4 | bit 3 | bit 2 | bit 1 | bit 0 | POR/RST value |
|---|---|---|---|---|---|---|---|---|---|---|
| IODIRA | 00 | IO7 | IO6 | IO5 | IO4 | IO3 | IO2 | IO1 | IO0 | 1111 1111 |
| IODIRB | 01 | IO7 | IO6 | IO5 | IO4 | IO3 | IO2 | IO1 | IO0 | 1111 1111 |
| IPOLA | 02 | IP7 | IP6 | IP5 | IP4 | IP3 | IP2 | IP1 | IP0 | 0000 0000 |
| IPOLB | 03 | IP7 | IP6 | IP5 | IP4 | IP3 | IP2 | IP1 | IP0 | 0000 0000 |
| GPINTENA | 04 | GPINT7 | GPINT6 | GPINT5 | GPINT4 | GPINT3 | GPINT2 | GPINT1 | GPINT0 | 0000 0000 |
| GPINTENB | 05 | GPINT7 | GPINT6 | GPINT5 | GPINT4 | GPINT3 | GPINT2 | GPINT1 | GPINT0 | 0000 0000 |
| DEFVALA | 06 | DEF7 | DEF6 | DEF5 | DEF4 | DEF3 | DEF2 | DEF1 | DEF0 | 0000 0000 |
| DEFVALB | 07 | DEF7 | DEF6 | DEF5 | DEF4 | DEF3 | DEF2 | DEF1 | DEF0 | 0000 0000 |
| INTCONA | 08 | IOC7 | IOC6 | IOC5 | IOC4 | IOC3 | IOC2 | IOC1 | IOC0 | 0000 0000 |
| INTCONB | 09 | IOC7 | IOC6 | IOC5 | IOC4 | IOC3 | IOC2 | IOC1 | IOC0 | 0000 0000 |
| IOCON | 0A | BANK | MIRROR | SEQOP | DISSLW | HAEN | ODR | INTPOL | — | 0000 0000 |
| IOCON | 0B | BANK | MIRROR | SEQOP | DISSLW | HAEN | ODR | INTPOL | — | 0000 0000 |
| GPPUA | 0C | PU7 | PU6 | PU5 | PU4 | PU3 | PU2 | PU1 | PU0 | 0000 0000 |
| GPPUB | 0D | PU7 | PU6 | PU5 | PU4 | PU3 | PU2 | PU1 | PU0 | 0000 0000 |
| INTFA | 0E | INT7 | INT6 | INT5 | INT4 | INT3 | INT2 | INT1 | INT0 | 0000 0000 |
| INTFB | 0F | INT7 | INT6 | INT5 | INT4 | INT3 | INT2 | INT1 | INT0 | 0000 0000 |
| INTCAPA | 10 | ICP7 | ICP6 | ICP5 | ICP4 | ICP3 | ICP2 | ICP1 | ICP0 | 0000 0000 |
| INTCAPB | 11 | ICP7 | ICP6 | ICP5 | ICP4 | ICP3 | ICP2 | ICP1 | ICP0 | 0000 0000 |
| GPIOA | 12 | GP7 | GP6 | GP5 | GP4 | GP3 | GP2 | GP1 | GP0 | 0000 0000 |
| GPIOB | 13 | GP7 | GP6 | GP5 | GP4 | GP3 | GP2 | GP1 | GP0 | 0000 0000 |
| OLATA | 14 | OL7 | OL6 | OL5 | OL4 | OL3 | OL2 | OL1 | OL0 | 0000 0000 |
| OLATB | 15 | OL7 | OL6 | OL5 | OL4 | OL3 | OL2 | OL1 | OL0 | 0000 0000 |

**Figure D : cartographie de la mémoire qui contient les valeurs des registres qui contrôlent le circuit MCP23017.**

Les registres IODIRA et IODIRB, d'une longueur chacun de 8 bits, permettent de configurer chaque broche d'entrée/sortie (I/O) des ports A ou B en « entrée » ou en « sortie ». Pour configurer une broche en « entrée », la valeur du bit correspondant dans le registre doit être égale à « 1 », tandis que pour la configurer en « sortie » le bit doit être à « 0 ».

Les registres GPIOA et GPIOB vous permettent de configurer la polarité de chaque broche respectivement des ports « PORTA » et « PORTB ». Si le bit correspondant à une broche est mis à « 1 », le registre GPIO reflète l'état logique opposé à celui présent sur la broche. Si le bit est à « 0 », le registre GPIO reflète le même état logique que celui présent sur la broche (d'entrée).

Pour définir les niveaux des broches configurées en « sortie », il est nécessaire de paramétrer les registres OLATA et OLATB de manière à ce que les bits correspondant aux broches prennent la valeur « 0 » si vous voulez un niveau bas et la valeur « 1 » pour un niveau haut.

Les registres IPOLA et IPOLB vous permettent de définir l'état logique des « entrées ». Si le bit correspondant à l'entrée est réglé à « 1 », les niveaux seront « interprétés » par l'entrée dans une logique inversée.

Si le bit est à « 0 », les niveaux seront « interprétés » dans une même logique.

Les registres GPPUA et GPPUB permettent d'activer les résistances internes de tirage (pull-up) sur les broches d'entrée qui sont mises à niveau haut. Si le bit correspondant à l'entrée est mis à « 1 », la résistance de pull-up est activée (bit = 0 ⟶ résistance désactivée).

Enfin, vcontinuons en approfondissant sur la gestion des interruptions disponibles avec le circuit MCP23017.

Les broches 19 et 20 sont connectées au module de gestion des interruptions, respectivement « INTA » pour le PORTA et « INTB » pour le PORTB. Avec le MCP23017 il est possible de configurer chaque broche de sorte que le changement de son état ait lieu lorsque la broche d'interruption du port correspondant est activée.

De cette façon, il n'est pas nécessaire que le système de gestion « MASTER » lise en continu l'état de toutes les broches pour vérifier si un changement a lieu, il suffit d'établir une gestion des interruptions des 2 broches INTA et/ou INTB pour être informé d'une variation sur l'une des broches.

À ce stade, vous pouvez lire le registre d'état du PORT correspondant et le comparer aux valeurs précédentes afin d'identifier un changement. Avant d'activer la surveillance des broches à l'aide des registres GPINTENA et GPINTENB, il est nécessaire de configurer les conditions dans lesquelles une interruption doit être activée. Pour cela, vous devez utilisez les 2 registres DEFVALA, DEFVALB et INTCONA, INTCONB.

La première paire de registres (DEFVALA, DEFVALB) permet de configurer, pour chaque broche des deux PORT, un niveau de référence correspondant au déclenchement de l'interruption. Dans tous les cas, cette configuration dépend de la configuration du registre « INTCON ». Si le bit correspondant à une broche est configuré à « 0 », l'interruption sera déclenchée lorsque la broche sera à un niveau « 1 », et l'inverse dans l'autre cas.

La configuration des bits des registres INTCONA et INTCONB vous permet de définir si l'interruption doit être contrôlée par les conditions imposées par les registres DEFVAL, ou simplement lors d'un changement d'état des niveaux des broches. Si vous réglez à « 0 » le bit correspondant à une broche dans le registre INTCON, l'interruption sera déclenchée à chaque changement d'état de la broche, si vous le réglez à « 1 », l'interruption sera déclenchée par les conditions définies par les registres DEFVAL.

Les conditions pour lesquelles il est possible de détecter une interruption sur les broches INTA et INTB dépend de la configuration du bit INTCOL (bit 1) du registre IOCON, ce qui vous permet de configurer différentes options de fonctionnement du circuit intégré.

La valeur par défaut est « 0 », ce qui signifie que la broche d'interruption est normalement à un niveau élevé, si elle passe à un niveau bas l'interruption est déclenchée.

Pour connaitre la broche qui a déclenché l'interruption, il suffit de lire l'un des registres INTFA, INTFB. Le ou les bits qui sont à « 1 » correspondent aux bits qui ont généré l'interruption. À ce stade, le niveau de la broche en question peut être trouvé en lisant les registres correspondant INTCAPA et INTCAPB. Ces registres « prennent une image » de l'état des broches au moment de l'interruption et le « garde en mémoire » jusqu'à ce qu'il soit lu.

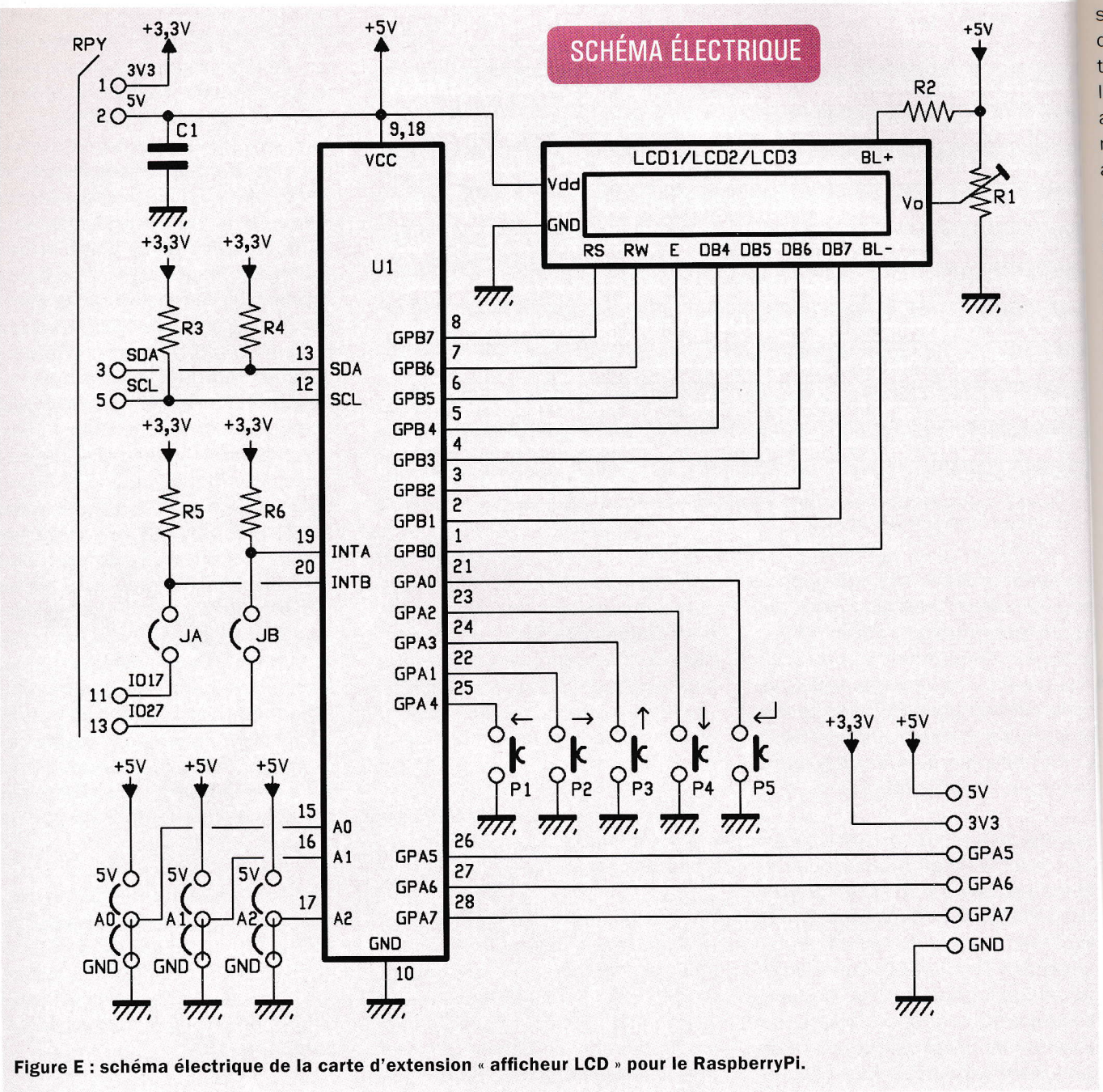



**Figure E : schéma électrique de la carte d'extension « afficheur LCD » pour le RaspberryPi.**

Un exemple concret d'« interblocage » peut se produire lorsque deux processus (ou applications) essayent d'acquérir deux verrous M1 et M2 dans un ordre différent. Un « interblocage » est possible si la séquence d'opération suivante survient :

- *la TâcheA obtient M1 ;*
- *la TâcheB obtient M2 ;*
- *la TâcheA attend pour obtenir M2 (qui est entre les « mains » de TâcheB) ;*
- *la TâcheB attend pour obtenir M1 (qui est entre les « mains » de TâcheA).*

Dans cette situation, les deux tâches (TâcheA et TâcheB) sont définitivement bloquées.

Dans cette perspective, nous décrivons une série de cartes d'extensions spécialement conçues pour le **RaspberryPi**, de manière à être utilisées simultanément les unes avec les autres sans créer de conflits matériels (hardware) ou logiciels (software). La deuxième d'entre elles est une carte à afficheur LCD qui permet de réaliser une interface de contrôle externe pour vos applications. Dans cet article, nous allons vous décrire la partie matérielle de la carte d'extension « **afficheur LCD** », la bibliothèque qui permet une simplification considérable de l'utilisation de l'afficheur et nous approfondirons le fonctionnement du circuit intégré **MCP23017** utilisé comme « interface » entre le **RaspberryPi** et l'afficheur LCD. Avant de commencer la description, une remarque sur le développement des applications.

Dans les précédents articles sur le **RaspberryPi**, nous avons parlé de la nécessité de concevoir des applications pour les systèmes embarqués et, parmi ceux-ci, ceux basés sur le

système d'exploitation **GNU/LINUX**, de sorte qu'ils soient en mesure de travailler l'un avec l'autre en utilisant les ressources disponibles de manière appropriée. Nous avons précédemment traité le sujet de manière plus approfondie, voici un bref rappel.

Prenons comme exemple la carte décrite dans cet article, qui, comme nous le verrons plus loin, nécessite l'utilisation du bus I2C pour la communication avec le **RaspberryPi**. Si nous développons une application qui gère exclusivement l'afficheur LCD, le logiciel équivalent que nous devrions écrire pour un microcontrôleur serait un programme dans une boucle sans fin qui gèrerait les boutons et écrirait selon une logique prédéfinie des caractères sur l'afficheur.

Si nous voulons que l'afficheur LCD puisse être utilisé pour différentes applications, nous devons traiter le problème d'une manière différente.

Supposons par exemple que nous voulions toujours utiliser la carte comme dans le cas précédent, mais aussi qu'elle soit capable d'afficher des messages provenant de différentes applications qui fonctionnent en association; Par exemple lors de l'arrivée d'un appel téléphonique, ou de la variation d'une entrée exploitée via un serveur Web, ou lors d'un problème d'alimentation ou de tout autre besoin.

Dans ce type d'architecture, différentes applications s'exécutant simultanément sur le système produisent un ensemble d'événements, et ont besoin d'engager une communication avec le bus I2C pour afficher leur message sur l'écran LCD.

Si une application occupe déjà le bus I2C, et si pour une raison quelconque d'autres applications ont besoin de communiquer avec le même bus, elles le trouveront occupé (« busy ») et ne pourront rien faire d'autre que « de se mettre en erreur » (ou « planter » dans le jargon informatique).

La première solution qui vient à l'esprit est de développer une énorme application qui gère tous vos besoins de façon unique, un peu comme un « énorme microcontrôleur ».

Il est évident que cette approche comporte plus de points négatifs que positifs : difficulté de conception de l'application, difficulté de réalisation et de mise à jour, difficulté de synchronisation entre les différentes parties, fragilité car si une partie de l'application se bloque c'est toute l'application qui ne fonctionne plus.

Pour résoudre ce type d'exigence des applications, nous avons besoin d'architectures qui peuvent soutenir différentes applications concurrentes et collaboratives et gérer la communication entre les différents acteurs, en fonction du serveur qui gère le partage des ressources. Dans ce dernier, les différentes exigences des applications doivent être mises en œuvre afin que les clients puissent utiliser les ressources « centralisées » à l'aide de requêtes ayant des protocoles communs.

Par exemple, supposons que 2 ressources utilisent exclusivement les ressources assignées à la carte « Afficheur LCD » qui utilise le bus I2C et à une autre carte GSM/GPS (que nous présenterons dans les prochains articles) qui utilise le port série.

Pour chacune de ces ressources il est nécessaire de créer un « serveur » qui gère de manière autonome les ressources et qui doit accepter les demandes des différents clients (ici les cartes d'extensions) sur le canal de communication par défaut.

Il existe de nombreuses alternatives possibles : les sockets TCP/IP, les bases de données, les simples fichiers, etc.

Pour le moment, nous n'approfondissons pas. L'important est que les mécanismes de communication soient capables de gérer correctement le phénomène, par exemple la génération d'un code ou la gestion des « feux de signalisation » afin de bien coordonner les requêtes des clients. Lorsque le client a besoin d'accéder à une ressource gérée de façon centralisée, il ne doit pas accéder directement à cette ressource, mais doit utiliser le canal de communication par défaut pour cette ressource avec le protocole et les procédures prévues.

En résumé, lors de la conception d'un système embarqué, celui-ci ne doit pas être dédié uniquement à une seule application, mais plutôt à l'ensemble de l'architecture que vous voulez développer.

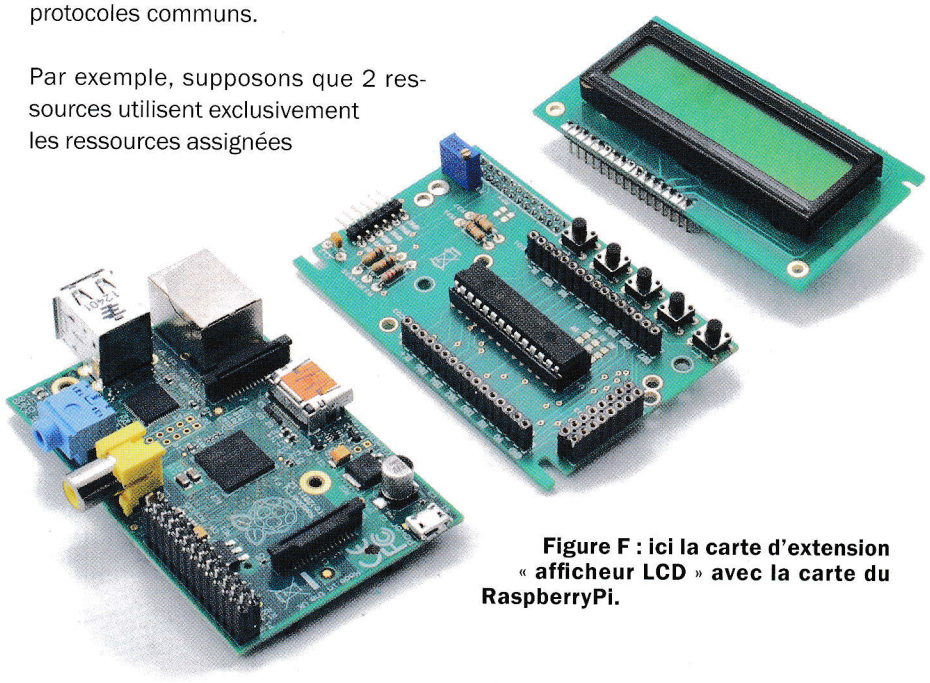


**Figure F : ici la carte d'extension « afficheur LCD » avec la carte du RaspberryPi.**

Est-ce mieux ou pire qu'un microcontrôleur ? Cette question n'a pas de réponse définitive. Cela dépend de ce que vous voulez réaliser et des exigences requises par le projet. Certes, nous trouvons de plus en plus de circuits intégrés à haute intégration capables de collaborer avec d'autres environnements tels que des FPGA ou des PLC.

Revenons à notre carte d'extension pour laquelle nous vous présentons un programme simple à utiliser, et facilement adaptable selon l'architecture que vous voulez employer.

## Le schéma électrique

Pour éviter d'utiliser la majorité des broches du connecteur d'entrées/sorties du **RaspberryPi**, nous avons utilisé le circuit intégré **MCP23017** de Microchip qui dispose de **16 entrées/sorties numériques** et peut être piloté via le **BUS I2C**.

Grâce à cette configuration, nous utilisons seulement 2 broches du **RaspberryPi**, et nous pouvons piloter un **écran LCD**, **5 boutons poussoirs** et gérer **3 entrées auxiliaires**.

La configuration de base est conçue pour réaliser un système de « **menu déroulant** » qui vous permet de choisir les entrées et configurer les paramètres ou afficher une information particulière. En fait, le programme peut utiliser les boutons et les entrées à votre convenance.

La **carte est munie de plusieurs connecteurs pouvant accueillir différents types d'afficheurs LCD** et en particulier les modèles suivants qui, tout en ayant une taille différente, ont en commun la même configuration des broches et les mêmes commandes pour la communication :

- **Afficheur LCD 2 lignes de 16 caractères rétroéclairé**. Il utilise le contrôleur HD44780 ; alimentation + 5 VDC ; dimensions 84 x 44 x 13 mm. (référence COMELEC ACM1602B-FL-YBW).
- **Afficheur LCD 2 lignes de 16 caractères blancs sur fond noir rétroéclairé**. Il utilise le contrôleur HD44780 ; alimentation + 5 VDC ; dimensions 80 x 36 x 8,40 mm. (référence COMELEC 1446-LCD16X2WB).
- **Afficheur LCD 2 lignes de 8 caractères blancs sur fond bleu rétroéclairé**. Il utilise le contrôleu Samsung ST7066U ; alimentatio + 5 VDC ; dimensions 58 x 32 11,8 mm. (référence COMELE 1446-LCD8X2BN).

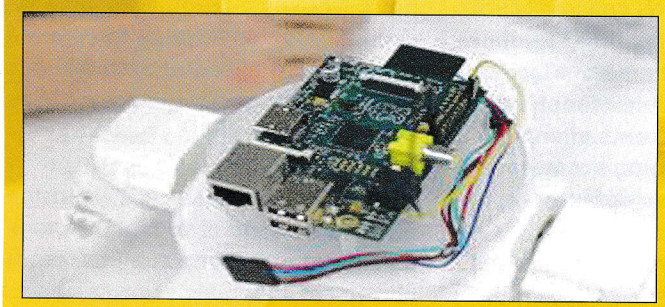
Retournons au schéma électrique visible sur la figure E, le circuit inté gré **MCP23017**, que nous décrivons plus en détail dans l'encadré dédié offre **16 entrées/sorties** numériques divisées en 2 « **PORT** » de **8 lignes chacun**, appelés **GPA** et **GPB**. Le port **GPB est dédié à la gestion de l'affi cheur LCD** de la manière suivante : les broches GPB1, GPB2, GPB3 et GPB4 sont connectées respectivement aux 4 lignes de données du LCD (DB7, DB6, DB5 et DB4). La broche GPB5 contrôle l'entrée « ENABLE » du LCD, la broche GPB6 l'entrée R/W et la broche GPB7 l'entrée RS qui vous permet de sélectionner les registres d'instructions et d'écriture.

# Rapiro,
## un robot humanoïde à base de RaspberryPi



**Parmi les nombreuses applications de type serveur Web, systèmes de reconnaissance, Médias Center, stations météo, à base de RaspberryPi, voici l'une des premières applications dans le domaine de la robotique utilisant une carte économique RaspberryPi/Linux. Rapiro, un petit robot humanoïde entièrement programmable est conçu principalement pour le domaine éducatif.**

**Rapiro dispose de 12 servomoteurs, il est capable de marcher, de bouger la tête et de répondre à des commandes vocales. Il contient une carte RaspberryPi qui gère tous les systèmes et peut être contrôlé à distance via un smartphone. La communication est réalisée à l'aide d'un « dongle » Bluetooth et une application spécifique qui doit être téléchargée sur le smartphone.**



**Le petit robot (environ 30 cm de haut) dispose également d'une caméra, ses yeux s'allument grâce à 2 LED, il dispose d'un capteur de proximité et d'un haut-parleur. Pour les commandes des servomoteurs, une carte compatible Arduino est utilisée afin d'avoir une compatibilité maximale pour la programmation.**

**Selon les concepteurs du projet qui l'ont baptisé « Kickstarter » et ont déjà recueilli les fonds nécessaires, Rapiro coûte ¼ du prix des robots semblables et un dixième du prix par rapport à d'autres robots humanoïdes à base de cartes Linux embarquées (www.kickstarter.com).**

## Plan de montage de la carte afficheur LCD

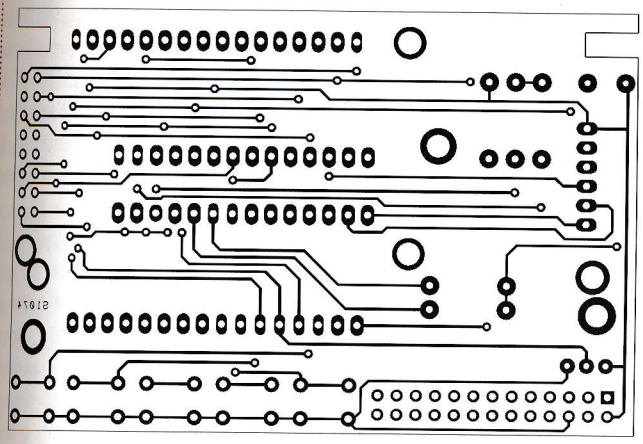


Figure G : circuit imprimé à l'échelle 1 : 1 côté soudures de la carte d'extension « afficheur LCD ».

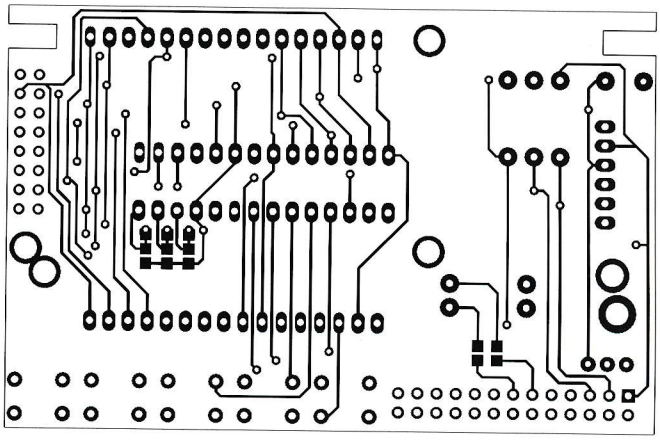


Figure H : circuit imprimé à l'échelle 1 : 1 côté composants de la carte d'extension « afficheur LCD ».

Figure I : schéma d'implantation des composants de la carte d'extension « afficheur LCD ».

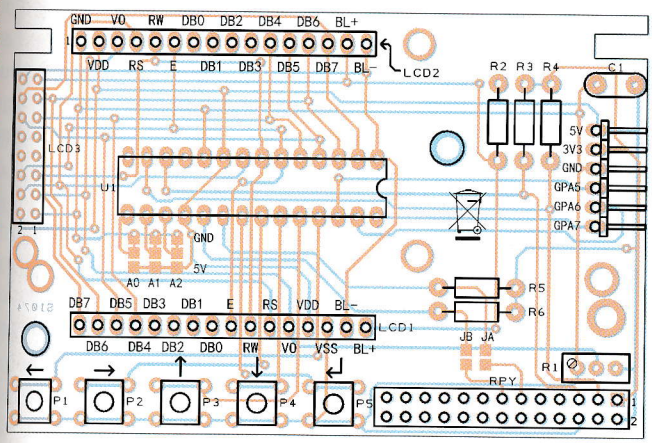
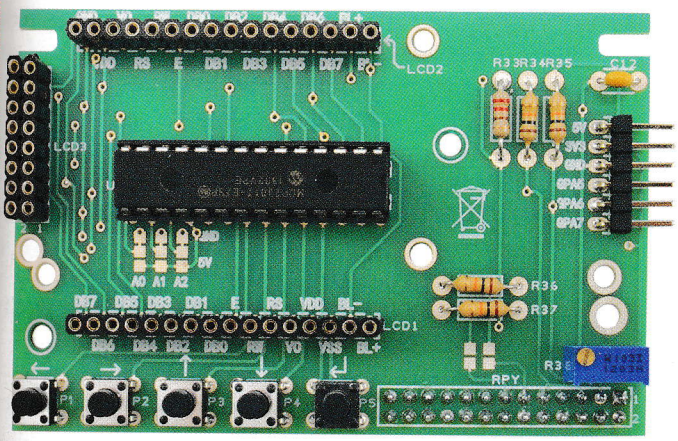



Figure J : photo de l'un de nos prototypes de la carte d'extension « afficheur LCD ».



## Liste des composants de la carte afficheur LCD

R1..... Trimmer multitours 10 kΩ
R2..... 100 Ω
R3..... 10 kΩ
R4..... 10 kΩ
R5..... 10 kΩ
R6..... 10 kΩ

C1..... 100 nF multicouche
U1..... MCP23017-E/SP

P1..... microswitch
P2..... microswitch
P3..... microswitch
P4..... microswitch
P5..... microswitch

Divers

Afficheur LCD voir texte *
Entretoises femelle/femelle 18 mm
Vis 3MA 10 mm
Support ci 2 x 14 broches
Barrette mâle 6 broches coudée 90 °
Barrette femelle 8 broches (x2)
Barrette femelle 16 broches (x2)
Connecteur femelle 26 broches pour RaspberryPi

Les **5 boutons poussoirs** sont connectés aux broches du **port GPA :** GPA0 à P5, GPA1 à P2, GPA2 à P4, GPA3 à P3 et GPA4 au poussoir P1.

L'alimentation **+ 5 V** provient de la broche 2 du connecteur **GPIO** du **RaspberryPi** et la masse est reliée à la broche 6.

Le trimmer de 10 kΩ est connecté à la broche V0 du LCD pour régler le contraste de l'écran. Les **broches d'interruption INTA** et **INTB** sont reliées par les jumpers **JA** et **JB** à GPIO17 et GPIO27 du **RaspberryPi**, de manière à les rendre accessibles pour différents types d'applications.

Sur la carte sont disponibles sur le connecteur du port GPA les broches GPA5, GPA6 et GPA7, qui permettent une gestion « avancée » des boutons. Sur le plan de montage, vous pouvez voir l'agencement des composants de la carte.

La seule remarque importante, en dehors des précautions habituelles, est de positionner le connecteur double « RPY » de façon à espacer convenablement la carte de celle du **RaspberryPi** ; ceci, afin de permettre le montage en « sandwich » lors d'une utilisation en conjonction avec d'autres cartes d'extensions, telles que la carte GSM que nous allons vous présenter dans les prochains numéros de la revue.

La possibilité de montage en « couches successives » est possible grâce au circuit **MCP23017** qui est connecté au port GPIO du **RaspberryPi** uniquement avec les broches nécessaires à la communication I2C et à l'alimentation.

Toutes les autres broches du **RaspberryPi** sont tout simplement reportées (en fait libres). En outre, même si la plupart des entrées/sorties du circuit intégré **MCP23017** sont utilisées pour la gestion des boutons et l'afficheur LCD présents sur la carte, il existe 3 broches avec une gestion avancée qui sont disponibles pour une utilisation personnalisée.

Pour le montage de la carte aidez-vous des figures I et J, les typons des circuits sont disponibles en téléchargement sur notre site web. Vous pouvez voir sur la figure 1 la carte d'extension « **Afficheur LCD** » montée sur le **RaspberryPi** et prête à fonctionner.

## Utilisation pratique de la carte

Dans un premier temps, vous devez monter l'un des 3 afficheurs LCD prévus, puis insérer le connecteur de la carte dans celui du **RaspberryPi**, en faisant attention que le dos de la carte ne vienne pas en contact avec la prise USB ou Ethernet du **RaspberryPi**. En cas de doute, protégez les connecteurs avec de l'adhésif isolant. Connectez les périphériques, le réseau et l'alimentation du **RaspberryPi** et démarrez de manière habituelle.

Pour communiquer avec le circuit **MCP23017** il est nécessaire d'utiliser

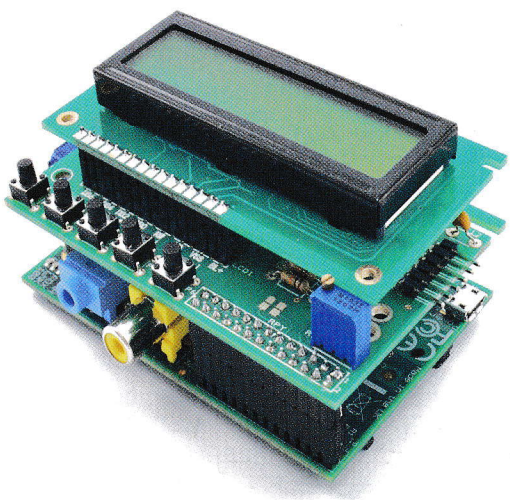


**Figure 1 : ici la carte d'extension « Afficheur LCD » montée sur le RaspberryPi et prête à fonctionner.**

le bus I2C et, par conséquent, nous devons activer le module de gestion du bus I2C, qui, comme vous le savez maintenant si vous nous suivez depuis le numéro 123 d'Electronique et Loisirs Magazine, est désactivé lors de l'installation par défaut de « Raspbian ». ce module est désactivé.

Pour ceux qui suivent depuis le début les articles consacrés au **RaspberryPi**, cette opération n'est pas un problème. Mais pour ceux qui sont nouveaux dans l'environnement « Raspbian » - système d'exploitation **GNU/LINUX** d **Raspberry Pi** – ils doivent se référ aux lignes directrices du numéro 12 d'Electronique et Loisirs Magazi qui traite de la carte d'extension A/ 16 bits pour **RaspberryPi** compatibl Arduino.

Rappelons seulement qu'en ce qu concerne les articles publiés dans l revue traitant du **RaspberryPi**, nou avons adopté le système d'exploita tion « Raspbian » qui est constamme mise à jour et pour la gestion à dis tance nous utilisons les outils « SSH » « Putty » (ou Kitty) et « WinSCP ».

Pour bien comprendre les séquences qui vont être décrites ci-après, il es nécessaire de se reporter aux article précédents concernant le **RaspberryP** publiés à partir numéro 123 d'Electro nique et Loisirs Magazine, car nous n pouvons pas décrire ici toute la procé dure depuis le début (la revue entière n'y suffirait pas).

Maintenant, nous devons activer le « driver » (pilote) pour la gestion du bus I2C, installer la bibliothèque en Python pour la gestion de l'afficheur LCD et écrire un premier programme de test pour voir que tout fonctionne correctement.

Rappelons brièvement le processus de mise à jour du système d'exploitation et activons le « driver » pour la gestion du bus I2C. Tapons les commandes en tant qu'utilisateur « **root** » :

```
apt-get update
```

```
apt-get upgrade
```

Pour rendre utilisable le module de gestion du bus I2C nous devons l'enlever de la liste noire, puis l'ajouter à l'ensemble des modules reconnus par le noyau. Ouvrons le fichier de configuration qui contient la liste noire des modules (masqué), avec la commande (voir la figure 2) :

```
nano /etc/modprobe.d/raspi-blacklist.conf
```

« Nano » est un éditeur de texte minimaliste qui s'exécute sur le terminal.

Eliminons le module I2C de la liste noire en effaçant la ligne ou, comme nous l'avons fait, en mettant un « **#** » en début de ligne (voir la figure 3). Appuyez sur les touches « **Ctrl+X** » et ensuite sur « **Y** » afin d'enregistrer le fichier après l'avoir édité. Effectuons un redémarrage pour que les modifications prennent effet.

Maintenant, nous devons faire en sorte que le module « libéré » soit chargé et devienne partie intégrante du noyau.

Pour cela, nous avons 2 possibilités :

1. la première est de charger le module à l'aide d'une commande, le module restera activé tant que le **RaspberryPi** sera allumé. Au prochain démarrage, le module doit être rechargé avec la commande ;
2. la seconde est de **charger le module directement au démarrage du système** d'exploitation et de le rendre disponible pour toutes les applications immédiatement après le démarrage, ce qui est essentiel dans un système de type serveur sans surveillance.

L'option 1 ci-dessus nécessite l'utilisation de la commande « **modprobe** ». Tapons la commande suivante (voir la figure 4) :

*modprobe i2c-dev*

Nous pouvons voir avec succès l'activation du driver grâce à la commande ci-après qui affiche la liste de tous les modules installés (voir la figure 5) :

*lsmod*

Etant donné que sous **GNU/LINUX** tout (ou presque) est sous forme de fichier, si nous allons dans le répertoire « /dev » nous verrons apparaître les fichiers liés aux périphériques (device) « **i2c-0** » et « **i2c-1** » (voir la figure 6). La commande « **modprobe** » vous permet de « charger » et de « décharger » (désactiver) les modules et conserve ceux-ci tant que le Raspberry Pi reste allumé.

Dans le cas d'un arrêt, ou même d'un redémarrage, les modules devront être rechargés manuellement.

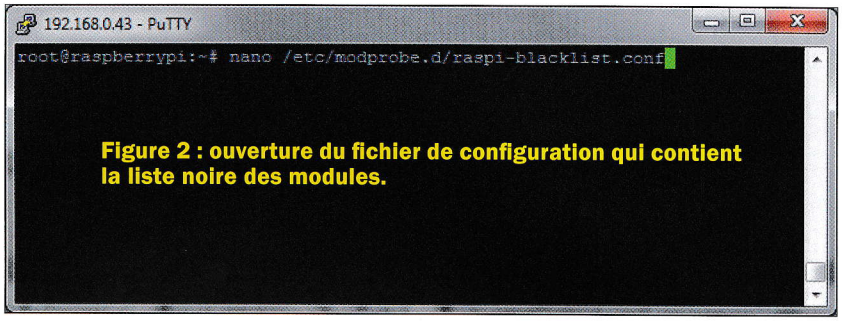



**Figure 2 : ouverture du fichier de configuration qui contient la liste noire des modules.**

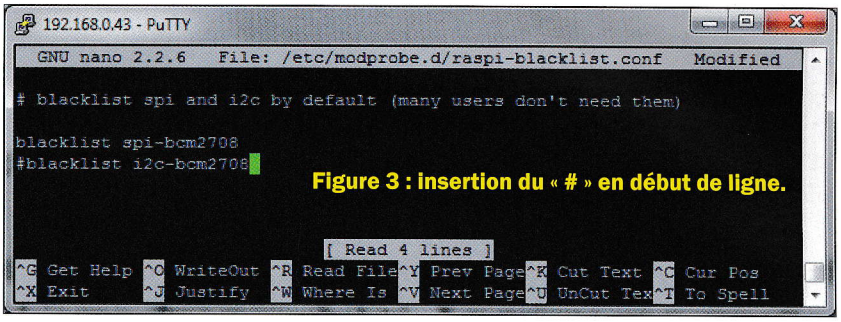



**Figure 3 : insertion du « # » en début de ligne.**

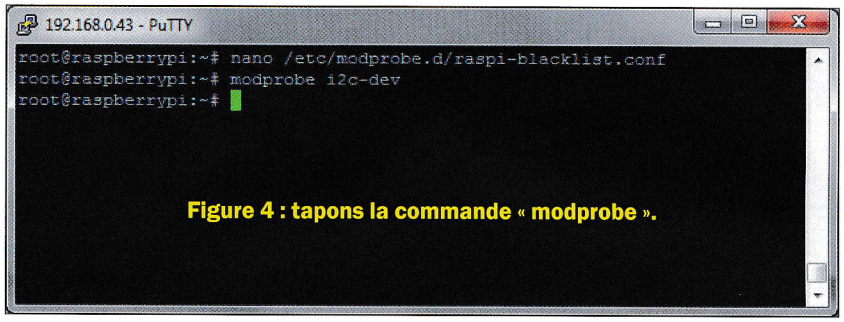



**Figure 4 : tapons la commande « modprobe ».**

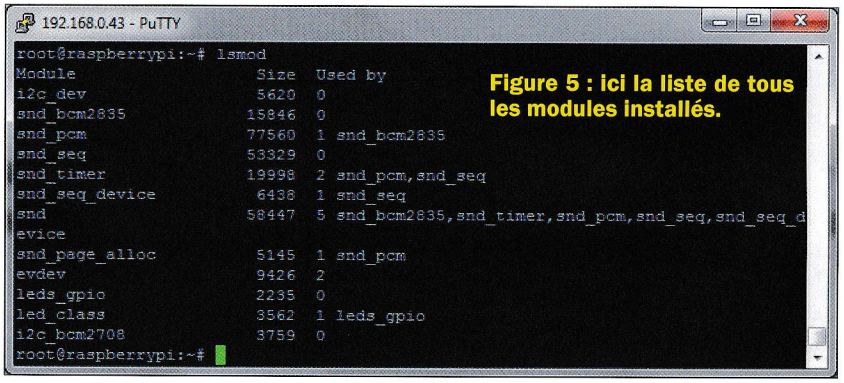



**Figure 5 : ici la liste de tous les modules installés.**

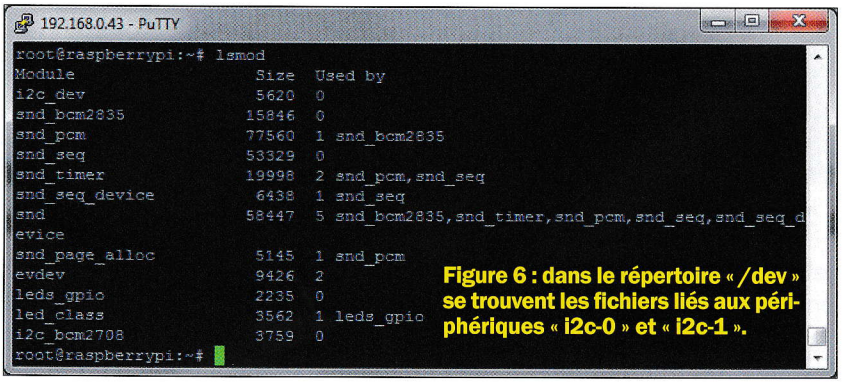



**Figure 6 : dans le répertoire « /dev » se trouvent les fichiers liés aux périphériques « i2c-0 » et « i2c-1 ».**

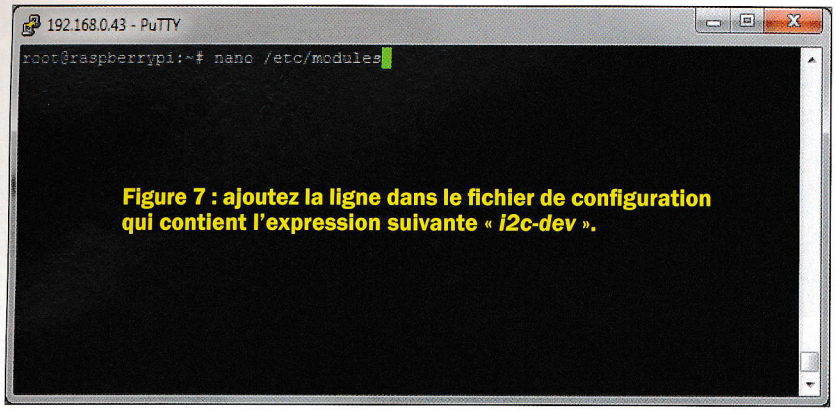



**Figure 7 : ajoutez la ligne dans le fichier de configuration qui contient l'expression suivante « *i2c-dev* ».**

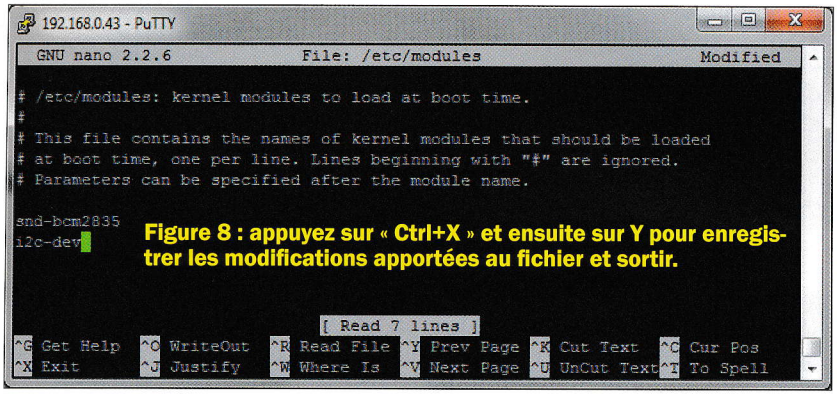



**Figure 8 : appuyez sur « Ctrl+X » et ensuite sur Y pour enregistrer les modifications apportées au fichier et sortir.**

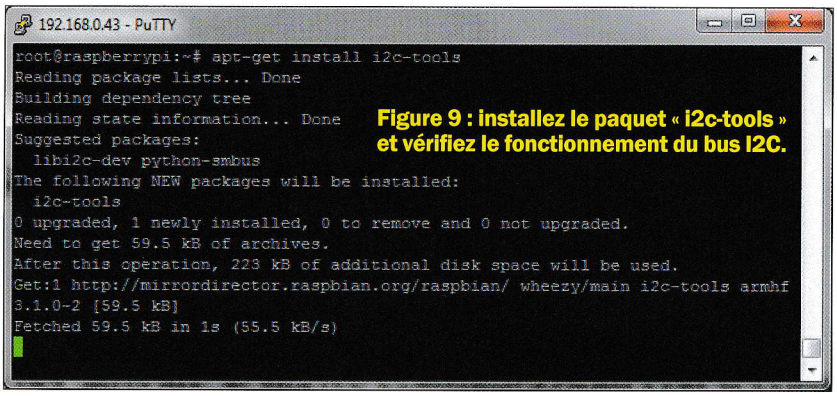



**Figure 9 : installez le paquet « i2c-tools » et vérifiez le fonctionnement du bus I2C.**

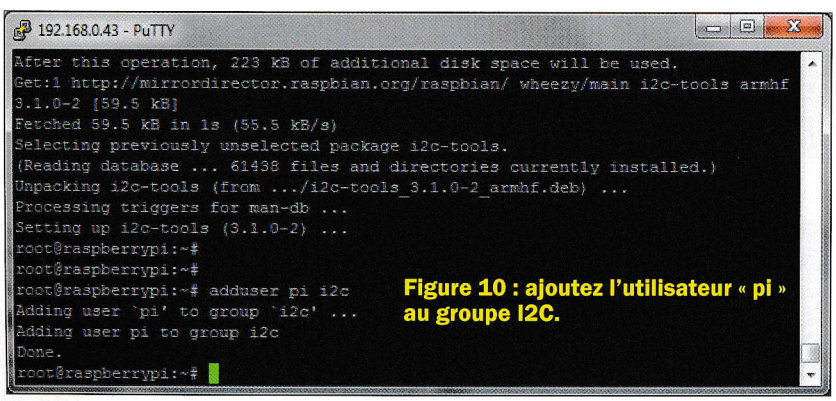



**Figure 10 : ajoutez l'utilisateur « pi » au groupe I2C.**

Cette condition n'est pas adaptée pour fonctionner avec une application autonome qui doit s'exécuter de manière automatique.

La commande « **modprobe** » avec l'option de suppression (remove), peut également être utilisée pour désactiver un module déjà chargé.

***modprobe -r i2c-dev***

Si vous voulez que le module s charge à l'allumage du **RaspberryP** il est nécessaire d'activer le charge ment permanent du driver (pilote) I2C il faut **modifier le fichier de config ration** qui se trouve dans le répertoir « **/etc/modules** », contenant la list des pilotes à charger au démarrage Sinon, dans le cas contraire nou devons charger le module avec l commande « **modprobe** » à chaqu allumage.

Pour modifier le fichier, utilisons la com mande :

***nano /etc/modules***

et ajoutons une nouvelle ligne dans le fichier de configuration qui contient l'expression suivante (voir la figure 7) :

***i2c-dev***

Appuyons sur « Ctrl+X » et ensuite sur Y pour enregistrer les modifications apportées au fichier et sortir (voir la figure 8).

Maintenant, installons le paquet « i2c-tools » qui fournit un ensemble de fonctions que vous pouvez utiliser avec une ligne de commande pour vérifier le fonctionnement du bus I2C (voir la figure 9) :

***apt-get install i2c-tools***

Ajoutons notre utilisateur « pi » au groupe I2C (figure 10) :

***adduser pi i2c***

Effectuons le redémarrage du **RaspberryPi** pour activer la nouvelle configuration avec la commande :

***reboot***

Après le redémarrage du **RaspberryPi**, nous nous connectons avec Putty ou Kitty, vérifions que le bus I2C est visible sur le convertisseur ADC avec la commande :

***i2cdetect -y 0*** pour **RaspberryPi rev. 1**

ou

***i2cdetect -y 1*** pour **RaspberryPi rev. 2**

Vous devez
similaire à ce
l'adresse « 0x
intégré **MCP2**

Maintenant,
librairie qui
allons utilise
sous la licenc
tries » dans
La meilleure
télécharger e
tion de la ver

Pour installe
mande suiva

***apt-get insta***

puis allons
« home » ave

***cd /home***

créons un d
par exemple

***mkdir LCD***

positionnon
à l'aide de

***cd LCD***

et télécha
Python dév
avec la com

***git clone ht***
***Adafruit-Ra***

Enfin, allo
contient la
de l'écran

***cd Adafrui***

Nous pou
dossier en
même dos
figure 12).

Allons dan
librairie «
l'aide de la

***cd Adafru***

Installons
protocole

Vous devez obtenir un résultat similaire à celui de la figure 11 où l'adresse « 0x20 » identifie le circuit intégré **MCP23017**.

Maintenant, nous allons installer la librairie qui gère l'écran LCD. Nous allons utiliser la librairie, disponible sous la licence BSD de « Adafruit Industries » dans le référentiel « GitHub ». La meilleure façon de procéder est de télécharger et d'utiliser l'outil de gestion de la version du logiciel « git ».

Pour installer « git », utilisons la commande suivante :

```
apt-get install git
```

puis allons dans notre répertoire « home » avec la commande :

```
cd /home
```

créons un dossier pour notre projet, par exemple :

```
mkdir LCD
```

positionnons-nous dans le répertoire à l'aide de la commande :

```
cd LCD
```

et téléchargeons les bibliothèques Python développées par « Adafruit » avec la commande :

```
git clone https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
```

Enfin, allons dans le dossier qui contient la bibliothèque de la gestion de l'écran LCD avec la commande :

```
cd Adafruit-Raspberry-Pi-Python-Cod
```

Nous pouvons voir le contenu du dossier en nous positionnant dans le même dossier avec WinSCP (voir la figure 12).

Allons dans le dossier qui contient la librairie « **Adafruit_CharLCDPlate** » à l'aide de la commande :

```
cd Adafruit_CharLCDPlate
```

Installons la librairie pour la gestion du protocole I2C avec le langage Python :

```
apt-get install python-smbus
```

À ce stade, nous sommes presque prêts à tester le fonctionnement de l'afficheur LCD. Dans la librairie « **Adafruit_CharLCDPlate** » les broches GPA6 et GPA7 sont utilisées pour gérer les couleurs d'un afficheur LCD de type « RVB ».

