ÉLECTRONIQUE ANALOGIQUE & NUMÉRIQUE • MICRO • AUDIO • TEST & MESURE

Septembre 2012 Nº 411 ISSN 0181-7450

elektor

LINUX INSIDE

LINUX EMBARQUÉ

DÉVELOPPEMENT LOGICIEL

+ AndroLED : pilotez votre électronique via WiFi sous Android

interface Nunchuk USB
recyclage de manette de jeux

+ isolateur USB
+ mettez le cap sur Arduino
+ interface pour train électrique

(B) 7,90 € • (CH) 13,20 FS • (CAN) 9,75 CAD • DOM surface 8,20 € • DOM avion 10,20 €
TOM surface 940 XPF • TOM avion 1620 XPF • Réu/A 9,70 € • Cal/S 950 CFP • Cal/A 1560 CFP • And 7,20 €

L 19624 -411- F: 7,20 €

elektor sur papier

le plaisir de la lecture classique

+

elektor à l'écran

le plaisir de la lecture sur les nouveaux supports : PC, portable ou tablette

=

elektor PLUS

le plaisir ultime de lire partout

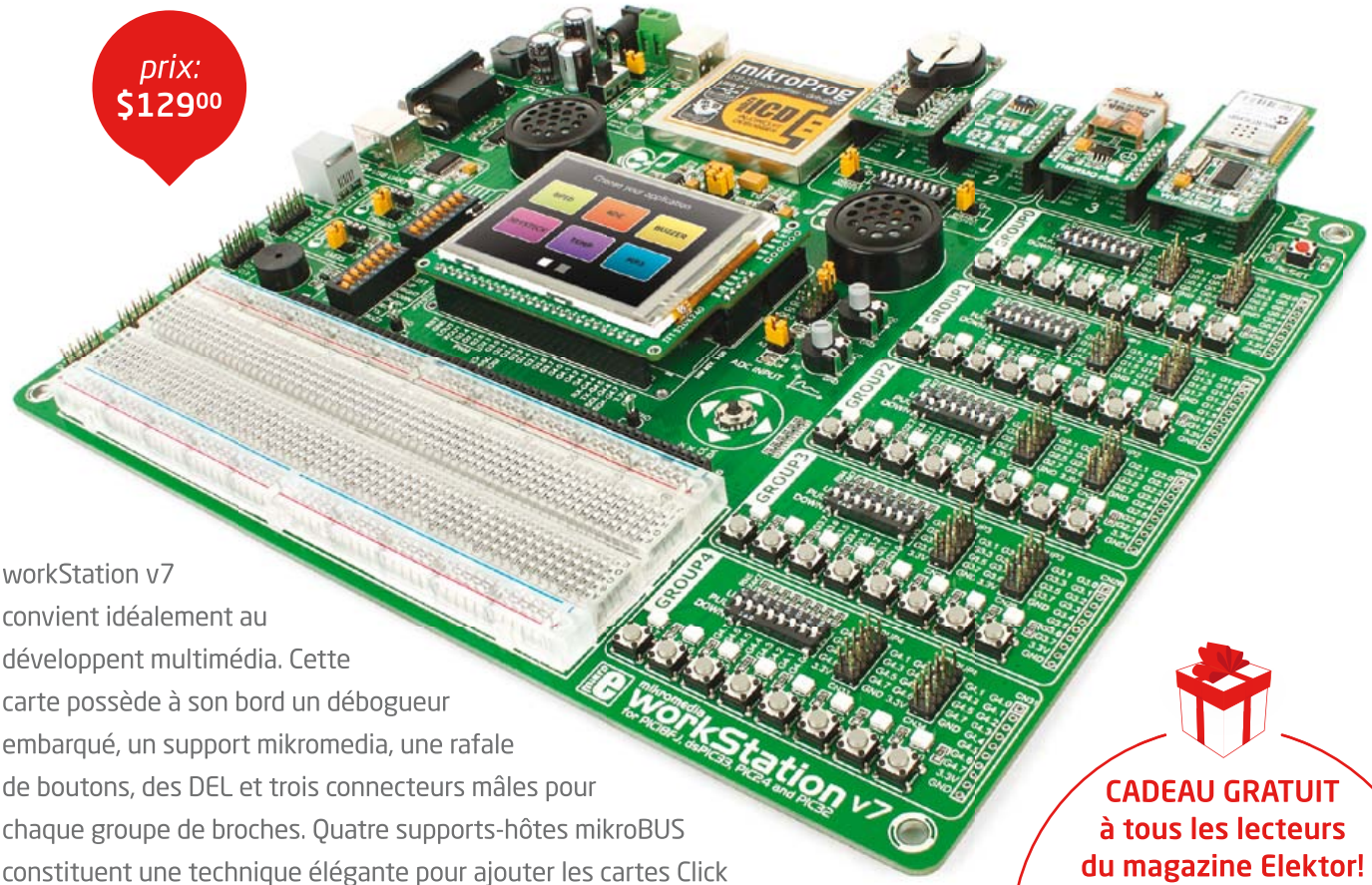
Profitez plus de votre magazine grâce à la formule avantageuse elektor PLUS !

Abonnez-vous ou changez de formule maintenant : www.elektor.fr/abo

mikromedia workStation v7

pour PIC18FJ®, dsPIC33®, PIC24® et PIC32®

prix: $129^{00}



workStation v7 convient idéalement au développent multimédia. Cette carte possède à son bord un débogueur embarqué, un support mikromedia, une rafale de boutons, des DEL et trois connecteurs mâles pour chaque groupe de broches. Quatre supports-hôtes mikroBUS constituent une technique élégante pour ajouter les cartes Click que l'on veut. Mettez à contribution la large carte de maquette d'essai pour assembler l'électronique de votre cru. Cette carte miracle est également fantastique lorsqu'il s'agit d'enseignement. Nous avons veillé tout particulièrement à fournir une riche palette de bibliothèques et d'exemples pour nos compilateurs mikroC, mikroBasic et mikroPascal aux fins de faciliter le développement.

CADEAU GRATUIT à tous les lecteurs du magazine Elektor!

ELEKTOR-09-2012

Entrez, à la fin de la page de Paiement, le code ci-dessus pour la GRATUITÉ du port!

*TOUTES COMMANDES INCLUSES

Code valide jusqu'au 14 septembre 2012.

PIC18FJ, dsPIC33, PIC24 ou PIC32?

workStation v7 supporte toutes les cartes mikromedia à microcontrôleurs Microchip®. Passez sans douleur à celui qu'il vous faut.

MikroElektronika
DEVELOPMENT TOOLS | COMPILERS | BOOKS

www.libstock.com

ALLEZ LE CHERCHER
www.mikroe.com

** La photo principale inclut : mikromedia workStation v7, mikromedia for PIC32, WiFi PLUS click, THERMO click, RTC2 click et SHT11 click.*

*** cartes mikromedia et click vendues séparément!*

Mobilier encombrant

Ce qui dans le mobilier d'Elektor impressionne le plus les visiteurs, ce sont les appareils de mesure du labo et, plus sûrement encore, la table en chêne massif de la salle de réunion. Sa taille surprend ceux qui ignorent que nos comités de rédaction réunissaient toujours au moins une quinzaine de personnes : d'abord une bonne demi-douzaine de représentants du labo (que ferions-nous sans nos rameurs ?), puis au moins autant de rédacteurs d'autant de nationalités différentes (ceux-là, toujours sur le pont, jacassent dans toutes les langues, ont un avis sur tout et le donnent même quand personne ne le leur demande), une secrétaire de rédaction (ô Vestale, précieuse ordonnatrice de nos débats, tu entretiens le feu du *Planning Continu*), un rédacteur en chef (il faut bien un capitaine mais on sait que les meilleurs sont ceux qui se savent remplaçables), un ou deux graphistes (ah, les illustrations et les schémas d'Elektor, comme ils sont bien dessinés), un ou deux stagiaires (il y en a de bons qu'on essaye de retenir) et vogue la galère. Même en l'absence de tout commercial (la rédaction tient à son indépendance et se crispe dès qu'il est question de notions aussi triviales que p. ex. le *bénéfice*), même sans représentant de la direction (à qui il faudrait expliquer chaque mot que de toute façon il ne comprendrait pas) ça faisait beaucoup de monde à la table ! Si je parle de cette situation à l'imparfait, c'est parce que nous sommes de plus en souvent en réunion non plus attablés dans la même pièce, mais devant nos tablettes aux quatre coins du monde, l'un dans un hôtel en Angleterre, l'autre dans son bureau en Inde, un autre encore aux Etats-Unis, certains autres à leur domicile et le reste autour de la grande table… à moitié inoccupée.
Après une période de valse-hésitation et de réticences narquoises, la tablette tactile numérique et sa *webcam* incorporée ont fini par s'imposer comme le lieu de réunion virtuel. Les conférences de rédaction sont devenues des téléconférences. Et ça marche. L'étape suivante de ce développement, c'est la virtualisation du labo , une opération à laquelle nous vous invitons à participer activement sur notre nouveau site elektor-projects.com —J'écris *notre* site, mais je pense ***VOTRE*** site.

Denis Meyer

SOMMAIRE

35e année
septembre 2012
nº 411

16 interface Nunchuck USB
recyclage de manette de jeux

La console de jeux Wii est livrée avec une deuxième manette qui contient un accéléromètre à trois axes, un stick et deux poussoirs. Il suffit d'un PIC pour communiquer avec cette interface homme machine et l'utiliser pour d'autres applications, p. ex. en robotique, en modélisme, pour le DMX etc.

28 interface pour train électrique
avec USB et éditeur de script

Ce train électrique qui dort dans un grenier, on en a rêvé, on nous l'avait offert ou on l'avait acheté pour les enfants, mais le plaisir s'est vite émoussé. Voici *le* circuit qui peut lui rendre vie : rallongez et agrémentez les voies, automatisez le fonctionnement. C'est plus simple et moins cher que vous croyez

34 isolateur USB
adieu parasites et boucles de terre

Si vous avez un appareil USB perturbé par des bruits dus à une boucle de masse ou si vous voulez protéger votre PC des tensions extérieures, utilisez un isolateur USB. Entre le PC et l'appareil connecté, il assure une séparation galvanique des lignes de bus et des lignes d'alimentation.

62 embarquez Linux
développement logiciel

L'éveil d'un µC vient d'une étincelle logicielle. Dans le monde de l'embarqué GNU/Linux, outre l'habituel micrologiciel, il faut aussi créer les composants du système d'exploitation, comme le montre cet article. Au passage nous saluerons le monde à l'aide du traditionnel *Hello World* !

Notre équipe

Rédacteur en chef : **Denis Meyer** (redaction@elektor.fr) Directeur éditorial : **Wisse Hettinga**

Rédaction internationale : **Harry Baggen, Thijs Beckers, Eduardo Corral, Jens Nickel, Clemens Valens**

Laboratoire : **Thijs Beckers, Ton Giesberts, Luc Lemmens, Raymond Vermeulen, Jan Visser**

Graphistes : **Giel Dols, Jeanine Opreij, Mart Schroijen** Secrétaire de rédaction : **Hedwig Hennekens**

Directeur *Elektor online* : **Daniëlle Mertens**

Ont coopéré à ce numéro : **Jean-Paul Brodier, Robert Grignard, Michel Kueneman, Hervé Moreau, Kévin Petit, NN**

Nos réseaux

audioXPRESS

elektor

CIRCUIT CELLAR
THE WORLD'S SOURCE FOR EMBEDDED ELECTRONICS ENGINEERING INFORMATION

techthefuture
Tech the Future explores the solutions for a sustainable future provided by technology, creativity and science.

VOICE COIL

Nos équipes internationales

United Kingdom
Wisse Hettinga
+31 (0)46 4389428
w.hettinga@elektor.com

USA
Hugo Vanhaecke
+1 860-875-2199
h.vanhaecke@elektor.com

Germany
Ferdinand te Walvaart
+31 46 4389417
f.tewalvaart@elektor.de

France
Denis Meyer
+31 46 4389435
d.meyer@elektor.fr

Netherlands
Harry Baggen
+31 46 4389429
h.baggen@elektor.nl

Spain
Eduardo Corral
+34 91 101 93 95
e.corral@elektor.es

Italy
Maurizio del Corso
+39 2.66504755
m.delcorso@inware.it

Sweden
Wisse Hettinga
+31 46 4389428
w.hettinga@elektor.com

Brazil
João Martins
+55 11 4195 0363
joao.martins@editorialbolina.com

Portugal
João Martins
+351 21413-1600
joao.martins@editorialbolina.com

India
Sunil D. Malekar
+91 9833168815
ts@elektor.in

Russia
Nataliya Melnikova
+7 (965) 395 33 36
Elektor.Russia@gmail.com

Turkey
Zeynep Köksal
+90 532 277 48 26
zkoksal@beti.com.tr

South Africa
Johan Dijk
+27 78 2330 694 / +31 6 109 31 926
j.dijk @elektor.com

China
Cees Baay
+86 21 6445 2811
CeesBaay@gmail.com

35ème année, n°411 septembre 2012
ISSN 0181-7450 Dépôt légal : août 2012
CPPAP 1113 U 83713

ELEKTOR / PUBLITRONIC SARL
c/o Regus Roissy CDG - 1, rue de la Haye - BP 12910
FR - 95731 Roissy CDG Cedex - France
Tél. : (+33) 01.49.19.26.19 - Fax : (+33) 01.49.19.22.37
www.elektor.fr

Banque ABN AMRO : Paris
IBAN : FR76 1873 9000 0100 2007 9702 603
BIC : ABNAFRPP

Nos adhérents

Nous avons **274920** adhérents dans **83** pays

Adhérez vous aussi au réseau Elektor...

ElektorHebdo en ligne

notre lettre hebdomadaire d'information électronique gratuite et à ...

Elektor, le mensuel d'électronique

Tarifs, conditions, offres, nouveautés :

www.elektor.fr/abo

Vos correspondants

Nous sommes à votre service pour
toute question relative à votre commande ou votre abonnement
par téléphone (+33) 01.49.19.26.19 ou par courriel : service@elektor.fr

Nos annonceurs

Pour placer votre annonce dans le prochain numéro d'Elektor

veuillez contacter **Mme Ilham Mohammedi** par téléphone au (+33) 01.49.19.26.19

ou par courrier électronique : i.mohammedi@elektor.fr

Elektor est édité par Elektor International Media B.V.
Siège social : Allee 1 - 6141 AV Limbricht, Pays-Bas

Imprimé aux Pays-Bas par Senefelder Misset – Doetinchem

Distribué en France par M.L.P. et en Belgique par A.M.P.

11 nouvelles pinces multimètres Chauvin Arnoux TRMS inégalées !

1000 V CAT IV, un niveau de sécurité sans précédent pour des pinces multimètres !

Sécurité assurée, les nouvelles pinces respectent les contraintes de la norme IEC 61010.

Exceptionnel, l'indice de protection IP54 protège l'appareil contre l'eau et les poussières.

La conception mécanique de ces pinces, dotées d'une ceinture antichoc, leur permet de passer le test normatif de chute pour une hauteur de 2 m.

Sécurité encore, sur tous les modèles, la détection automatique AC/DC du type de signal est disponible en intensité, tension et puissances.

Prise en main et maniabilité, y compris avec des gants de protection : d'une seule main, l'utilisateur peut connecter sa pince, sélectionner la mesure et ses fonctions. Pour une efficacité maximale, chaque mesure correspond à une position du commutateur.

Selon les modèles, ces pinces sont dotées de grands afficheurs LCD rétro éclairés, de 6000 à 10000 points.

Le principe « 1 touche égale 1 fonction » renforce la simplicité.

Différents diamètres d'enserrage sont disponibles, jusqu'à 60 mm, afin de couvrir un maximum de besoins dans les meilleures conditions.

La gamme comporte 3 séries avec chacune 3 à 4 modèles dotées de spécificités différentes pour les professionnels du marché de l'électricité.

- F200 pour des mesures jusqu'à 600 AAC et 900 ADC,
- F400 jusqu'à 1000 AAC et 1500 ADC
- F600 jusqu'à 2000 AAC et 3000 ADC

Ensuite les modèles dépendent de l'utilisation :

- F201, F401 et F601 pour applications courant alternatif
- F203, F403 et F603 pour applications courant alternatif ou continu
- F205, F405 et F605 pour applications mixtes AC+DC, analyse ou expertise
- F407 et F607 pour applications mixtes AC+DC, analyse ou expertise et en plus l'analyse harmonique jusqu'au rang 25 et l'enregistrement.

Le système d'acquisition numérique TRMS 12 bits rapide offre une grande qualité de mesure. Leur large bande passante et leur facteur de crête élevé viennent améliorer la justesse et la précision des mesures, quelle que soit la nature du signal.

Mesures complémentaires :

Les pinces F205, F405 et F605 possèdent la rotation de phases.

Les modèles F407 et F607 réalisent également les mesures de puissances monophasées et totales triphasées (actives, réactives et apparentes, et le PF et DPF) ainsi que l'analyse d'harmoniques, THDf et THDr.

Nouveauté Chauvin Arnoux, la fonction *TrueInrush* permet de mesurer le démarrage moteur, d'une machine seule ou d'un parc machines en fonctionnement.

La fonction ΔRel est applicable sur toutes les mesures et aussi sur les fonctions d'analyses (Min, Max, Peak- et Peak+). Elle permet une comparaison instantanée de la mesure par rapport à une référence.

Des fonctions performantes selon les modèles :

- Mesures TRMS des Min et Max calculées sur 100 ms
- Peak+ et Peak- sur 1 ms et sur chaque fonction
- HOLD étendu, pour la mémorisation de l'état complet des mesures et fonctions en cours
- Mesure du THD selon l'IEC 61000-4-7
- Enregistrement jusqu'à 1000 mesures sur plusieurs campagnes
- Communication Bluetooth pour l'exportation des données

Applications

De la production à la consommation d'électricité, les nouvelles pinces Séries F couvrent toutes les applications électriques Basse Tension.

Les pinces Série F200 conviennent aux applications BT pour les petites et moyennes puissances telles que maintenance d'installations électriques tertiaires ou industrielles, de parc machine, diagnostic et/ou dimensionnement de l'alimentation électrique, mise en route de climatisation & chauffage, intervention de véhicules électriques...

Basse Tension moyennes puissances, la Série F400 s'utilise dans les secteurs de la production et de la distribution d'électricité BT, les industries, les réseaux ferrés... Il convient également aux ascensoristes ou autres spécialistes des équipements de evage et de transport. Maintenance, contrôle, surveillance, diagnostic ou raccordement sont les principales applications des pinces de cette série.

La Série F600 est conçue pour le marché de la BT fortes puissances, tels que :

- la distribution d'énergie électrique HT/BT,
- les industries chimiques ou pétrochimiques,
- la métallurgie, les transports...
- maintenance, contrôle, surveillance, diagnostics, dimensionnement, raccordement...

Les modules à LED Zenigata sous les feux de la rampe !

Après avoir récemment enrichi sa gamme avec des modèles jusqu'à 50 W, Sharp propose plus de 100 modules à LED couvrant tous les flux lumineux entre 5 et 7000 lumens, avec un large spectre de températures de couleur et des indices de rendu des couleurs (IRC) élevés.

50 W, un IRC (indice de rendu des couleurs) jusqu'à 93, un flux lumineux pouvant atteindre 7000 lumens : tels sont les paramètres clés des modules à LED de la série Mega Zenigata.
En fonction du flux lumineux désiré, les modules regroupent un nombre plus ou moins élevé de puces LED. De plus, ces modules se déclinent selon des valeurs de température de couleur de 2700 K à 6500 K pour couvrir de multiples teintes de blanc allant du « blanc chaud » au « blanc froid » en passant par le « blanc neutre » et le « blanc pur ».
Pour les spots lumineux et les lampes à LED, Sharp propose sa gamme de modules Mega Zenigata qui délivre des flux lumineux de 1150 à 7000 lumens selon les versions, ainsi que la série Mini Zenigata qui se situe dans la fourchette 300 à 1550 lumens. À partir des modules Mega Zenigata de Sharp, il est possible d'assembler des lampes à LED en y associant des réflecteurs, des lentilles et des formes de boîtiers identiques compte tenu de leur compatibilité mécanique et optique. Mieux adaptées aux sources d'éclairage de forme surfacique, les séries SAE et Pico Zenigata présentent des flux lumineux de 5 à 75 lumens.
La durée de vie peut aller jusqu'à 40 000 h à une température de fonctionnement de 90 °C. Ces LED présentent, en valeurs typiques, des IRC d'au moins 80 pour toutes les températures de couleur proposées. Les modules de la série Mega Zenigata sont même disponibles pour des valeurs typiques d'IRC jusqu'à 93. Ils sont ainsi compatibles avec le label Energy Star et avec la plupart des autres standards industriels.

24 bits et une précision de 0,1 % pour des capteurs de pression numériques

Sensortechnics propose pour des applications de mesure de pression pointues, par

exemple dans le domaine médical ou de l'instrumentation, des capteurs de pression piézorésistifs personnalisés, à partir d'une pleine échelle de mesure de 2,5 mbar, avec une résolution et une précision très élevées. Grâce à un amplificateur à faible bruit, suivi d'un convertisseur A/N sur 24 bits, ces capteurs présentent des signaux de sortie numériques à haute résolution, comparables à ceux de capteurs analogiques, avec un rapport signal/bruit très élevé. Les capteurs utilisent un µC et des algorithmes de correction offrant une grande souplesse de programmation pour compenser la température et linéariser le signal de sortie du capteur. La bande d'erreur totale est réduite à 0,1 %, les temps de réponse à 250 µs.
Pour ses capteurs de pression avec µC, Sensortechnics offre de multiples possibilités d'étalonnage, de traitement du signal et de transmission du signal pour répondre aux besoins spécifiques. Sur la plage totale de mesure du capteur, il est possible d'étalonner des plages partielles avec des données différentes de résolution et de précision. Cette fonction est intéressante pour les appareils médicaux de ventilation qui détectent le flux respiratoire par une mesure différentielle et ont besoin d'une sensibilité supérieure dans les faibles pressions afin de détecter de manière précise et rapide les flux même les plus faibles. Par ailleurs, Sensortechnics propose une adaptation spécifique de l'interface numérique (I^2C, SPI) du capteur et peut programmer des protocoles de transmission et des fonctions additionnelles comme les demandes d'état, les messages de diagnostic et une fonction d'auto-zéro.

Principales propriétés des capteurs de pression avec microcontrôleur :
- Plages de pression à partir de 2,5 mbar de pleine échelle
- Amplificateur à faible bruit et convertisseur analogique-numérique sur 24 bits
- Précision totale typiquement de 0,1 %
- Personnalisation possible de l'interface numérique et du protocole de transmission de données

Circuit de jauge de batteries multicellulaires adapté à différents types de batteries au lithium

Pour exploiter intégralement les ressources énergétiques d'une batterie multicellulaire, il faut une mesure fine de la capacité restante,

qui tienne compte de la tension de charge, des caractéristiques et des propriétés de la batterie. *Texas Instruments* présente le premier circuit dans sa gamme de circuits de gestion de l'état de batteries multicellulaires adaptés à différents types de batteries et dotés de la technologie de mesure de capacité de la batterie *Impedance Track*™.

Le **bq34z100** prend en charge un large éventail de batteries au lithium-ion et lithium-ferphosphate en packs de 2 à 16 cellules, ce qui permet d'allonger l'autonomie des batteries, par exemple dans l'électronique médicale, les outils électriques, les vélos électriques et les alimentations sans interruption (ASI). De nouveaux systèmes de mesure prendront en charge les batteries au plomb, au nickel cadmium et au nickel-métal hydrure, et offriront une exactitude de l'ordre de 94 % sur toute la durée de vie de la batterie.
Ce circuit fonctionne indépendamment des configurations de cellules en série et peut réduire la consommation par le biais d'un circuit de translation de tension externe.

(120175-2)

Horloge numérique FINDER à programmation journalière ou hebdomadaire

FINDER France propose sa nouvelle horloge numérique série 1251, avec écran LCD rétro éclairé, pour automatiser des installations électriques pour la commande programmée d'éclairage, d'enseignes lumineuses, d'arrosage automatique ou de contrôle d'accès (ouverture/fermeture de portail, etc.) à n'importe quelle heure. Grâce à une programmation très simple par manche de commande en façade (plus de boutons !), l'opérateur choisit les jours et les horaires auxquels le contact doit s'ouvrir et se fermer.

Exemples d'applications :

- Allumer l'enseigne lumineuse d'une grande surface de 20h00 à 00h00 du lundi au samedi
- Arroser automatiquement la pelouse d'un stade de foot 3 jours par semaine pendant 30 min (lundi, mercredi, vendredi de 07h00 à 07h30)

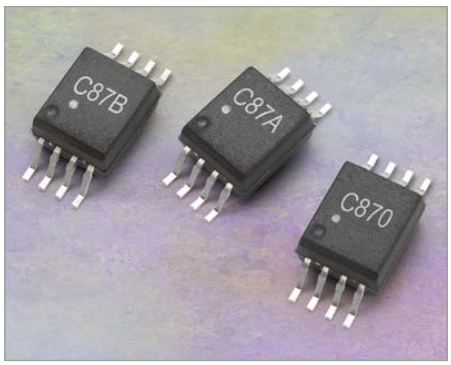
Caractéristiques techniques :

- Contact : 1 inverseur 16A
- Alimentation en 230 V AC
- Programmation journalière ou hebdomadaire, intervalle de programmation : 30 min
- Pile interne pour réglage et programmation sans alimentation, remplacement facile en façade de l'appareil
- Réserve de marche : 6 ans
- Écran LCD rétro éclairé
- Montage sur rail DIN 35 mm
- IP 20
- Plage d'utilisation : -20 à +50 °C
- Contact sans Cadmium.

(120175-3)

Amplificateurs d'isolation optique optimisés pour la mesure de tension

Avago Technologies, l'un des principaux fabricants de composants pour applications industrielles, automobiles et d'énergies renouvelables propose une nouvelle famille de capteurs de tension avec isolation optique, la première série d'amplificateurs d'isolation optique du secteur fondée sur la modulation sigma-delta et optimisée pour la mesure de tension. Conçus pour des applications comme la mesure des tensions dans les systèmes électroniques de conversion que l'on trouve dans les commandes de moteurs, dans les alimentations électriques à découpage et dans les systèmes d'énergies renouvelables ainsi que dans les interfaces de capteur comme l'isolation



par thermistances CTN dans les modules IGBT (transistor bipolaire à grille isolée), les modèles ACPL-C87B, C87A, et C870 offrent aux concepteurs une plage d'entrée de 2 V et une impédance très élevée de 1 GΩ.

L'ACPL-C87A (tolérance de gain de ±1 %) et l'ACPL-C870 (tolérance de gain de ±3 %) sont recommandés pour les applications plus générales. L'ACPL-C87B (tolérance de gain de ±0,5 %) peut être utilisé pour les applications de haute précision comme les onduleurs des systèmes d'énergies renouvelables où il est important de bien mesurer la tension de bus CC fluctuante (en raison des conditions d'ensoleillement changeantes) et pour les entraînements servo et moteur. Les capteurs de tension possèdent un gain d'unité 1 V/V avec une faible dérive de gain de –35 ppm/°C et la nonlinéarité reste inférieure à 0,1 %. Les capteurs de tension disposent d'une broche d'arrêt active à l'état haut qui réduit le courant I_{DD1} à seulement 15 µA en mode de veille, si bien que les capteurs sont particulièrement adaptés aux applications alimentées par batterie ou disposant d'une alimentation délicate.

(120175-4)

Retour aux sources (7)
clignotants et oscillateurs

Après les bascules statiques et les triggers de Schmitt du dernier épisode, nos circuits s'animent : les condensateurs entrent en jeu, la rétroaction donne le rythme, les LED clignotent de plus en plus vite, et les générateurs de signaux jouent leurs fonds sonores.
Eh oui, l'électronique c'est encore plus cool quand ça clignote et ça bipe !

Burkhard Kainka (Allemagne)

Un circuit bistable, qui, de lui-même et sans excitation extérieure, change périodiquement d'état, est appelé multivibrateur (astable). La **figure 1** montre un tel multivibrateur. Aucun doute ici, ce sont les condensateurs qui sont responsables de la rétroaction. Avec des condensateurs électrolytiques, il est impératif de veiller à la polarité, car le potentiel d'un collecteur est en moyenne supérieur au potentiel de la base du transistor opposé. L'état du circuit est stable tant que dure la montée de la tension aux bornes des condensateurs. Ensuite le circuit bascule et le processus reprend. Plus la capacité du condensateur est forte, moins il se charge rapidement, plus la période de stabilité de la bascule est longue.
Avec deux électrolytiques de 10 µF, la fréquence de clignotement des LED est faible, la période dure environ 1 s. Changer la valeur des condensateurs permet de faire varier la fréquence de commutation du multivibrateur dans un large intervalle. Des condensateurs plus petits ou différents l'un de l'autre donnent également des résultats intéressants. Avec des valeurs de 100 µF et 100 nF, vous n'observerez que de brefs éclairs sur l'une des deux LED. Deux condensateurs de 100 nF donnent un clignotement rapide et symétrique.

Multivibrateur simplifié

Le circuit a par principe besoin de deux transistors montés en émetteur commun (ils agissent comme inverseurs et déphasent chacun le signal d'entrée de 180 degrés). Un couplage direct des étages permet de se passer d'un des condensateurs, et donc de simplifier le multivibrateur (**fig. 2**).
En l'absence de rétroaction, l'amorçage infaillible des oscillations nécessite un point de fonctionnement (ou de basculement) moyen. Si le point de basculement est placé trop loin, le transistor de sortie est soit entièrement bloqué, soit complètement saturé quand ce point est atteint. Le pouvoir d'amplification du circuit serait alors insuffisant pour engendrer l'oscillation. Le point de fonctionnement moyen est obtenu ici grâce à la forte contre-réaction appliquée sur le premier transistor (c.-à-d. grâce à la résistance de 10 kΩ insérée entre collecteur et base). La rétroaction par circuit RC prime toutefois sur la contre-réaction, et au final le transistor de sortie passe tour à tour de l'état bloqué à saturé.

Montons le circuit sans nous préoccuper dans un premier temps du condensateur de rétroaction, c'est plus facile à comprendre. La LED devrait s'allumer faiblement, puisque le transistor de sortie n'est pas complètement saturé. Le condensateur une fois inséré, la LED se met à clignoter, environ une fois par seconde avec le condensateur de 22 µF. Le circuit fonctionne également avec des condensateurs aussi petits que 10 nF, mais le clignotement est alors si rapide que la LED semble ne pas s'éteindre. Si vous connectez un transducteur acoustique, vous entendrez un crépitement qui confirme l'oscillation.

LED convertisseuse de tension

Une LED rouge a besoin de 1,5 à 2 V, une bleue ou blanche de 3 à 4 V. Pour obtenir cette tension, on recourt en général à 3 piles, qui ensemble fournissent 4,5 V, l'excès de tension étant éliminé dans une résistance chutrice. Ne pourrait-on pas, au lieu de ce gaspillage, utiliser une seule pile de 1,5 V avec un convertisseur de tension ?

L'élément clé est une petite inductance fixe de 1,5 millihenry. Sous l'enrobage de ce composant, qui ressemble à une résistance, il y a une bobine à noyau de ferrite. On peut construire soi-même ce bobinage en enroulant 200 spires autour d'un barreau de ferrite.
Le circuit de la **figure 3** montre un autre multivibrateur simple. Le courant qui circule à travers la bobine est interrompu et relancé sur un rythme rapide. La bobine

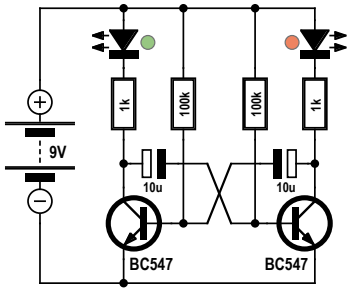


Figure 1. Le multivibrateur

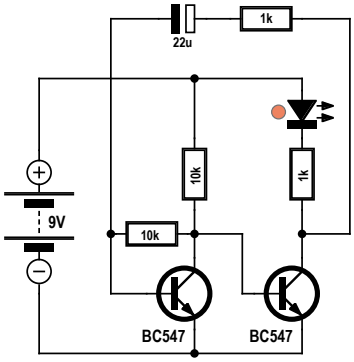


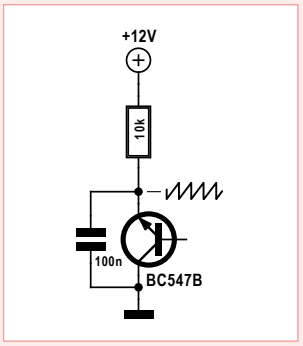
Figure 2. Clignotant simplifié.

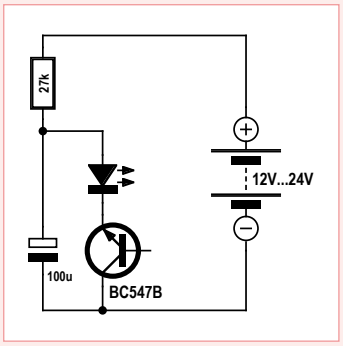
Oscillateur à relaxation avec NPN

Un oscillateur à relaxation fonctionne aussi avec un seul transistor, exploité ici d'une façon inhabituelle. Le transistor NPN est en effet monté « à l'envers », son émetteur étant relié au positif et sa base laissée en l'air. Une mesure de la tension aux bornes du condensateur montre qu'elle s'élève chaque fois jusqu'à 9 V. Le transistor passe ensuite à l'état saturé de façon abrupte, puis décharge le condensateur jusqu'à 7 V. La fiche technique du transistor reste muette sur ce point. Comme chaque transistor a un comportement qui lui est propre, il est intéressant d'en tester plusieurs.

L'impulsion formée par ce courant de décharge est suffisamment forte pour faire fonctionner une LED. La tension d'alimentation vaut ici plus de 12 V. Le circuit marche très bien avec deux piles de 9 V presque déchargées. La LED clignote longtemps, tirant jusqu'aux derniers électrons des piles. La fréquence de clignotement décroît à ce moment-là.

Avec cette polarisation inverse du transistor, la pente de sa caractéristique V_{CE} est négative, ce qui est facilement vérifiable. La diode BE subit un effet d'avalanche vers 9 V : le champ électrique élevé présent dans la mince jonction accélère suffisamment les porteurs de charge libres pour qu'ils libèrent à leur tour d'autres porteurs. Leur nombre que fait que croître, et avec lui le courant. C'est le même effet que celui qui apparaît dans une zener de 9 V, la résistance interne de cette diode étant toutefois positive.





Vient maintenant le transistor à polarisation inversée. Les rôles de l'émetteur et du collecteur sont intervertis, mais en raison de sa symétrie intrinsèque ce transistor fonctionne à peu près de la même façon. Sa fonction est d'amener les porteurs de charge jusqu'à la jonction via la mince base, jonction dans laquelle apparaît ici aussi l'effet d'avalanche. D'où encore plus de porteurs, qui libèrent d'autres porteurs du réseau cristallin, et au final un effet d'avalanche qui se renforce de lui-même.

fonctionne alors comme un réservoir d'énergie magnétique. Chaque interruption du courant induit une tension qui s'ajoute à celle de la pile. La hauteur de cette tension s'adapte d'elle-même au type de récepteur connecté : une LED blanche verra par exemple une tension plus forte qu'une LED rouge. De tels convertisseurs sont la plupart du temps accompagnés d'un redresseur et d'un condensateur électrolytique de lissage. On peut s'en passer ici, car la LED est son propre redresseur. Il circule à travers elle un courant continu pulsé, dont l'intensité est en moyenne un peu plus faible que celle délivrée par la pile, car la tension est plus haute. En fin de compte, le circuit possède un meilleur rendement que l'habituelle solution consistant en une résistance pour abaisser une tension de pile plus forte.

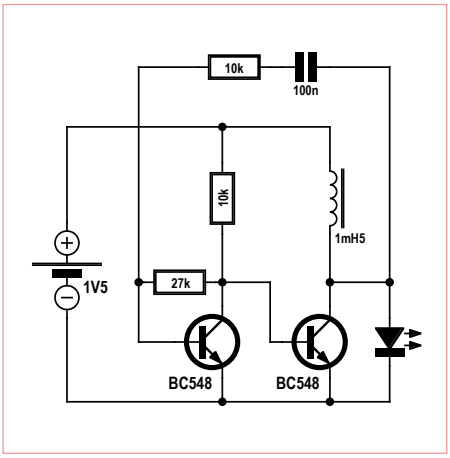


Figure 3. Une LED convertisseuse de tension.

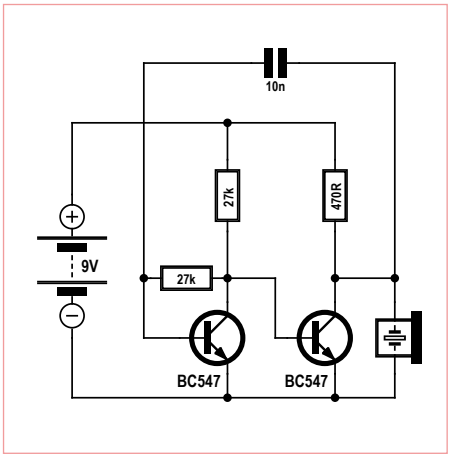


Figure 4. Attaque d'un transducteur acoustique.

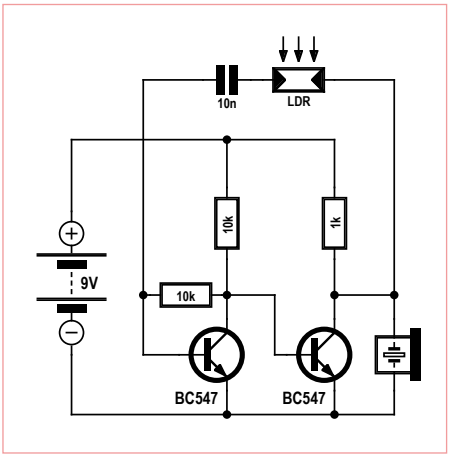


Figure 5. Un générateur d'audiofréquences réglable.

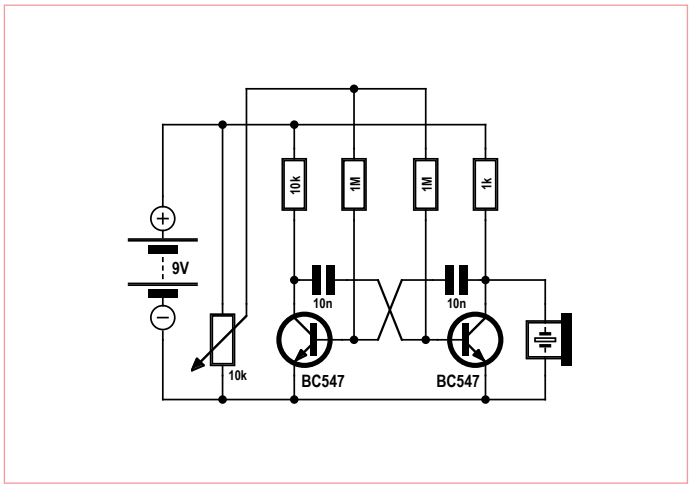


Figure 6. Un multivibrateur servant de VCO.

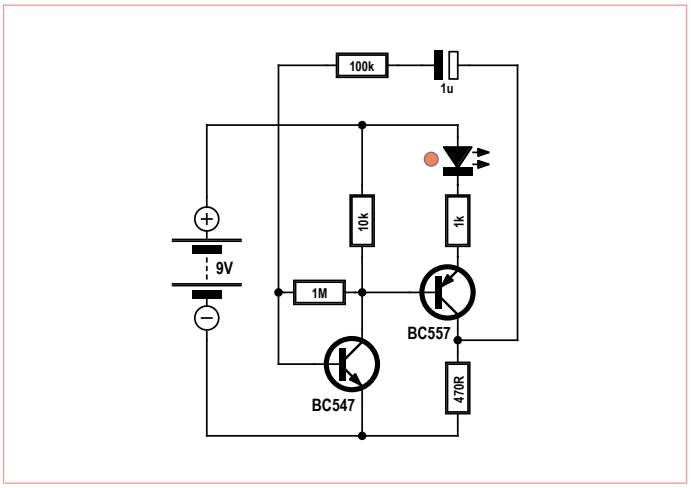


Figure 7. Un clignotant à base de NPN et PNP.

Générateur d'audiofréquences

Équipé d'un petit condensateur, le multivibrateur simple produit une fréquence plus élevée, qui va jusqu'à gagner le domaine des fréquences sonores. Un transducteur piézo-électrique (**fig. 4**) connecté au circuit produira donc des sons audibles. Le transducteur, lui-même une sorte de condensateur, influe sur la fréquence sonore. Petite expérience en passant : touchez la membrane du transducteur avec un doigt ou un objet dur. L'intensité, mais aussi la hauteur du son, sont affectées. L'élément piézo-électrique oscillant produit de son côté une tension alternative qui a un effet sur le générateur. En principe, même une onde sonore réfléchie par la membrane du transducteur pourrait elle aussi exercer une influence sur l'oscillateur, mais dans une moindre mesure.

Comme la valeur de la résistance placée dans la branche de réaction influence elle aussi la fréquence produite, nous pourrions exploiter cet effet avec un potentiomètre. Avec une photorésistance (LDR), la hauteur du son serait déterminée par l'intensité de la lumière incidente.

Le circuit de la **figure 5** permet de différencier à la fois l'intensité et la nature de la lumière. Une lumière artificielle, qui clignote rapidement, module ainsi la fréquence du signal sonore. Ainsi la lumière d'un tube fluorescent provoque un bourdonnement caractéristique. La lumière de l'écran d'un PC modifie également le son, modulé par la fréquence de rafraîchissement de l'image.

Convertisseur tension-fréquence

La **figure 6** montre comment réaliser un oscillateur commandé en tension (VCO, *voltage controlled oscillator*) à partir d'un

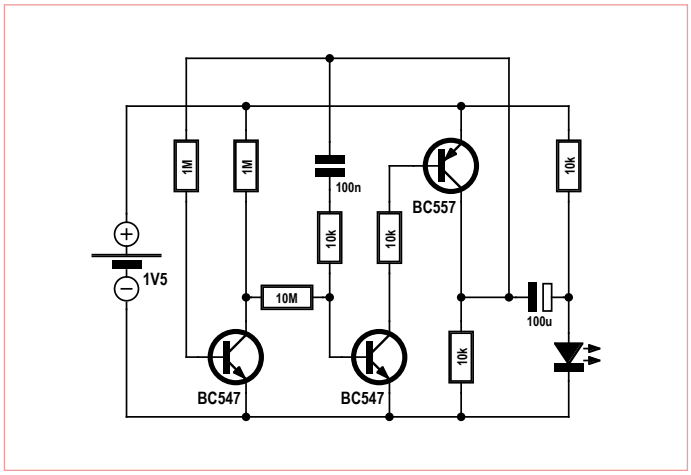


Figure 8. Flash à LED économique.

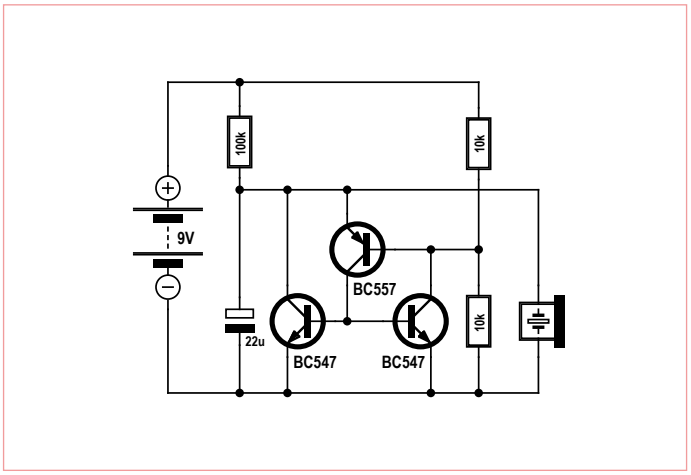
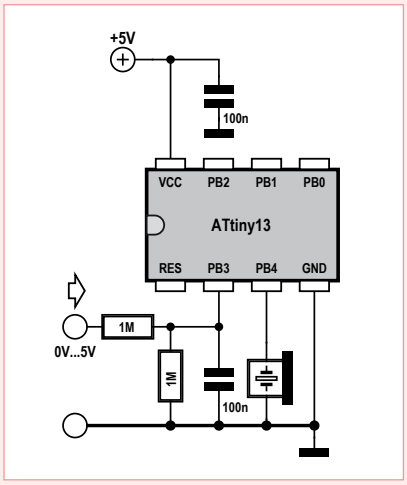


Figure 9. Un oscillateur à relaxation.

Convertisseur U/F avec ATtiny13

Le convertisseur tension-fréquence a ceci de bien qu'il peut servir de voltmètre acoustique. Ainsi, un son de hauteur donné pourrait signifier que la tension d'une batterie est correcte. Notre oreille est très sensible aussi aux variations de hauteur lentes, d'où l'idée d'utiliser un microcontrôleur pour obtenir une fréquence variant de façon linéaire avec la tension d'entrée.



Puisque l'ATtiny13 fonctionne ici sous 5 V, la plage de mesure du convertisseur A/N atteindra elle aussi 5 V. Nous la portons à 10 V grâce à un diviseur de tension à haute impédance. Un transducteur piézo-électrique est relié au port B4. Le programme réalise un simple générateur à synthèse numérique directe (DDS) avec une sortie à ondes rectangulaires. Les valeurs mesurées sont rangées dans l'accumulateur A, jusqu'à ce que le bit de poids fort s'inverse ; le programme commute alors la sortie numérique. Le format binaire de l'accumulateur est de 12 bits. Pour la tension la plus haute, il faut quatre périodes de mesure avant que l'état de la sortie ne change, soit une fréquence d'environ 600 Hz.

Le code source est à télécharger depuis www.elektor.fr/120007.

```
'U/f Converter  0...5 V 0...600 Hz
$regfile = «attiny13.dat»
$crystal = 1200000
$hwstack = 8
$swstack = 4
$framesize = 4

Dim U As Word
Dim A As Word

Config Adc = Single , Prescaler = Auto
Start Adc
Ddrb.4 = 1

Do
  U = Getadc(3)
  A = A + U
  A = A And &H0FFF
  If A >= &H0800 Then
    Portb.4 = 1
  Else
    Portb.4 = 0
  End If
Loop

End
```

multivibrateur. Les deux résistances de base sont reliées à la même tension d'entrée. Plus la différence de potentiel appliquée est grande, plus l'intensité du courant de charge est forte, et plus augmente la fréquence de l'oscillateur. Un diviseur de tension réglable permet de spécifier une tension quelconque, et donc de régler la fréquence. Ce circuit peut aussi servir de voltmètre acoustique pour la plage de mesure allant de 1 V à 9 V. Il permettrait p. ex. de tester rapidement (à l'oreille) différentes piles.

Bistable NPN/PNP

Les deux transistors ne sont pas forcément des NPN. Le circuit de la **figure 7** utilise deux transistors complémentaires pour réaliser un clignotant. Comme ci-dessus, la contre-réaction collecteur-base du transistor NPN fournit un point de fonctionnement stable. Le transistor PNP fonctionne comme émetteur-suiveur. La résistance auxiliaire placée sur la liaison au collecteur permet d'y prélever le signal en phase nécessaire à la rétroaction.
L'impédance de la branche de rétroaction est très haute. Un condensateur de rétroaction de seulement 1 µF donne ainsi une période d'environ une seconde.

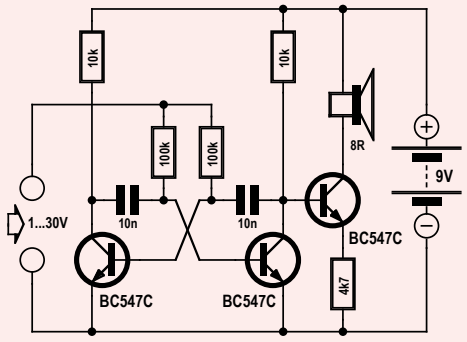
Flash à LED économe

Avez-vous déjà remarqué ces panneaux publicitaires dotés d'une LED qui semble clignoter éternellement sur une simple pile ? Le circuit de la **figure 8** est composé d'un multivibrateur astable aux propriétés spéciales. Un condensateur électrolytique de 100 µF se charge relativement lentement et avec un faible courant, puis se décharge à travers la LED sous la forme d'une courte impulsion, apportant ainsi l'élévation de tension nécessaire, puisqu'une tension de 1,5 V est insuffisante pour une LED.

Le circuit a été optimisé pour ne dissiper que peu de puissance. Aucun courant de commande n'est gaspillé, car le multivibrateur fonctionne avec un NPN et un PNP. Les deux transistors ne sont conducteurs que durant la courte période où la LED brille. La présence d'un étage supplémentaire avec contre-réaction en tension continue garantit que les conditions restent stables et que le circuit oscille ainsi de façon fiable. Ici aussi nous avons veillé à ce que les résistances soient de valeur élevée et la consommation minimale.
Le PNP n'est conducteur qu'un bref instant, à intervalle de quelques secondes, durant lesquelles le condensateur de sortie se charge jusqu'à près de 1,5 V. Cette tension s'ajoute à celle de la pile durant la phase active. Il se forme donc en marche à vide des impulsions de près de 3 V. Une LED rouge ou verte, qui tolère une tension directe de 1,8 à 2 V, lance des éclairs vifs.
La consommation peut être estimée à partir du courant de charge du condensateur. En moyenne, il règne aux bornes des deux résistances de charge de 10 kΩ une tension totale de 1 V, soit un courant de charge moyen de 50 µA. Une même intensité est prélevée sur la pile pour la durée des impulsions de la LED, ce qui donne en tout un courant moyen de 100 µA. Avec une capacité de 2.000 mAh, la pile devrait durer 20.000 heures, c'est-à-dire plus de deux ans. Ce n'est pas l'éternité, mais pour un clignotement permanent, c'est impressionnant.

Jeu-concours

Un simple multivibrateur commandé en tension sert de voltmètre acoustique grâce à un haut-parleur de 8 Ω (noté 8R). Le circuit a été analysé à l'oscilloscope avec une tension de service de 9 V et une tension de mesure elle aussi égale à 9 V. Lorsqu'un des transistors du multivibrateur devient conducteur, la tension de base de l'autre transistor chute à –9 V. Le condensateur de 10 nF est ensuite chargé par la tension de mesure positive qui lui est appliquée via la résistance de 100 kΩ. La charge dure 0,65 ms et atteint +0,6 V, ce qui provoque un nouveau basculement du circuit.



1) Quelle est la fréquence produite pour une tension de mesure de + 9 V ?

A) environ 3,3 kHz
B) environ 330 Hz
C) environ 770 Hz

2) Que devient la fréquence lorsque la tension de service baisse, mais que la tension de mesure reste la même ?

D) La fréquence baisse.
E) La fréquence reste constante.
F) La fréquence augmente.

3) Comment augmenter le volume sonore ?

G) Diminuer la valeur de la résistance d'émetteur de 4,7 kΩ du transistor de droite.
H) Prendre une résistance d'émetteur plus grande.
I) Remplacer la résistance d'émetteur par un condensateur électrolytique de 100 µF.

Si vous avez trois bonnes réponses, vous gagnerez peut-être par tirage au sort un ***Minty Geek Electronic 101 Kit*** !

Envoyez avant le 30 septembre 2012 votre réponse dans l'objet d'un message électronique adressé à basics@elektor.com sous forme d'un code composé des lettres correspondant à vos trois réponses, à l'exclusion de toute autre mention. L'objet du message sera lu automatiquement, le corps du message ne sera pas lu du tout…

Tout recours est exclu de même que le sont, de ce jeu, les personnels d'Elektor International B.V. et leur famille. Un seul gagnant par foyer.

La solution du jeu de mai était le code ADH :

Solution 1 :

Le BF245B stabilise environ 10 mA. La bonne réponse était donc la A).

Solution 2 :

Pour une tension d'entrée élevée, l'intensité d'un circuit avec résistance-talon augmenterait de façon significative, ce qui entraînerait une grande dissipation de chaleur. Le FET, en revanche, maintient le courant constant, d'où des pertes de chaleur moindres. D) était donc le bon choix.

Solution 3 :

Le condensateur électrolytique soutient la tension lors des variations rapides du courant de charge, et améliore la stabilisation pour les hautes fréquences. Il possède p. ex. une résistance capacitive de 1,6 Ω pour 1 kHz, et réduit ainsi la résistance interne de la tension de stabilisation. Il contribue de façon notable au maintien de la tension durant 1 ms, mais guère plus longtemps. La réponse H) était la bonne.

Oscillations à relaxation

Les oscillations à relaxation, avec leur forme typique en dents de scie, se produisent lorsqu'un condensateur se charge périodiquement jusqu'à atteindre une certaine tension, puis se décharge de façon soudaine. Sur le circuit de la **figure 9**, le transistor PNP reste bloqué tant que le condensateur est chargé, et aucun courant de base ne circule non plus dans les deux NPN. Le seuil de basculement est déterminé par le diviseur de tension que forment les deux résistances de 10 kΩ, soit ici 4,5 V + 0,6 V (d'où un potentiel de la base inférieur de 0,6 V à celui de l'émetteur). À partir d'environ 5,1 V circule donc un courant, que la rétroaction amplifie en un courant de décharge intense. La tension chute alors jusqu'à 0,6 V. Les transistors passent ensuite à l'état bloqué, et le processus de charge recommence. La sortie de ce circuit à trois transistors, conçu pour une charge très lente, est une sorte de métronome : tic, tic, tic…

Se passer du transistor de gauche simplifie le circuit. L'ensemble fonctionnera comme précédemment dans de nombreux cas, mais si le courant de charge est faible, la disparition du « thyristor » formé par le couple NPN-PNP (cf. l'épisode précédent) laissera parfois le circuit à l'état conducteur. La variante à trois transistors n'a pas ce défaut ; le circuit fonctionne de façon stable dans une large plage de courants de charge. Nous pouvons encore simplifier ce circuit en retirant le condensateur électrolytique, puisque le transducteur piézo est lui-même un semblant de condensateur. Avec l'accélération des oscillations, le lent générateur d'horloge se transformera alors en générateur sonore !

(120007-I – version française : Hervé Moreau)

interface Nunchuk USB
recyclage de manette de jeux

Anthony Le Cren (Le Mans)

La console de jeux *Wii* est livrée avec un accessoire appelé *Nunchuk* : une deuxième manette qui contient un accéléromètre à trois axes, un stick analogique et deux poussoirs. Il suffit d'un PIC 18F2550 pour communiquer avec cette interface homme machine au moyen du protocole I²C et l'utiliser pour d'autres applications, p. ex. en robotique, en modélisme, pour le DMX etc.

La célèbre console de jeux *Wii* de la multinationale japonaise *Nintendo* utilise une manette sans fil *Bluetooth* appelée *Wiimote*. Celle-ci peut être connectée par un câble à une autre unité de commande dite *Nunchuk*, grâce à laquelle le joueur peut utiliser ses deux mains dans un jeu vidéo, *Wiimote* dans une main, *Nunchuk* dans l'autre. Ce qui m'intéresse ici est de détourner la manette *Nunchuk* à l'aide d'une carte à base du µC PIC18f2550. Celui-ci se fait passer pour la *Wiimote*, et accède ainsi au moyen du protocole de communication I²C aux données contenues dans la *Nunchuk*. Une fois les données récupérées, l'utilisateur aura plusieurs possibilités pour les exploiter :

- les afficher sur un écran LCD,
- les envoyer vers le PC par une interface homme-machine (souris, clavier),
- les utiliser pour commander des actionneurs,
- les rediriger vers une ligne RS232 pour les exploiter avec un autre microcontrôleur.

Ces données se composent des indications correspondant aux trois axes X, Y et Z de l'accéléromètre, à la position du stick analogique et à l'état des deux boutons appelés C et Z. On en verra les détails ci-dessous. Voilà pour le principe de mon interface de détournement de *Nunchuk* (fig. 1).

Fonctionnement

Comme on s'y attend, l'ensemble s'articule autour du microcontrôleur, le 18F2550, associé ici à un quartz de 20 MHz dont la fréquence mérite une explication. Ce µC peut être configuré de 12 façons différentes. Or, pour la fréquence d'horloge, l'utilisation du bus USB impose un multiple de 12 MHz. C'est pourquoi la fréquence du quartz de 20 MHz est divisée par 5, avant d'attaquer une PLL interne verrouillée sur 96 MHz. Pour finir, cette fréquence est divisée par 2, soit une fréquence finale de 48 MHz. Ce détail n'est pas anodin, il faut en tenir compte dans l'élaboration des programmes. Vous pouvez retrouver le synoptique de configuration de l'oscillateur dans la documentation du fabricant du PIC (pag.24 *Oscillator types*) [4].

Caractéristiques de l'interface Nunchuk-USB

- manette connectée directement sur le circuit imprimé principal
- protocole I²C
- lecture des données numériques de l'accéléromètre (10 bits), du stick analogique (8 bits) et des boutons poussoirs (actifs au niveau bas).
- connecteur USB pour interface avec le PC
- programmation du PIC avec un *bootloader* et logiciel PDFSUSB
- programmation des applications avec Flowcode
- connecteurs DB9 femelles compatibles avec les modules E-blocks
- extension à 4 broches pour future liaison série

L'alimentation sous 5 V est fournie par le câble USB (K2) connecté sur le PC. La tension d'alimentation de la manette est ramenée à 3,3 V par le régulateur LP2950ACZ-3.3 (IC2). Les résistances de polarisation du bus I²C sont intégrées à la manette, il est donc inutile d'en prévoir sur notre carte.

Cette carte d'interface cumule aussi le rôle de carte de développement. C'est pourquoi toutes les broches du microcontrôleur sont rendues accessibles sur deux connecteurs DB9 : K4 regroupe toutes les broches d'entrées analogiques du 18F2550, K5 est câblé sur les 6 bits de poids fort du PortB, dont les labels du port sont nommés spécialement pour accueillir un afficheur classique à cristaux liquides de deux lignes de 16 caractères. Mais cet accessoire n'est évidement pas obligatoire. Les 2 bits de poids faible du PortB sont réservés au bus I²C (K2) pour la communication avec la manette, tandis que K3 est prévu pour une future liaison série UART externe.
Il est facile d'ajouter d'autres extensions, des capteurs ou d'autres périphériques. Les LED D1 à D4, actives au niveau logique haut, sont disponibles pour vos propres applications.

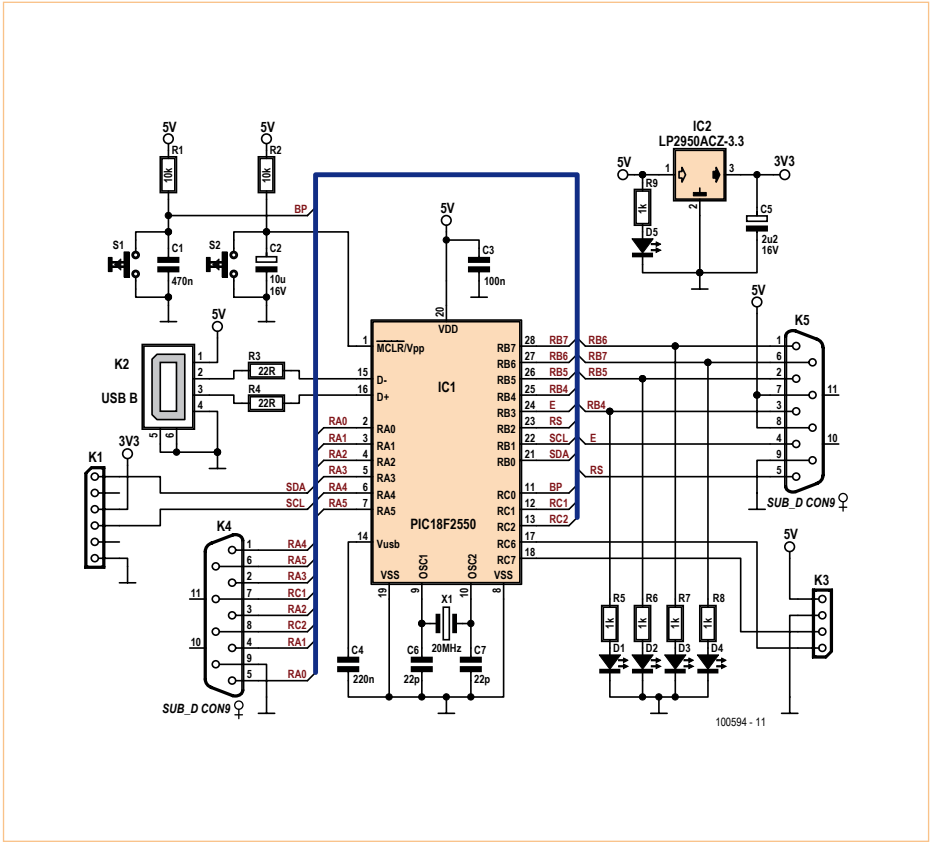


Figure 1. Le schéma de la carte d'interface se résume à un PIC flanqué de quelques connecteurs.

Deux boutons poussoirs permettent l'un l'initialisation du microcontrôleur (S2) et l'autre d'accéder au programme d'amorçage (ou *bootloader*) du PIC (S1).

Pour mesurer la trame I²C (fig. 2) à l'aide d'un oscilloscope ou d'un autre outil analyseur de bus, il suffira de relever le signal sur les lignes SCL et SDA, et, pour la synchronisation, la ligne RC2 (broche 8 de K4), sans oublier la liaison de masse. Ce signal SYNCRO passe au niveau haut lorsque l'on commence à lire des données de la *Nunchuk*.

Les applications possibles pour un accéléromètre comme celui de la manette sont nombreuses ; il est donc important de pouvoir changer à votre guise les exemples fournis sur le site d'Elektor [1], d'où l'utilisation d'un programme d'amorçage (*bootloader*) comme sur de nombreuses applications à microcontrôleur.

Une fois le micrologiciel *Nunchuk.hex* [1] implanté dans le PIC, l'utilisateur pourra accéder au contenu de la ROM Flash de 32 Ko à l'aide du logiciel *Pdfsusb* fourni par *Microchip* [3] et ainsi adapter les programmes d'exemples en fonction de ses besoins.

Protocole I²C

Sur le bus I²C, le PIC sera toujours le maître et la manette l'esclave.

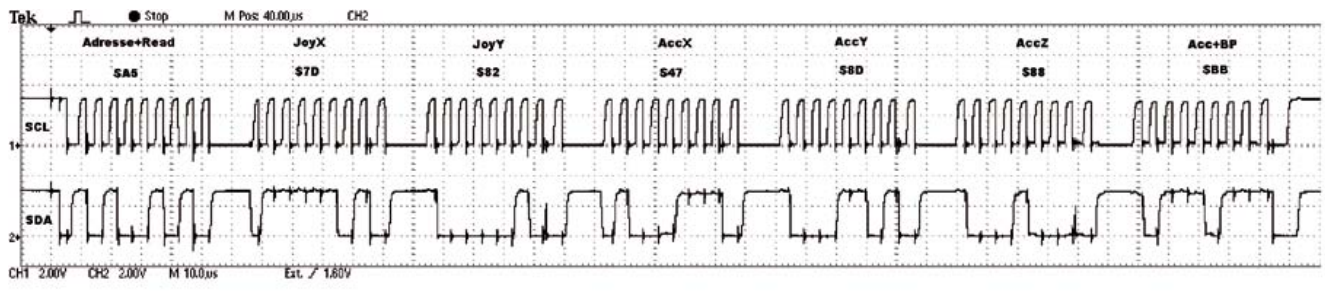


Figure 2. La lecture de la trame complète s'effectue en sept octets.

Tableau 1.

Opération	adresse								R/W	hex
	D7	D6	D5	D4	D3	D2	D1	D0		
écriture : transfert de données de l'interface vers la manette *Nunchuk*	1	0	1	0	0	1	0	0		$A4
lecture : transfert de données de la manette *Nunchuk* vers l'interface	1	0	1	0	0	1	0	1		$A5

Tableau 2. Initialisation de la manette *Nunchuk*

start bus I²C	Les données du bus I²C circulent la carte d'interface vers la manette *Nunchuk* (fig. 3)
envoi 0xA4	
envoi 0xF0	
envoi 0x55	
stop bus I²C	
start bus I²C	
envoi 0xA4	
envoi 0xFB	
envoi 0x00	
stop bus I²C	

Tableau 3. Initialisation du pointeur dans la RAM de la manette *Nunchuk*

start bus I²C	Les données du bus I²C circulent la carte d'interface vers la manette *Nunchuk* (fig. 4)
envoi 0xA4	
envoi 0x00	
stop bus I²C	

Tableau 4. Lecture des données de la manette *Nunchuk*

start bus I²C	Les données du bus I²C circulent de la manette *Nunchuk* vers la carte d'interface sauf pour le 1er octet 0XA5 (commande de lecture) (fig. 4)
envoi 0xA5	
réception joystick X dans **NUN_BUF[0]**	
réception joystick Y dans **NUN_BUF[1]**	
réception accéléromètre X (MSB) **NUN_BUF[2]**	
réception accéléromètre Y (MSB) **NUN_BUF[3]**	
réception accéléromètre Z (MSB) **NUN_BUF[4]**	
réception accéléromètres (LSB) + état des boutons C et Z **NUN_BUF[5]**	
stop bus I²C	

L'adresse I²C de la *Nunchuk*, figée et fixée par le fabricant, codée sur 7 bits, est $52 (1010010). Il existe en effet d'autres périphériques, comme la manette *wii* classique, que l'on peut connecter également à la *Wiimote*. Chaque périphérique *Wii* possède donc son adresse propre.
Comme pour tout bus I²C, le premier octet transmis par le maître n'est pas une donnée mais une adresse. Le format de l'octet d'adresse est un peu particulier, puisque la fonction du bit D0 (R/W) est d'indiquer si le maître demande une lecture à l'esclave ou si, au contraire, le maître impose une écriture à l'esclave. En partant de l'adresse $52 codée sur 7 bits, cela donne les deux octets de commande du tableau 1 sur la page ci-contre.

Avant de commencer la lecture des données de la manette, il faut l'initialiser en écriture pour accéder aux registres qui nous intéressent, avec la suite d'octets du tableau 2 sur la page ci-contre.
En suivant le lien [2] vous trouverez de plus amples explications sur ces registres internes.
Ensuite vient la mise à zéro du pointeur RAM contenant les données de la manette par la séquence (fig. 7) du tableau 3 sur la page ci-contre.

Pour finir, la lecture de la trame complète (fig. 2) s'effectue en sept octets :

- condition de début (*start*)
- 1 octet d'adresse plus l'opération de lecture ($A5) émis par le PIC (maître)
- 6 octets de données émis par la manette (esclave).
- condition de fin (*stop*)

Les six données reçues seront stockées sous forme de tableau de valeurs dans la RAM du 18F2550 (**NUN_BUF[0]** à **NUN_BUF[5]**) des tableaux 4 et 5 sur la page ci-contre.

Comme le format binaire du mot de l'accéléromètre est de 10 bits, on récupère les bits de poids faible (LSB) dans le dernier octet. Il

Tableau 5. Protocole I²C en lecture de la manette *Nunchuk*

octet 1 0xA5	octet 2 NUN_BUF[0]	octet 3 NUN_BUF[1]	octet 4 NUN_BUF[2]	octet 5 NUN_BUF[3]	octet 6 NUN_BUF[4]	octet 7 NUN_BUF[5]
adresse Nunchuk + opération de lecture	joystick X (0 à 255) 8 bits	joystick Y (0 à 255) 8 bits	accéléromètre X poids fort (MSB) (bits 9 à 2)	accéléromètre Y poids fort (MSB) (bits 9 à 2)	accéléromètre Z (bits 9 à 2)	accéléromètre poids faible + boutons C & Z

Tableau 6.

D7	D6	D5	D4	D3	D2	D1	D0
ac-cél. Z	ac-cél. Z	ac-cél. Y	ac-cél. Y	accél. X	accél. X	bouton C	bouton Z
bit 1	bit 0	bit 1	bit 0	bit 1	bit 0	actif au niveau bas	actif au niveau bas

faut concaténer le mot afin d'obtenir une valeur comprise entre 0 et 1023. L e détail de ce codage binaire apparaît clairement sur le tableau 6 ci-dessus.

On détermine des valeurs des trois axes grâce au calcul suivant :

```
joy_x = NUN_BUF[0]
joy_y = NUN_BUF[1]
accel_x = (NUN_BUF[2]*4)+((NUN_BUF[5]/4) AND 0x03)
accel_y = (NUN_BUF[3]*4)+((NUN_BUF[5]/16) AND 0x03)
accel_z = (NUN_BUF[4]*4)+((NUN_BUF[5]/64) AND 0x03)
bouton_c = (NUN_BUF[5] /2 ) AND 0x01
bouton_z = NUN_BUF[5] AND 0x01
```

Réalisation

Comme il n'y a pas de composants CMS, l'implantation des composants ne pose pas de problème particulier. On commencera par souder les plus petits composants, puis le support de circuit intégré, les LED et les condensateurs, pour terminer par le quartz et les connecteurs. Auparavant, la connexion de la *Nunchuk* au circuit imprimé (K2) mérite une mention spéciale : en effet, il faut pratiquer à la mini-disqueuse une double découpe dans le circuit imprimé afin de pouvoir enficher le connecteur d'origine de la manette.

Sur le circuit imprimé il n'y a pas de connecteur, le contact s'établit directement avec les pistes de cuivre. Attention au sens d'insertion : tourner le connecteur avec le détrompeur (creux au centre) du côté soudures (fig. 5 & 6), le coté plat étant tourné vers le côté composants.

Reste la programmation du PIC avec son micrologiciel USB (fichier *nunchuk.hex*). Pour cela deux options : se procurer le PIC préprogrammé auprès d'Elektor [1], ou bien utiliser un programmateur universel si votre microcontrôleur est vierge.

Le fichier *nunchuk.hex* contient le *bootloader* plus le programme d'exemple `hid_clavier_trame.fcf` (voir ci-dessous les exemples de programmes).

Il n'y a pas de pilote à installer car, vue du PC, l'interface est un clavier appartenant à la classe de périphérique HID : ces pilotes sont

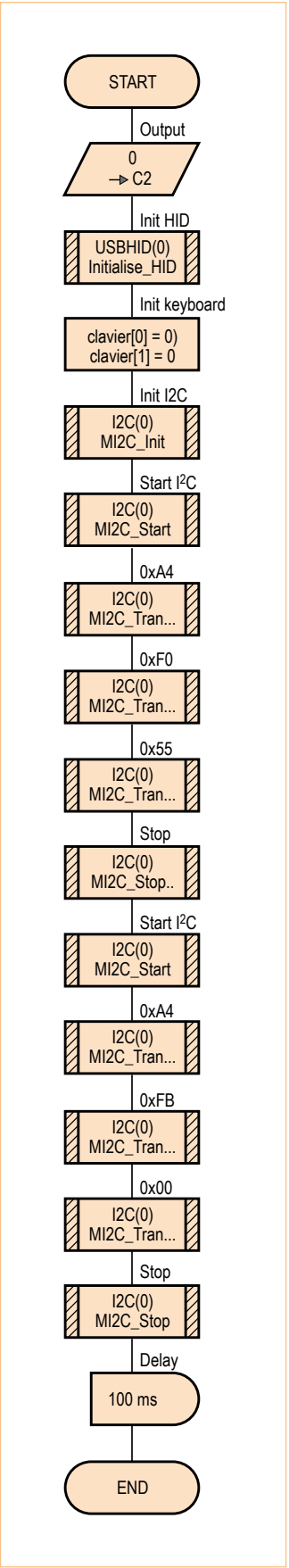


Figure 3. Initialisation de la manette *Nunchuk*.

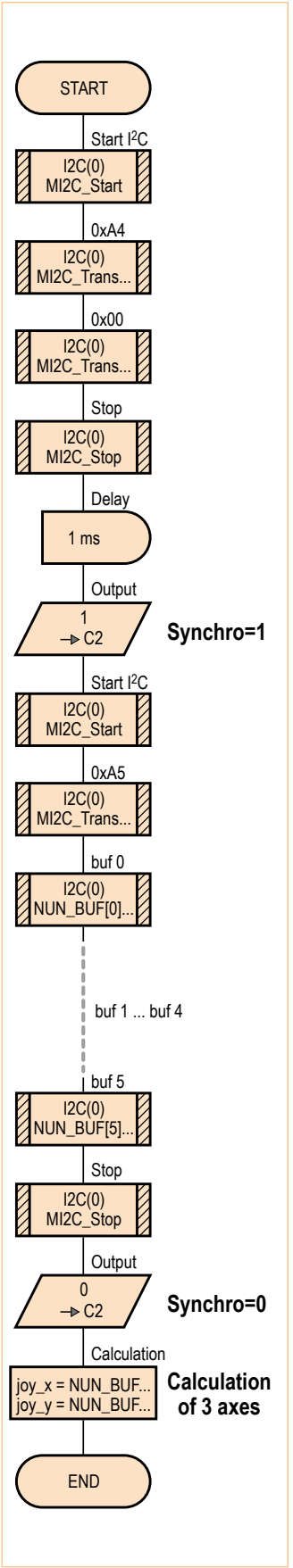


Figure 4. Lecture des données de la manette *Nunchuk*.

inclus dans les systèmes d'exploitation, de sorte que l'interface est compatible sans installation avec Windows, Linux, MacOS etc.

Logiciel

Tous les programmes fournis sont réalisés avec *Flowcode V4*. Ce logiciel est tout indiqué pour programmer le 18F2550 car la gestion des pilotes HID (human interface device) pour la liaison USB ainsi que pour le bus I²C est abordable par un débutant en informatique. Les adeptes du langage C transposeront plus difficilement les programmes, mais n'oublieront pas de préciser au compilateur de démarrer le programme à partir de l'adresse $800 pour ne pas effacer le *bootloader* si l'on veut continuer à utiliser la programmation USB (voir le tableau 7 ci-dessus).

Utilisation du bootloader
Dès la mise sous tension ...
Pour programmer un fichier hex Flowcode, il suffit de maintenir enfoncé le bouton *boot* lors de la mise sous tension de la carte. La LED D1 clignote pour indiquer que les pilotes doivent être installés.

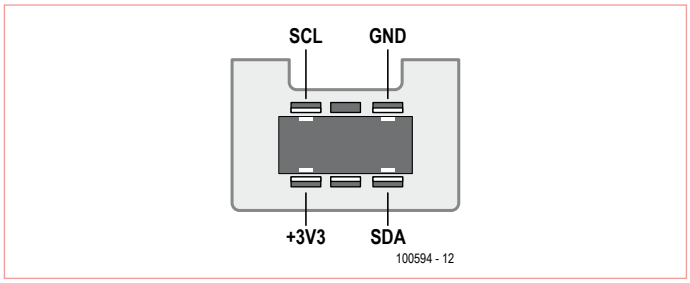


Figure 5. Connecteur de la *Nunchuk* avec le détrompeur (creux) à tourner du côté soudure.

Figure 6. Les deux encoches faites à la mini-disqueuse sont indispensables pour pouvoir enficher le câble de la manette.

Tableau 7.