Dans notre cas, nous avons opté pour un écran LCD monochrome, donc laissez ces broches disponibles. Pour les utiliser, vous devez modifier la

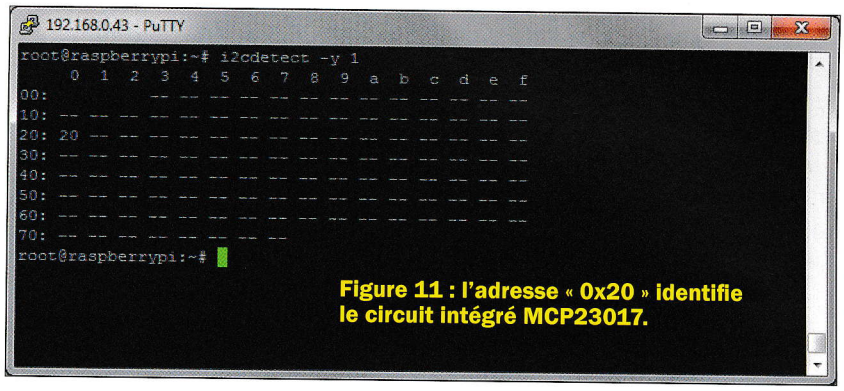



**Figure 11 : l'adresse « 0x20 » identifie le circuit intégré MCP23017.**

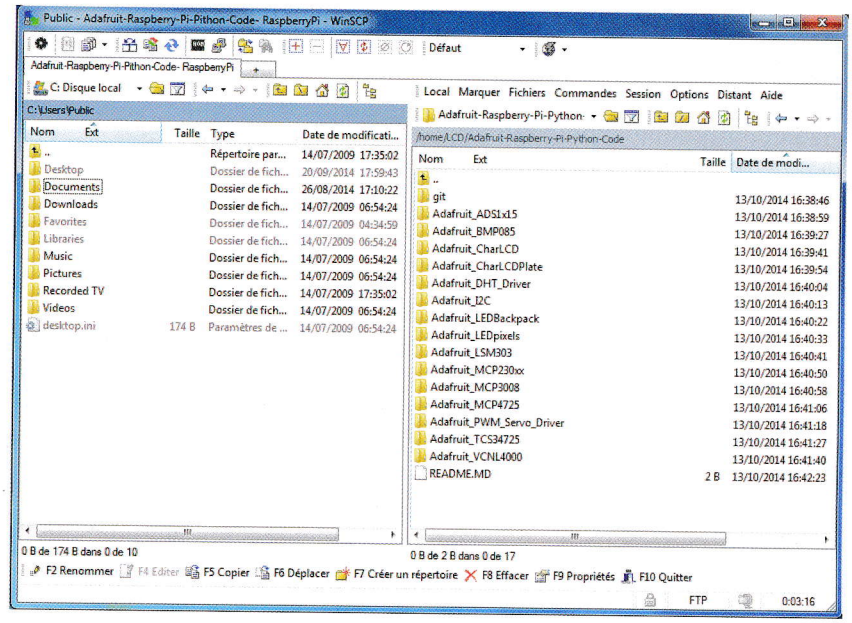
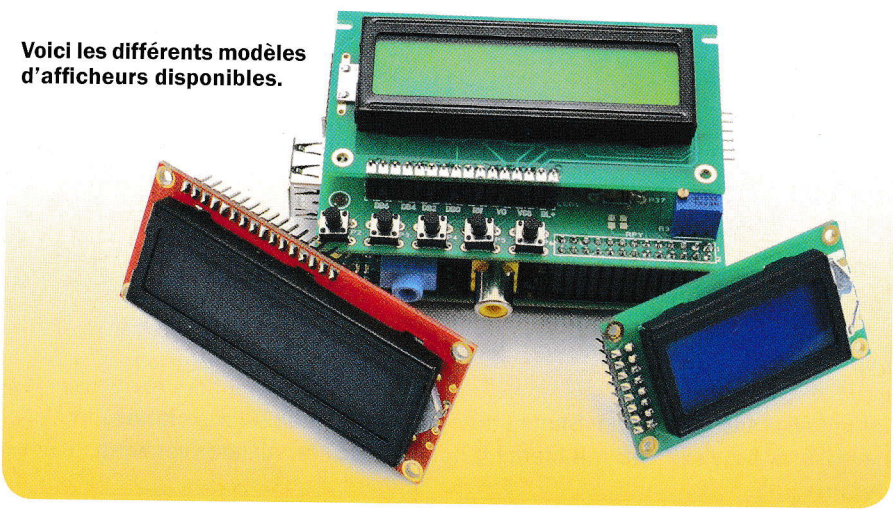



**Figure 12 : affichage du contenu du dossier « home » avec WinSCP.**

**Voici les différents modèles d'afficheurs disponibles.**



**Listing 1**

```
#!/usr/bin/python

from time import sleep
from Adafruit_CharLCDPlate import Adafruit_CharLCDPlate
from subprocess import *

cmd_ip = "ip addr show eth0 | grep inet | awk '{print $2}' | cut -d/ -f1"
cmd_cpu = "mpstat | awk '$11 ~ /[0-9.]+/ { print 100 - $11}'"
cmd_dfh = "df -h | grep /dev/root | awk '{ print $2 }'"
cmd_dfh_p = "df -h | grep /dev/root | awk '{ print $5 }'"
cmd_ps = "ps -ef | wc -l"

def run_cmd(cmd):
    p = Popen(cmd, shell=True, stdout=PIPE)
    output = p.communicate()[0]
    return output

class CPU(object):
  def __init__(self):
    """Init a CPU status object"""
    stat_fd = open('/proc/stat')
    stat_buf = stat_fd.readlines()[0].split()

    self.prev_total = float(stat_buf[1]) + float(stat_buf[2]) + float(stat_buf[3]) + float(stat_buf[4]) + float(stat_buf[5]) +
float(stat_buf[6]) + float(stat_buf[7])
    self.prev_idle = float(stat_buf[4])
    stat_fd.close()

  def usage(self):
    """return the actual usage of cpu (in %)"""
    stat_fd = open('/proc/stat')
    stat_buf = stat_fd.readlines()[0].split()
    total = float(stat_buf[1]) + float(stat_buf[2]) + float(stat_buf[3]) + float(stat_buf[4]) + float(stat_buf[5]) + float(s-
tat_buf[6]) + float(stat_buf[7])
    idle = float(stat_buf[4])
    stat_fd.close()
    diff_idle = idle - self.prev_idle
    diff_total = total - self.prev_total
    usage = 1000.0 * (diff_total - diff_idle) / diff_total
    usage = usage / 10
    usage = round(usage, 1)
    self.prev_total = total
    self.prev_idle = idle
    return usage

  def memUsage():
    free_fd = os.popen('free -b')
    free_buf = free_fd.readlines()[1].split()
    usage = (float(free_buf[2]) / (float(free_buf[1]))) * 100
    usage = round(usage, 1)
    return usage

cpu = CPU()

# Initialize the LCD plate.  Should auto-detect correct I2C bus.  If not,
# pass '0' for early 256 MB Model B boards or '1' for all later versions
```

```
lcd = Adafruit_CharLCDPlate()

# Clear display and show greeting, pause 1 sec
lcd.clear()
lcd.message("  RaspberryPi\n Electronique et Loisirs Magazine")
sleep(5)
lcd.clear()
lcd.message("P1=M P2=IP P3=CPU\nP4=DSK  P5=Proc.")

# Poll buttons, display message & set backlight accordingly
btn = ((lcd.LEFT  , 'Pulsante 1' , lcd.BLUE),
    (lcd.UP    , 'Pulsante 2' , lcd.BLUE),
    (lcd.DOWN  , 'Pulsante 3' , lcd.BLUE),
    (lcd.RIGHT , 'Pulsante 4' , lcd.BLUE),
    (lcd.SELECT, ''           , lcd.ON))
prev = -1

while True:
  pin5 = lcd.buttonpin5()
  pin6 = lcd.buttonpin6()
  pin7 = lcd.buttonpin7()
  pinT = pin5 * 100 + pin6 * 10 + pin7
  pinS = ("000" + str(pinT))[-3:]

  for b in btn:
    if lcd.buttonPressed(b[0]):
      if b is not prev:
        print b[0]
        if b[0] == 0:
          lcd.clear()
          lcd.backlight(lcd.ON)
          ps = run_cmd(cmd_ps)
          lcd.message('N.Processi %s\nP1=Menu  pin %s' % (ps , str(pinS) ))
        elif b[0] == 1:
          lcd.clear()
          lcd.backlight(lcd.ON)
          ipaddr = run_cmd(cmd_ip)
          lcd.message('IP %s \nP1=Menu  pin %s' % ( ipaddr , str(pinS) ))
        elif b[0] == 2:
          lcd.clear()
          lcd.backlight(lcd.ON)
          dfh = run_cmd(cmd_dfh)
          dfh_p = run_cmd(cmd_dfh_p)
          lcd.message('DSK %s USED %s\nP1=Menu  pin %s' % (dfh.strip(), dfh_p.strip() , str(pinS)))
        elif b[0] == 3:
          lcd.clear()
          lcd.backlight(lcd.ON)
          cpu1 = float(cpu.usage())
          lcd.message('CPU %.2f%%\nP1=Menu  pin %s' % (cpu1 , str(pinS)))
        elif b[0] == 4:
          lcd.clear()
          lcd.backlight(lcd.ON)
          lcd.message("P1=M P2=IP P3=CPU\nP4=DSK  P5=Proc.")
        prev = b
      break
```

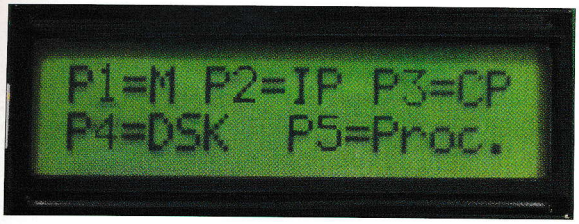



Figure 13.




Figure 14.

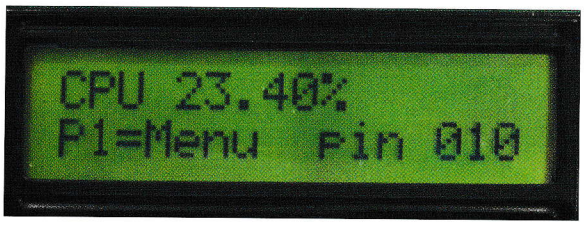



Figure 15.




Figure 16.




Figure 17.

bibliothèque de sorte que les broches soient utilisées comme des entrées par exemple.

Pour comprendre les modifications, reportez-vous à l'encadré dédié au **MCP23017**. La librairie contient le code qui exprime la classe « **Adafruit_CharLCDPlate** » pour la gestion de l'afficheur LCD.

Nous avons modifié dans la classe le réglage initial des registres pour paramétrer les broches GPA5, GPA6 et GPA7 en « entrées » avec les résistances internes de « pull-up » et qui fonctionnent en logique inversé (« 1 » lorsque les broches sont mises à la masse). Vous pouvez télécharger la bibliothèque modifiée à l'adresse :

**www.raspberrypi.electroniquemagazine.com/telechargement.html**

En particulier, nous avons modifié les lignes suivantes :

**ligne 99** *0b11111111, # IODIRA R+G LEDs=outputs, buttons=inputs orig. 0b00111111*

**ligne 101** *0b11111111, # IPOLA Invert polarity on button inputs orig. 0b00111111*

**ligne 112** *0b11111111, # GPPU Enable pull-ups on buttons ori 0b00111111*

À la **ligne 425**, nous avons modifié méthode de rétro-éclairage afin de pas utiliser les broches du port GP pour la gestion des couleurs.

```
def backlight(self, color):
    c = ~color
  #     self.porta = (self.porta
0b00111111) | ((c & 0b011) << 6)
        self.portb = (self.portb
0b11111110) | ((c & 0b100) >> 2)
    # Has to be done as two write
because sequential operation is off.
  #    self.i2c.bus.write_byte_data(
  #          self.i2c.address, self
MCP23017_GPIOA, self.porta)
    self.i2c.bus.write_byte_data(
    self.i2c.address, self.MCP23017
GPIOB, self.portb)
```

Enfin à la **ligne 436**, nous avons ajouté 3 routines pour lire individuellement les broches « avancées ».

```
# Read and return bitmask of pin A5
        def buttonpin5(self):
        return (self.i2c.readU8(self
MCP23017_GPIOA) >> 5) & 1

# Read and return bitmask of pin A6
        def buttonpin6(self):
        return (self.i2c.readU8(self
MCP23017_GPIOA) >> 6) & 1

# Read and return bitmask of pin A7
        def buttonpin7(self):
        return (self.i2c.readU8(self
MCP23017_GPIOA) >> 7) & 1
```

Nous avons préparé un programme de test en modifiant le programme de test inclus dans le dossier de la bibliothèque, nous obtenons le programme visible sur le « **Listing 1** ».

Le programme, à l'aide des boutons poussoirs de la carte, permet d'obtenir des informations sur la performance et l'état du système, résumées sur l'afficheur LCD.

Copiez le programme dans un dossier nommé « **LCDTest.py** », ou téléchargez-le à partir de www.raspberrypi.electroniquemagazine.com. N'oubliez pas d'exécuter la commande :

*modprobe i2c-dev*

Recherchez le dossier « **Adafruit_CharLCDPlate** » et exécutez le programme avec la commande :

*python LCDTest.py*

Vous devriez voir le message d'accueil sur l'afficheur LCD, puis le menu à l'aide des boutons. Si rien ne s'affiche à l'écran vérifiez le contraste, et réglez-le éventuellement à l'aide du potentiomètre R1 jusqu'à ce que vous obteniez un résultat satisfaisant.

Comme vous pouvez le voir, le menu est très simple. La plupart du travail s'effectue selon les modalités prévues par la bibliothèque.

Au démarrage, le programme charge la classe « **sleep** » qui gère les retards d'exécutions, la librairie « **Adafruit_CharLCDPlate** » et la librairie « **subprocess** ».

La première instruction du programme appelle une méthode qui permet de déterminer la « **révision** » (ou version) du **RaspberryPi** en votre possession et régler correctement le bus I2C, 0 ou 1. Ensuite le message d'accueil est envoyé à l'afficheur, et une seconde après apparaît le menu.

Dans la boucle « **While** » sont **lues les valeurs des 3 broches d'entrée** et l'**état des boutons** (pression sur une touche) afin d'exécuter la fonction correspondante et afficher les résultats.

Les valeurs des broches sont représentées dans la partie inférieure droite de l'écran de chaque commande.

Dans un but didactique, nous avons présenté dans ce programme d'exemple les résultats de certaines commandes qui vous permettent de collecter des informations sur l'état et/ou les performances du système. Inutile de vous dire que les mêmes commandes peuvent être tapées dans une fenêtre du terminal pour obtenir les mêmes résultats.

Les commandes sont exécutées par des processus externes au programme en cours qui récupèrent le résultat à la fin du processus (voir le Listing 1).

En particulier, une pression sur **P2 affiche le menu**, le bouton **P3 affiche l'adresse IP** du **RaspberryPi** et l'**état des 3 broches** d'entrées supplémentaires. **P4 affiche l'utilisation du processeur**, **P5 affiche l'espace occupé** d'une carte SD ou d'un disque – soit en « Gb » (gigabit), soit en %.

Le bouton **P6 affiche le nombre de processus en cours d'exécution** sur le **RaspberryPi**.

Pour modifier l'état des broches d'entrée, vous pouvez utiliser un morceau de fil avec, à chaque extrémité, une cosse femelle et en faisant très attention de ne pas faire de court-circuit.

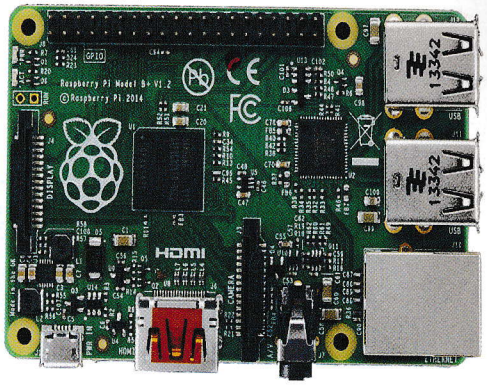
Reliez une des extrémités du fil à la masse et l'autre à l'une des broches d'entrée.

Appuyez sur l'un des boutons P3 à P6, vous devriez voir l'état de la broche en bas à droite (voir les figures 13, 14, 15, 16, 17).

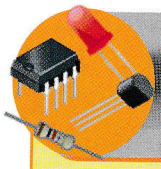
Dans les prochains numéros, nous allons vous décrire d'autres cartes d'extensions dédiées au **RaspberryPi** et approfondir les méthodes de conception et de mise en œuvre des programmes qui respectent l'architecture décrite en début d'article.

**NB** : une **nouvelle version du RaspberryPi nommée type B+** est disponible sur le marché (voir les publicités dans ce numéro). Elle est dotée de 4 ports USB au lieu de 2, d'une nouvelle puce contrôleur USB/Ethernet, d'une consommation réduite ainsi que de fixations optimisées. Le **port GPIO a été amélioré avec un connecteur à 40 broches**, les **26 premières broches étant identiques** à celles du modèle d'origine B. La **carte d'extension A/D 16 bits pour RaspberryPi compatible Arduino ET1041** décrite dans le **numéro 128 d'Electronique et Loisirs Magazine** ainsi que la carte **Afficheur LCD ET1074** décrite dans ce numéro **129 sont parfaitement compatibles avec le nouveau RaspberryPi B+.**

De même les futures cartes qui seront décrites dans les prochains numéros seront compatibles avec ce nouveau modèle. ■



**La nouvelle version du RaspberryPi B+ avec ses 4 ports USB.**



## Comment construire ce montage

Les **typons des circuits imprimés** ainsi que la **librairie sous licence BSD** de « **Adafruit Industries** » sont **téléchargeables gratuitement** aux adresses suivantes :

**www.electroniquemagazine.com**

dans le sommaire détaillé du numéro **129** à l'onglet « **Télécharger** » et/ou

**www.raspberrypi.electroniquemagazine.com/telechargement.html**

La carte d'extension « Afficheur LCD » pour RaspberryPi est **disponible sous forme de kit ET1074**. Le kit est livré avec l'afficheur LCD de 2 lignes de 16 caractères avec rétro-éclairage (référence ACM1602B-FL-YBW).

Avec ce kit vous pouvez aussi utiliser en option deux autres types d'afficheurs LCD, références : 1446-LCD16X2WB et 1446-LCD8X2BN.

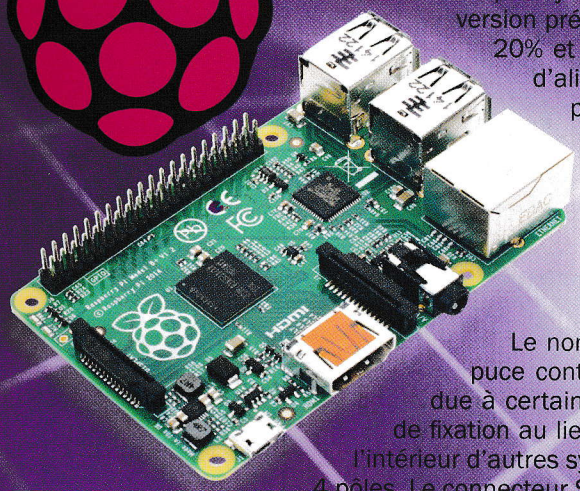
Tous les composants nécessaires à la réalisation du projet décrit dans ces pages ainsi que le **nouveau RaspberryPi B+** sont disponibles auprès de la société COMELEC

**COMELEC**

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

# RASPBERRY PI B+

## Nouveauté

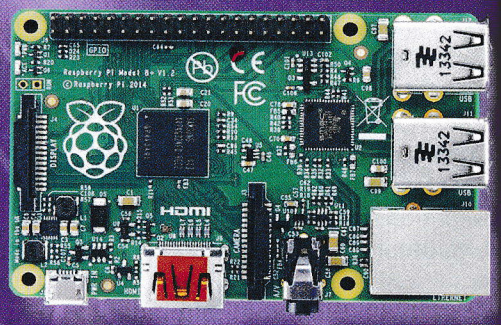


Le RaspberryPi type B+ est la version améliorée du RaspberryPi type B. Par rapport à la version précédente, la consommation a été réduite de manière significative (entre 20% et 30% de moins) grâce à l'utilisation d'un double abaisseur de tension d'alimentation (buck) pour le 3,3 V et le 1,8 V, plus efficace que celui utilisé précédemment.

L'alimentation 5 V est protégée contre les inversions de polarité et les courts-circuits par un fusible de 2 A. Le port GPIO a été amélioré avec plus d'I/O et un connecteur à 40 broches, les 26 premières broches restent identiques à celles du modèle d'origine B, il y a 9 broches GPIO supplémentaires et 2 broches d'identification d'EEPROM.

Le nombre de ports USB a été doublé, passant de 2 à 4 avec une nouvelle puce contrôleur USB/Ethernet. L'amélioration de la qualité du signal audio est due à certaines fonctions liées au DSP. Le circuit imprimé est composé de 4 trous de fixation au lieu de 2, pour améliorer la stabilité de la carte lorsqu'elle est insérée à l'intérieur d'autres systèmes. La sortie vidéo composite est disponible sur un jack 3,5 mm à 4 pôles. Le connecteur SDcard a été remplacé par un connecteur microSD.

**Caractéristiques techniques** :
Processeur 700 MHz Broadcom GPU BCM2835 et FPU ARM1176JZFS avec VideoCore 4 - Le GPU supporte OpenGL ES 2.0 et OpenVG l'accélération matérielle de décodage H.264 1080p 30 images/seconde - Le GPU a une capacité de 1 GPixel/s, 1,5 Gtexel/s ou 24 GFLOPs avec filtrage de la texture et infrastructure DMA - Mémoire 512 Mo de SDRAM - Le système d'exploitation LINUX fonctionne à partir de la carte microSD - Connecteur Ethernet 10/100 BaseT - Prise femelle HDMI - 4 ports USB 2.0 - Connecteur MicroSD - Prise audio jack 3,5 mm et vidéo composite - Connecteur 2 x 20 broches GPIO au pas de 2,54 mm et pour bus série - Connecteur JTAG - Connecteur à 15 broches pour caméra vidéo RaspberryPi Haute Définition (7757731RS) - Alimentation 5 V 2 A via micro prise USB - Dimensions (mm): 85 x 56 x 17.



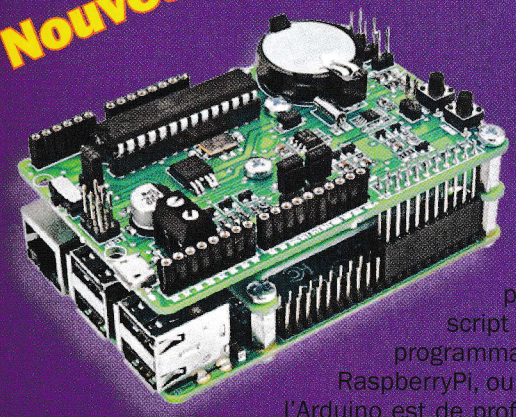
**Réf. 7310-8111284RS... 39,90 €**

# RandA

## carte de développement pour RaspberryPi & Arduino

**Réf. RANDA... 44,60 €**

Nouveauté



RandA (RaspberryPi and Arduino) est une carte de développement pour RaspberryPi. Elle est dotée d'un processeur Atmel 328, d'une horloge temps réel RTC (Real Time Clock), de boutons programmables, de 2 connecteurs d'alimentation 5 V (un micro USB et un bornier) et de connecteurs pour le montage d'une carte. Elle vous permet d'utiliser l'énorme potentiel de la plate-forme LINUX - RaspberryPi et une grande quantité de cartes Arduino. Le lien entre les deux plates-formes est réalisé par le connecteur du RaspberryPi, elle est compatible RaspberryPi types B et B+. Le port série est utilisé pour la programmation de l'Arduino et sert de communication pour le RaspberryPi. La programmation de l'Arduino s'effectue via un script LINUX qui se déclenche grâce au RESET, ce qui permet l'« uploading ». La programmation de l'Arduino peut être faite localement à travers l'IDE installé sur le RaspberryPi, ou à distance via un PC. La meilleure façon de faire collaborer le RaspberryPi et l'Arduino est de profiter des caractéristiques spécifiques des deux. Par exemple, le RaspberryPi pourrait utiliser l'Arduino comme un périphérique configurable dont le programme est choisi sur la base de librairies de « sketch » (programmes spécifiques à Arduino) prédéfinies, et qui sont automatiquement téléchargées dans l'Arduino. Autre exemple, l'Arduino peut fonctionner comme un contrôleur du RaspberryPi en lançant une commande et communiquer avec lui. En pratique, le RaspberryPi deviendrait une extension sophistiquée de l'Arduino, permettant l'accès à des environnements complexes tels que les réseaux ou le multimédia. Le RaspberryPi et l'Arduino peuvent être activés à une date et heure prédéfinies grâce à la RTC. En plus de fournir l'heure au RaspberryPi lorsqu'il n'est pas connecté à Internet, elle est également reliée au circuit d'allumage. L'Arduino peut fonctionner comme un contrôleur du RaspberryPi et peut allumer ce dernier dans le cas d'une condition détectée par les entrées analogiques ou digitales.

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

Photos non contractuelles, publicité valable pour les mois de parution. Prix exprimés en euros TTC. sauf erreurs typographiques ou omissions

# Afficheur LCD pour RaspberryPi

**Nouveauté**



Basée sur le circuit intégré MCP23017 de chez Microchip, cette carte d'extension à afficheur LCD pour le RaspberryPi vous permet de visualiser vos applications sans le connecter en permanence à un moniteur. Le MCP23017 est connecté au port GPIO du RaspberryPi en utilisant seulement les broches nécessaires pour le bus I2C et l'alimentation, toutes les autres broches du RaspberryPi sont reportées. En outre, même si la plupart des E/S du MCP23017 sont utilisées pour la gestion de l'écran LCD et les 5 boutons de commande présents sur la carte, 3 E/S supplémentaires sont disponibles pour une utilisation personnalisée. L'alimentation en 5V est prélevée sur les broches 2 et 6 du port GPIO. Un potentiomètre permet de régler le contraste des différents afficheurs.

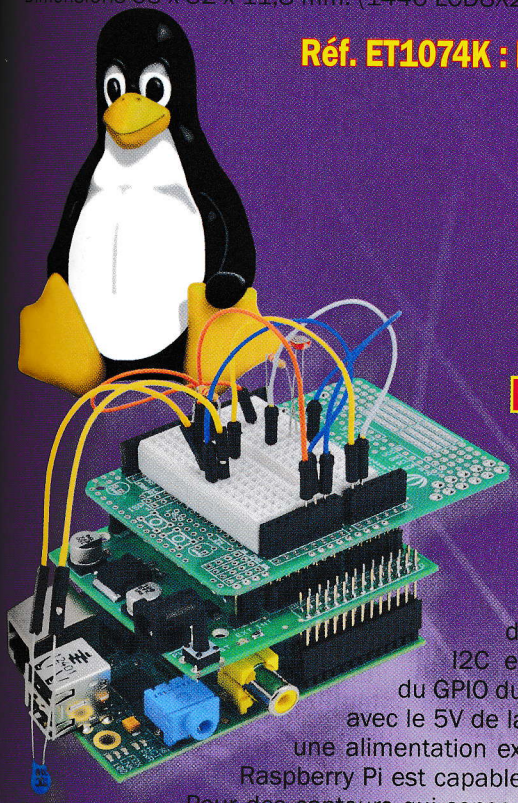
La carte est munie de plusieurs connecteurs pouvant accueillir différents types d'afficheurs LCD et en particulier les modèles suivants qui, tout en ayant une taille différente, ont en commun la même configuration des broches et les mêmes commandes pour la communication : Afficheur LCD 2 lignes de 16 caractères rétroéclairé. Il utilise le contrôleur HD44780 ; alimentation + 5 VDC ; dimensions 84 x 44 x 13 mm. (ACM1602B-FL-YBW). - *Afficheur LCD 2 lignes de 16 caractères blancs sur fond noir rétroéclairé. Il utilise le contrôleur HD44780 ; alimentation + 5 VDC ; dimensions 80 x 36 x 8,40 mm. (1446-LCD16X2WB). - *Afficheur LCD 2 lignes de 8 caractères blancs sur fond bleu rétroéclairé. Il utilise le contrôleur Samsung ST7066U ; alimentation + 5 VDC. Dimensions 58 x 32 x 11,8 mm. (1446-LCD8X2BN).

**Réf. ET1074K : Kit complet avec l'afficheur ACM1602B-FL-YBW... 16,40 €**

***Afficheur 1446-LCD16X2WB... 14,50 €**

***Afficheur 1446-LCD8X2B... 7,60 €**

* afficheurs en option



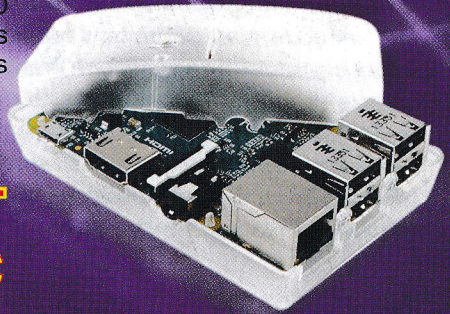
# Carte d'extension A/D 16 bits

## pour RaspberryPi compatible Arduino

**Réf. 1041M... 27,90 €**

Conçue pour être utilisée en conjonction avec le RaspberryPi (non inclus), cette carte permet d'étendre les fonctionnalités du port GPIO avec un convertisseur A/D à 4 canaux (entrées asymétriques ou différentielles). Elle convertit le niveau de 3,3 V à 5V (et vice versa). De plus elle permet d'utiliser certaines cartes disponibles pour l'Arduino à travers les bus USB/I2C et USB/SPI. Elle est dotée d'un connecteur à 26 broches pour la connexion du GPIO du Raspberry Pi, les jumpers « EXT » et « INT » permettent d'alimenter la carte avec le 5V de la broche 2 du connecteur GPIO ou par le régulateur 7805. De cette façon, une alimentation externe peut être évitée quand cela n'est pas nécessaire. La broche 2 du Raspberry Pi est capable de délivrer 500 mA pour la version « rev.1 » et 300 mA pour la « rev.2 ». Pour des capteurs qui consomment plus de courant ou pour la carte à relais qui nécessite une tension plus élevée, une alimentation externe doit être utilisée. N.B : La carte est livrée avec tous les composants CMS montés, tous les autres composants tels que les connecteurs, la fiche d'alimentation, les poussoirs et les cavaliers doivent être soudés par l'utilisateur.

**Caractéristiques techniques** : Étend les fonctionnalités du port GPIO en le dotant d'un convertisseur A/D asymétrique ou différentiel à 4 canaux, directement sur la carte - Alimentation directe de la carte prélevée sur le 5 V du RaspberryPi ou avec une alimentation externe de 5 V à 12 V compatible avec la carte Arduino - Conversion des niveaux de 3,3 V à 5 V et vice versa des I/O digitales, des bus I2C, SPI et du port série, afin de répondre à la fois aux exigences électriques du RaspberryPi, des capteurs et périphériques externes - Possibilité d'utiliser le convertisseur A/D avec des entrées asymétriques ou des entrées différentielles - Connecteur à 26 broches pour la connexion du port GPIO du RaspberryPi - Connecteurs pour la connexion directe d'une carte Arduino et des cartes prévues pour ce dernier - Possibilité de connecter des unités supplémentaires externes USB/I2C et USB/SPI.



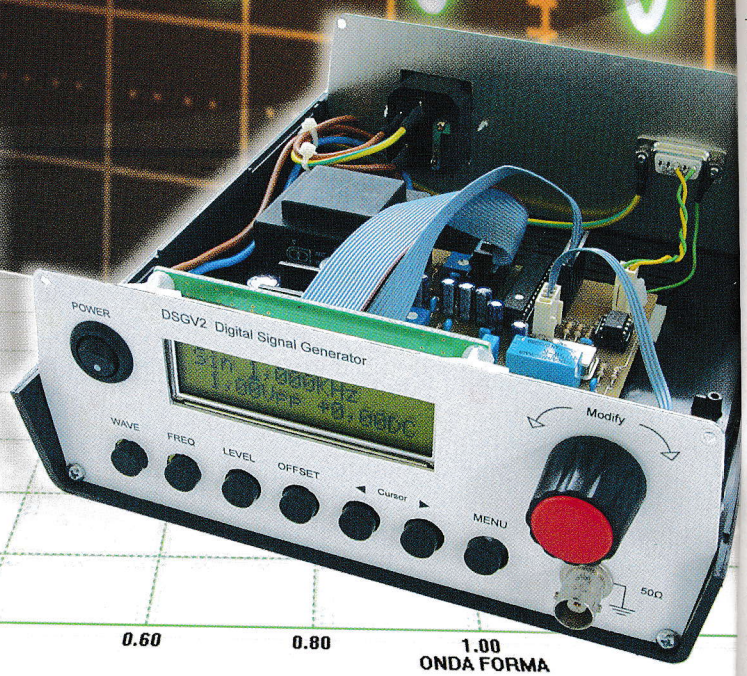
# Boitier transparent pour RaspberryPi B+

**Réf. 6168-BOXRASPBERRYB+ ...13,10 €**

ET758

# Générateur de fonction DDS de 0,1 Hz à 5 Mhz

**Nous continuons notre description d'appareils dédiés au laboratoire. Après avoir publié dans le précédent numéro 128 d'Electronique et Loisirs Magazine une alimentation de laboratoire contrôlée par PIC, nous vous proposons dans cet article un générateur de fonction à synthétiseur numérique DDS qui peut générer un signal sinusoïdal, carré et triangulaire jusqu'à une fréquence de 5 MHz. Il dispose d'un contrôle d'amplitude et d'offset (décalage) pour tous les types de signaux. Il peut être géré par le port série d'un PC via un programme de contrôle simple fonctionnant sous Windows.**







*De Mirco Segatell*

Le générateur de fonction, qui est un instrument présent dans tous les laboratoires d'électronique qui se respectent, est un outil qui permet de produire un signal de référence précis. Il est utile pour le test d'un étage amplificateur, pour l'étalonnage d'un filtre, pour le contrôle du temps de réponse d'un circuit sur les fronts d'une onde carrée.

Les facteurs les plus importants à prendre en considération lors du choix d'un générateur de fonctions sont la fréquence maximale générée, les types des signaux en sortie et les contrôles disponibles (amplitude du signal, offset et symétrie), et enfin la stabilité dans le temps à savoir un signal de sortie stable tout en modifiant les paramètres de réglage.

Cependant, plus la qualité d'un géné rateur est grande et plus son pri devient prohibitif. Un bon générateu de fonction finit par être très coûteu

Dans cet article, nous vous décrivons u générateur de fonction approprié pou une utilisation « amateur » mais ave des critères conçus pour les produit professionnels.

En effet c
un **synth**
signifie «
qui, en pl
d'une gra
fréquence
tous les p
à l'utilisat
classique
composan

Les **para**
générée s
**LCD** et au
**série RS**
fonctionn
DDS.

## Princip

Pour être
grande p
fréquenc
recours
spécifiqu
électroni
formes d
le schém
utilisé, il
chez **Ana**

Le circui
**teur bin**
d'une **ta**
signal si
**seur** nur
synchron

L'accum
**able** rep
24 à 48)
chaque i
eur nume
nous app
**variable**
DDS via
**valeur d**
**pour cha**
la **somm**
**flow** » co
n'est pas

A **chaque**
**les plus**
**partir du**
ces bits
« **index** »
**valeurs**
l'onde sin

En effet ce générateur est basé sur un **synthétiseur numérique** (DDS signifie « Direct Digital Synthesis ») qui, en plus d'offrir les avantages d'une grande précision et stabilité en fréquence, vous permet de contrôler tous les paramètres facilement grâce à l'utilisation d'un logiciel (au lieu des classiques potentiomètres et autres composants passifs).

Les **paramètres** de la forme d'onde générée sont **affichés sur un écran LCD** et aussi à **distance via le port série RS232**. Voyons en détails le fonctionnement de notre générateur DDS.

### Caractéristiques techniques du générateur de fonction DDS

- Forme d'onde :
  - sinusoïdale : de 0,1 Hz à 5 MHz
  - carrée : de 0,1 Hz à 4 MHz
  - triangulaire : de 0,1 Hz à 1 MHz
  - résolution : 4 digits ou 0,1 Hz
  - precision : ± 0,2 %
- Niveau de sortie :
  - impedance : 50 Ω
  - amplitude : de 0 V à 10 Vcc
  - résolution : 3 digits ou 1 mV
  - offset (décalage) : de - 5 V à + 5 V
- Enregistrement des paramètres dans une mémoire non volatile.
- Contrôle complet via l'interface série RS-232 opto-isolée.
- Affichage LCD 2 lignes de 16 caractères rétroéclairé
- Alimentation : 230 VAC 10 VA.

## Principe de fonctionnement

Pour être en mesure d'obtenir une grande précision et une stabilité en fréquence élevée, nous avons eu recours à l'utilisation d'un circuit spécifique DDS qui est un composant électronique dédié à la génération de formes d'ondes. La figure 1 montre le schéma synoptique du circuit DDS utilisé, il s'agit de l'**AD9833BRMZ** de chez **Analog device**.

Le circuit est composé d'un **sommateur binaire**, d'un **accumulateur**, d'une **table** contenant les valeurs du signal sinusoïdal et d'un **convertisseur** numérique/analogique, le tout synchronisé par une horloge.

L'accumulateur contient une **variable** représentée par « **n** » **bits** (de 24 à 48) à laquelle est ajoutée (pour chaque impulsion d'horloge) une valeur numérique fournie à l'entrée (que nous appellerons **M**) qui est aussi une **variable** à « **n** » **bits** transférée au DDS via le port série ou parallèle. La **valeur de l'accumulateur augmente pour chaque impulsion d'horloge**, la **somme continue à l'infini** « **overflow** » comme un « **dépassement** » qui n'est pas pris en compte.

A **chaque impulsion d'horloge**, les **bits les plus significatifs** sont **prélevés à partir du registre** de l'**accumulateur**, ces bits sont ensuite utilisés comme « **index** » pour la **table contenant** les **valeurs** (en décimal) des **niveaux** de l'onde sinusoïdale.

Ces valeurs sont ensuite converties en un signal analogique par le convertisseur D/A (voir la figure 3).

Le signal de sortie est un signal sinusoïdal dont la fréquence vaut :

$$f_{out} = (M \times f_{clock}) / 2^n$$

où « **n** » est le **nombre de bits dont est constitué l'accumulateur**, **M est la variable fournie au DDS** (contenant « n » bits) et $f_{clock}$ **la fréquence d'horloge** servant de référence aux opérations de sommation et de conversion. Etant donné que l'horloge est réalisée à l'aide d'un oscillateur à quartz, la précision et la stabilité de la fréquence sont plus élevées que celles pouvant être obtenues par un procédé analogique, normalement basé sur un oscillateur résistif-capacitif.

**Figure 1 : schéma synoptique du circuit DDS AD9833BRMZ de chez Analog device.**

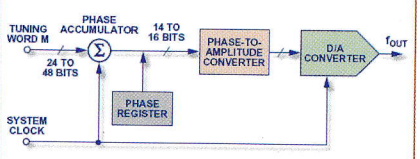
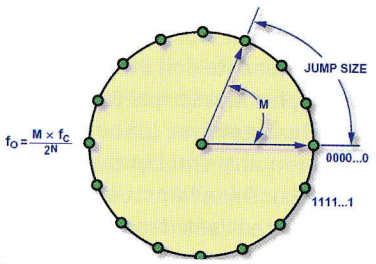



**Figure 2 : représentation de la fonction de sortie du circuit DDS.**




Tout ce qui précède peut sembler assez complexe, nous allons clarifier le concept à travers un exemple simple. Supposons que nous travaillons avec un système sur « 4 bits », notre accumulateur peut prendre toutes les valeurs comprises entre 0 et 15 (il y a 16 valeurs en tout soit $2^4 = 2 \times 2 \times 2 \times 2 = 16$), ainsi que la variable d'entrée M. Nous décidons de prendre la valeur « 3 ». Supposons également que nous avons une table qui associe la valeur de l'amplitude de la sinusoïde à chaque valeur de l'indice qui varie de 0 à 15.

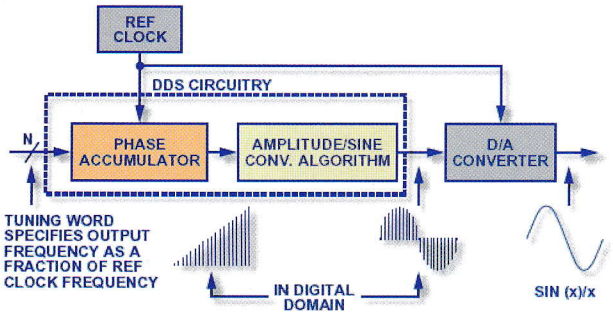
A chaque impulsion d'horloge, l'accumulateur (dont la valeur M est initialisée à 0) est incrémenté de 3 unités. Dans la pratique, c'est comme si le rayon se déplaçait de 3 points sur l'hypothétique cercle comportant 16 points (voir la figure 2). La séquence numérique résultante est la suivante :

**M=0,3,6,9,12,15,2,5,8,11,14,1,4,...**

Chaque valeur est ensuite insérée dans le tableau qui fournit l'amplitude correspondante de la sinusoïde, l'amplitude numérique est transformée en signal analogique par le convertisseur D/A (voir la figure 3).

Le **convertisseur génère une onde sinusoïdale approximative correspondant aux « rectangles »** (voir la figure 4).

Figure 3 : à gauche, la section de synthèse de l'onde sinusoïdale, obtenue par l'algorithme de conversion du circuit DDS.



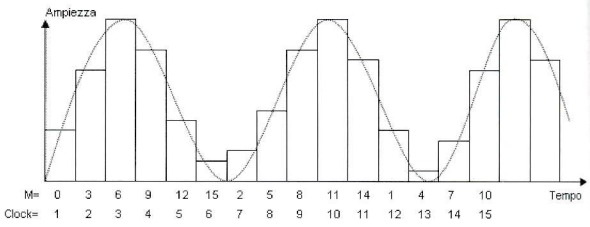




Figure 4 : à droite l'onde sinusoïdale approximative.

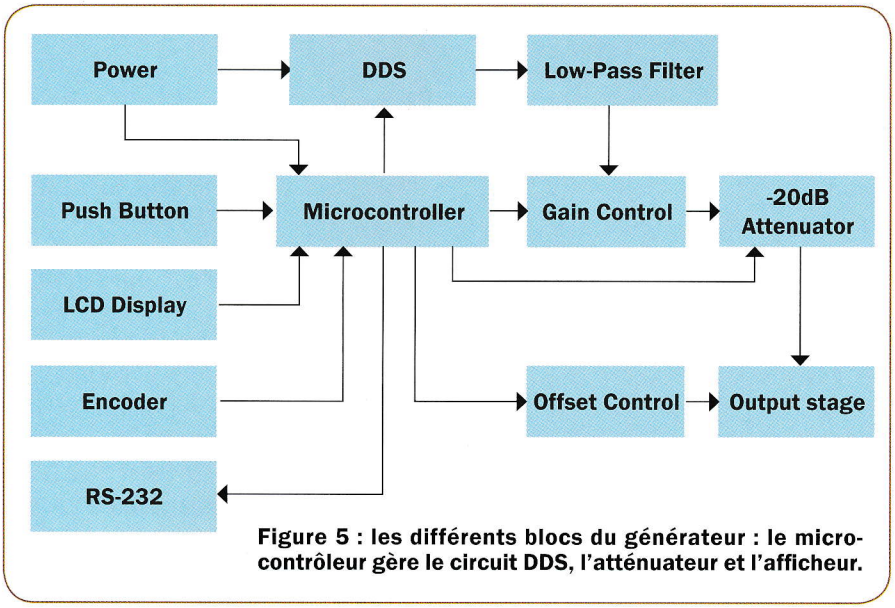



Figure 5 : les différents blocs du générateur : le microcontrôleur gère le circuit DDS, l'atténuateur et l'afficheur.

L'**onde est d'autant plus approximative que la fréquence à générer est élevée**, cela se traduit par une **augmentation de la distorsion harmonique du signal de sortie**. Pour atténuer les harmoniques indésirables qui se trouvent en sortie du circuit DDS, un **filtre passe-bas, dont la fréquence de coupure se situe en dessous de la moitié de la fréquence d'horloge (celle requise par le composant), est utilisé.** Nous obtenons ainsi **une sinusoïde avec une faible distorsion**.

Certains circuits DDS, tels que celui utilisé dans ce projet, peuvent également être programmés pour générer des formes d'ondes **triangulaires** ou **carrées**, mais à des **fréquences inférieures** à la valeur maximale obtenue pour une **onde sinusoïdale**.

Les **générateurs de fonction analogiques génèrent une onde triangulaire qui est ensuite transformée en une onde sinusoïdale** ou **carrée**, ayant la même fréquence par des circuits spéciaux. Or, avec un **générateur DDS la situation est différente** car nous ne pouvons pas **utiliser le filtre passe-bas pour nettoyer un signal triangulaire**.

Ainsi la **fréquence maximale de l'onde triangulaire** est d'environ un **dixième de la plus haute fréquence obtenue** pour une **onde sinusoïdale**.

Pour l'**onde carrée**, la **contrainte** est représentée par la **bande passante** du système, car pour obtenir un bon signal carré à une certaine fréquence, la **bande passante doit être au moins 5 à 10 fois plus élevée que la fréquence maximale à générer** (dans notre cas il faut au moins une bande passante de 50 MHz). Dans la pratique avec les composants utilisés dans notre projet, vous pouvez obtenir une **onde carrée de bonne qualité jusqu'à 1 MHz** et acceptable jusqu'à 4 MHz. Les limites du système sont évidentes pour un appareil dans cette catégorie de prix. Dans les générateurs DDS professionnels, des filtres appropriés éliminent tout problème, cependant le prix reste très élevé pour un usage de type amateur.

## Le schéma électrique

Nous entrons enfin dans les détails de notre projet avec sa représentation en différents blocs (voir la figure 5).

Puisque que nous avons voulu créer un appareil de qualité, avec de bonnes caractéristiques techniques et facilement reproductible par la majorité de nos lecteurs, nous avons évité d'utiliser des circuits complexes et des références de tensions d'une grande précision (chers et difficiles à trouver), en tirant les références des tensions de classiques régulateurs d'alimentation disponibles partout.

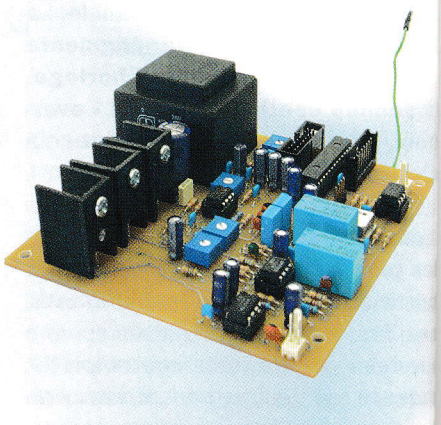
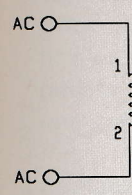
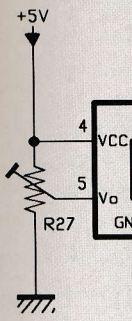


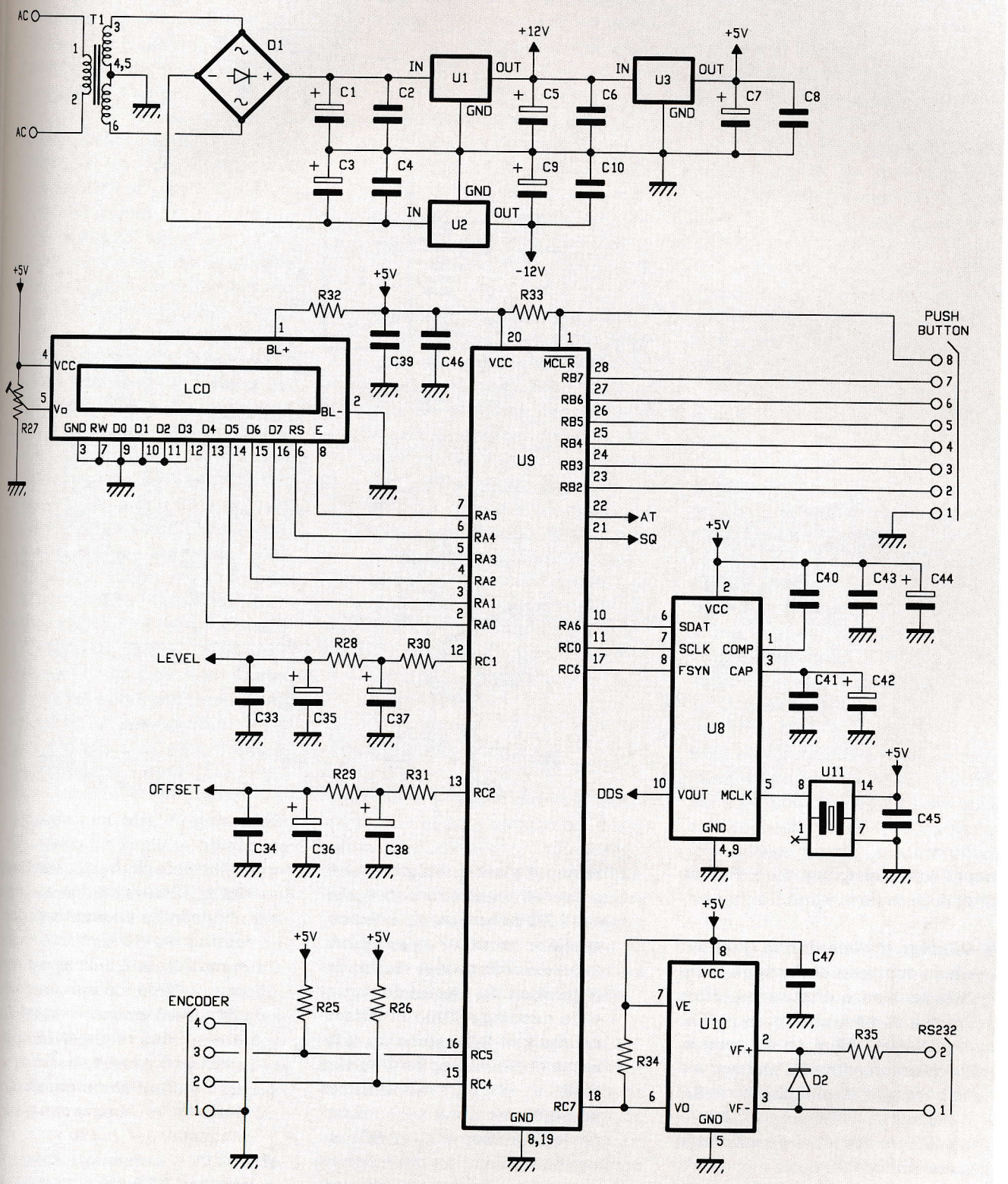
Figure 5a : s




Nous avons
tous les com
ditionnel, à l
qui n'est ma
qu'en versio
fréquences

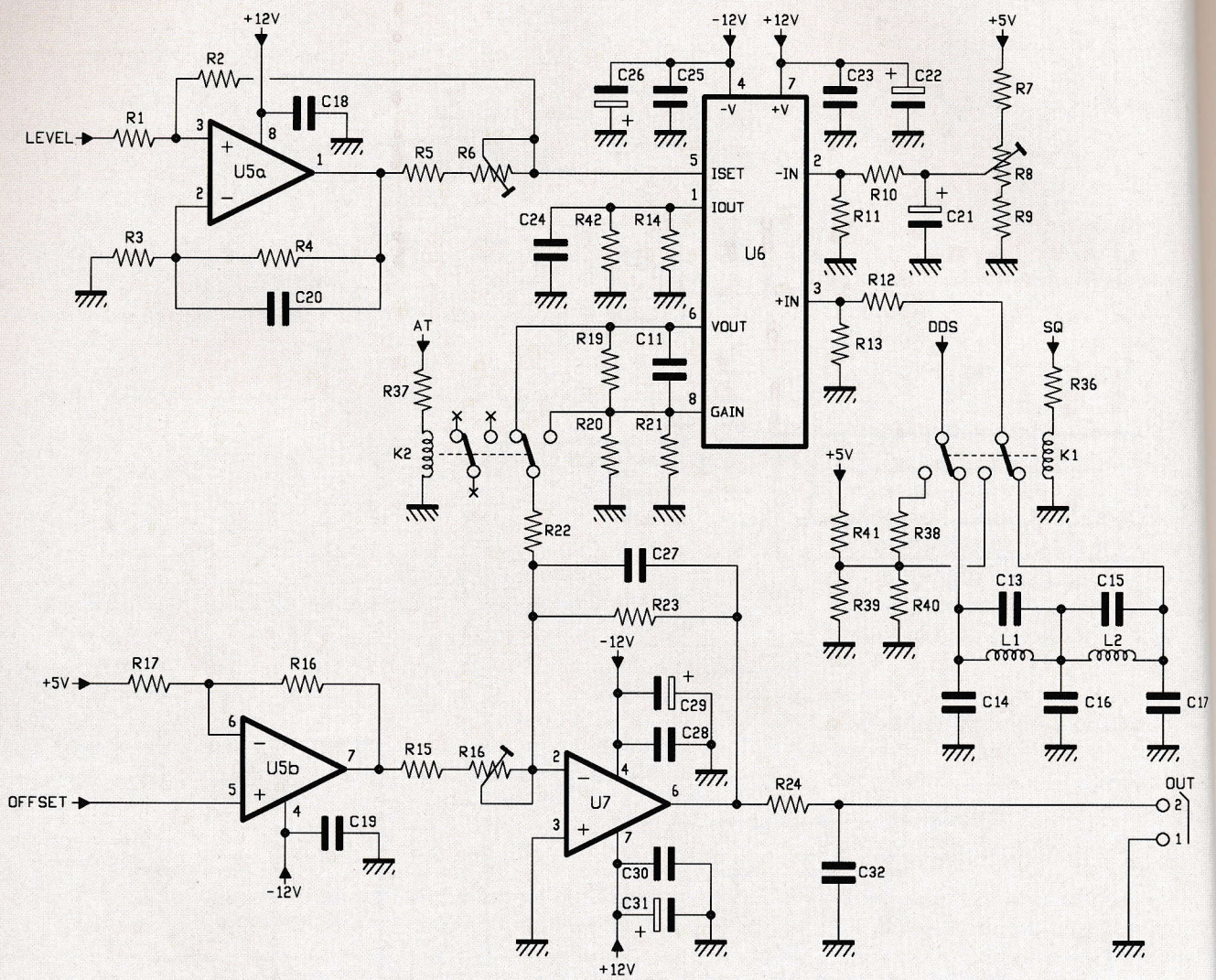
**Figure 5a : schéma électrique de la section numérique du générateur de fonction.**




Nous avons également sélectionné tous les composants au format DIL traditionnel, à l'exception du circuit DDS qui n'est malheureusement disponible qu'en version CMS. Compte tenu des fréquences relativement élevées de 0,1 Hz à 5 MHz et des caractéristiques particulières du projet, tous les composants ont été choisis parmi les plus communs et sont facilement disponibles. Presque toutes les résistances ont une tolérance de 1 %.