Adresse	
	mémoire Flash
	programme utilisateur
	Flowcode
$0800	
$07FF	*bootloader*
	(ne pas effacer)
$0000	

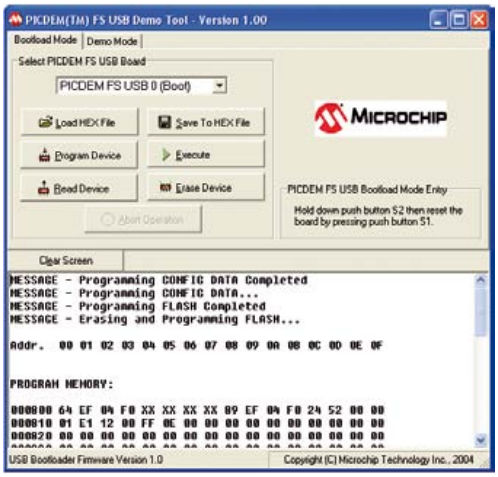


Figure 7. Exécution du logiciel de programmation PDFSUSB.exe permettant d'accéder au contenu de la ROM Flash du 18f2550.

Les divers fichiers sont contenus dans le paquet *USB Framework* de Microchip [3]. Il est vivement conseillé de choisir l'installation dans le répertoire par défaut. Vous trouverez les pilotes dans le répertoire `\Microchip Solutions v2010-10-19\USB Tools\MCHPUSB Custom Driver\MCHPUSB Driver\Release`. Ils sont téléchargeables sur le site d'Elektor [1].

Après installation du pilote, exécutez le logiciel de programmation PDFSUSB.exe qui se situe dans le répertoire `\Microchip Solutions v2010-10-19\USB Tools\Pdfsusb`.

Sélectionnez le pilote PICDEM FSUSB 0 (Boot).
Chargez un fichier hex, cliquez sur *program device*, puis *execute* (fig. 7).

Exemples de programmes

Je propose quatre exemples d'applications [1] qu'il sera facile de modifier pour les réutiliser ailleurs ou autrement.

hid_clavier_trame.fcf (programme de base inclus dans le firmware)
Ce programme permet de tester le bon fonctionnement de l'ensemble. Lorsque le bouton Z de la *Nunchuk* est pressé, les valeurs numériques des trois axes sont envoyées sur la liaison USB. L'affichage se fera dans le bloc-notes, puisque la carte d'interface émule un clavier de PC. Lorsque le bouton BP1 est maintenu enfoncé c'est seulement la valeur numérique de l'axe X qui est transmis au PC. Une analyse peut être alors effectuée dans un tableur comme le montre la figure 8.

hid_joytick.fcf
La *Nunchuk* se transforme en joystick sur les axes X et Y. On accède au calibrage du joystick dans les paramètres de configuration de Windows (Contrôleurs de jeu). Attention : sans ce calibrage, le fonctionnement du joystick n'est pas optimal.

hid_souris_accel.fcf
Transforme la manette *Nunchuk* en souris d'ordinateur : axe X =

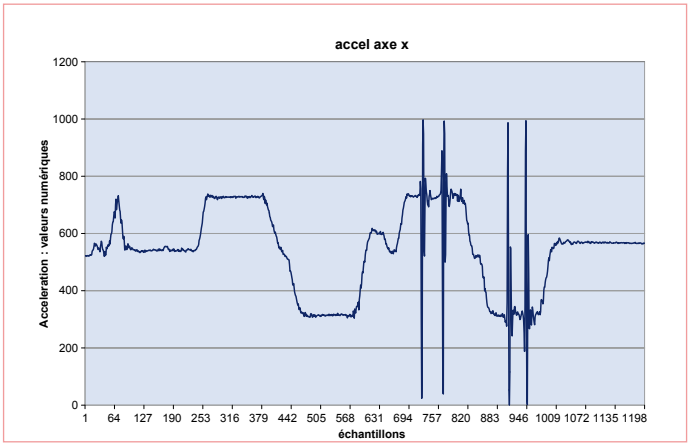


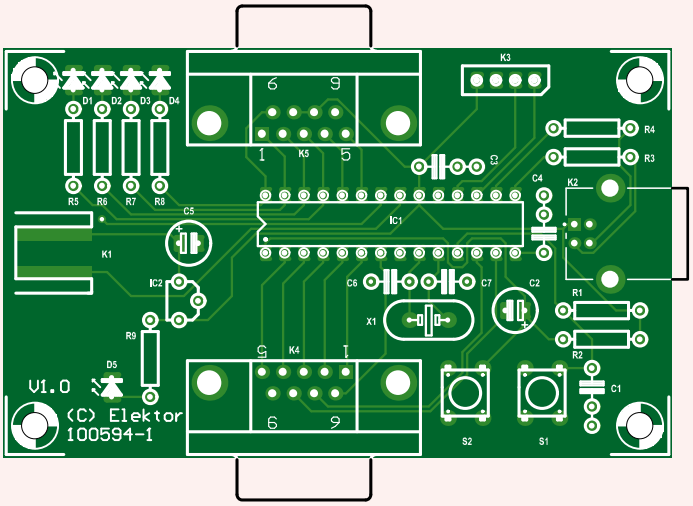
Figure 8. Le bouton *boot* permet aussi d'envoyer les valeurs de l'axe X de l'accéléromètre, à partir desquelles une analyse peut alors être effectuée dans un tableur.

Liste des composants

Résistances:
R1, R2 = 10 k
R3, R4 = 22 Ω
R5 à R9 = 1 k

Condensateurs :
C1 = 470 n
C2 = 10 µ/16 V rad.
C3 = 100 n
C4 = 220 n
C5 = 2µ3/16 V rad.
C6,C7 = 22 p

Semi-conducteurs :
D1 à D5 = LED 3 mm à faible courant
IC1 = PIC18F2550-I/P DIP à 28 broches (EPS 100594-41)
IC2 = LP2950ACZ-3.3



Divers :
S1, S2 = bouton poussoir
K2 = connecteur USB-B à encarter
K3 = embase SIL mâle à 4 broches
K4, K5 = connecteur subD à 9 broches, femelle, coudé, à encarter
X1 = quartz 20 MHz

circuit imprimé EPS100594-1

déplacement droite/gauche et axe Y = déplacement haut/bas. Boutons Z & C : clic gauche/droit. Au début, on est un peu dérouté par la sensibilité de la manette.

nunchuck_dmx.fcf
La *Nunchuk* se transforme en contrôleur de lyre (projecteur de scène) utilisant le protocole DMX sur le connecteur série K3 (RC6 : TXD). Attention : il faudra adapter le programme en fonction du projecteur utilisé et ajouter un circuit intégré de bus RS485 : 75176.

(100594)

Liens

[1] www.elektor.fr/100594

[2] http://wiibrew.org/wiki/Wiimote/Extension_Controllers#Nunchuk

[3] USB FrameWork Microchip : http://ww1.microchip.com/downloads/en/DeviceDoc/MCHP_App_Lib_v2010_10_19_Installer.zip
Liste des mises à jour :
www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2896

[4] www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010280

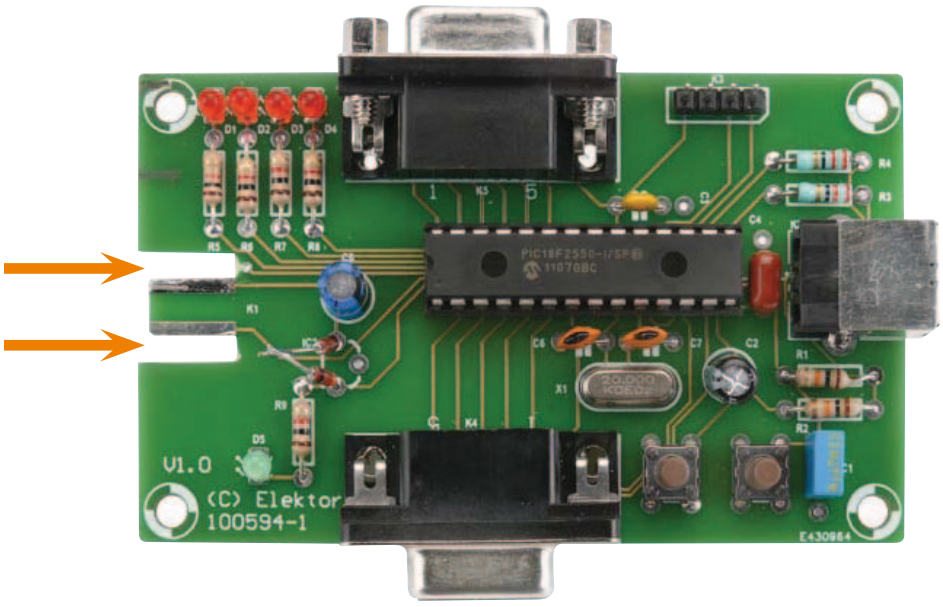
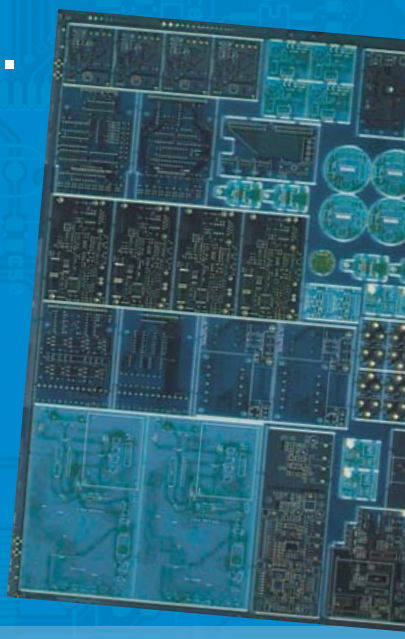


Figure 9. D'origine, les encoches dans le circuit imprimé de part et d'autre de K1 pour la prise de la *Nunchuk* ne sont pas faites. L'interface sérielle (K3) n'a pas encore été implantée dans la version actuelle du logiciel n'est utilisée que par le programme nunchuck_dmx.fcf et les applications personnelles que vous imaginerez.

elektor PCB Service

Présentation par Eurocircuits

Lé référence Européenne pour les C.I. Prototypes et petites séries.



Accélérez votre cycle de développement tout en réduisant les coûts grâce nos services pooling rapides et faciles d'accès en ligne:

- Haute qualité professionnelle et prix modiques
- Livraisons rapides pour répondre à vos contraintes
- Soutien technologique performant
- Sans frais de dossier et d'outillages
- Sans minimum de commande à partir d'une pièce
- Sans règlement par avance
- Service pochoirs

PCB proto – *service rapide à prix très attractifs dédié aux bureaux d'études*

- 1 ou 2 C.I. en 2, 3, 5 ou 7 jours ouvrés
- Contrôle complet du dossier, finition comprenant 2 vernis épargne verts et une sérigraphie blanche – Isolements/conducteurs >/= 150µm
- 1 C.I. 100 x 80mm en 7 jours - 2 faces 46,49 € - 4 couches 94,41 €
- 2 C.I. 100 x 80mm en 7 jours - 2 faces 36.47 € pièce - 4 couches 73.89 € pièce

Prix avec TVA Française de19.6% transport non compris

STANDARD pool – *la plus large gamme d'options en pooling d'Eurocircuits*

- 1 à 8 couches isolements/conducteurs >/= 150µm
- A partir de 2 jours ouvrés

TECH pool – *la force du pooling au service des C.I. haute densité*

- 2 à 8 couches isolements/conducteurs >/= 100µm
- A partir de 4 jours ouvrés

IMS pool – *C.I. à haute dissipassion thermique pour application de puissance (LED, conversion d'énergie, …)*

- C.I. 1 face sur semelle métallique (SMI) – Isolements/conducteurs >/= 200µm
- Cuivre 35µm avec isolant 75µm sur semelle en aluminium 1.5mm
- A partir de 3 jours ouvrés

On demand – *C.I. spéciaux minces et rigides*

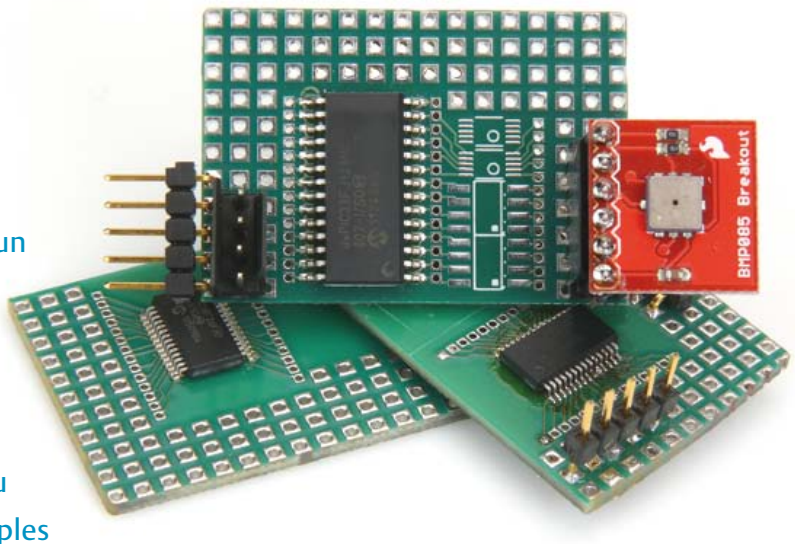
- 0 à 16 couches isolements/conducteurs >/= 90µm
- CI hyperfréquences et haut Tg
- A partir de 2 jours ouvrés

www.elektorPCBservice.com

PICo PROto

outil de prototypage minimaliste pour PIC16 ou 18

Michel Kuenemann (Riedisheim)



Comment s'y prendre pour tester rapidement un nouveau capteur ou évaluer une nouvelle idée sans avoir à modifier ou bien câbler une carte de développement complexe et coûteuse ? Cette question, je me la pose deux fois par jour en période faste, et vous ? En vertu du principe selon lequel les choses les plus simples sont celles qu'on utilise le plus, j'ai eu l'idée du PICo PROto, un outil de prototypage minimaliste...

Principales caractéristiques techniques :

- accepte les boîtiers SO28 et SSOP28
- taille et coût minimaux
- souplesse maximale
- mise en œuvre éclair

Boulimique de nouvelles technologies, j'aime par-dessus tout évaluer moi-même les nouveaux composants, surtout dans le domaine des capteurs de type MEMS (*microelectromechanical systems*). Alimentés en général sous 3,3 V, ils délivrent leurs données sur des sorties analogiques ou au travers d'un bus I²C, plus rarement sur bus SPI. Beaucoup de ces capteurs modernes sont encapsulés dans des boîtiers minuscules, impossibles à souder manuellement. Heureusement, les fabricants de ces merveilles se soucient des développeurs censés les mettre en œuvre et leur proposent souvent une petite carte d'évaluation sur laquelle est soudé le capteur. Ils appellent *break out board* ces cartes munies d'un connecteur au pas de 2,54 mm, relié fil par fil aux pattes du composant principal.

J'utilise aussi de façon courante plusieurs types de microcontrôleurs de la famille *Microchip* PIC18 en boîtiers à 28 broches. Comme *Microchip* a eu la très bonne idée de standardiser le brochage de ces composants, le passage d'un type de composant à l'autre est facile. De plus, comme ces composants existent aussi en boîtiers traditionnels DIP traversants, il est envisageable de réaliser des montages d'essai sur plaque d'expérimentation sans aucune soudure. Ces plaques à ressorts présentent cependant trois défauts majeurs : primo, leur taille et leur poids sont incompatibles avec mes applications embarquées dans le domaine de l'aéromodélisme. Secundo, leurs contacts électromécaniques sont aléatoires, voire chaotiques et tertio, il n'est pas possible de conserver ces montages bien longtemps, car il faut dare-dare libérer la précieuse plaque pour une nouvelle expérimentation. Vous auriez raison de m'opposer ici que *Microchip* propose de nombreuses cartes d'évaluation économiques qui ne demandent qu'à s'adapter à mes idées, mais là encore, pour la plupart de mes essais, la disproportion est flagrante entre mes besoins d'une part et d'autre part la richesse de ces cartes en connecteurs, leur encombrement et le prix qui en découlent.

J'ai donc conçu une carte minimaliste double-face, pour un PIC 16 ou 18, en boîtier SO28 d'un côté ou SSOP28 de l'autre, ainsi que quelques composants CMS supplémentaires. Autour de la partie occupée par les composants CMS, une zone de pastilles carrées au pas de 2,54 mm peut accueillir des composants traversants et les incontournables barrettes qui connecteront le proto au monde extérieur. Sa petite taille de 28 x 42 mm permet de l'intégrer aussi bien dans un modèle réduit volant ou un petit robot mobile que

Ce brochage convient à tous les PIC 16 ou 18 en boîtier à 28 broches

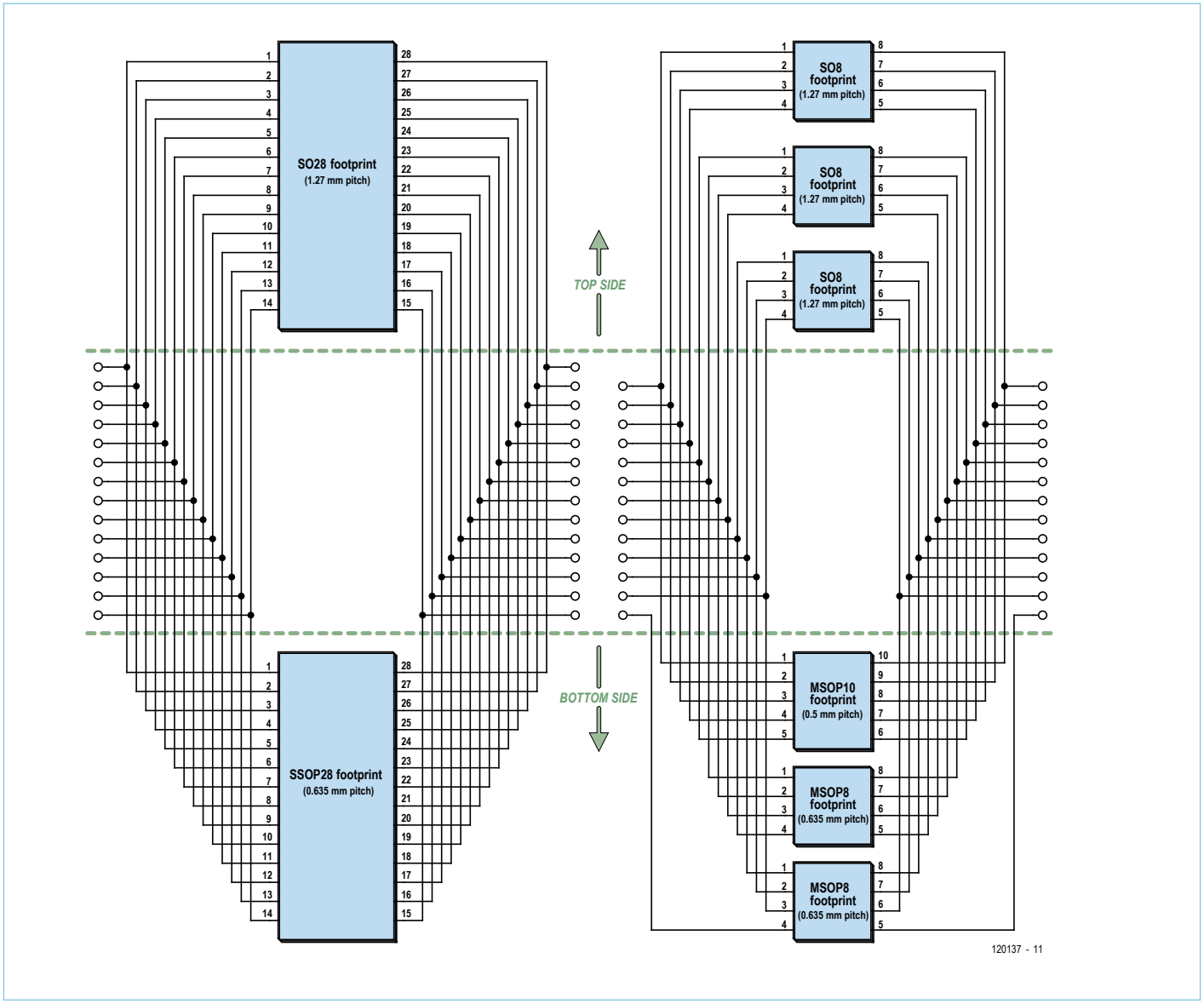


Figure 1 – Ceci n'est pas un schéma au sens habituel, mais plutôt un fantôme puisqu'il ne représente pas les composants, seulement leur empreinte. Le PICo PROto est compatible avec tous les PIC à 8 bits, en boîtier à 28 broches.

dans une réalisation plus volumineuse. La contrainte de l'exiguïté a ceci de bon qu'elle oblige le concepteur à rester focalisé sur l'objectif de sa manipulation sans céder à la tentation de « charger la mule » avec des composants superflus.

Le schéma

Pas de surprise, c'est la transparence même ; d'ailleurs, on peut difficilement appeler ça un schéma (**fig. 1**) puisqu'il n'y a pas de composants. On peut donc passer directement à la pratique. En plus des empreintes SO28 et SSOP28 propres au contrôleur (un seul monté à la fois, s'il vous plaît !), j'ai prévu des empreintes SO8 et MSOP8. C'est ce que montre plus clairement encore la reproduction agrandie du circuit imprimé double-face (**fig. 2**). Voici les photos de deux de mes **PICo PROto,** côté pile et côté face. Leur tracé diffère un peu de la version définitive. À gauche sans composant, à droite, côté face avec un TSSOP28 (dont la qualité des soudures est tout sauf exemplaire).

Comment utiliser PICo PROto ?

Selon le type de son boîtier, soudez tout d'abord le microcontrôleur choisi côté pile ou côté face. Soudez ensuite au bord de

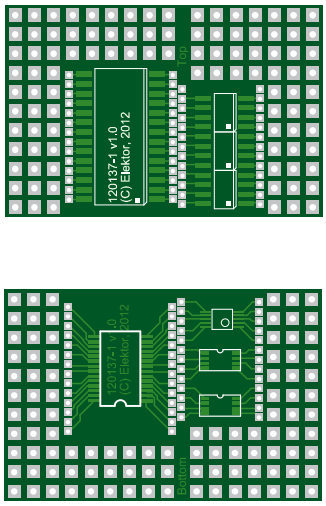


Figure 2 – La carte double-face à trous métallisés PICo PROto

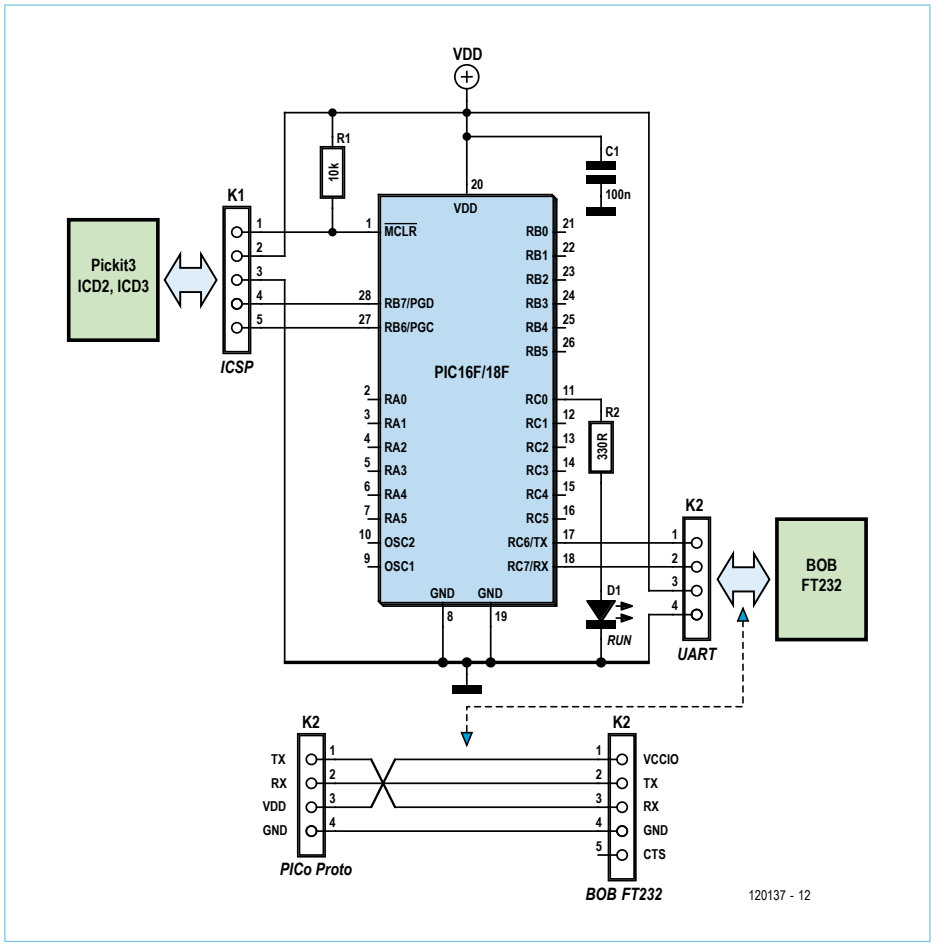


Figure 3 – Les modalités de câblage du PICo PROto.

la carte une barrette de 5 broches au pas de 2,54 mm, éventuellement coudée. Elle permettra de connecter l'indispensable programmateur Pickit ou ICD pour flasher et déboguer vos programmes. Connectez la barrette au micro comme vous l'indique le schéma de mise en œuvre (**fig. 3**), mais n'allez pas plus loin pour l'instant.

Vérifiez que votre environnement de développement MPLAB reconnaît et voit le composant. Pour ceci, connectez votre programmateur à la platine et à votre PC de développement. Lancez MPLAB, puis, dans le menu *Programmer*, sélectionnez votre outil et vérifiez qu'il se connecte correctement. Supposons que vous utilisez un Pickit3. À ce stade, MPLAB vous indiquera l'erreur suivante :

You must connect to a target device to use PICkit.

Cette erreur est liée au fait que votre micro n'est pas encore alimenté. Allez dans *Programmer→settings→Power*, vérifiez que le curseur est sur la tension adaptée à votre application et cochez la case *Power target circuit from Pickit3*.

Si MPLAB vous gratifie d'une erreur de type *Target Device ID (00005b60) does not match expected DeviceID (00005960)*, c'est bon signe si *Target Device ID* est différent de 0.

Dans *Configure→Select device* sélectionnez le type exact de micro que vous utilisez et tout devrait rentrer dans l'ordre. La partie la moins passionnante de l'aventure est derrière vous !

Si vous avez envie de communiquer avec votre micro au moyen de l'interface USB, appelez à la rescousse un module BOB FT232 [2]. Après avoir soudé et câblé le connecteur K2, la connexion pourra être établie ; le BOB FT232 se chargera aussi d'alimenter votre platine (voir au bas de la fig. 4 le schéma d'interconnexion de PICo PROto avec le BOB). Dans ce cas, avant de connecter le BOB, n'oubliez pas de décocher dans MPLAB la case *Power target circuit from Pickit3*. Sur le BOB, vous aurez aussi pris soin de sélectionner la

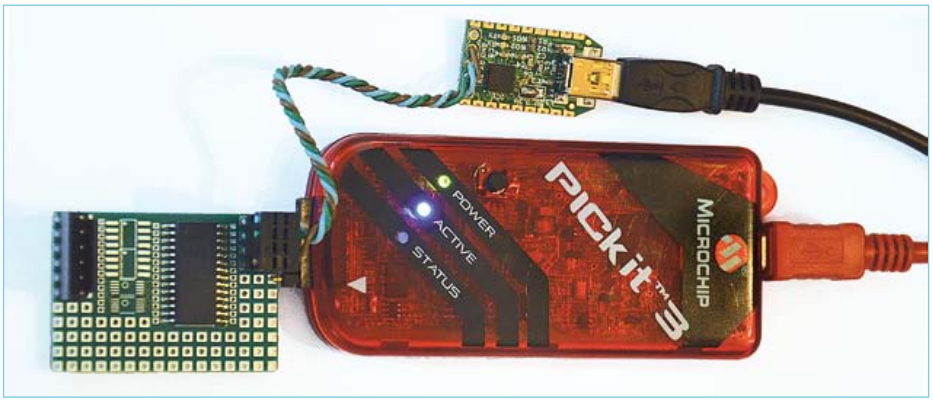


Figure 4a – Application avec le BOB pour l'évaluation d'un capteur barométrique.

tension d'alimentation qui convient à votre montage au moyen du cavalier JP1.

Si le clignotement d'une LED vous rassure sur le bon déroulement du programme en cours de fonctionnement, n'hésitez pas à câbler une LED rouge comme indiqué sur le schéma.
Il faudra moins d'une heure à un expérimentateur aguerri pour parvenir à ce stade de la réalisation. La place libre est à présent disponible pour vos essais les plus audacieux.

Pour conclure voici encore deux exemples (**fig. 4**) de mise en œuvre du **PICo PROto** :

- Évaluation d'un capteur barométrique, avec le BOB

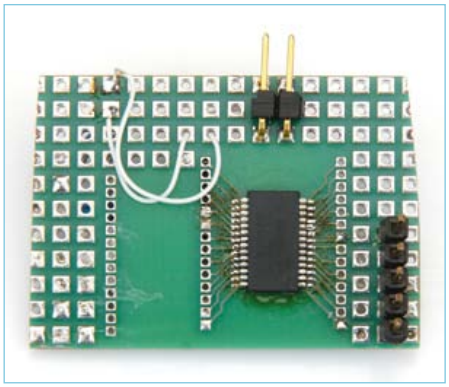


Figure 4b – Côté pile : PIC18F27J13 en boîtier SSOP28.

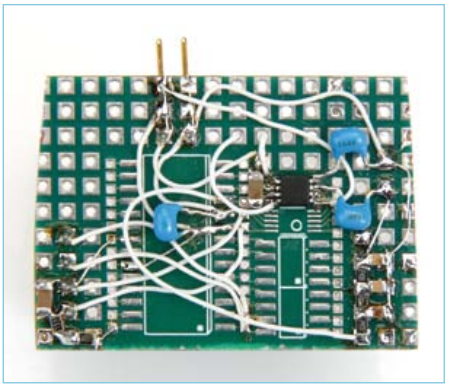


Figure 4c – Côté face, un amplificateur en classe D SSM2301 testé pour le compteur de longueurs de natation publié dans le numéro d'été 2012.

- Évaluation d'un procédé de synthèse vocale avec, côté pile, un PIC18F27J13 en boîtier SSOP28, et côté face, un amplificateur en classe D SSM2301

120137-I

Liens

[1] www.elektor.fr/120137

[2] Passerelle USB/série BOB-FT232R, Elektor sept. 2011, www.elektor.fr/110553

interface pour train électrique

avec USB et éditeur de script

Willem Tak (Pays-Bas)

Combien de trains électriques ont pris la voie de garage et dorment désormais sous la poussière d'un grenier ? On en rêvait, on nous l'avait offert ou on l'avait acheté pour les enfants, mais après l'avoir fait tourner quelque temps, le plaisir s'est émoussé. On n'a pas pensé à allonger les voies, à les agrémenter, à automatiser le fonctionnement. Voici un petit circuit qui peut rendre vie à votre train miniature et en augmenter les possibilités sans achat onéreux.

Chaque année, au mois de décembre, ma voisine construit dans son jardin un paysage de Noël animé par un train miniature qui le parcourt en boucle. À force de tourner sans jamais varier même d'un iota, le spectacle finit par lasser. Il y a mieux à faire pour varier les plaisirs, changer de sens ou de vitesse. Et pourquoi pas le faire s'arrêter en gare de temps à autre ?

Le potentiel est vaste, pensons à l'excellent EDiTS d'Elektor, mais il y a plus simple. L'idée est de rédiger un script qui rassemble différentes missions : marche avant ou arrière, arrêt d'autant de secondes, retour à la gare, etc. Il n'y a plus qu'à le faire exécuter par un circuit spécialement conçu à cet effet.

Le matériel

Le circuit a été développé pour des trains miniatures qui fonctionnent en courant continu. Il a été essayé avec un *Fleischmann 9336*.
Le cœur du montage est un PIC 18F4550 de *Microchip* (cf. **fig. 1**). Le choix repose sur son interface USB intégrée pour communiquer avec l'éditeur de script sur le PC. C'est très simple : vous rédigez sous *Windows* un script qui commande l'interface. Pour faire fonctionner l'USB, il faut un quartz à 24 MHz.

Pour manier l'ensemble, j'ai choisi un LCD à deux lignes et un encodeur rotatif à bouton-poussoir. L'écran se raccorde directement sur le PIC. Il y a une résistance (R14) sur la ligne de port RD7 parce que le logiciel utilise le drapeau *busy*.

Avec son relais Re1, l'alimentation a un air étrange, c'est dû au fait que le circuit combine deux fonctions, l'interface USB et la commande de train. Quand le relais est au repos, la tension de 5 V sur le connecteur USB sert à alimenter le contrôleur. Il fonctionne alors en interface homme-machine (IHM ou *HID*) pour l'USB et l'on peut lire et charger des scripts. Si c'est l'alimentation de 15 V qui est commutée, le relais s'active et le contrôleur est alimenté par un 7805 (IC2).

Les impulsions de voie pour le train sortent de la ligne RB4 du contrôleur. Il s'agit de signaux modulés en largeur d'impulsion à une fréquence de 100 Hz. Des impulsions brèves le font rouler au ralenti, les larges lui donnent l'air d'un TGV. En sortie du contrôleur, elles ont une amplitude de 5 V que T1 agrandit à 15 V avant que des triggers de Schmitt IC3.A à IC3.C n'en rectifient les flancs. Elles passent alors dans un MOSFET à canal N du type FQP12N60. Il est capable de fournir un fort débit, jusqu'à 10 A, mais sans être un connaisseur du modélisme ferroviaire, j'imagine aisément que certains trains miniatures consomment nettement plus que celui qui a servi aux essais du circuit. La plupart du temps, cependant, 1 A devrait suffire. L'inversion de sens de marche est réalisée par le relais Re2 qui inverse simplement la polarité sur les rails. Il faut déterminer par un essai le bon sens de branchement. Si le programme veut faire aller le train en avant et qu'il va dans l'autre sens, on permute les fils sur les rails.

Comme le moteur du train présente une forte induction, il renvoie des parasites de grande amplitude. Aussi ai-je prévu, entre les contacts du relais et le connecteur K2, un réseau antiparasite C12/C13/L1/L2 destiné à protéger l'électronique de commande.

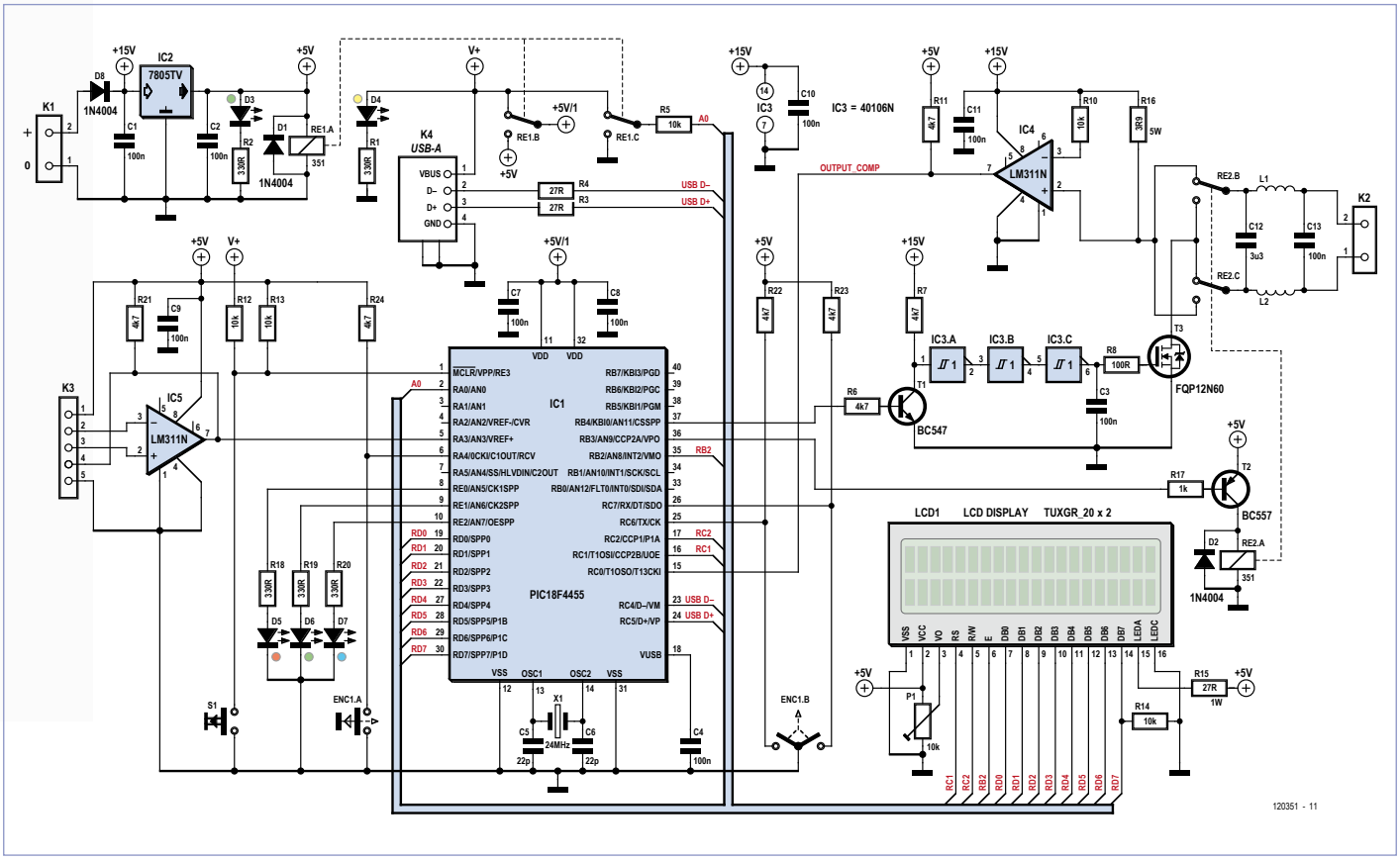


Figure 1. Le contrôleur produit un signal modulé en largeur d'impulsion qui active le train miniature par l'intermédiaire du MOSFET T3.