En l'absence d'indication, il est possible d'utiliser des résistances de tolérance 5 %. **Préférez les résistances de type MRS25 métal au lieu des classiques en carbone**, vous aurez une meilleure stabilité.

**Figure 5b : schéma électrique de la section analogique du générateur de fonction.**




Nous allons maintenant décrire les différents blocs de la figure 5 :

- ***l'étage d'alimentation (Power)*** : il se compose de **3 régulateurs** de tension qui fournissent les tensions nécessaires de **+ 5 V**, **+ 12 V** et **- 12 V**. La consommation est très faible, l'élément qui consomme le plus est le rétroéclairage de l'afficheur LCD. Celui-ci peut être **désactivé en retirant la résistance R32** ;

- ***l'étage microcontrôleur (Microcontroller)*** : la gestion du système est confiée à un microcontrôleur **PIC18F2550** (U9) de chez Microchip. Sa fonction est de contrôler tous les éléments du circuit et gérer les interfaces que sont l'écran, les boutons, l'interface série et l'encodeur rotatif. Pour le choix des paramètres, nous utilisons **7 touches**, tandis que pour **modifier la valeur** nous faisons appel à un **encodeur rotatif**. La **régulation du niveau du signal** et le **décalage** (**Gain** et **Offset**) nécessitent **2 tensions de référence** obtenues par les **2 sorties PWM** du PIC à travers **2 filtres passe-bas** (2 réseaux RC connectés aux sorties RC1 et RC2 du microcontrôleur). La résolution de cette tension est de **10 bits** soit 1024 valeurs. Le module **UART** (RC7) est utilisé pour **communiquer avec le PC** via le port série **RS232**. Nous n'avons pas utilisé le module USB contenu dans le PIC car il est difficile de réaliser une connexion isolée, ce qui n'est pas le cas avec l'interface série grâce à l'optocoupleur U10. Les **2 relais K1** et **K2** de la section analogique sont **pilotés directement par les sorties du PIC** sans interfacer de transistor, car leur bobine présente une impédance élevée et donc un courant limité. Si vous devez **utiliser des relais différents de ceux préconisés, assurez-vous qu'ils aient les mêmes caractéristiques, sinon vous pourriez endommager le PIC** ;

- ***l'étage DDS*** : le circuit DDS utilisé dans notre montage est un **AD9833BRMZ** (U8) qui peut générer des signaux **sinus**, **carrés** et **triangulaires** et peut être géré par le PIC via l'interface série avec seulement 3 lignes (RA6, RC0, RC6). Le signal de sortie est seulement de **500 mVcc** et **doit donc**

**être ampl**
priée en lu
réglable av
L'horloge
tir d'un os
qui assure
une grand

- ***l'étage fil***
***Filter)*** : le
circuit DD
relais K1
des sign
triangula
(by-pass)
carrés. L
conçu pou
type de fil
« **filtre el**
**caractéri**
tant en «
« **bande a**
tré qu'il
ce sens
donné, **n**
**plus rai**
**tiques**. L
lisé à l'a
**L1** et **L2**,
**sible** à la
alors qu'
**shev** » es
aussi **m**
des com
compose

- ***l'étage***
***Control***
**signal** à
est un a
« **large b**
du gain
traverse
teur opé
comme
**sion/co**
rant de
proporti
de référ

- ***l'étage a***
***nuator)***
gain de
signal d
**niveau**
**un rapp**
signal de
crête à c
rant un
rapport

être **amplifié** de manière appropriée en lui appliquant un « **offset** » réglable avant d'atteindre la sortie. L'horloge U8 est cadencée à partir d'un oscillateur à quartz (U11) qui assure une grande précision et une grande stabilité ;

- ***l'étage filtre passe-bas (Low Pass Filter)*** : le filtre passe-bas pour le circuit DDS est activé à l'aide du relais K1 lorsque le circuit génère des signaux sinusoïdaux ou triangulaires, et il est contourné (by-pass) dans le cas des signaux carrés. Le circuit imprimé a été conçu pour un filtre de « **Cauer** ». Ce type de filtre est également appelé « **filtre elliptique** » sa **réponse est caractérisée par une ondulation** tant en « **bande passante** » qu'en « **bande atténuée** ». Cauer a montré qu'ils étaient optimaux en ce sens qu'**aucun filtre**, à ordre donné, **ne présente une coupure plus raide que les filtres elliptiques**. Le filtre elliptique est réalisé à l'aide du réseau **C13**, **C17**, **L1** et **L2**, il est cependant **très sensible** à la valeur des composants, alors qu'un filtre de type « **Chebyshev** » est **moins performant** mais aussi **moins sensible** à la valeur des composants passifs qui le composent ;

- ***l'étage contrôle du gain (Gain Control)*** : il règle le **niveau du signal** à travers le circuit **U6** qui est un amplificateur opérationnel « **large bande** », et dont le contrôle du gain dépend du courant qui traverse la broche 5. L'amplificateur opérationnel **U5a** fonctionne comme un **convertisseur** de « **tension/courant** » et régule le courant de la broche 5 du circuit U6 proportionnellement à la tension de référence fournie par le PIC ;

- ***l'étage atténuateur (- 20 dB Attenuator)*** : il effectue le réglage du gain de la tension de sortie du signal qui est maintenue à un **niveau très élevé pour assurer un rapport signal/bruit élevé**. Le signal de sortie est inférieur à **1 V** crête à crête, il est obtenu en insérant un diviseur de tension d'un rapport 1/10 (R19 à R21) ;

- ***l'étage Offset (Offset Control)*** : il permet de **superposer une tension continue au signal alternatif de sortie**. Cette opération est effectuée par l'amplificateur opérationnel **U5b** qui fournit un courant continu à l'entrée inverseuse de l'amplificateur opérationnel **U7**, **courant qui est proportionnel à la tension de référence** générée par le **PIC**. Cette composante continue en courant est reconstituée en une tension superposée au signal de sortie alternatif ;

- ***l'étage de sortie (Output stage)*** : il est constitué d'un amplificateur opérationnel **U7** (**LM6181**) ayant une **bande passante élevée et capable de fournir des courants élevés**. Ce n'est pas un composant critique, il peut être remplacé par des amplificateurs similaires, à condition qu'ils aient au moins une **bande passante de 50 MHz** avec un courant de sortie d'au moins **50 mA** et qu'ils soient compatibles broche à broche. Dans notre projet, nous avons obtenu d'excellents résultats avec les modèles **LM6181** et **LM7171**.

## Réalisation pratique

Les typons du circuit imprimé double face sont, comme d'habitude, disponibles gratuitement en téléchargement à l'adresse **www.electroniquemagazine.com dans le sommaire détaillé du numéro 129 à l'onglet « Télécharger »**. Le seul composant qui nécessite une **attention particulière lors du montage est le circuit DDS**.

Il se présente sous la forme d'un boitier CMS de type MSOP de très petites dimensions (3 x 3 mm). Il est doté de 10 broches à souder sur autant de pistes d'une largeur de 0,3 mm, ce qui signifie que les soudures doivent être effectuées avec soin et précision.

Pour le soudage du circuit DDS, il est nécessaire de disposer d'un fer à souder de faible puissance (20 W) avec une pointe de **0,5 mm de diamètre** et de l'étain fin pour composants CMS. Il vous faudra aussi une lampe « loupe » avec un bon éclairage et une pince à épiler pour manipuler le composant avec beaucoup d'attention, de calme et de patience. Ce composant, cependant, doit être soudé en dernier.

Les autres composants doivent être soudés selon la procédure classique, en premier commencez par les composants à bas profil (résistances, diodes, condensateurs non polarisés, supports des circuits intégrés) et continuez par les plus grands à savoir les condensateurs polarisés en respectant la polarité, les régulateurs (faites attention à l'orientation), soudez en dernier le transformateur car il est le plus lourd. Aidez-vous avec le plan de câblage et les images.

Afin de faciliter l'installation dans un boitier, nous avons prévu des connecteurs pour le clavier, l'afficheur LCD et l'encodeur. La figure 10 montre le prototype de notre générateur une fois assemblé dans le boitier (dans notre cas, nous avons utilisé le modèle **AUS22.9** de **Teko**). Les boutons peuvent être soudés sur une plaque à trou fixée derrière la face avant à l'aide d'entretoises. Le circuit principal est fixé sur le fond du boitier à l'aide aussi d'entretoises.

Enfin, l'afficheur LCD est relié au circuit imprimé par l'intermédiaire d'un câble plat dont l'une des extrémités est munie d'un connecteur à 16 broches en correspondance directe avec le connecteur de l'afficheur (la broche 1 du LCD avec la broche 1 du circuit, et ainsi de suite).

La face avant est imprimée sur du papier photo, découpée puis collée sur la façade en aluminium du boitier correctement percée. A l'arrière du boitier vous devez découper 2 encoches correspondant à l'embase secteur mâle 3 pôles et au connecteur de port série.

Connectez les boutons, l'afficheur et l'encodeur ainsi que les connexions de l'alimentation 230 VAC, faites en sorte que la prise de terre de l'embase soit reliée au panneau arrière du boitier et à la masse du circuit imprimé.

Les condensateurs **C24** et **C11 peuvent être ôtés** de façon à obtenir une **bande passante maximale** et par conséquent un meilleur signal carré.

## Plan de montage

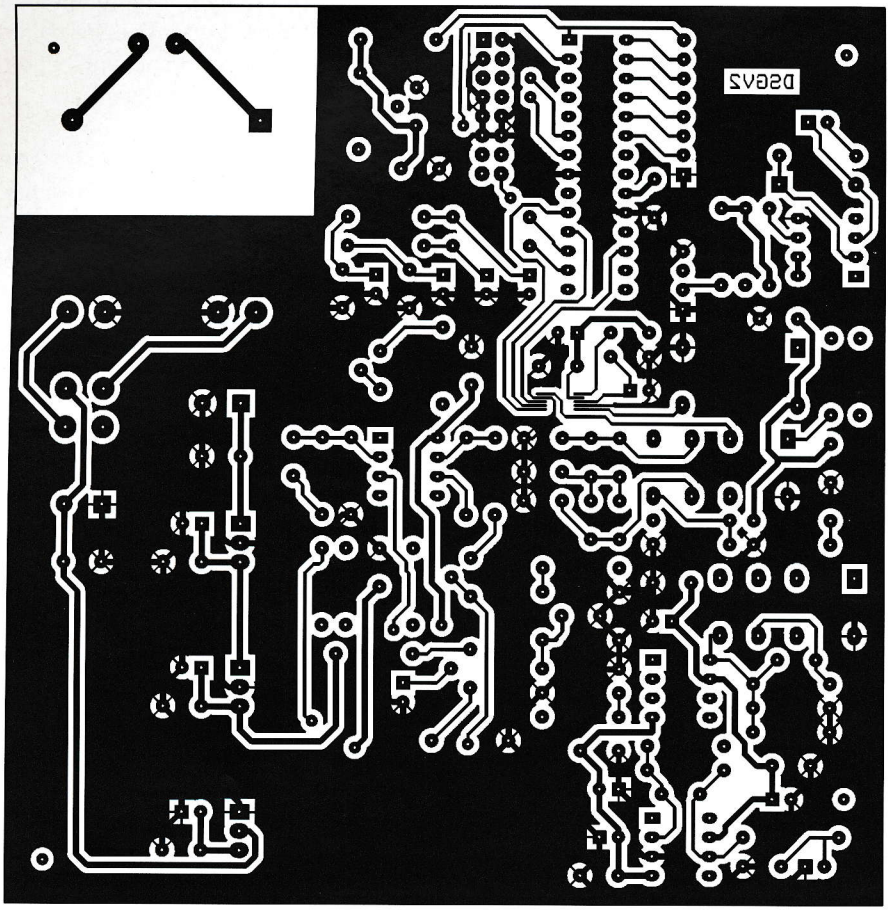
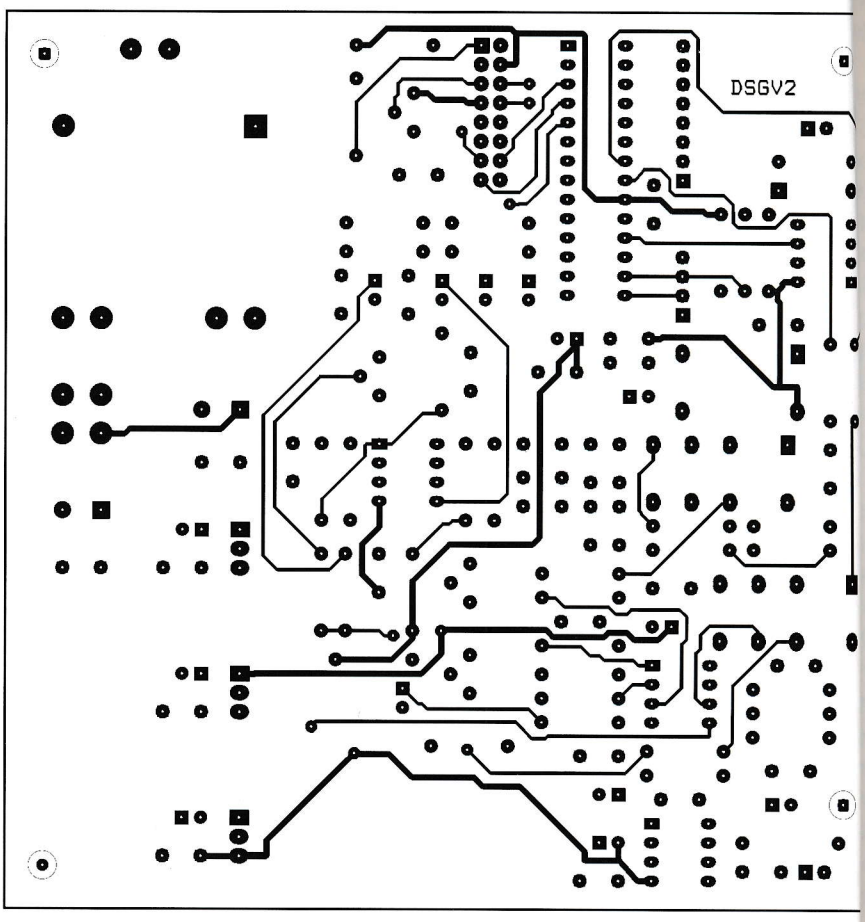



Figure 5c : circuit imprimé à l'échel
1 : 1 côté soudures

Figure 5d : circuit imprimé à l'échell
1 : 1 côté composants du générateu
de fonction.

Sur le schéma électrique de la section analogique (figure 5b), vous pouvez constater la présence des condensateurs **C13** et **C15** qui permettent de réaliser un filtre de type « **Cauer** », dont la pente est très élevée dans la bande d'atténuation. **Dans la version définitive, nous avons décidé de ne pas monter C13** et **C15**, nous nous retrouvons ainsi avec un **filtre de type « Chebyshev »** moins critique dans le choix des composants bien que moins performant.




## Les essais

Vérifiez soigneusement que toutes les soudures soient correctement effectuées, et qu'il n'y ai pas de courts-circuits et de « cheveux d'anges » entre les broches du circuit DDS. Vérifiez si les valeurs et, éventuellement, l'orientation des composants sont correctes.

**Avant de souder le circuit DDS**, vérifiez que **chaque tension d'alimentation arrive bien aux broches correspondantes des supports des circuits**

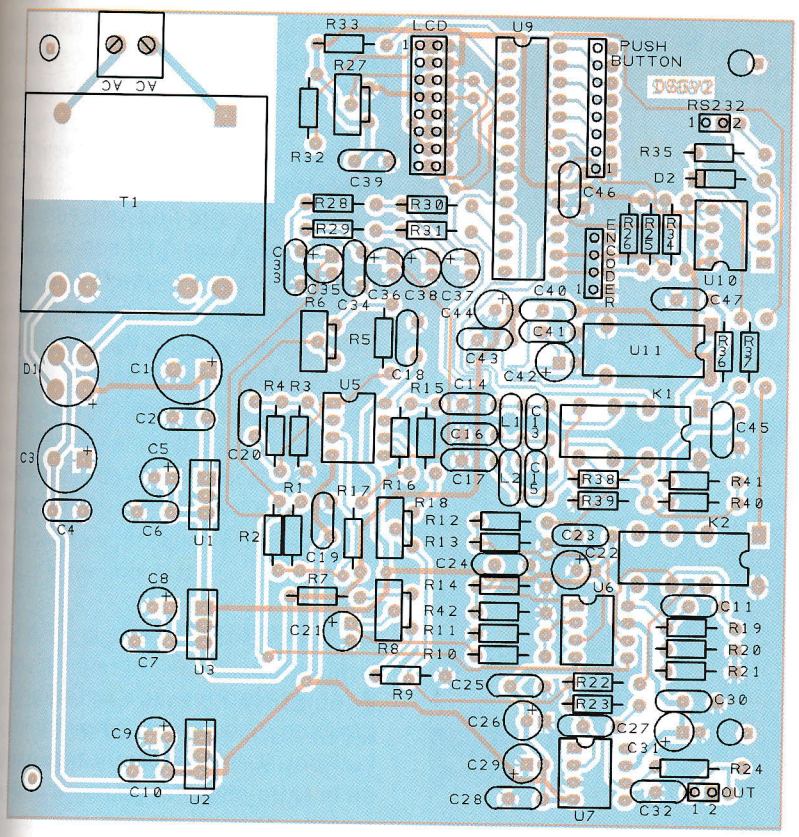
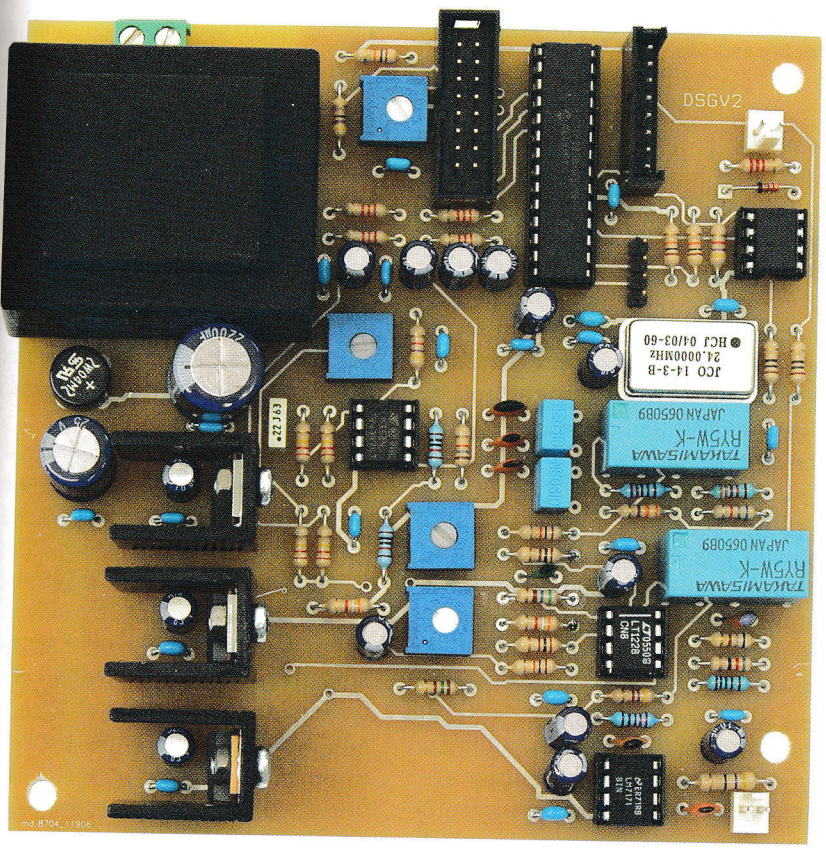



**Figure 5e : schéma d'implantation des composants du générateur de fonction.**

**Figure 5f : photo de l'un des prototypes du générateur de fonction.**




## Liste des composants du générateur DDS

R1..... 220 kΩ 1%
R2..... 220 kΩ 1%
R3..... 220 kΩ 1%
R4..... 220 kΩ 1%
R5..... 4,7 kΩ
R6..... Trimmer Cermet 500 Ω
R7..... 3,9 kΩ
R8..... Trimmer Cermet 100 Ω
R9..... 150 Ω
R10.... 1 kΩ 1%
R11.... 100 Ω 1%
R12.... 1 kΩ 1%
R13.... 100 Ω 1%
R14.... 560 Ω 1%
R15.... 1,8 kΩ
R16.... 10 kΩ 1%
R17.... 10 kΩ 1%
R18.... Trimmer Cermet 500 Ω
R19.... 680 Ω 1%
R20.... 82 Ω 1%
R21.... 1 kΩ 1%
R22.... 470 Ω 1%
R23.... 2,2 kΩ 1%
R24.... 47 Ω 1/2W
R25.... 10 kΩ
R26.... 10 kΩ
R27.... Trimmer Cermet 10 kΩ
R28.... 2,2 kΩ
R29.... 2,2 kΩ
R30.... 2,2 kΩ
R31.... 2,2 kΩ
R32.... 47 Ω
R33.... 10 kΩ
R34.... 4,7 kΩ
R35.... 2,2 kΩ
R36.... 100 Ω
R37.... 100 Ω
R38.... 2,2 kΩ 1%
R39.... 330 Ω 1%
R40.... 6,8 kΩ 1%
R41.... 33 kΩ 1%
R42.... 3,9 kΩ 1%

C1..... 2200 µF / 25 V électrolytique
C2..... 100 nF polyester
C3..... 1000 µF / 25 V électrolytique
C4..... 100 nF polyester
C5..... 10 µF / 25 V électrolytique
C6..... 100 nF polyester
C7..... 10 µF / 25 V électrolytique
C8..... 100 nF polyester
C9..... 10 µF / 25 V électrolytique
C10.... 100 nF polyester
C11.... 4,7 pF céramique
C13.... voir texte *
C14.... 47 pF céramique
C15.... voir texte *
C16.... 33 pF céramique
C17.... 10 pF céramique

C18.... 100 nF polyester
C19.... 100 nF polyester
C20.... 220 nF multicouche
C21.... 10 µF / 25 V électrolytique
C22.... 10 µF / 25 V électrolytique
C23.... 100 nF polyester
C24.... 15 pF céramique
C25.... 100 nF polyester
C26.... 10 µF / 25 V électrolytique
C27.... 2,2 pF céramique
C28.... 100 nF polyester
C29.... 10 µF / 25 V électrolytique
C30.... 100 nF polyester
C31.... 10 µF / 25 V électrolytique
C32.... 10 pF céramique
C33.... 100 nF polyester
C34.... 100 nF polyester
C35.... 4,7 µF / 25 V électrolytique
C36.... 4,7 µF / 25 V électrolytique
C37.... 4,7 µF / 25 V électrolytique
C38.... 4,7 µF / 25 V électrolytique
C39.... 100 nF polyester
C40.... 10 nF multicouche
C41.... 100 nF polyester
C42.... 10 µF / 25 V électrolytique
C43.... 100 nF polyester
C44.... 10 µF / 25 V électrolytique
C45.... 100 nF polyester
C46.... 100 nF polyester
C47.... 100 nF polyester

D1..... Pont de diode 100 V / 1 A
D2..... 1N4148
L1 ..... Inductance 10 µH
L2 ..... Inductance 10 µH
U1..... 7812
U2..... 7912
U3..... 7805
U5..... LM358
U6..... LT1228CN8
U7..... LM6181
U8..... AD9833BRMZ
U9..... PIC18F2550-I/SP
U10.... 6N137
T1 ..... Transformateur 12 V + 12 V / 220 VAC 3VA résine
K1* ... Relais TAKAMISAWA RY5W-K 2 contacts 5 VDC 167 Ω
K2* ... Relais TAKAMISAWA RY5W-K 2 contacts 5 VDC 167 Ω

Divers

Bornier 2 pôles
Barette mâle 2 pôles x 2
Barette femelle 2 pôles x 2
Barette mâle 4 pôles
Barette femelle 4 pôles
Barette mâle 8 pôles
Barette femelle 8 pôles
Connecteur HE-10 mâle 2 x 8 poles pour ci
Connecteur HE-10 femelle 2 x 8 poles à insertion
Support ci 2 x 4 broches ( 4 pièces)
Support ci 2 x 14 broches ( 1 pièce)
Afficheur LCD 2 x 16 (CDL4162)
Fusible 315 mA
Interrupteur 230VAC / 1A
Poussoirs (7 pièces)
Encodeur rotatif 24 à 30 pas
Boitier TEKO AUS22
Embase secteur mâle avec porte fusible

* voir le site : www.weclonline.com

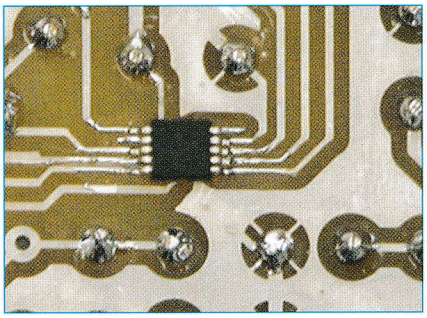


**Figure 5g : Le montage du générateur est assez simple, le seul composant qui a besoin d'attention est le circuit DDS au format CMS qui est monté « côté soudure » du circuit et visible sur l'image. Faites attention à l'orientation de la broche 1 qui est indiquée par le point rouge. Le câblage des connecteurs du LCD et de l'encodeur dépendent des modèles utilisés.**

**intégrés**, **ne mettez pas en place les circuits dans les supports** pour effectuer ce test. Une fois que vous avez vérifié que les tensions étaient correctes, éteignez le circuit et attendez quelques minutes pour que les condensateurs se déchargent, puis **insérez les circuits intégrés dans leur support respectif en faisant attention à l'orientation** (la broche 1 est indiquée soit par un point soit un repère en « U »).

N'oubliez pas de **programmer le PIC**, le fichier « **MF758.hex** » se trouve à l'adresse **www.electroniquemagazine.com** dans le sommaire détaillé du **numéro 129** à l'onglet « **Télécharger** ».

Connectez l'écran LCD, les boutons, l'encodeur puis allumez l'appareil. L'**afficheur s'allume pendant 2 secondes et indique le nom du projet et la version du firmware**, puis pas à l'écran de **fonctionnement**. À stade, vous pouvez interagir avec l boutons et l'encodeur rotatif.

Réglez le **contraste de l'afficheur LC** en ajustant le trimmer **R27**. Contrôl le sens d'insertion du connecteur d l'encodeur : pour qu'il soit correct faut que lorsque vous effectuez un rotation vers la droite, les paramètre affichés augmentent. Vous êtes mai tenant prêt à régler le générateur DD et à procéder à un test complet.

Sachez que vous devez avoir un osc loscope pour afficher la forme d l'onde.

## L'étalonnage

Pour étalonner correctement l générateur, vous devez savoir commen l'utiliser. Il est également **nécessair de le faire chauffer de sorte qu'il s stabilise thermiquement**. Une fois qu vous avez allumé le générateur, lais sez-le sous tension pendant un quar d'heure, vous pouvez pendant ce temps lire la partie « utilisation pratique » (un peu plus loin) de cet article qui vous sera utile pour procéder à l'étalonnage

Avez-vous lu le chapitre « utilisation pratique » ? L'instrument est-il chaud ?

Et bien, gardez les réglages initiaux de la fréquence et de la tension et **vérifiez à l'aide d'un oscilloscope le signal présent sur la broche 6 du circuit U6**. Il doit être parfaitement **centré** avec le « **zéro** » c'est à dire qu'il doit avoir une tension moyenne égale à 0. Ajustez le trimmer (potentiomètre) **R8** pour obtenir cette condition.

Maintenant visualisez le signal de sortie (bornier OUT) et, avec le bouton « **MENU** », allez à la section « **calibration** » (voir la figure 11) pour afficher une valeur comprise entre 500 et 520, valeur qui peut être affinée par l'encodeur rotatif. **Assurez-vous que le signal à l'oscilloscope soit parfaitement centré avec l'axe du « 0 V »** qui correspond à une valeur moyenne égale à 0. Ceci assure que l'offset est nul lorsque l'on règle la valeur de l'offset à zéro.

Quittez la phas
**appuyant de n**
« **MENU** ». Le
**quement enreg**
**non volatile**.

Ensuite réglez
utilisant le trim
la **valeur de l'a**
l'écran LCD co
**réelle** sur l'osci
que l'**oscillosc**
**crête à crête d**
**sortie**. À ce sta
**set différente** e
que le signal de
port au « 0 » à
du générateur.
ajustez le trimm

Pour le réglag
vez également
**calibré sur un**
appliquer ses b
varier l'offset
descende jusq

Ces phases d
ralement effec
de 1 kHz, ma
ces étapes à d
amplitudes af
étalonnage po

Sachez auss
tion prolongé
période d'inac
signal peuven
celles mesuré
d'étalonnage.
des valeurs d
du temps et d
de stockage (

## Utilisation

L'écran princi
figure 6) mon
de la forme d
mètres qui so
des boutons e

Le **paramètre**
**cation** » est to
tez-vous à la
face avant d
vous appuyez
le **caractère**
appelé « **und**

Quittez la phase de « calibration » en **appuyant de nouveau** sur le bouton « **MENU** ». Le **réglage est automatiquement enregistré dans la mémoire non volatile**.

Ensuite réglez le **niveau de sortie** en utilisant le trimmer **R6** de sorte que la **valeur de l'amplitude affichée** sur l'écran LCD corresponde à la **valeur réelle** sur l'oscilloscope. Rappelez-vous que l'**oscilloscope visualise la valeur crête à crête de la forme du signal de sortie**. À ce stade, fixez une valeur d'**offset différente** et vérifiez à l'oscilloscope que le signal de sortie se décale par rapport au « 0 » à une valeur égale à celle du générateur. Si cela n'est pas le cas, ajustez le trimmer **R18**.

Pour le réglage de l'offset, vous pouvez également utiliser un **multimètre calibré sur une tension continue** et appliquer ses broches à la sortie, faites varier l'offset et vérifiez que le signal descende jusqu'à 0 V (offset nul).

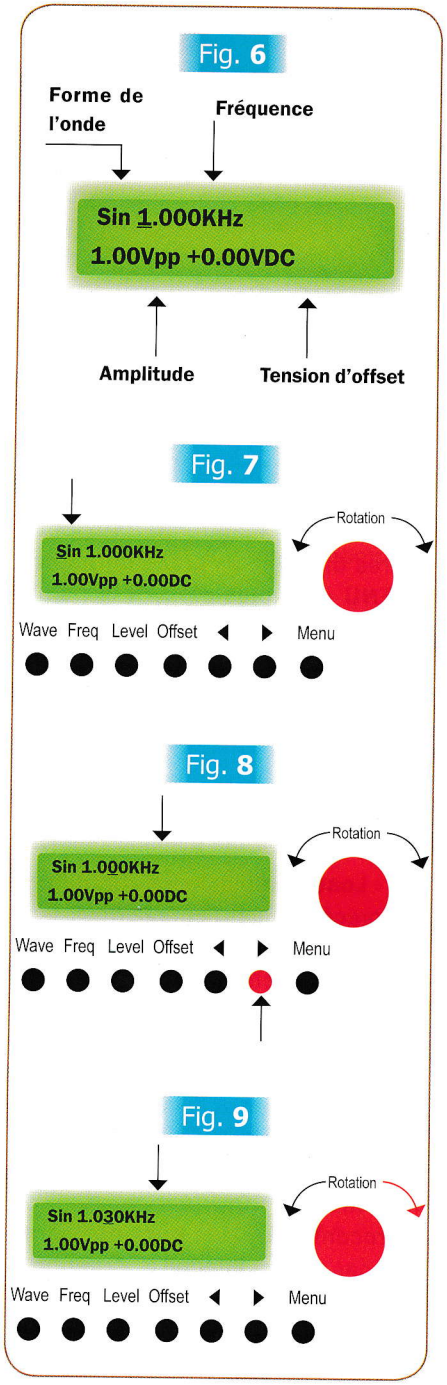
Ces phases d'étalonnage sont généralement effectuées à une fréquence de 1 kHz, mais il est bon de répéter ces étapes à différentes fréquences et amplitudes afin d'obtenir le meilleur étalonnage possible.

Sachez aussi qu'après une utilisation prolongée ou après une longue période d'inactivité, les paramètres du signal peuvent différer légèrement de celles mesurées au cours de la phase d'étalonnage. Cela est dû à la dérive des valeurs des composants au cours du temps et des différentes conditions de stockage (dérive en température).

## Utilisation pratique

L'écran principal de l'afficheur (voir la figure 6) montre tous les paramètres de la forme du signal en sortie, paramètres qui sont sélectionnés à l'aide des boutons et modifiés par l'encodeur.

Le **paramètre « en cours de modification »** est toujours **souligné**. Reportez-vous à la figure 7 qui montre la face avant du générateur, lorsque vous appuyez sur le bouton « **Wave** », le **caractère de soulignement** (aussi appelé « **underscore** » ou « _ ») **se déplace sous le premier caractère** indiquant la forme de l'onde en sortie (ici nous avons « **S** »).




En appuyant sur le bouton « **Offset** », l'« **underscore** » **se déplace sous le 1er chiffre à gauche** de la valeur de l'offset (+ **0**.00 DC).

La même chose se produit avec les touches « **Freq** » et « **Level** ». Le paramètre est modifié lorsque vous tournez l'encodeur rotatif. Relativement aux touches « **Freq** », « **Level** » et « **Offset** » l'excursion des valeurs est grande, par exemple l'excursion de la fréquence varie de 0,1 Hz à 5 MHz. Pour **accélérer la variation du paramètre**, choisissez d'abord le chiffre à modifier à l'aide des touches « ◄ » ou « ► », et ensuite

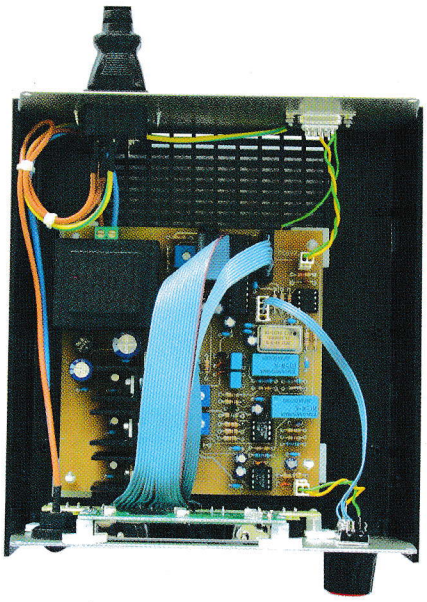


**Figure 10 : prototype du générateur assemblé dans un boîtier modèle AUS22.9 de Teko.**

réglez la valeur avec l'encodeur (en fait on règle chaque « digit » du paramètre). Le générateur mémorise chaque paramètre, si vous avez besoin d'effectuer une nouvelle modification de n'importe quel paramètre, le caractère « underscore » se positionne toujours sous le dernier paramètre modifié.

Lorsque vous allumez le générateur, le réglage se positionne toujours sous le chiffre le plus significatif de la fréquence, de sorte que lorsque vous tournez l'encodeur rotatif vous pouvez augmenter ou diminuer la fréquence par pas de 1 kHz.

Par exemple pour régler la fréquence à 1030 Hz, la modification s'effectue en 2 temps. Appuyez 2 fois sur la touche « ► » pour déplacer le curseur sous le chiffre des dizaines (voir la figure 8), puis tournez de 3 « pas » l'encodeur dans le sens horaire (voir la figure 9).

Vous pouvez ainsi augmenter ou diminuer la fréquence par pas de 10 Hz. Si la fréquence descend en dessous de 1000 Hz, le générateur change de

gamme automatiquement (1.030 kHz ; 1.020 kHz ; 1.010 kHz ; 1.000 kHz ; 990.0 Hz ; 980.0 Hz), de même si la fréquence passe au-dessus de 990 Hz (dans le sens d'une fréquence croissante). Vous pouvez toujours modifier manuellement la gamme à l'aide des touches « ◄ » ou « ► » lorsque le curseur se trouve sous un « chiffre » extrême (à gauche ou à droite).

Par exemple, si vous visualisez une fréquence de 1.000 kHz et que le curseur se trouve sous le chiffre le plus significatif (c'est-à-dire le chiffre « 1 »), en appuyant sur la touche « ► » vous changez de gamme à des fréquences plus élevées (1.000 kHz ; 10.00 kHz ; 100.0 kHz ; 1.000 MHz).

La même manipulation peut être effectuée dans l'autre sens lorsque vous voulez passer rapidement vers des gammes de fréquences plus basses. En maintenant les touches « ◄ » ou « ► » une demi-seconde, la modification devient automatique.

La fréquence générée la plus élevée est limitée à 5 MHz, mais comme nous l'avons déjà mentionné, à cette fréquence le signal sinusoïdal est de bonne qualité, tandis que le signal triangulaire commence à se déformer à partir de 1 MHz.

Tous les autres paramètres peuvent être modifiés de la même manière. Dans le cas de l'**amplitude**, appuyez simplement sur la touche « **Level** » et sélectionnez le chiffre à modifier, puis à l'aide de l'encodeur rotatif modifiez la valeur.

Dans ce cas également le passage d'une gamme à l'autre se fait de manière automatique (1.00 Vcc ; 2.00 Vcc ; 3.00 Vcc ; ou 1.00 Vcc ; 900 mVcc ; 800 mVcc ; 700 mVcc).

Le passage de la gamme des « V » aux « mV » implique l'affichage du terme « m » accompagné par le « clic » du relais indiquant une atténuation de 20 dB comme nous l'avons décrit précédemment.

En ce qui concerne le **réglage de l'offset**, la manière est similaire, mais la **variation n'est pas symétrique**. Supposons que vous avez réglé un offset de 1,80 V, si vous voulez diminuer la valeur du chiffre le plus significatif, l'offset va prendre les valeurs suivantes : + 1.80 V ; + 0.80 V : - 0.20 V ; - 1.20 V.

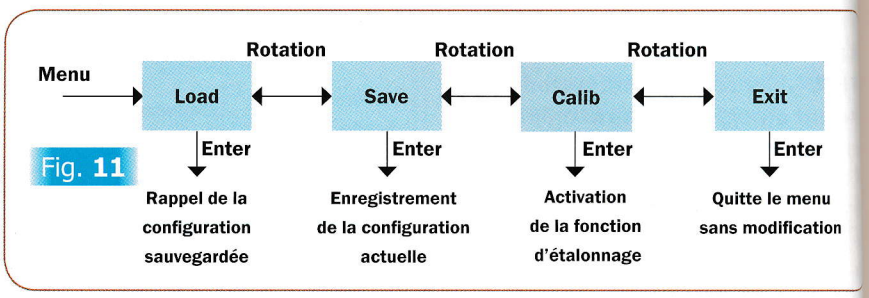


Fig. 11

Nous arrivons maintenant aux paramètres disponibles grâce au bouton « **MENU** ». En plus d'ouvrir le menu, il confirme la sauvegarde des paramètres affichés. En agissant sur l'encodeur vous vous déplacez dans le menu dans l'ordre de la séquence représentée en figure 11. Les différentes fonctions affichées par chaque écran sont intuitives :

- « **Load** » : rappelle les paramètres précédemment enregistrés ;
- « **Save** » : permet de sauvegarder en mémoire les paramètres actuellement définis ;
- « **Calib** » : active la fonction d'étalonnage ;
- « **Exit** » : permet de sortir du Menu.

La **sauvegarde** est effectuée dans la mémoire **EEPROM** (non volatile) du microcontrôleur, les **paramètres ne sont pas perdus lorsque le générateur est mis hors tension**.

## Communication série

Le générateur dispose d'un port de communication série **RS232** standard opto-isolé (en réception uniquement), grâce auquel il peut **recevoir des commandes à partir d'un PC**. La communication dont les caractéristiques sont les suivantes : **9600 bauds, pas de parité, 8 bits de données** et **1 bit d'arrêt**, est toujours active en ce sens que le **générateur est toujours en attente de données entrantes sans la nécessité d'une manipulation particulière**.

**Tableau 1 : Structure des données**

| | |
|---|---|
| Byte0.7, Byte0.6 | Adresse du générateu (non utilisé) |
| Byte0.5 à Byte0.3 | Active |
| Byte0.2 à Byte0.0 | Gamme |
| Byte1 | Données de l'octet hau (F = byte1 x 256 byte2) |
| Byte2 | Données de l'octet bas (F = byte1 x 256 byte2) |

Le protocole de communication prévo la réception de **3 octets consécutifs byte0** (octet0), **byte1** (octet1) et **byte** (octet2), pour un total de 24 bits formé par 3 caractères de type ASCII codé selon le **Tableau 1**. Le « **byte0** » (l premier envoyé) contient l'**adresse d générateur** (bits 7 et 6), le **paramètr à modifier** (bits 5, 4 et 3, paramètr « **Active** ») et la **valeur de la gamm** (bits 2, 1 et 0).

Le **Tableau 2 indique la valeur de l gamme** (ou calibre) en **fonction** de l **fréquence** et de la **tension de sortie** tandis que le **Tableau 3** montre l **valeur du paramètre** « **Active** ».

Comme déjà mentionné, les bits 7 et 6 du « byte0 » contiennent l'adresse du générateur, à utiliser si vous **connectez plusieurs générateurs** (jusqu'à 4) **en même temps sur le même port série** (fonction non prise en charge par la version 1.0 du firmware).

**Tableau 2 : Gamme des fréquences / tensions**

| | Min | Max | Echelle | Gamme |
|---|---|---|---|---|
| Fréquence | 0.001 | 999.9 | Hz | 4 |
| | 1.000 | 9.999 | kHz | 3 |
| | 10.00 | 99.99 | kHz | 2 |
| | 100.0 | 999.9 | kHz | 1 |
| | 1.000 | 5.000 | MHz | 0 |
| Tensions | 100 | 999 | mVpp | 0 |
| | 1.00 | 10.00 | Vcc | 1 |

Pour comprendre les commandes à envoyer, supposons que vous vouliez régler le générateur avec les paramètres suivants :

*Wave = sans changement*
*F (Freq) = 12,34 kHz*
*Level = 2,34 V*
*Offset = sans changement*

Vous devez modifier 2 paramètres et envoyer 6 octets (3 pour la fréquence et les 3 autres pour l'amplitude).

Pour modifier la fréquence, la valeur du « byte0 » doit être :

Adresse = 0 (00)
Active = 1, (001, car vous voulez changer la fréquence d'après le Tableau 3)
Gamme = 2 (010, 10 à 99,99 kHz)

c'est-à-dire que le byte vaut : 00001010.

La valeur 1234, pour être insérée dans les « byte1 » et « byte2 », doit être décomposée en 2 octets selon les formules indiquées dans le Tableau 1, à partir duquel il est possible de déduire que :

*byte1 = partie entière de (F/256)*

*byte2 = F - (Byte1 x 256)*

**Tableau 3 : Valeur du paramètre « Active »**

| Active | Modification liée à : |
|---|---|
| 0 | Forme d'onde |
| 1 | Fréquence |
| 2 | Amplitude |
| 3 | Offset |

donc

byte1 = Int(1234/256) = Int(4,82) = 4

soit 00000100 en binaire.

byte2 = 1234 - (4 x 256) = 1234 – 1024 = 210

soit 11010010 en binaire.

En effet 4 x 256 + 210 = 1234

A ce stade, à chaque octet doit être attribué un caractère en fonction du code ASCII, cette fonction est réalisée automatiquement par le logiciel qui gère la transmission.

De même, pour régler l'amplitude nous devons envoyer :

Adresse = 0 (00)

Active = 2 (010, variation d'amplitude)

Gamme = 1 (001 ; 1 à 10 Vcc)

Ainsi, le « byte0 » vaut 00010001.

En appliquant les mêmes formules que précédemment vous obtenez la valeur 234 qui doit être décomposée en 2 octets

byte1 = 0 (00000000)

byte2 = 234 (11101010)

Chaque fois que le générateur reçoit un ensemble de 3 octets, il vérifie la validité. Si elle n'est pas correcte, les données sont ignorées.

Pour la **gestion du port série à partir d'un PC**, nous avons développé un logiciel nommé « **DSGV2.msi** » **téléchargeable gratuitement** à l'adresse :

**www.electroniquemagazine.com**

dans le sommaire détaillé du numéro 129 à l'onglet « Télécharger ».

Ce programme est écrit en Visual Basic et vous permet de réaliser automatiquement toutes les opérations décrites ci-dessus. L'écran principal du logiciel est représenté sur la figure 12. Il est intuitif et sa prise en main est très simple.

Le réglage du générateur se fait en temps réel au moment exact où, après avoir modifié un paramètre dans le logiciel, vous **relâchez le bouton gauche de la souris**.

Vous pouvez maintenir le bouton gauche de la souris sur les flèches gauche ou droite de la barre de défilement, vous pouvez ainsi faire varier plus rapidement le paramètre concerné.

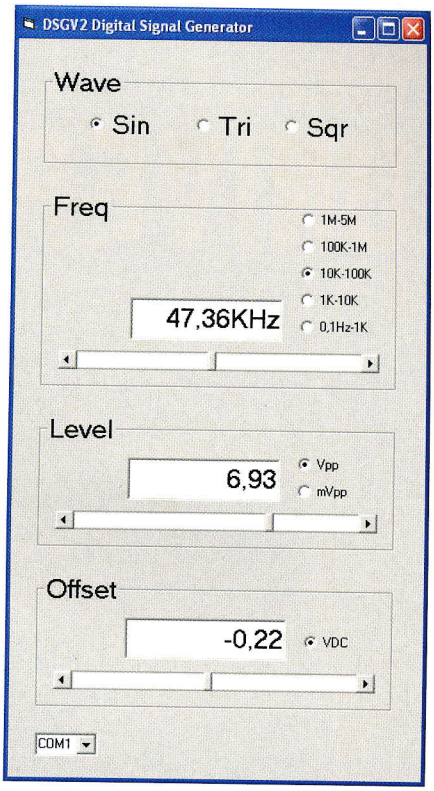
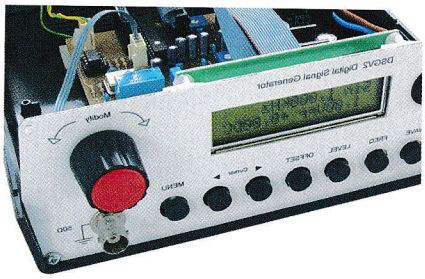
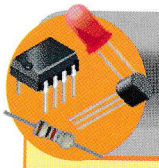



**Figure 12 : ici le logiciel de contrôle ; les réglages du générateur sont pris en compte dès le relâchement du bouton gauche de la souris.**





**Comment construire ce montage**

Tous les composants ainsi que l'**AD9833BRMZ** en CMS sont disponibles auprès de **COMELEC**.

Les typons des circuits imprimés ainsi que les programmes « **MF758.hex** » pour le PIC et « **DSGV2.msi** » pour **Windows** sont téléchargeables gratuitement sur notre site : **www.electroniquemagazine.com** dans le sommaire détaillé de la revue numéro **129** section « **Télécharger** ».

ET798

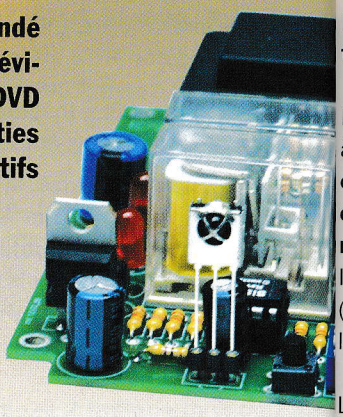
# RÉCEPTEUR UNIVERSEL 2 CANAUX

## POUR TÉLÉCOMMANDE INFRAROUGE






**Ce récepteur infrarouge peut être commandé avec les télécommandes infrarouges des téléviseurs, des appareils Hi-Fi, des lecteurs de DVD et des magnétoscopes. Il dispose de 2 sorties sur relais qui peuvent commuter des dispositifs fonctionnant sous 230 VAC.**



*de Arsenio Spadoni*

Ce montage s'adresse à tous ceux qui veulent allumer et/ou éteindre, sans bouger de leur fauteuil, des dispositifs avec des télécommandes infrarouges que l'on retrouve un peu partout dans la maison et dont on ne se sert pratiquement jamais.

Le projet décrit ici est un **récepteur infrarouge à auto-apprentissage**, capable d'**interpréter les codes des télécommandes infrarouges** (IR) des **téléviseurs**, des **magnétoscopes**, des **lecteurs DVD**, des **décodeurs satellite** ou des **systèmes Hi-Fi**, etc.

Plus précisément ce montage est capable d'**apprendre les codes d'un maximum de 2 touches d'une même télécommande ou 1 touche de 2 télécommandes différentes et d'activer les sorties correspondantes**. Tout ce que vous avez à faire est de lui apprendre les codes, choisissez un bouton d'une télécommande (même celles universelles pour TV, SAT, VCR, etc.) de préférence correspondant à une fonction que vous n'utilisez jamais (les télécommandes ont souvent des touches qui sont rarement utilisées avec certaines versions des appareils avec lesquels elles sont fournies).

Par exemple avec la télécommande du téléviseur, vous pouvez commander un volet roulant motorisé, ou allumer et/ou éteindre les lumières d'une pièce, etc. Tout cela est rendu possible grâce à l'utilisation d'un seul composant extrêmement simple qui est le circuit intégré « **SIS-2** ». C'est un composant fabriqué par la firme **SIMEREC** (www.simerec.com) et conçu pour fonctionner comme récepteur à auto-apprentissage, capable de reconnaître le code des télécommandes IR des produits grand public utilisés dans les maisons.

Le composant est contenu dans un boitier **DIL 8 broches** (semblable à ceux des amplificateurs opérationnels) et alimenté avec une tension continue comprise entre 2 V et 5,5 V, il est capaple de fournir sur chacune de ses 2 sorties un courant de 25 mA.

## Le schéma électrique

Si vous regardez le schéma électrique du récepteur en figure 1, vous pouvez remarquer qu'il très simple. Notre récepteur est composé d'un **circuit intégré décodeur « SIS-2 » (U1)**, d'un **récepteur intégré infrarouge IR38**, de **2 relais** et d'une source d'alimentation **fonctionnant sur le secteur**, ce qui vous permet de connecter le montage directement à une prise secteur ordinaire de votre maison de manière **à disposer sur les sorties des relais 230 VAC pour alimenter les dispositifs** (ou charges).

Le récepteur infrarouge (IR38) est un module contenant une photodiode et un amplificateur à transistor qui élève le niveau de la tension obtenue à partir de la photodiode et la rend utilisable. Entre les **broches S** (« + ») et la **masse** (« - ») **est restitué le code transmis par la télécommande**.

Le circuit intégré « **SIS-2** » s'**apparente à un microcontrôleur programmé qui est adapté pour analyser le signal électrique provenant de la sortie du module infrarouge**. Le signal, donc le **code**, est appliqué à la **broche 11** (« **IRIN**) de **U1**, puis le compare avec les codes déjà programmés.

L'**apprentissage du code** se produit lorsque vous appuyez sur le **bouton P1** (**LEARN**), le circuit mémorise alors le code reçu.

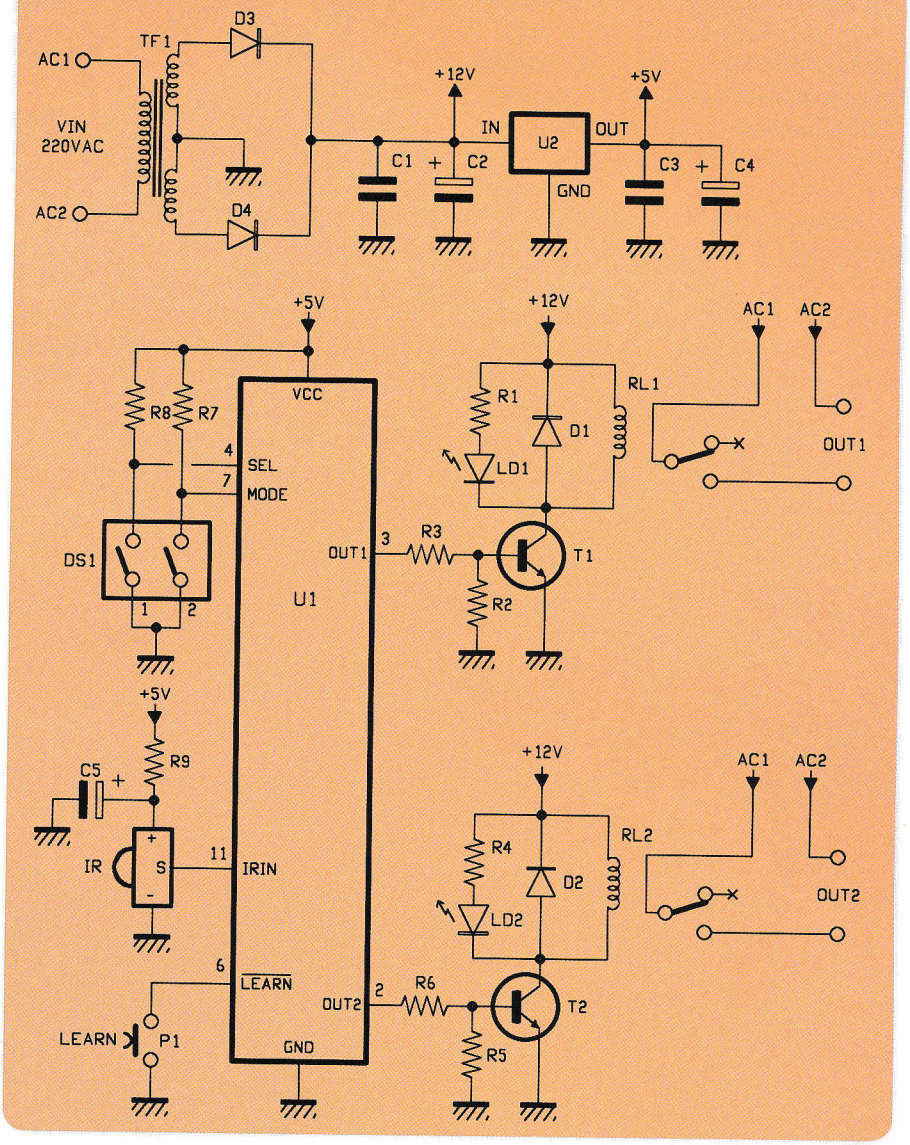
Comme le circuit « **SIS-2** » a été conçu pour **commander 2 sorties séparées** (OUT1 et OUT2), il permet de **mémoriser 2 codes distincts, ou deux fois le même.**

Dans le premier cas, **chaque code correspond à un bouton** de la télécommande, tandis que dans le second cas **les codes stockés correspondent à la même touche** de la télécommande.

Le **mode de fonctionnement des sorties** n'est pas imposé, mais vous pouvez le **paramétrer grâce aux 2 broches de configuration** ; la **4** (**MODE**) et la **7** (**SEL**).

Lorsque la première broche (4) est à un état logique 1 et la seconde à 0, les 2 canaux travaillent en **mode bistable**,

**Figure 1 : schéma électrique du récepteur, il est composé d'un circuit intégré décodeur « SIS-2 » (U1), d'un récepteur intégré infrarouge IR38, de 2 relais et d'une source d'alimentation 230 VAC.**




c'est-à-dire que chaque fois qu'un code « 1 » est reçu, il **inverse l'état de la sortie « OUT1 »**. Il en est de même pour le code « 2 » qui inverse l'état de la sortie « OUT2 ».

En d'autres termes, lors de la première réception d'un code, la sortie correspondante passe à l'état logique 1, à la deuxième réception du même code la même sortie passe à l'état logique 0, à la troisième réception du code elle repasse à 1 et ainsi de suite.

Par contre si les **2 broches de configuration** 4 (MODE) et 7 (SEL) sont à un **niveau haut** (1), les 2 canaux travaillent en **mode impulsion**.

Chaque fois qu'un **code est reçu**, la **sortie correspondante passe au niveau logique 1 et reste dans cet état jusqu'à la fin de la transmission**. Il est évident que dans cette configuration, vous ne pouvez pas avoir les deux sorties actives en même temps, car dans ce cas vous devriez transmettre simultanément 2 codes différents, ce qui empêcherait le « SIS-2 » de les distinguer.

Un **troisième mode de fonctionnement** est prévu par le constructeur du circuit « SIS-2 », ce mode est appelé « **Discrete Button Mode** », et prévoit l'**utilisation uniquement de la sortie OUT1** (**OUT2 est inactive**).

## Plan de montage

Si vous utilisez le récepteur pour piloter des charges qui consomment plus de 4 A, vous devez augmenter la taille des pistes de cuivre du circuit imprimé allant du bornier « VIN » aux contacts des 2 relais.

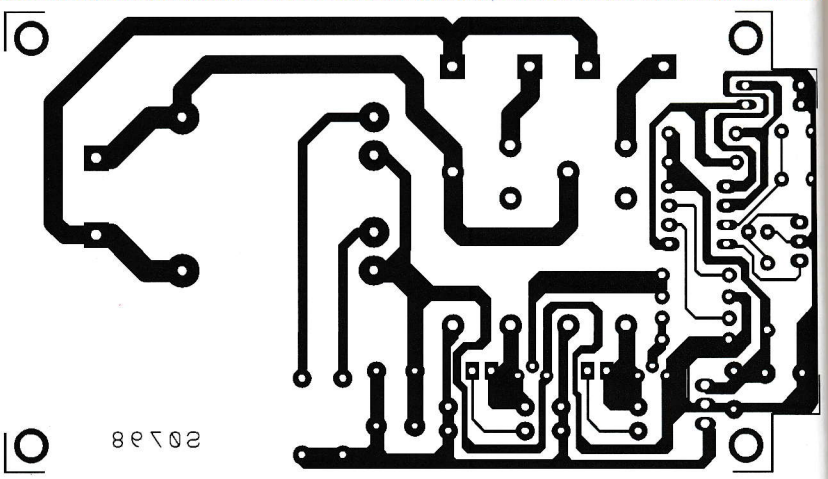


**Figure 2 : dessin du circuit imprimé simple face côté soudures à l'échelle 1 : 1 du récepteur infrarouge 2 canaux.**

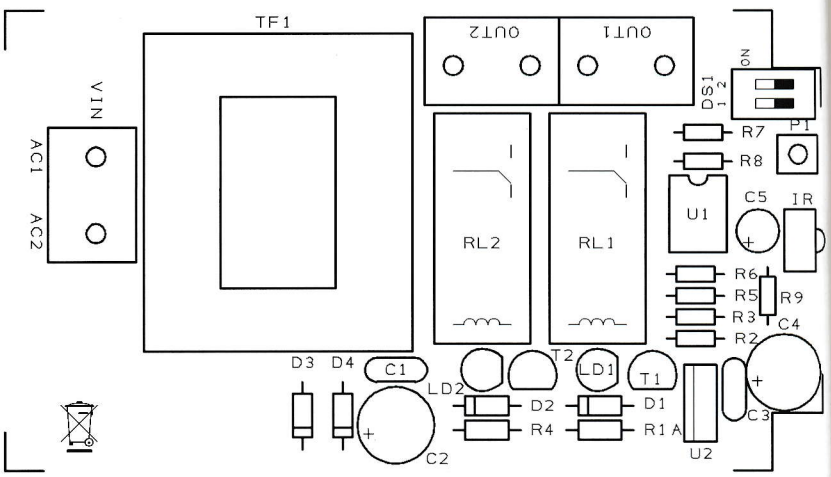



**Figure 3 : schéma d'implantation des composants du récepteur infrarouge 2 canaux.**

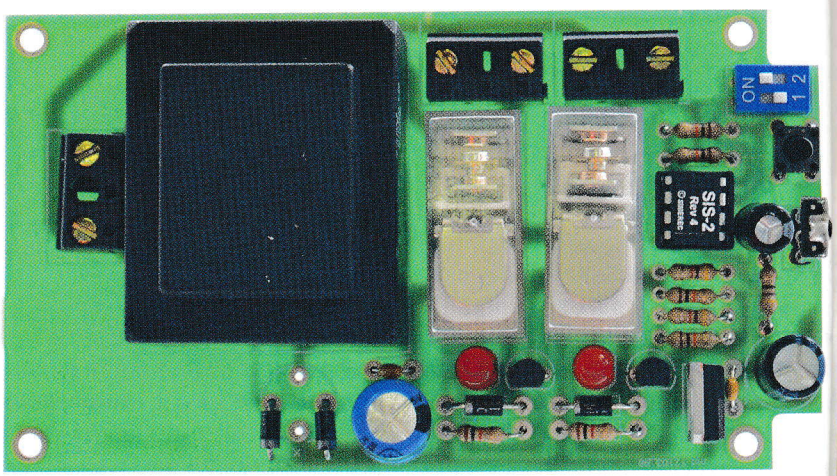



**Figure 4 : photo de l'un de nos prototypes du récepteur infrarouge 2 canaux.**

## Liste des composants du récepteur infrarouge 2 canaux

R1..... 1 kΩ
R2..... 10 kΩ
R3..... 4,7 kΩ
R4..... 1 kΩ
R5..... 10 kΩ
R6..... 4,7 kΩ
R7..... 10 kΩ
R8..... 10 kΩ
R9..... 47 Ω

C1..... 100 nF multicouche
C2..... 470 µF / 25 V électrolytique
C3..... 100 nF multicouche
C4..... 470 µF / 16 V électrolytique
C5..... 47 µF / 16 V électrolytique
U1..... SIS2 Simerec
U2..... 7805
D1..... 1N4007
D2..... 1N4007
D3..... 1N4007
D4..... 1N4007
LD1.... 5 mm rouge
LD2.... 5 mm rouge

T1 .....BC547
T2 .....BC547
P1 .....Microswitch
RL1 ....Relais 12 V 220 VAC / 10 A
RL2 ....Relais 12V 220 VAC / 10 A
TF1 ....Transformateur 220 VAC / 2 x 9 VAC 3VA
IR .....IR38DM
DS1....Dip switch 2 voies

Divers
Bornier 2 pôles pas 10 mm (x 3)
support ci 2 x 4 broches

Cette so
**table** et
**codes**. E
code cor
celle-ci p
repasse
code cor
de fonctio
2 broche
et 7 (SEL

Les **2 so**
« **SIS-2** »
**NPN mon**
**rant**. Les
**relais RL**
sortie **OU**
**1**, **T1 est**
proche de

## Qui

La téléc
d'avoir 5
cette étu
problème
livrent u

Des étud
pays : Da
et les dif

Les dispo
des téléc
Européen

En France
si les fem
qui leur c
contrôle s

L'étude m
le moins.
beaucoup
une interf
les téléco

Bien qu'e
télécomm
alors qu'il
est de plu

Cette sortie fonctionne en **mode bistable** et **est commandée par les 2 codes**. En résumé dès réception du code correspondant à la sortie OUT1, celle-ci passe à un état logique 1 et repasse à 0 lors de la réception du code correspondant à OUT2. Ce mode de fonctionnement est actif lorsque les 2 broches de configuration 4 (MODE) et 7 (SEL) sont à un niveau logique 0.

Les **2 sorties OUT1** et **OUT2** du circuit « **SIS-2** » sont reliées à **2 transistors NPN montés en amplificateur de courant**. Les transistors **pilotent les 2 relais RL1** et **RL2**. Chaque fois que la sortie **OUT1 passe à un niveau logique 1, T1 est saturé** et son collecteur est proche de la masse, **ce qui fait circuler un courant dans la bobine du relais RL1** et colle les contacts du relais. Cette **condition est indiquée par la LED LD1** alimentée en parallèle avec la bobine du relais. Le même raisonnement s'applique pour la sortie OUT2.

Notez qu'en parallèle avec les bobines des relais, en plus des LED avec leur résistance respective pour limiter le courant, **2 diodes sont connectées de manière à être normalement bloquées en fonctionnement normal**, elles **protègent les transistors lors des changements d'états des sorties OUT1** et **OUT2**.

Ces diodes **D1 et D2 placées en parallèle avec les bobines** des relais permettent d'**éviter de détruire les transistors T1 et T2 lors de l'ouverture** de ces derniers. On les appelle généralement « diodes de roue libre ».

Elles sont très importantes et ne doivent pas être négligées. En effet le pilotage d'un relais se fait via une bobine : lors du passage d'un 1 logique à un 0 logique à l'entrée du transistor (base), ce dernier s'ouvre. Or **le courant ne peut pas s'annuler immédiatement dans la bobine du relais**, il se crée alors une **surtension au niveau du collecteur du transistor qui peut être très élevée** et ainsi être **au-dessus de la tension de claquage Vce max** (tension collecteur émetteur maximale préconisée par le constructeur).

## Qui gère la télécommande ?

La télécommande fait partie intégrante de notre vie quotidienne, elle l'a rendue certainement plus pratique. Elle vient d'avoir 50 ans, et c'est la firme Philips qui a mené des recherches sur l'utilisation des télécommandes en l'Europe. Dans cette étude est apparue une situation grotesque et emphatique dans les foyers, à savoir que la télécommande est un vrai problème et peut amener des situations conflictuelles dans la vie des familles. En effet les personnes d'un même foyer se livrent une véritable guerre pour le contrôle de la télécommande et de ce fait du téléviseur.

Des études ont été menées sur le comportement des personnes dans la gestion de la télécommande dans un grand nombre de pays : Danemark, France, Allemagne, Grande-Bretagne, Italie, Pays-Bas et Suède. Le but de l'étude était d'identifier les attitudes et les différences culturelles sur l'utilisation de la télécommande des appareils de divertissement dans les foyers européens.

Les dispositifs domotiques et la diffusion du multimédia dans les maisons modernes, et par conséquent la multiplication des télécommandes qui régissent leur fonctionnement, sont la source de beaucoup de conflits dans les familles : 22% des Européens disent au moins se disputer 12 fois par semaine avec leur partenaire à cause de la télécommande.

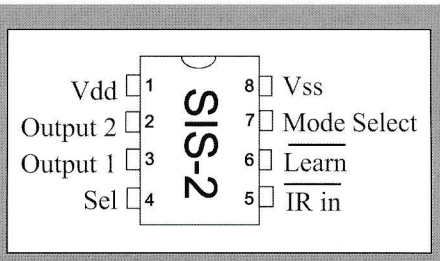
En France, dans la majorité des cas (70 %) la télécommande n'est pas un sujet particulier de conflits dans la famille, même si les femmes (30 %) se plaignent plus souvent que les hommes (18 %) de ne pas être en mesure de choisir le programme qui leur convient. Dans l'ensemble, environ les trois quarts (72 %) des Français estiment qu'il est important d'exercer un contrôle sur la télécommande. Et cela devrait nous faire réfléchir sur la « légèreté » de l'ambition de beaucoup de personnes.

L'étude montre que les Britanniques sont ceux qui utilisent les télécommandes et les Français sont ceux qui les utilisent le moins. Les Espagnols sont prêts à se battre pour les appareils électroniques de la maison, tandis que les Danois sont beaucoup plus détendus. Par contre une idée commune à tous les Européens est le désir que la télécommande devienne une interface intuitive, facile à utiliser et qui simplifie le divertissement (plus de 51 % des Européens échangeraient toutes les télécommandes de leur foyer contre une seule qui serait capable de commander tous les appareils).

Bien qu'en France, 71 % des personnes interrogées (des deux sexes) soutiennent la parité sur « le responsable » de la télécommande, l'étude révèle que les hommes (35 %) et que les enfants (34 %) exercent un contrôle sur la télécommande, alors qu'il y a seulement 11 % de femmes. Cependant on observe une tendance en Europe du Sud, où la télécommande est de plus en plus sous le contrôle féminin.

lus vous vous déplacez vers le nord, plus il existe une relation harmonieuse entre les hommes et les femmes au sujet de la télécommande. 53 % des hommes Néerlandais et 42 % des hommes Allemands exercent un contrôle sur la télécommande, les hommes dans les pays nordiques sont plus détendus à ce sujet, avec seulement 20 % des Danois et 22 % es Suédois qui considèrent que le contrôle de la télécommande est un ment important.

Le récepteur a été conçu pour **commander des charges** (appareils ou dispositifs) **fonctionnant sur le secteur 230 VAC**, les **contacts des relais** supportent **10 A sous 230 VAC**. Cependant **les dimensions des pistes de cuivre du circuit imprimé allant du bornier « VIN » aux contacts des 2 relais ne peuvent supporter qu'un maximum de 5 A**. Si vous voulez **augmenter la puissance de commutation**, il vous faudra **augmenter la taille des pistes de cuivre**.




**Brochage du circuit intégré « SIS-2 » de Simerec : le composant est encapsulé dans en boîtier plastique DIL à 8 broches et dispose de deux entrées (« Sel » et « Mode Select ») pour choisir le mode de fonctionnement. La broche « LEARN » permet d'activer le mode « apprentissage des codes » en fonction des touches de la télécommande. Le circuit « SIS-2 » est alimenté avec une tension comprise entre 2 V et 5,5 V.**

L'ensemble du circuit fonctionne à l'aide d'une simple alimentation AC/DC constituée par le transformateur **TF1** (220 VAC 2 x 9 VAC 3 VA), par un **redresseur double alternance** (D3 et D4) et par les condensateurs C1 et C2 qui filtrent les ondulations sinusoïdales sortant du redresseur formé par D3 et D4.

Nous obtenons une tension continue filtrée mais non stabilisée de 12 V aux bornes de C2 qui alimente les bobines des relais.

Pour obtenir le 5 V stabilisé nécessaire au fonctionnement du récepteur infrarouge (IR38) et du circuit intégré (SIS-2), nous utilisons un classique régulateur de tension U2 (7805) qui fournit à partir du 12 V une tension stabilisée de 5 V. Les condensateurs C3 et C4 filtrent la tension régulée des éventuels parasites.

## Réalisation pratique

À ce stade, nous allons voir l'aspect de la construction du circuit de ce récepteur de télécommandes infrarouges universel dont le circuit imprimé est réalisé sur une simple face donc relativement facile à fabriquer. Le **typon du circuit imprimé** à l'échelle 1 : 1 est **téléchargeable sur notre site www.electroniquemagazine.com** dans le sommaire détaillé du **numéro 129** à l'onglet « **Télécharger** ».

Une fois le circuit imprimé gravé et percé, commencez par souder les composants ayant un bas profil c'est-à-dire en premier les résistances et les diodes (attention à la polarité qui est indiquée par une bague sur le boitier des diodes représentant la cathode).

Ensuite soudez le support du circuit intégré U1, le DIP switch DS1 et les condensateurs non polarisés et après les électrolytiques (faites attention à la polarité le « - » est indiqué sur le boitier).

Enfin soudez les 2 transistors T1 et T2, le régulateur de tension U2 (attention à l'orientation, le méplat métallique doit être du côté du transformateur), le module infrarouge IR38, sa face arrondie doit être positionnée vers l'extérieur du circuit imprimé. Il ne vous reste plus qu'à souder les 3 borniers, le transformateur TF1 et les 2 relais RL1 et RL2. Aidez-vous du schéma d'implantation des composants (figure 3) et de la photo du prototype (figure 4) pour vérifier l'orientation des composants polarisés.

Finissez l'assemblage et vérifier que tout est en ordre, vous pouvez installer le montage dans un boitier approprié avec une fenêtre en plexiglas rouge ou transparente pour le module infrarouge. Prévoyez le passage des fils allant aux borniers (4 pour les relais allant aux charges et 2 pour le secteur 230 VAC), pour cela aidez-vous de la figure 5. Il serait utile de prévoir un trou pour rendre accessible le bouton P1.

**Avant d'effectuer les tests**, vous devez choisir la manière de **configurer DS1**, c'est-à-dire le mode de fonctionnement du récepteur. Pour cela aidez-vous du **Tableau 1** qui résume les différents modes.

Une fois le mode défini, vous devez effectuer l'apprentissage des codes, il est donc nécessaire d'alimenter le circuit en connectant le bornier « VIN » à un cordon d'alimentation terminé par une prise secteur. Comme le **montage est directement relié au 230 VAC, nous vous recommandons de le placer sur un endroit isolé s'il n'est pas dans le boitier**. Assurez-vous qu'il n'ait pas de pièces métalliques en dessous ou à proximité.

**Pendant l'apprentissage des codes ne reliez pas de dispositifs ou charges** au montage (les borniers de relais ne doivent pas être connectés). Dès que vous alimentez le montage vous pouvez commencer la procédure d'apprentissage des codes.

D'abord choisissez la télécommande que vous souhaitez appairer (jumelée avec le récepteur et, à ce sujet, sélectionnez 2 touches que vous n'utiliserez pas. **Mettez hors tension l'appareil** (TV, lecteur DVD,...) qui est **commandé par la télécommande**. Une fois les touches identifiées, **appuyez sur P1** et attendez que la procédure débute (la réception est signalée par l'**allumage de la LED LD1** et par l'**activation de la sortie OUT1**).

À ce stade, **lâchez P1** et **appuyez sur le bouton** de la télécommande que vous voulez faire correspondre au **code 1**, et **relâchez-le lorsque la LED LD1 s'éteint** (le relais RL1 retourne au repos). Vérifiez qu'au **bout de 2 secondes le relais s'enclenche de nouveau**.

Maintenant, **répétez 3 fois la procédure décrite ci-dessus**, c'est à dire faites une nouvelle transmission avec le même code et vérifiez que le relais revienne au repos, puis attendez qu'il s'active de nouveau après 2 secondes. Les répétitions servent à confirmer l'appariement avec la touche voulue, confirmant avec succès que le « SIS-2 » a reçu 3 fois le même code pour un canal donné (canal 1 ou 2).

## Connexions des relais et des charges au réseau

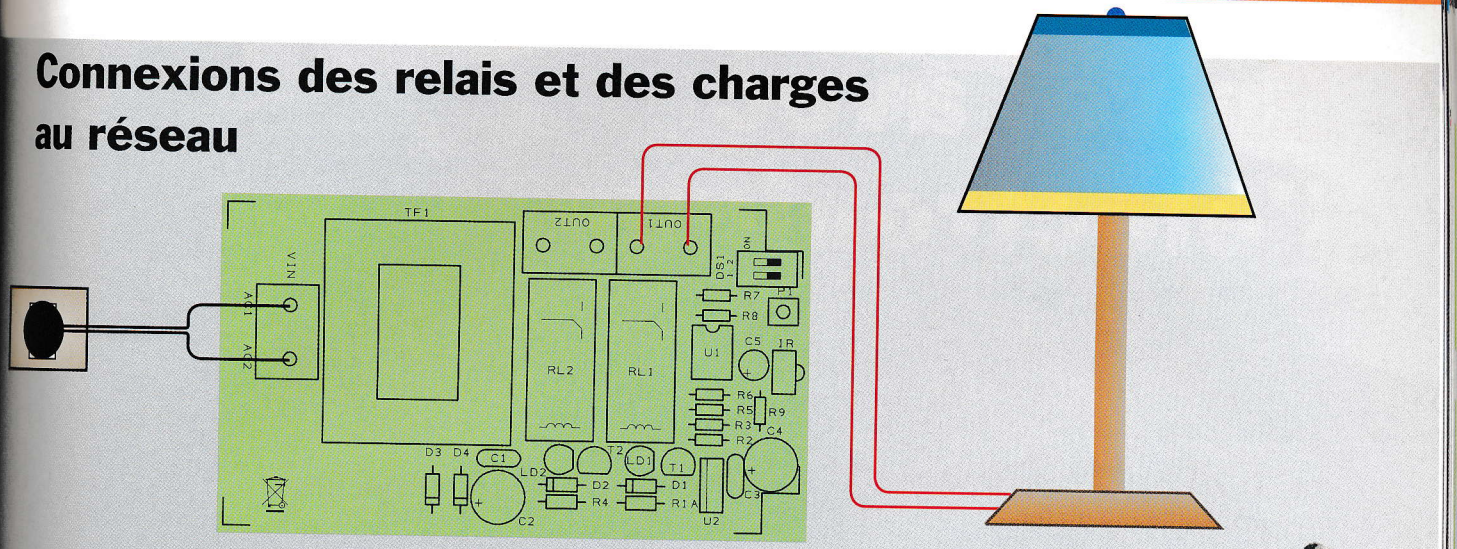
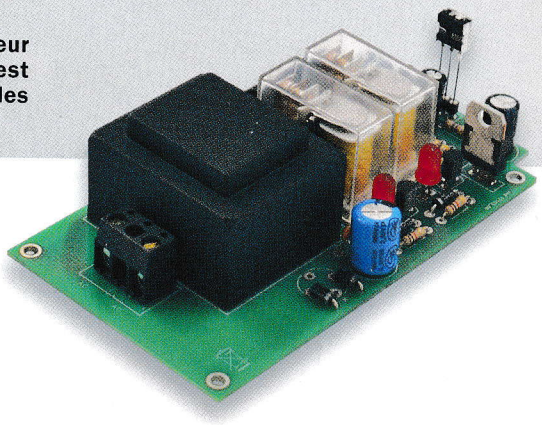



**Figure 5 : les contacts des relais sont directement alimentés par la tension secteur (à travers le circuit imprimé) provenant de la prise dont l'une des extrémités est branchée sur le bornier « VIN » et permet à l'utilisateur de connecter directement les charges sur les borniers OUT1 ou OUT2.**



L'apprentissage et la mémorisation d'un code sont réalisés en 3 étapes, et sont signalés par le fait que le relais RL1 et la LED LD1 « pulsent » plusieurs fois, puis restent activés. Lorsque cela se produit, le circuit est prêt pour l'apprentissage du deuxième code.

Dirigez la télécommande vers le récepteur et appuyez sur la touche que vous voulez faire correspondre avec le 2eme code (deuxième canal) pour voir s'éteindre LD2 et désactiver RL2. Comme pour l'apprentissage du premier code, le relais et la LED se réactivent après 2 secondes environ. Quand cela se produit, il sera nécessaire de répéter 3 fois la transmission, comme vous l'avez fait pour le premier canal.

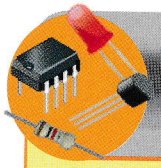
Pour que l'**apprentissage ait lieu**, **LD1** et **RL1** « **pulsent** » à **plusieurs reprises puis restent actifs**. À ce stade, le circuit est prêt à fonctionner.

Lors d'une transmission avec la touche correspondant au canal 1, en fonction du mode choisi, le relais RL1 change d'état (mode bistable) ou restera actif et reviendra au repos lors du relâchement de la touche de la télécommande (mode monostable).

Si vous avez choisi le mode « **Discrete Button Mode** », le relais **RL1 s'enclenche** lorsque vous appuyez sur la touche correspondant au **canal 1** et **revient au repos** lorsque vous appuyez sur la touche correspondant au **canal 2**. Une autre façon de désactiver le relais après la fin de l'apprentissage est de mettre hors tension le circuit pendant quelques secondes.

La procédure d'appariement des touches de la télécommande peut être répétée sans limite, lorsqu'il est nécessaire par exemple de changer de touches car vous vous êtes rendu compte qu'elles étaient affectées à d'autres fonctions de la TV ou du lecteur DVD. Il est bien entendu qu'à chaque apprentissage vous devez mémoriser les 2 codes. Cela signifie que ceux qui n'ont pas besoin de 2 canaux distincts peuvent mémoriser deux fois la même touche, même si cela n'a pas de sens parce que le « SIS-2 » ne fonctionne pas avec deux commandes simultanément.

| Mode | Description | dip 1 | dip 2 |
|---|---|---|---|
| bistable normal | Lors de la réception d'un code déjà mémorisé, le canal correspondant inverse son état par rapport à sa condition antérieure. | 1 | 0 |
| monostable | Lors de la réception d'un code déjà mémorisé, la sortie correspondante passe à un niveau haut et revient à 0 quand la transmission s'arrête. | 1 | 1 |
| Discrete Button | Ne fonctionne qu'avec la sortie OUT1, le code 1 active la sortie, le code 2 la désactive. | 0 | 0 |



### Comment construire ce montage

Tous les composants ainsi que le circuit décodeur **SIS2** et le module infrarouge **IR38DM** sont disponibles auprès de **COMELEC**.

Le typon du circuit imprimé est téléchargeable gratuitement sur notre site : **www.electroniquemagazine.com** dans le sommaire détaillé de la revue numéro **129** section « **Télécharger** ».

# En vol avec l'électronique



L'aviation civile est le secteur des transports où les dépenses et les innovations sont les plus importantes. Cela crée une synergie entre le développement des technologies et la stratégie commerciale afin de nous proposer des vols moins chers mais sans compromettre notre sécurité. Depuis le premier avion avec des ailes constituées de bois et de toile jusqu'au moderne Boeing « 787 Dreamliner » et Airbus « A320 Neo », beaucoup d'eau a coulé sous les ponts, ou plutôt « beaucoup d'air s'est écoulé sous les ailes ».

Cependant les innovations les plus importantes au cours des dernières décennies dans l'aviation se sont produites au niveau de l'électronique embarquée. Ainsi, dans l'industrie aéronautique est né le mot « avionique » qui représente l'ensemble des équipements électroniques, électriques et informatiques qui aident au pilotage des avions dans des conditions de pression, température, humidité inhabituelles pour des systèmes électriques, électromécaniques et informatiques classiques. Dans ces pages, nous allons vous expliquer comment l'électronique a changé le monde de l'aviation, en particulier dans les commandes de vol.

Le **premier automatisme important** qui a été adopté dans l'aviation civile fut le « **pilote automatique** » (**PA**). Il a été développé en **1914** par **Lawrence Sperry** et à l'époque il assurait les tenues d'altitude et de cap. Initialement basé sur des capteurs (anémomètres, variomètres, compas, altimètres traditionnels) et des actionneurs mécaniques, aujourd'hui le « PA » est un sous-système électronique complexe, qui travaille avec l'ordinateur de bord. Le « PA » est un dispositif électronique qui transmet des ordres à des sous-systèmes hydrauliques ou électriques sollicitant les gouvernes et les moteurs d'un avion à la place du pilote en fonction des instructions que celui-ci aura préalablement programmées. Il ne remplace pas le pilote durant l'intégralité du trajet. Le « PA » permet le réglage automatique des paramètres comme le maintien de l'altitude, la vitesse de croisière souhaitée, le taux de montée ou de descente (il ne doit pas être supérieur à 3 % car un angle plus important pourrait effrayer les passagers et induire des douleurs dues à la variation rapide de la pression). Le **taux de montée** représente la **vitesse ascensionnelle** d'un avion qui est la valeur de la projection du vecteur vitesse sur l'axe vertical. Elle correspond à une variation d'altitude pendant une unité de temps. La vitesse ascensionnelle est mesurée par un instrument de bord appelé « variomètre ». Ces instruments sont généralement gradués en pied par seconde ou par minute (« ft/s » ou « ft/min ») ou en mètre par seconde (m/s). Pour information un **Rafale** de chez **Dassault** a une **vitesse ascensionnelle** de 18 000 m/min soit **300 m/s**.

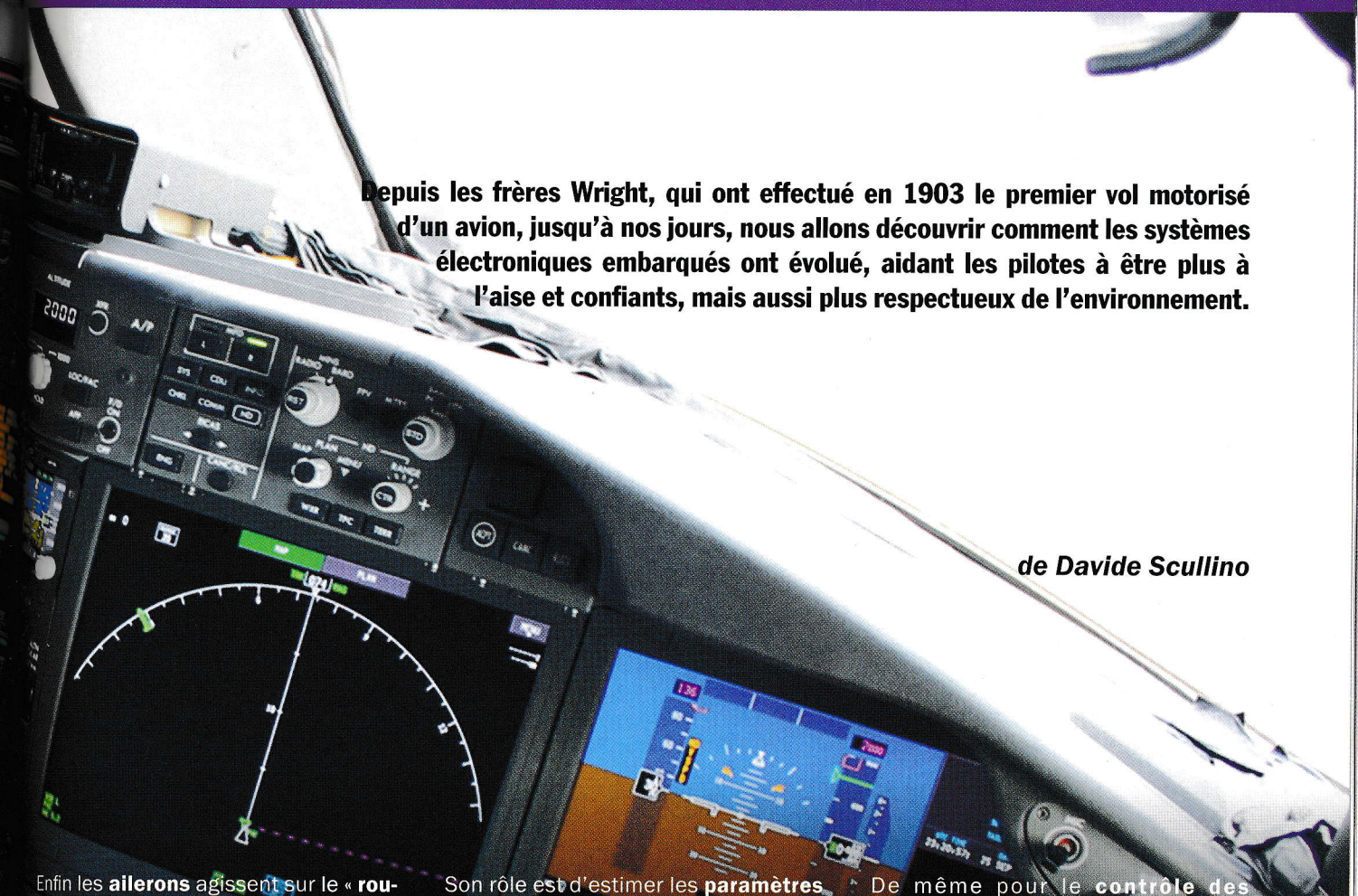
Par exemple pour les avions de ligne modernes, le taux de montée est entièrement contrôlé par le « PA » en gérant automatiquement la poussée des moteurs lorsque celui-ci est couplé à l'« auto-manette ».

La **trajectoire est contrôlée** par le réglage des **gouvernes de direction** et de **profondeur** ainsi que par les **ailerons**. La **gouverne de direction** agit sur le mouvement de « **lacet** ». Le « lacet » est un mouvement de rotation horizontale autour d'un axe vertical, ce mouvement est commandé par l'action sur les palonniers qui agissent sur la gouverne de direction. La **gouverne de profondeur** agit sur le « **tangage** » de l'avion qui représente un mouvement de rotation autour de l'axe transversal.

Enfin les **ailerons** » de l'avion vement de roulis longitudinal

De nombreux du « **Yaw dam** « **Yaw** », qui e tion afin de c **hollandais** » « lacet-roulis vant pour le ceux placés mécanisme a mouvements verne de dire utilisé même et agit sur le l'aide de sys ou électromé

Le **pilote aut le décollage** bien que les guidage « **G** atterrir sans matique est à au système de l'avion. L' Reference Sy grant 3 sous (Air Data Ine



**Depuis les frères Wright, qui ont effectué en 1903 le premier vol motorisé d'un avion, jusqu'à nos jours, nous allons découvrir comment les systèmes électroniques embarqués ont évolué, aidant les pilotes à être plus à l'aise et confiants, mais aussi plus respectueux de l'environnement.**

*de Davide Scullino*

Enfin les **ailerons** agissent sur le « **roulis** » de l'avion qui représente un mouvement de rotation autour de son axe longitudinal (axe de roulis).

De nombreux appareils sont équipés du « **Yaw damper** », appelé simplement « **Yaw** », qui est un dispositif d'atténuation afin de contrer les effets du « **roulis hollandais** », phénomène de couplage « lacet-roulis » particulièrement éprouvant pour les passagers, surtout pour ceux placés loin du centre de gravité. Ce mécanisme automatique coordonne les mouvements des ailerons et de la gouverne de direction. Ce système peut être utilisé même sans pilote automatique et agit sur les ailerons et la gouverne à l'aide de systèmes électrohydrauliques ou électromécaniques.

Le **pilote automatique ne sert qu'après le décollage** et **avant l'atterrissage**, bien que les avions modernes avec le guidage « **GLS** » très précis puissent atterrir sans lui. La tâche du pilote automatique est relativement facile grâce à au système « **ADIRS** » intégré à bord de l'avion. L'« ADIRS » (Air Data Inertial Reference System) est un système intégrant 3 sous-unités appelées « **ADIRU** » (Air Data Inertial Reference Unit).

Son rôle est d'estimer les **paramètres** « **inertiels** » et « **air** » de l'avion.

Les **paramètres** « **air** » sont l'**angle d'incidence**, l'**angle de dérapage**, la **pente** « **air** », la **vitesse CAS** ainsi que le nombre de **Mach**, et la **composante horizontale** de la **TAS**. Ils sont estimés à partir des mesures de l'écoulement de l'air (capteurs de pression, de température et d'angle d'incidence) puis des calculs de l'ADC.

Les **paramètres inertiels** sont l'**assiette latérale**, l'**angle de lacet**, l'**assiette longitudinale**, le **cap** ainsi que la **route**. Les paramètres « **GS** » (ground speed), « **FPA** » (Flight Path Angle), « **WS** » (Wind Speed), « **DA** » et « **POS** » sont estimés à partir des mesures d'accélérations et des vitesses angulaires puis des calculs de l'**IRS**. Grâce à l'acquisition de tous ces paramètres, l'« ADIRS » permet de détecter et de transmettre à l'ordinateur tout changement de direction, d'inclinaison, mais aussi de vitesse et d'accélération. A l'aide de **gyroscopes**, **accéléromètres à 3 axes**, il permet de percevoir la **rotation** (« roll ») et l'**inclinaison** (« pitch ») avec une grande précision et fiabilité.

De même pour le **contrôle des moteurs** d'un avion, qu'ils soient de type turboréacteurs, turbopropulseurs ou statoréacteurs, on fait appel au système électronique nommé « **FADEC** » (**Full Authority Display Engine Control**).

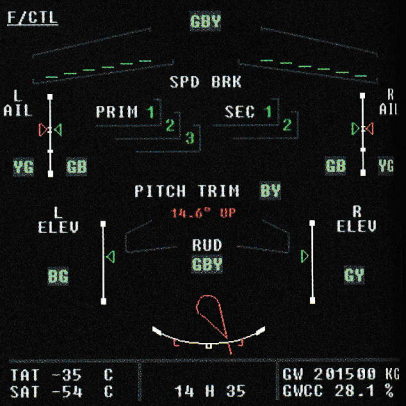
Le FADEC est une unité à microprocesseurs qui s'interface entre le « cockpit » et le moteur d'un avion. Il permet d'assurer le fonctionnement des turboréacteurs ou turbopropulseurs.

Il contrôle les paramètres de fonctionnement de la combustion, du refroidissement et de la lubrification. Il ajuste aussi le nombre de tours de la turbine en fonction des exigences de la commande de puissance gérée par le pilote ou l'auto-manette avec le pilote automatique.

Le système FADEC est composé d'un calculateur à 2 voies symétriques, redondantes et à pleine autorité. Les capteurs principaux et les actionneurs sont, pour leur partie électrique, dupliqués (un par voie). Seuls les organes hydrauliques (pompes, servovalves, générateurs de pression) sont uniques (non redondants).

# Les instruments d'hier et d'aujourd'hui

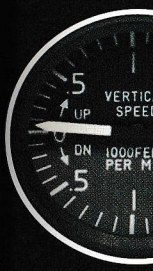





**Ici sont représentés sur un écran tous les paramètres de l'« ADIRS » notamment ceux de la rotation (« roll ») et de l'inclinaison (« pitch »).**








Il suffit de grimper à bord du cockpit d'un avion moderne et de regarder des photos d'avions de la Seconde Guerre Mondiale, pour constater une réelle différence. A l'époque, les avions disposaient d'instruments à main tels que des compas, des manomètres, etc.

Aujourd'hui tous les instruments ont été remplacés par de grands écrans LCD qui affichent toutes les données fournies par l'ordinateur de bord. Il n'y a pas si longtemps, on volait en regardant l'horizon artificiel pour évaluer l'inclinaison (« pitch »), la boussole pour la rotation, et on vérifiait l'altimètre pour avoir la bonne altitude.

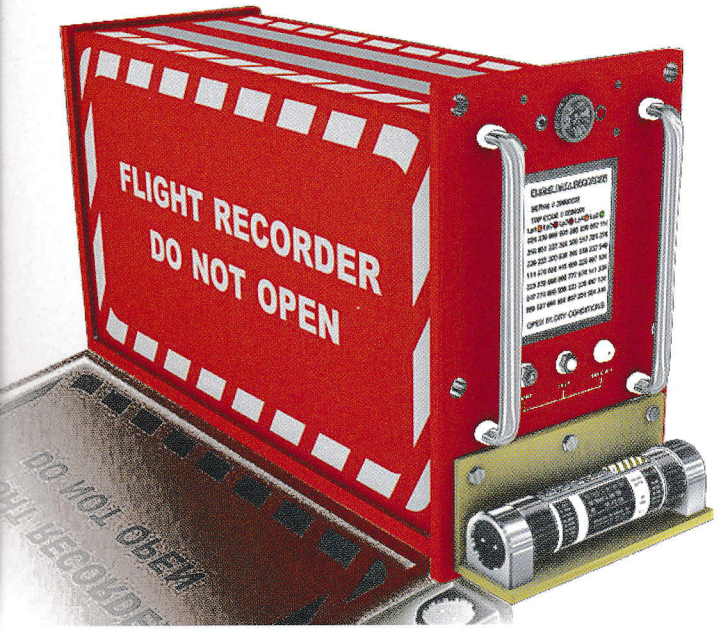
D'autres instruments précieux tels que l'indicateur de vitesse (anémomètre) et le variomètre informaient respectivement de la vitesse relative et du changement d'altitude, c'est-à-dire le taux de montée ou de descente.

La boussole, vous savez certainement comment elle fonctionne, est constituée d'une aiguille aimantée pointant vers le Nord terrestre.

L'horizon artificiel ou « indicateur d'assiette » est un peu plus complexe, c'est un instrument de bord qui mesure l'assiette de l'avion par rapport à l'horizon. Il fonctionne un peu comme la bulle d'un niveau de maçon, il contient une sphère suspendue (gyroscope) avec 2 lignes orthogonales qui suivent le niveau de la mer et, par conséquent, indiquent si l'avion s'incline latéralement ou longitudinalement.

Le gyroscope qui est réglé à zéro avant le décollage conserve cette position fixe dans l'espace quels que soient les mouvements de l'avion. Il permet donc de mesurer l'angle entre l'axe de l'avion et l'horizontale, l'angle de tangage, l'angle avec la verticale et l'angle de roulis.

L'altimètre permet de déterminer la distance verticale entre un point et




Les fonctions de régulation du FADEC sont :

- assurer la poussée en fonction de la demande de l'auto-manette (position Alpha) ;
- garantir la sécurité de fonctionnement du réacteur ;
- assurer les performances requises ;
- améliorer le fonctionnement opérationnel du réacteur ;
- réduire la charge de pilotage ;
- améliorer la maintenance par l'intégration d'un système de recherche de pannes.

## De nombreux capteurs, moins de crainte

Un autre grand pas vers la fiabilité a été réalisé avec l'installation d'unités de diagnostics qui reportent dans le cockpit (poste de pilotage, si vous préférez) les informations de plusieurs paramètres qui doivent être maintenus sous surveillance.

L'ordinateur a permis à de gros avions de voler avec seulement 2 pilotes (le commandant et le copilote qui se trouve normalement à la droite du commandant). L'ingénieur de vol, qui était aussi un membre d'équipage lisait uniquement les instruments et les indicateurs situés sur le côté à l'arrière de la cabine (il surveillait surtout les paramètres des moteurs).

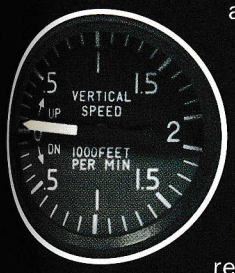
L'ingénieur d
vée des avio
le Boeing 76
MD-11.

Les avions
de capteurs
contrôles » a
exercées sur
la pression a
pérature ext
luer le risque
les surfaces
(ailerons, go

D'autres cap
sion des pne
sage, ainsi q
des moteurs
défauts de to
issues des ca









une surface de référence. L'altitude est mesurée par rapport au niveau de la mer, la hauteur par rapport au niveau du sol local et le niveau de vol par rapport à une surface isobare théorique. L'altimètre mesure la différence de la pression atmosphérique entre sa position et une surface de référence. L'instrument affiche alors l'altitude correspondante en supposant une atmosphère standard.

Alors que l'altimètre fonctionne en détectant la pression d'air atmosphérique, l'indicateur de vitesse exploite le principe du tube de « **Pitot** » (il doit son nom au **physicien français Henri Pitot**). Un tube de « **Pitot-statique** » est constitué de 2 tubes coudés concentriques dont les orifices, en communication avec le fluide (l'air) dont on veut mesurer la vitesse, sont disposés de façon particulière :

- le tube extérieur s'ouvre perpendiculairement à l'écoulement du fluide. La pression à l'intérieur de ce tube est donc égale à la pression ambiante ou pression statique ;
- le tube intérieur est parallèle à l'écoulement du fluide, et est ouvert à son extrémité, face au flux. La pression à l'intérieur de celui-ci est donc la pression totale, somme de la pression statique et de la pression dynamique.

Un manomètre mesure la différence de pression entre les deux tubes, c'est-à-dire la pression dynamique, et permet donc de calculer la vitesse d'écoulement du fluide autour du tube. Cette vitesse correspond au vent relatif et est, en aéronautique, une des informations primordiales pour le pilote qui doit toujours maintenir son appareil au-dessus de sa vitesse de décrochage et au-dessous de sa vitesse maximale. Elle permet en outre, connaissant la vitesse du vent, de calculer la « vitesse sol » et la consommation de l'avion.

Aujourd'hui, l'horizon artificiel a évolué, ses informations sont issues d'accéléromètres « **MEMS** » (en nanotechnologie), de variomètres et altimètres qui contiennent des capteurs piézoélectriques de pression. L'ensemble constitue un accéléromètre 3 axes.

On utilise également des gyroscopes qui mesurent la rotation de l'avion autour d'un axe donné, permettant ainsi d'évaluer les changements de direction par rapport à la position actuelle. La combinaison de plusieurs gyroscopes permet d'obtenir des indications sur les 3 axes.

Pour la mesure de la vitesse d'un avion, on pourrait utiliser un GPS, mais pour le moment le signal n'est pas assez précis, alors on utilise toujours les tubes de « **Pitot** ». Dans ce cas la pression n'agit pas sur un système pneumatique, mais plutôt sur des capteurs piézoélectriques.

Toutefois, pour des raisons de sécurité, dans le cockpit il reste quelques instruments traditionnels tels que l'horizon artificiel et l'altimètre, ainsi que la jauge de carburant.

En cas de panne de l'ordinateur et par conséquent l'extinction de l'instrumentation LCD, ces instruments traditionnels peuvent fournir des informations clés.

L'ingénieur de vol a disparu avec l'arrivée des avions tels que l'Airbus A300, le Boeing 767 et le McDonnell Douglas MD-11.

Les avions modernes sont équipés de capteurs dans les « surfaces de contrôles » afin de détecter les forces exercées sur le fuselage, notamment la pression atmosphérique et la température extérieure de l'air, pour évaluer le risque de formation de gel sur les surfaces sensibles de l'appareil (ailerons, gouvernes).

D'autres capteurs mesurent la pression des pneus des trains d'atterrissage, ainsi que tous les paramètres des moteurs (température, rotation, défauts de toutes sortes). Les données issues des capteurs convergent vers les écrans multifonctions qui remplacent les instruments à mains et les indicateurs mécaniques. Cependant ceux-ci sont toujours présents et sont utilisés en cas de défaillance électronique pour visualiser les paramètres vitaux tels que la vitesse, l'altitude, l'horizon artificiel, le niveau de carburant dans les réservoirs, la pression d'huile moteur.

Aujourd'hui, dans les avions modernes il y a un grand écran LCD, souvent tactile, qui résume les données telles que la direction, le cap, la vitesse, le taux de montée, la vitesse des moteurs, la position par rapport à l'horizon dans la phase de montée ou de descente et qui fonctionne comme une interface bidirectionnelle entre le pilote et l'avion.

Parmi les centaines de mesures effectuées par les capteurs, il existe une **alarme de décrochage** adoptée depuis plusieurs décennies sur les avions de ligne qui est constituée d'une alarme sonore ou d'un dispositif de vibration monté dans le manche du pilote ou du copilote (dans les DC-10 et dans les MD-11) afin de percevoir la vibration.

Le détecteur de décrochage est basé sur le principe suivant : sur le bord d'attaque de l'aile (l'angle d'attaque est l'angle que fait la ligne longitudinale de l'aile avec la direction de l'air) se trouve une petite palette placée horizontalement et mobile autour d'un axe. Au sol elle est normalement maintenue en position basse par son poids, mais si on la soulève avec son doigt, une alarme retentit à l'intérieur du cockpit.

# Les commandes de vol primaires d'un avion de lig

Les avions exploitent, pour voler, le phénomène de « portance ».

C'est une force qui est générée par les ailes en raison de la différence entre la vitesse de l'air qui circule sur la face inférieure (intrados) et celle qui circule sur la face supérieure (extrados).

Pour produire le phénomène de « portance », les ailes sont conçues avec un profil dissymétrique de manière à créer des vitesses plus élevées sur l'extrados et plus faibles sur l'intrados. Selon le « Théorème de Bernoulli » (Daniel Bernoulli, né à Groningue le 8 février 1700 et mort à Bâle le 17 mars 1782, était un médecin, physicien et mathématicien Suisse) cela se traduit par des pressions plus fortes sur l'intrados que sur l'extrados, donc une portance dirigée vers le haut.

Bien évidemment pour produire ce phénomène qui s'exerce perpendiculairement à la direction du mouvement, il faut créer un mouvement horizontal à l'aide de moteurs qui font avancer l'avion.

Une fois que l'avion a décollé, il est suspendu dans l'air par ses ailes, cependant pour le faire monter ou descendre, il faut un « contrepoids » qui le dirige vers le haut ou vers le bas.

Nous ne parlons pas d'un contrepoids mécanique mais d'un système aérodynamique, qui se compose d'un stabilisateur. Celui-ci est constitué de 2 petites ailes symétriques situées à l'arrière de l'avion (l'empennage), centrées par rapport au fuselage, et dont l'inclinaison détermine un effort sur l'axe transversal des ailes en orientant le « nez » vers le haut ou vers le bas.

Dans les avions de ligne, le stabilisateur se positionne finement au moyen d'un compensateur (ou « trim » en anglais), qui règle la distribution de la charge aérodynamique devant ou derrière les ailes, selon la vitesse et l'altitude.