Le condensateur C3 entre le CD40106 et le MOSFET y participe aussi. Les flancs du signal de commande en sont légèrement moins raides.

L'amplificateur opérationnel IC4 procure au contrôleur, par la broche 16, une rétroaction de manière à ce qu'il sache si les instructions du script sont vraiment exécutées. Tant que le train circule, IC4 reçoit les impulsions sur son entrée non inverseuse (+). C'est une barrière lumineuse, dont le schéma séparé est à la **figure 2**, qui assure

Ce qu'indiquent les LED

LED internes (D3/D4) :

- jaune (D4) — allumée en mode USB
- verte (D3) — allumée en mode train avec alimentation externe

LED de face (D5/D6/D7), mode USB :

- bleue(D7) — clignote pour l'initialisation du mode USB
- bleue(D7) — allumée sous le programme trein.exe
- bleue(D7) — peut être commandée par le programme trein.exe

LED de face (D5/D6/D7), mode train :

- rouge (D5) — allumée en mode train et si le train roule
- rouge (D5) + verte(D6) — allumées si le train est arrêté par la commande d'attente
- verte(D6) — allumée si le train est arrêté en gare
- bleue(D7) — allumée si le train s'est arrêté sans avoir trouvé de gare

- rouge + verte+ bleue — allumées pour une erreur LCD
- rouge + bleue — allumées pour une erreur EEPROM
- chenillard de LED — le train ne roule plus selon le script

la détection de l'entrée en gare. C'est une simple construction en forme de U par laquelle deux planchettes de bois se dressent de part et d'autre de la voie. L'une porte une LED, l'autre une LDR. Quand le train arrive dans la barrière, il interrompt le faisceau lumineux que projette la LED sur la LDR. On règle le niveau de détection à l'aide d'un potentiomètre de 10 kΩ raccordé à K3. Les différentes LED sur la platine indiquent les états des tensions et de la communication. Pendant l'alimentation par le câble USB, la LED jaune D4 est allumée. Quand la liaison USB est mise en service, la LED bleue D7 en façade clignote. Dès le lancement du programme Trein.exe sur le PC, D7 s'allume en continu. Sous une alimentation externe en 15 V, la LED verte D3 est allumée et la LED rouge D5 en façade commence à clignoter pour signaler l'initialisation, puis reste allumée. Une vue d'ensemble de la signalisation par LED se trouve ci-contre dans l'encadré **Ce qu'indiquent les LED**.

Composition des octets

Le script fournit 2 octets par pas. Le codage est le suivant :

octet haut bit 7 :	1 en marche avant
octet haut bit 6 :	1 en marche arrière
octet haut bit 5 :	1 en recherche de gare
octet haut bit 4 :	1 en attente
octet haut bit 3 :	bit de service
octet haut bit 2 à 0 :	vitesse du train (1 à 7)
octet bas :	durée de 1 à 255 s

Le bit de service (bit 3) a plusieurs fonctions, selon l'octet dans lequel le mot est utilisé.

dans l'octet 0:	bit de bascule pour une écriture en EEPROM
dans l'octet 2:	un 1 indique le script1
dans l'octet 4:	un 1 indique le script2
dans l'octet 6:	un 1 indique le script3
dans l'octet 8:	état de la LED bleue USB de face

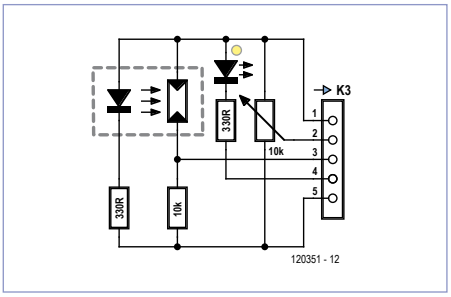


Figure 2. Ces composants sont à monter près de la barrière lumineuse, reliés à la platine avec K3.

Le logiciel

Le logiciel pour le 18F4550 est totalement rédigé en assembleur, celui pour le PC, qui permet de créer les scripts, est en *Visual Basic* (VB).
Le logiciel en assembleur se compose de deux parties : l'interface USB-IHM et la commande de train.

La section USB

Le programme USB est une interface HM standard pour le PIC 18F4550. Un appareil USB s'identifie par la combinaison VID/PID (*Vendor ID* et *Product ID*). Ici, elle est fixée d'origine à 0D59/5275. Après une énumération correcte, la boucle USB standard se lance, réglée sur 6 ms. L'interface USB communique avec le PC par l'intermédiaire d'un tampon de 64 octets. C'est par lui que transitent les scripts de l'environnement

VB pour être chargés dans le PIC. Le programme USB a pour tâche de stocker et de lire les scripts dans l'EEPROM. On peut utiliser 3 scripts.

Les commandes ainsi que les données proviennent du programme VB. J'ai choisi une taille de 32 actions par script, ce qui s'est révélé suffisant en pratique, car ce format s'adapte convenablement aux 64 octets du tampon de l'IHM. Il y a aussi de la place pour un certain nombre de bits de commande, puisque l'octet de l'instruction ne totalise que 7 bits. Vous trouverez leur classification dans l'encadré **Composition des octets**.
C'est le programme USB qui règle la chronologie de la transmission avec le PC. Pour éviter au programme des difficultés dues à la lenteur d'écriture dans l'EEPROM, Microchip parle de 4 ms, on n'y écrit qu'un seul octet par tour dans la boucle USB.

Toute la procédure d'écriture est mise en route par un octet de basculement dans l'environnement VB et un certain nombre de drapeaux dans la section USB. Elle prévoit l'écriture d'un octet pendant 64 cycles consécutifs. Pendant cette période, le PC lève un drapeau pour indiquer que le temps d'attente est en cours et que l'on peut introduire une donnée.

Sauf lors du démarrage, le programme USB ne lit jamais l'EEPROM, mais s'arrange pour que le tampon USB soit toujours au courant de l'état le plus récent du script actuel par l'intermédiaire de tampons séparés du PIC.

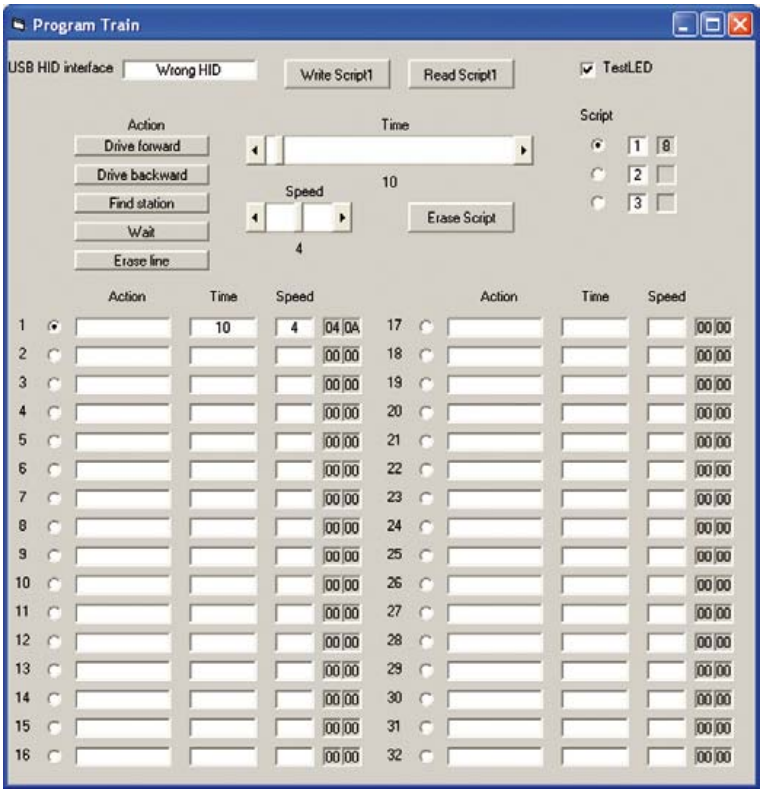


Figure 3. Le programme *Windows* correspondant facilite la composition d'un scénario d'évolution pour le train électrique.

La section train

Le PIC connaît un second mode de démarrage, le mode train ; le choix s'opère à l'aide de l'entrée A0 du PIC.
L'interface de train communique avec l'utilisateur par le LCD à deux lignes dont le contraste se règle avec P1. L'encodeur rotatif sert à parcourir les différentes tâches ; on en sélectionne une en poussant sur le bouton.

Le logiciel en assembleur est très simple. Vous chargez une ligne du script pour être exécutée. La routine qu'on appelle le plus souvent, c'est celle qui fournit les signaux MLI. Un temporisateur y détermine la durée de l'action choisie. L'écran indique en permanence le déroulement des pas.
En mode train, tout commence par le programme de menu qui lance un des trois scripts, puis la conduite du train passe en mode manuel. On atteint les réglages par le bouton de mise à zéro.

Le cœur du programme, c'est la routine DRIVE. Elle tourne à 100 Hz pour les boucles de temporisation normales (pas pour les interruptions) et elle commande le train en variant la largeur des impulsions qui se répètent à 100 Hz. Leur rapport cyclique est consigné dans une table et dépend de la largeur minimum réglée. La routine fournit aussi les impulsions de secondes pour la chronométrie des différentes actions.
Deux routines ont accès à l'encodeur rotatif avec des sensibilités différentes (*fast* ou *secure*). Le bouton poussoir pour le démarrage et l'interruption fait partie de l'encodeur.

Rédaction des scripts

Le programme Trein.exe (**figure 3**) sous *Windows* sert à la rédaction des scripts que l'interface de train exécutera ensuite. L'ordre des connexions est important avec ce programme. En tout premier lieu, raccorder le câble USB d'un PC en marche. Lors de la première liaison, il se peut que *Windows* affiche un message d'erreur, mais en général, l'appareil s'installe correctement et les fois suivantes, il n'y aura plus de souci. Lancer alors le programme Trein.exe. La LED bleue en façade s'allume et le programme inscrit dans la boîte de l'interface USB IHM le message : *Treininterface*. On peut vérifier le bon fonctionnement de l'interface USB

avec le bouton TestLED : mettre ou enlever la coche active ou éteint la LED bleue. Le programme permet de composer et d'adapter des scripts, de les lire dans la mémoire de l'interface de train et de les remettre en mémoire.

On peut y mettre trois scripts, chacun de 32 lignes au maximum. Une de ces lignes permet de programmer les actions suivantes : marche avant, marche arrière, chercher une gare ou attendre.
Le programme est totalement commandé par la souris. Du même coup, il n'est pas toujours logique. Chacune des quatre actions énumérées est assortie d'une variable de temps à régler au moyen d'une glissière entre 1 et 255 s. Ce sera la durée de l'action.

À l'exception de l'attente, les actions sont aussi accompagnées d'un réglage de vitesse. Une autre glissière la définit entre 1 et 7, en proportion directe de la vitesse à donner au train.
Le fonctionnement est simple : choisir un script, sélectionner une action ou un temps, qui s'inscrit dans la ligne d'instruction.
On termine le script par une ligne blanche. Si les 32 lignes contiennent des actions, le programme bouclera automatiquement vers l'exécution de la première.
Il y a une exception à cette règle pour le premier script. Si ses 32 lignes sont complétées, le programme continuera à s'exécuter à la 1re ligne du deuxième script. Et si le script 2 contient aussi 32 lignes actives, l'exécution continuera à la 1re ligne du script 3. Il est donc possible de composer un script de 96 pas.

Le programme dispose de boutons pour rappeler les scripts de la mémoire (*read script*) et pour écrire le script actuel dans la mémoire du contrôleur (*write script*). Il faut savoir que les opérations d'écriture ne demandent aucune confirmation, elles s'exécutent immédiatement. Pendant les opérations de lecture et d'écriture, le programme reste inaccessible un court instant, ce qui est notifié dans une fenêtre d'attente.

Le programme montre aussi un certain nombre de fenêtres en gris. Elles contiennent les valeurs écrites en mémoire et n'intéressent que le programmeur.

Construction

Le laboratoire Elektor a conçu pour cette interface pour train électrique une platine (**figure 4**) qui porte l'ensemble du système, y compris l'étage de puissance. Sa construction est simplifiée du fait de l'utilisation exclusive de composants à fils. Le stabilisateur IC2 est doté d'un petit radiateur. Le MOSFET se place en bordure de la platine de manière à pouvoir y visser aussi un radiateur si nécessaire.

La construction de la platine terminée et vérifiée, on peut installer le microcontrôleur programmé dans son support. Il est disponible préprogrammé auprès d'Elektor, si vous ne pouvez pas le programmer vous-même. On monte ensuite l'écran par-dessus la platine à l'aide de quatre entretoises de 2 cm de long. Les connexions entre l'écran et la platine peuvent se réaliser au moyen de petits morceaux de fil ou avec un connecteur encartable à 2x8 contacts et sa fiche correspondante avec des picots très longs. Après quoi, il n'y a plus qu'à relier la platine au PC par un câble USB et commencer la rédaction d'un script. Vous pourrez alors le charger dans l'EEPROM du contrôleur.

Utilisation

Le raccordement de l'interface au circuit ferroviaire est simple. Branchez sur K1 un bloc secteur capable de délivrer 15 V sous un courant d'un ampère ou plus, selon le modèle de train utilisé, et reliez K2 aux rails du train. Avec les valeurs indiquées pour L1 et L2 dans la liste des composants, le circuit peut fournir 1 A maximum, ce qui plus que suffisant pour la plupart des trains.
Les composants repris dans la figure 2, à vous de les monter sur ou près de la barrière lumineuse.

Dès la mise en service de l'alimentation de 15 V, l'interface démarre en mode train. La LED rouge s'allume et la commande s'effectue au moyen de l'encodeur et de son interrupteur à poussoir, sous l'écran LCD.
Après le message de bienvenue, vous pouvez choisir avec l'encodeur rotatif l'une des cinq tâches suivantes : script1, script2, script3, commande manuelle et scripts de test. La tâche choisie s'effectuera après une pression sur le bouton.
Les trois premières lancent le script correspondant.

L'écran LCD indique sur la première ligne la commande actuelle du script ainsi que son numéro d'ordre. Sur la ligne du bas, l'action réellement prise, y compris le sens de marche et le temps qu'il lui reste en secondes.

La marche avant et arrière, comme l'attente, sont évidentes ; la recherche d'une gare demande un mot d'explication. Vous en trouverez davantage dans un document à consulter sur le site d'Elektor [1].

Liste des composants

Résistances :
R1, R2, R18 à R20 = 330 Ω
R3, R4 = 27 Ω
R5, R10, R12 à R14 = 10 kΩ
R6, R7, R11, R21 à R24 = 4,7 kΩ
R8 = 100 Ω
R15 = 27 Ω / 1 W
R16 = 3,9 Ω / 5 W
R17 = 1 kΩ
P1 = 10 kΩ ajustable

Condensateurs :
C1 à C4, C7 à C11, C13 = 100 nF
C5, C6 = 22 pF
C12 = 3,3 µF MKT, au pas de 15 mm

Inductances :
L1, L2 = self de choc 330 µH, 0,9 A, 0,32 Ω (p.ex. Panasonic ELC10D331E, ref. Farnell n° 1749073)

Semi-conducteurs :
D1, D2, D8 = 1N4004
D3, D6 = LED verte 5 mm
D4 = LED jaune 5 mm
D5 = LED rouge 5 mm
D7 = LED bleue 5 mm
T1 = BC547
T2 = BC557
T3 = FQP12N60
IC1 = PIC18F4455-I/P (préprogrammé : EPS 120351-41)
IC2 = 7805
IC3 = 40106
IC4, IC5 = LM311

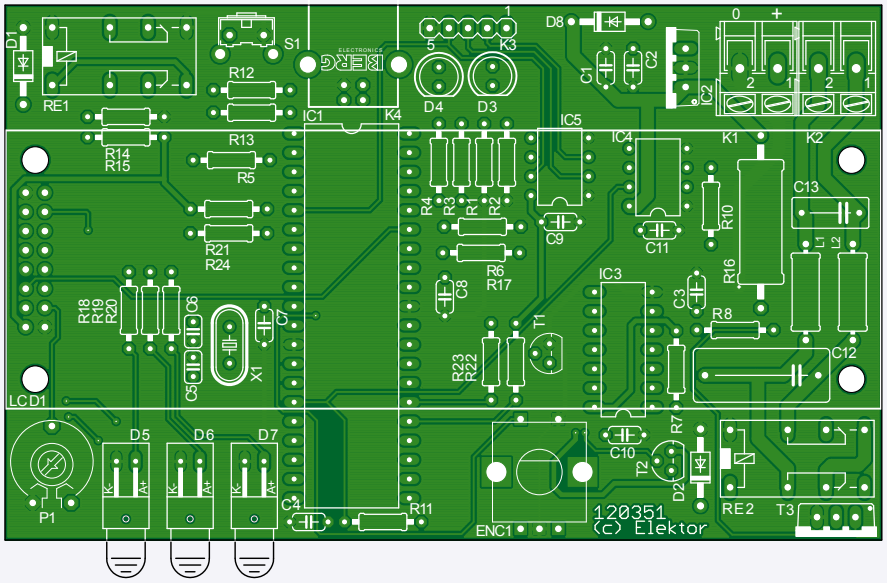


Figure 4. Les dimensions de la platine sont accordées à celles de l'écran qui vient s'y superposer, monté sur quatre entretoises.

Divers :
RE1, RE2 = relais encartable5 V (p.ex. V23106-A5401-A201)
LCD1 = LCD 2 x 20 caractères éclairage arrière (p.ex. Midas MC22005A6W-SPTLY)
ENC1 = encodeur rotatif + poussoir (p.ex. Bourns PEC11-4230F-S0024)
K1, K2 = embase encartable à 2 contacts au pas de 5 mm
K3 = embase SIL à 5 contacts
K4 =connecteur USB-B coudé encartable
S1 = bouton-poussoir (p.ex. Omron B3F-3100)
X1 = quartz à 24 MHz
platine 120351-1 (cf. www.elektor.fr/120351)

Sachez déjà que trouver une gare demande en fait deux actions : d'abord, le train miniature doit franchir la barrière lumineuse qui balise la gare. Au cours de cette action, le temps maximum s'affiche. Le temps de parcours est toujours inconnu, puisqu'il dépend de la position du train et de sa vitesse. Dès que la gare est trouvée, le temps d'attente, prévu dans le script à la ligne de la gare, s'indique.

On peut à tout moment mettre fin à l'exécution d'un script en poussant sur le bouton.

Outre le logiciel pour PC et le micrologiciel pour le microcontrôleur, des informations complémentaires (120351-FR-W.docx) sont disponibles sur [1].

(120351-I)

Lien

[1] www.elektor.fr/120351

isolateur USB

adieu parasites et boucles de terre

Raymond Vermeulen (Elektor)

Si vous avez un appareil USB perturbé par des bruits gênants dus à une boucle de masse ou si vous voulez protéger votre PC des tensions extérieures, un isolateur USB constitue une solution de choix. Le circuit décrit ici assure une séparation galvanique optimale autant des lignes de bus que des lignes d'alimentation entre le PC et l'appareil connecté par USB.

Ce projet est une conséquence collatérale de l'idée antérieure d'un oscilloscope portable pour Androïd. Le problème d'alors était le bruit d'une boucle de masse avec l'appareil branché à un chargeur ou à un PC. Quel était le meilleur point d'insertion d'un isolateur, la liaison USB ou la partie analogique ? Après une analyse approfondie, nous sommes arrivés à la conclusion que le mieux était d'isoler la ligne USB et non le côté analogique.

Un circuit très répandu d'*Analog Devices*, l'ADuM3160, a été choisi pour isoler les lignes de données. Il supporte les débits USB dits *Low-speed* et *Full-speed*. L'isolement de l'alimentation est assuré par un régulateur indirect (*flyback*) avec le contrôleur LT3575.

CI spécialisé

Le circuit intégré ADuM3160[1] utilisé dans ce projet et son frère plus musclé, ADuM4160, sont à notre connaissance les seuls isolateurs USB sur le marché à ce jour. Ce sont eux qui sont utilisés dans la plupart des isolateurs USB du commerce. Ils supportent les débits USB *Low-speed*

Caractéristiques techniques

- Séparation galvanique des lignes de données et d'alimentation
- Débit réglable à 1,5 Mbit/s ou 12 Mbit/s
- Alimentation pour le côté périphérique possible par un raccordement USB supplémentaire à l'hôte ou par alimentation externe (5 V/min. 0,5 A)
- Tension d'isolement du transformateur = 1500 V_{AC} pendant 1 min
- Tension d'isolement du ADuM3160 = 565 $V_{crête}$ pendant 50 ans
- Intensité maximale de sortie I_{out} = 500 mA sous 5 V

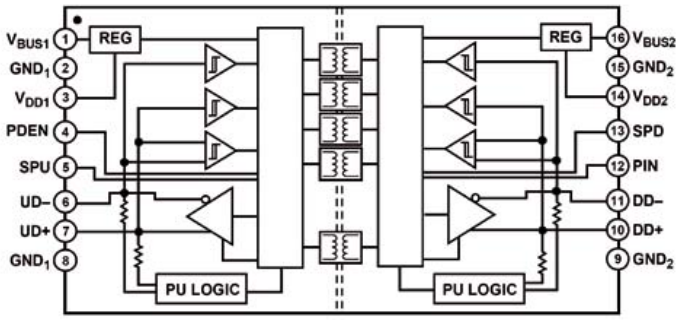


Figure 1 - Schéma synoptique des entrailles du CI ADuM3160. Plusieurs transformateurs séparés assurent la séparation galvanique entre les deux côtés USB.

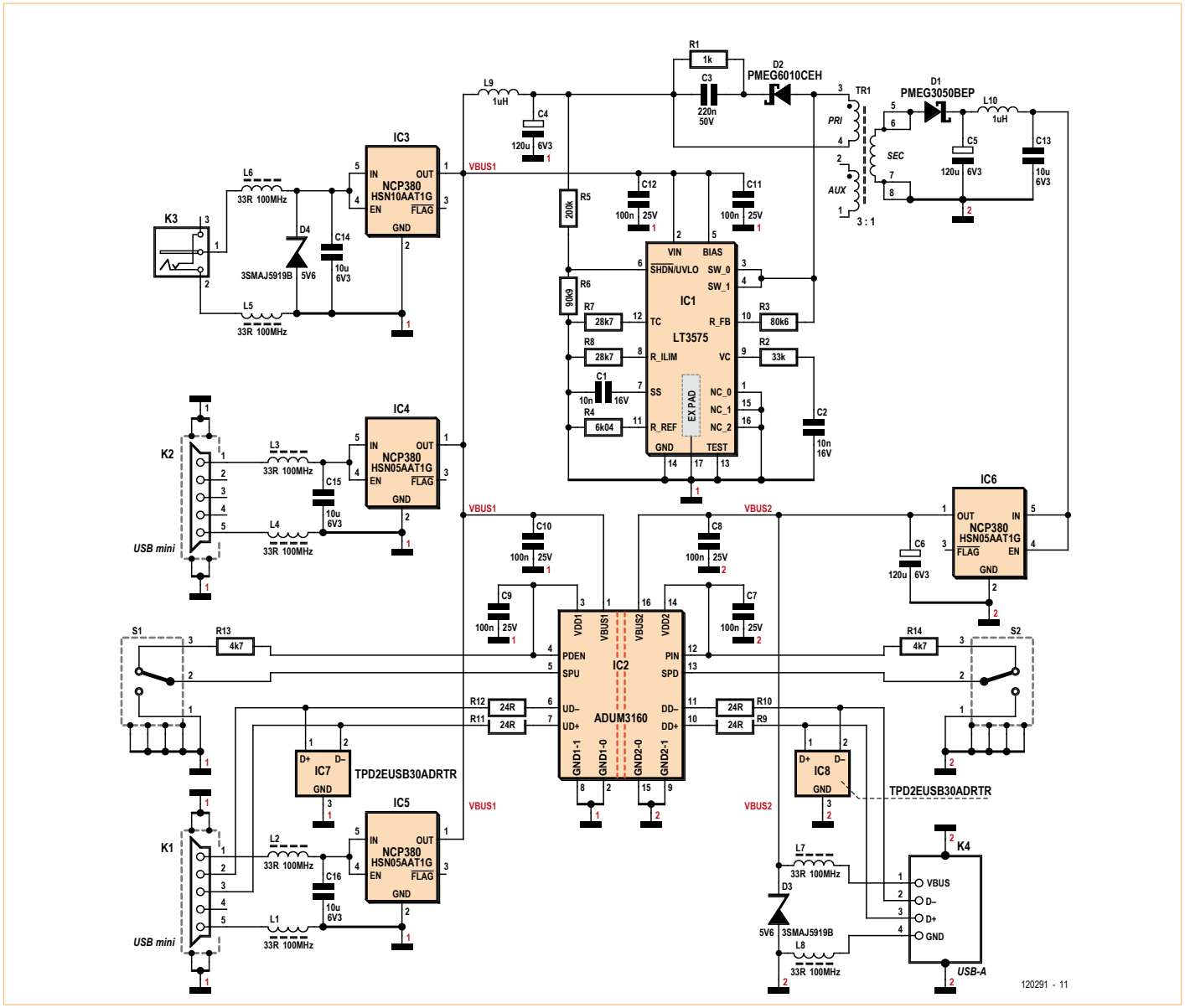


Figure 2 - Le schéma de l'isolateur USB. Le rôle principal est confié à un circuit intégré spécialisé d'Analog Devices (IC2) qui assure la liaison isolée des signaux USB.

(1,5 Mbits/s) et *Full-speed* (12 Mbits/s). Si aucun isolateur USB *High-speed* (480 Mbits/s) n'est fabriqué, c'est selon *Analog Devices* parce qu'aucun circuit intégré ne fonctionne correctement à cette haute fréquence. Pour l'instant aucun nouveau circuit intégré n'est prévu.

La **figure 1** montre la conception interne du circuit intégré. La combinaison de la technologie CMOS rapide, de transformateurs et de couplages optiques (technologie *iCoupler* d'AD) réalise un excellent isolement entre côtés primaire et secondaire. Pour détecter le sens des données sur les lignes USB (en vue de la commutation des tampons de sortie), le CI met en œuvre un mécanisme basé sur les paquets entrés. Le délai de propagation (temps de retard entre l'entrée et la sortie) est court malgré les hautes exigences d'isolement, il est comparable à celui d'un hub USB standard.

Schéma et alimentation

On peut trouver bizarre de citer spécialement l'alimentation dans ce titre, mais elle joue un rôle essentiel dans ce projet. En effet, le circuit ADuM3160 ne se

Liste des composants

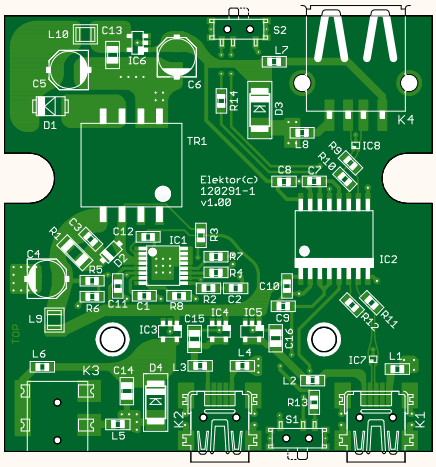
Résistances :
(1%/0,1 W, SMD 0603, sauf indication contraire)
R1 = 1 kΩ/1% 0,5 W (CMS 1206)
R2 = 33 kΩ
R3 = 80,6 kΩ
R4 = 6,04 Ω
R5 = 200 kΩ
R6 = 90,9 kΩ
R7,R8 = 28,7 kΩ
R9,R10,R11,R12 = 24 Ω
R13,R14 = 4,7 kΩ

Condensateurs :
C1,C2 = 10 nF/16 V X7R (CMS 0603)
C3 = 220 nF/50 V X5R (CMS 0603)
C4,C5,C6 = 120 µF/6,3 V (Nichicon RFS-0J121MCN1GS, type C)
C7,C8,C9,C10,C11,C12 = 100 nF/25 V X7R (CMS 0603)
C13,C14,C15,C16 = 10 µF/6,3 V X5R (CMS 0805)

Bobines :
L1,L2,L3,L4,L5,L6,L7,L8 = bobine sur ferrite 33 Ω à 100 MHz (CMS 0603, p.ex. Murata BLM18PG330SN1D)
L9,L10 = 1 µH/1200 mA (CMS 1007, p.ex. Taiyo Yuden CB2518T1R0M)

Semi-conducteurs :
D1 = diode Schottky PMEG3050BEP (NXP, SOD-128)
D2 = diode Schottky PMEG6010CEH (NXP, SOD-123f)
D3,D4 = zener 5,6 V/3 W
IC1 = régulateur à découpage *flyback* LT3575EFE#PBF (LT, TTSOP-16)
IC2 = isolateur de données USB ADUM3160BRWZ (AD, SOICW-16)
IC3 = interrupteur limiteur de courant 1,0 A, NCP380HSN10AAT1G (On Semiconductor, TSOP-5)
IC4,IC5,IC6 = interrupteur limiteur de courant 0,5 A, NCP380HSN05AAT1G (On Semiconductor, TSOP-5)
IC7,IC8 = protection électrostatique pour USB TPD2EUSB30ADRTR (TI, SOT-3)

Divers :
S1,S2 = inverseur à glissière (p.ex. C&K Components PCM12SMTR)
K1,K2 = connecteur USB mini-B pour montage CI (p.ex. On-Shore USB-M26FTR)
K3 = connecteur alimentation 1,35 mm (p.ex. CUI PJ-014DH-SMT)
K4 = connecteur USB A pour montage CI (p.ex. FCI 87583-2010BLF)
TR1 = transformateur 3:1, 25 µH (p.ex. Würth 750310471)
Coffret Hammond 1593KBK
2 vis autotaraudeuses à tête cruciforme, 4 x 6,4 mm, DIN7981
circuit imprimé 120291-1 ou
module assemblé, testé, prêt à l'emploi avec coffret 120291-91



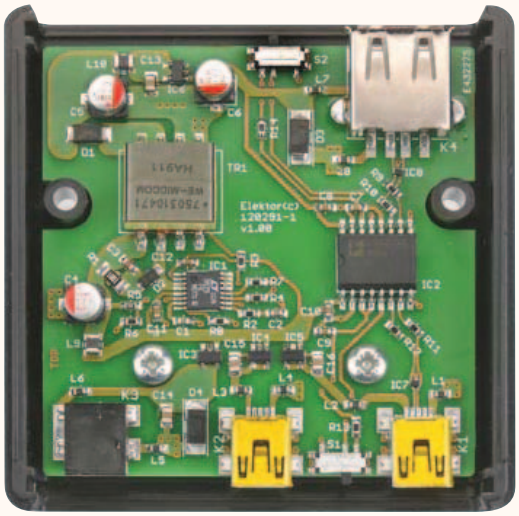


Figure 3 - Les dimensions du circuit imprimé de l'isolateur sont accordées à celles d'un coffret Hammond de type 1593KBK.

charge que de l'isolement des lignes de données, pas des lignes d'alimentation. En fait, chaque côté du CI doit être muni de son alimentation propre. Nous avons choisi d'installer sur le circuit imprimé (figure 2) une alimentation à découpage isolée. Elle se charge non seulement d'alimenter les deux côtés de l'isolateur, mais aussi de fournir du côté de la sortie le courant maximal autorisé pour l'appareil USB connecté (effectivement complètement isolé de l'entrée !).

Pour arriver à 500 mA sous 5 V en sortie, le choix s'est porté sur le régulateur indirect LT3575 de *Linear Technology* [2]. La particularité de ce circuit intégré est qu'il se passe de rétroaction par un optocoupleur. Il fonctionne en mode « critique » [*boundary-mode*, la limite en question se situe entre modes de conduction continue (fréquence de découpage variable) et discontinue (fréquence de découpage fixe) NdT] avec lecture de la tension de sortie à travers le couplage magnétique entre primaire et secondaire du transformateur. Le fonctionnement est le suivant :

quand le commutateur s'ouvre, la tension s'élève jusqu'au niveau :

- $V_{FLBK} = (V_{OUT} + V_F + I_{SEC} \cdot ESR) \cdot Nps$
- où V_F = tension sur la diode D1,
- I_{SEC} = courant à travers l'enroulement secondaire,
- ESR = impédance totale du côté secondaire,

- N_{PS} = rapport de transformation entre enroulements primaire et secondaire.

L'information tirée de là, combinée à une compensation de température, est suffisante pour réguler la sortie. La limitation induite est qu'un temps minimal d'ouverture du commutateur est nécessaire pour que le circuit intégré puisse traiter la rétro-action.
Le transformateur est un modèle tout fait de Würth, moulé pour montage en surface.
Les lignes de données sont protégées contre les surtensions par IC7 et IC8. C'est le circuit intégré à deux canaux TPD2EUSB30A de TI, destiné à la protection contre les décharges électrostatiques, qui a été choisi (il convient aussi pour les lignes de données USB3).
Les dimensions du circuit intégré sont si petites (1 x 1 mm) qu'il est possible de le caser entre les lignes de données sur le circuit imprimé.
Toutes les connexions d'alimentation sont protégées contre les surintensités à l'aide d'interrupteurs du point chaud (*high side*) de type NCP380 de *ON Semiconductor*. Ces CI contiennent un circuit de détection et de régulation, avec un MOSFET à canal P, qui limite le courant de sortie s'il dépasse une intensité déterminée. Deux types de NCP380 sont mis en œuvre : IC4, 5 et 6 limitent à 0,5 A, IC3 à 1 A. Ainsi toutes les entrées et sorties (USB comme alimentations) ont une protection optimale contre (presque) toutes les situations de surcharge possibles.

Les ports USB sont limités à 500 mA et la ligne 5 V est limitée à 1000 mA. Un appareil USB connecté dispose toujours de 500 mA. Cela ne correspond pas exactement à la norme USB officielle. Précisément, la limite devrait être située initialement à 100 mA, et elle ne devrait être portée à 500 mA que si l'appareil en a besoin et le demande à l'hôte. En pratique, il semble qu'on dispose toujours de 500 mA sur un port USB de PC, sans avoir besoin de les demander.
Pour maintenir faibles les ondulations au primaire et au secondaire du circuit à découpage, des filtres LC sont ajoutés des deux côtés (L9/C4 et L10/C3). Les courants d'ondulation qui traversent C4 et C5 sont très intenses, ce dont il faut tenir compte pour le choix des composants. Il va de soi que les condensateurs chimiques devront présenter une résistance série équivalente aussi faible que possible. Pour garder dans les limites la pointe à l'ouverture du côté primaire de l'alimentation, un réseau amortisseur constitué de R1, C3 et D2 est ajouté en parallèle sur le primaire.

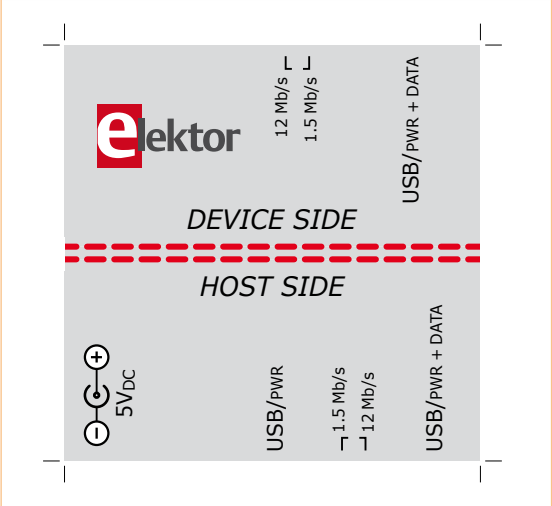


Figure 4 - Vous pouvez copier cette étiquette et la coller sur le capot du coffret.

Construction

Pour un montage avec autant de circuits intégrés spécialisés (tous en CMS), un circuit imprimé bien conçu est un *must*. La **figure 3** représente l'œuvre du labo Elektor. Naturellement il est disponible nu et vous pourrez vous y attaquer vous-mêmes. Mais comme un grand nombre de composants sont non seulement difficiles à souder à cause de leurs dimensions, mais également difficiles à trouver, Elektor propose aussi une carte terminée avec son coffret. Il ne vous restera qu'à pratiquer quelques ouvertures dans les côtés du coffret pour les connecteurs et commutateurs à glissière. Soudez ensuite pour la sécurité les deux languettes de K4 qui traversent la platine, avec un fer ordinaire, et vous pouvez fixer la platine dans le coffret avec les deux vis fournies.

La **figure 4** représente une étiquette que vous pouvez copier et coller sur le capot du coffret. Le dessin de l'étiquette est disponible aussi sur la page du site d'Elektor consacrée à ce projet [3].

Utilisation

L'utilisation de l'isolateur USB est très simple. Réglez tout d'abord la vitesse désirée avec les interrupteurs S1 et S2, qui doivent être **tous les deux dans la même position**. Normalement, un appareil USB fait savoir à l'hôte à quelle vitesse il fonctionne, au moyen d'une résistance de polarisation sur une des lignes de données. Ce n'est pas possible avec l'isolateur et il faut effectuer le réglage à la main, avec les interrupteurs (ce qui est le cas de la plupart des isolateurs USB du commerce).
Raccordez l'« hôte », un PC le plus souvent, au connecteur K1 (HOST SIDE : USB/PWR=DATA) et l'appareil USB à isoler à K4 (DEVICE SIDE : USB/PWR=DATA). Si l'appareil a besoin de plus de 300 mA, nous conseillons de raccorder un deuxième câble USB à K2 (USB/PWR) ou une alimentation stabilisée 5 V (minimum 500 mA) au connecteur d'alimentation K3 ($5V_{DC}$).
Nous déconseillons d'utiliser cet isolateur dans le cas où la différence de tension entre les côtés hôte et périphérique risque d'être supérieure à la tension du secteur. Il est déconseillé aussi d'appliquer longtemps une forte différence de potentiel entre les côtés hôte et périphérique, parce que l'isolement peut se dégrader avec le temps si les composants sont soumis en permanence à une forte tension.