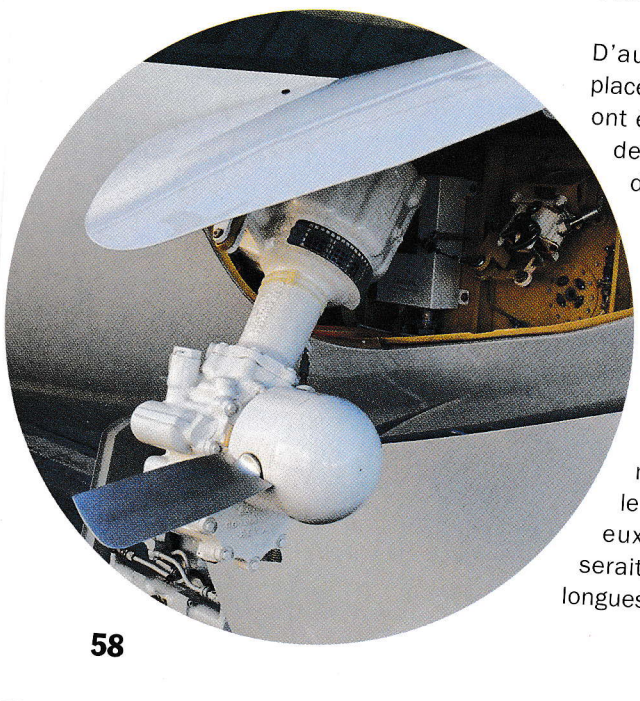
Un compensateur est un s aérodynamique ou mécanique met de maintenir la gouverne ( de contrôle) dans une position tant l'équilibre de l'avion.

Lorsque le compensateur e rectement réglé, l'avion conse trajectoire dans l'air (assiette gage, inclinaison en roulis, dé en lacet). Les efforts sur le m sont annulés, le pilote peut lâch commandes.

Comme nous l'avons menti ci-dessus le compensateur (ou « peut être de 2 types :

- aérodynamique : c'est un face mobile généralement culée sur le bord de fuite gouverne ;
- mécanique : c'est un ressort la tension peut être réglée e est disposé dans la chaîn commande.

De la même façon, en vol, quand l'avion est aux grands angles, la palette se soulève et actionne l'avertisseur qui prévient le pilote de l'approche du décrochage. Un deuxième type de capteur repose sur la pression exercée par l'air sur la surface de l'aile, le capteur est positionné juste au-dessous du bord de l'aile.



Pour de petits angles d'attaque, le bord d'attaque est une zone de basse vitesse et de haute pression, tandis que lorsque l'angle d'attaque atteint la limite du décrochage, il devient une zone de haute vitesse et de basse pression. Dans ce cas, la pression de l'air au niveau du capteur de pression diminue, ce qui déclenche l'alarme dans le cockpit.

D'autres capteurs importants sont placés sur les trains d'atterrissage, ils ont été mis en œuvre après un accident survenu sur un Boeing 757, dans lequel un moteur s'était actionné en sens inverse provoquant la rotation de l'appareil. Aujourd'hui, les inverseurs de poussée (c'est le système qui dirige l'écoulement des gaz du réacteur vers l'avant) inversent la poussée des moteurs après l'atterrissage pour donner, en même temps que les aérofreins, le premier freinage puissant (sans eux, la distance d'atterrissage serait excessive et nécessiterait de longues pistes).

L'inverseur de poussée ne peut activé que lorsqu'un capteur éle mécanique détecte l'appui du t d'atterrissage sur le sol.

## Des tringles aux câbles électriques

Il y a quelques décennies, les c mandes du pilote étaient transmi aux ailerons et aux gouvernes par tringles et des câbles métalliques. nos jours, les commandes du poste pilotage sont électriques, ce sont moteurs électriques qui actionnent surfaces de contrôle.

Normalement, le **contrôle d'un vol** obtenu par **2 commandes primai** qui sont le « **manche** » pour monter descendre ou/et pour effectuer u rotation dans le sens horaire ou a horaire, et la « **manette des gaz** » po commander les moteurs.

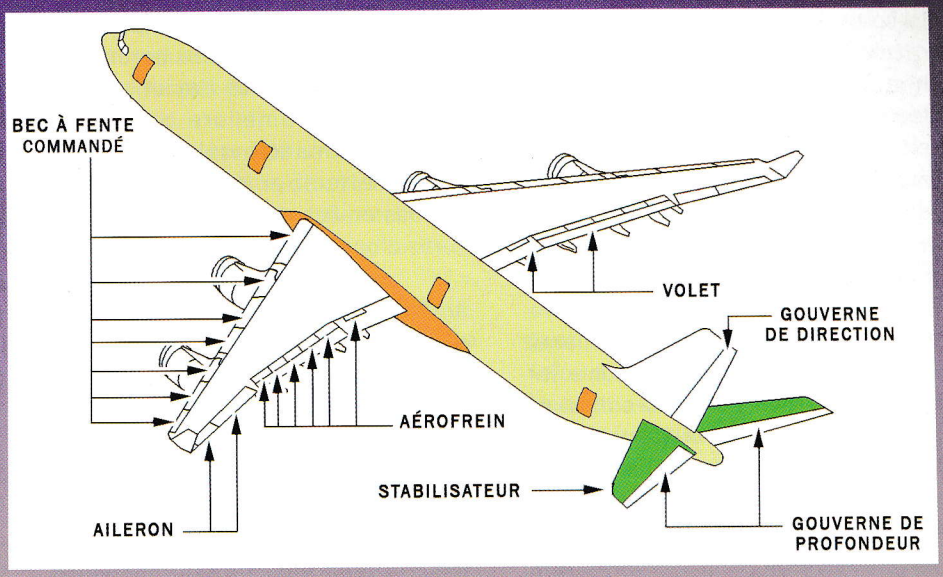
Dans les petits avions et même ce de taille moyenne construits il y a u

Enfin, il existe 3 types de compensation selon les 3 axes :

- en tangage : le compensateur quasiment indispensable est le compensateur de profondeur qui sert à équilibrer l'appareil sur l'axe de tangage (tendance à piquer ou à cabrer). Pour une masse donnée, à chaque vitesse de vol correspond une incidence de vol et donc une position de la gouverne de profondeur. À chaque changement de vitesse, l'incidence change, ainsi il faut compenser à nouveau (ou trimer) pour garder cette nouvelle incidence. Le « trim » correspond à une incidence et donc une vitesse, pour une masse donnée de l'avion ;

- en roulis : le compensateur d'ailerons sert à contrer l'inclinaison sur l'axe de roulis due à un chargement décentré.

- en lacet : le compensateur de direction sert à contrer le couple en lacet dû au souffle hélicoïdal de l'hélice (pour un avion à hélice).




Cependant lorsque le pilote décide de cabrer (tirer vers le haut le nez de l'avion) ou descendre (pointer le nez vers le bas), il agit sur l'aileron dit « élévateur » aussi appelé « gouverne de profondeur » disposé sur les bords de fuite du stabilisateur arrière. L'aileron « élévateur » est mobile, il peut pivoter vers le haut ou vers le bas à l'aide d'articulations. La « gouverne de profondeur » est commandée par le manche. En tirant le manche, le pilote incline vers le haut l'aileron « élévateur » et donc la queue fléchit vers le bas ce qui fait monter l'avion, de même lorsque le manche est poussé, la gouverne de profondeur s'incline vers le bas ce qui fait monter la queue de l'appareil donc l'avion descend.

trentaine d'années, le pilote agit sur le manche et le palonnier pour déplacer les surfaces de contrôle à l'aide de tiges rigides ou de câbles en acier. Avec l'augmentation du poids et de la taille, mais aussi de la vitesse de croisière, cela augmente considérablement l'effort pour le pilote qui doit contrer la résistance à l'air (c'est-à-dire la poussée qui apparait sur les surfaces de contrôle due au mouvement dans l'air) et doit exercer des efforts constants.

Au début, le problème a été résolu par l'amplification de l'effort exercé par le pilote au moyen de leviers démultipliés. Cependant cela impliquait une augmentation de l'excursion de la commande. Pour éviter de recourir à un asservissement hydraulique ou électrique, considéré comme pas suffisamment fiable à l'époque, les concepteurs du McDonnell Douglas MD-80 ont eu recours pour consolider les commandes secondaires (dispositifs hypersustentateurs : volets et becs de bord d'attaque ; ainsi que les aérofreins), qui ne sont utilisées que pendant certaines phases du vol, à une tringle qui était seulement de 10 % par rapport à l'empennage contre 25 % habituellement.

Ce choix a grandement simplifié le système de commandes de vol, mais ils ont dû accepter une réduction de la capacité de la gouverne horizontale. Le MD-80 est toujours un des avions préférés des pilotes de ligne.

Ce principe de commandes secondaires est simple : placées à la queue de l'appareil, lorsque le pilote actionne le manche, il se crée un mouvement dans la direction opposée à celle requise, de sorte que la force exercée par l'air sur les surfaces de contrôles soit plus petite. Par exemple, pour monter le pilote pousse le manche vers le bas. Cela facilite grandement la tâche du pilote, car l'effort sur le manche est plus faible par rapport à une commande classique.

Avec l'arrivée d'avions toujours plus grands, ce système a montré ses limites.

Un pilote aurait difficilement pu contrôler un Boeing 747, sans compter les dimensions des tringles car de la cabine à la queue il y a plus de 70 mètres. Cela serait problématique pour transmettre les mouvements aux surfaces de contrôle, les ingénieurs aéronautiques ont donc adopté des circuits hydrauliques équipés de servovalves électrohydrauliques. C'est un organe à action proportionnelle, dont l'entrée est un courant électrique et la sortie un débit hydraulique (parfois une pression).

Une confusion très répandue est faite entre servovalve et électrovalve. En principe, la désignation d'électrovalve devrait être réservée à l'organe tout ou rien dont l'entrée est un courant électrique et la sortie un débit hydraulique.

Il existe plusieurs types de servovalves correspondant à plusieurs principes différents, mais une servovalve comporte au minimum un moteur électrique et un distributeur hydraulique commandé par ce moteur.

La sortie de la servovalve est reliée au circuit de pression constitué de pistons hydrauliques qui agissent sur les surfaces de contrôle (ailerons, gouverne, etc.). Ce système a été adopté depuis plusieurs années sur les avions de ligne de taille moyenne et de grande taille, comme l'Airbus A300, le McDonnell Douglas DC-10, et le MD-11.

Bien que fonctionnelle, la commande hydraulique a longtemps été controversée lorsque les concepteurs d'avions ont posé le problème du gain de poids et donc de consommation. Ils ont remplacé les pièces métalliques par des matériaux composites (les premiers A300 étaient constitués de boulons en acier et d'autres en composite), ce qui facilitait les commandes partant de la cabine. Cela peut sembler étrange, mais des tringles et des tuyaux en acier qui courent tout le long d'un avion de 50 à 70 mètres de long pèsent des tonnes !

Le premier pas dans la réduction du poids a été l'adoption de commandes de vol électriques car les actionneurs hydrauliques sont, à puissance égale, plus lourds que les moteurs électriques. Ce système s'appelle le « **Fly-By-Wire** » (**FBW**) et consiste à remplacer les **commandes mécaniques** par des **moteurs** (actionneurs) **électriques**. Le premier avion non expérimental à être équipé de commandes de vol électriques est l' « Avro Canada CF-105 Arrow » un intercepteur qui vola pour la première fois en 1958. Le **Concorde fut le premier à exploiter cette technologie dans le domaine du transport commercial**. Par la suite, les commandes numériques ont été expérimentées sur un « F-8C Crusader » appelé « F-8 Digital Fly-By-Wire » le 25 mai 1972. Les premiers avions à en être équipés entièrement de série furent les « F-15 » puis « F-16 » de l'US Air Force au début des années 1970, puis les « Mirage 2000 » en 1978. En 1984 l'**Airbus A320** devient le premier avion commercial dont les **commandes sont entièrement contrôlées par un calculateur**.

La présence de commandes électriques pilotées par un calculateur permet de construire des avions normalement instables (l'avion ne revient pas à une position d'équilibre suite à une perturbation).

La réduction des surfaces stabilisant l'avion sur sa trajectoire permet de réduire le poids de l'avion et par conséquence améliore considérablement la manœuvrabilité. L'avion pourra effectuer des manœuvres rapides en cas de problèmes et ainsi sortir d'une situation difficile.

Après de nombreuses incertitudes, d'autres fabricants ont suivi l'exemple d'Airbus, si bien que les avions d'aujourd'hui ne contiennent plus de câbles en acier avec des systèmes mécaniques pour démultiplier les efforts, les **surfaces de contrôle** (ailerons, gouvernes, volets, etc.) **sont gérées par des signaux électriques envoyés à travers un BUS CAN pour commander les systèmes hydrauliques ou électromécaniques**, comme par exemple dans l'Airbus A318 et A319.

La substitution des commandes mécaniques à l'aide de câbles électriques a permis d'alléger l'avion et de réduire la consommation de carburant, tout en maintenant un ratio acceptable entre les avantages et les risques.

## FBW ou « Fly-By-Wire » : un système, plusieurs développements

Le « **Fly-By-Wire** » (FBW) a été développé de différentes manières, mais il est toujours critiqué, d'autant plus qu'il a été développé pour Airbus. En réalité, remplacer les systèmes mécaniques par des actionneurs électriques commandés à partir du poste de pilotage ne présente pas plus de risques que les systèmes hydrauliques et mécaniques. Pour assurer une sécurité maximale, les circuits électriques sont doublés voir même triplés (par exemple sur les versions les plus récentes comme l'Airbus A340).

Fondamentalement, un FBW fonctionne comme ceci : les commandes sont introduites par des potentiomètres, des encodeurs ou des capteurs magnétiques pour détecter un mouvement. Les signaux qui en sont dérivés sont amplifiés en courant et transmis aux servomoteurs ou servovalves qui entraînent les surfaces de contrôle. Le seul inconvénient du « **Fly-By-Wire** » est l'**absence de retour de force** (« force-feedback ») car sans liaison mécanique entre les commandes et les surfaces de contrôle, le pilote ne perçoit pas l'ampleur de l'effort à effectuer.

**En cas de conditions météorologiques défavorables ou d'une vitesse de vol excessive, certaines manœuvres peuvent compromettre la structure de l'avion et causer des dommages**, par exemple sur les gouvernes. Pour pallier cet inconvénient, diverses solutions ont été imaginées. La première a été pensée pour l'Airbus A320, elle consistait en un système à ressorts insérés dans le manche afin de le raidir lors des déplacements vers les butées ; sans doute une technique très discutable.

Dans un second temps, le **retour de force a été détecté au moyen de capteurs électriques d'efforts** (par exemple, des transducteurs de pression piézo-électriques) placés sur les surfaces de contrôle, en transmettant les signaux de retours correspondant directement aux commandes (au moyen de masses d'inertie ou d'actionneurs électromagnétiques), exercées par le pilote pour lui donner une sensation d'effort. Ce système ressemble à celui utilisé sur certaines consoles de jeux « Wii » à retour de force. Le retour de force électronique est l'une des caractéristiques remarquables des Airbus. L'ordinateur vérifie le caractère « exécutable » (ou réalisable) des commandes. Par exemple, la limite d'excursion de la gouverne de direction avec l'augmentation de la vitesse (sur l'A340 est de ± 31,6° jusqu'à 150 nœuds et de ± 3,5° jusqu'à 350 nœuds). Malheureusement, si l'ordinateur tombe en panne, le pilote est incapable de percevoir l'effort qu'il soumet aux surfaces de contrôle. Ce risque n'existerait pas dans un « FBW » pur, car les capteurs qui surveillent les surfaces de contrôle transmettraient indépendamment au pilote les signaux correspondant aux efforts.

Le « **Fly-By-Wire** » a été développé principalement selon 2 philosophies différentes. Chez Airbus c'est l'**ordinateur qui gère la totalité : non seulement il régit la transmission des commandes, calcule leur faisabilité réelle sur la base des paramètres du logiciel et détermine le retour de force**.

La gestion du vol est confiée à un ordinateur de vol plus 2 en réserve, en cas de défaillance. Il est peu probable que les 3 ordinateurs tombent en panne simultanément, bien que certains incidents impliquant l'Airbus A330 (passés inaperçus) soulèvent de nombreuses questions.

Un **incident grave s'est produit le 18 novembre 2012 sur un Airbus 330 d'Eva Air**. L'avion a plongé avec les commandes de vol bloquées. Les pilotes, après réflexion, ont finalement trouvé une solution impensable ; aggraver volontairement la situation de panne en coupant des systèmes vitaux, espérant ainsi retrouver un pilotage plus ou moins manuel en passant outre les ordinateurs qui avaient planté. Ce jour-là les pilotes ont pu rattraper la situation, **ce qui n'a pas été le cas du vol AF447 en juin 2009**, bien que la cause soit différente.

L'ordinateur principal des Airbus modernes peut corriger les commandes du pilote, ce qui rend plus sûr et plus facile le pilotage de l'avion en allégeant le pilote des tâches répétitives qui peuvent être fatigantes sur des vols longs de nuit. Le degré d'intervention de l'ordinateur peut être réduit jusqu'au contrôle total manuel, bien que dans ce mode les commandes arrivent toujours dans l'ordinateur, mais celui-ci ne les gèrent pas.

Afin **d'éviterqu'en cas de défaillance totale de l'ordinateur, les commandes soient interrompues** et que l'avion devienne incontrôlable, le premier A320, ainsi que toute la série A330 et A340-200, 300 et 500, a été doté d'un « **backup** » c'est-à-dire d'un **doublon** pour le « TRIM » (compensateur) et la gouverne de profondeur. Ces **commandes sont transmises mécaniquement** par une mollette pour le « TRIM » et le palonnier pour les surfaces de contrôle.

Sur les derniers A340-600 et A380, un « backup » des systèmes électriques a été étendu à toutes les commandes principales de direction, de profondeur et des ailerons. Donc, si tous les ordinateurs sont hors service, le pilote a toujours la possibilité de commander les surfaces de contrôle, mais il doit éviter les manœuvres brusques car dans ce cas il n'y a pas de retour de force.

Boeing a adopté le système « **Fly-By-Wire** » avec plus de prudence, en effet sur le « 737 Next Generation », le « FBW » est doublé par un backup mécanique, ainsi que dans les « 777 », « 787 Dreamliner » et le nouveau « 747-8 » afin de consommer environ 10 % de moins que son concurrent direct l'A380. Boeing considère l'ordinateur comme un assistant et que c'est le pilote qui doit gérer le vol.

Par exemple, dans le « 777 », le « FBW » commande les servovalves et contrôle les circuits hydrauliques (un circuit primaire et 2 de secours). L'ordinateur de bord, grâce à ses grands écrans LCD tactiles, sert d'interface pour le pilote qui peut modifier les commandes s'il juge nécessaire d'adopter une autre stratégie de vol en cas de risque de décrochage ou de dommages sur la structure de l'avion (ce qui n'est pas à exclure).

Dans le Boeing 787 une variante numérique à fibre optique du « FBW » a été utilisée. En pratique les commandes ne sont pas câblées directement mais se propagent dans l'avion au moyen d'un réseau local à haute fiabilité (sans collisions des paquets de données). Cependant il existe toujours les commandes directes électriques de « backup », la sécurité est donc garantie.

Cette différence est fondamentale avec le « FBW » d'Airbus, car si les pilotes sont moins fatigués et donc font moins d'erreurs, lorsque l'ordinateur est défaillant l'Airbus peut devenir très difficile à piloter.

En réalité l'ordinateur réduit pendant une grande partie du vol le travail du pilote en lui « disant » ce qu'il doit faire, l'ordinateur exécute alors les commandes tant qu'il fonctionne correctement. Il en résulte une préparation moindre des pilotes en cas de défaillances graves car le fabricant considère ces situations « presque impossibles ». En plus le pilote se trouve dans l'incapacité de gouverner l'avion sans ordinateur dans des conditions de vol particulièrement critiques.

Par exemple le vol AF447, qui était un Airbus A330-200, était considéré comme un avion impossible à faire « décrocher », mais cela a été la cause principale de l'accident qui a fait 225 victimes de 34 nationalités différentes.

Il convient d'ajouter que, pour des raisons économiques, **Airbus a fortement compté sur l'intervention de l'ordinateur pour permettre à un pilote titulaire d'une seule licence de piloter toute une classe d'avions**, par exemple l'A330 et l'A340.

Selon le **rapport du BEA** concernant le **vol AF447**, les copilotes n'avaient pas reçu d'entraînement à haute altitude, à la procédure « IAS douteuse » (indication erronée de la vitesse) et au pilotage manuel. Bien qu'ayant identifié et annoncé la perte des indications de vitesse, aucun des deux copilotes n'a appelé la procédure « IAS douteuse ».

Moins d'une minute après le désengagement du pilote automatique et une tentative de cabrer l'appareil en pilotage manuel, l'avion est sorti de son domaine de vol. Aucune annonce standard concernant les écarts d'assiette et de vitesse verticale n'a été faite. Bien que l'altitude maximale eût été une préoccupation permanente avant la déconnexion du pilote automatique, aucun des deux copilotes n'y fait référence lorsque l'avion atteint cette limite à la suite des actions manuelles.

**L'approche du décrochage a été caractérisée par l'activation de l'alarme puis l'apparition de vibrations caractéristiques au niveau des ailes. Aucun des pilotes ne s'est préoccupé de l'alarme de décrochage** et **aucun n'a formellement identifié la situation de décrochage**. Aucun des 3 membres d'équipage n'a semblé en mesure de déterminer à quelles informations se fier ; les valeurs d'assiette, de roulis et de poussée devaient leur paraître incohérentes avec les valeurs de vitesse verticale et d'altitude.

Cela vient du fait que l'ordinateur ajuste les commandes du pilote pour gérer l'aérodynamisme et toutes les autres caractéristiques de l'avion en vol, lorsque l'ordinateur est défaillant le pilote doit faire face à la réalité physique de l'avion.

Actuellement, d'autres fabricants se tournent vers le « **FBW** ». Le brésilien **Embraer**, 3eme constructeur mondial d'avions, l'adopte dans ses jets civils de la série « **ERJ 145** », ainsi que le canadien **Bombardier** (qui travaille également dans le domaine du transport ferroviaire) l'utilise sur son dernier « **CRJ-1000** ».

Voler dans un avion avec des **commandes électroniques peut être inquiétant**, car les **circuits de bord peuvent se retrouver sans alimentation** (à cause par exemple d'un manque de carburant ou d'un défaut des générateurs électriques).

Il existe une **unité auxiliaire** appelée l'« **APU** » (« **Auxiliary Power Unit** » ou « **Unité de Puissance Auxiliaire** ») qui est un groupe auxiliaire (turbogénérateur en général) destiné à **produire de l'énergie** à bord des avions afin de permettre le fonctionnement au sol des différents systèmes de bord (tensions électriques, pressions pneumatiques et hydrauliques, climatisation) quand les moteurs sont arrêtés (économie de carburant). Elle peut être ou non utilisée en vol.

Les « APU » sont généralement positionnées à l'arrière de l'avion, dans le cône de queue et alimentées par le kérosène des réservoirs. Elles produisent aussi l'énergie pour démarrer les moteurs.

Si l' « **APU** » tombe en panne, les avions modernes sont dotés d'une petite **éolienne** « **RAT** » (« **Ram Air Turbine** ») placée en dessous ou sur le côté du fuselage et qui exploite l'écoulement de l'air le long de l'avion pour entraîner un alternateur suffisant pour recharger les batteries et faire fonctionner le « **Fly-By-Wire** ».

La « **RAT** » a également été installée sur les avions traditionnels tels que le **Boeing 767** et a **sauvé la vie de passagers**. Le vol **143** d'**Air Canada** en a été l'exemple. Surnommé le « **Planeur de Gimli** », c'était un vol commercial d'un Boeing 767 le 23 juillet 1983. Il s'est retrouvé à **court de carburant** à **12 000 mètres** au-dessus du centre du Canada et a plané pour faire un atterrissage forcé sans moteur, jusqu'à une ancienne base aérienne militaire à Gimli.

Le 767, au lieu d'instruments mécaniques à cadrans, utilise des écrans électroniques pour l'affichage des informations de vol. Ce principe fonctionne très bien la plupart du temps, mais comme les moteurs de l'avion fournissaient aussi l'électricité, à l'arrêt des moteurs la plupart des instruments s'éteignirent.

Cependant Boeing avait prévu ce cas de figure et avait inclus dans l'avion la « Ram Air Turbine » sur le côté de l'appareil. Elle a pu entraîner la pompe hydraulique pour commander les gouvernes et atterrir sans problèmes.

## Sans tringles et sans câbles électriques, il est appelé le « Fly-By-Wireless »

En 2010, un institut de recherche Portugais a expérimenté une variante particulièrement audacieuse du « FBW ». Ils ont considéré que les câbles électriques, bien que moins lourds que les systèmes mécaniques, ont quand même une certaine masse.

Les chercheurs ont suggéré la possibilité de transmettre les commandes du « FBW » par radio. Le système a été baptisé « **Fly-By-Wireless** » (Fly-By sans fil) et s'est inspiré du réseau de fibre optique du 787.

Cependant, il utilise un réseau sans fil, pour envoyer les ordres aux surfaces de contrôle et recevoir les informations des capteurs. Naturellement l'avion est pour l'instant seulement un modèle télécommandé, vu qu'il y a trop d'interrogations sur la fiabilité de cette technologie.

En effet, tandis que le réseau filaire du « FBW » classique est un moyen sûr pour la transmission des informations, un réseau sans fil a une fiabilité limitée, en raison de la sensibilité de la liaison radio aux interférences électromagnétiques (par exemple celles qui sont dues au téléphones mobiles à l'intérieur de l'avion qui pourraient bloquer ou ralentir la transmission de commandes ou encore aux éruptions solaires qui engendrent un fort champ électromagnétique dans l'atmosphère terrestre).

## Les systèmes de navigation

L'électronique a révolutionné non seulement les contrôles aérodynamiques des avions, mais aussi le contrôle du trafic aérien.

Il y a quelques années, les avions étaient équipés de systèmes de radiocommunications aéronautiques assez simples, pour l'identification et la communication. Les **conversations entre les pilotes et les centres de contrôle au sol** (mais aussi entre les pilotes de différents avions en vol) étaient réalisées en **VHF** dans la gamme des 110 à 130 MHz. L'**identification était obtenue par un transpondeur actif c'est-à-dire** un dispositif électronique qui émettait un **signal radio contenant le code d'identification** de l'avion lorsqu'il recevait une interrogation par radio (il a été utilisé la 1ere fois pendant la Seconde Guerre Mondiale pour compenser l'incapacité du radar à distinguer les avions ennemis des alliés).

Les transpondeurs des avions modernes fonctionnent en réception à une fréquence de 1,03 GHz, et répondent sur une porteuse à 1,09 GHz. La **portée du système est de 200 miles nautiques soit environ 370 km** (1 mile nautique = 1,852 km).

Le **transpondeu**r permet de réaliser le **radar d'anticollision** ou « **SSR** » (radar secondaire de surveillance) dérivé du système militaire « **IFF** ». Son principe est basé sur les modes d'interrogations et des réponses associées retransmises par les transpondeurs embarqués.

Le système se compose essentiellement de :

- au sol, d'un interrogateur associé au radar primaire du contrôle aérien (antennes liées mécaniquement) ;
- à bord, d'une antenne omnidirectionnelle associée au répondeur.

Interrogé sur un mode déterminé, le répondeur de bord (ou transpondeur) répond par une suite d'impulsions HF qui forment un code destiné à

être décodé par l'équipement au sol, délivrant ainsi l'information désirée. Les **signaux d'interrogations sont constitués** d' une **paire d'impulsions P1** et **P3,** de fréquence **1030 MHz** et de puissance crête de **1** à **2 kW**. L'espacement entre ces deux impulsions détermine le mode d'interrogation selon la procédure suivante :

1. signal d'interrogation en mode A/C : le signal « réponse » est constitué d'un train d'Impulsions de fréquence 1090 MHz comprenant 2 impulsions d'encadrement espacées de 20,3 µs toujours présentes et de 12 impulsions espacées de 1,45 µs, qui sont présentes ou non suivant le code utilisé. L'impulsion spéciale de positionnement « Special Position Indicator ou SPI » est transmise à la demande du contrôleur ;

2. signal de réponse en mode A/C : les principaux modes d'interrogation et de réponses associées sont les modes 1, 2, 3/A, C, et le plus récent, le mode S (sélectif), permet par un protocole de communication différent un échange plus riche en informations.

Au radar d'anticollision a été ajouté le récepteur « **VOR** », capable d'**interagir avec le pilote automatique** ou tout simplement d'afficher sur un écran la route et les écarts éventuels. Le « **VOR** » (abréviation de « **VHF Omni-directional Range** ») est un **système de positionnement radioélectrique** utilisé en navigation aérienne et fonctionnant avec les fréquences VHF (ou UHF pour les militaires).

Un récepteur « VOR » permet de déterminer un relèvement magnétique par rapport à une station au sol (balise émetteur « VOR » dont la position est connue), et donc la position radiale sur laquelle le récepteur (donc l'avion) est situé.

Par déduction il permet de suivre n'importe quelle route passant par la station (en rapprochement ou en éloignement de celle-ci), ou même de déterminer la position exacte de l'avion en utilisant deux balises « VOR ». Rappelons que la route magnétique est l'angle orienté du Nord magnétique vers la trajectoire que suit l'avion.

Par exemple, la route d'un avion se déplaçant vers l'est est égale à 90°. Le signal émis va de 108 à 118 MHz. La porteuse, qui contient le nom de la station en code Morse, est modulée en amplitude (AM) par une sous-porteuse à 9 960 Hz elle-même modulée en fréquence par un signal de 30 Hz. L'ensemble du signal obtenu est modulé en amplitude (30 %) par un autre signal de 30 Hz. Le fait d'avoir choisi la même fréquence de 30 Hz permet de faire une comparaison de phase entre les deux signaux informatifs.

Cette phase est l'image directe du relèvement de la balise. Elle peut évoluer de 0° à 360°. Pour ce faire, on emploie l'un des signaux comme référence. Celui-ci est émis de manière omnidirectionnelle. Le second est quant à lui dit « variable », car il est émis de manière directionnelle tournant avec une vitesse de rotation de 30 t/s.

Connaissant la vitesse de rotation de la phase et de la différence de phase, le récepteur « VOR » est en mesure de déterminer à tout instant quel est l'angle de l'avion par rapport à la balise.

Souvent les stations au sol sont également équipées de « **DME** » (**Distance Measuring Equipment**), qui mesure la distance entre l'émetteur et le récepteur, et permet d'évaluer si l'avion se déplace vers la station ou s'en éloigne. Le « **VOR** » fonctionne dans un **rayon variant** de **46** à **240 km** et sur des fréquences valables dans le monde entier entre 108 MHz et 117,95 MHz avec une largeur de canal de 50 kHz.

En utilisant le bouton appelé « **OBS** » (**Omni-Bearing Selector)** ou **sélecteur d'azimut**, le pilote peut **réinitialiser l'angle d'une station**. Par exemple, s'il veut voler à 90° par rapport à une station, il réinitialise l'« OBS » de sorte que le signal local soit déphasé de -90°, ainsi l'indicateur marque le « zéro centré » lorsque l'avion vole à 90° de la balise. Pour suivre une direction, il suffit de faire en sorte que le déphasage reste constant, c'est-à-dire diriger l'avion de sorte que l'indicateur pointe toujours dans la même direction.

En continuant avec les équipements électroniques de navigation, nous ne pouvions pas ne pas vous parler de l'« **ILS** » (**Instrument Landing System**) ou système d'atterrissage aux instruments basé sur des stations terrestres qui émettent des signaux directionnels de guidage.

Il comprend 2 éléments :

- un « **localizer** » (LOC) qui fournit l'écart de l'avion par rapport à l'axe de la piste ;
- un « **glide path** » qui fournit l'écart de l'avion par rapport à la pente nominale d'approche (le plus souvent 3°).

Ces 2 informations sont fournies soit sous forme d'aiguilles sur un indicateur VOR, soit sous forme d'index (barres, triangles, ...) sur 2 échelles, l'une horizontale, l'autre verticale, situées de part et d'autre de l'horizon artificiel. Les signaux « localizer » et « glide path » étant très sensibles aux perturbations, leur interception doit toujours être validée à l'aide d'une autre source de navigation.

Pour le « localizer » cela peut être réalisé à l'aide du « VOR ». Pour la validation du « glide path », on utilise une ou deux balises à émission, remplacées de plus en plus par un DME (Distance Measuring Equipment) dont l'avantage est de fournir une information de distance en continu. Le DME est le plus souvent implanté avec le « glide path », donnant ainsi directement la distance au seuil de piste, ce qui est très pratique.

Un voyant lumineux et un signal sonore sont activés au passage de chacune des 2 balises. L'information de distance DME est quant à elle fournie par l'afficheur DME.

D'un point de vue pratique, le pilote affiche une seule fréquence, celle du « localizer », comprise dans la gamme VHF 108 MHz à 111,975 MHz.

Les fréquences « glide path » et DME sont dans des gammes de fréquences différentes (UHF) mais appariées à celle du « localizer », ce qui reste transparent pour le pilote.

# Fly-By-Wire : comparaison Airbus et Boeing

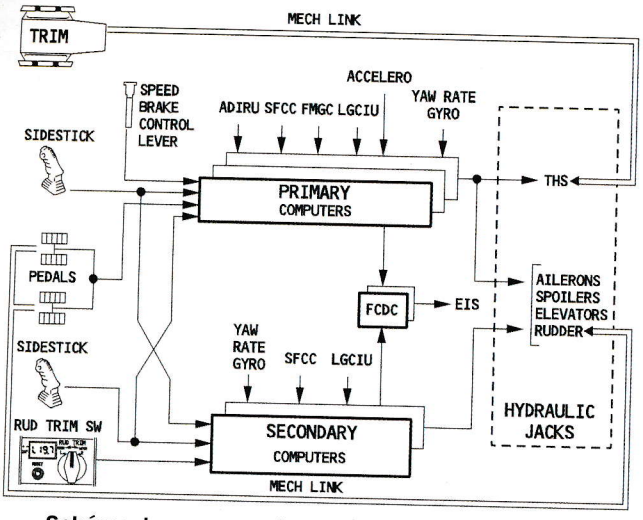
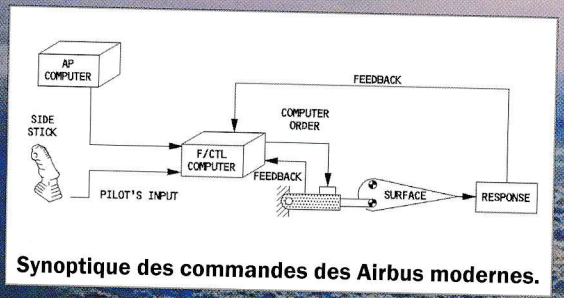



**Schéma des commandes du FBW de l'Airbus A340.**

La vraie révolution dans le monde aéronautique a été l'introduction de l'électronique et plus particulièrement le « Fly-By-Wire », qui consiste à transmettre les commandes aux surfaces de contrôle (gouvernes, ailerons, etc.) au moyen de signaux électriques. Salué par les partisans de la réduction du poids mais critiqué par ceux qui y voient le début de la fin de l'art du pilotage, la transition fut obligatoire pour réduire le poids de l'avion et donc la consommation. En particulier dans les avions ayant de grandes dimensions, l'utilisation de systèmes (dupliqués) avec des tringles et des câbles en acier de plusieurs dizaines de mètres de la cabine aux gouvernes et ailerons, augmente inutilement le poids.

Pour réaliser un « Fly-by-Wire », il faut disposer de suffisamment de capteurs sur les surfaces de contrôle et renvoyer les informations dans la cabine, ensuite le pilote effectue les commandes nécessaires en envoyant des signaux électriques aux actionneurs situés à proximité des surfaces de contrôle. Pour les manœuvrer, les constructeurs sont allés encore plus loin : en particulier Airbus, qui a adopté son système dans l'A320. Il a considéré qu'un système électrique pouvait interagir facilement avec un ordinateur et a donc développé des « gouvernes informatisées » sur ses avions. Dans la pratique les Airbus 318, 319, 320, 321, 330, 340 et 380 en sont équipés, à la place du classique volant en forme de demi-cercle, , il s'agit d'un joystick sur le côté du siège de chaque pilote appelé « mini-manche ». Il est doté d'un amortisseur électromagnétique qui fournit le retour de force pour le pilote. Il contrôle la gouverne de profondeur et les ailerons alors que la gouverne de direction est toujours actionnée par le palonnier.




**Synoptique des commandes des Airbus modernes.**

Les commandes provenant du « mini-manche » fournissent des signaux électriques qui sont envoyés à un ordinateur appelé « Flight Control Data Concentrator » (FCDC).

L'ordinateur « FCDC » est composé de 7 calculateurs de commandes de vol traitant les ordres du pilote ou de l'auto-pilot selon les lois suivantes :

- 2 ELAC (Elevator Aileron Computer) assurent la commande normale de la profondeur et du THS, ainsi que la commande des ailerons;
- 3 SEC (Spoilers Elevator Computer) assurent la commande des Spoilers et la commande de secours de la profondeur et du THS;
- 2 FAC : (Flight Augmentation Computer) assurent la commande électrique de la direction;
- de plus 2 FCDC (Flight Control Data Concentrator) sont chargés d'acquérir des données venant des ELAC et des SEC pour les envoyer à l'EIS (Electronic Instruments System) et au CFDS (Centralized Fault Data System).

L'ordinateur, en grande partie redondant, traite normalement les demandes du pilote et, sur la base des données issues de l'acquisition des paramètres de vol de l'ADIRU (Air Data Inertial Reference Unit) détermine le domaine (ou l'enveloppe) de vol optimal. C'est l'espace en vitesse air, facteur de charge et altitude à l'intérieur duquel l'avion peut fonctionner en sécurité. En abscisse on retrouve la vitesse qui va de zéro à la VNE (Velocity Never Exceed ou vitesse à ne jamais dépasser), et en ordonnée le facteur de charge. Si l'avion dépasse par sa vitesse ou son facteur de charge son domaine de vol, sa structure sera endommagée voire détruite et donc l'ordinateur corrige les commandes provenant du pilote en cas d'erreur.

De plus les différents calculateurs coordonnent le mouvement des ailerons et des gouvernes (auto-YAW) et règlent la fonction « TRIM » de répartition de la charge aérodynamique, ainsi que l'altitude et la vitesse.

Par exemple, si le pilote demande un cabrage à 40° et que l'ordinateur retient qu'à la vitesse actuelle cet angle pourrait compromettre la structure de l'avion, il définit un angle de montée optimal en ignorant la demande du pilote.

Cette méthode, également appelée « Normal Law » (Loi normale) est en vigueur lors des vols dans des conditions normales. Il existe un autre niveau de protection réduite, appelé « Alternate Law » (Loi alternative), dans lequel le pilote peut exclure certaines protections lorsque les capteurs fournissent des signaux contradictoires ou absents.

Un exemple typique a été malheureusement le cas du tube de « Pitot » (ou sonde Pitot) du vol AF447, dont le signal peu fiable a bloqué l'ADIRU et l'ordinateur a abandonné les commandes en les « redonnant » au pilote, qui s'est retrouvé en contrôle direct (Direct Law). Plusieurs Airbus A330 et A340 ont manifesté ces problèmes et, dans certains cas il y a eu une fin tragique. Notez qu'il y a 3 « ADIRU » en cas de défaillance.

Dans le cas peu probable de défaillance totale de l'ordinateur, les commandes primaires deviennent inefficaces. Pour éviter ce scénario, des commandes de « backup » (sauvegarde) ont été prévues sur les premiers modèles d'A340, la fonction « TRIM » et la commande des gouvernes étaient actionnées mécaniquement sauf si l'avion était en « Normal Law ».

Dans les nouveaux modèles d'Airbus A340-600 et A380, toutes les commandes principales à savoir la gouverne de direction et la « TRIM » sont mécaniques, tandis que les ailerons et la gouverne de profondeur ont des circuits électriques de secours qui peuvent se passer de l'ordinateur en cas d'avarie. En théorie, pour atterrir dans des conditions météorologiques normales, il suffit de disposer des commandes « TRIM » et de la gouverne de direction.

Les doutes soulevés par le système d'Airbus sont nombreux, tout d'abord en ce qui concerne la formation des pilotes : selon certains, ils ne sont pas préparés à faire face à de telles situations.




L'accident du vol AF447 a soulevé le problème et le rapport final du BEA (Bureau d'Enquêtes et d'Analyses) recommande d'inclure dans la formation des pilotes la simulation de vol dans des conditions météorologiques défavorables sans l'aide de l'ordinateur de vol et avec l'ADIRU en avarie, comme cela a été le cas dans l'accident. Sans compter que l'aide de l'ordinateur permettait de piloter des avions différents avec une seule licence. Ce fut l'un des points forts de la politique commerciale d'Airbus avec un coût réduit pour les A330 et A340 qui dérivent de l'A300 avec un fuselage allongé et des moteurs et des ailes différents. Ceci réduit les coûts de conception (Boeing repartait à zéro pour chaque appareil) et de formation, car l'avion était plus facile à piloter en partie géré par l'ordinateur. La question est que se passe-t-il lorsque le pilote doit faire face à des situations critiques ?

L'approche de Boeing est plus prudente, il a introduit son premier « FBW » avec le « 777 », qui est l'un des modèles les plus populaires dans le monde grâce à sa faible consommation d'énergie qui est due (l'avion est conforme aux normes « ETOP » et peut être utilisé dans des vols transocéaniques avec seulement 2 moteurs) à ses 2 moteurs imposants « General Electric GE90 », ce qui le rend moins cher que le quadrimoteur A340-600 (Airbus travaille actuellement sur son nouvel A350 bimoteurs pour reprendre les parts de marchés perdues).

Le système « FBW » de Boeing permet d'assister les pilotes et corriger les commandes, cependant il peut être complètement désactivé, alors que dans l'Airbus les commandes entrent dans l'ordinateur et les contrôles y ressortent. Dans le Boeing, les contrôles « FBW » sont simplement une assistance et l'ordinateur peut être complètement désactivé sans risques.

Ainsi, le « Fly-By-Wire » de Boeing est une sorte de pilote automatique évolué, rien de plus. En outre, Boeing comme tous les autres constructeurs qui ont adopté le système « FBW », a maintenu le volant traditionnel qui pour de nombreux pilotes est plus facile à manipuler en cas de fortes turbulences car ils peuvent le saisir avec les deux mains et qu'il centré par rapport à l'assise. Ce n'est pas le cas du « mini-manche » qui est décalé par rapport au pilote et qu'il ne peut manipuler que d'une seule main.

Pourtant, sur le « FBW » du Boeing les conditions de vitesses indiquées anormales (« unreliable IAS ») sont simplement signalées au pilote, tandis que dans l'Airbus l'ordinateur est déconnecté et l'avion passe en mode « Direct Law ». Avant cela, le « FBW » de l'Airbus peut également modifier les données du vol.

NB : l'IAS ou Vi est la vitesse indiquée par l'instrument de mesure anémobarométrique d'un avion (le tube de « Pitot » et le badin ou anémomètre), corrigée des effets de compressibilité en conditions atmosphériques standards au niveau de la mer. Les erreurs proviennent principalement de la mesure de pression statique, de l'écoulement de l'air autour de l'avion perturbant la mesure.

## La gestion des moteurs par le FADEC (Full Authority Display Engine Control)

Les moteurs sont une partie essentielle d'un avion, ils permettent de décoller, d'assurer la vitesse de croisière et de maintenir la vitesse au-dessus de la limite du décrochage. Cependant, ils sont délicats et exigeants, il ne suffit pas de mettre les gaz au maximum avec l'auto-manette, comme dans une voiture la pression d'huile. La vitesse, la température doivent être vérifiées en permanence avec d'autres paramètres. Gérer tout cela surtout sur un avion quadrimoteur est une tâche trop importante pour le pilote, c'était le rôle de l'ingénieur de vol.

Par conséquent, les constructeurs ont développé des systèmes électroniques pour contrôler et optimiser le fonctionnement des moteurs pendant le vol. Ainsi sont nés les premiers systèmes de gestion qui géraient les paramètres par rapport à la puissance demandée par le pilote et signalaient tout problème. Le premier système était un contrôle analogique commandant des actionneurs pour la variation des paramètres de fonctionnement du moteur.

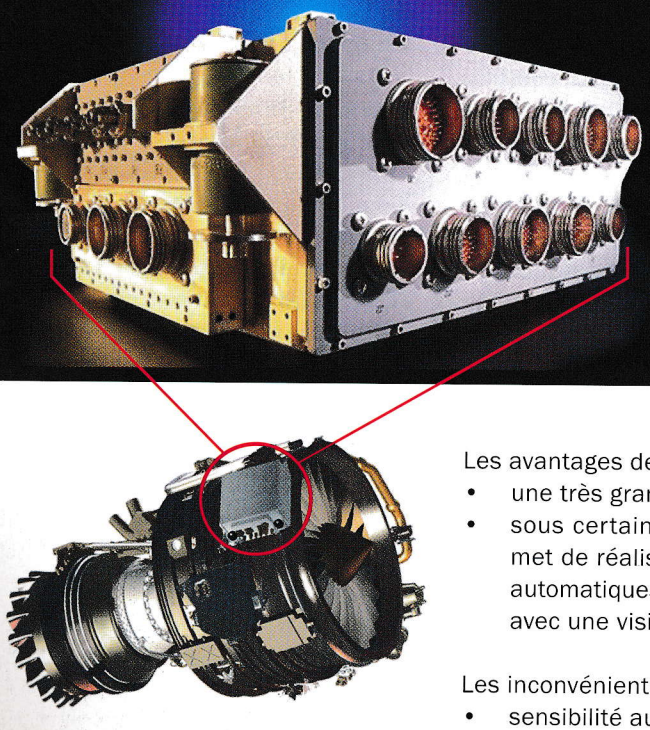
Il a été utilisé dans les années 60 sur les moteurs « **Rolls Royce Olympus 593** » installés sur le « **Concorde** ».

Pour surmonter la sensibilité des systèmes électroniques analogiques aux interférences, la NASA et Pratt & Whitney (l'un des principaux fabricants de moteurs à réaction) ont conçu un système nommé « FADEC » qui est encore utilisé aujourd'hui. Sa fonction principale est de contrôler et réguler la pompe à carburant afin d'injecter le carburant nécessaire dans le moteur pour un fonctionnement optimal.

Le FADEC mesure la température des gaz d'échappement, la position relative de la manette des gaz et la pression dans la chambre de combustion.

Il gère tous les contrôles nécessaires pour garantir le fonctionnement des moteurs dans les limites des paramètres normaux ainsi que l'arrêt du moteur en cas de défaillance détectée.

Pour la mesure des paramètres, le FADEC dispose de sondes de pressions, de thermocouples, de capteurs de positon de la manette des gaz, de convertisseurs A/D pour la tension d'alimentation de la pompe à carburant et un port de communication pour le programmer et lire en temps réel les données.



Les mesures effectuées par le FADEC sont :

- la pression de la chambre de combustion ;
- la température des gaz d'échappement (EGT) ;
- la tension de la batterie de la pompe à carburant ;
- la position de la manette des gaz transmise par émetteur radio ;
- la température interne du FADEC ;
- le nombre de tours minute (tr/min) du moteur.

Toutes les mesures et anomalies sont mémorisées et peuvent être lues et/ou affichées sur le terminal d'acquisition de données (HDT) connecté au FADEC lors de l'entretien. Le FADEC est programmable par le pilote afin d'adapter la marge d'opération à chaque moteur. Les paramètres programmables sont :

- la pression maximale à pleine puissance ;
- la pression au ralenti ;
- la pression minimale (arrêt moteur) ;
- la température maximale ;
- la température minimale du démarrage moteur ;
- la position de la manette des gaz à pleine puissance ;
- la position de la manette des gaz au ralenti ;
- la position de la manette des gaz lorsque le moteur est arrêté ;
- la vitesse d'accélération/décélération du moteur ;
- le point de démarrage de la pompe à carburant ;
- la pente de démarrage du moteur.



Les avantages de l'« ILS » sont :

- une très grande précision ;
- sous certaines conditions il permet de réaliser des atterrissages automatiques et donc de se poser avec une visibilité très faible.

Les inconvénients de l'« ILS » sont :

- sensibilité aux perturbations ;
- existence occasionnelle de faux axes par réflexion sur le relief ;
- signaux étroits nécessitant une aide pour la capture.

Aujourd'hui pour réduire les coûts de ces systèmes et permettre un atterrissage en toute sécurité dans les aéroports dépourvus d'« ILS », les ingénieurs testent un **système d'approche par satellite** (**GLS**).

Avec ces paramètres, les données du moteur et la position de la manette des gaz, le FADEC ajuste la puissance de la pompe à carburant comme suit : en premier lieu, le FADEC vérifie que la largeur des impulsions (trame) émises par l'émetteur radio est correcte. Si l'impulsion n'est pas correcte ou n'est pas reçue pendant 0,5 seconde, il arrête le moteur. Pendant ce délai, le système suppose que la manette des gaz est au ralenti et décélère le moteur vers cette position.

Ensuite, le système calcule la position relative de la manette des gaz, déterminée par les limites programmées par le pilote. Ce calcul donne une valeur entre 0 % et 100 %. Cette valeur est transformée en valeur de pression équivalente aux valeurs de pression maximale (pleine puissance) et pression minimale (au ralenti) programmées par le pilote.

Par exemple, si le pilote programme une pression pleine puissance de 0,7 bar, cette pression sera atteinte avec la position de la manette des gaz à 100 %. Avec la manette des gaz à 50 %, la pression équivalente serait 0,35 bar. Cette pression est comparée à la pression du moteur et la puissance de la pompe est ajustée pour atteindre la pression désirée de 0,35 bar.

Dans le cas où la pression lue est supérieure à celle programmée, le système va baisser la puissance de la pompe jusqu'à atteindre la pression désirée.

La vitesse de décélération est un paramètre programmable par le pilote. S'il doit accélérer le moteur, le système contrôle la température des gaz d'échappement et réduit l'accélération si la température approche la température maximale programmée. A la température maximale, le système réduit le courant dans la pompe à carburant jusqu'à amener le moteur dans les limites de fonctionnement normal et arrête la pompe s'il ne peut pas baisser la température. Ce système garantit au moteur l'accélération dans un temps minimum possible sans dépasser la température ou la pression maximale et l'adapte aux variations du moteur, de température ambiante, de pression du carburant, etc.

Avec ce système, la poussée délivrée par le moteur, en fonction de la position de la manette des gaz, est indépendante du type de pompe, de batterie ou de moteur. Cet avantage est très utile sur des avions à plusieurs moteurs car la poussée des moteurs est toujours identique.

Le système incorpore aussi 2 protections supplémentaires :

- la première est l'arrêt de la pompe à carburant dans le cas où la température des gaz d'échappement est inférieure à la valeur minimale programmée, afin de protéger le moteur contre le remplissage de carburant non brûlé en cas d'extinction de la flamme ou d'une panne de thermocouple ;

- la seconde est l'arrêt de la pompe à carburant dans le cas où la pression du compresseur est inférieure à la valeur minimale programmée afin de protéger le moteur dans le cas où la pression/vitesse du rotor est inférieure à celle exigée pour que le moteur soit capable de fonctionner.

Le FADEC est aussi doté d'un système de démarrage semi-automatique. A chaque démarrage du moteur, les circuits d'autodiagnostics vérifient le fonctionnement du FADEC et de ses capteurs. Pour démarrer, le pilote doit positionner la manette des gaz au minimum. La LED s'éclaire en indiquant que le système est « prêt pour démarrer », ensuite le pilote doit :

1. lancer le moteur avec de l'air comprimé ;
2. ouvrir le gaz et allumer la bougie de démarrage. Quand le FADEC enregistre une température des gaz d'échappement plus élevée que la température minimale programmée, la LED commence à clignoter ;
3. le système commence à envoyer le carburant au moteur et augmente la puissance de la pompe à carburant jusqu'à ce que la pression/vitesse « ralenti » soit atteinte. Cette phase finale est signalée par le système qui éteint la LED. La puissance minimale de démarrage de la pompe et la vitesse d'accélération dans la phase démarrage sont programmables par le pilote.

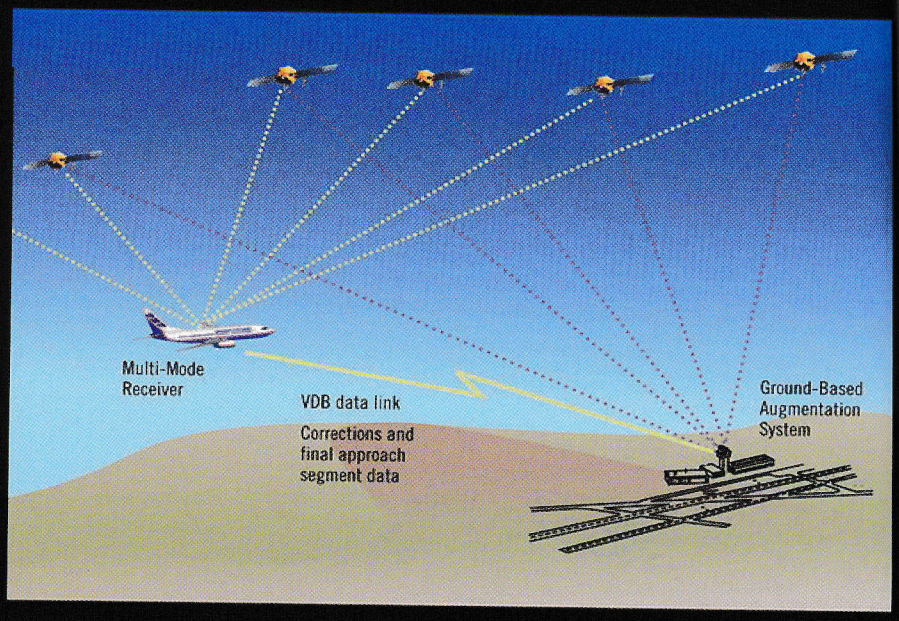
Dans la pratique, chaque FADEC est composé de 2 unités complètes appelées « ECU » dans les moteurs General Electric et « EEC » chez Pratt & Whitney. Elles travaillent indépendamment l'une de l'autre. Normalement, un canal pilote le moteur, tandis que l'autre vérifie le fonctionnement correct. Si le canal actif (ou un sous-système) connait une avarie, c'est l'autre canal qui prend le contrôle du moteur.

Récemment, il a également été lancé le test de l'« **ADS-B** » (**Automatic Dependent Surveillance Broadcast**) qui sera en mesure de combiner les fonctionnalités de plusieurs instruments, à la fois sur terre et à bord de l'avion.

L'« **ADS-B** » **détermine la position de l'avion grâce à un système de positionnement par satellite** (**GNSS**) et envoie périodiquement cette position et d'autres informations aux stations au sol et aux autres avions équipés de l'« ADS-B » évoluant dans la zone. Les données obtenues sont affichées sur les moniteurs des centres de contrôle aériens, dont les contrôleurs évaluent la distance entre 2 avions, l'altitude de vol et l'itinéraire dans les voies aériennes prédéfinies par le plan de vol.

Avec le perfectionnement du GPS civil et en particulier l'« **EGNOS** » (et son équivalent américain le « WAAS »), il est possible en installant un récepteur à bord de l'avion de connaître exactement sa position et son altitude. Si les données sont envoyées à une station au sol, on obtient la même information que le radar, mais avec une précision plus grande.

# GLS : l'atterrissage en toute sécurité avec Galileo




Lorsque les conditions de visibilité empêchent l'atterrissage en mode manuel en toute sécurité, le pilote utilise le système d'atterrissage aux instruments « ILS » lorsqu'il est disponible. Aujourd'hui, avec la certification « Safety Of Life » du signal « EGNOS » du projet Galileo, le but est d'intégrer l'« ILS » dans un guidage par satellite. L'« ILS » est basé sur une transmission au sol fonctionnant dans les bandes VHF entre 108,10 MHz et 111,95 MHz et UHF entre 328,6 MHz et 335,40 MHz, dont les signaux radio créent un « chemin d'accès » afin que l'avion ait une approche avec un angle de descente lui permettant de toucher le sol en début de piste.

Sur le terrain, le système se compose de 3 dispositifs appelés « LOC », « GP » et « MARKER ». Le « MARKER » (marqueur) est situé dans le prolongement de l'axe de la piste sous la trajectoire prévue de l'avion. Il émet un signal directif très puissant. L'antenne la plus éloignée (Outer Marker) est située entre 6 et 11 km en amont du début de piste, au milieu (Middle Marker) il y a une autre antenne entre 900 et 1 200 m, tandis qu'un 3eme marqueur (Inner Marker) se trouve en début de piste.

À bord de l'avion, il y a un appareil composé d'un récepteur multifréquence et d'un cadran qui indique si l'avion dérive latéralement et/ou horizontalement par rapport à la trajectoire de descente idéale. Les survols des marqueurs sont indiqués par des signaux sonores et lumineux.

Il existe 3 catégories d'« ILS » (CAT I, CAT II et CAT III) qui définissent une précision garantie au sol. La 3eme catégorie est divisée en 3 sous-catégories (CATIIIA, CATIIIB et CATIIIC) dont la dernière permet l'atterrissage dans des conditions de visibilité nulle.

Aujourd'hui est expérimenté un système basé sur une approche de guidage par satellite appelé « GLS » (GPS Landing System). Le but est de permettre l'atterrissage d'un avion en toute sécurité dans les endroits où il n'y a pas d'« ILS » pour des raisons économiques. Le système exploite la localisation par satellite grâce au système « Galileo », sur lequel l'ADS-B est également basé, qui permet de garantir une grande précision dans la localisation.



Depuis plusieurs mois, le signal du système de correction d'erreur de positionnement européen « EGNOS » (European Geostationary Navigation Overlay Service) a été certifié comme un « ILS CATI » et testé lors d'un atterrissage. Le fonctionnement du « GLS » est très simple, des récepteurs GPS sont installés à proximité des pistes, dans des endroits précis. La réception du signal permet d'évaluer la position fournie et de la comparer à la réalité afin de déterminer l'écart et le facteur de correction à appliquer aux données du GPS. Le signal d'erreur est transmis par une porteuse VHF à tous les avions à proximité de l'aéroport, les instruments de bord l'utilisent pour corriger les données fournies par le récepteur GPS embarqué, obtenant ainsi une précision inférieure au mètre, ce qui permet un atterrissage dans des conditions de visibilité nulle.

Un tel système est moins cher qu'un « ILS » parce qu'il requiert des équipements au sol et en vol moins coûteux ; il est également plus fiable et nécessite moins d'entretien.

Dans un avenir proche, les « GLS » pourraient permettre aux avions d'atterrir de manière entièrement automatique, comme cela est le cas pour les drones (UAV).

Le système qui en résulte prend le nom de « ADS-B » (Automatic Dependent Surveillance-Broadcast) et peut être utilisé pour transmettre à la fois au sol et en vol aux autres avions équipés du même dispositif, des données telles que l'identifiant, la position, la trajectoire, le type d'avion, la compagnie aérienne, le numéro de vol, et plus encore.

Une démonstration intéressante du système « ADS-B » peut être suivie sur le site www.flightradar.com, dont le serveur reçoit les signaux des récepteurs du monde entier et décrypte les données, en fournissant les positions mises à jour d'avions volant dans une certaine zone.

La nécessité d'utiliser des systèmes d'assistance de vol basés sur des satellites se pose principalement pour faire face à l'augmentation du trafic aérien, mais aussi parce que les systèmes actuels de contrôle de la circulation aérienne ne couvrent pas la totalité du globe (en raison de la portée limitée du radar ou l'absence d'infrastructures dont souffrent par exemple les vols transatlantiques).

En outre, le prix d'un équipement « ADS-B » dans un avion est de l'ordre de quelques milliers d'euros, de sorte que la technologie puisse être adaptée sur de petits avions de tourisme.

Dans le contrôle au sol, c'est-à-dire la gestion des mouvements des avions dans un aéroport, l'« ADS-B » peut être très utile. Il remplacera dans le futur les systèmes radars de contrôles au sol utilisés pour éviter les collisions dans des conditions de mauvaise visibilité, telles que par exemple l'accident de **Milan Linate** où un **MD-82 a heurté un petit avion privé** et a fini dans les flammes sur un hangar, ou la terrible catastrophe de **Tenerife où 2 Boeing 747 se heurtent au décollage faisant ainsi 583 victimes**.

Les données transmises par les avions équipés de l'« ADS-B » sont captées par un réseau de stations de réception au sol (beaucoup moins couteuses que les radars) reliées aux centres de contrôle. Dans le cas des vols transocéaniques, le signal est capté par un réseau de satellites de radiocommunications et retransmis au sol.

Le même signal radio peut être reçu par un autre avion dans le voisinage, et peut ensuite être affiché sur un écran qui montre la situation du trafic aérien.

Les données sont transmises toutes les 2 secondes sur la fréquence de 1,09 GHz en utilisant le « Mode S étendu ».

Le signal radio est modulé en PPM (Pulse Position Modulation ou modulation en position d'impulsions), c'est une technique de modulation utilisée pour transmettre sur une liaison « point-à-point » une chaîne de 112 bits à un débit de 1 Mbps. Le signal d'identification est encodé afin d'éviter qu'un émetteur pirate génère de faux signaux de nature à créer des problèmes pour le contrôle du trafic aérien.

Enfin nous terminons cet aperçu sur l'électronique embarquée à bord des avions sur 2 détails : tout d'abord l'éclairage à LED dans le Boeing 767 qui a remplacé les lampes classiques avec une économie d'énergie et des couleurs différentes, ensuite les boîtes noires qui sont maintenant entièrement numériques constituées de puces mémoires. ■

## ADIRU : une centrale inertielle

Dans les avions dotés du système « Fly-By-Wire d'Airbus », en mode « Normal Law », l'ordinateur gouverne l'avion grâce aux données transmises par l'« ADIRS » (Air Data Inertial Reference System) et sont également affichées sur les écrans LCD de l'« EFIS » (Electronic Flight Instrumentation System) dans la cabine.

Ces données concernent la vitesse, la pression atmosphérique, la température extérieure, mais aussi l'attitude de vol.

L'« ADIRS » est composé de 3 « ADIRU » qui constituent ensemble une centrale inertielle (conceptuellement similaire à l'« ARDUIMU » basée sur « Arduino ») et un « ADR » (Air Data Reference).

Il fournit l'altitude (pression statique), la vitesse (tube de « Pitot »), le « Mach », l'« AOA », la température. La centrale inertielle (Inertial Reference) fournit les données de position, d'assiette, de vecteur vitesse, d'accélération linéaire et d'accélération angulaire (gyroscope).

Les signaux analogiques des capteurs sont convertis par un « ADM » numérique afin d'être lus par l'« ADIRU ».




**L'« ADIRU » d'un Airbus A330.**

3DRAG - 8

# Faisons de la lumière pour la 3DRAG




**La Rédaction tient à remercier les nombreux lecteurs qui ont félicité les deux articles consacrés à la fabrication des circuits imprimés à partir de l'imprimante 3DRAG. Dans cet article nous vous proposons encore d'améliorer la 3DRAG en lui ajoutant un anneau d'éclairage à LED au niveau de la tête d'extrusion.**

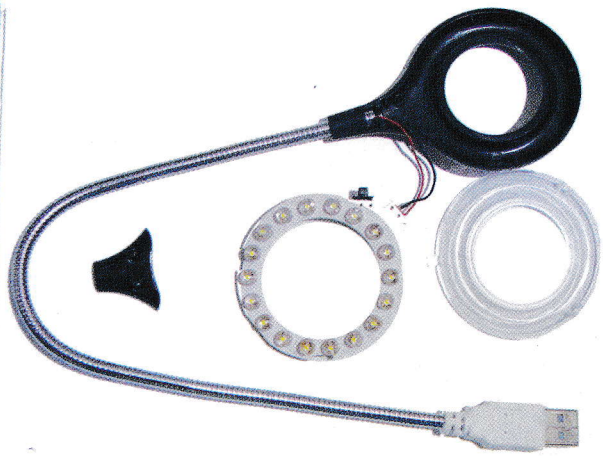
*de Simone Majocchi*

Chaque imprimante 3D qui se respecte doit refléter le style et les goûts de son propriétaire, et pouvoir passer d'une imprimante classique à « une imprimante 3D personnalisée ». Il s'agit d'un côté simplifié de son caractère « open ». En effet, bien que la 3DRAG ait été inspirée sur les aspects mécaniques et électroniques par la « RepRap », il existe toujours une possibilité d'ajouter un élément qui apporte une touche très personnelle au niveau de sa conception sur mesure.

Chacun d'entre vous, surtout si vous êtes le nouveau propriétaire d'une imprimante 3D – que ce soit la nôtre la 3DRAG ou un autre modèle - peut alors comprendre qu'avec un peu d'imagination et d'inventivité on peut améliorer les performances d'une « Maker » traditionnelle, de simplifier certaines tâches ou de rendre encore plus belle son imprimante.

Nous vous proposons dans cet article l'ajout d'un anneau formé de LED afin d'éclairer correctement toute la zone autour de l'extrudeuse (« tête d'impression »). De toute évidence, nous n'avons pas choisi pour cette réalisation la voie traditionnelle, à savoir une conception sur mesure de pièces, mais nous nous sommes plutôt efforcés d'utiliser ce qui existait déjà sur le marché.



**Cette lampe à LED en forme d'anneau et alimentée par une prise USB est facilement disponible dans le commerce.**

**Deux vis maintiennent la structure en plastique en forme de triangle, le reste est clipsé par des agrafes en plastique.**

## Vérifier l'impression

Le point de départ de notre mini-projet vient de la nécessité de voir clairement et sans ombres la zone d'extrusion, c'est-à-dire d'éclairer correctement la zone de la « tête d'impression » de notre 3DRAG afin de contrôler l'épaisseur de la première couche. La configuration de l'imprimante et la structure du bras sur lequel est monté le mécanisme d'extrusion permet l'ajout de LED dans la partie inférieure, mais cette méthode ne permet pas d'éclairer efficacement la zone concernée.

Le point critique est en effet au-dessous de l'extrémité de l'extrudeuse, sans aucune source de lumière à proximité immédiate. Les ombres créées par la structure elle-même empêchent un éclairage de bonne qualité. Dans la pratique, la solution la plus correcte serait d'installer une ou plusieurs LED autour de la partie inférieure de l'extrudeuse.

La bande de LED en forme d'anneau créerait un cylindre qui éclairerait la partie interne. Cependant sa structure ne permet pas de formes très différentes, de sorte que nous nous sommes concentrés sur seulement 2 possibilités :

- soit fabriquer un circuit imprimé en forme de cercle ;
- soit adapter quelque chose qui existe dans le commerce.

## Cherchons ce qui existe

Avant de nous plonger dans notre logiciel de conception de circuits imprimés, nous avons pensé qu'il était intéressant d'essayer de chercher quelque chose qui existe dans le commerce et qui pourrait être facilement adapté à l'imprimante. En fouillant sur la toile nous avons constaté qu'il existait différents gadgets et accessoires USB.

Parmi les réchauds, aspirateurs et petits ventilateurs nous avons trouvé une lampe à LED blanches en forme d'anneau avec un interrupteur M/A. Créée pour aider à taper sur le clavier la nuit ou peut-être à éclairer le visage face à une webcam, cette lampe ne coûte que quelques euros avec un diamètre extérieur d'environ 55 mm et un trou central de 32 mm, sa structure semble facilement adaptable au niveau de la tête d'impression de la 3DRAG.

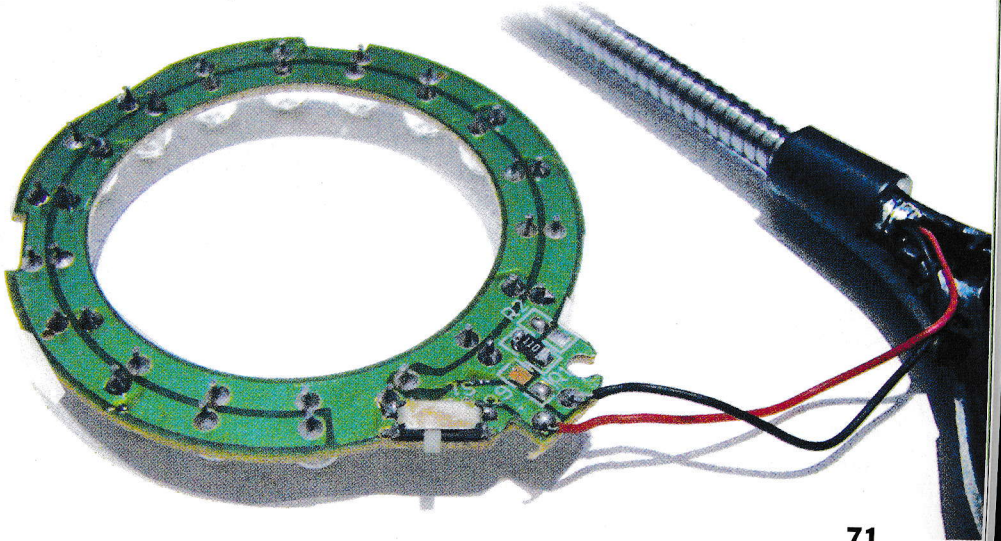
Une vérification rapide confirme que le bloc isolant qui maintient la pièce en laiton chauffée fait 42 mm de large, tandis que le diamètre interne de la lampe à LED en forme d'anneau sans la coque en plastique est de 38 mm, avec un chevauchement de 2 mm de chaque côté. Si nous l'avions conçu nous-même, nous aurions probablement pris des mesures très semblables.

## Un point chaud

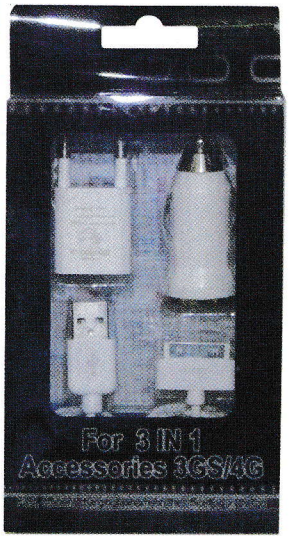
Maintenant que nous avons trouvé la lampe, nous devons trouver la meilleure façon de la positionner et de la fixer. Le plus pratique est d'installer la lampe juste sous le bloc isolant, mais elle est en contact avec la pointe de l'extrudeuse.

Cela va sûrement provoquer un échauffement important, les systèmes de fixation normaux ne peuvent pas supporter de telles températures. Ainsi pour des raisons d'amplitude thermique, nous devons écarter la possibilité d'« imprimer » avec la 3DRAG un quelconque support. Utiliser des vis et des boulons nécessiterait des trous et, dans le meilleur des cas, nous devrions démonter la moitié du bloc de l'extrudeuse.

**Sur cette image vous pouvez voir le circuit imprimé de l'anneau qui contient 18 LED blanches, un interrupteur et une résistance de limitation de courant. Toutes les LED sont connectées en parallèle.**



Donc nous avons opté pour le silicone rouge à haute résistance thermique, qui est approprié pour résister à des températures de 200 °C, et qui est flexible compte tenu des effets possibles de dilatation thermique des pièces. Il dispose en plus d'une bonne adhérence sur de l'isolant et sur de la bakélite comme celle de la base de l'anneau de LED, et enfin il est facile à appliquer. Ce silicone rouge est le même qui a été appliqué autour du bloc de chauffage de l'extrudeuse et donc son aptitude à résister à la chaleur de la 3DRAG a été déjà largement testée.






**Pour alimenter l'anneau de LED, nous utilisons un convertisseur pour véhicule, récupéré dans un kit pour téléphones mobiles, il fournit 5V sur la prise USB à partir de la prise « allume-cigare ». Il est peu couteux et fonctionne très bien.**

## Un montage avec du maquillage

Pour pouvoir fixer l'anneau de LED au-dessus de l'extrudeuse, mais sous l'isolant (c'est le rectangle composé d'un matériau de couleur beige avec l'autocollant « Attention chaleur »), nous avons prévu d'utiliser, pour les fils de la lampe la partie où passent le câble de l'alimentation de l'élément chauffant et celui de la sonde de température. Il y a 4 fils qui sortent de l'anneau de LED et nous choisirons 2 points pour la fixation avec le silicone rouge.

Conscient du processus d'assemblage, nous avons immédiatement pensé que nous pourrions dévisser les deux vis cruciformes qui maintiennent la pointe chauffante fixée au bloc de l'extrudeuse, libérant ainsi le haut à partir duquel il devient possible de fixer l'anneau de LED. Une fois l'anneau inséré avec une simple manœuvre, en le glissant sous la partie inférieure de l'isolant, il est prêt à être siliconé afin de tenir de manière stable. Aidez-vous des images pour le montage.

## L'alimentation

Sur le bloc de l'extrudeuse, arrive une série de fils dédiés au moteur et à l'extrémité chaude de l'extrudeuse. Cependant, dans notre projet d'imprimante 3D nous avons également un ventilateur sur le côté qui dispose d'un câble qui amène le + 12 V de la carte Sanguinololu – broches GND et + 12 V du connecteur EXT V2 - jusqu'à l'interrupteur de commande, avec une gaine thermorétractable qui arrive à hauteur du moteur de l'extrudeuse.

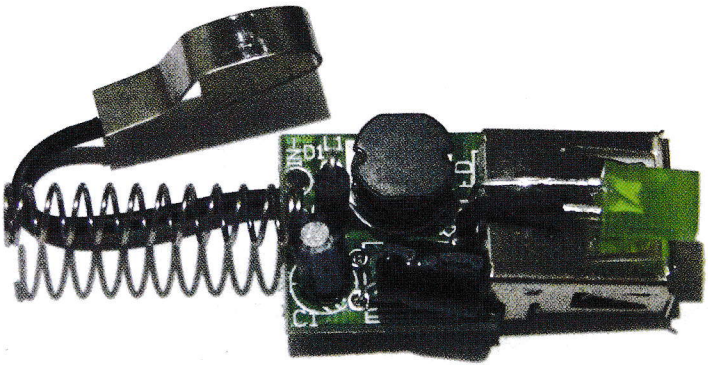
A regarder de plus près, ce n'est même pas du + 12 V, mais la tension qui est présente aux extrémités de la carte Sanguinololu, c'est-à-dire aux bornes de « Vin ». Le régulateur de la carte sert à transformer cette tension en + 5 V nécessaires à la carte « Atmega » ainsi qu'au reste du circuit à basse tension, tandis que la tension d'origine est directement délivrée aux drivers de puissance pour être transférée vers les moteurs et l'élément chauffant à travers les transistors MOSFET.

Notre anneau de LED qui se connecte via un port USB ne consomme que 150 mA sous 5 V et donc il ne peut pas être alimenté directement à partir du 12 V. Nous pourrions « tirer » 2 fils du + 12 V de la carte et ajouter un petit circuit avec un régulateur.

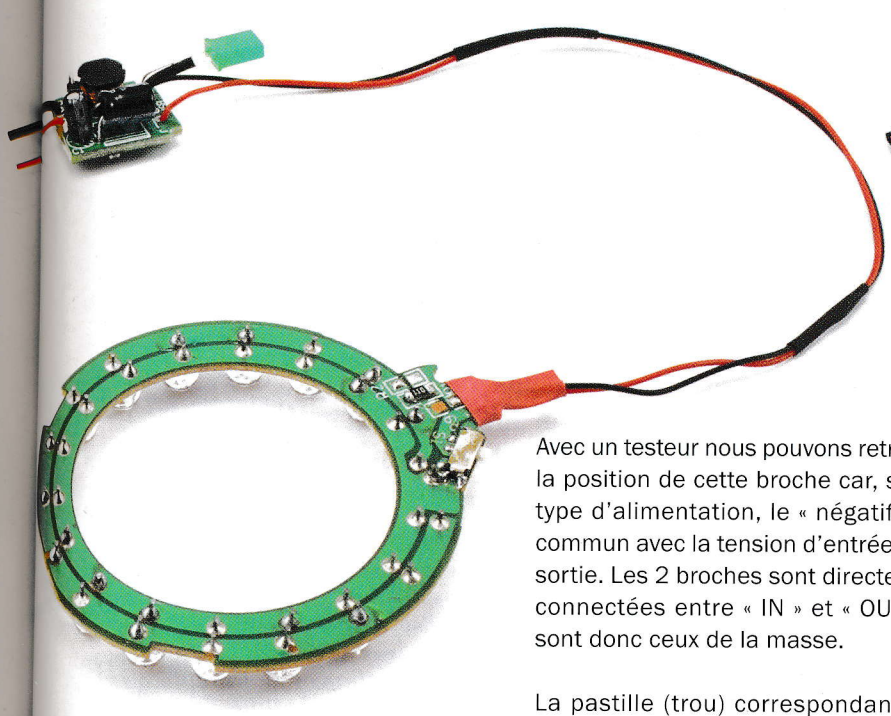
La première solution qui vient à l'esprit est d'utiliser un régulateur linéaire classique de type 7805 qui fournit la bonne tension en sortie, mais présente l'inconvénient de dissiper une chaleur importante car la tension à l'entrée du régulateur atteint 15 V sous 150 mA, ce qui correspond à une puissance dissipée de 1,5 W. Sans radiateur le régulateur travaillerait dans des conditions très défavorables.

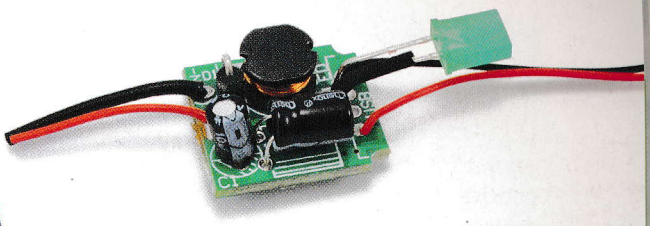
A cause du manque de place pour installer le radiateur, la solution du 7805 a été rejetée. L'alternative est d'utiliser un régulateur à découpage (commutation) caractérisé par un rendement beaucoup plus élevé, et une dissipation de chaleur moins importante, même avec une tension d'alimentation de 15 V pour obtenir 5 V en sortie.

L'idée de réaliser et fabriquer sur mesure un tel circuit ne semblait pas



**Le petit circuit contenu dans l'adaptateur pour allume-cigare est de type « commutation » et fournit 5 V sous 800 mA.**





être justifiée, car avec une simple modification d'un adaptateur que l'on trouve facilement dans le commerce nous avons atteint notre but. En fait la solution se trouve dans les accessoires pour voiture sous la forme d'un adaptateur pour GSM, un de ceux qui valent quelques euros et qui fonctionnent à partir d'une prise allume-cigare de 12 V ou 24 V et fournissent une tension régulée de 5 V sur un port USB.

En quelques minutes, nous avons démonté l'adaptateur et nous avons retiré le circuit imprimé sur lequel nous pouvions voir un circuit intégré MC34063A, une inductance, une paire de condensateurs électrolytiques et quelques composants CMS passifs. Le circuit intégré MC34063A est un convertisseur continu-continu capable de supporter jusqu'à 40 V sur son entrée avec un courant de sortie de 1,5 A.

Les concepteurs de l'adaptateur ont utilisé ce circuit en configuration « step-down » (abaisseur de tension) afin de fournir en sortie 5 V régulés et stabilisés avec un courant nominal compris entre 500 mA et 800 mA. Nous n'aurions pas pu faire mieux dans un si petit espace et avec un coût aussi faible.

Selon le tableau, le « négatif » (masse) se situe sur le côté gauche lorsque les contacts sont orientés vers le bas.

Avec un testeur nous pouvons retrouver la position de cette broche car, sur ce type d'alimentation, le « négatif » est commun avec la tension d'entrée et de sortie. Les 2 broches sont directement connectées entre « IN » et « OUT » et sont donc ceux de la masse.

La pastille (trou) correspondant à la broche 4 de la prise USB - retirée du circuit imprimé - doit être reliée au fil noir venant de l'anneau de LED, tandis que la broche 1 est reliée à la borne positive. La taille des fils déjà soudés peut sembler petite, mais dans l'ensemble les fils sont adaptés à l'usage.

La seule précaution à prendre est d'isoler avec de la gaine thermorétractable, le côté du circuit imprimé des Led sur lequel arrivent les fils. Il faut isoler, avec de la gaine, les 2 fils ainsi que la partie du circuit à proximité, afin d'éviter, lors des mouvements de l'extrudeuse, des courts-circuits entre les points 5 V et GND. Aidez-vous des photos que vous pouvez voir tout au long de cet article.

Maintenant, nous pouvons récupérer le 12 V provenant des fils rouge et noir qui sont fixés sur la barre de support de l'extrudeuse. De la carte « Sanguinololu » de la 3DRAG vers le bras de l'extrudeuse. 2 fils multi-brins (un blanc et un orange) et 2 conducteurs monofilaires de couleur marron partent de la carte « Sanguinolu » de la 3DRAG vers le bras de l'extrudeuse.

Le câble blanc est constitué de 4 fils isolés, comme le orange du moteur pas à pas. Le câble blanc est celui qui nous intéresse car il dispose à l'intérieur d'un fil rouge et d'un noir provenant de la carte « Sanguinololu » des broches 12 V et GND (comme mentionné ci-dessus, le 12 V n'est pas stabilisé), et des fils bleu et jaune qui se connectent au capteur de température.

Les fils rouge et le noir doivent dépasser de la gaine du câble blanc d'une quinzaine de centimètres et doivent être reliés aux fils rouge et noir qui alimentent le ventilateur via l'interrupteur.

Nous devons retirer la gaine thermorétractable qui protège le câble, ensuite dessouder les fils et insérer de nouveau de la gaine thermorétractable puis, les souder ensemble avec les fils qui vont au ventilateur. Les 2 autres extrémités des fils rouge et noir se connectent à la petite alimentation à découpage (c'est-à-dire l'adaptateur).

Lorsque vous avez fait cela, avant de poursuivre vous pouvez faire un test rapide. Alimentez la 3DRAG et vérifiez que la LED verte de la petite alimentation (adaptateur) s'allume, ensuite vérifiez que la lampe à LED placée près de l'extrudeuse s'allume.

Si la lampe ne s'allume pas, manœuvrez le petit interrupteur situé près du connecteur de l'alimentation, il peut être sur la position OFF. Si par contre l'anneau de LED ne s'allume pas mais que le voyant vert de l'adaptateur est allumé, c'est que vous avez un problème de polarité ou de continuité (fil coupé ou mal soudé). Si la LED verte de l'adaptateur n'est pas allumée, c'est que vous avez inversé les fils rouge et noir du câble blanc.

Avec les connexions effectuées et testées, nous pouvons passer à la fixation de nos 2 éléments, à savoir l'anneau de LED sur le bloc isolant et le petit circuit de l'adaptateur à la barre de support en aluminium de l'extrudeuse.

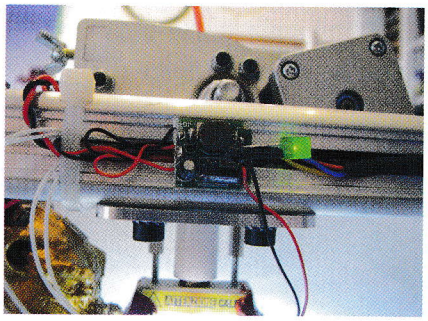
| Broche | Nom | Couleur du fil | Description |
|---|---|---|---|
| 1 | VBUS | Rouge | +5V |
| 2 | D - | Blanc | Données - |
| 3 | D + | Vert | Données + |
| 4 | GND | Noir | Masse |

## Fixations

Nous avions laissé de côté l'anneau de LED placé au-dessus de l'extrudeuse, après avoir dévissé les 2 vis à tête cruciforme qui fixent le bloc de la pointe chaude au reste de l'extrudeuse. Maintenant que le câblage est testé, nous pouvons utiliser le silicone haute température et fixer le circuit imprimé sur la partie inférieure du bloc isolant. Sur les différentes photos vous pouvez voir comment positionner l'ensemble par rapport aux limites du cadre.

Avant d'utiliser le silicone, nettoyez avec un dégraissant le bloc isolant de couleur beige et assurez-vous que le circuit imprimé des LED soit propre. Essayez de le positionner et de le mettre en place en exerçant de petites pressions sur le circuit imprimé circulaire (attention de ne pas forcer !). Les soudures des LED peuvent toucher l'isolant, cela ne pose pas de problème.

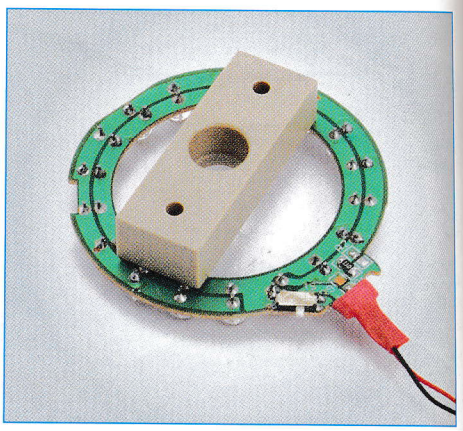
Notez également la façon dont les fils et l'interrupteur sont positionnés ; les fils ne doivent pas gêner, tandis que l'interrupteur peut être utile lorsque vous ne voulez pas éclairer la surface d'impression.



**Le régulateur à découpage est positionné, avec un adhésif double-face, derrière le bloc d'extrusion et est connecté en parallèle avec les fils qui alimentent le ventilateur.**

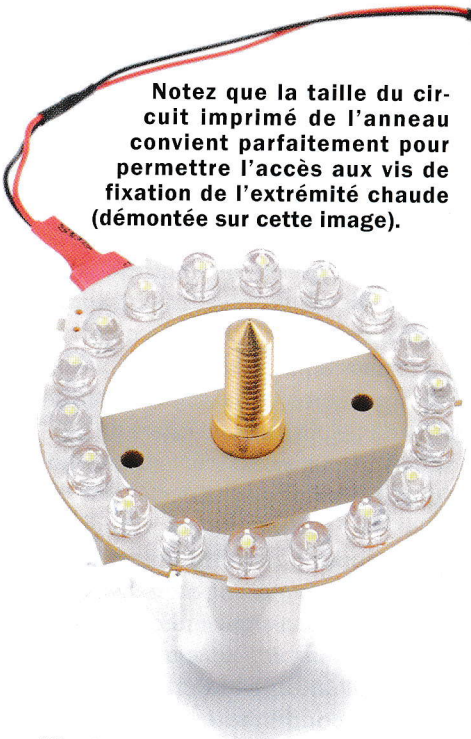
Nous avons décidé de positionner le fil et l'interrupteur à l'arrière, légèrement tournés par rapport aux fils de l'extrudeuse.

Lorsque le circuit imprimé est stable, horizontal et le plus près possible du bloc isolant, vous pouvez commencer à enduire de silicone les 2 angles formés par le circuit et le bloc isolant. Remplissez les angles avec le silicone, puis après vous avoir mouillé les doigts avec de l'eau savonneuse, façonnez le silicone en le faisant adhérer à divers endroits tout en étant le plus propre possible.



**Voici comment positionner le circuit imprimé en forme d'anneau par rapport à l'isolant constitué d'un matériau de couleur beige.**

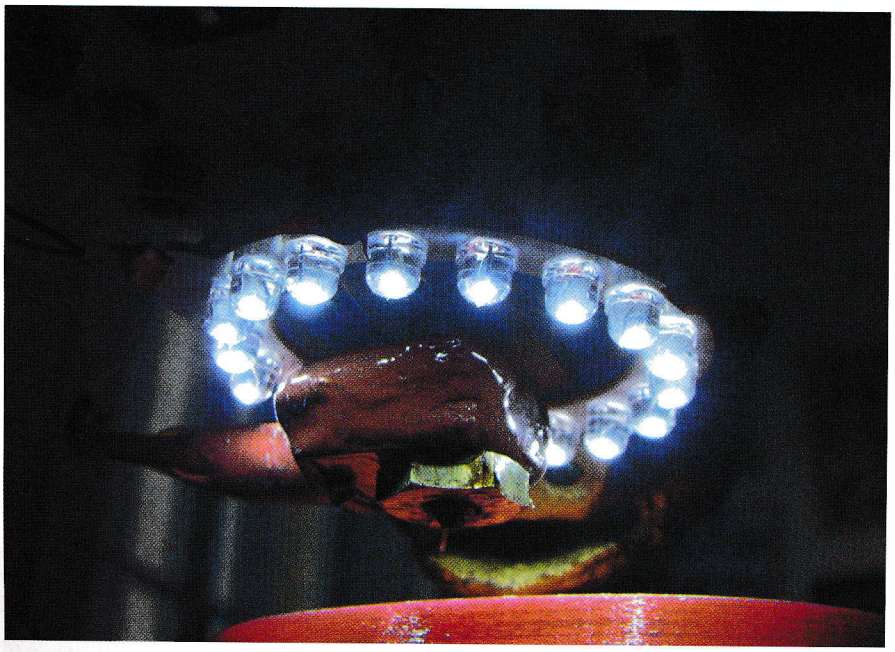
**Notez que la taille du circuit imprimé de l'anneau convient parfaitement pour permettre l'accès aux vis de fixation de l'extrémité chaude (démontée sur cette image).**



N'oubliez pas qu'il faut une vingtaine de minutes avant que le silicone durcisse, vous pouvez avoir besoin de le repositionner s'il a tendance à couler ou à tomber.

Toute astuce pour garder la bague en place pendant que le silicone durcit est la bienvenue. Dans notre cas, les soudures des LED ont créé un espace et nous avons dû en utiliser plus, cela a permis de créer un joint qui maintenait la bague.

Pour l'alimentation à découpage, étant donné sa petite taille, après avoir vérifié qu'il n'y a pas de parties saillantes, vous pouvez utiliser un ruban adhésif double face pour la fixer.



**Sur cette photo, vous pouvez apprécier la puissance de l'éclairage et la beauté de l'anneau de LED blanches. Leur positionnement est de nature à fournir un éclairement relativement homogène sur une surface presque parfaitement centrée autour de la pointe de l'extrudeuse. Cela vous permet de vérifier facilement l'état d'avancement de l'impression de la première couche.**

Recouvrez l'ensemble de la carte d'alimentation avec le ruban adhésif double face du côté des composants CMS en faisant attention de ne pas endommager le circuit, puis fixez-la à la barre en aluminium.

N'appuyez pas trop fort et assurez-vous qu'il n'y a pas de courts-circuits entre la barre en aluminium et les fils ; si nécessaire doublez la couche de ruban adhésif double-face pour assurer une bonne isolation électrique.

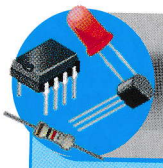
## Le matériel

Pour cette modification de notre imprimante 3DRAG, nous avons utilisé des pièces que vous pouvez vous procurer facilement dans les rayons « auto » des supermarchés voire dans les magasins d'électronique.

La lampe à LED USB en forme d'anneau coûte quelques euros, tandis que l'adaptateur allume-cigare 12 V vers 5 V USB peut se trouver soit individuellement, soit dans un kit d'alimentation pour téléphones mobiles. Dans ce cas également, le coût ne dépasse pas quelques euros.



Une touche théâtrale à la structure de la 3DRAG.



### Comment se procurer la 3DRAG

L'imprimante **3DRAG** 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.

Pour plus de renseignements consultez l'adresse :
**http://comelec.fr/6-imprimante-3d**

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

## Le site de la 3DRAG :

### 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.



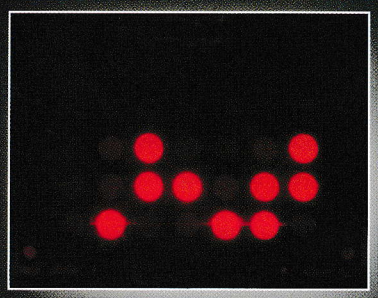
**Le site : www.3dprint.electroniquemagazine.com**

Le site comprend également une section consacrée aux **firmwares** de l'imprimante et aux **logiciels pour l'impression 3D**, au fur et à mesure nous améliorerons les programmes en proposant des mises à jour. Vous trouverez tous les programmes consacrés à l'imprimante **3DRAG** à l'onglet « **Logiciels** ».

ET790

# HOROLOGE BINAIRE

**Au lieu d'utiliser de classiques horloges à cadrans traditionnels avec des aiguilles ou avec un affichage numérique, cette horloge binaire affiche l'heure en utilisant des LED, dont chacune est associée à une valeur binaire. Les « Heures », « Minutes » et « Secondes » sont chacune visualisées par deux rangées de six LED.**





*de Arsenio Spadoni*

Il y a une quarantaine d'années, l'heure était donnée par des horloges qui disposaient d'un système mécanique dont le composant principal était un ressort enroulé en spirale et constituant avec le balancier la mécanique du mouvement des aiguilles. De même, les montres, à cette époque, reposaient sur le même principe mécanique si bien qu'au moins une fois par jour il fallait les « remonter » à la main (remontage manuel), d'autres disposaient d'un système de « remontage automatique » qui s'effectuait grâce à un mécanisme qui permettait de réarmer le ressort à l'aide des mouvements du poignet.

Ensuite la mode des montres à quartz se développa, en particulier celles à affichage numérique. Elles étaient très appréciées parce qu'elles étaient plus lisibles que les montres traditionnelles (surtout dans l'obscurité).

Elles ont connu une croissance exponentielle dans ces années. Une situation qui n'a jamais été appréciée par les connaisseurs (ou collectionneurs) de montres classiques et qui a disparu en une décennie comme tous les phénomènes de mode. Il faut dire que depuis quelque temps, pour les nostalgiques des diodes électroluminescentes ou cristaux liquides, des modèles spéciaux sont apparus, certainement pas pour être utilisés, mais plutôt pour perpétuer un mystère et présenter une nouveauté. Un peu comme des œuvres d'art modernes qui veulent se démarquer pour être appréciées.

Nous parlons là des horloges ou des **montres binaires**, dont certains modèles sont désormais disponibles au poignet, et qui **indiquent l'heure sous forme de « points »** (par LED ou segment LCD) dans le format utilisé par les ordinateurs et les microprocesseurs, c'est-à-dire sous la forme binaire.

Chaque « **point** » représente un **bit**. Dans la pratique, la montre est constituée par plusieurs rangées de LED et chaque LED correspond à un « point » (à un bit) dont chacun exprime une valeur qui est convertie en décimal.

Par exemple, pour afficher l'heure on utilise 4 LED et on applique la logique correspondante soit 1 ou 0 respectivement pour allumer ou éteindre chaque LED comme dans le cas d'une logique binaire. La première LED correspond à 1 ($2^0$), la seconde correspond à 2 ($2^1$), la troisième à 4 ($2^2$) et la quatrième à 8 ($2^3$). Donc, si nous voyons les LED allumées en « 1 » et « 4 » cela signifie que nous avons « 5 ».

Pour les minutes et les secondes, qui sont en base « 60 », il faut plus de LED pour afficher correctement l'heure. La première LED correspond à 1 ($2^0$), la seconde correspond à 2 ($2^1$), la troisième à 4 ($2^2$), la quatrième à 8 ($2^3$), la cinquième à 16 ($2^4$), la sixième à 32 ($2^5$), et enfin comme il y a 2 rangées (pour les dizaines et les unités) cela fait un maximum de 64.

Ainsi, en gardant la même logique (LED allumée = 1, LED éteinte = 0), si nous voyons les LED allumées en « 1 » et « 6 », cela signifie que les minutes ou les secondes valent 33 (1 + 32). Si, au contraire, nous voyons les LED allumées en « 5 » et « 6 », cela signifie que les minutes ou les secondes valent 48 (16 + 32).

L'horloge (ou nous pouvons dire aussi la montre) décrite dans cet article affiche les heures, minutes et secondes. Elle est dotée d'un module RTC (« Real Time Clock » ou « horloge temps réel ») dédié à être interfacé avec un microcontrôleur.

En utilisant le format binaire, chaque nombre est exprimé par un certain nombre de LED (allumées ou éteintes) sur lesquelles on applique une puissance de deux : 1, 2, 4, 8, etc. L'affichage est au format 24 heures et les heures, minutes et secondes sont visualisées chacune par 2 rangées de LED (une pour les dizaines et l'autre pour les unités). Il s'ensuit que la rangée des unités est composée de 4 éléments, parce que la plus grande valeur à afficher est « 9 ».

Pour avoir une idée de la façon dont l'affichage fonctionne, imaginez qu'il soit **20 H 05 M 45 S**. Seule la deuxième LED de la rangée la plus à gauche (c'est-à-dire les dizaines de l'heure) s'allume (2), aucune LED de la rangée des unités (à droite) n'est allumée (0).

Sur la 4$^{eme}$ rangée à partir de la gauche qui correspond aux unités des « minutes », nous avons la 1$^{ere}$ LED et la 3$^{eme}$ LED qui sont allumées pour indiquer « 5 ». Enfin, la 3$^{eme}$ LED de la 5$^{eme}$ rangée à partir de la gauche s'allume pour indiquer « 4 » et, sur la 6$^{eme}$ rangée la 1$^{ere}$ LED et la 3$^{eme}$ LED sont allumées pour indiquer « 5 » (45 secondes).

## Schéma électrique

Notre horloge (montre), dont le schéma électrique est visible en figure 1, est basée sur un **module RTC** (« **Real Time Clock** » ou « **Horloge Temps Réel** ») que nous avons déjà utilisé plusieurs fois dans d'autres montages.

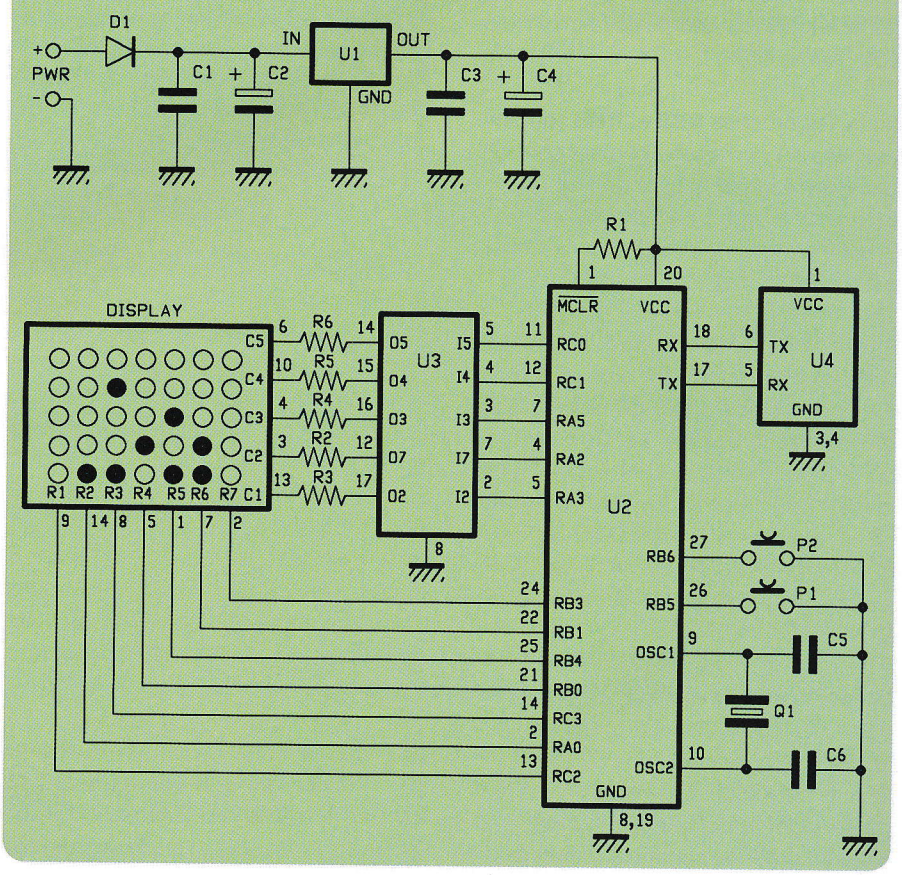
Ce circuit est basé sur une puce **DS1307** de la firme **Maxim** qui est interfacée à l'aide du bus I2C à un microcontrôleur **PIC16F876A** de chez **Microchip**. Celui-ci dispose de lignes de données (TX / RX) séries bidirectionnelles à un niveau TTL, de lignes pour la programmation en circuit (ICSP), d'un BUS I2C et d'une ligne d'alimentation Vpp.

L'horloge, c'est-à-dire le **module RTC**, est **interrogée de manière cyclique par le microcontrôleur U2** afin de lire les paramètres des « heures », « minutes » et « secondes » et les convertir pour les afficher sur une matrice à LED. Le **PIC interroge le module RTC une fois par seconde via l'interface série bidirectionnelle** (lignes TX et RX). En fait l'interface fonctionne en mode semi-duplex et est paramétrée de la manière suivante :

***vitesse 9600 bps, 8 bits de données, pas de parité et 1 bit d'arrêt***

Le **microcontrôleur lit et stocke les données**, ensuite il transforme les

**Figure 1 : schéma électrique de notre horloge binaire.**



« heures », « minutes » et secondes en **valeurs binaires**, représentées par un certain nombre de lignes d'entrées/sorties. Pour remédier au manque de lignes d'entrées/sorties, nous avons mis en œuvre un **multiplexage des lignes** sous forme « **matricielle** », afin d'afficher rapidement les séquences des données binaires.

Les **sorties du PIC ne peuvent pas alimenter directement toutes les LED**, car elles ne disposent pas d'un courant suffisant (les ports A, B et C sont limités à 25 mA chacun). Le nombre de LED pour visualiser les « heures », « minutes » et « secondes » est assez élevé. En effet il faut 2 LED pour les dizaines et 4 LED pour les unités des « heures », 3 LED pour les dizaines et 4 LED pour les unités des « minutes », de même pour les « secondes ». Par conséquent il faut un total de 6 + 7 + 7 = **20 LED** et autant de lignes.

Lors de l'initialisation des entrées/sorties, le programme du microcontrôleur U2 définit les broches **RA0**, **RB0**, **RB1**, **RB3**, **RB4**, **RC2**, **RC3** en « **sorties** », elles pilotent les **lignes** (anodes) de la matrice à LED. De même les broches **RA2**, **RA3**, **RA5**, **RC0**, **RC1** sont aussi définies en « **sorties** », mais elles pilotent les **colonnes** (cathodes) de la matrice à LED (voir la figure 2).

Enfin, les broches **RB5** et **RB6** sont initialisées en « **entrées** » et disposent de **résistances internes de pull-up** (résistances de tirage). Elles permettent de **lire directement l'état des boutons poussoirs** de réglage **P1** et **P2**.

Sur le schéma électrique (figure 1), vous pouvez remarquer que les **lignes de la matrice sont commandées directement par les sorties** correspondantes du **PIC** et sont actives à un niveau haut. Par contre les **colonnes** de la matrice sont pilotées à l'aide d'un **driver de ligne** (7 pour être exact) contenu dans le circuit intégré **ULN2803** (U3).

Chaque sortie du circuit U3 est formée par 2 transistors NPN montés en Darlington avec une diode de protection sur le collecteur (qui dans ce cas n'est pas nécessaire, puisque nous n'avons pas à commander des charges inductives).