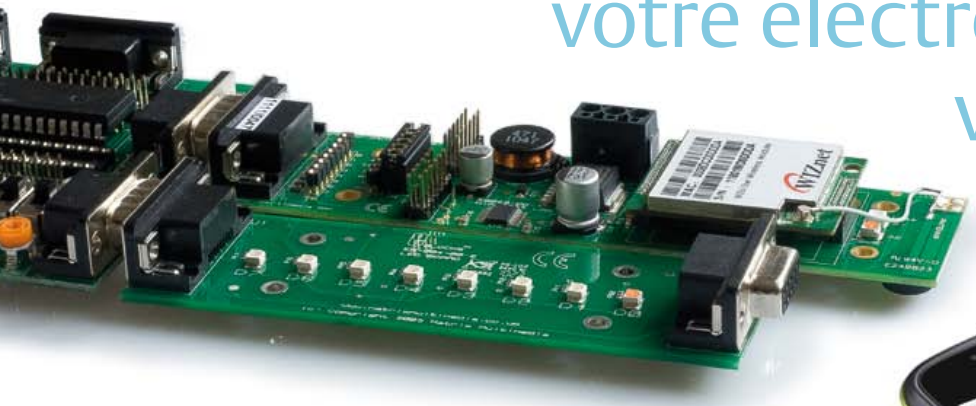
(120291 – version française : Jean-Paul Brodier)

Liens

[1] www.analog.com/static/imported-files/data_sheets/ADuM3160.pdf
[2] http://cds.linear.com/docs/Datasheet/3575f.pdf
[3] www.elektor.nl/120291

AndroLED

votre électronique pilotée via le WiFi sous Android



Voici une manière simple, peut-être la plus simple, pour piloter vos circuits à distance : **Android+WiFi+E-Block**. Pour mémoire, Android est un système d'exploitation pour téléphones et tablettes tactiles, WiFi un ensemble de protocoles de communication sans fil, et les E-blocks des modules électroniques prêts à l'emploi proposés par Elektor. Dans une série de deux articles, Android sera utilisé pour deux réalisations complètes : AndroLED et AndroCAR.

Nicolas Stouls, Benoit Pierret, et Yann Ricotti (INSA Lyon)

Le caractère innovateur de cet article et de celui qui suivra n'est pas de piloter à distance un dispositif électronique avec un téléphone Android. C'est la simplicité extrême de mise en œuvre qui est séduisante ici. En effet, le circuit électronique sera assemblé en quelques minutes seulement grâce aux E-Blocks, tandis que le choix du WiFi permet une programmation très simple coté Android. Enfin, l'utilisation conjointe d'un E-Block WiFi EB069 et du logiciel Flowcode [1] donne le résultat voulu en moins de 10 étapes de programmation. Les fichiers binaires et sources sont fournis [2] mais les explications qui suivent vont vous permettre de faire facilement votre propre version. Si vous suivez cet article pas à pas, vous serez aptes à attaquer ensuite des projets beaucoup plus complexes.

Des antécédents

Depuis 2008, l'arrivée des téléphones tactiles et des tablettes Android révolutionne la manière de concevoir nos montages électroniques. Le magazine Elektor accompagne cette révolution pour le plus grand bénéfice de ses lecteurs. En effet, dès le n° 396 de juin 2011, un article d'Elektor explique comment installer un système Android sur une carte Beagle Board [3]. Rapidement, ces tablettes deviennent le moyen idéal de créer une jolie interface entre l'homme et l'électronique. Un exemple est proposé dans le n° 402, page 30, de décembre 2011, qui permet de régler des prises de vues en automatique depuis une tablette vers une interface mécatronique montée sur un appareil photo. Le moyen de communication utilisé était le signal audio émis par les haut-parleurs de la tablette. Dans ce même numéro d'Elektor, page 52, sont anticipées les futures évolutions en termes d'interface homme-machine, mais également en termes de moyens de communication disponibles : Internet, WiFi, Bluetooth, USB... Quelques mois suffiront pour que cette dernière partie soit mise en lumière : dans le n° 404 de février 2012, l'article intitulé **AndroPOD** permet, grâce à son interface électronique USB maître, de connecter l'USB esclave d'un smartphone. Cette carte peut en outre se connecter en USB esclave à un USB Maître d'un PC (débogage) et à une sortie RS485 pour des échanges avec une autre carte électronique. Ce dernier dispositif, publié dans le n° 405 de mars 2012, page 54, permet l'échange de données entre une tablette et un PC qui simule une électronique de mesure à travers une interface HTML. Dans la poursuite de la conquête des moyens de communication offerts par Android, le n° 405 illustre en page 18 comment un smartphone Android peut transférer des données en Bluetooth, permettant à une électronique Arduino d'allumer des LED ou de commander une sortie analogique. En retour, cette carte électronique envoie des données à partir de ses capteurs vers le smartphone.

Dans le même esprit, nous vous proposons ici d'aborder la communication WiFi à partir de moyens de communication disponibles sur Android. L'objectif de cet article est de démontrer la simplicité de mise en œuvre d'une telle communication lorsque l'on utilise les outils adaptés. Dans le prochain article, nous mettrons en œuvre cette communication dans un exemple plus élaboré de pilotage d'un petit véhicule. Nous utiliserons alors l'accéléromètre, un de ces capteurs embarqués dans les téléphones tactiles.

Un cas d'école pour le WiFi

L'architecture proposée (**fig.1**) est un cas d'école pour la communication sans fil : programmation très simple et composants choisis pour faciliter l'interfaçage. Sur l'écran tactile d'une plate-forme Android (téléphone ou tablette tactile), il suffira d'appuyer sur un bouton pour allumer ou éteindre une LED sur un circuit électronique distant via une communication WiFi.

D'un côté nous avons un microcontrôleur PIC sur son E-Block EB006 connecté à deux cartes filles (**fig.2**) : l'E-Block EB069 WiFi et l'E-Block EB004 LED. Nous avons retenu le microcontrôleur 16F887 pour son port RS232 matériel, que nous utiliserons pour la communication avec la carte WiFi.

De l'autre, nous avons un terminal Android, supportant la communication WiFi.

Entre les deux, l'E-Block WiFi pour transformer en émission WiFi les données reçues sur la ligne RS232 et inversement. La liaison est bidirectionnelle, mais ici seul le sens Android vers E-Bock sera utilisé. La mise en œuvre est d'autant plus simple que l'E-Block WiFi est configurable en point d'accès (serveur de réseau). Comme le réseau n'est constitué que du terminal Android et de l'E-Block, il est inutile de se connecter sur un réseau existant (parfois délicat à configurer, surtout s'il est chiffré).

Pour le développement des quelques lignes de code embarquées sur le microcontrôleur, nous utiliserons Flowcode et Eclipse pour le développement du programme Android.

Les E-Blocks permettent de réaliser facilement le montage de la figure 2 : il suffit de brancher l'E-Block EB004 LED sur le port D et l'E-Block EB069 WiFi sur le port C. Le cavalier de l'E-Block WiFi doit être sur B et les borniers +14V et +5V de la carte WiFi doivent être reliés à la carte-mère.

Le montage est présenté avec un PIC16F887, mais vous pouvez, sans modification de câblage, opter pour un 16F877. Il est possible d'utiliser tout autre microcontrôleur avec un port RS232 en branchant l'E-Block WiFi sur les ports correspondants au RS232.

Principes d'un réseau WiFi

La communication WiFi est simple à mettre en œuvre, car elle est nativement supportée par Android, comme par la plupart des plate-formes informatiques récentes. Il n'y a donc pas à importer de bibliothèques comme avec le Bluetooth. Quelques lignes de programmation suffisent à envoyer des données.

Dans les réseaux modernes, les protocoles de communication les plus utilisés sont TCP/IP et UDP. Le lecteur intéressé par la différence consultera la page Wikipedia [4] traitant du sujet. Nous utiliserons UDP, plus simple à mettre en œuvre. Comme il convient aussi pour communiquer sur de grands réseaux, tels que l'internet, et avec des ordinateurs exécutant plusieurs applications simultanément, il est nécessaire d'identifier non seulement l'hôte auquel on parle, mais aussi l'application avec laquelle on veut communiquer. Nous parlerons respectivement d'**adresse** et de **numéro de port**.

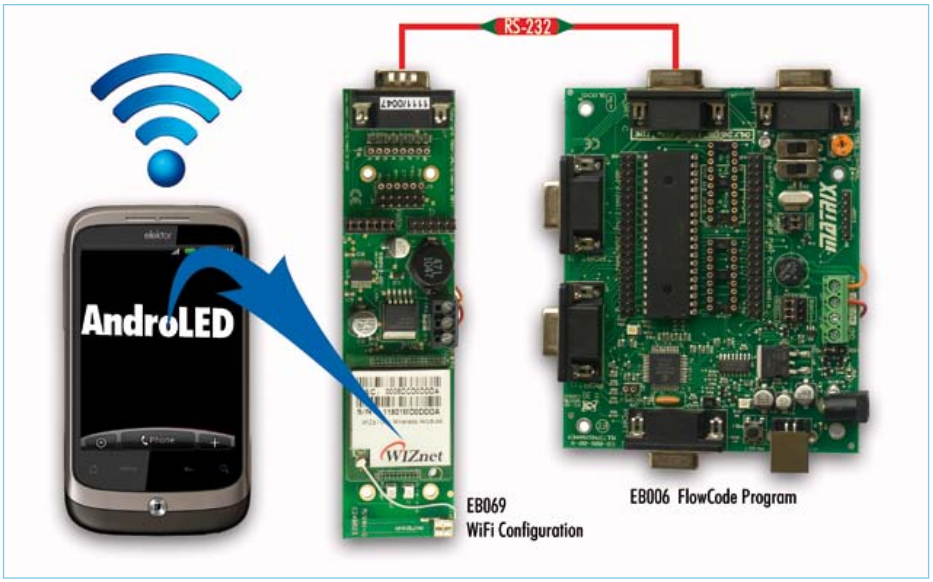


Figure 1 : L'architecture proposée comporte trois composants configurables : un terminal Android, une carte WiFi EB069 et une carte-mère EB006.

Dans cette section, nous traitons d'abord la configuration de l'E-Block WiFi, qui sera le point d'accès du réseau, avant de décrire la programmation d'une routine Android permettant de s'y connecter.

Point de vue du prototype : configuration de l'E-Block WiFi EB069

L'E-Block WiFi cumule différentes fonctions qui facilitent son utilisation :

- Serveur DHCP : Les adresses IP sont distribuées automatiquement aux clients qui se connectent sur le réseau local dont la tête de réseau est l'E-Block.
- Interface web : Permet de configurer très facilement l'E-Block
- Passerelle WiFi ↔ RS232 : les données

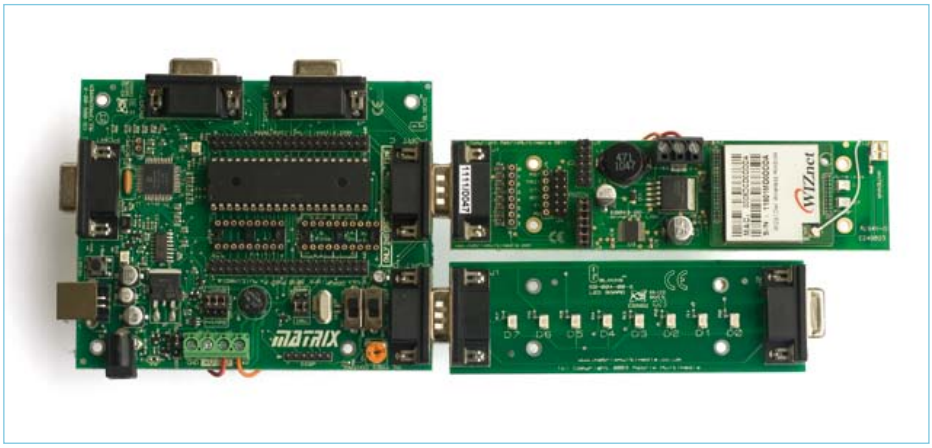


Figure 2 : Montage de la partie électronique à base d'E-Blocks. Bien respecter la position des cavaliers visibles sur cette image.

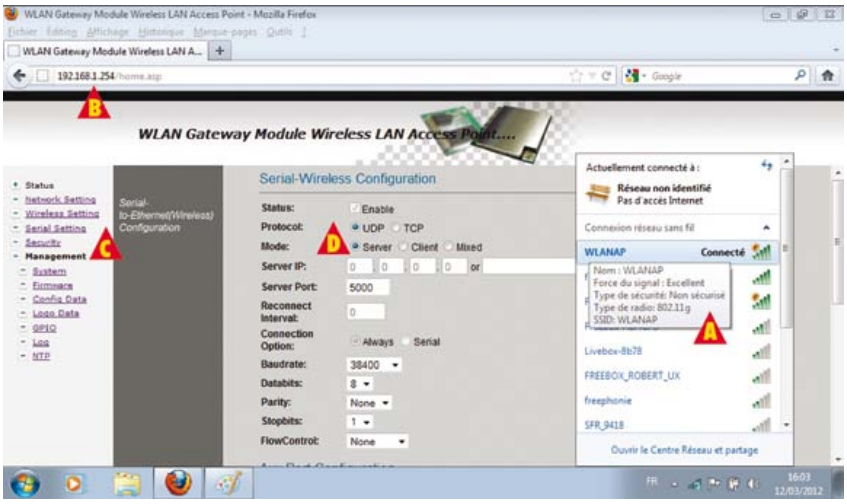


Figure 3 : Configuration de la carte WiFi : **A** = connexion au réseau WiFi, **B** = aller à 192.168.1.254, **C** = cliquer sur *Serial settings*, **D** = sélectionner UDP.

circulant sur le WiFi sont simplement renvoyées en RS232 en émission et en réception.

Afin de maîtriser totalement sa configuration, nous commençons par initialiser notre E-Block WiFi (paramètres d'usine). Pour cela, mettre le cavalier sur PATCH et relier la broche *reset* (RST) au +5 V pendant plus de 3 s. Après cette manipulation, le nom de votre réseau est WLAN et l'adresse est 192.168.1.254.

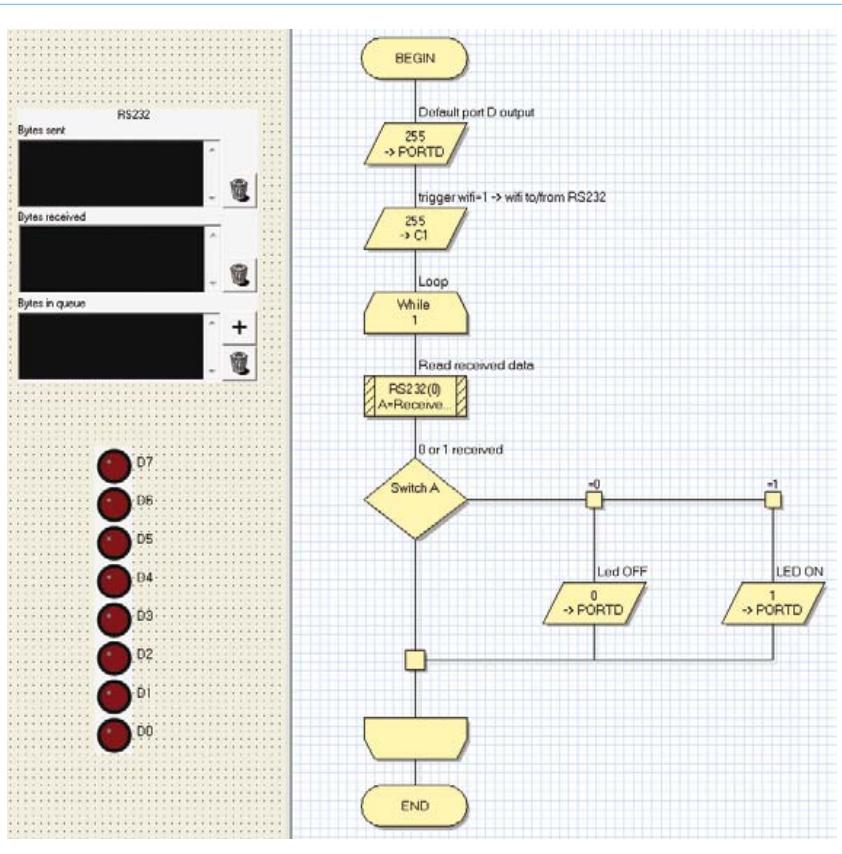


Figure 4 : Programme Flowcode.

La configuration « Paramètres d'usine » est très proche de nos besoins :

- Adresse IP:192.168.1.254
- Port 5000
- Mode Serveur
- Débit RS232 : 38400 bauds

Le seul paramètre qui ne convient pas par défaut est le protocole qui doit être modifié de TCP en UDP. Pour cela (**fig. 3**), connecter un ordinateur au réseau WiFi WLAN. Ensuite, utiliser un navigateur Internet pour se connecter à l'adresse http://192.168.1.254. Le nom d'utilisateur et le mot de passe par défaut sont *admin* et *admin*.
Dans l'onglet « *serial settings* », cocher le bouton UDP. Vérifier que le débit est bien de 38400 bauds. Il sera possible, dès lors, de se connecter sur le réseau WiFi de la carte. Tout message arrivant en WiFi à l'adresse 192.168.1.254 port 5000 sera retransmis sur la ligne RS232 à 38400 bauds. Réciproquement, tout octet reçu de la ligne RS232 sera renvoyé depuis l'adresse 192.168.1.254 port 5000 vers l'adresse et le port du client ayant envoyé un message. Toute la configuration du réseau est donc faite dans cette interface web. Côté microcontrôleur, il suffira d'envoyer et de recevoir les données sur le port RS232, d'où une programmation très simple.

Point de vue Android : programmation d'une émission de message

En Java, et donc sous Android, le principe d'une communication réseau est que l'on ne choisit pas le médium à utiliser pour une communication (WiFi, 3G, Bluetooth, USB ...). C'est la plate-forme qui le choisira en fonction des moyens disponibles pour joindre l'adresse demandée. Ainsi, pour réaliser un programme Java communicant, il suffit de définir *à qui* l'on veut parler (quel hôte) et *de quelle manière* (en utilisant quel protocole).
Depuis un programme Java, le réseau est matérialisé par une « socket ». Une fois initialisé, un objet de type *socket* fournit un accès à tout le réseau. Il permet d'envoyer un message qui aura été forgé à partir des données à envoyer, de l'adresse du destinataire et de son port. Le **listing 1** est l'exemple d'une méthode simple, émettant

un octet sur le réseau, vers un hôte *address* sur son *port* prédéfini et en utilisant le socket *UDPSocket*.

Programmation du microcontrôleur

Grâce à l'architecture choisie, le programme Flowcode est extrêmement simple *(**fig. 4**)* : seulement 7 cases dans l'organigramme.
Si la ligne RS232 reçoit 1, la LED est allumée et éteinte si c'est 0. Il est facile de modifier le programme pour piloter toutes les LEDs et pourquoi pas d'autres dispositifs.
Il faut juste prendre garde de configurer la même vitesse de communication du côté microcontrôleur et du côté E-Block WiFi. Comme pour la configuration de la carte WiFi, choisir 38400 bauds dans *RS232 properties*.
Charger le programme Flowcode sur le microcontrôleur.

Réalisation du programme Android

L'installation de l'ADT (*Android developpement tool chain*) a déjà été expliquée dans les articles précédents d'Elektor (mars 2012 par exemple). Pour les piqûres de rappel, visitez le site d'*Android developpers* [5].
La réalisation de ce programme se décompose en deux parties : la mise en place du projet Eclipse et la réalisation du code Java.

Création d'un projet Android et de son interface graphique

Pour écrire notre programme Android, nous créons sous Eclipse un projet « Android Project » (File/New/Project.../Andoid/Android Project). Dans notre exemple, nous nommerons notre projet « AndroLed », nous choisirons la cible « Android 1.6 » (Car la compatibilité ascendante fait que notre programme sera utilisable sur la plupart des plateformes) et nous nommerons le *package* « elektor.androLed ».
À ce stade, la structure du projet est créée sur la base d'un programme « bonjour tout le monde (*hello world)* ». Pour notre démonstration, une interface graphique minimaliste fera l'affaire, puisqu'il suffit d'afficher un unique bouton. Ainsi, dans le fichier « main.xml » décrivant notre interface graphique par défaut et se trouvant dans « res/layout », nous ajouterons simplement un ToggleButton. Pour pouvoir parler de ce bouton depuis le code Java, nous le nommerons par exemple « onOffBtn ».
Enfin, pour qu'un programme Android ait le droit d'utiliser le réseau, il faut l'autorisation explicite de l'utilisateur. Il faut donc ajouter une permission dans le descripteur d'application (le fichier AndroidManifest.xml). La permission à rajouter est INTERNET. Le **listing 2** montre la ligne à ajouter dans ce fichier.

Listing 1: Envoi d'un message via un socket

```
// Address of the E-Block WiFi card
private InetAddress address;

// Port number of the WiFi Card
private int port;

// Network access variable
private DatagramSocket UDPSocket;

/** Méthode qui envoie un unique octet (value) via le réseau (UDPSocket) à
  * l'hôte address, sur le port défini. */
public void sendOneByte(byte value) {
    try {
        /* Building the data vector from the getted value */
        byte[] data = {value};

        /* Data packaging */
        DatagramPacket packet;
        packet = new DatagramPacket(data, data.length, address, port);

        /* Sending packet */
        UDPSocket.send(packet);
    } catch (Exception e) {
        Toast.makeText(this,"Sending error", Toast.LENGTH_LONG).show();
    }
}
```

Listing 2 : Configuration des droits d'accès au réseau pour le fichier AndroidManifest.xml

```
<uses-permission android:name=»android.permission.INTERNET» />
```

Code Java de l'AndroLed

Le projet est défini, il faut écrire maintenant quelques lignes de Java qui établiront le lien entre le bouton et l'émission d'un paquet réseau. L'intégralité de la classe *AndroLedActivity* se trouve dans le **listing 3**. Nous allons donc essayer de comprendre la démarche ayant donné lieu à ce programme plutôt que de le commenter.
Notre objectif est d'envoyer un octet contenant la valeur 1 ou la valeur 0 à chaque pression du bouton présent sur l'interface graphique. Pour la partie émission de données, nous avons déjà décrit comment faire (listing 1). Il nous suffit donc d'initialiser les variables du réseau et d'associer au bouton une action consistant à appeler la méthode *sendOneByte*.
Lors de la création du projet, la méthode *onCreate* est produite automatiquement. C'est cette méthode qui est exécutée lors du lancement de l'application et c'est dans cette méthode que nous initialisons le réseau et que nous mettons en place l'écouteur d'événements sur le bouton.
Tout d'abord, initialiser les variables réseau consiste à demander (ligne 14) au système un accès au réseau au moyen de l'instruction **new** DatagramSocket(). Si le système refuse cet accès, une erreur est renvoyée et le message de la ligne 18 s'affiche. Ensuite, il faut définir l'adresse et le port de notre prototype (lignes 15 et 16). Dans cet exemple, nous avons choisi d'utiliser l'adresse IP par défaut d'une carte E-Block WiFi qui vient d'être ré-initialisée.
Notre bouton doit être associé à une action. Pour ce faire, commençons par demander au système de nous transmettre un pointeur vers l'objet créé dans l'interface graphique et portant l'identifiant *onOffBtn* (ligne 22). Enfin, il suffit d'associer à ce

Listing 3 : Code source complet (sauf imports) de la classe AndroLedActivity

```
public class AndroLedActivity extends Activity {
  private int port;
  private InetAddress address;
  private DatagramSocket UDPSocket;

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);

      // Network initialization
      try {
          UDPSocket = new DatagramSocket();
          address = InetAddress.getByName("192.168.1.254");
          port = 5000;
      } catch (IOException e) {
         Toast.makeText(this,"Network error",Toast.LENGTH_LONG).show();
      }

      // Button management
      ToggleButton btn = (ToggleButton) findViewById(R.id.onOffBtn);
      btn.setOnClickListener(new OnClickListener() {
          public void onClick(View v) {
             if (((ToggleButton)v).isChecked()) sendOneByte((byte)1);
              else sendOneByte((byte)0);
              return true;
          }
      });
  }

  /** Méthode qui envoie un unique octet (value) via le réseau
    * (UDPSocket) à l'hôte address, sur le port défini. */
  public void sendOneByte(byte value) {
      try {
          // Building data
          byte[] data = {value};
          DatagramPacket packet;
         packet = new DatagramPacket(data, data.length, address, port);

          // Sending packet
          UDPSocket.send(packet);
      } catch (Exception e) {
         Toast.makeText(this,"Sending error", Toast.LENGTH_LONG).show();
      }
  }
}
```

bouton une action qui lit l'état du bouton (pressé ou relâché) et qui appelle la méthode *sendOneByte* avec la valeur 1 ou 0, en fonction de cet état.
Une fois le programme fini, il ne reste plus qu'à le charger sur le périphérique Android de votre choix. Si les pilotes sont correctement installés et le périphérique correctement configuré, il vous suffit de le connecter à l'ordinateur par USB et de cliquer sur l'icône RUN dans Eclipse pour exécuter l'application. Sinon, pour pérenniser cette installation, choisir d'« exporter » le projet. Cela produira un fichier avec l'extension .apk. En ouvrant ce fichier depuis Android, l'application sera installée.

Et voilà !

Si vous avez suivi à la lettre les étapes de ce guide, vous devriez avoir compris les principes d'une communication WiFi UDP et être en mesure, avec des E-Blocks et un appareil Android, de la mettre en œuvre sans encombre. Qui plus est, vous disposez également d'un prototype fonctionnel, ainsi que d'une application Android capable de se connecter sur le prototype et de commander l'allumage d'une LED en utilisant le réseau WiFi.

Dans le prochain article, la programmation du dispositif sera encore plus simple pour un résultat encore plus séduisant : il s'appelle *AndroCAR*, et c'est un véhicule télécommandé par les accéléromètres du téléphone. Nous verrons aussi comment vérifier le bon fonctionnement des deux mailles de votre réseau : le terminal Android et l'E-Block EB069. Ne ratez pas ce rendez-vous !

120364-I

Liens

[1] www.matrixmultimedia.com/

[2] www.elektor.fr/120364

[3] http://beagleboard.org/

[4] http://fr.wikipedia.org/wiki/Suite_des_protocoles_Internet

[5] http://developer.android.com/tools/

Liste des composants

E-Blocks

EB069 : E-Block WiFi
EB006 : E-Block Multiprogrammateur pour PIC
EB004 : E-Block LED
Alimentation 14V

Semi-conducteurs

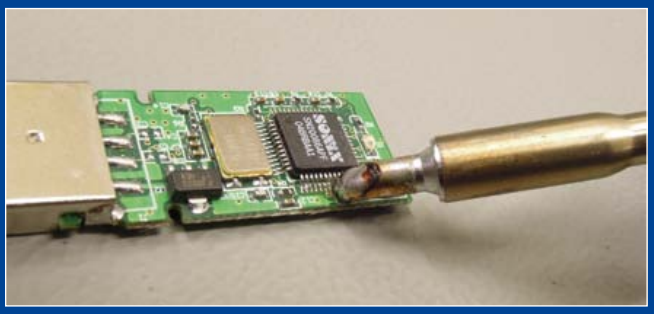
16F887

soudez les CMS sans faire de vagues

Thijs Beckers (Elektor)

La soudure des CI CMS continue apparemment d'effrayer beaucoup d'électroniciens inexpérimentés, malgré les efforts des anciens et autres experts pour les rassurer. « C'est vraiment trop petit », « Je n'ai pas les bon outils », « Ils ont tous fini dans l'aspirateur de maman » et cetera. De bien mauvaises excuses. Voici un truc de plus pour souder les CI aux pattes très rapprochées.
Comme vous pouvez le voir sur les photos, le boîtier de ce CI n'est pas le plus petit qui soit — il s'agit d'un SN20086APF de *Sonix* en boîtier LQFP à 48 pattes, utilisé sur une clé USB plutôt ancienne de 128 Mo. Ses pattes espacées de 0,5 mm en font un parfait candidat pour vous montrer une nouvelle technique. Ce que j'espère ici, c'est vous faire accéder à cette méthode de soudure des LQFP à l'aide d'une panne plutôt grosse. Tout d'abord, assurez-vous que le CI est correctement positionné et bien immobilisé, par exemple en soudant deux pattes diagonalement opposées. Puis, appliquez un peu de flux de soudure le long des pattes. Utilisez maintenant une panne en forme de ciseau — type mini-vague — garnie d'un peu de soudure. Déplacez-la lentement le long des pattes à souder, partie creuse vers le haut et inclinée à environ 30° ; inspirez-vous des photos. Admirez la soudure qui s'écoule du creux de la panne vers les pattes et pastilles ; elle est pas belle la vie ? Oui, il vous faudra pratiquer un peu, mais rien d'insurmontable. Après quelques essais, vous trouverez vite la bonne vitesse et la quantité de soudure adéquate et vous chercherez alors de plus petits CI à cannibaliser pour mettre à l'épreuve vos nouveaux talents.
Si les CMS, ce n'est vraiment pas votre dada et que les 0,5 mm vous effraient pour un premier essai, essayez notre détecteur d'e-smog à large bande en kit, le TAPIR, il ne comporte que quelques CMS. C'est un projet amusant et notre site regorge de documentation à son sujet : www.elektor.com/120354.

(120234 – version française : Kévin PETIT)



USB : courant illimité !?

Raymond Vermeulen (Elektor)

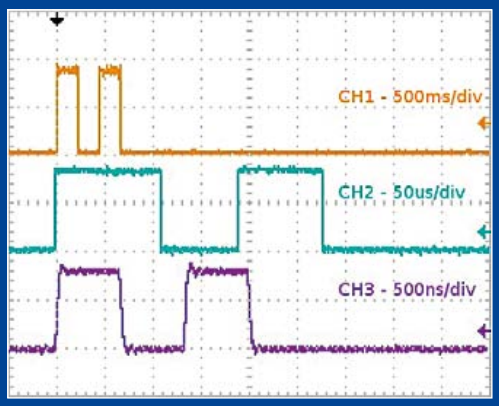
Après avoir travaillé sur un certain nombre de projets dotés d'USB, j'ai fini par me dire que quelque chose ne tournait pas rond au niveau de l'alimentation. J'ai commencé à mettre en doute l'idée, qui fait pourtant consensus, selon laquelle le courant maximal fourni n'était que de 100 mA jusqu'à ce que le périphérique se déclare gros consommateur pour bénéficier du maximum de 500 mA.
Armé de l'ampèremètre pour port USB publié dans le numéro estival de cette année et d'une simple charge constituée de deux résistances 22 Ω/10 W en parallèle, j'ai effectué quelques mesures sur plusieurs PC du labo. Elles sont sans appel : le courant n'est apparemment pas limité — du moins pas à 100 mA !
Supposons maintenant qu'il n'y ait pas de limitation du courant pour les périphériques qui ne communiquent pas sur le bus — c'est le moins qu'on puisse dire au sujet de nos résistances. Pour tenter de prouver que ce n'est pas le cas, j'ai relié un périphérique (un projet avec un ATmega32, qui sera bientôt publié) qui se déclare comme un périphérique de faible puissance et négocie donc 100 mA. Je rajoute une charge (résistance) en parallèle sur les lignes d'alimentation du bus USB et que se passe-t-il : les 500 mA sont disponibles !?
De ces observations je ne peux que conclure que sur leurs bus USB les PC, portables ou non, débitent les 500 mA sans discrimination. Je suppute que la limitation de l'intensité à 100 mA n'avait cours qu'au début de l'USB, dans les années 90, et qu'il n'y a plus de raison pour limiter le courant disponible, peut-être grâce aux progrès réalisés sur les composants.
Si vous avez des informations là-dessus ou une meilleure explication, n'hésitez pas à nous mettre au courant (de préférence à r.vermeulen@elektor.nl), nous en ferons bien évidemment profiter la communauté.

(120436 – version française : Kévin PETIT)

carte Linux Elektor : « vite un GPIO ! »

François-Xavier Maurille (Elektor)

J'étais en plein développement d'un projet musical basé sur la carte Linux Elektor, quand je me suis aperçu qu'il me fallait un CNA SPI. Rien de bien compliqué, sauf qu'il me fallait utiliser des mots de 24 bits, non pris en charge par la cellule SPI de la puce. Je devais donc gérer la sortie de sélection de puce (*Chip Select*, CS) avec une patte d'E/S (GPIO) ; or, l'implantation de cette solution dans le système d'exploitation s'est révélée difficile : le temps de réponse après l'envoi d'une commande était loin d'être négligeable, sans doute à cause des nombreux et complexes échanges entre l'application et le matériel.



L'illustration donne le résultat de mes tentatives pour accélérer la séquence '1010' sur CS. J'ai commencé par essayer d'utiliser le système de fichiers de Linux et la fonction C *fprintf*. Le résultat, avec des commandes du style `fprintf(«/sys/class/gpio/gpio11/value», «%d», state)` est visible sur la voie 2. J'ai pu descendre jusqu'à un temps de réponse de 90 µs ; bien trop lent pour mon application. J'ai ensuite essayé de toucher à la GPIO avec la commande shell *echo*. Ici, c'est la voie 1 qui représente le résultat, les commandes utilisées ressemblent à `system(«echo 1 > /sys/class/gpio/gpio11/value»)`. Pas d'amélioration. En fait, c'est même bien pire : environ 200 ms avant que CS ne change d'état. Enfin, j'ai essayé d'accéder directement aux registres GPIO du microcontrôleur. C'est beaucoup plus difficile que d'utiliser le système de fichiers, mais le résultat est autrement plus rapide. Avec les conseils glanés sur http://forum.gnublin.org, j'ai pu écrire quelques fonctions pour commander l'état des GPIO en accédant directement aux registres MODEx, rendus accessibles à mon application grâce à un appel à la fonction *mmap()*. C'est la voie 3 qui présente les résultats que l'on peut obtenir avec des accès par ce genre de commandes : `*(unsigned int *)(ptr + GPIO_OFFSET + GPIO_MODE0) = 1 << nGPIO;`. Le temps de réponse est ici d'environ 700 ns ; j'ai enfin le sourire.

Quelques remarques :

- - Soyez vigilant : le **4**e bit de MODEx correspond au GPIO**4**, alors que le **5**e correspond au GPIO**11** (CS).
- - Votre carte Linux, vous permettra de « travailler dans le futur », lorsque la date gérée par la carte n'est pas correctement synchronisée avec celle de votre PC. J'ai reçu l'avertissement suivant : *"make: Warning: File 'GPIO.c' has modification time 52 s in the future"*.

(120457 – version française : Kévin PETIT)

script de correction de carte SD

Francois-Xavier Maurille (Elektor)

La carte Linux Elektor (série d'articles lancée dans le numéro de mai 2012) permet aux débutants d'accéder à Linux. Le matériel fonctionne à la perfection, mais le logiciel pose parfois quelques problèmes. Ceux d'entre vous qui l'ont utilisée se sont peut-être déjà retrouvés nez-à-nez avec un message d'erreur laissant croire que la carte SD était corrompue : `« EXT2-fs (mmcblk0p1): error: ext2_lookup: deleted inode referenced: 694962 »`.

Pas de panique, la solution se trouve dans ce numéro d'Elektor. Si vous avez la flemme de recopier les commandes, ou n'êtes pas sûr de vous, il vous suffit de télécharger un bout de logiciel qui s'en chargera pour vous. Mon petit script en *bash* tourne sur un PC Linux. Il lance exactement les mêmes commandes que celles de l'encart de l'article. La différence, c'est que tout est automatique. Il cherchera d'abord le nom du périphérique corrompu à l'aide des commandes `grep` et `sed`, le dé-montera puis lancera la commande `e2fsck`.

Il corrigera votre carte SD sans que vous ayez à (trop) vous casser la tête. Vous n'aurez qu'à suivre les instructions affichées à l'écran et à répondre en appuyant sur « Entrée » ou « y ». Voici le mode d'emploi :

Après avoir téléchargé le script [1], ouvrez le répertoire contenant le fichier, puis extrayez-le (clic droit puis *Extraire Ici* la plupart du temps). Pour pouvoir être lancé, le script devra être rendu exécutable. Ouvrez un invite de commande, déplacez-vous dans le dossier contenant le script et tapez la commande suivante : *sudo chmod 777 correctSD.sh*. Vous pouvez maintenant lancer le script en tapant *./correctSD.sh*. Laissez-le ensuite s'occuper de votre carte SD corrompue. Un petit fichier texte (README) vous fournira des informations complémentaires sur la procédure.