# Au bout des doigts

**Êtes-vous fatigué par les montres ? Vous ne supportez pas d'avoir quelque chose à votre poignée ? Elle vous serre ou elle est trop large. Elle est mouillée de sueur et vous ne la supportez plus pour le reste de la journée ? Vous êtes allergique au métal ? Et bien, maintenant vous pouvez porter une montre à un endroit plus confortable. Non, nous ne vous proposons pas de la porter à la ceinture, mais une nouvelle technologie proposée par la firme Timex la « TX54 ». C'est une montre que vous pouvez positionner à la pointe de votre doigt, elle se présente essentiellement sous la forme d'un faux ongle dans lequel il y a une puce qui pilote un afficheur LCD rétroéclairé.**

**En cette époque où les femmes sont en compétition pour avoir des ongles artificiels de plus en plus pittoresques, une horloge en forme d'ongle est certainement un moyen de se démarquer au niveau de l'originalité. La montre adhère parfaitement à l'ongle du pouce (pour lequel elle a été conçu), à tel point qu'elle est pratiquement invisible, et est parfaitement camouflée par le reste de la main. Elle peut s'illuminer dans de nombreuses couleurs. La TX54 dispose également d'un calendrier et est commandée par un seul bouton. Quelle surprise pour les personnes qui vous demande l'heure, car au lieu de la voir à votre poignet ils la verront au bout de vos doigts ...**

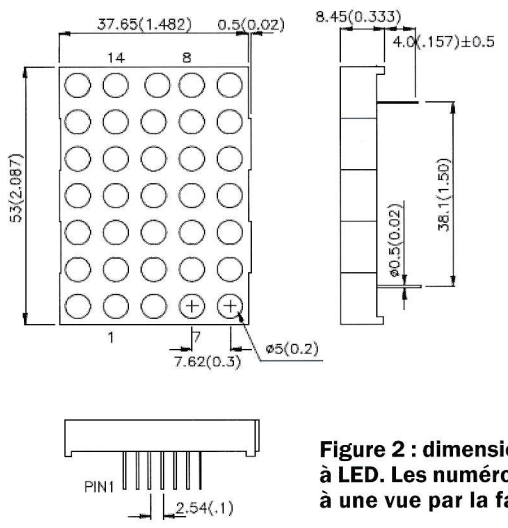
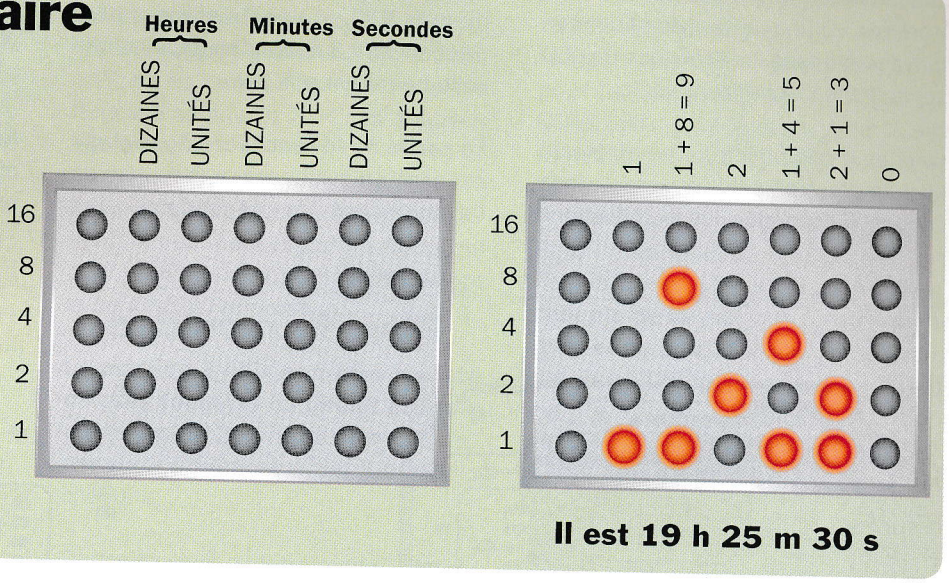



**Figure 2 : dimensions et brochage de la matrice à LED. Les numéros des broches correspondent à une vue par la face arrière de la matrice.**

# L'heure en binaire

Voici comment notre horloge indique l'heure, chaque ligne représente au format binaire (LED allumée = 1 ; LED éteinte = 0) une valeur numérique décimale. Pour la représentation, nous utilisons deux colonnes de LED pour les « heures », 2 colonnes pour les « minutes » et encore 2 colonnes pour les « secondes ».




Cette structure de circuit est due au fait que le pilotage direct de la matrice à LED par les sorties du microcontrôleur provoque une chute de tension sur la ligne d'alimentation de l'anode (niveau haut), et que dans le circuit de multiplexage la cathode est à un niveau logique zéro, la différence de potentiel disponible serait trop faible pour éclairer correctement les diodes.

En plaçant en aval de la cathode un driver avec une résistance en conduction beaucoup plus faible que celle des sorties du PIC, le problème est résolu.

En effet, malgré la chute de tension des lignes RA0, RB0, RB1, RB3, RB4, RC2, RC3 qui n'est pas négligeable, celle du driver de lignes, avec les courants mis en jeu, est négligeable et la différence de potentiel sur les LED permet d'assurer une bonne luminosité.

L'horloge fonctionne avec une tension d'alimentation continue comprise entre 9 VDC et 15 VDC appliquée aux point « PWR + » et « PWR - ». Un classique régulateur 7805 (boitier TO-220) fournit + 5 V au microcontrôleur et au module RTC.

## Les réglages

Toutes les montres, y compris la nôtre, nécessitent le réglage de l'heure. Dans notre cas nous utilisons les poussoirs P1 et P2. Le premier règle les « minutes » et le second les « heures », plus exactement chaque fois que vous appuyez sur l'un des poussoirs, la valeur correspondant aux « minutes » ou aux « heures » (selon le bouton pressé) augmente d'une unité.

Chaque pression sur **P1 fait avancer** les « **minutes** » d'une unité, tandis que

# Technique et tendance




Incroyable mais vrai, il y a plusieurs constructeurs de montres binaires qui présentent des modèles accrocheurs au design futuriste avec des bracelets. Bien sûr, nous ne pouvons pas dire que la lisibilité est immédiate par rapport aux montres classiques, mais elles apportent certainement un moyen de se démarquer, de porter quelque chose qui est à la fois utile et original. Essayez de porter à votre poignet ce type de montre et beaucoup de personnes vous demanderont certainement comment elle fonctionne. Un conseil cependant, si quelqu'un vous demande l'heure dites-lui ... que vous n'avez pas de montre, car si vous la laissez entrevoir, il vous sera difficile de donner l'heure exacte...

Dans le commerce il existe tout type de montre, celles qui indiquent simplement l'heure, les minutes et les secondes. D'autres affichent le jour de la semaine, la date, etc. Cependant les différents fabricants ont recherché de nouvelles formes de représentations plus originales en affichant l'heure en valeurs binaires, soit sous la forme de rangées de LED, soit sous la forme de cristaux liquides de diverses couleurs.

Si vous êtes intéressé par ce type de montre bracelet, visitez les sites suivant :
***http://www.heure-binaire.com***
***http://www.tokyoflash.com/fr/watches/1/***

chaque pression sur **P2 fait avancer** les « **heures** » d'une unité. Les **secondes n'avancent pas** (les 2 rangées de LED sur la droite ne changent pas).

Lorsque le microcontrôleur ne détecte plus de pression sur l'un des boutons pendant **5 secondes**, les modifications apportées sont mémorisées dans le module RTC, et l'écran s'éteint pendant environ **3 secondes**, puis la nouvelle heure est affichée.

La **sortie de la procédure de réglage** est mise en évidence par la reprise de l'**avancement** des « **secondes** ».

## Réalisation pratique

Passons maintenant à la fabrication de l'horloge.

Nous devons d'abord fabriquer un petit circuit imprimé double face sur lequel prendront place tous les composants.

## Plan de montage

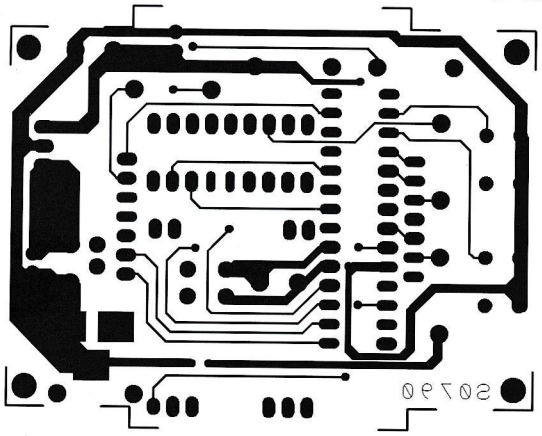


**Figure 3 : dessin du circuit imprimé à l'échelle 1 : 1 côté soudures de l'horloge binaire.**

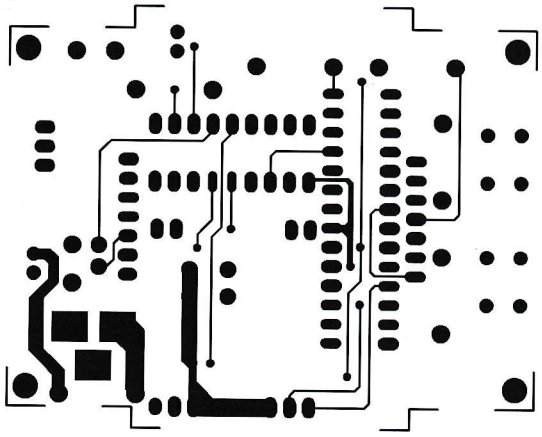


**Figure 4 : dessin du circuit imprimé à l'échelle 1 : 1 côté composants de l'horloge binaire.**

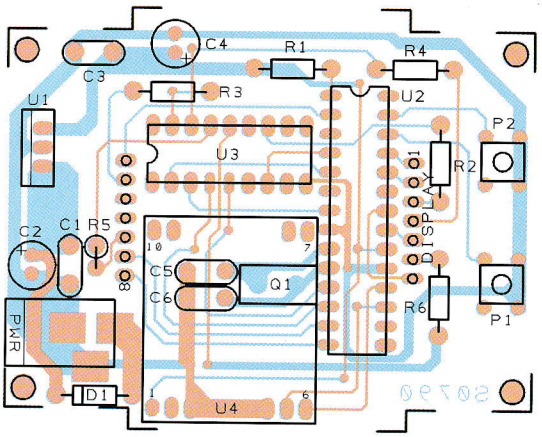



**Figure 5 : schéma d'implantation des composants de l'horloge binaire. L'afficheur est disposé du côté soudures.**

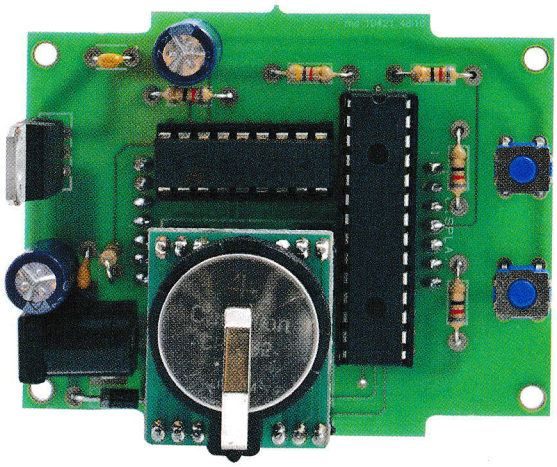


**Figure 6 : photo de l'un de nos prototypes de l'horloge binaire.**

### Liste des composants de l'horloge binaire

R1..... 4,7 k
R2..... 33
R3..... 33
R4..... 33
R5..... 33
R6..... 33
C1..... 100 nF multicouche
C2..... 470 µF / 16 V électrolytique
C3..... 100 nF multicouche
C4..... 470 µF / 16 V électrolytique
C5..... 15 pF céramique
C6..... 15 pF céramique

D1.....1N4007
Q1.....Quartz 20 MHz bas profile
U1.....7805
U2.....PIC16F876A (MF790)
U3.....ULN2803
U4.....Module RTC (FT674M)
P1.....Microswitch
P2.....Microswitch

Afficheur à matrice 7x5 (TC20-11SRWA)

Divers
Prise d'alimentation pour ci type FA
Support circuit intégré 2 x 9 broches
Support circuit intégré 2 x 14 broches
Barrette femelle 2 pôles x 2 (RTC)
Barrette femelle 3 pôles x 2 (RTC)
Barrette femelle 7 pôles x 2 (Matrice)

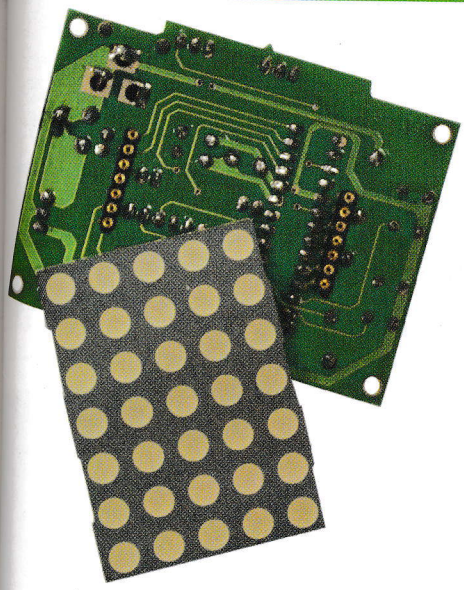


**Figure 7 : l'afficheur est une matrice à LED rondes de 7 x 5. La matrice est enfichée sur 2 barrettes femelles de 7 contacts chacune et soudées du côté soudures du circuit imprimé. Le chiffre 1 indique la broche 1 de l'afficheur.**

Les **typons des circuits à l'échelle 1** et le **programme du PIC** au format « .hex » **sont téléchargeables** sur notre site **www.electroniquemagazine.com** dans le sommaire détaillé du **numéro 129 à l'onglet « Télécharger »**. Une fois le circuit réalisé, commencez par souder les composants à bas profil, les résistances et les condensateurs non polarisés. Le module RTC doit être enfiché en dernier dans 2 barrettes femelles au pas de 2,54 mm soudées sur le circuit.

Ensuite soudez tous les autres composants à commencer par les supports pour le PIC et l'ULN2803 en respectant la polarité et l'orientation.

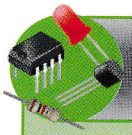
La **matrice à LED doit être montée du côté soudures du circuit imprimé** au moyen de 2 barrettes femelles au pas de 2,54 mm et ayant 7 contacts chacune.

Rappelez-vous que la broche 1 est du côté gauche (vers le bas) de l'afficheur vu par l'arrière et doit être enfichée en bas du circuit du côté droit (chiffre 1 en rouge) comme indiqué sur la figure 7.

Pour terminer le montage, nous vous recommandons de fabriquer un masque sur lequel sont indiquées les heures, minutes et secondes correspondant à chaque « valeur » des LED (1, 2, 4, 8, 16). Ceci afin de vous familiariser avec la visualisation binaire.

Pour recouvrir par la suite la matrice à LED nous vous conseillons d'utiliser le cache référence « **BMK153** » prévu pour l'afficheur **TC20-11SRWA** (chez COMELEC), qui est rouge et qui a des dimensions correctes pour accueillir l'ensemble du circuit.

Pour l'alimentation vous pouvez utiliser un bloc secteur non stabilisé capable de fournir une tension comprise entre 9 et 15 V avec un courant d'environ 100 mA.

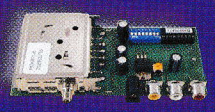


### Comment construire ce montage

Le pic programmé ainsi que le cache **BMK153** et la matrice à LED **TC20-11SRWA** sont disponibles auprès de **COMELEC**. Les typons des circuits imprimés et le programme « **MF790.hex** » pour le PIC sont téléchargeables gratuitement sur notre site : **www.electroniquemagazine.com** dans le sommaire détaillé de la revue numéro **129** section « **Télécharger** ».


## ÉMETTEURS & RÉCEPTEURS 1,2 & 2,4 GHz

**EMETTEURS 1.2 & 2,4 GHz**
**20, 200 et 1000 mW**



Alimentation : 13,6 VDC. 4 fréquences en **2.4 GHz :2,4 - 2,427 - 2,454 - 2,481 GHz** ou 8 fréquences en **1.2 GHz 20 mW: 1,112 - 1,139 - 1,193 - 1,220 - 1,247 - 1,264 - 1,300 GHz** ou 4 fréquences en **1.2 GHz 1 W: 1,120 - 1,150 - 1,180 - 1,255 GHz.** Sélection des fréquences : dip-switch. Stéréo : audio 1 et 2 (6,5 et 6,0 MHz). Livré sans alim ni antenne.

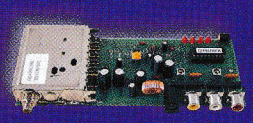
TX2.4G..... Emetteur 2,4 GHz 4 c monté 20 mW ..... 39,20 €
TX2.4G-200 mW Emetteur monté 4 canaux 200 mW..... 119,20 €
TX1.2G..... Emetteur 1,2 GHz 20 mW monté 8 canaux ..... 48,20 €
TX1.2G-1-... Emetteur 1,2 GHz monté 1 W 4 canaux ..... 79,30 €

**VERSION 256 CANAUX**

Extensions pour émetteurs **TX2.4G / TX1.2G** et récepteurs **RX2.4G / RX1.2G** : permettent d'atteindre **256** canaux.
Le pas est de 1 MHz. Sélection des canaux par dip-switch.
Fréquences de départ : **2,3** pour les versions **TX2,4G / REX2.4G** et **1,2** pour les **TX 1,2G / RX 1,2G**. Extension vendue sans émetteur ni récepteur.

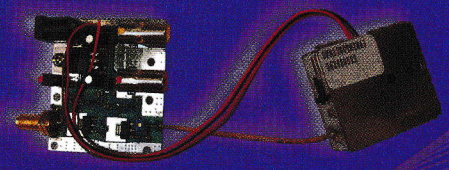
TEX1.2..... Kit extension 1,2 à 1,456 GHz ..... 19,90 €
TEX2.3..... Kit extension 2,3 à 2,556 GHz..... 19,90 €
REX1.2..... Kit extension 1,2 à 1,456 GHz ..... 19,90 €
REX2.3..... Kit extension 2,3 à 2,556 GHz ..... 19,90 €

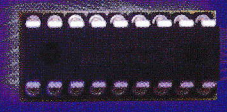
**RÉCEPTEURS 4 CANAUX**
**1,2 & 2,4 GHz**

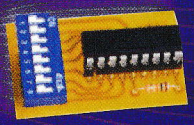


Alimentation : 13,6VDC. 4 fréquences en **2.4 GHz :2,4 - 2,427 - 2,454 - 2,481 GHz** ou 8 fréquences en **1.2 GHz : *1,112 - 1,139 - 1,193 - 1,220 - 1,247 - 1,264 - 1,300 GHz***. Sélection des fréquences : dip-switch pour le 1,2 GHz et par poussoir pour les versions 2,4 GHz. Stéréo : audio 1 et 2 (6,5 et 6,0 MHz). Fonction scanner pour la version 1.2 GHz. Livré sans alimentation ni antenne.

RX2.4G.....Récepteur monté 2.4 GHz 4 canaux..... 39,00 €
RX1.2G.....Récepteur monté 1.2 GHz 8 canaux ..... 48,20 €







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

Photos non contractuelles. Publicité valable pour les mois de parution. Prix exprimés en euros TTC. Sauf erreurs typographiques ou omissions



3DRAG - 9

# 3DRAG
# Contrôle à distance avec le RaspberryPi

**Dans cet article nous vous proposons d'imprimer à distance et de contrôler la 3DRAG à l'aide d'une solution LINUX embarquée qui n'est autre que le RaspberryPi, sans accaparer un ordinateur.**

*de Francesco De Fabiani*

Il nous est arrivé d'être lassés de devoir laisser notre ordinateur personnel pendant des heures et des heures dans le garage, sans pouvoir s'en servir pour d'autres tâches, sans compter le gaspillage d'énergie, afin d'imprimer avec notre imprimante **3DRAG**. La solution qui nous est venue à l'esprit était d'utiliser comme remplaçant d'un notebook ou d'un PC une carte **RaspberryPi**.

Au début, nous avons supposé que l'installation du logiciel « Repetier-Host » pour LINUX directement sur **RaspberryPi** était compliquée, mais nous avons trouvé une **version serveur spécifique** de « **Repetier-Host** » pour la petite carte marquée de la « framboise ». Son nom est « **Repetier-Server** » et grâce à la création d'une page web avec l'adresse IP du **RaspberryPi**, nous pouvons soit commander à distance la **3DRAG**, soit simplement imprimer des modèles en chargeant le G-Code sur le serveur, sans avoir besoin d'utiliser un PC.

Pour installer « **Repetier-Server** », vous pouvez vous rendre à l'adresse suivante :

*http://www.repetier.com/repetier-serveur-download/*

et bien entendu **nous mettrons le fichier directement en téléchargement sur nos sites dédiés** à la **3DRAG** et au **RaspberryPi**.

Téléchargez le fichier « **Repetier-Server 0.24 for RaspberryPi** » et suivez les instructions que nous vous présentons dans l'encadré dédié ci-après.

Après l'installation, la page web sera disponible à l'**adresse IP** du **RaspberryPi** au **port 8080**.

Après avoir réalisé la commande à distance de l'imprimante **3DRAG**, nous avons voulu voir le travail de l'impression à distance afin de vérifier que tout fonctionne correctement, nous avons donc connecté une webcam sur le 2eme port USB du **RaspberryPi**.

## Jetons un coup d'œil avec la webcam

Après la réalisation de l'ensemble du système, il est clair qu'il faut vérifier que tout fonctionne correctement. Encore une fois le **RaspberryPi** nous rend service, avec quelques modifications il se transforme en un serveur de **webcam qui nous permet de visualiser à distance sur un écran l'opération d'impression en cours** (en temps réel), directement depuis le **panneau de contrôle de la page web** de « **Repetier** » (voir la figure 1).

Nous devons d'abord installer une **application qui permet l'acquisition d'images à partir de la webcam connectée au port USB** et ensuite lancer « Repetier ».

Nous avons utilisé le logiciel « **Motion** » car en tant que logiciel de vidéo surveillance, il est simple à utiliser et dispose d'un large éventail de paramètres personnalisables. Pour installer « **Motion** », la première étape consiste toujours à nous assurer que nous avons installé sur notre **RaspberryPi** les dernières versions des logiciels et pilotes, en tapant à partir du terminal du **RaspberryPi** les commandes :

***sudo apt-get update***
et
***sudo apt-get upgrade***

À ce stade, nous pouvons installer « **Motion** » avec la commande :

***apt-get install motion***

Pour que ce dernier exécute correctement sa tâche de serveur de webcam, nous devons modifier certains paramètres.

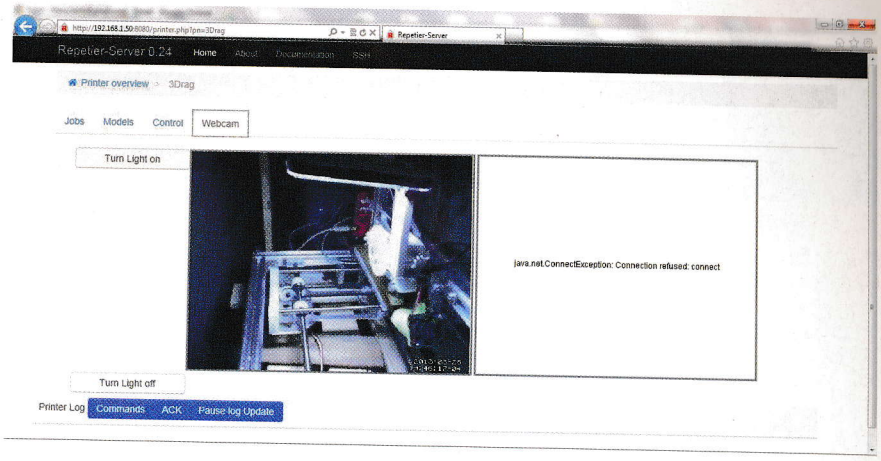



**Figure 1 : panneau de contrôle de la page web de « Repetier ».**

Tapons la commande :

***sudo nano /etc/motion/motion.conf***

nous pouvons le configurer en cherchant la ligne :

***Daemon = OFF***

remplacons ***OFF*** par ***ON***, ce qui permettra à « **Motion** » de s'exécuter en tâche de fond, c'est à dire en arrière-plan.

Cherchons la ligne :

***webcam_localhost = ON***

et modifions ***ON*** par ***OFF***, ce qui permettra également à d'autres ordinateurs connectés à notre réseau de visualiser l'image de la page web créée par « **Motion** ».

Toujours avec l'éditeur « **Nano** », nous pouvons modifier d'autres paramètres, tels que la fréquence des images par seconde ou la taille en pixels de l'image, mais il est recommandé de laisser les paramètres par défaut afin de ne pas trop accaparer le CPU du **RaspberryPi** qui n'est pas très puissant.

Pour lancer le service « **Motion** » en tâche de fond, nous devons modifier la configuration comme ceci :

***sudo nano /etc/default/motion***

remplacons le « ***no*** » de ***start_motion_daemon= no*** par ***yes***

Enfin, nous pouvons exécuter le programme avec l'instruction :

***sudo service motion start***

et vérifier que cela fonctionne en allant à l'**adresse IP** du **RaspberryPi** au **port 8081**.

Pour visualiser directement l'image à partir de la page de contrôle de « **Repetier** », nous avons besoin de faire quelques modifications.

Nous devons d'abord copier l'application Java « **Cambozola.jar** » dans le répertoire « **/var/lib/Repetier-Server/www** ».

Cette application est téléchargeable à l'adresse : **http://www.charliemouse.com/code/cambozola/**, nous mettrons le **fichier directement en téléchargement sur nos sites dédiés** à la **3DRAG** et au **RaspberryPi**.

Ouvrons le fichier « **printer.php** » qui se trouve dans le répertoire d'installation du serveur « **Repetier** » comme mentionné ci-dessus.

Maintenant, il suffit d'ajouter au fichier « **Control Panel** » à l'intérieur de la balise ***<body>*** du fichier php la ligne de code suivante :

***<applet code=com.charliemouse.cambozola.Viewer archive=cambozola.jar width="320" height="240" style="border-width:1; border-color:gray; border-style:solid;"> <param name=url value="http://localhost:8081"> </applet>***

Il apparaît alors sur la page des commandes manuelles de l'imprimante une fenêtre avec l'image de la webcam mise à jour chaque seconde.

## Installation de Repetier-Server sur RaspberryPi

1. Allez à l'adresse : **http://www.repetier.com/repetier-server-download/** ou sur notre site dédié au RaspberryPi : **http://raspberrypi.electroniquemagazine.com/telechargement.html.**
2. Téléchargez le fichier « **Repetier-Server 0.24 for Raspberry Pi** »

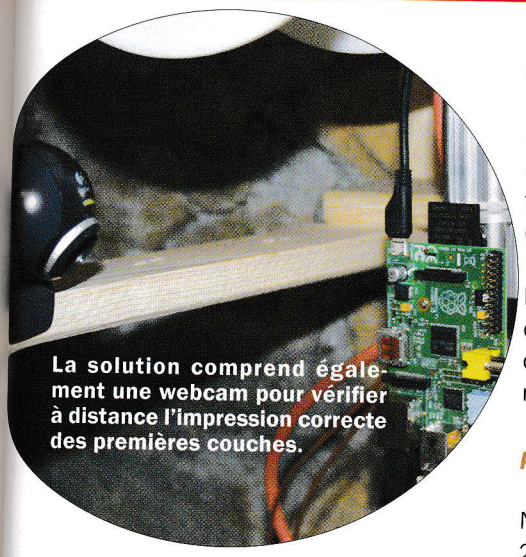



3. Déplacez le fichier dans le répertoire du RaspberryPi « **/home/pi** » en utilisant **WinSCP**.
4. Décompressez le fichier avec WinSCP :
    1. Cliquez avec le bouton droit de la souris sur le fichier que vous venez de déplacer ;
    2. Allez dans « Commandes personnalisées » ;
    3. Sélectionnez « UnTar/GZip »... ;
    4. Confirmez en cliquant sur « OK ».
5. À ce stade, ouvrez le dossier que vous venez de créer : ***Repetier-Server-Pi***
6. Dans le fichier « **install.txt** » vous trouverez les instructions pour installer « Repetier-Server » à partir de lignes de commandes. Vous pouvez également effectuer les étapes avec WinSCP :
    1. Modifiez la configuration contenue dans le fichier « *configs* » comme suit :
    - Version=1.0
    - Active=false
    - Name=3Drag
    - slugName=3Drag
    - device=/dev/serial/by-id/<vous trouverez le nom de votre périphérique en connectant l'imprimante et en vous rendant dans le répertoire mentionné>
    - baudrate=250000
    - pingPong=false
    - protocol=0
    - okAfterResend=true
    - Complétez les autres options avec vos paramètres et vos préférences
    - tempUpdateEvery=1
    - Enregistrez le nouveau fichier avec le nom « 3Drag.conf »
    2. Copiez le fichier exécutable « ***RepetierServer*** » dans le dossier « ***/usr/bin*** »
    3. Copiez le fichier « ***repetier-server.conf*** » dans « ***/etc*** »
    4. Créez le dossier « ***Repetier-Server*** » dans « ***/var/lib*** »
    5. Créez les dossiers « ***www*** », « ***configs*** », « ***storage*** » et « ***languages*** » dans le nouveau répertoire « **/var/lib/Repetier-Server** »
6. Copiez le fichier « **3Drag.conf** » *dans le répertoire* « ***/var/lib/Repetier-Server/configs*** »
7. Copiez les contenus des répertoires « ***www*** » et « ***languages*** » dans les dossiers respectifs du répertoire « **/var/lib/Repetier-Server** »
8. Pour vous assurer que le serveur démarre automatiquement lorsque vous allumez le RaspberryPi, effectuez les opérations suivantes :
    - Copiez le fichier « **Repetier-Server.init** » dans « **/etc/init.d/Repetier-Server** »
    - Donnez les autorisations de lecture et d'exécution des fichiers en tapant la ligne de commande « ***sudo chmod 755 /etc/init.d/Repetier-Server*** » à partir du terminal
    - Activez le script de démarrage avec la commande : ***sudo update-rc.d Repetier-Server defaults***
    - Pour activer, désactiver et redémarrer le serveur manuellement tapez les commandes suivantes :

***sudo /etc/init.d/Repetier-Server start***
***sudo /etc/init.d/Repetier-Server stop***
***sudo /etc/init.d/Repetier-Server restart***

9. L'installation est terminée, vous pouvez maintenant accéder au serveur avec votre navigateur préféré à l'adresse : **<IP du Raspberry>:8080**

Il est conseillé de régler la carte avec une adresse IP statique afin qu'elle ne change pas à chaque redémarrage.



La solution comprend également une webcam pour vérifier à distance l'impression correcte des premières couches.

## Contrôle du ventilateur et de l'éclairage de l'imprimante à distance

Poursuivons dans la personnalisation du poste de travail de l'imprimante grâce à la gestion via la page web, afin de contrôler le ventilateur et l'éclairage de la **3DRAG**. Pour cela nous devons effectuer quelques changements simples.

Commençons par le logiciel présent sur la carte contrôleur. Pour ceux qui ne l'ont pas déjà fait, le firmware **Marlin** est disponible dans la section « **Logiciels** » de notre site : **www.3dprint.electroniquemagazine.com**.

Une fois téléchargé, nous devons le **modifier** (l'éditer), le **compiler** et le « **charger** » dans la carte **Sanguinololu**. Pour cela nous avons besoin de la version **0023** de l'**IDE Arduino** qui est disponible dans la section « **Logiciels** » de notre site : **www.3dprint.electroniquemagazine.com**.

Commençons par éditer et modifier le fichier « **pins.h** », en recherchant la section relative à la carte **Sanguinololu** et en modifiant la ligne :

```
#define FAN_PIN
```

Puis, remplacer « - 1 » (ce qui désactive la fonction) par « 4 » (la broche 4 du microprocesseur Atmel est la seule inutilisée et qui est en mesure de générer un signal PWM). Ensuite, nous ajoutons la déclaration de la broche qui allume/éteint la lumière de cette manière :

```
#define LIGHT_PIN  27
```

la broche « 27 » correspond à la broche « 4 » (entrée analogique), mais elle peut très bien être configurée en « sortie » digitale.

Maintenant, nous pouvons nous consacrer au fichier « **Marlin.pde** », ajoutons dans la routine « **void setup ()** » la déclaration suivante :

```
pinMode (27, OUTPUT);
```

Nous avons donc configuré la broche 27 en sortie digitale.

A ce stade nous devons nous assurer que lors de la réception d'une commande donnée, la lumière s'allume ou s'éteint.

Nous devons ajouter à l'instruction « **switch** » relative aux commandes « **M** » qui se trouvent dans la routine « **void loop ()** », 2 instructions « **case** » avec un nombre égal à celui de la commande (pas encore utilisée par le firmware) que nous voulons associer à la mise sous tension ou hors tension de la lumière, tel que :

```
case 800:
    WRITE (LIGHT_PIN, HIGH);
    break;
case 801:
    WRITE (LIGHT_PIN, LOW);
    break;
```

L'instruction « **switch** » permet de faire plusieurs tests de valeurs sur le contenu d'une même variable.

Ce branchement conditionnel simplifie beaucoup le test de plusieurs valeurs d'une variable, car cette opération aurait été compliquée (mais possible) avec des « **if** » imbriqués.

Lorsque l'expression testée est égale à une des valeurs suivant un « **case** », la liste d'instructions qui suit celui-ci est exécutée. Maintenant, nous pouvons enregistrer le nouveau firmware. Pour le transférer dans la carte Sanguinololu nous devons télécharger le fichier « **Sanguino1284p.zip** » qui se trouve dans la section « **Logiciels** » du site **www.3dprint.electroniquemagazine.com**.

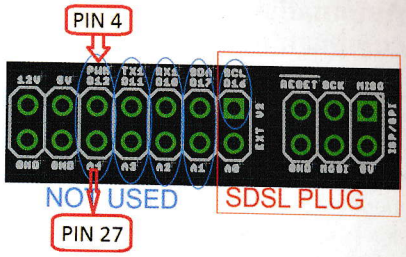



**Figure 2 : brochage des pins 4 et 27.**

Ensuite, il faut extraire et copier le fichier dans la carte Arduino.

Maintenant, nous pouvons ouvrir l'**IDE 0023** et sélectionner dans le menu :

« ***Tools > Board > Sanguino1284p*** ».

Coupons l'alimentation de l'imprimante, repérons le bouton « **RESET** » (réinitialisation) de la carte Sanguinololu et préparons-nous à le presser quelques secondes après avoir démarré le téléchargement à partir de l'IDE.

En ce qui concerne le matériel, il suffit de gérer les signaux produits par les sorties des deux MOSFET connectées au ventilateur et à l'éclairage (voir la figure 2).

***N.B** : à cause d'une erreur de fabrication, la broche 4 du microprocesseur Atmega644p utilisée pour commander le ventilateur, ne correspond pas à la broche portant le même nom sur la carte « Sanguinololu 1.3 », mais bien à la broche 12.*

Enfin, pour être en mesure d'allumer et d'éteindre la lumière directement à partir de la page web, nous devons ajouter deux boutons au fichier « **printer.php** » en copiant les deux lignes de code dans le « **Control Panel** » comme ci-dessous :

```
<button class="btn span2 inputonline notprinting" onclick="sendCmd('M800')"><?php _("Turn Light on")?></button>

<button class="btn span2 inputonline notprinting" onclick="sendCmd('M801')"><?php _("Turn Light off")?></button>
```

## Automatisons les commandes

En profitant de la flexibilité du programme « **Slic3r** », nous pouvons faire en sorte que le ventilateur soit correctement géré par des commandes à partir du « **G-code** ». Grâce aux options qui sont spécifiées ci-après, nous pouvons insérer aux endroits appropriés dans « **Slic3r** » les commandes pour l'activation et l'arrêt du ventilateur. Voici où intervenir.

Ouvrez la console de « **Slic3r** », dans la page de configuration des paramètres de l'imprimante allez dans le menu « **Filament Settings** » (paramètres des filaments) et ensuite dans la section « **Cooling** » (refroidissement). Activez le refroidissement automatique et enregistrez en cliquant sur l'icône en forme de disque à côté du nom du profil.

Ensuite, allez dans le menu « **Printer Settings** » dans la section « **Custom G-Code** », puis ajoutez les commandes pour allumer et éteindre l'éclairage du plateau d'impression à la liste des commandes qui sont envoyées avant et après l'impression (M800 pour allumer et M801 pour éteindre), puis sauvegardez.

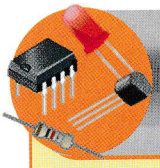
## Imprimons notre premier objet à partir de « Repetier-Server »

Après l'ouverture de la page web générée par notre « **Repetier-Server** », c'est-à-dire la page d'Accueil ou « **Home** », cliquez sur le bouton « **Activate** » pour connecter l'imprimante **3DRAG** au **RaspberryPi**. Après la modification du statut de « **Offline** » (non connectée) à « **Online** » (connectée), cliquez sur « **Use** ».

Maintenant, vous pouvez aller à la section « **Control** » et effectuer une calibration des axes en les ramenant à la **position** « **Home** », cliquez sur l'icône en forme de « maison ». Enfin, cliquez sur l'onglet « **Job** » et chargez le fichier « **G-code** » contenant les instructions qui doivent être envoyées à l'imprimante pour imprimer l'objet.

A la fin du chargement du fichier, cliquez sur « **Start** » (Démarrer), l'impression de l'objet débutera une fois que l'extrudeuse aura atteint la température de consigne (prédéfinie). A partir de la page « **Job** », vous pouvez voir la progression de l'impression.

Dans le panneau de contrôle (« Control Panel »), il est possible de visualiser l'impression en temps réel grâce à la webcam (l'image est mise à jour une fois par seconde). ■



### Comment se procurer la 3DRAG et/ou le RaspberryPi

L'imprimante **3DRAG** et le **RaspberryPi** sont disponibles auprès de notre partenaire **COMELEC** (voir les publicités) sous forme de kit complet comprenant tous les équipements mécaniques, électriques et électroniques.

Les programmes mentionnés dans cet article sont téléchargeables gratuitement sur nos sites :

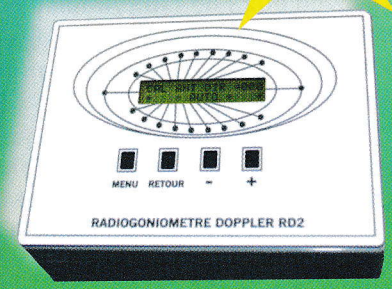
**www.3dprint.electroniquemagazine.com**
**www.raspberrypi.electroniquemagazine.com**

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

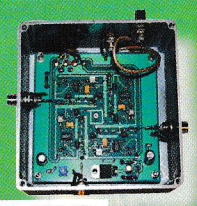

**GONIOMÈTRE DOPPLER DE 50 MHz à 1.2 GHz**

Réf. RD2
199,00 €
Vendu
sans antennes

• Commutation pour 4 antennes • Sélection d'impulsions vers le + 5V ou vers le 0V pour activer les antennes. • Rotation des antennes; CW ou CCW. • Contrôle indépendant de chaque antenne. • Auto calibration vers le devant du véhicule. • Afficheur LCD standard de 2 lignes X 16 caractères. • Un affichage similaire à 36 LED et aussi numérique "000-359" de la direction. • Tous les menus sont montrés clairement sur l'afficheur LCD. • Mémoire permanente pour toutes les calibrations et options. • Traitement principal du signal fait par le soft. • Microcontrôleur PIC 16F877, mémoire de programmation Flash, mémoire EEDATA, USART, ADC, chrono... • Mémorisation de la calibration de 3 radios. • Sortie chronométrée ou sur demande vers APRS, interface GPS. • Option d'affichage d'un S-mètre, l'entrée est ajustable de 0 < 2 à 5 V. pour un affichage de 00 < 99. • 7 niveaux de traitement du signal. Possibilité d'affichage instantané des données brutes. • Sélectivité Maximum des filtres audio analogues et numériques de +/- 0.1 Hz. • En cas de perte du signal, mémorisation de la dernière bonne direction. • Haut-parleur intégré et alimentation 12 Vdc. • Rétro-éclairage LED de l'afficheur. **Le Gonio Doppler RD2** présenté ici n'intègre pas de récepteur particulier. Il est prévu pour être utilisé conjointement à des matériels déjà existants, portatifs, mobiles (dans le cas de recherches sur le terrain) voire fixes. Ainsi, tout récepteur VHF ou UHF, disposant d'une sortie BF, peut être couplé à ce gonio Doppler capable de couvrir une très large plage de fréquences, en fonction des besoins (de 50 MHz à 1,2 GHz). Nous ne sommes donc plus limités, dans le cadre des recherches de balises de détresse, aux seules fréquences 121,5 (ou 121,375), 243 et 406 MHz





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

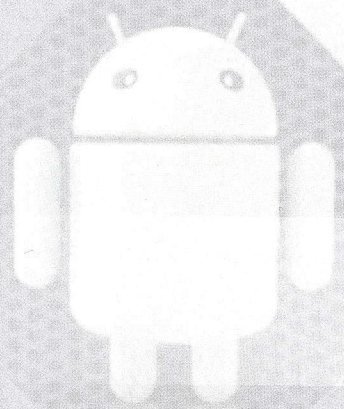
Comelec 12 / 2014 - Prix TTC

# PROGRAMMEZ AVEC ANDROID

## Deuxième partie

de Andrea Chiappori

**Dans cette deuxième partie, nous continuons le cours sur Android et plus particulièrement sur l'environnement de développement « Eclipse ». Nous allons émuler notre première application et l'exécuter sur un périphérique réel.**






Après l'introduction dans le monde du système **Android** avec le **1er article du numéro 128 d'Electronique et Loisirs Magazine**, nous allons programmer notre première application en utilisant l'environnement de développement « **Eclipse** » présenté dans l'article précèdent.

Avant de passer à la programmation, nous analyserons la structure du projet dans le détail. Nous serons ainsi en mesure de tester notre projet sur l'émulateur fourni par le **plugin** « **ADT** » (voir n° 128) pour Android afin de voir en action notre application.

Le projet que nous vous proposons n'a pas d'applications pratiques et il est dépourvu d'interface graphique utilisateur, mais il a surtout une valeur didactique qui vous permettra de comprendre le processus complet de programmation, de tests et de publication de votre propre application sur l'« **Android Market** », de nos jours « **Google Play** ». « **Google Play** » est une **boutique en ligne créée par Google** le 6 mars 2012 par fusion des services Android Market, Google Movies, Google ebookstore et Google Music. Elle regroupe une boutique d'application pour le système d'exploitation Android, une boutique de location de films et de séries télévisées, une boutique d'achat de musique, de livres, etc. Pour avoir accès à « Google Play » **il est nécessaire de posséder une adresse « Gmail »**. Si l'utilisateur n'en possède pas, il peut en créer une directement via l'application.

### Eclipse

« **Eclipse** » est un **environnement de développement** intégré (IDE) créé à l'origine par **IBM**, par la suite il est devenu **Open Source** et il est développé en **Java**. Cette application permet de développer sous plusieurs langages, par exemple en **C** ou **C++**, à condition d'installer les plugins nécessaires.

En effet, il est bon de rappeler qu'« **Eclipse** » **seul permet de développer des programmes en Java**, mais cela **n'est pas suffisant pour créer un projet sous Android**.

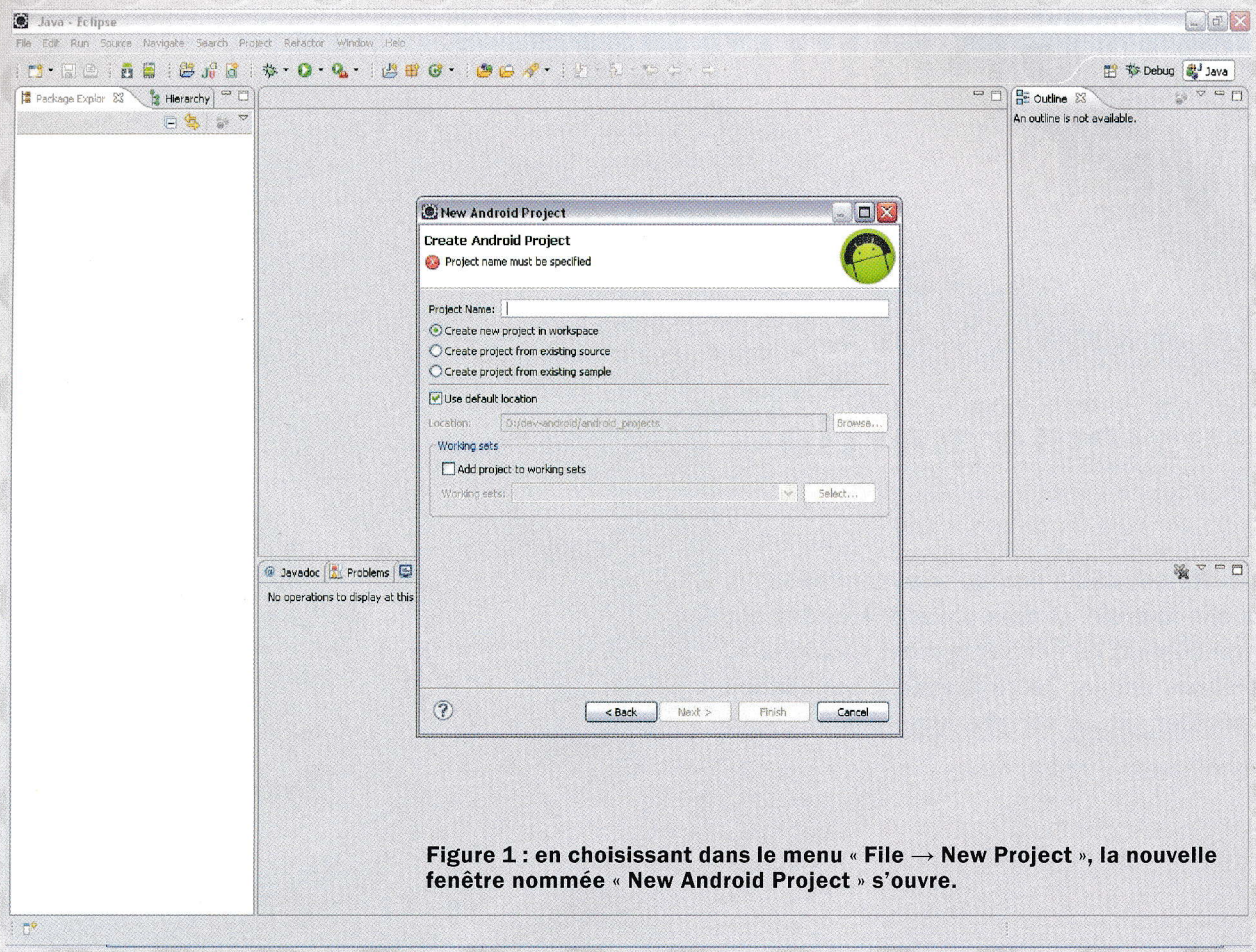



**Figure 1 : en choisissant dans le menu « File → New Project », la nouvelle fenêtre nommée « New Android Project » s'ouvre.**

Dans notre cas, nous allons programmer en **Java**, nous aurons besoin du plugin « **ADT** » (Android Developers Tools) que **nous avons déjà installé** (reportez-vous au 1er article du numéro 128 d'Electronique et Loisirs Magazine). Ce plugin permet à « **Eclipse** » de créer notre projet et de fournir les outils nécessaires tels que l'**émulateur**, les **pilotes** (drivers), les **bibliothèques** et un ensemble d'exemples.

Il existe d'autres environnements de développement pour la programmation sous Android (tels que IntellijIDEA et NetBeans pour n'en citer que quelques-uns), mais nous avons choisi d'utiliser « **Eclipse** » car il est le plus largement utilisé dans le domaine des applications embarquées. Beaucoup de lecteurs qui ont déjà utilisé « **Eclipse** » pour des plateformes autres qu'Android finiront par l'adopter, ils auront déjà les bases pour « naviguer » à travers l'environnement de développement.

D'autre part, cet article n'est pas un cours spécifique sur « **Eclipse** », au premier abord cela peut sembler un peu confus en raison de la présence de nombreuses fonctionnalités qui ne seront pas analysées dans ces pages, mais c'est un point de départ pour aborder la programmation sous le système Android. A ce stade, nous pouvons commencer.

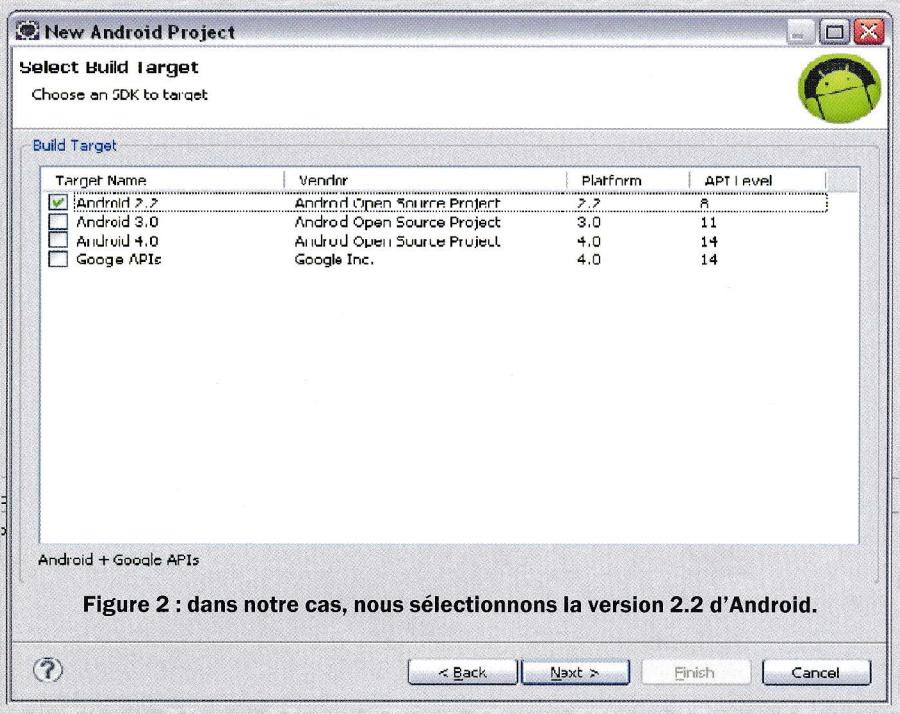



**Figure 2 : dans notre cas, nous sélectionnons la version 2.2 d'Android.**

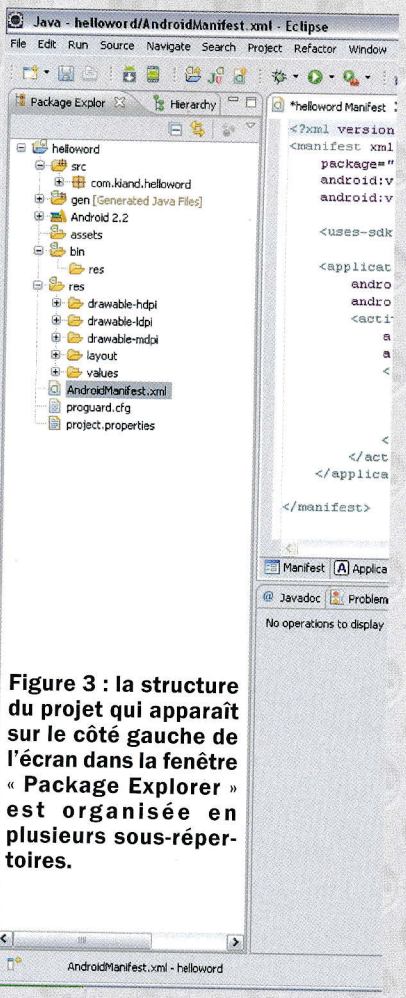



**Figure 3 : la structure du projet qui apparaît sur le côté gauche de l'écran dans la fenêtre « Package Explorer » est organisée en plusieurs sous-répertoires.**

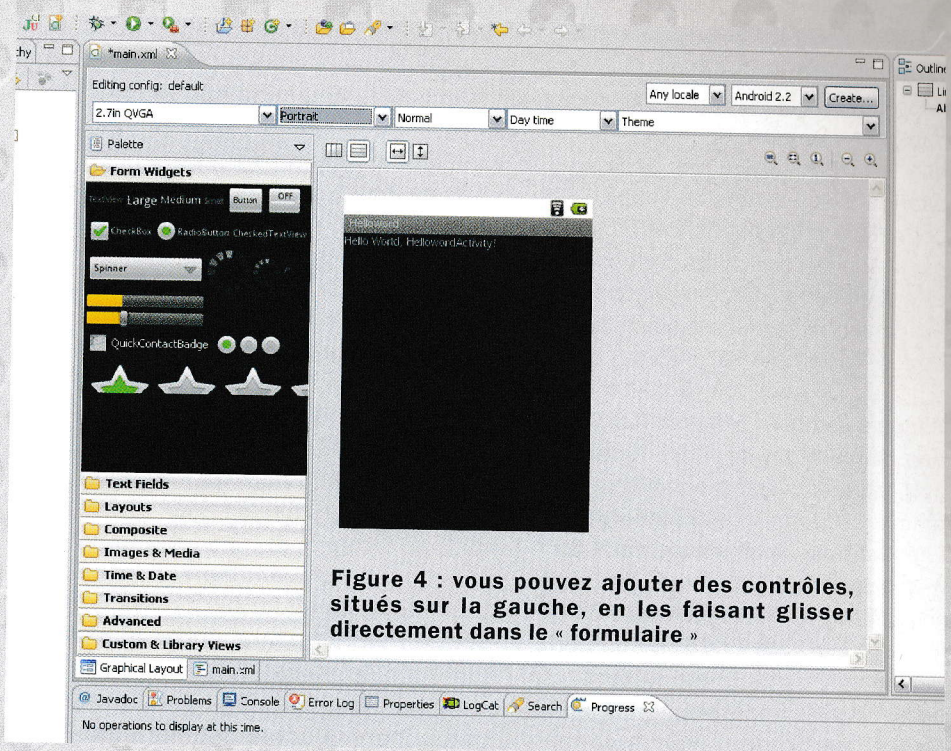



**Figure 4 : vous pouvez ajouter des contrôles, situés sur la gauche, en les faisant glisser directement dans le « formulaire »**

## Notre premier projet

Une fois que vous avez démarré « **Eclipse** », vous pouvez créer votre premier projet en choisissant dans la fenêtre principale « **File → New Project** » (Fichier → Nouveau Projet) dans le menu (en haut vers la gauche). En cliquant sur « **Android Project** » (Projet Android), une nouvelle fenêtre nommée « **New Android Project** » s'ouvre comme vous pouvez le voir sur la figure 1.