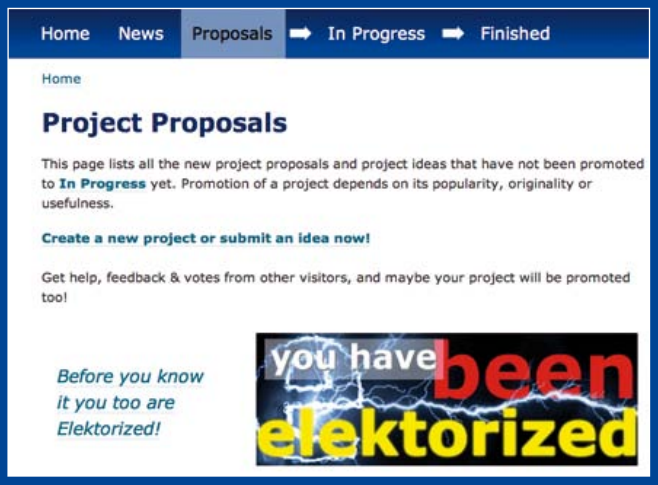
Ce n'est bien sûr pas une raison pour appuyer sur le bouton RESET de la carte à tout bout de champ…

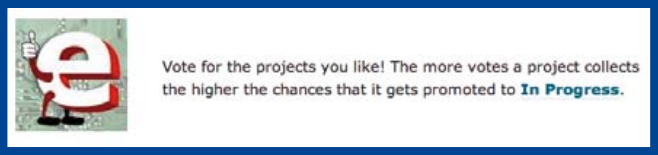
(120443 – version française : Kévin PETIT)

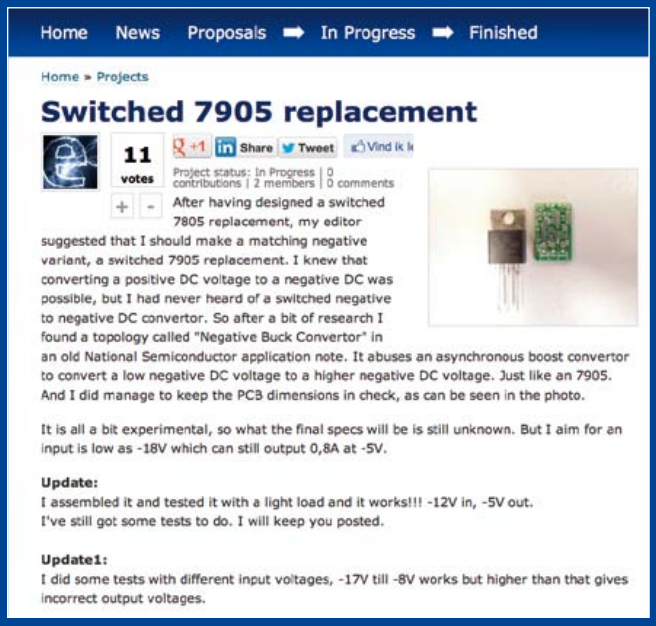
Liens

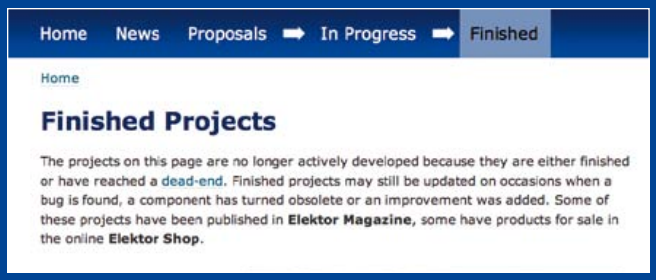
[1] www.elektor.fr/120026

Elektor-projects.com 4U2









« *Elektor Labs 4U2* », autrement dit « *l'Elektor de demain, c'est aussi votre affaire* ». Si la langue anglaise en usage sur ce nouveau site vous embarrasse, allez-y en français, et vous verrez bien ce qui se passera. L'essentiel est de donner corps à la communauté d'idées et de projets lancée et animée par Clemens Valens. Participez au labo virtuel d'Elektor : vos idées, vos projets, vos rêves y sont les bienvenus. Les propositions d'articles, jusqu'ici adressées en catimini à la rédaction d'Elektor, passeront désormais par elektor-projects.com. Les auteurs, potentiels ou aguerris, sont invités à y publier leurs propositions, pour que la communauté du labo virtuel d'Elektor les accueille par ses suffrages, en infléchisse éventuellement l'évolution et qu'ensuite un choix mûrement réfléchi aboutisse dans le magazine à la publication des idées les plus intéressantes ou les plus demandées.

Même le propre laboratoire d'Elektor, autour duquel votre magazine s'organise depuis plus de 30 ans en une formule éditoriale unique au monde, fonctionnera désormais de cette manière. La tradition de sérieux, de qualité et de fiabilité n'est pas remise en question, mais elle sera conjuguée au potentiel illimité offert par la virtualisation et plus généralement la communication directe en ligne.

Suivez par exemple dans la section *In Progress* l'évolution du projet *Switched 7905 Replacement* dont la publication est prévue dans l'édition d'octobre 2012.

(120484)

En mode de lecture, tous nos lecteurs sont les bienvenus sur Elektor Projects. En mode d'écriture, seuls les membres de la communauté Elektor Plus sont admis. Si vous êtes abonné à la formule Elektor Plus, vous pouvez vous identifier en tant que tel sur Elektor Projects.

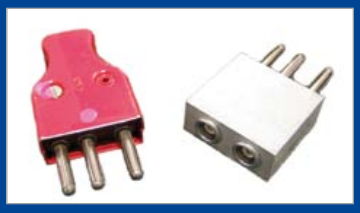
histoire(s) de prises 2.0

Thijs Beckers (Elektor)

Présentée dans le numéro de mars 2012, une sélection hétéroclite de connecteurs, en provenance de la riche collection *rétro* de notre collègue Jan Buiting, avait déclenché chez nos lecteurs de nombreuses réactions intéressées. Suite à cette publication, nous avons reçu des descriptions copieuses et des photos des connecteurs les plus étranges ; la plupart méritent le coup d'œil et même quelques commentaires.

Le premier lot montre la collection de prises *Hirschmann* appartenant à Klaus Rohwer, dont certaines datent probablement des années 50. Les quatre premières sont des prises d'alim, la grosse est un « OBNI », Objet Branchable Non Identifié. Nous pensons qu'elle faisait partie d'un télex ; l'un de mes collègues du labo d'Elektor se souvient de quelque chose de similaire, vu sur un vieux standard téléphonique dans un ancien centre de commande de l'armée.

Le deuxième lot contient divers connecteurs audio utilisés autrefois par le WDR, c'est-à-dire Radio Cologne. Un lecteur d'Elektor, W. Richter, se souvient de les y avoir utilisés dans les années 50, avec une gamme infiniment variée d'adaptateurs. Il raconte avec humour : « Des adaptateurs, je crois qu'on a dû en confectionner pour à peu près tout ce qui existait de connecteurs à cette époque, sauf peut-être des… *Gardena*."



Ces connecteurs appartiennent au « Pansanitor », qui date de 1928 (nous vous le présenterons le mois prochain dans notre rubrique *Rétronique*). Si l'on en croit leur fier propriétaire, M. Butte, un autre lecteur allemand, ces connecteurs sont faits de porcelaine vernissée. Il s'agit sans doute là des plus vieilles prises parmi les photos que nous avons reçues.

Cette petite collection a été photographiée par W. Haas. Admirez pour commencer le connecteur d'antenne LMK en haut à gauche, et remarquez ses broches orientées l'une verticalement et l'autre horizontalement. À côté se trouve un connecteur pour antenne VHF, prévu pour une ligne bifilaire d'impédance 240 Ω. Viennent ensuite deux prises VHF symétriques pour la bande III (240 Ω) dotées de broches horizontales. À côté se trouvent deux connecteurs UHF rouges, symétriques, pour les bandes IV/V (240 Ω) avec des broches verticales. La prise en bas à droite possède des broches de section carrée, pour la VHF et l'UHF. Les autres sont des prises adaptatrices diverses, banane vers une broche ; la bleue est prévue pour accueillir un fil serré à l'aide d'une vis.

P. van de Meerendonk nous a envoyé de Hollande la photo d'un autre OBNI qui ressemble à un support pour tube, mais n'en est pas un. Tout ce que nous y voyons c'est une sorte de prise en bakélite avec huit broches disposées en octogone. Utilisation inconnue.

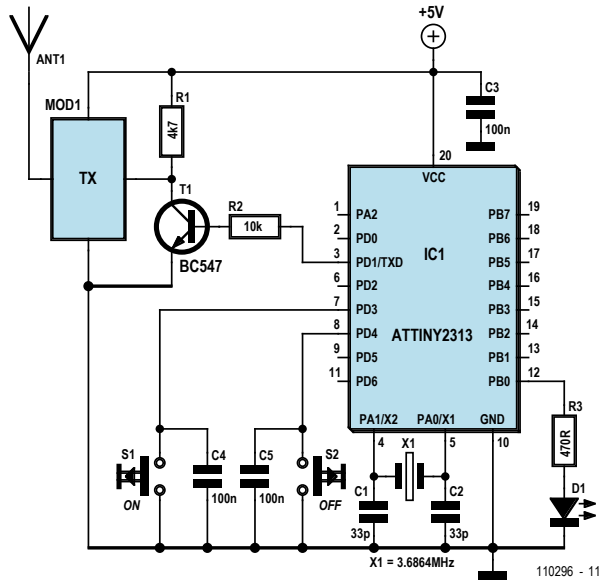
Outre les nombreuses photos reçues de nos lecteurs, parfois de connecteurs encore bien plus vieux que ceux que nous montrons ici, le contenu du premier article a aussi suscité d'intéressantes remarques, p. ex. sur le connecteur nº 10 autour duquel un consensus s'est formé : il s'agirait d'un connecteur d'antenne équilibré (240 Ω) pour radios, compatible avec des trous de 3 mm pour la VHF et l'UHF, et de 4 mm pour la radio FM en VHF.

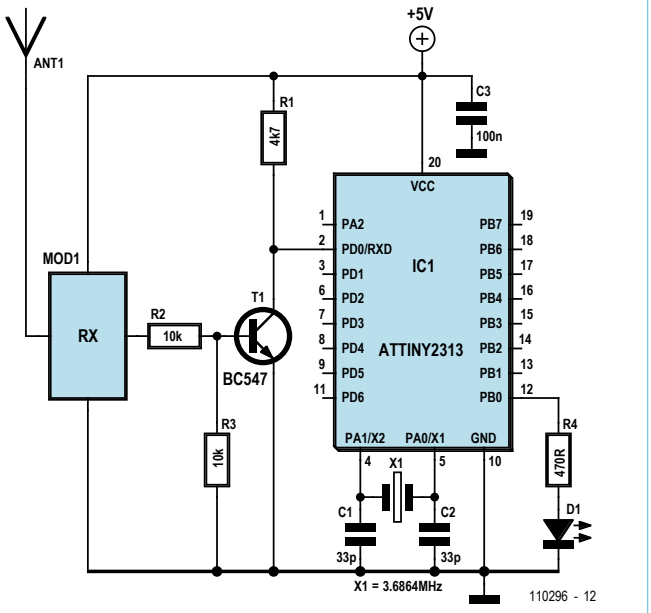
Le connecteur nº 11 serait son successeur. Polarisé, il permettrait de relier une antenne AM ; signal verticalement et masse horizontalement.

On nous a suggéré que les connecteurs nº 6 ne servaient à relier des haut-parleurs que dans leur version bakélite ; le branchement direct des HP à haute impédance sur l'anode se traduisait donc par de dangereux potentiels sur les câbles et les connecteurs.

(120303 – version française : Kévin PETIT)

l'ATtiny sans fil





Jurgen Stannieder (Allemagne)

Y a-t-il chez vous des appareils que vous voudriez commander à distance ? Voici comment faire avec un microcontrôleur ATtiny et un jeu de modules émetteur & récepteur de *Conrad*. Tel quel, le projet est simple ; vous l'étofferez selon vos besoins.
Pour en montrer le fonctionnement, j'ai symbolisé l'appareil par une LED côté récepteur. Sur l'émetteur, deux boutons-poussoirs S1 et S2, dans le schéma de gauche, servent à l'allumer et l'éteindre. Quand on appuie sur l'un d'eux, l'entrée correspondante, PD3 ou PD4, est mise à la masse. Le logiciel reconnaît le signal et le transforme en message de commande. Dans notre exemple, le message est un E pour l'**a**llumage et un A pour l'**e**xtinction (en allemand, *ein* = marche et *aus* = arrêt). L'appel est alors passé à la sortie PD1 (TXD) de l'ATtiny, inversé par T1 et finalement acheminé au module émetteur. S'il faut un inverseur, c'est que la sortie TXD est normalement au niveau haut, même quand il n'y a pas de message à envoyer. Comme le module émetteur est toujours en fonctionnement quand un signal est présent à son entrée, l'inverseur empêche le module d'émettre en permanence quand il n'y a rien à transmettre. Asservie par le quartz X1, la fréquence d'horloge de 3,6864 MHz du microcontrôleur lui permet de tenir un débit binaire de 9600 à 76500 bauds. Ici, il travaille à 9600 bauds parce que les modules radios sont plus fiables à cette vitesse. La LED D1 avec résistance série sert uniquement de témoin, pour savoir si le µC réagit convenablement. Quand on appuie sur l'un des deux boutons, elle s'éteint.

On retrouve sur le récepteur, dans le schéma de droite, un microcontrôleur tout pareil qui obtient du module de réception le signal porteur de l'information émise pour la traiter. Comme le signal émis a été inversé, le transistor T1 le remet à l'endroit. Le programme observe le signal sur l'entrée PD0 (RXD) et chaque fois qu'il reconnaît un E ou un A, il lui fait commuter la LED dans le sens prescrit. Ici aussi, le signal d'horloge provient d'un quartz X1 à la fréquence de 3,6864 MHz.

J'ai utilisé l'ensemble émetteur/récepteur radio 433 MHz AM de Conrad [1]. Leurs antennes sont dimensionnées d'usine et ne doivent pas être modifiées. Pour le reste, la construction est relativement simple, les circuits sont montés chacun sur une moitié de platine Elex-1 comme on le voit sur les photos. J'ai ainsi établi des liaisons sans difficulté sur une douzaine de mètres à travers la maison.

(110296 – version française : Robert Grignard)

[1] www.conrad.fr
référence produit 130428

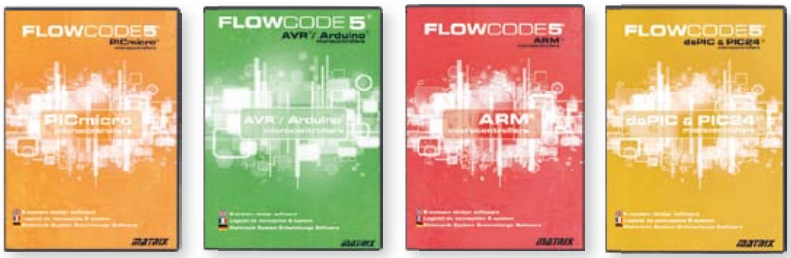
Produits et services Elektor

- microcontrôleur programmé ATtiny2313
- émetteur : 110296-41
- récepteur : 110296-42
- platine d'expérimentation ELEX-1
- téléchargement (gratuit) du logiciel

Tous les produits et téléchargements sont disponibles ici : www.elektor.fr/110296

Flowcode 5 pour concev

FLOWCODE5

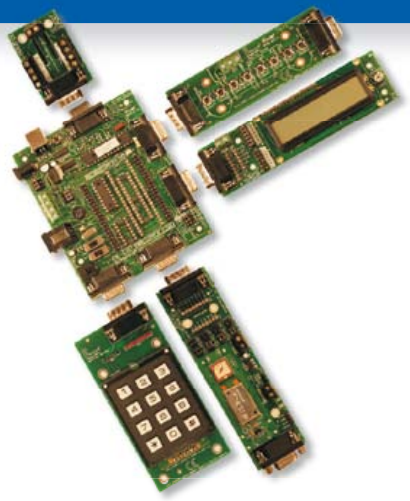


Flowcode 5 est l'un des langages de programmation graphique pour microcontrôleurs (PIC, AVR, ARM et dsPIC/PIC24) les plus avancés au monde. Son avantage principal est de permettre la création de systèmes électroniques et robotiques complexes même si l'on manque encore d'expérience.

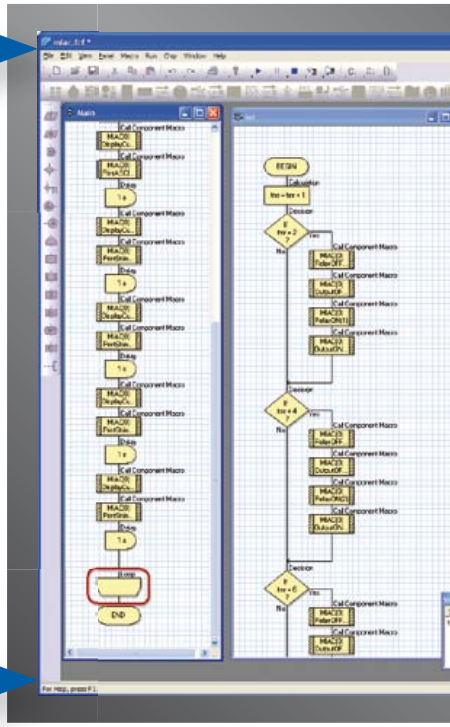
Flowcode est utilisé :

- dans l'enseignement, pour initier les étudiants à la programmation
- dans l'industrie, autant pour le prototypage rapide que pour des projets d'envergure

… en électronique



Les E-blocks sont des circuits électroniques compacts, correspondant chacun à une fonction autonome comme on les trouve dans les circuits embarqués. Il en existe une quarantaine, dont la complexité va croissant, depuis le simple afficheur à LED jusqu'au circuits de programmation, aux modules Bluetooth ou TCP/IP.
Les E-blocks peuvent être assemblés aisément pour élaborer des systèmes propices à l'apprentissage par l'expérimentation. Ils conviennent aussi pour le prototypage rapide de systèmes complexes. L'ensemble est complété efficacement par une gamme étendue et sans cesse renouvelée de logiciels puissants, et de capteurs variés.



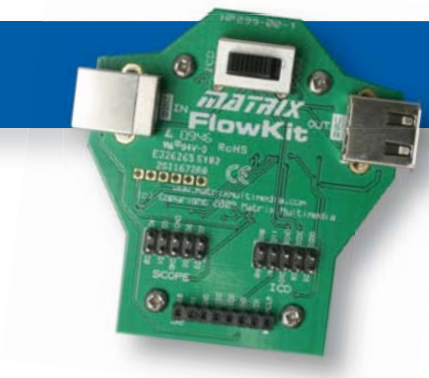
… pour la commande industrielle

MIAC (**M**atrix **I**ndustrial **A**utomotive **C**ontroller) est une unité de commande industrielle pour circuits électroniques variés avec pour champs d'application privilégiés la capture, la mesure, la surveillance et l'automatisation.
Le MIAC lui-même est construit autour d'un puissant microcontrôleur PIC de la série 18 qui se connecte directement au port USB et se programme en Flowcode, en C ou en assembleur. Flowcode est fourni avec le MIAC, lequel est équipé d'origine du bus CAN, qui facilite la connexion en réseau de plusieurs MIAC.

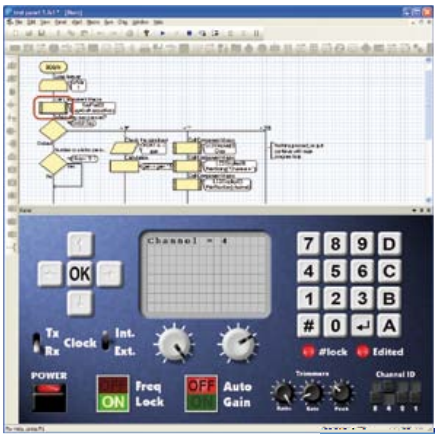
Outil de débogage FlowKit

L'outil FlowKit offre la fonction ICD (In Circuit Debug) pour une gamme étendue d'applications Flowcode dans des projets PIC et AVR :

- marche, arrêt, pause et pas-à-pas pour programmes en Flowcode en temps réel
- suivi des variables de votre programme
- modification des variables
- débogage en circuit du robot mobile Formula Flowcode Buggy, ECIO et de projets autour de MIAC

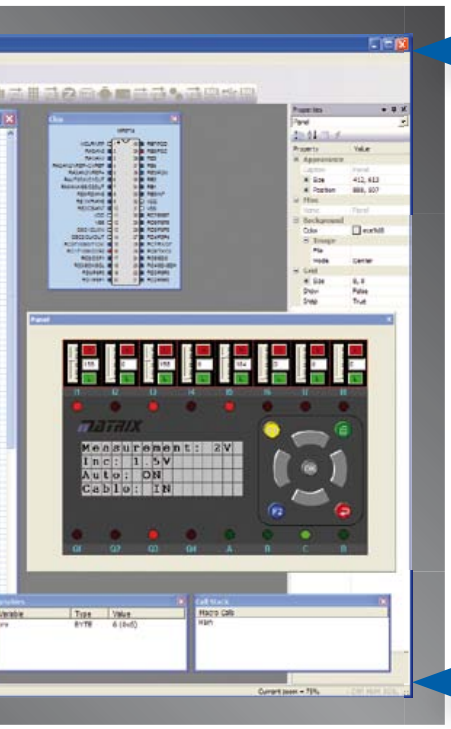


oir et pour se former…



NOUVEAU dans Flowcode 5 :

- Nouvelle présentation personnalisable du code C
- Simulation améliorée
- Fonction de rechercher et de remplacement
- Nouveaux types et nouvelles fonctions des variables, des constantes et des variables de port
- Documentation automatique du projet
- Codage facilité par le nouvel explorateur de projet
- Mise en place de signets de code pour la navigation dans le programme
- La refonte complète du système d'interruption offre aux développeurs l'accès direct à plus de fonctions intégrées
- Amélioration de la signalisation des erreurs de compilation
- Désactivation de fonctions des icônes
- Amélioration des annotations
- Amélioration des liens vers les supports média



… en robotique

Formula Flowcode Buggy est le nom d'un petit robot mobile remarquable par son rapport performances/prix. Ce véhicule fournit aussi bien un support adéquat pour l'apprentissage de la robotique, qu'une plate-forme idéale pour des compétitions de robotique. Loin d'être un jouet, ce robot programmable par l'USB, est doté d'une détection de ligne, de capteurs de proximité, de 8 LED incorporées, d'un capteur sonore, d'un haut-parleur et du connecteur d'extension E-blocks. Ce véhicule se prête à de nombreux exercices de robotique depuis la simple détection de ligne jusqu'à l'analyse de labyrinthe. Le connecteur d'extension E-blocks autorise l'adjonction d'afficheurs, ou de modules Bluetooth, ZigBee ou d'un GPS.

… pour les projets USB

Les ECIO sont de puissants modules à microcontrôleurs programmables par l'interface USB, avec une empreinte au choix de 28 ou 40 broches au standard DIL (0,6 pouce). Construits sur des microcontrôleurs des séries PIC18 et des ARM7, les modules ECIO conviennent bien à l'étudiant et à l'autodidacte. Ils sont programmables en Flowcode, en C ou en assembleur. De nouvelles routines USB sous Flowcode favorisent le développement rapide de projets autour de l'USB, avec y compris les fonctions USB HID, USB esclave, et USB bus sériel (PIC seulement). Vous pouvez incorporer ECIO à vos propres circuits de façon à les doter de la fonction de reprogrammation.

Retrouvez les E-blocks et leur documentation sur :

www.elektor.fr/eblocks

pratique de la commande des moteurs pas-à-pas (2)

les micro-pas

Ed Nisley (Poughkeepsie)

La majorité des systèmes à pas-à-pas récents s'appuient sur des pilotes par micro-pas avec régulation active du courant, souvent intégrés dans une seule puce. Cet article présente les fondamentaux et les avantages des pilotes de moteurs par micro-pas et décrit le fonctionnement de leur commande à découpage de courant. Vous découvrirez aussi en chemin pourquoi les pas-à-pas se mettent parfois à faire de la musique.

Les pilotes classiques de type L/4R que nous avons étudiés dans le premier volet étaient certes acceptables à l'époque où les moteurs pas-à-pas faisaient leur apparition, mais les progrès de l'électronique de puissance et des circuits logiques les ont rendus obsolètes. Les pilotes par micro-pas ont de leur côté l'avantage de réduire considérablement la dissipation d'énergie tout en fluidifiant les mouvements au prix d'une réduction du couple et d'une complexification de la configuration du pilote.

Montage de test

Le schéma de la figure 1 pourrait représenter quasiment n'importe quel système de commande de moteur pas-à-pas : un gros bloc dont les entrées *Step* (Pas) et *Direction* commandent les courants appliqués aux enroulements d'un moteur pas-à-pas bipolaire. Les autres entrées du circuit permettent la configuration du mode micro-pas, l'activation du pilote et le réglage d'autres options.

Les autres blocs représentent le matériel que j'ai utilisé pour cet article afin de synchroniser l'oscilloscope et mesurer le courant. La photo 1 vous donnera une idée de l'enchevêtrement de fils et de sondes qui entourent le circuit de commande (en vert) disposé sur ma plaque d'essais ; un moteur NEMA 17 trône sur le dessus du châssis.

J'utilise une carte de commande *Polulu* A4988 parce qu'elle est simple, bon marché, et montre bien les progrès des circuits de commande. Montée sur un radiateur approprié, la puce DMOS A4988 d'*Allegro* dont la carte est dotée peut commander des courants d'enroulement jusqu'à 2 A sous une tension de 35 V, même si cette minuscule carte ne fait qu'un peu de plus de 3 cm² et que la place occupée par les 16 E/S y est pour beaucoup. J'ai fabriqué un petit radiateur en aluminium qui s'adapte autour de la carte, mais l'ai enlevé pour la photo.

Le moteur NEMA 17 de récup, un Minebea 17PM-J034, (hors catalogue, c'est un moteur particulier) de cet article fonctionne mieux avec un pilote par micro-pas que celui (L/R) à haute résistance et haute inductance que nous avons utilisé la dernière fois. Ses enroulements ont une résistance de 2,3 Ω et une inductance de 2,6 mH, ce qui donne une constante de temps L/R d'environ 1 ms. Cela peut paraître proche de l'autre moteur, mais l'inductance beaucoup plus faible permettra les variations rapides du courant dont ont besoin les pilotes par micro-pas.

Il n'est pas conseillé d'utiliser les plaques d'essai sans soudure pour quoi que ce soit d'autre qu'un circuit de démonstration rapide. En fait, les contacts de ces plaques ne sont pas prévus pour plus d'un ampère et la distribution inappropriée des masses et alimentations est une quasi-garantie de problèmes avec les circuits analogiques. Si votre montage fonctionne sur une telle plaque, tant mieux, dans le cas contraire, utilisez une meilleure technique de prototypage !

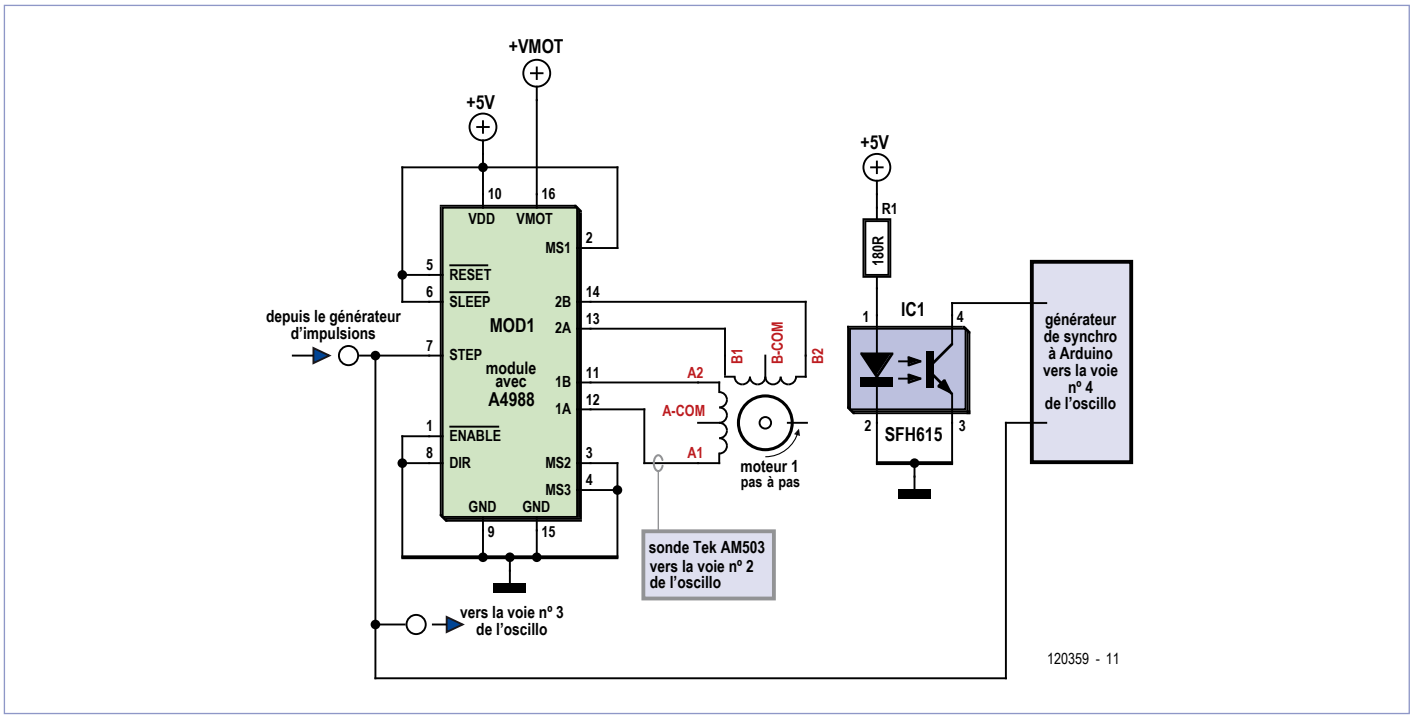


Figure 1. Ce montage de test utilise un générateur d'impulsions pour l'avancement des pas et une sonde de courant pour surveiller le courant dans un enroulement. La carte à Arduino produit une impulsion de synchronisation pour l'oscilloscope, alignée avec chaque groupe de quatre pas entiers et liée fermement à la rotation du moteur à l'aide d'une fourche optique.

Autour des micro-pas

En interne, le A4988 ressemble au schéma de la **figure 2** du premier article de cette série de trois. Quatre transistors montés en pont en H relient chacun des enroulements du moteur à l'alimentation, ce qui permet le passage du courant dans les deux sens. Le A4988 comporte, en plus des deux ponts en H, des circuits numériques de commande par pas et des circuits analogiques qui mesurent et limitent le courant dans les enroulements.

Un circuit de commande par micro-pas divise chaque pas en un nombre fixe de pas plus petits (micro) et ajuste le courant dans les deux enroulements afin d'interpoler la position du rotor en fonction. Un signal sinusoïdal découpé en pas donne un mouvement fluide, même si d'autres pilotes par micro-pas utilisent d'autres interpolations.

La photo 2 montre les formes de courant dans les deux enroulements durant quatre pas entiers, le circuit de commande étant en mode micro-pas 1/8. Comme je n'avais qu'une seule sonde de courant sous la main, j'ai enregistré la première trace puis redéclenché l'oscilloscope avec la sonde sur l'autre enroulement.

Les deux pics, positifs et négatifs, des signaux de courant de la photo 2 correspondent aux quatre positions en pas entiers du **tableau 1** du précédent article : le courant nominal dans un seul enroulement entraîne l'alignement du rotor avec les pôles de l'induit. Les fronts montants de la *Trace 3*, en bas de l'écran, corres-

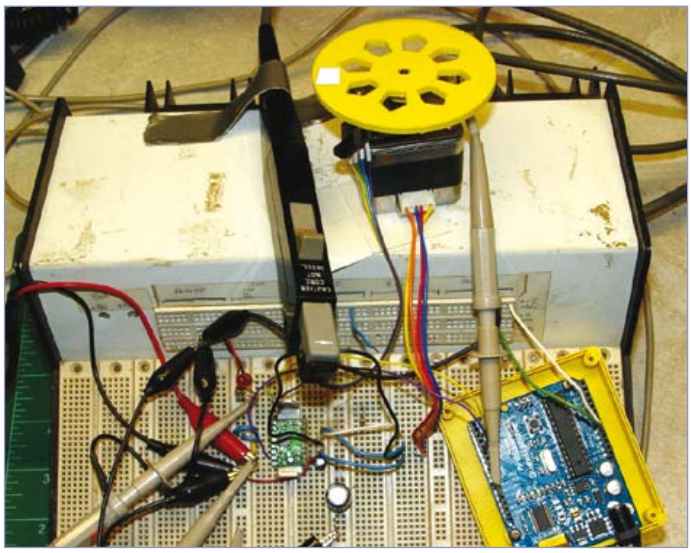


Photo 1. Cet enchevêtrement brouillon de composants, fils et autres sondes à servi à produire les données de cet article. Ne faites surtout pas ça sur un système en production. La roue jaune fournit une impulsion par tour à la carte Arduino pour réaliser la synchronisation de l'oscilloscope.

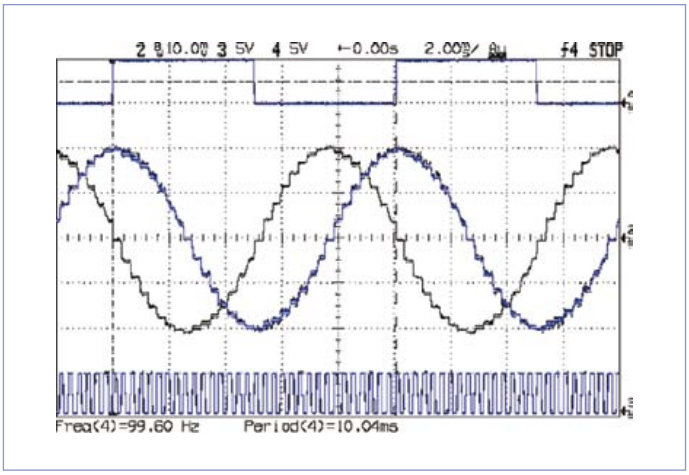


Photo 2. Chaque cycle de ces signaux sinusoïdaux (enroulement A en bleu, B en noir) montre 32 micro-pas qui correspondent à quatre pas entiers. Le front montant de chacune des impulsions de la trace du bas marque le début d'un micro-pas. L'échelle verticale est de 500 mA/div.

pondent au début des micro-pas ; il y a huit impulsions entre deux pics successifs.
Ce schéma de signaux sinusoïdaux se répète pour chaque groupe de quatre pas entiers du moteur. Le moteur que j'utilise possède 200 pas entiers par tour, donc 50 de ces groupes par tour, ce qui, en mode micro-pas 1/8, veut dire 1600 micro-pas par tour.

Le fait que les circuits de commande de moteurs possèdent une entrée *Step* qui accepte des impulsions au rythme des micro-pas ou des pas entiers peut entraîner une certaine confusion. Cependant, les fiches techniques ne donnent les spécifications des moteurs que pour le rythme par pas entiers, et vous devrez donc ajuster ces valeurs en fonction de la configuration du mode micro-pas de votre circuit de commande.

Courant et trigonométrie

Comme nous l'avons vu dans le premier article [10], le courant d'enroulement maximal détermine le couple du moteur et la dissipation énergétique dans les enroulements. L'intensité du courant lors des pics ne doit pas dépasser cette valeur ; quand le moteur s'arrête sur le micro-pas correspondant, ce qui n'était vu que comme une valeur transitoire devient la dissipation continue de l'enroulement.
Le circuit de commande *Allegro* A4988 limite I_{MAX}, la valeur maximale du courant dans un enroulement, en fonction de la tension présente sur son entrée V_{REF}. Cette tension peut provenir d'un diviseur à résistances fixe, d'une ajustable, ou d'un CNA de microprocesseur, mais doit généralement pouvoir être considérée comme continue et stable.
Le circuit compare V_{REF} à la tension issue d'une résistance de mesure du courant qui convertit le courant d'enroulement en une petite tension. La carte *Pololu* A4988 utilise des résistances de mesure de 50 mΩ et l'amplificateur de mesure du A4988 applique un gain de 8, la relation entre V_{REF} et I_{MAX} est donc :

$$I_{MAX} = \frac{V_{REF}}{8 \times R_{SENSE}}$$

En exprimant V_{REF} à partir de cette équation, on peut calculer que pour un courant maximal de 1 A, V_{REF} doit valoir 0,400V = 1 A × (8 × 0,050 Ω).

Comme on peut s'y attendre, le courant d'enroulement pour chaque micro-pas est une fraction du courant de pointe déterminé par une fonction sinusoïdale dont la période correspond à quatre pas entiers. En mode micro-pas 1/8, chaque sinusoïde comporte 32 micro-pas par période et le courant pour le micro-pas N (N variant entre 0 et 31) est donné par :

$$I_{Winding\ A} = I_{MAX} \times \cos\left(\frac{N}{32} \times 360^{\circ}\right)$$

Le déphasage du courant dans l'enroulement B par rapport au A détermine le sens de rotation du moteur. Lorsque l'entrée *Direction* du A4988 est à 0 V, comme sur la photo 2, le courant dans l'enroulement B est donné par :

$$I_{Winding\ B} = -I_{MAX} \times \sin\left(\frac{N}{32} \times 360^{\circ}\right)$$

Si l'entrée *Direction* est mise au niveau haut, la phase s'inverse et le courant devient :

$$V_L = L\frac{\Delta I}{\Delta T}$$

Le circuit A4988 utilise une table de valeurs pré-calculées pour convertir les numéros de micro-pas en fraction du courant : les calculs trigonométriques ne sont pas faits à la volée ! Les pattes MSx de sélection du mode micro-pas permettent de choisir l'incrément utilisé pour faire avancer l'index sur la table à chaque impulsion sur l'entrée *Step*, la patte *Direction* détermine si l'incrément est ajouté à ou soustrait de l'index.
D'ailleurs, vous pouvez aussi changer le sens de rotation d'un moteur bipolaire via un changement de câblage, l'inversion de deux connections d'un enroulement change le sens du courant dans cet enroulement. Même si la fiche technique donne la polarité des enroulements et que vous pouvez déduire le sens de rotation de manière théorique, la probabilité que le moteur tourne dans le bon sens au moment de la configuration d'un système reste invariablement de 0,5.