Lançons l'assistant qui permet d'utiliser les paramètres déjà vus dans l'article précédent. N'oubliez pas de choisir, comme vous pouvez le voir à la figure 2, les **bibliothèques appropriées en fonction de la version d'Android installée sur votre périphérique**. Dans notre cas, nous avons choisi la version **2.2** car elle est utilisée sur notre Smartphone qui « tourne » sous **Android Froyo** . Vous pouvez bien sûr travailler avec des versions plus récentes. Une fois cette phase terminée, « **Eclipse** » va créer un squelette d'un projet qui vous permettra d'afficher le classique « **Hello World** » (« Bonjour à tous ») sur l'écran en mode console.

## Structure du projet

Avant de commencer à modifier et/ou ajouter du code, il est bon de s'attarder sur la structure du projet qui est organisée en plusieurs sous-répertoires. Ceux-ci apparaissent sur le côté gauche de l'écran dans la fenêtre « **Package Explorer** » et montre l'arborescence des fichiers, vous pouvez aussi les afficher en naviguant avec l'**Explorateur Windows** dans le dossier de votre projet que vous venez de créer.

Vous pouvez voir à quoi cela ressemble en vous reportant à la figure 3, analysons la structure brièvement ci-dessous.

Le répertoire « **src** » contient les fichiers sources avec l'extension « **.java** ». Lors de la création du projet, ce répertoire contiendra un seul fichier avec l'unique classe principale de l'application. Tous les fichiers portant l'extension « **.java** » présents dans ce dossier seront compilés afin d'un construire un « exécutable ». Si nous voulons ajouter des classes, il suffit de créer un fichier dans le même répertoire et de rafraîchir l'affichage en appuyant sur la touche « **F5** » ou en utilisant la commande « **Refresh** » (« Actualiser »). Ce dossier contient le code du projet, mais comme nous allons le voir maintenant il n'est pas suffisant pour générer le projet final.

Le répertoire « **gen** » contient les fichiers générés lors de la compilation du projet. Ceux-ci contiennent les références des ressources utilisées dans le projet, par exemple les boutons, les images, les fichiers audio ou des caractères, mais comme nous ne devons pas les modifier, nous pouvons temporairement les ignorer.

Le répertoire « **Android 2.2** » est en réalité un lien vers un fichier « **.jar** » précompilé qui n'est pas situé dans le répertoire du projet, mais dans le répertoire créé lors de l'installation du **plugin** « **ADT** ». Il contient toutes les bibliothèques qui nous permettent de développer sous Android.

« **Eclipse** » nous permet de visualiser et lister toutes les fonctions que chacun d'eux présente. Cette fonction, même si elle n'est pas indispensable, peut être utile surtout pour les plus curieux, car elle fournit des informations sur la façon de prendre le contrôle du dispositif et peut suggérer de nouvelles idées pour nos futures applications.

Le répertoire « **bin** » contient les **fichiers binaires** qui sont utilisés par l'application : l'application elle-même dans son format natif (« **.dex** ») ainsi que le fichier généré lors de la compilation avec l'extension « **.apk** » qui permet d'installer l'application sur le périphérique. Les fichiers portant cette extension correspondent par analogie aux fichiers exécutables sous Windows portant l'extension « .exe ».

Rappelons que le **résultat final sera une compilation de notre fichier « .apk »** qui est un fichier compressé contenant à l'intérieur des **ressources de l'application**, l'**exécutable natif** pour **Android** (« **.dex** ») et d'autres fichiers que vous pouvez voir en **renommant le fichier « .apk »** en « **.zip** ». Le fichier qui porte l'**extension «. apk »** sera **le seul** que nous mettrons **en ligne** sur l'« Android Market » ou « Google Play » lorsque nous aurons terminé notre projet.

Le répertoire « **res** » contient les **ressources** utilisées par notre projet. Ce dossier est aussi divisé en plusieurs dossiers et sous-dossiers en fonction du type des spécifications des ressources. Il est important d'analyser les dossiers et les fichiers de ce répertoire, car ils prennent un aspect fondamental en ce qui concerne la « charte graphique », le style et l'interface graphique de l'application et ne doivent donc pas être négligés car ils seront le premier contact direct avec l'utilisateur final.Analysons donc les dossiers du répertoire « **res** ».

Le dossier « **values** » contient les valeurs présentes à l'intérieur d'un fichier « .xml » nommé « **strings.xml** » qui contient les chaînes de caractères utilisées par notre application. Par exemple, dans le fichier de notre projet que nous venons de créer, le fichier contient la chaîne de caractère « Hello World ». Nous pouvons bien sûr ajouter plusieurs chaînes de caractères, nous verrons plus tard comment faire référence à elles afin de les utiliser en les rappelant à l'aide du code de l'application. Les **valeurs** du dossier « **values** » peuvent également être contenues dans plusieurs fichiers de type « **chaînes de caractères** », par exemple lorsque nous voulons réaliser une application qui prend en charge plusieurs langues de différentes nationalités.

Le dossier « **layout** » contient un fichier « **.xml** » qui **représente la mise en page** (ou charte graphique) de l'interface graphique de notre application. Pour le moment nous voyons un seul fichier (« **main.xml** ») qui est de petite taille, mais plus tard nous verrons comment le compléter en fonction de nos besoins, en ajoutant des boutons et des contrôles afin d'agrémenter l'interface.

Si l'application utilise une boîte de dialogue plutôt qu'une fenêtre avec des options ou même un menu, il suffit d'ajouter un autre fichier « .xml » qui contient un graphique, nous verrons plus tard comment l'appeler à partir du code. En cliquant sur l'un de ces fichiers, l'écran principal s'ouvre (visible sur la figure 4) et apparaît alors un « formulaire » dans lequel nous pouvons ajouter des contrôles (situés sur la gauche) en les faisant glisser directement sur le « formulaire ».

**Eclipse**, sur la base de nos actions, va générer automatiquement le fichier « .xml » (dans notre exemple « main.xml ») que nous pouvons voir en cliquant sur l'onglet situé en bas de la fenêtre qui indique le nom du fichier (voir la figure 5). Rien ne vous empêche, et il est même conseillé, de changer le « look » de votre application en insérant des contrôles et en définissant leurs propriétés (couleur, position, style) directement dans le fichier « .xml ». Cela suppose une bonne connaissance des bibliothèques Android, mais cela vous permet un meilleur contrôle et une plus grande clarté lorsque vous devez changer ou ajouter d'autres éléments. Il est possible, après avoir modifié manuellement le fichier « .xml », de visualiser le résultat en mode graphique en cliquant sur l'onglet « **Graphical Layout** ». Nous aborderons cet aspect en détail.

Les dossiers « **drawable-hdpi** », « **drawable-mdpi** », « **drawable-ldpi** » contiennent les images utilisées par l'application afin d'adapter celles-ci aux **différentes tailles d'écrans** des périphériques (Smartphones, Tablettes, P*hablettes).*

Au moment de la création du projet, ces dossiers contiennent déjà l'icône classique qui représente notre application Android. Dans chacun de ces dossiers se trouvent des fichiers images au format « **.png** », portant le même nom, mais avec une résolution différente en fonction de la taille de l'écran.

En particulier dans le dossier « **drawable-hdpi** » nous trouvons les images des icônes ayant une grande définition et conçues pour les **écrans à haute résolution** (généralement 480 x 640 dpi), dans le dossier « **drawable-mdpi** » nous avons

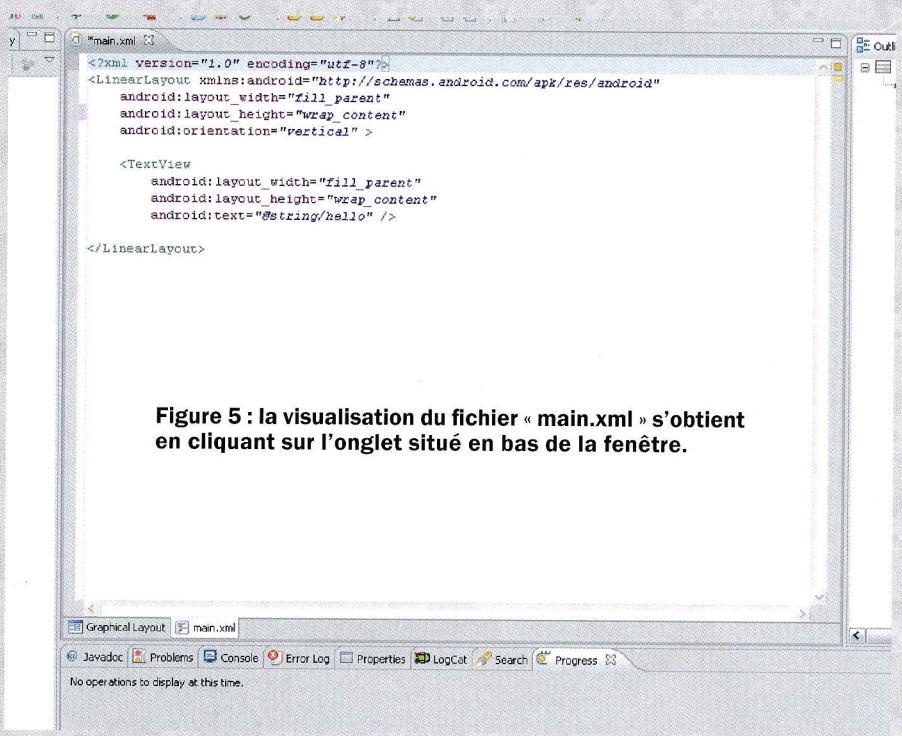



**Figure 5 : la visualisation du fichier « main.xml » s'obtient en cliquant sur l'onglet situé en bas de la fenêtre.**

les mêmes images des icônes, mais avec une définition moindre, adaptée pour les **écrans de moyenne résolution** (320 x 480 dpi) et enfin dans le dossier « **drawable-ldpi** » nous trouvons les mêmes images en **basse résolution** pour des écrans de petite taille ( résolution de 240 x 320 dpi).

Notez que le terme « **dpi** » veut dire « **Dots Per Inch** » ou « **Point par Pouce** » et représente une unité de précision utilisée pour définir la résolution.

Bien évidemment lorsque nous ajouterons nos graphiques personnels, nous devrons tout simplement générer la même image avec une résolution différente et l'insérer dans le dossier approprié. La résolution des images doit correspondre à certaines normes qui sont expliquées en détail à l'adresse suivante :

***http://developer.android.com/guide/practices/screens_support.html***

Pour les lecteurs ne maîtrisant pas bien la langue de Shakespeare, voici un résumé succinct concernant la gamme d'écrans pris en charge. À partir de la version d'**Android 1.6** (API niveau 4), un **support pour plusieurs tailles et résolutions d'écrans** est fourni, reflétant ainsi les différentes configurations d'écrans qu'un périphérique peut avoir. Vous pouvez utiliser les fonctions du système Android pour optimiser l'interface utilisateur de votre application pour chaque configuration d'écran et vous assurer que votre application s'affiche correctement et offre le meilleur confort possible pour l'utilisateur sur chaque type d'écran.

Pour simplifier la façon dont vous devez concevoir vos interfaces pour de multiples écrans, Android divise la gamme des **tailles** et des **résolutions en 4 tailles différentes** :

- ***petite*** *(small) ;*
- ***normale*** *(normal) ;*
- ***grande*** *(large) ;*
- ***très grande*** *(xlarge).*

et en **6 résolutions différentes** :

- ***ldpi*** qui correspond à faible (low) avec une résolution de 120 dpi environ ;
- ***mdpi*** qui correspond à moyenne (medium) avec une résolution d'environ 160 dpi ;
- ***hdpi*** qui correspond à haute (high) avec une résolution de 240 dpi ;
- ***xhdpi*** qui correspond à très haute (extra-high) avec une résolution d'environ 320 dpi ;

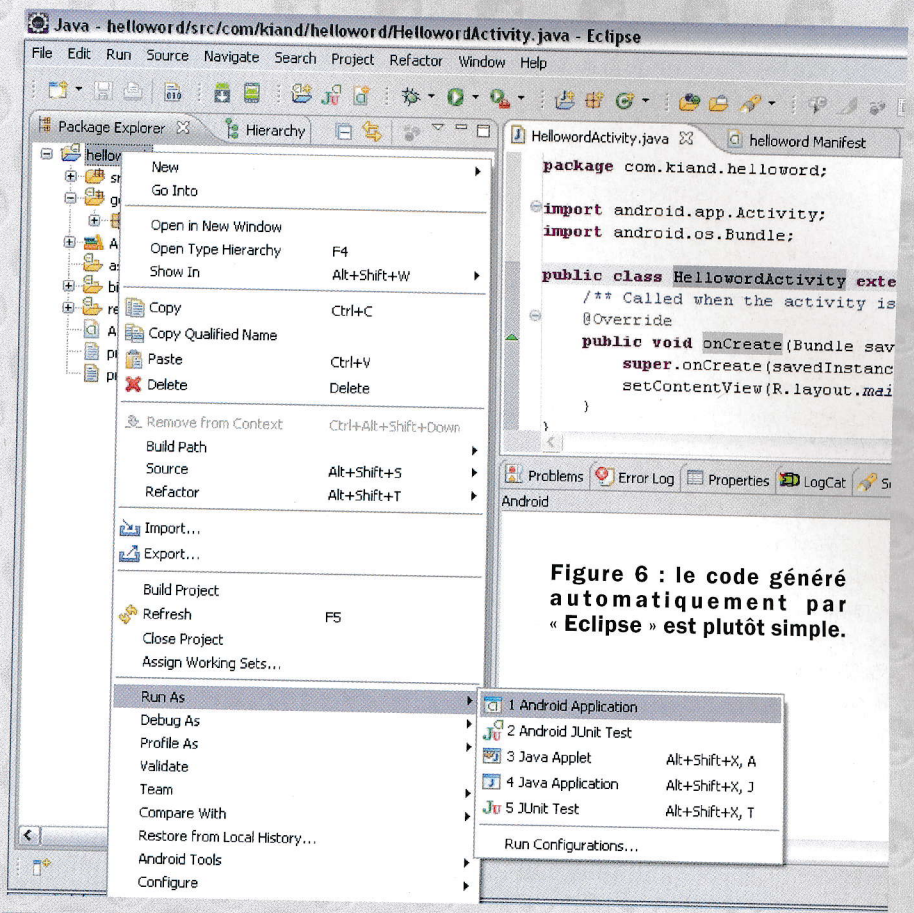



**Figure 6 : le code généré automatiquement par « Eclipse » est plutôt simple.**

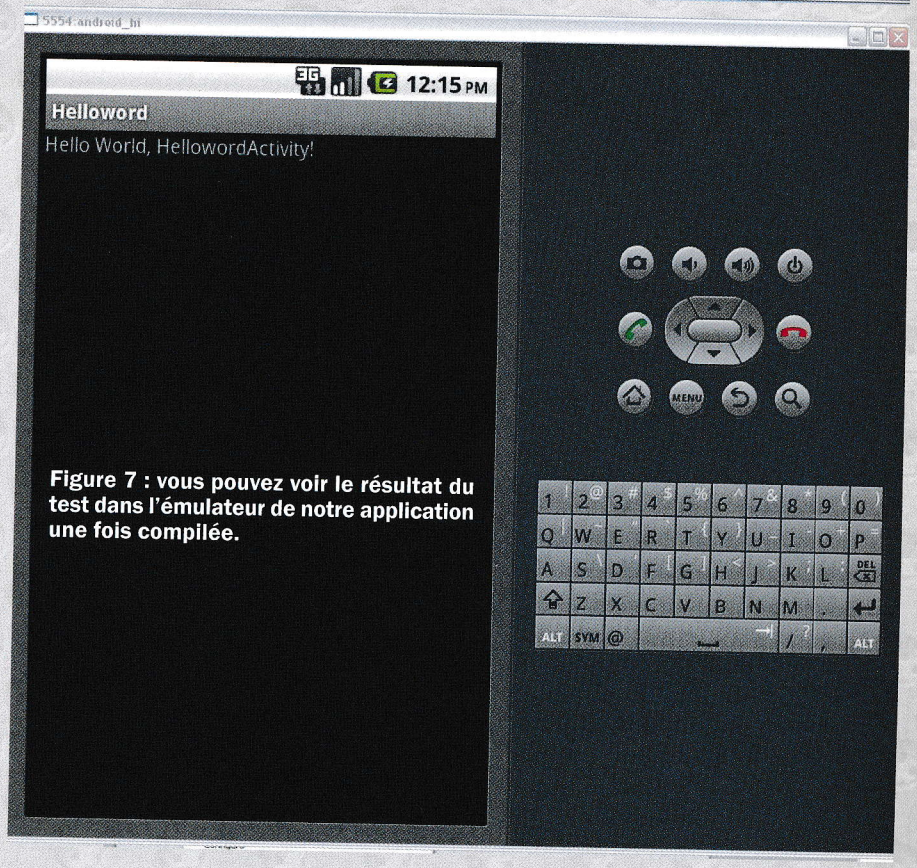



**Figure 7 : vous pouvez voir le résultat du test dans l'émulateur de notre application une fois compilée.**

- ***xxhdpi*** qui correspond à très très haute (extra-extra-high) avec une résolution d'environ 480 dpi ;
- ***xxxhdpi*** qui correspond à très très très haute (extra-extra-extra-high) avec une résolution de 640 dpi.

Les tailles et les résolutions se sont généralisées autour de ce que l'on peut considérer comme une configuration de base qui est la norme « mdpi ». Cette référence est basée sur la configuration de l'écran du « **HTC G1** » (HTC Dream) qui fut **le premier Smartphone commercialisé avec le système Android en 2008**. Il disposait d'un écran TFT-LCD de 3,2» de 320 x 480 en HVGA. Jusqu'à la version « Android 1.6 » c'était la seule configuration d'écran qu'Android prenait en charge.

Chaque taille et résolution normalisée correspond à une gamme réelle d'écrans. Par exemple, 2 appareils qui disposent tous les 2 d'une taille d'écran correspondant à la norme peuvent avoir des tailles d'écran réelles qui diffèrent légèrement. De même, 2 appareils qui disposent d'une résolution de type « ***hdpi*** » peuvent avoir des résolutions réelles légèrement différentes. **Android fait abstraction de ces différences**, de sorte que vous puissiez utiliser l'interface conçue pour des tailles et résolutions normalisées et laisser gérer les derniers ajustements au système.

Lorsque vous concevez votre interface pour différentes tailles d'écran, vous vous apercevez que chaque modèle nécessite un minimum d'espace. Ainsi, chaque taille d'écran normalisée a une résolution minimale associée qui est définie par le système. Ces dimensions minimales sont exprimées en unités « **dp** », les mêmes que vous devez utiliser pour définir votre mises en page, ce qui permet au système d'éviter les changements de résolutions.

Les écrans de type « **xlarge** » ont au moins une taille de **960 dp x 720 dp**, ceux de type « **large** » au moins une taille de **640 dp x 480 dp**, ceux de type « **normal** » une taille de **470 dp x 320 dp** et enfin ceux de type « **small** » une taille de **426 dp x 320 dp**.

Voici donc ce qu'il faut retenir sans entrer dans les détails sur l'utilisation des images et plus généralement sur l'interface utilisateur par rapport aux différents types d'écrans.

Le dossier « **assets** » peut contenir des **ressources de toutes sortes**, telles que des images, de l'audio, du texte, des fichiers binaires de données. Il n'est généralement pas beaucoup utilisé, mais il dispose d'une librairie « **getAssets()** » grâce à laquelle vous pouvez accéder à ces fichiers et les utiliser dans votre application.

Enfin, pour conclure avec la structure des fichiers du projet, nous trouvons le fichier « **manifest.xml** » que nous pouvons éditer et qui contient les **informations spécifiques liées à l'application**, comme son **nom**, le **numéro de la version** de l'application, la fonction d'origine, le **style graphique**, l'orientation et la **liste des permissions**.

En effet, c'est une bonne idée de mettre à disposition de l'utilisateur final les informations sur les permissions, c'est-à-dire qui peut utiliser l'application.

Par exemple, si l'application doit utiliser le module Bluetooth ou doit avoir accès au réseau, plutôt que mettre ces informations dans les fichiers « internes » de la carte SD, il est préférable de le spécifier au moment de l'installation afin que l'utilisateur final puisse avoir le choix entre continuer ou quitter l'installation.

## Aperçu du code

Nous avons vu dans l'article précédent qu'Android dispose d'un **plugin ADT** qui permet d'émuler et de tester notre application avant de l'utiliser directement sur un appareil réel. A ce stade, nous pouvons créer notre première application « **apk** », la tester d'abord sur l'émulateur et ensuite sur l'appareil Android.

Avant de lancer la compilation de l'application, analysons le code généré automatiquement par « **Eclipse** » afin de se familiariser avec les commandes de base de l'interface d'« **Eclipse** ». Comme vous pouvez le constater sur la figure 6, le code résultant est plutôt simple et peut être utilisé pour introduire quelques notions que nous verrons plus en détail dans les prochains articles.

Pour bien comprendre notre application, il nous faut d'abord connaitre les **éléments de base d'une application Android**. Les éléments de base sont :

### 1. *Activities* (Activités en Français)

Une activité est la composante principale pour une application Android. Elle représente l'implémentation et les interactions de vos interfaces. Prenant l'exemple d'une application qui liste toutes les images présentes dans votre téléphone, le projet pourrait se décomposer comme ci-dessous :

- une vue pour afficher la liste des images ;
- une activité pour gérer le remplissage et l'affichage de la liste ;
- pour rajouter, supprimer des images, il suffit d'ajouter d'autres activités.

### 2. *Services*

Un service, à la différence d'une activité, ne possède pas de vue mais permet l'exécution d'un algorithme sur un temps indéfini. Il ne s'arrêtera que lorsque la tâche sera finie ou que son exécution sera arrêtée. Il peut être lancé à différents moments :

- au démarrage du téléphone ;
- au moment d'un événement (arrivée d'un appel, SMS, mail, etc...) ;
- au lancement de votre application ;
- lors d'une action particulière dans votre application.

### 3. *Broadcast and Intent Receivers*

Un Broadcast Receiver, comme son nom l'indique, permet d'écouter ce qui se passe sur le système ou sur votre application et de déclencher une action que vous aurez prédéfinie. C'est souvent par ce mécanisme que les services sont lancés.

### 4. *Content providers*

Les « **content providers** » servent à accéder à des données, depuis votre

application vous pouvez accéder aux contacts stockés dans le téléphone, à l'agenda, aux photos, ainsi qu'à d'autres données grâce aux « content providers ».

Notre application basique contient une seule classe qui est la classe « **HellowordActivity** », elle est dérivée de la classe de base « **Activity** » d'où la nécessité d'importer « **android.app.Activity** ». La classe « **Activity** » sous Android peut être considérée comme un écran graphique qui peut être associé à une mise en page. Celle-ci est **configurée à l'aide d'un fichier** « **.xml** » présent dans le dossier « **layout** », comme nous l'avons décrit précédemment et qui contient l'interface graphique réelle.

Cette association est faite grâce à la fonction « **setContentView ()** » qui est utilisée avec l'argument « **R.layout.main** » afin d'afficher les données du fichier « **main.xml** » :

```
setContentView(R.layout.main)
```

affiche les données incluses dans le fichier « main.xml ».

Plus précisément, pour entrer dans la philosophie d'Android, la **classe** « **R** » est générée directement par « **Eclipse** » et se trouve dans le fichier « **R.java** » présent dans répertoire « **gen** ». Dans la **classe** « **R** » sont contenues d'autres classes qui correspondent aux ressources de l'application et qui contiennent une **valeur unique** (**Id**) d'éléments individuels (images, audio, chaînes de caractères, fichiers « .xml »). Le fichier « **R.java** » contient ainsi les **références vers l'ensemble des ressources du projet** (chaines de caractères, images, layouts, etc...).

Donc l'**argument** « **R.layout.main** » permet d'**utiliser la ressource spécifique contenue** dans le fichier « **main.xml** » que l'on retrouve dans le dossier « **layout** » lui-même contenu dans le répertoire « **res** » (ressources). Il est à noter que nous n'avons pas besoin de spécifier l'extension « .xml » du fichier « **main.xml** ».

Une application Android peut contenir plusieurs classes de type « **Activity** », et chacune peut être associée à un fichier de type ressources différent, de manière à présenter une mise en page différente avec une interface graphique différente.

La fonction « **setContentView ()** » doit être appelée une seule fois au **moment du démarrage de l'application**, par conséquent nous utilisons la **méthode** « **onCreate ()** » de la classe de base « **Activity** » à laquelle nous ajoutons la fonction « **setContentView ()** », après la méthode classique « **onCreate ()** » de la classe de base. « **OnCreate ()** » sert à **initialiser votre activité** ainsi que toutes les données nécessaires à cette dernière.

## Tests sur l'émulateur

Maintenant, pour **compiler** notre application, nous pouvons lancer le « **build** » (construction du projet) par un clic droit sur le nom du projet et en cliquant sur « **Build Project** » (Générer le Projet) dans le menu.

Dans le dossier « **bin** » vous trouverez alors le fichier « **helloworld.apk** » qui vient d'être généré. Si la compilation n'a pas réussi, vous pouvez visualiser les erreurs ou les avertissements dans la fenêtre du bas de l'environnement « **Eclipse** » et en particulier à l'onglet « **Problems** » où sont signalées les lignes contenant des erreurs.

A ce stade, nous pouvons tester l'application avec l'émulateur en faisant un clic droit sur le nom du projet, puis en cliquant sur « **Run As** » (Exécuter en tant que) et en sélectionnant « **Android Application** » (Application Android) comme indiqué en figure 6.

La première fois que vous démarrez l'émulateur, il se peut que le processus prenne plusieurs secondes, cependant vous serez en mesure de suivre les différentes étapes (et éventuellement les erreurs) du processus à travers la « **Console d'Eclipse** », après quoi vous verrez apparaître une fenêtre semblable à celle de la figure 7. Bien évidemment, l'application est très simple mais elle permet de mieux comprendre les mécanismes de la programmation sous Android.

Notez que la chaîne de caractères « **Hello world, HelloWordActivity!** » qui apparaît en haut de l'écran **n'est pas présente dans le code** du fichier « **HelloworldActivity.java** » mentionné précédemment.

L'explication est la suivante, la chaîne de caractères est insérée dans le fichier de ressources « **strings.xml** » dans le dossier « **res** » qui est identifié par le nom « **hello** », nous trouvons ce même nom dans le fichier des ressources « **main.xml** » qui représente notre mise en page. Ce fichier a été généré automatiquement en comprenant déjà une première commande simple, de type « **TextView** », dont les propriétés présentent la chaîne initiale « **hello** » (voir la figure 8).

Au cours de l'émulation, vous pouvez également afficher les messages du **journal des évènements** (logs) qui sont envoyés par Android vers « **LogCat** » de l'IDE « **Eclipse** » en bas de la fenêtre. Dans « **Eclipse** » ouvrez la perspective « **DDMS** » puis la vue « **LogCat** ». En fait la vue « **LogCat** » est la vue dans laquelle seront affichés les logs de votre application. Parfois rien ne s'affiche dans cette vue, il vous suffit de cliquer dans la vue « Device » dans l'émulateur ou sur le « device » que vous souhaitez, vos logs s'affichent dans le vue « **LogCat** ».

Nous pourrons cependant, comme nous allons le voir dans les articles suivants, insérer quelques lignes pour personnaliser le journal de notre application et de suivre l'évolution à travers la fenêtre.

## Les essais sur les appareils Android

Nous pouvons maintenant analyser la procédure pour télécharger notre application vers un périphérique réel. Il faut tout d'abord **installer les pilotes** (drivers) **USB** de l'appareil qui sont généralement fournis au moment de l'achat de l'appareil Android, de sorte que votre PC le reconnaisse correctement lorsqu'il est connecté au port USB.

Ensuite, vous aurez besoin de configurer l'appareil de façon à permettre

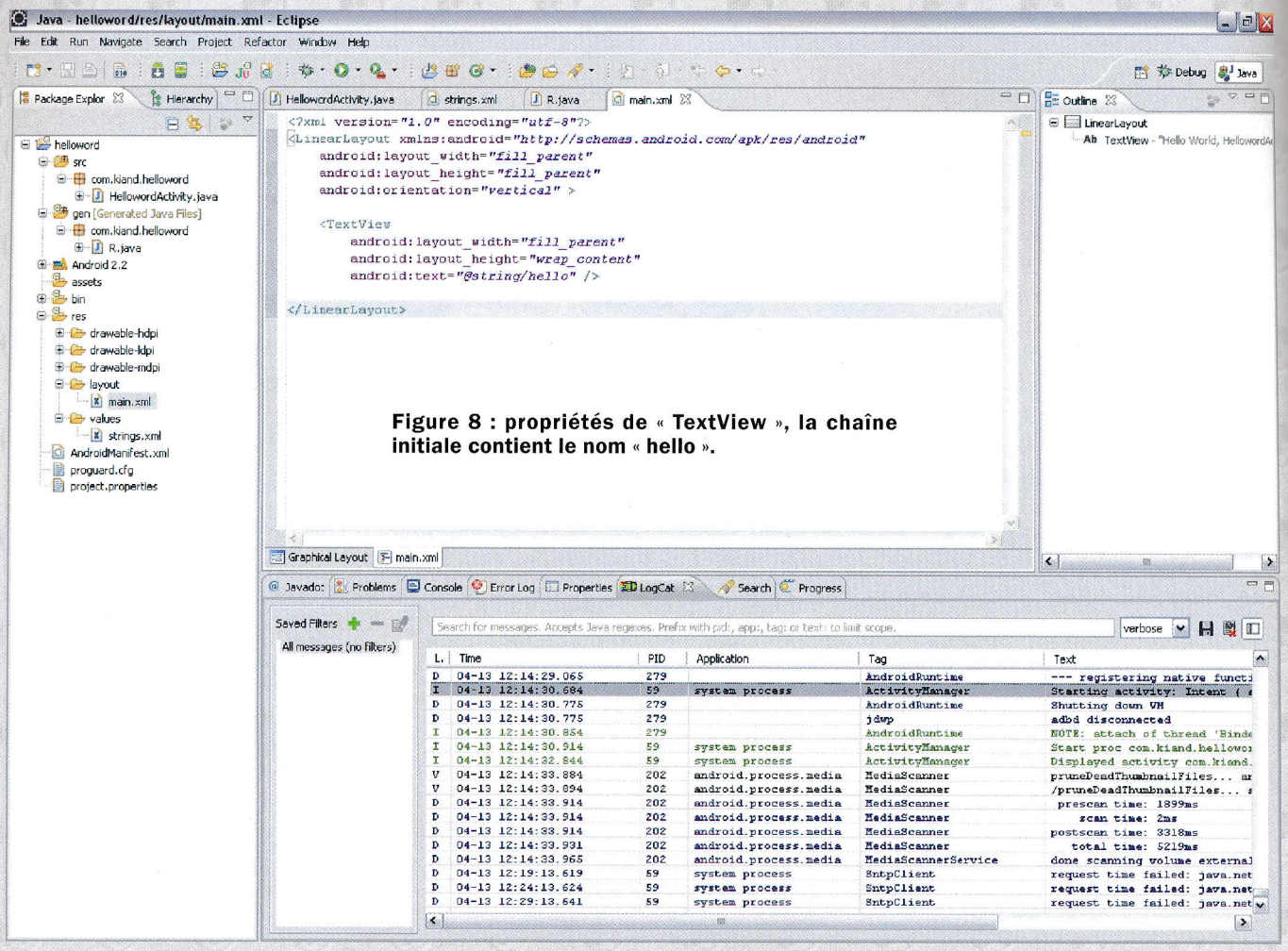



**Figure 8 : propriétés de « TextView », la chaîne initiale contient le nom « hello ».**

le téléchargement et le débogage de l'application. Pour cela, nous devons entrer dans le menu « **Paramètres** » Android de l'appareil, sélectionnons « **Applications** », puis développement et cocher « **Debug USB** » comme le montre la figure 9. Apparaissent alors les symboles sur la partie haute de l'écran de l'appareil Android indiquant qu'il est en mesure de communiquer via « **Eclipse** ». La prochaine fois que vous démarrerez « **Eclipse** », relancez l'application en utilisant « Run As » en tant que « Android Application », à droite de la console vous voyez le téléchargement qui prend quelques secondes et ensuite votre application sur le périphérique réel. Bien sûr, même à ce stade vous pouvez voir le journal des évènements dans la vue « **LogCat** ».

## Publication sur Google Play

Ce n'est pas le but principal de cet article car les applications que nous allons développer dans les prochains articles seront toujours combinées avec nos dispositifs à microcontrôleurs, cependant si vous voulez publier votre application sur **Google Play** (Android Market) nous vous donnons quelques explications brèves.

Les avantages d'utiliser Google Play sont plutôt nombreux, il contient 600 000 applications en tout, et 1,5 milliard d'applications sont téléchargées tous les mois. Il vous permet de mettre à disposition à un très grand nombre d'utilisateurs tous vos travaux, dans 190 pays, à moindre frais. En revanche, vous ne pourrez vendre vos applications que dans 132 pays, pour des raisons légales.

De plus, Google Play dispose d'outils pour vous permettre d'analyser le comportement des consommateurs, de traquer les bugs qui traînent dans votre application et de gagner de l'argent en récompense de votre labeur. La première chose à faire est d'avoir au moins un **compte Google valide**.

Vous pouvez en créer un à partir de cette page :

***https://accounts.google.com/SignUp***

Ce site est en français, **ensuite vous devez créer un compte développeur Android** à cette adresse :

***https://play.google.com/apps/publish/signup***

On vous demandera :

- de créer un compte développeur ;
- de signer virtuellement la charte de distribution des applications Android ; *(https://play.google.com/intl/ALL_fr/about/developer-distribution-agreement.html) ;*
- puis de payer la somme de 25$ (vous aurez besoin d'une carte de crédit valide).

Une fois votre compte créé, le premier écran qui apparaît est la console pour développeurs.

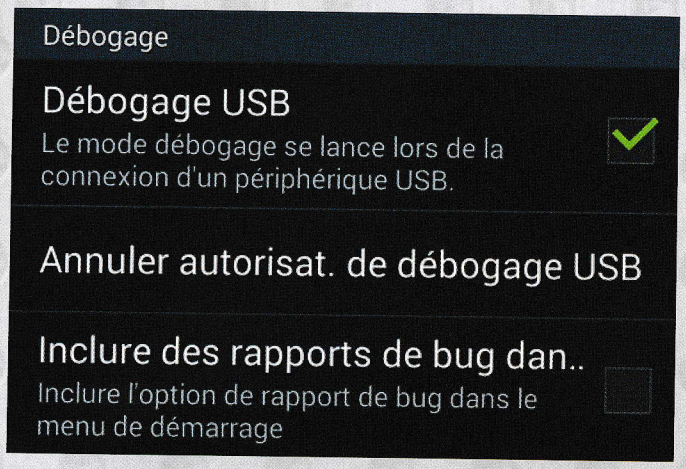



**Figure 9 : entrez dans le menu « Paramètres » Android de l'appareil, sélectionnez « Applications », puis « développement » et cocher « Debug USB ».**

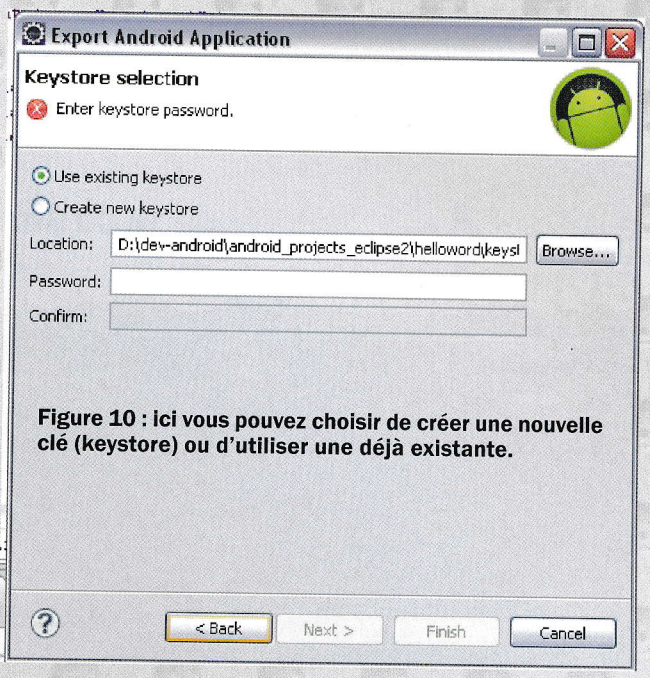



**Figure 10 : ici vous pouvez choisir de créer une nouvelle clé (keystore) ou d'utiliser une déjà existante.**

C'est dans cet écran que tout se fait, vous pouvez :

- ajouter un développeur avec qui vous travaillez en équipe. Vous pourrez déterminer les différentes parties auxquelles ont accès vos collègues. Tous les développeurs n'ont, par exemple, pas besoin de voir les revenus financiers de vos projets ;
- publier une application et avoir des informations dessus ;
- se constituer un compte Google marchand pour pouvoir vendre vos applications.

Enfin pour publier une application vous devez d'abord la signer. Pour qu'une application puisse être installée sous Android, elle doit obligatoirement être signée. Signer une application signifie lui attribuer un certificat qui permet au système de l'authentifier.

En fait « **Eclipse** » émet un certificat. Le problème est qu'il génère une clé de « **debug** » et que ce type de clé, n'étant pas défini par un humain, n'est pas digne de confiance et n'est pas assez sûre pour être utilisé de manière professionnelle pour publier vos projets sur un marché d'applications.

Si vous voulez publier votre application, il faudra générer une clé privée unique manuellement. Nous allons donc vous expliquer comme créer une clé pour signer vos applications.

Depuis « **Eclipse** », cliquez sur le nom du projet avec un clic droit, puis choisissez « **Android Tools** » (Outils Android) et ensuite « **Export Signed Application Package** » (Exporter un application signée).

Une fenêtre s'ouvre, comme indiquée sur la figure 10, dans laquelle vous pouvez choisir de **créer une nouvelle clé** (**keystore**) ou d'**utiliser une déjà existante**.

Le « **keystore** » est en fait un fichier qui contiendra un ou plusieurs de vos certificats. Pour cela, vous aurez besoin d'un mot de passe qui soit assez sûr pour le protéger. Une fois votre choix fait, cliquez sur « Next ».

Ensuite entrer les informations dans les 4 premiers champs :

- un alias qui permettra d'identifier votre clé ;
- un mot de passe et sa confirmation ;
- une limite de validité en années, sachant que la clé doit être disponible jusqu'en 2039 au moins. De manière générale, utilisez une valeur supérieure ou égale à 25.

Pour les champs suivants, il vous suffit d'en renseigner au moins un. Cliquez ensuite sur « **Next** », une fenêtre s'ouvre et choisissez l'emplacement où sera créé l'**APK** et terminez en cliquant sur « **Finish** ».

Une fois que vous avez terminé cette étape, vous aurez alors **votre application « .apk » signée**, vous pourrez ainsi la télécharger sur **Google Play** via votre compte Google.

Ensuite vous devez **effectuer au moins 2 captures** d'écran de votre application, **rédiger une courte description** et choisir une **icône à haute résolution** qui la représente. Vous pouvez consulter cette adresse :

***https://support.google.com/google-play/android-développeur***

à laquelle vous trouverez un ensemble d'explications pour se conformer avec le style d'Android.

Enfin, une fois le téléchargement effectué, vous trouverez après quelques heures votre application sur Google Play.

## Conclusion

Vous avez maintenant tous les éléments de base pour vous familiariser avec le monde Android et créer quelque chose de nouveau. Dans le prochain article, nous allons analyser le code d'une première application qui servira d'interface avec les dispositifs électroniques externes. Profitez de l'occasion pour examiner les contrôles graphiques d'Android et leur interaction avec l'utilisateur.

# 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''



**65,50 €**
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 Décembre / 2014 Photos non contractuelles, publicité valable pour les mois de parution. Prix exprimés en euros TTC. sauf erreurs typographiques ou omissions

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

# ABONNEZ-VOUS

**OUI,** Je m'abonne à ELECTRONIQUE ET LOISIRS magazine — LE MENSUEL DE L'ÉLECTRONIQUE POUR TOUS

**A PARTIR DU N°** 130 ou supérieur

N°

E0129

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**
Presstalis

**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 :

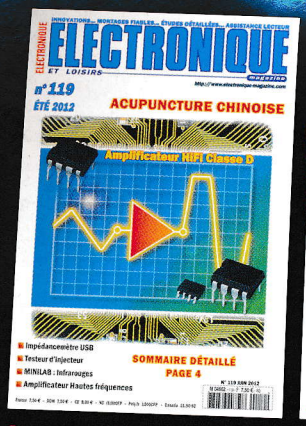
NUOVA ELETTRONICA
Elettronica In

**IMPORTANT**

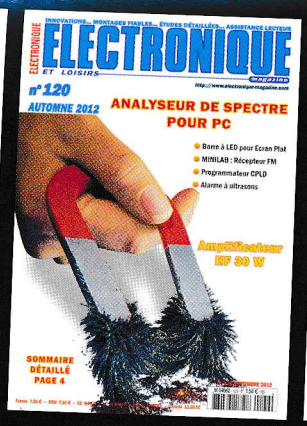
Reproduction, totale ou partielle, par tous moyens et sur tous supports, y compris l'internet, interdite sans accord écrit de l'Editeur. Toute utilisation des articles de ce magazine à des fins de notice ou à des fins commerciales est soumise à autorisation écrite de l'Editeur. Toute utilisation non autorisée fera l'objet de poursuites. Les opinions exprimées ainsi que les articles n'engagent que la responsabilité de leurs auteurs et ne reflètent pas obligatoirement l'opinion de la rédaction. L'Editeur décline toute responsabilité quant à la teneur des annonces de publicités insérées dans le magazine et des transactions qui en découlent. L'Editeur se réserve le droit de refuser les annonces et publicités sans avoir à justifier ce refus. Les noms, prénoms et adresses de nos abonnés ne sont communiqués qu'aux services internes de la société, ainsi qu'aux organismes liés contractuellement pour le routage. Les informations peuvent faire l'objet d'un droit d'accès et de rectification dans le cadre légal.

# 10,50 €* la revue frais de port inclus pour la France Métropolitaine

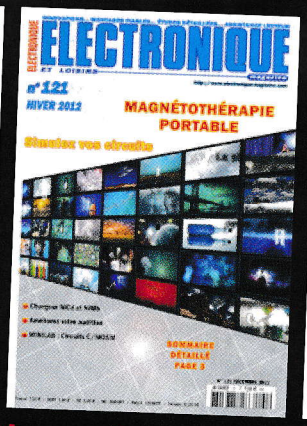
CEE, les DOM-TOM et autres pays calcul sur le site www.electroniquemagazine.com




**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...




**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** : Le Radon, ce tueur invisible, ce gaz qui plus que tout autre provoque le cancer comment le combattre et comment le mesurer ? pour cela nous avons besoin d'un appareil doté d'une chambre à ionisation très sensible - Tremblements de terre : causes et mesures - Contrôle de la distance de sécurité en voiture, un PIC, pour vous sauver la vie, calcule la distance de freinage de votre voiture et la distance de sécurité. Un appareil anti bégaiement - Un moniteur pour le Radon - Une alarme pour les séismes - Inducteur magnétique du sommeil pour éviter les insomnie - LTSpice pour apprendre à simuler vos circuits, un cours pour répondre à cette question.




**Au sommaire** : Localisate GSM/GPS, combine les techniques localisation sur le réseau cellulaire GS et le réseau satellite GPS - L'imprimant 3D qui existe depuis des décennies dan le monde professionnel, mais depu deux ans environ commence à fair partie intégrante du monde des amateur - Un lecteur MP3 Juke-Box sur clé usb 1 go - Comment utiliser une LED comm veilleuse et comme détecteur de lumière - Interrupteur crépusculaire miniature RaspberryPi un PC au format carte de crédit : Première partie - Amplificateu stéréo 2 x 10 W - Cours : Tutoriel EAGLE CAD V 6

* 14,10 € pour la revue 127 frais de port inclus pour la France Métropolitaine uniquement (autres voir le site ou nous consulter).

# CD-ROM ENTIÈREMENT IMPRIMABLE

## *51 € Les 3 CD du Cours d'Électronique en Partant de Zéro










**COURS**

**Niveau 1,2 ou 3**

***19,50 € l'unité**










**Numéros spéciaux l'unité *5,50 €**

*CD - FRAIS DE PORT INCLUS POUR LA FRANCE METROPOLITAINE (DOM-TOM ET AUTRES PAYS : NOUS CONSULTER).

JMJ EDITIONS - B.P. 20025 - 13720 LA BOUILLADISSE règlement par chèque à l'ordre de JMJ ÉDITIONS et/ou règlement par Paypal sur notre site : www.electroniquemagazine.com - Téléphone : 0820 820 534

Offre publicitaire valable pour les mois de parution. Prix exprimés en euros TTC. sauf erreurs typographiques ou omissions

# Articles, Revues et CD téléchargeables au format PDF sur Internet

## www.electroniquemagazine.com

**LISEZ ET IMPRIMEZ VOTRE REVUE SUR VOTRE ORDINATEUR PC OU MACINTOSH**




**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...




**Au sommaire** : La 3DRAG sans secrets, dans ce numéro nous allons aborder les logiciels et les aspects matériels de l'imprimante 3DRAG. Synthèse vocale pour RaspberryPi, transformez le RaspberryPi en un serveur web contrôlant à distance des entrées et des sorties - Capacimètre numérique de 10 pF à 10 000 µF, réalisez un capacimètre digital sans microcontrôleur et donc facile à reproduire - Chargeur d'accumulateur universel - MINI BUS, un bus pour l'automatisation : troisième partie. Alimentation symétrique de ± 1,25 V à ± 18 V / ± 1 A - Amplificateur pour MP3 & Smartphones Etc...




**Au sommaire** : Réalisez vos circuits imprimés avec la 3DRAG - Carillon électronique - Amplificateur Hi-Fi à MOSFET 2 x 200 W/4 Ω ou 350 W/8 Ω en pont - Protection pour amplificateur BF de puissance - Alarme pour voiture avec antidémarrage. Répétiteur de sonnerie pour téléphone. - Station météo avec RaspberryPi - Détecteur de gaz pour camping-cars et caravanes - Booster Classe H 2 X 70W pour voiture - Capacimètre numérique de 10 pF à 10 000 µF : 2ème partie - VU-mètre à LED - MINIBUS, un bus pour l'automatisation : 4 ème partie Etc...




**Au sommaire** : 3DRAG, passer d'un fichier image BMP à un fichier interprétable directement par Repetier-Host à l'aide du logiciel ArtCam - Carte d'extension A/D 16 bits pour RaspberryPi compatible Arduino - Alimentation de laboratoire contrôlée par un PIC16F876A, cette alimentation professionnelle de laboratoire est capable de fournir une tension réglable de 0 à 25 V et un courant de sortie réglable de 0 à 2,5 A - « Tracker » ou suiveur de Soleil motorisé pour panneaux solaires - Testeur de servomoteurs pour modélisme - Émetteur 4 canaux codé sur 9 digits - Timer programmable de 1 seconde à 60 heures - Programmez avec Android : Première partie, Etc...

# CD 6 Numéros 25 €* / CD 12 Numéros 45 €*

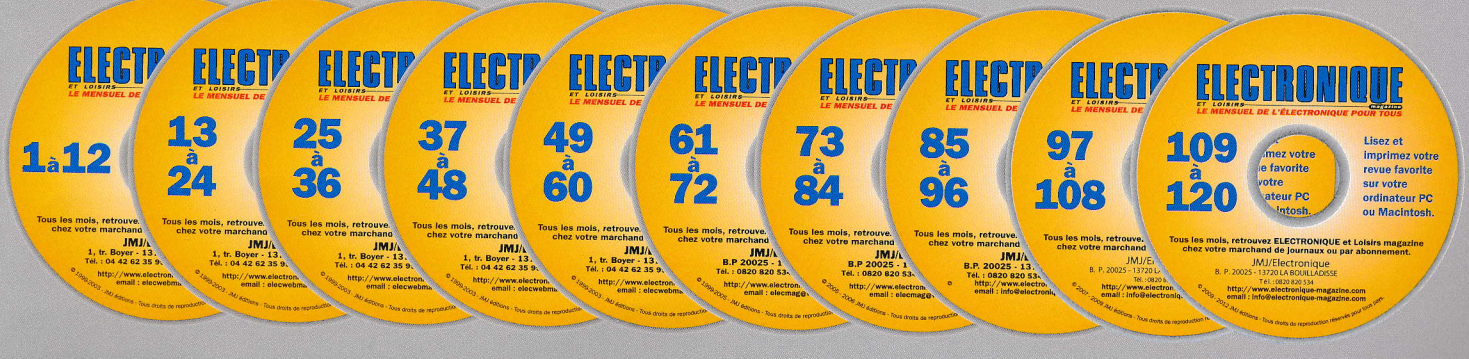
*CD - FRAIS DE PORT INCLUS POUR LA FRANCE METROPOLITAINE (DOM-TOM ET AUTRES PAYS : NOUS CONSULTER).







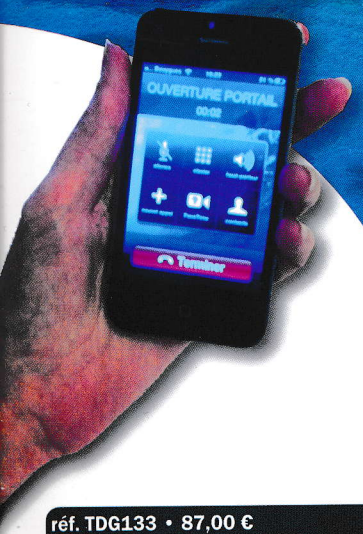
**50% de remise pour nos abonnés sur tous les CD de 6 ou 12 numéros**



# Contrôle à distance
# Utiliser son téléphone GSM
## comme télécommande et récepteur d'alarme

Comelec 12 / 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.**



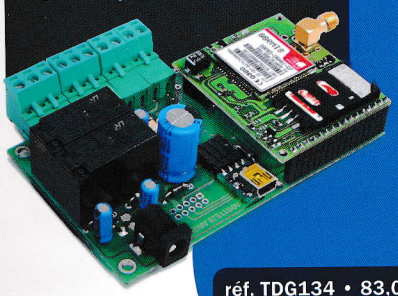

### SYSTÈME GSM INTELLIGENT

- 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.

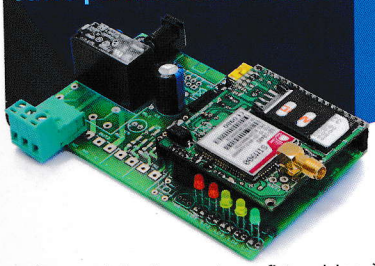
**réf. TDG135**
**115,50 €**

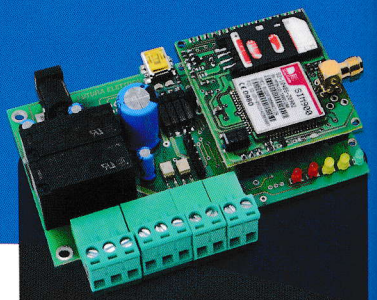


**réf. TDG133 • 87,00 €**
**TÉLÉCOMMANDE GSM**
**bidirectionnelle**
**2 entrées/2 sorties relais**



**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