Commande à découpage de courant

Étant donné que le couple d'un moteur pas-à-pas dépend du courant dans ses enroulements, un pilote idéal produirait des changements instantanés du courant dans les enroulements et des impulsions de courant rectangulaires qui dureraient tout le micro-pas. Hélas, c'est tout simplement impossible : les bobines de fils enroulés autour d'armatures en fer ont une fâcheuse tendance (les coquines!) à constituer de bonnes inductances. Les pilotes L/4R classiques amélioraient le temps de montée en réduisant la constante de temps par l'utilisation d'une forte résistance, avec l'inconvénient d'un rendement médiocre.
Les pilotes par micro-pas prennent une autre route en contournant la constante de temps L/R des enroulements avec une tension relativement haute couplée à une limitation active du courant.
Dans les deux cas, l'équation fondamentale qui décrit le phénomène est :

$$200\ mA = (1\ A) \times \cos\left(\frac{25}{32} \times 360^{\circ}\right)$$

L'augmentation de la tension appliquée à l'inductance produit un changement proportionnellement plus rapide du courant dans l'enroulement. Le problème devient la gestion de l'augmentation de la dissipation énergétique due à un courant plus fort.
Cette équation montre pourquoi les moteurs ayant une inductance d'enroulement faible fonctionnent mieux en mode micro-pas : pour une tension donnée, la vitesse d'évolution du courant est inversement proportionnelle à l'inductance de l'enroulement. La résistance des enroulements du moteur autorisera un courant extrêmement fort pour la tension appliquée mais la limitation active du courant le limitera à la valeur bien plus faible requise pour chaque micro-pas.
La photo 3 donne une vue détaillée du courant d'enroulement au début du micro-pas visible au centre de la photo 2. Pour un pic de 1 A, le courant pour ce micro-pas est :

$$13{,}9\ V = 2{,}6\ mH\ \times \frac{240\ mA}{45\ \mu s}$$

La montée jusqu'à 240 mA dure environ 45 µs, ce qui indique que la tension aux bornes de l'enroulement est de :

$$840\ \mu s = \frac{2{,}6\ mH}{2{,}3 + 0{,}8\ \Omega}$$

Mon alimentation de laboratoire était réglée sur 18 V, ce qui veut dire que 4,1 V ont été perdus quelque part entre les MOSFETs, la résistance de l'enroulement, la force contre-électromotrice, et les autres pertes de câblage. Les deux MOSFET du A4988 ajoutent 800 mΩ au total, la constante de temps L/R vaut donc :

$$17\ V = 2{,}6\ mH \times \frac{80\ mA}{12\ \mu s}$$

C'est beaucoup plus que les 200 µs correspondant à la durée d'un micro-pas, les approximations linéaires des courbes exponentielles donneront donc des résultats raisonnablement précis.
Contrairement aux montées exponentielles douces que l'on peut voir sur la **figure 7** du premier article, le A4988 limite le courant en coupant ses MOSFET lorsque le courant atteint la limite haute pour chaque micro-pas puis les remet en marche après un délai fixe. Sur la carte *Polulu* ce délai est réglé à 12 µs, ce qui correspond à la durée des moments où le courant baisse sur la photo 3.
La feuille de caractéristiques du A4988 d'*Allegro* parle de décroissance rapide, mixte ou lente (*fast decay*, *mixed decay* et *slow decay*) sans les définir précisément. Il s'avère que cela fait référence à la vitesse de changement du courant lorsque les MOSFET sont éteints et, par conséquent, à la tension appliquée à l'enroulement dont il est question. Ces termes sont plus faciles à comprendre si l'on s'aide de modèles de simulation de la circuiterie interne du A4988 qui peuvent nous montrer comment le courant se comporte.
Le modèle de simulation *LTSpice IV* de la photo 4 comprend Q1 et Q4, les deux MOSFET du pont en H qui commandent le courant d'enroulement pour le micro-pas visible sur la photo 3, plus D2 et D3, les diodes de corps de Q2 et Q3, les MOSFET du pont en H qui

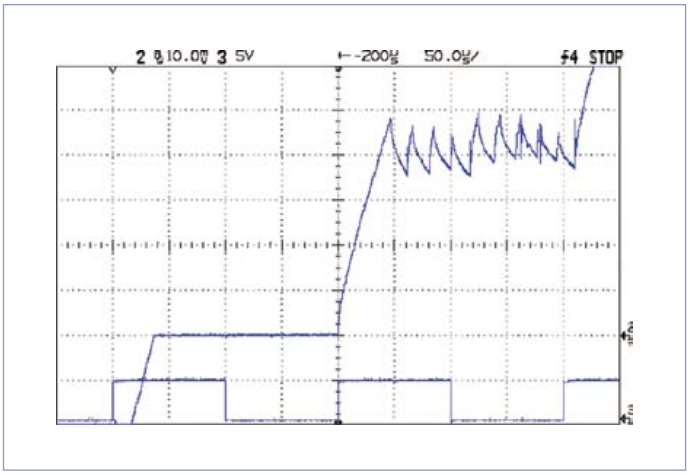


Photo 3. Le courant change très rapidement au début de chaque micro-pas. Avec 18 V appliqués à un enroulement de 2,3 Ω, le A4988 limite le courant à environ 200 mA pour le premier micro-pas. L'échelle verticale est de 50 mA/div.

sont bloqués dans ces conditions. Bien que j'ai choisi des MOSFET et diodes discrets avec des propriétés similaires à ceux qui sont dans le A4988, les résultats produits par ce modèle ne correspondront pas exactement à ceux de la vraie puce.
En complément, j'ai simulé (ce qui n'est pas simple) la partie analogique de mesure du courant et les circuits numériques de commande à l'aide d'une paire de sources de tensions impulsionnelles bidouillées pour produire des résultats équivalents ; une simulation plus détaillée sortirait allègrement du cadre de cet article.
La photo 5 montre que le courant simulé ressemble au courant réel de la photo 3. L'ondulation du courant est bien plus grande, cependant le modèle de simulation fonctionne dans ce qu'Allegro appelle le mode à décroissance rapide, le courant d'enroulement traversant D2 et D3 quand Q1 et Q4 sont bloqués, ce qui relie l'enroulement directement aux bornes de l'alimentation. Le courant décroît donc sensiblement plus vite qu'il ne monte :

$$0{,}56\ V = 2{,}6\ mH \times \frac{2{,}6\ mA}{12\ \mu s}$$

Vous remarquerez que le courant traversant R2, la résistance de mesure, change de direction lorsque le courant d'enroulement décroît. Si vous mesurez le courant avec une sonde d'oscilloscope sur la résistance de mesure, vous verrez que celle-ci devient négative dans cette partie du cycle.
Avec le mode à décroissance lente du A4988, les deux MOSFET du bas du pont en H sont passants lorsque ceux du haut sont bloqués, le courant d'enroulement se retrouve « piégé » dans la boucle inférieure du pont en H. Ceci permet de tirer parti de trois caractéristiques des MOSFET : leur faible résistance à l'état passant, l'absence de tension de saturation, et la conduction dans les deux sens.
Le modèle de simulation de la photo 6 ressemble à celui de la photo 4, sauf que maintenant Q3 est passant et conduit le courant à travers D3, sa diode de corps, de la source vers le drain lorsque Q1 se bloque. Une bizarrerie de LTSpice nous contraint à utiliser des OU logiques distinctes pour piloter les grilles des deux MOSFET.
La photo 7 montre que le mode à décroissance lente engendre des ondulations du courant beaucoup plus faibles lors des micro-pas qu'avec le mode à décroissance rapide de la photo 5 ; les deux gra-

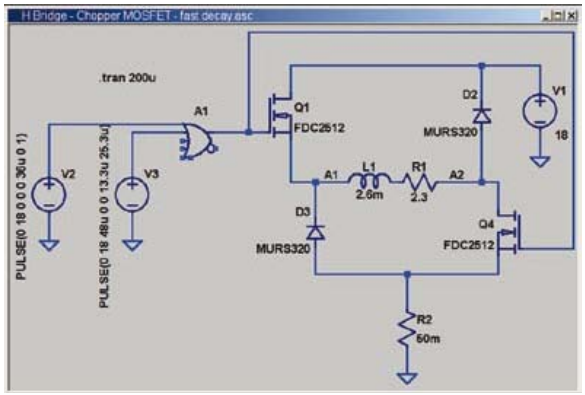


Photo 4. Ce modèle de simulation montre le comportement du courant d'enroulement pour le mode à décroissance rapide, seuls les MOSFET Q1 et Q4 sont actifs. D3 représente la diode de corps de Q3, le transistor en bas à gauche du pont en H.

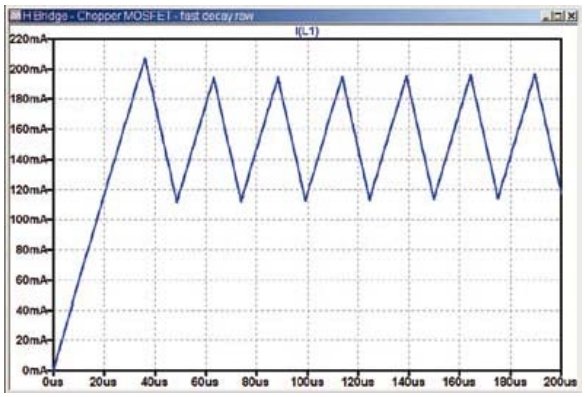


Photo 5. Le mode à décroissance rapide engendre des ondulations du courant relativement fortes, la faute aux diodes de corps des MOSFET qui laissent la tension totale d'alimentation s'établir aux bornes des enroulements lors des décroissances du courant.

phiques utilisent les mêmes échelles et les temps de coupure des MOSFET sont les mêmes. En agrandissant le graphique (ce que l'on peut faire avec le modèle de simulation disponible en téléchargement), on s'aperçoit que la tension aux bornes de l'inductance est étonnamment faible :

$$0{,}400\ V = 1\ A \times (8 \times 0{,}050\ \Omega)$$

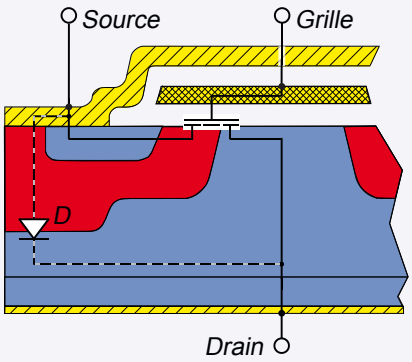
Une ondulation plus faible du courant réduit les pertes d'énergie dues à l'hystérésis dans le noyau en fer de l'induit, ce qui réduit aussi les bruits audibles dûs à la magnétostriction. Cependant, la fréquence d'ondulation baisse également, ce qui peut déplacer les bruits restants dans le domaine audible. Il faut prendre en compte les interactions entre les harmoniques et sous-harmoniques et le produit de leur mélange vis-à-vis de la fréquence de pas ; presque tout semble possible.
C'est cette énergie acoustique qui produit un sifflement à haute fréquence sur les moteurs pas-à-pas à l'arrêt. Lorsque le moteur est en mouvement, toutefois, il peut produire des vibrations mécaniques audibles : en mode micro-pas 1/8, un moteur 200 pas/tr qui tourne à 1 tr/s ronronne à 1600 Hz. Vous trouverez même dans les liens des séquences de commande qui produisent de la musique.
Le A4988 utilise généralement le mode à décroissance mixte, qui passe du mode rapide au mode lent à mesure que le courant décroît durant les périodes de non-conduction. La résolution de la photo 3 ne permet pas de le voir clairement, mais vous pouvez quand même y voir une légère inflexion de la courbe à environ 1/3 de la période de non-conduction, au moment où le circuit change de mode. La simulation de ces effets requière un modèle plus complexe que celui présenté ici ; on devrait toutefois pouvoir s'en sortir en ajoutant quelques sources de tension réglées avec précaution.
La forme du courant d'enroulement nous rappelle pourquoi les modes micro-pas sont aussi appelés commande à découpage de courant. Comme les MOSFET sont soit totalement bloqués, soit totalement passants, il ne dissipent que peu d'énergie, c'est ce qui permet à une très petite puce de commander des moteurs étonnamment gros. Le A4988 est capable de commander 2 A depuis une source 35 V, c'est assez pour beaucoup de pas-à-pas en taille NEMA 23.

Diode de corps

La partie métallique, reliée à la source, des transistors de type VDMOS (MOS à diffusion verticale) est en contact à la fois avec les zones dopées P+ et N+, afin d'éviter de laisser en l'air la base du transistor NPN parasite, présent du fait de leur structure, ce qui les rendrait incontrôlables lorsque l'intensité du courant de drain est forte.

Cela présente l'inconvénient de créer l'équivalent d'une diode (jonction PN), dite de corps, entre la partie dopée P côté source et la partie dopée N côté drain. Le courant ne peut donc être bloqué que dans un sens.

Lire notamment le paragraphe ***Body diode*** sur http://en.wikipedia.org/wiki/Power_MOSFET, et même la page entière, ça ne fait pas de mal si l'on n'est pas spécialiste de la micro-électronique.



- En jaune les parties métalliques
- En rouge, les zones dopées P
- En bleu les zones dopées N

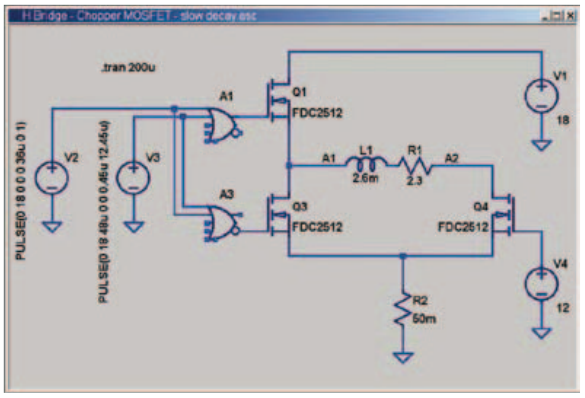


Photo 6. Ce modèle de simulation montre le comportement du courant pour le mode à décroissance lente. Les deux MOSFET du bas sont passants lorsque Q1 se bloque, le courant d'enroulement se retrouve piégé dans la boucle inférieure du pont en H.

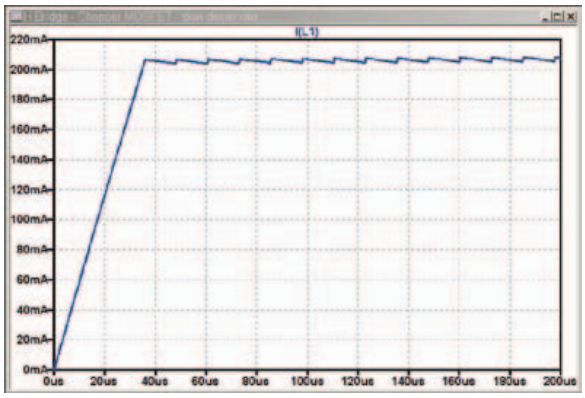


Photo 7. Avec le mode à décroissance lente, l'ondulation du courant est très faible, les MOSFET faisant circuler le courant avec une très faible chute de tension.

Bien entendu, il faut faire les calculs et s'assurer que la température de la puce reste dans des limites acceptables, mais l'époque où de grosses banques de résistances de puissance trônaient dans des boîtiers ventilés au dessus des machines à commande numérique est loin derrière nous : bon débarras !

On coupe le contact, jusqu'à la prochaine fois

Dans cet article, nous avons parlé de la partie « montante » des signaux. Ce qui se passe sur l'autre pente de la sinusoïde est bien différent à cause de la force contre-électromotrice générée par la rotation du moteur. La prochaine fois, nous parlerons mécanique et vous verrez cette tension en action.

Mon ami Eks m'a passé un antique amplificateur pour sonde de courant Tektronix AM503 ainsi qu'une sonde à effet Hall A6302 pour capturer les signaux de courant. Une sonde A622 moderne constituerait une alternative à peu près aussi coûteuse, mais plus facilement disponible. Si l'on veut faire des mesures significatives, la bande passante de l'amplificateur doit inclure la composante continue, ce qui exclut d'office les pinces ampèremétriques à induction. La synchronisation de l'oscilloscope à la rotation du moteur et aux impulsions de pas fait appel à un programme Arduino vite fait qui utilise un temporisateur matériel de l'ATmega168 pour s'affranchir de la gigue inhérente aux logiciels. C'était bien plus facile à mettre en place qu'une poignée de portes et de compteurs ; après tout, c'est bien pour cela que les microcontrôleurs existent !

(120359 – version française : Kévin PETIT)

Bibliographie

E. Nisley, Stepper Drive Part 1: Analog, Circuit Cellar 169, 2004

E. Nisley, Stepper Drive Part 1: Digital, Circuit Cellar 171, 2004

E. Nisley, Stepper Failure: Death By Disconnection, Circuit Cellar 191, 2006

E. Nisley, Crystal Properties: Circuit Models, Measurement, and Conversion, Circuit Cellar 241, 2010

Liens et références

[0] La page de cet article sur le site d'Elektor
www.elektor.fr/120359

[1] Pilote par micro-pas Allegro A4988 d'Allegro MicroSystems
www.allegromicro.com/en/Products/Part_Numbers/4988

[2] Cartes à microcontrôleurs Arduino
www.arduino.cc/en

[3] Microcontrôleur ATmega168 d'Atmel
www.atmel.com

[4] Logiciel de schémas et PCB Eagle de CadSoft1www.cadsoftusa.com

[5] Simulation de circuits et modèles de composants, Linear Technology
www.linear.com/designtools/software

[6] Platine de pilotage de moteurs pas-à-pas A4988, Pololu
www.pololu.com/catalog/product/1182

[7] Amplificateur pour sonde de courant AM503 et sonde de courant à effet Hall A622, Tektronix
www.tek.com/products/accessories/current.html

[8] RepRap.org, bibliothèque Eagle
http://reprap.org/wiki/Eagle_Library

[9] Le fabricant des Makerbots : Makerbot Industries, Makerbot Music
http://wiki.makerbot.com/makerbot-music

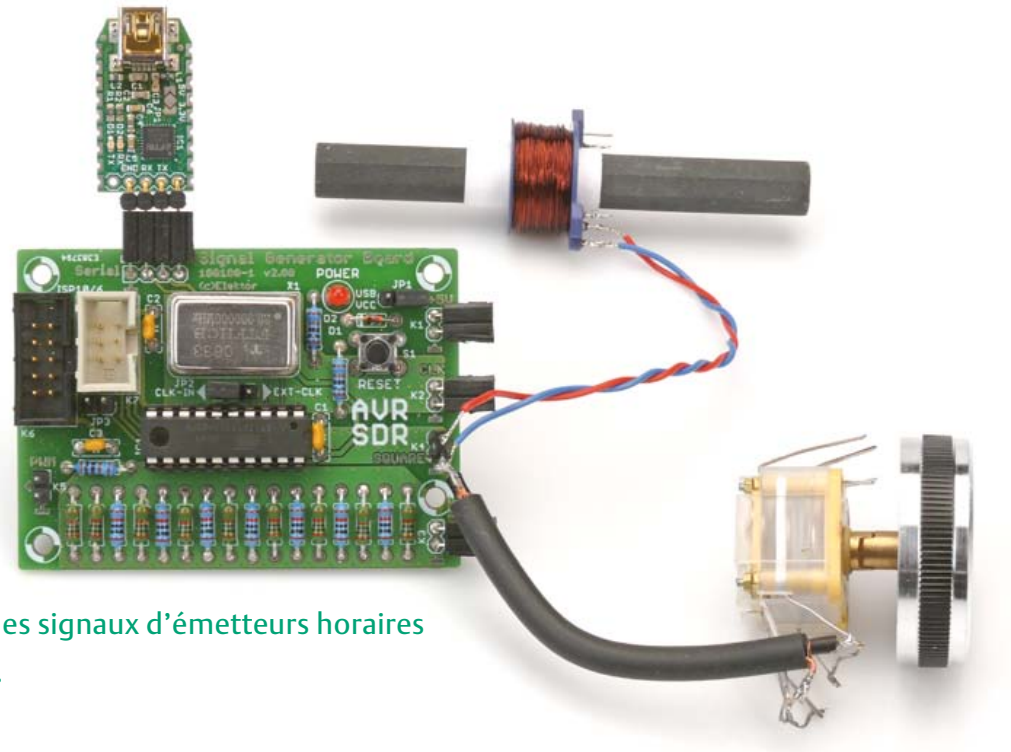
[10] Premier article de cette série
www.elektor.fr/120358

[11] Modèles LTspice simulation and code source Arduino
ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2011/255

radio logicielle avec AVR (5)

décoder DCF77, MSF et TDF162, du RII au *matched filter*

Martin Ossmann (Aix-la-Chapelle)



Les articles de cette série montrent tout le parti que l'on peut tirer des contrôleurs AVR dans le traitement numérique du signal. Aujourd'hui, nous allons utiliser différentes méthodes de décodage et de filtrage pour traduire en données numériques les signaux d'émetteurs horaires allemand, britannique et français.

Dans l'épisode précédent [4], nous avons travaillé sur la réception numérique de la radio. Avec le récepteur décrit alors, nous avons pu non seulement capter les signaux de DCF77 et des services météo, mais aussi les décoder. Nous pouvons à présent jouer avec différents décodages. Après l'émission et la réception de signaux RTTY, intéressons-nous à ceux de l'allemand DCF77, du britannique MSF et du français TDF162. Enfin, nous décortiquerons un *matched filter*, un filtre adapté, pour nous aider à décoder les bits.

Transmission sans fil de données sur 125 kHz

Si le signal de DDH (nous en parlions à la fin de l'article précédent, dans le numéro de juin) n'est pas suffisamment fort pour une bonne réception, on peut construire son propre émetteur de test. Le matériel est tout pareil à celui utilisé pour DCF dans la 3e partie [3]. Il n'y a qu'à combiner une antenne ferrite et un condensateur variable pour former un circuit résonant série à exciter par le signal rectangulaire du générateur. La figure en tête de cet article vous montre comment faire.

On prend comme logiciel pour l'émetteur le programme « EXP-SQTX-FM-RTTY-V01.c ». Celui de réception, EXP-125kHz-RTTY-RX-V01.c, il suffit de l'adapter pour 125 kHz et c'est parti pour une expérience d'émission et réception. J'ai obtenu avec ce système simple une portée de plus de 5 m, même à travers les murs. Il marche aussi fort bien comme télécommande.

Le même logiciel convient également au décodage des signaux RTTY classiques dans les bandes pour radioamateurs. Le tout est d'adapter le filtre et la chronologie aux paramètres de transmission utilisés. Pour des essais en RTTY, on peut aussi bien se servir de la carte son et des logiciels pour PC tels que MMtty [6].

Le décodage de DCF77

Nous avons déjà pu observer à la figure 3 de la 3e partie de cette série l'oscillogramme du signal démodulé de DCF77. Comment faire à présent pour récupérer les informations contenues dans les impulsions ? Comme la force du signal reçu peut varier considérablement en fonction des circonstances, mieux vaut commencer par déterminer un seuil qui permette de savoir si l'amplitude a diminué ou pas.

Cette valeur peut être simplement la moyenne du signal, pour peu que l'on se base sur une période assez longue. Voudrait-on la déterminer à l'aide d'un filtre CIC, il y faudrait un très grand tampon intermédiaire. Préférons-lui un filtre dit RII [7].

Le filtrage RII

Le sigle RII dérive de réponse impulsionnelle infinie (*IIR* en anglais). On obtient une réponse impulsionnelle longue, c'est-à-dire un long temps de calcul de la moyenne, à l'aide d'un filtre récursif. Voici la formule pour calculer un filtre RII simple :

$$y_{k+1} = ax_k + (1-a)\, y_k$$

On y trouve y_k pour désigner la suite de valeurs de la sortie, x_k pour la suite des valeurs d'entrée et *a* qui est un nombre inférieur à 1, mais de peu. La nouvelle valeur y_{k+1} représente la valeur totale moyenne de l'ancienne et de la nouvelle valeur d'entrée. Dans ce calcul, les grandeurs *a* et *1-a* sont normalement plus petites que 1. On serait donc tenté d'utiliser des opérations en virgule flottante pour l'effectuer, mais ce serait beaucoup trop long. Dans notre réalisation (**listage 1**), les coefficients de filtrage sont exprimés sous forme de fraction dont le dénominateur, 65536, est une puissance de deux.

Dans ce cas-ci, a = (65536–50) / 65536 = 0,999237… et 1–a prend alors la valeur de 0,000763… Les calculs s'effectuent sur des entiers de 32 bits. Le compilateur ne prend que les 16 bits de poids fort pour effectuer la division par 65536, ce qui accélère considérablement la vitesse de traitement, une manière de réaliser un filtre RII très efficace. Le signal se retrouve ainsi directement en binaire. L'analyse de la longueur d'impulsion fournit alors l'information horaire.

Pour la réception et le décodage de DCF77, il nous faut juste le syntoniseur simple ou le récepteur universel, accompagné de l'antenne ferrite active, le tout aligné sur 77,5 kHz. Le programme « EXP-DCFdecode-RX-V01.c » dans le contrôleur du récepteur nous donne un temporisateur en seconde et le signal est restitué par le convertisseur N/A et MLI. La force du champ et l'heure s'affichent ainsi sur le LCD et les données décodées sont transmises en série par RS232 ou USB.

Le décodage de MSF

Il existe en Europe d'autres émetteurs horaires et des standards de fréquence. L'un d'eux est en Grande-Bretagne, il émet sur 60 kHz, c'est MSF [8]. Là où j'habite, la réception en est relativement faible, il me faut donc quelque chose en plus. Sur MSF, l'amplitude de la porteuse est réduite chaque seconde (cf. **fig. 1**). Au début de chaque minute, la porteuse est réduite pendant 0,5 s. Pour chaque seconde suivante, elle s'affaiblit pendant 0,1 s. Le dixième de seconde qui suit correspond au bit A. Si dans ce créneau temporel la porteuse est diminuée, le bit A est au niveau haut. Le créneau suivant de 0,1 s reflète l'état du bit B : ici aussi, un affaiblissement de la porteuse dénote un niveau haut. Les bits A et B servent à transmettre les informations horaires, mais aussi comme synchronisation supplémentaire et détection d'erreur.

Lors de mes premiers essais de réception, je n'ai obtenu qu'un signal fort perturbé, le filtre simple CIC s'est révélé insuffisant. Il me fallait un meilleur filtre avec une fréquence limite d'environ 10 Hz et une atténuation plus forte. Il ne devait pas non plus affecter outre mesure la forme de la courbe et j'ai obtenu ces propriétés avec un filtre RIF, à **r**éponse **i**mpulsionnelle **f**inie, *FIR* en anglais [9] d'ordre suffisant.

Pour recevoir MSF, il faut, outre le récepteur, l'antenne ferrite active accordée sur 60 kHz et le logiciel EXP-RX-MSF60decode-V01.c. On en retire alors la force du champ et l'heure affichées à l'écran ainsi que les données sérielles. Le filtre RIF du 5e ordre est représenté à la **figure 2**. Les valeurs d'entrée circulent dans une mémoire tampon. La valeur de sortie s'obtient en faisant la somme des valeurs dans le tampon, pondérées par le facteur a_k.

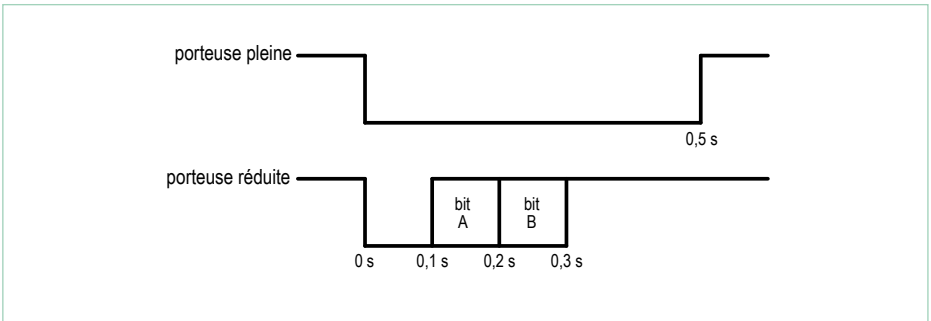


Figure 1. Réduction de la porteuse sur MSF.

Listage 1 : Filtre RII

```
#define ALPHA 50
#define BETA  65536
int32_t Threshold ;

  Threshold=(ALPHA*(int32_t)Amplitude+(BETA-ALPHA)*Threshold)/BETA ;
  if (Amplitude>Threshold) {
    DCFlevel=0 ;
    }
   else {
    DCFlevel=1 ;
    }
```

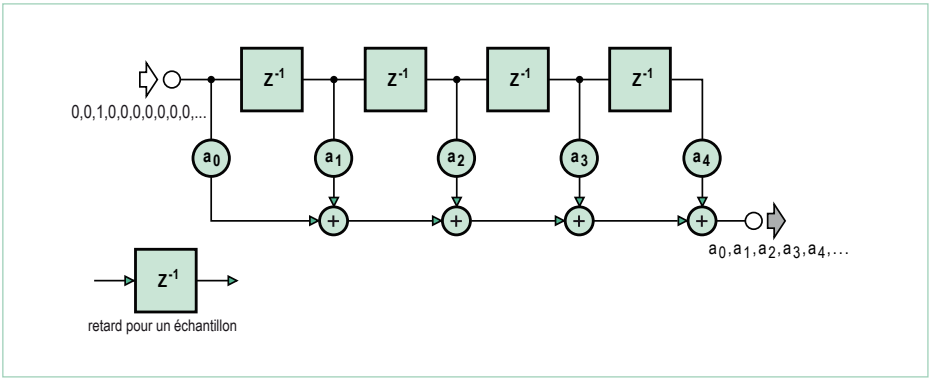


Figure 2. Trajet du signal dans un filtre RIF.

Listage 2 : Sommation RIF

```
#define FIRLength 60

prog_int16_t FIRCoeffs []={
   112,     3,   -17,
   -52, ............. 112 }  ;

  FIRSum=0 ;
  j=FIRPointer ;
  for (k=0 ; k<FIRLength ; k++ ) {
    FIRcoef=pgm_read_word( FIRCoeffs1 + k ) ;
    FIRSum += (int32_t)FIRcoef*(int32_t)FIRBuffer[j] ;
    j++ ; if (j==FIRLength) { j=0 ; }
    }
  return FIRSum/0x10000L ;
```

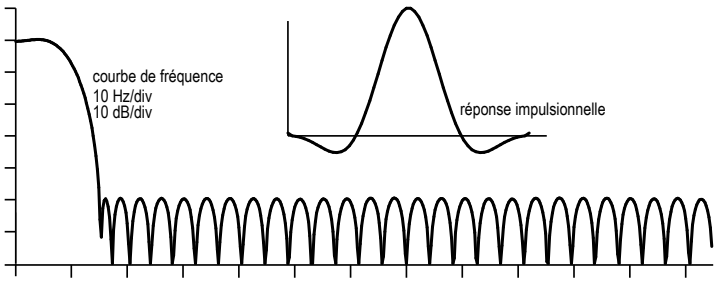


Figure 3. Réponse impulsionnelle et caractéristique de fréquence du filtre RIF.

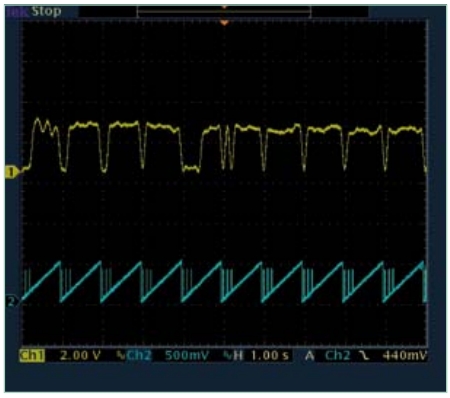


Figure 4. Signal de MSF démodulé et filtré, en bleu la chronométrie.

Figure 5. Les trois impulsions montrent à quel moment le signal est échantillonné.

Le **listage 2** constitue l'ensemble du programme rédigé en C. Il utilise lui aussi l'arithmétique rapide des nombres entiers ainsi que la division finale. Cette routine filtre le signal qui est enregistré à une fréquence de 250 échantillons par s. Le temps de calcul, même avec un 60e ordre, est assez court, 300 µs, alors qu'on dispose ici de 4 ms.

La **figure 3** montre la réponse impulsionnelle et la caractéristique de fréquence du filtre. Il commence à atténuer fortement à 8 Hz et atteint environ 50 dB au-dessus de 15 Hz.

On voit à la **figure 4** le signal MSF démodulé et filtré, c'est la trace jaune. À peu près au milieu de l'image, on distingue la marque de la minute, l'impulsion dure 0,5 s. Deux secondes avant l'impulsion de la minute, on aperçoit une impulsion qui représente A=0 et B=1. Ensuite viennent plusieurs impulsions avec A=0 et B=0. Sous la trace jaune, la trace bleue a la forme de dents de scie. Ces signaux permettent de suivre la chronologie.

On voit mieux les rapports dans la **figure 5**. Il y a dans le logiciel une variable qui s'appelle SecondTimer, elle compte sans arrêt de 0 à 250 à la vitesse de 250 Hz. Cette fréquence correspond au taux d'échantillonnage pour le signal démodulé. Lors de la détection de l'impulsion de minute, le SecondTimer est remis à zéro, de sorte que le comptage recommence toujours au début d'une seconde. La variable SecondTimer est alors produite par l'un des convertisseurs N/A MLI. Les trois impulsions bleues indiquent où le logiciel évalue le signal. C'est là qu'il détermine les valeurs des bits A et B. Quand tous les bits ont été collectés, ils sont mis à profit et l'information temporelle est envoyée à l'écran et sur RS232. On peut voir dans le **listage 3** un exemple de données de sortie.

Le décodage de France Inter (TDF)

L'émetteur de France Inter diffuse aussi un signal horaire. Il le fait en modulation de phase. Les signaux pour les niveaux bas et haut, vous les voyez à la **figure 6**. Les informations sont codées de la même manière

Listage 3 : Exemple de données de la station MSF

```
<20000000000000000001000000011001000001000111000100101111111103>  07:09 08.03.10 MONDAY    P=1 P=1 P=1 P=1
<20000000000000000001000000011001000001000111001000001111313103>  07:10 08.03.10 MONDAY    P=1 P=1 P=1 P=1
<20000000000000000001000000011001000001000111001000010111111103>  07:11 08.03.10 MONDAY    P=1 P=1 P=1 P=1
```

que sur DCF77. Mais pour le décodage, il y a une difficulté supplémentaire. Au début de chaque seconde, il envoie le bit du code temporel, mais après, il arrive un tas de signaux de modulation de phase à usage interne dans lesquels le récepteur doit d'urgence déterminer exactement l'emplacement de l'impulsion de seconde. Avant l'impulsion de seconde qui correspond à la seconde « 0 », il y a une longue pause dans la modulation, puisque l'impulsion « 59 » n'existe pas. Quand le récepteur a détecté cette pause, il recherche le prochain maximum dans le déroulement de la phase (cf. fig. 6). Ce maximum se situe exactement à l'endroit de l'impulsion de seconde. À partir de là, il examine, à l'intervalle exact d'une seconde, si c'est un 1 ou un 0 qu'il reçoit.

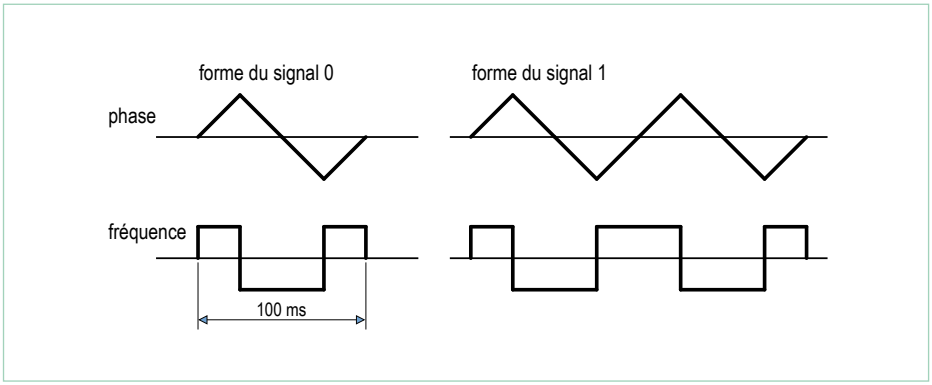


Figure 6. Les conformations du signal de TDF162.

Un filtre adapté avec CIC

Si vous voulez évaluer directement la phase, il ne faut pas qu'elle dérive, évidemment. En conséquence, vous auriez besoin d'une boucle à phase asservie, une PLL, pour la stabiliser. Mais c'est assez compliqué, d'autant que les circuits de réglage peuvent se montrer peu stables. Cherchons une approche différente. Au lieu de s'attacher à la phase, envisageons simplement la fréquence. On engendre une fréquence à partir de la phase par la mesure de la différence entre les valeurs successives. Ces valeurs, nous allons ensuite les filtrer de nouveau dans un filtre passe-bas CIC, pour que la forme du signal soit plus propre. Pour déterminer si c'est un 0 ou un 1 qui a été émis, on vérifie si un signal correspondant à un 1 a été reçu au bon moment. Si ce n'est pas le cas, le récepteur décide que c'est un 0.

Reste à savoir si c'est bien la forme de signal prévue qui a été reçue. Dans la technique des télécommunications, on utilise pour cela ce qu'on appelle un filtre adapté (*matched filter*). Comme réponse impulsionnelle, il possède exactement la forme d'onde recherchée (la réponse impulsionnelle est simplement l'image inversée dans le temps). Notre signal 0 se compose d'une combinaison de trois impulsions rectangulaires. On peut donc simplement réutiliser un filtre adapté avec un filtre CIC (la réponse impulsionnelle, donc réponse à un unique 1 à l'entrée, est sur ce filtre une seule impulsion rectangulaire, voyez sur [4]). C'est ce que la **figure 7** présente schématiquement. Là, l'échantillonnage du signal s'opère à 500 Hz. La première impulsion rectangulaire est large de 25 ms et correspond donc à 12 échantillons. La deuxième impulsion dure 50 ms et recouvre alors 25 échantillons. La dernière a aussi une largeur de 25 ms. C'est la largeur des impulsions qui dicte directement la longueur nécessaire des étages à retard. Ce filtre détecte parfaitement la forme de signal du 0. Comme le signal du 1 lui ressemble (deux signaux 0 sont simplement envoyés l'un derrière l'autre) rien n'empêche d'utiliser le même filtre, il suffira d'aller voir 50 ms plus tard. C'est exactement la tactique adoptée dans notre récepteur.

Les oscillogrammes des **figures 8 et 9** montrent les formes d'ondes correspon-

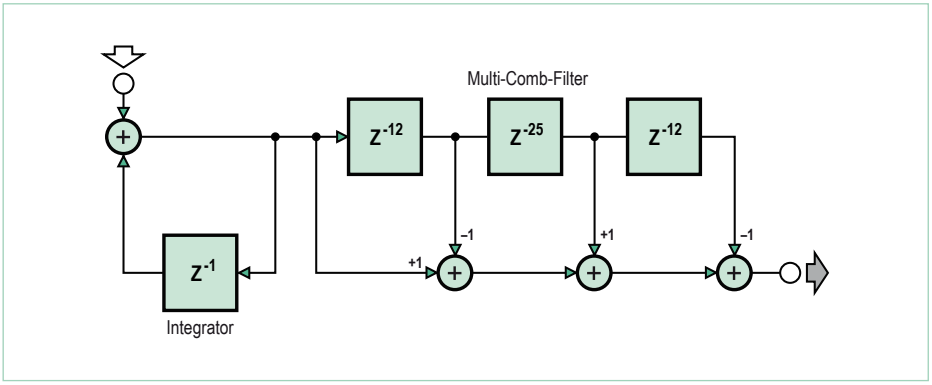


Figure 7. Un « filtre adapté » réalisé comme un filtre CIC.

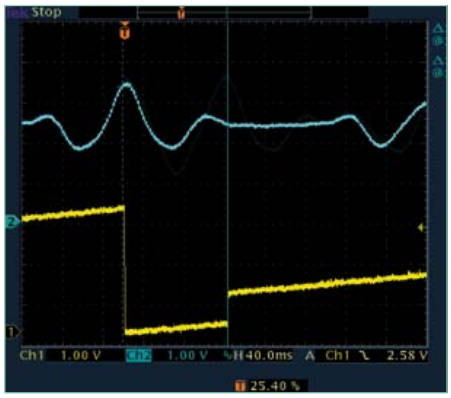


Figure 8. Le signal de sortie du filtre adapté pendant la réception d'un 0.

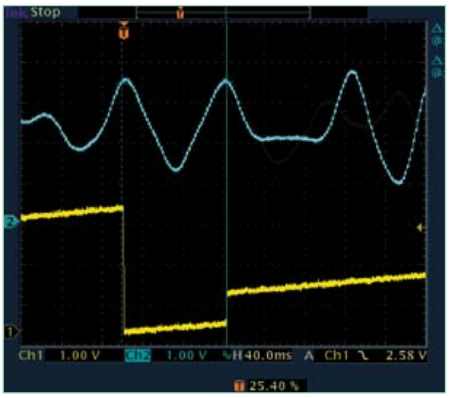


Figure 9. Le signal de sortie du filtre adapté pendant la réception d'un 1.

dantes. La trace jaune représente la temporisation qui détecte la pause pendant la seconde 59. Le signal d'entrée est saisi au moment du saut dans le signal jaune, exactement là où se situent les maxima du signal.
Les zéros et les uns sont mémorisés dans un champ. Ces données seront alors décodées et affichées après la réception des derniers bits. Elles sont transmises en même temps à 19 200 bauds à l'interface sérielle. Le **listage 4** vous montre un exemple de ces données reçues.

Dans le prochain numéro, vous trouverez la dernière partie de cette série, nous y envisagerons entre autres la synchronisation de l'horloge de bits et le *Early Late Gate Synchronizer*. Finalement, nous décoderons l'émetteur BBC198 et nous découvrirons encore quelques procédés de décodage.

(120089 - version française : Robert Grignard)

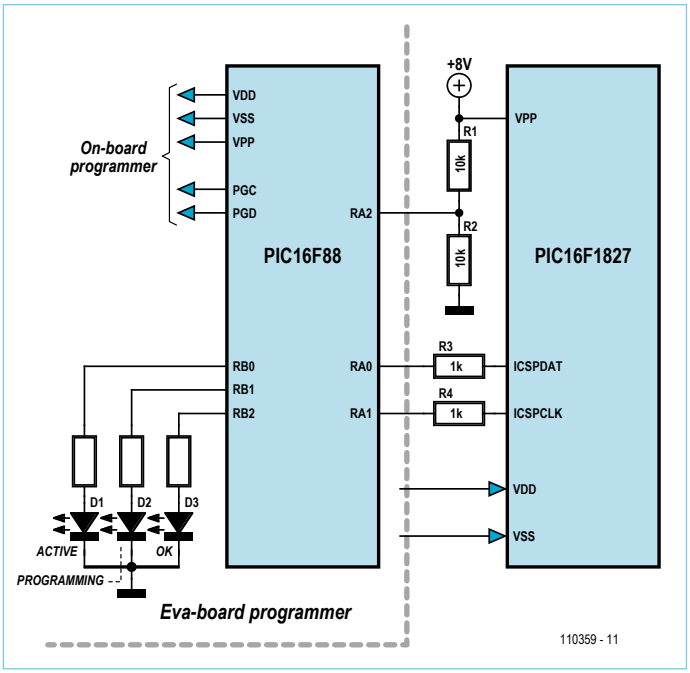
Listing 4 : Exemple de données de la station TDF162

```
TDF 162
0001110000000000001011001100100010011110101100100000000100000: 08:19 17.02 WEDNESDAY 2010 p000
0000110000000000001010000010100010011110101100100000000100000: 08:20 17.02 WEDNESDAY 2010 p000
0000110000000000001011000010000010011110101100100000000100000: 08:21 17.02 WEDNESDAY 2010 p000
0000110000000000001010100010000010011110101100100000000100000: 08:22 17.02 WEDNESDAY 2010 p000
0001110000000000001011100010100010011110101100100000000100000: 08:23 17.02 WEDNESDAY 2010 p000
```

Liens

[1] www.elektor.fr/100180
[2] www.elektor.fr/100181
[3] www.elektor.fr/100182
[4] www.elektor.fr/120088
[5] www.elektor.fr/120089
[6] http://hamsoft.ca/pages/mmtty.php
[7] www.dspguru.com/dsp/faqs/iir/basics ou http://fr.wikipedia.org/wiki/Filtre_à_réponse_impulsionnelle_infinie
[8] www.npl.co.uk/science-technology/time-frequency/time/products-and-services/msf-radio-time-signal
[9] www.dspguru.com/dsp/faqs/fir/basics ou http://fr.wikipedia.org/wiki/Filtre_à_réponse_impulsionnelle_finie

programmateur PIC de secours



Mode opératoire

- compiler le programme pour en faire un fichier ihex pour le PIC16F1827.
- convertir ce fichier ihex avec ihex2pic, disponible par [1].
- dans MPLAB, ouvrir SimProg.asm et vérifier si c'est le bon fichier *include* qui est appelé.
- installer le PIC16F88 sur la carte d'évaluation sans la brancher et raccorder le PIC16F1827 selon le schéma de la figure 1.
- programmer le PIC16F88 au moyen du programmateur de la carte d'évaluation.
- après programmation, la LED D1 s'allume. Appliquer la tension de 8 V.
- les LED D2 et D3 s'allument aussi. Attendre que D1 et D2 soient toutes deux éteintes. Si tout va bien, D3 est encore allumée, sinon il y a eu une erreur de programmation.
- débrancher le 8 V et l'alimentation de la carte d'évaluation. Le PIC16F1827 est prêt.

(110359 – version française : Robert Grignard)

[1] www.elektor.fr/110359

Elbert Jan van Veldhuizen (Pays-Bas)

J'avais acheté une série de microcontrôleurs PIC16F1827 au jeu d'instructions étendu. Les programmateurs disponibles ne les reconnaissaient pas et les mises à jour du logiciel tardaient à venir. Si vous disposez d'une carte d'évaluation avec programmateur, il existe un moyen facile de programmer quand même les nouveaux contrôleurs. Il n'y faut que quatre résistances, une alimentation externe et un PIC supplémentaire programmable avec la carte existante.

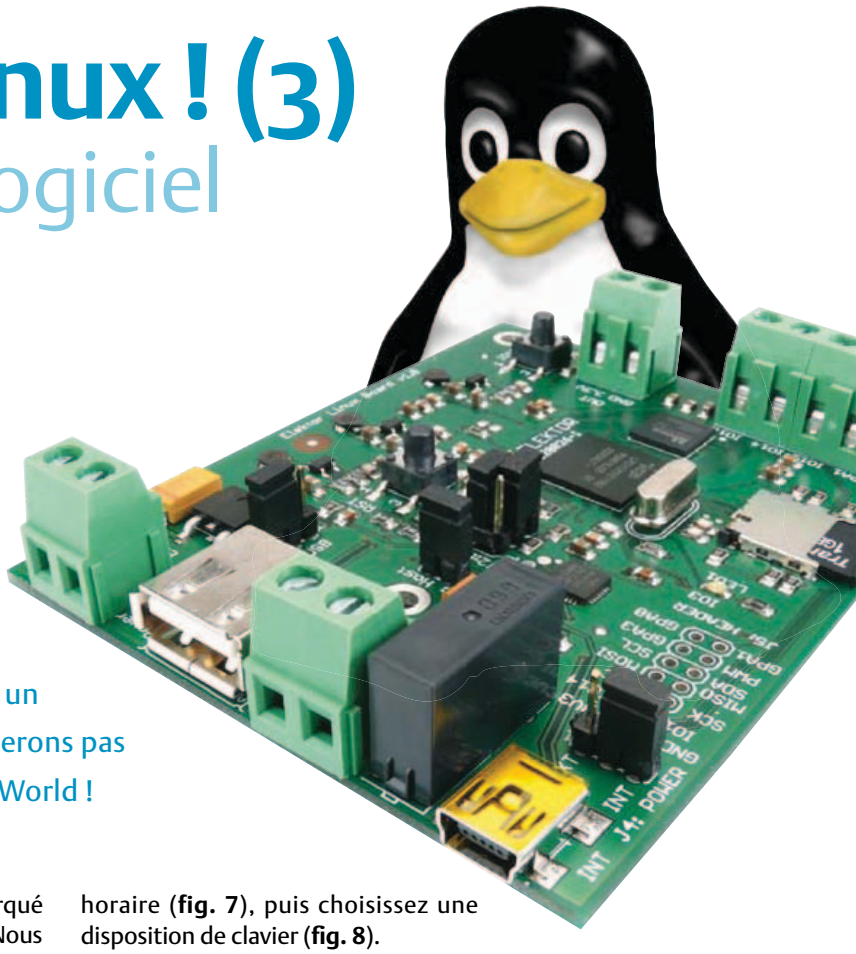
Elle reconnaissait en effet un PIC16F88 que j'avais en stock. J'ai écrit un petit logiciel d'environ 500 mots pour programmer le PIC16F1827 ; il est à présent disponible sur le site d'Elektor [1]. Il fallait encore un logiciel sur le PC pour convertir le fichier ihex pour PIC16F1827 en un fichier *include* pour le PIC16F88. Ces données sont alors placées dans le reste de la mémoire à programme disponible, environ 3 500 mots, du PIC16F88 qui seront lues ensuite comme données brutes.

Trois LED sur la carte d'évaluation révèlent le stade de la programmation. La LED D1 dit que le contrôleur est prêt pour la programmation. Il faut alors brancher la tension de programmation de 8 V sur le PIC16F1827. Les LED D2 et D3 indiquent que l'opération est en cours. D3 s'éteint dès que survient une erreur. D1 et D2 restent toutes deux éteintes une fois l'opération terminée. Si aucune erreur ne s'est produite, D3 reste allumée. On peut alors retirer la tension de 8 V et l'alimentation de la carte d'évaluation, le PIC16F1827 est prêt.

Les résistances R1 et R2 du schéma permettent au PIC de détecter si la tension de 8 V est présente. Les résistances R3 et R4 ne sont pas vraiment indispensables, elles servent de protection quand les deux processeurs, à cause d'une faute de programmation, envoient des données simultanément, ce qui peut provoquer un court-circuit, ou si la puce est branchée à l'envers.

Embarquez Linux ! (3)
Développement logiciel

Benedikt Sauter [1]



L'éveil d'un microcontrôleur vient d'une étincelle logicielle. Dans le monde de l'embarqué GNU/Linux, outre l'habituel micrologiciel, il est également nécessaire de créer les composants du système d'exploitation, un processus au menu de cet article. Et nous n'oublierons pas de saluer le monde à l'aide du traditionnel Hello World !

Il est en général plus facile de développer un système embarqué Linux depuis un PC lui-même équipé d'une distribution Linux. Nous vous proposons ici d'adopter la version 12.04 d'Ubuntu [2]. Son installation ne nécessite guère plus qu'un espace suffisant sur le disque dur et, idéalement, une connexion Internet.

Commencez par télécharger l'image du CD d'installation [2]. Ubuntu « Desktop » existe en versions 32 bits et 64 bits. Choisissez la version 32 bits si vous ne connaissez pas le type de votre processeur. L'étape suivante consiste à graver l'image du CD, donc sélectionnez l'option « Graver une image disque/ISO » de votre logiciel de gravure. Sous Windows 7, il suffit de sélectionner l'image ISO, puis de cliquer sur *Graver l'image disque*.

Insérez le CD gravé dans votre lecteur de disque, puis amorcez l'ordinateur depuis ce lecteur (en modifiant si nécessaire l'ordre d'amorçage défini dans le BIOS). La **figure 1** montre le premier écran affiché par Ubuntu. Dans notre cas, travailler avec un système installé sur le disque dur est plus simple, donc choisissez la seconde option du menu (**fig. 2**). L'installation se déroule par étapes. Sélectionnez d'abord la langue de votre choix (**fig. 3**). Inutile ensuite de cocher la case d'installation des logiciels tiers (**fig. 4**). L'écran suivant (**fig. 5**) vous demande si Ubuntu doit ou non occuper tout le disque dur. Cet écran n'apparaît que sur les disques sans système d'exploitation présent. Dans le cas contraire, vous pouvez soit partitionner votre disque, soit en utiliser un second. Cliquez sur le bouton d'installation une fois l'espace disque alloué (**fig. 6**). Sélectionnez ensuite votre fuseau horaire (**fig. 7**), puis choisissez une disposition de clavier (**fig. 8**).

Nous expliquerons un peu plus loin pourquoi l'étape suivante (**fig. 9**) demande un nom d'utilisateur et un mot de passe. Si tout s'est déroulé sans accroc, un diaporama (**fig. 10**) vous aide à patienter jusqu'à la fin du processus d'installation.

Si l'idée de partitionner votre disque dur vous effraie, vous préférerez peut-être exécuter Linux sur une machine virtuelle. L'auteur a justement conçu un système Linux pour le développement. Vous pouvez télécharger son image depuis la page web associée à cet article [3]. Comme logiciel de virtualisation, nous vous suggérons le programme gratuit VirtualBox [4]. Sous ce logiciel, l'importation de l'image se fait depuis le menu *Fichier → Importer une application virtuelle*. Une fois la machine importée, cliquez sur le bouton *Démarrer* pour la lancer.

Figure 1. Sélectionnez *Install Ubuntu* à l'aide des touches de direction.

Figure 2. *Install Ubuntu* est sélectionné : validez.

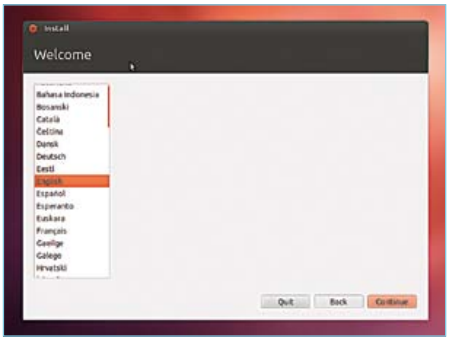


Figure 3. Sélection de la langue.

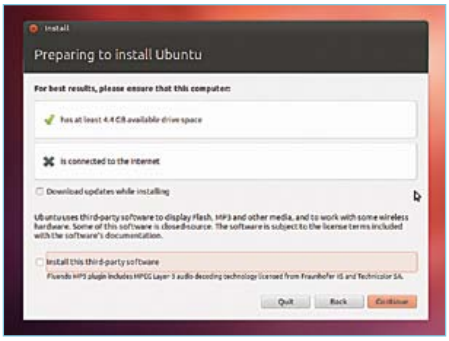


Figure 4. Préparation de l'installation.

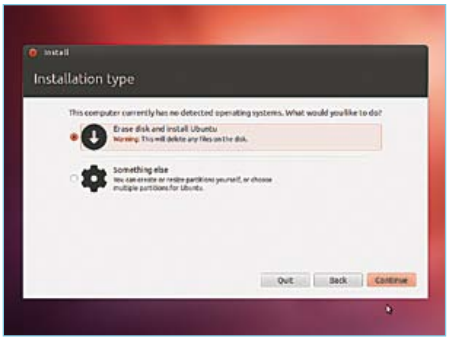


Figure 5. Allocation de l'espace disque.

Monter le « CD »

Installons maintenant la chaîne d'outils. Inutile de suivre les étapes de cette section et de la suivante si vous avez opté pour l'image VirtualBox de l'auteur, la chaîne d'outils y est déjà installée.

Sous Linux, le moyen le plus rapide d'accomplir une tâche est bien sûr de passer par la console. Ouvrez-la avec la combinaison de touches Ctrl-Alt-T.
Placez-vous ensuite dans le répertoire */tmp* en tapant la commande suivante :

```
cd /tmp
```

L'utilitaire *wget* va nous servir à télécharger l'image du CD contenant la chaîne d'outils ARM :

```
wget ftp://ftp.denx.de/pub/eldk/5.0/iso/armv5te-qte-5.0.iso
```

Nous pouvons lire le contenu de ce « CD » sans avoir à le graver. Pour cela nous devons d'abord le « monter », un verbe que l'on pourrait traduire ici par « attacher le système de fichiers du support de données au système de fichiers du système d'exploitation ».

Commençons donc par choisir le répertoire qui servira de point de passage vers le CD. Ce répertoire, appelé point de montage, peut en principe être choisi de façon arbitraire. Nous le placerons toutefois sous */media*, puisque sous Ubuntu il s'agit du répertoire standard pour les points de montage des médias amovibles. Linux respecte en effet une norme sur la hiérarchie des systèmes de fichiers qui facilite la localisation des fichiers. Nous reviendrons plus tard sur cette arborescence. Plaçons-nous donc dans le répertoire */media* :

```
cd /media
```

Créons ensuite le point de montage, un répertoire que nous appellerons *eldk-iso*. Que

Figure 6. Un choix radical.

Figure 7. Vous habitez dans le coin ?

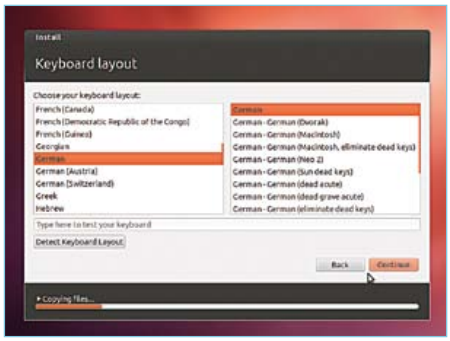


Figure 8. Disposition du clavier.

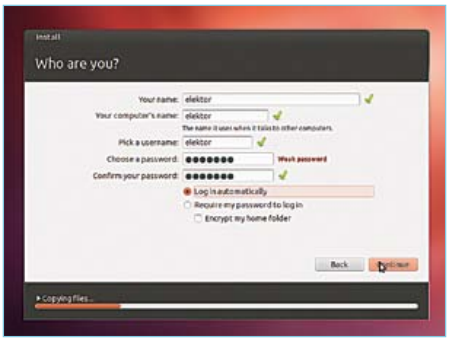


Figure 9. Création du compte utilisateur.

Figure 10. Un diaporama pour patienter.

Linux et les droits d'accès

Il existe sous Linux un utilisateur privilégié appelé super-utilisateur, ou utilisateur *root*. Cet utilisateur possède tous les droits, les autres ne jouissant que de droits d'accès limités. Le montage d'un système de fichiers nécessite par exemple des droits d'administrateur. Sous peine que cela ne devienne une mauvaise habitude, oubliez d'emblée l'idée de ne travailler qu'en tant qu'utilisateur root pour vous simplifier la vie : Linux est un système sûr justement parce qu'un utilisateur aux droits limités ne peut pas causer de dommages irréparables. Un utilisateur normal ne doit se voir accorder des droits que pour l'exécution des programmes nécessaires à son travail, mais en aucun cas ne doit pouvoir intervenir sur les fichiers du système ou autres données vitales.

En tant que programmeurs Linux, nous pouvons toujours circonscrire les permissions de l'utilisateur de façon à ce qu'il ne soit pas forcé de travailler en permanence en tant que *root*. Avec notre carte Linux-Elektor, nous créerons un compte utilisateur sous lequel nous travaillerons, et qui sera p. ex. autorisé à lancer les programmes que nous aurons écrits.

Pour les tâches qui nécessitent des droits d'administrateur, p. ex. la création d'un dossier dans un répertoire du système, la plupart des distributions Linux (dont Ubuntu) possèdent un utilitaire dit sudo : une commande qui commence par sudo est une commande qui sera exécutée avec des droits d'administrateur, le mot de passe à entrer étant celui du compte de l'utilisateur (ou *elektor* en minuscules si vous utilisez l'image Elektor-VirtualBox).

l'utilisateur habitué à Windows ne se méprenne pas : il ne s'agit pas ici de copier l'image du CD dans *eldk-iso*, mais bien de définir le point par lequel le système accédera à son contenu. Eh oui, sous Linux tout est symbolisé par des fichiers et des répertoires !

La création d'un dossier dans le répertoire */media* nécessite des droits d'administrateur (voir **l'encadré** « Linux et les Droits d'accès »). Nous demanderons donc au système de nous accorder ces droits grâce à l'utilitaire *sudo*. Le système vérifiera que nous disposons d'un compte privilégié en nous demandant un mot de passe. Ce compte est par défaut celui qui a été créé au moment de l'installation d'Ubuntu, donc validez la commande suivante pour créer le répertoire *eldk-iso*, puis entrez votre mot de passe :

```
sudo mkdir eldk-iso
```

Si vous utilisez l'image Elektor-VirtualBox, le mot de passe pour toute commande *sudo* est : elektor (en minuscules).
Attachons le système de fichiers du CD à celui du système d'exploitation :

```
sudo mount -o loop /tmp/armv5te-qte-5.0.iso /media/
eldk-iso
```

Nous pouvons maintenant accéder au contenu du CD depuis le point de montage :

```
cd eldk-iso
```

Installer la chaîne d'outils

Le CD contient un script d'installation de la chaîne d'outils. L'installation nécessite elle aussi des droits d'administrateur :

```
sudo ./install.sh -s -i qte armv5te
```

Cette commande a pour sortie :

```
*** Installing ./targets/armv5te/eldk-eglibc-i686-
arm-toolchain-qte-5.0.tar.bz2
```

Nous pouvons détacher le CD une fois la routine exécutée. Quittons d'abord le répertoire en cours :

```
cd ..
```

Le système de fichiers du CD aurait en effet été considéré comme encore actif si nous n'étions pas revenus au répertoire parent, et le système d'exploitation aurait retourné un message d'erreur si nous avions voulu le démonter, ce que nous pouvons maintenant faire :

```
sudo umount /media/eldk-iso
```

Le point de montage peut désormais être supprimé :

```
sudo rmdir eldk-iso/
```

Le script a installé la chaîne d'outils dans différents sous-répertoires du dossier */opt/eldk-5.0/*. Pour pouvoir appeler ces outils depuis n'importe quel répertoire, nous devons enregistrer leurs chemins d'accès dans la variable PATH, la variable d'environnement qui contient les chemins d'accès vers les exécutables du système.
Nous ne voulons ajouter ces chemins que pour la session en cours, donc nous le faisons à l'aide d'un script, que l'on appellera depuis la console au début de chaque session. Servons-nous de l'éditeur de texte gedit pour créer et ouvrir un fichier appelé *set.sh* :

```
gedit set.sh
```

Copiez-y le contenu suivant :

```
#!/bin/bash

    P1=/opt/eldk-5.0/armv5te/sysroots/i686-oesdk-linux/
usr/bin/armv5te-linux-gnueabi/
    P2=/opt/eldk-5.0/armv5te/sysroots/i686-oesdk-linux/
bin/armv5te-linux-gnueabi/


    export ARCH=arm
    export CROSS_COMPILE=arm-linux-gnueabi-
    export PATH=$P1:$P2:$PATH
```

La dernière commande ajoute à la variable PATH les chemins définis par P1 et P2.

Sauvegardez ce script dans un dossier, p. ex. votre dossier d'utilisateur (ou */home/nom_utilisateur*, le répertoire dans lequel vous devriez vous trouver). Nous n'allons pas exécuter ce script mais le « sourcer ». De cette façon le script sera lu et exécuté dans l'environnement du shell en cours (l'interpréteur de commandes), et non pas dans un shell fils ; les variables d'environnement resteront ainsi définies durant toute la session de la console :

```
. ./set.sh
```

Le premier point est une abréviation de la commande *source*, donc ne l'oubliez pas !

Si vous ne voulez pas avoir à sourcer le script à chaque ouverture d'une nouvelle console, ouvrez le fichier caché *.bashrc* de votre dossier personnel (commande `cd` pour vous y rendre depuis n'importe où) avec la commande `gedit .bashrc`, puis copiez-y le contenu du script. Le fichier *.bashrc* est lu automatiquement à chaque lancement d'un nouveau shell.

Quelques lignes pour occuper le compilateur

Les programmes de la chaîne d'outils pour la création du noyau Linux et du chargeur de démarrage sont préfixés par *armv5te-*. Pour connaître le numéro de version du compilateur GCC (appelé *armv5te-gcc)*, entrez, en notant bien la présence de deux tirets :

```
armv5te-gcc --version
```

Vous vérifierez par la même occasion que la variable PATH contient bien les chemins d'accès nouvellement définis.
Ce sont les outils préfixés par *arm-linux-gnueabi-* (et non *armv5te-* !) qui serviront à traduire nos fichiers source en exécutables Linux. Prenons l'exemple du classique « Hello world », en créant d'abord un fichier *hello.c* :

```
gedit hello.c
```

Entrez le contenu suivant :

```
#include <stdio.h>

int main(void)
{

  printf(«Hello World!\r\n»);
  return 0;

}
```

Cliquez sur *Enregistrer,* puis sur *Quitter*. Vous êtes à nouveau dans la console.

Compilons ce fichier sur le PC :

```
arm-linux-gnueabi-gcc -o hello hello.c
```

Copions maintenant le fichier *hello* sur la carte SD afin de nous assurer du succès de la compilation. Retirez la carte SD de la carte Linux-Elektor (débranchée), puis insérez-la dans un lecteur de cartes pour PC. Patientez un court instant sous Ubuntu après la connexion du lecteur et de la carte, jusqu'à l'ouverture automatique d'une fenêtre. Vous pouvez la fermer, nous passerons par la console pour copier le fichier sur la carte SD.

Un système d'exploitation comme Ubuntu est capable de monter lui-même un support de données externe. Placez-vous d'abord dans le répertoire */media* pour trouver le point de montage de la carte SD :

```
cd /media
```

La commande `ls` (un alias pour *list*) liste les dossiers présents :

```
ls
```

Vous pouvez visiter chacun d'entre eux avec :

Carte SD abîmée : les premiers soins.

Ne pas arrêter avec `halt` ou `poweroff` un système qui a été démarré depuis la carte SD augmente les risques de se retrouver avec un système de fichiers abîmé, ce que le système signale par un message du type « EXT2-fs errors » :

```
e2fsck 1.42 (29-Nov-2011)
/dev/sdh1 was not cleanly unmounted, check forced.
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/sdh1: 66/24576 files (0.0% non-contiguous), 8294/97988 blocks
```

```
Filesystem «EXT2-fs (mmcblk0p1): error: ext2_lookup:
deleted inode referenced: 694962»:
```

La restauration d'un système de fichiers abîmé est heureusement possible depuis une console Linux. Première étape, insérer la carte SD dans le lecteur de cartes du PC, puis chercher son nom de périphérique dans la sortie de `dmesg` :

```
[ 1549.424156] sd 7:0:0:2: [sdh] Assuming drive cache: write through [ 1549.425624] sdh: sdh1 sdh2 [
1549.427527] sd 7:0:0:2: [sdh] Assuming drive cache: write through [ 1549.427533] sd 7:0:0:2: [sdh] Attached SCSI
removable disk [ 1549.730223] EXT2-fs (sdh1): warning: mounting unchecked fs, running e2fsck is recommended
```

La partition abîmée étant la première, il s'agit dans cet exemple de *sdh1*. La sortie indique en outre qu'elle porte un système de fichiers ext2.

Étape suivante, détacher la partition touchée :

```
umount /dev/sdh1
```

L'utilitaire `e2fsck` permet enfin de restaurer le système de fichiers :

```
sudo e2fsck /dev/sdh1
```

(résultat identique avec `fsck.ext2`). La capture d'écran reproduit la sortie de cette commande. Dans certains cas, le programme demande à l'utilisateur si une certaine action doit ou non être effectuée ; répondez toujours « y » (pour *yes*) ou « o » (pour *oui* si le programme est francisé).

Et voilà, dites merci au pingouin !

```
cd <nom_du_dossier>
```

(et revenir au répertoire parent avec `cd ..`). Le nom du répertoire que vous cherchez, celui qui permet d'accéder au contenu de la carte SD, est composé de plusieurs chiffres. Il contient le système de fichiers du système Linux, par exemple les fichiers *zImage* et *swapfile1*.
Rappelons aux égarés que la commande `cd` ramène au répertoire personnel. Autre commande utile, `pwd` (pour *print working directory*), qui affiche le chemin complet du répertoire en cours.

Salut tout le monde !

Restez dans le dossier de votre carte SD pour y copier le fichier hello :

```
cp ~/hello ./
```

Le répertoire doit ensuite être détaché manuellement afin que le système d'exploitation écrive bien sur la carte mémoire tous les blocs qui pourraient encore se trouver en zones tampon :

```
sudo umount /media/<nom_du_dossier>
```

<nom_du_dossier> est comme ci-dessus le nom du dossier attribué par le système d'exploitation à votre carte SD.
Insérez-la à nouveau dans la carte Linux-Elektor, puis démarrez-la. Connectez enfin Linux et la carte depuis un émulateur de terminal, p. ex. picocom, comme nous l'avons vu dans l'article précédent [5]. Depuis le terminal, placez-vous dans le répertoire racine du système de fichiers de la carte :

```
cd /
```

Appelez le programme :

```
./hello
```

La sortie devrait être la ligne « Hello World ! » (**fig. 11**).

Pensez toujours, après un démarrage depuis la carte SD et une fois votre travail terminé, à bien arrêter la carte avec la commande `halt` :

```
halt
```

Ce n'est qu'après l'affichage du message `System halted` que vous pouvez la mettre hors tension. Sans cette précaution, la table du système de fichiers de la carte SD risquerait d'être altérée, ce qui sera signalé au démarrage par un message du type « EXT2-fs error ». Cela dit, restaurer un système de fichiers corrompu est possible sous Linux (voir **encadré**).

Chargeur de démarrage et noyau

Nous voici prêts à compiler le chargeur de démarrage (*bootloader*) et le noyau du système d'exploitation.

Les indispensables codes source (290 Mo) peuvent être téléchargés depuis le site web d'Elektor [3]. Firefox, le navigateur par défaut d'Ubuntu, place généralement les fichiers téléchargés dans le dossier *Téléchargements*. Vous pouvez aussi utiliser `wget` en ligne de commande.

À supposer que l'archive *120026-11.zip* ait été placée dans le dossier *Téléchargements*, vous pouvez la déplacer dans votre répertoire personnel avec :

```
mv ~/Téléchargements/120026-11.zip ~/
```

Revenez à votre dossier personnel avec `cd`, puis décompressez l'archive :

```
unzip 120026-11.zip
```

La **figure 12** montre la sortie de la console.

Créer le chargeur de démarrage

Après mise sous tension du processeur, le chargeur de démarrage stocké dans la carte SD est copié dans la SRAM du LPC3131 (voir [5] pour le positionnement correspondant du cavalier). Avant compilation du chargeur et son transfert sur la carte SD pour démarrer depuis celle-ci, deux paquets Linux doivent être installés :

```
sudo apt-get install patch libncurses5-dev
```

Plaçons-nous ensuite dans le dossier suivant :

```
cd  ElektorLinuxBoardDownload_20120509
```

Décompressons l'archive tar contenant le chargeur de démarrage :

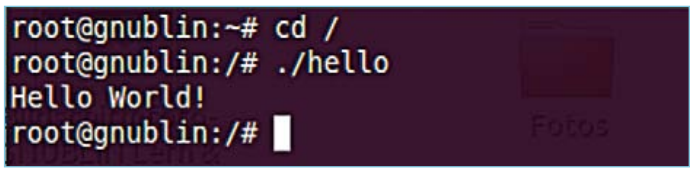


Figure 11. Salut les embarqués.

```
tar xvzf bootloader.tar.gz
```

Positionnons-nous dans le dossier *bootloader* nouvellement créé :

```
cd bootloader
```

Il contient le code source, que nous décompressons :

```
tar xvzf apex-1.6.8.tar.gz
```

L'étape qui suit est importante car elle permettra plus tard de transmettre les modifications apportées au code source. Nous allons créer une « copie de travail » du code source original, copie sur laquelle nous effectuerons nos changements. Il sera ainsi facile de créer un correctif (*patch*) et de le partager.

```
mv apex-1.6.8 work_1.6.8
```

```
cd work_1.6.8
```

Appliquons les correctifs nécessaires au fonctionnement de la carte Linux-Elektor :

```
patch -p1 < ../apex-1.6.8_lpc313x.patch
patch -p1 < ../gnublin-apex-1.6.8.patch
```

Nous nous servirons d'un fichier de configuration pour compiler le chargeur de démarrage. Nous devons d'abord copier ce fichier dans notre répertoire de travail, puis le renommer en *.config* (n'oubliez pas le point de *.config*, il signale au système d'exploitation qu'il s'agit d'un fichier caché, le plus souvent un fichier de configuration) :

```
cp ../gnublin-apex-1.6.8.config .config
```

Si vous avez installé la chaîne d'outils mais n'avez pas entré les variables d'environnement dans le fichier *.bashrc*, appelez maintenant le script *set.sh* (`. ~/set.sh`).

Lançons la compilation :

```
make apex.bin
```

Nous pourrions maintenant transférer le micrologiciel (le

chargeur) dans la mémoire flash du microcontrôleur à l'aide d'un programmateur, mais sous Linux l'écriture sur la carte SD est une opération facile.
Insérez donc une nouvelle fois la carte SD dans le lecteur de cartes de votre PC. La commande :

```
dmesg
```

(pour *display message*) affiche les messages du noyau. La **figure 13** montre la sortie de `dmesg` sur le PC de l'auteur : la carte SD a été reconnue comme étant le périphérique */dev/sdh* et contient deux partitions, *sdh1* et *sdh2*.

Comme nous n'allons pas écrire le micrologiciel dans le système de fichiers mais directement sur les blocs de la carte, il est plus prudent de démonter ces deux partitions :

```
sudo umount /dev/sdh1
sudo umount /dev/sdh2
```

Pour que le chargeur trouve le LPC3131 au démarrage, nous créons une copie physique du fichier *apex.bin* sur la carte SD avec la commande `dd` suivante (à copier à l'identique !) :

```
sudo dd if=src/arch-arm/rom/apex.bin of=/dev/sdh2 bs=512
```

L'option `of` (*ouput file*) spécifie ici le répertoire de destination, tandis que `bs` (*block size*) définit en octets la taille des blocs d'écriture. Sous Linux, il est impossible de savoir quand les blocs modifiés seront effectivement écrits dans la mémoire NAND de la carte SD. La commande :

```
sync
```

forcera donc le système d'exploitation à écrire les données qui auraient pu encore se trouver dans des zones tampon. Nous sommes à présent certains que le nouveau chargeur se trouve bien sur la carte SD, ce que vous pouvez néanmoins vérifier en effectuant un test de démarrage.

Compiler le noyau

La compilation du noyau reprend les mêmes étapes que celles suivies pour la compilation du chargeur. Placez-vous dans votre dossier d'utilisateur avec la commande `cd`, puis dans le dossier contenant les sources :

```
cd  ElektorLinuxBoardDownload_20120509
```

Décompressez le code source du noyau :

```
tar xvzf linux-2.6.33-lpc313x-gnublin-032012.tar.gz
```

Positionnez-vous dans le dossier du noyau :

```
cd linux-2.6.33-lpc3131x
```

Nous lancerons la compilation avec l'option *zImage*. Cette option permet d'obtenir une image compressée et amorçable du noyau :

```
make zImage
```

Compiler les modules permet d'obtenir des pilotes rechargeables :

```
make modules
```

Modules et noyau compilés doivent être copiés sur la carte SD. Un noyau compressé *zImage* y est déjà présent, mais le remplacer par celui que vous venez de créer vous familiarisera avec cette opération. Écrivez donc le fichier *arch/arm/boot/zImage* sur la première partition de la carte, sans oublier de la démonter à la fin de votre travail.
C'est bon, vous vous en êtes sorti tout seul ?

Pour vous allécher

Et pourquoi pas également modifier le noyau, maintenant que nos savons le compiler ? Sans entrer dans les détails, nous pouvons nous faire une première idée de la méthode de configuration d'un noyau Linux en lançant la commande suivante depuis le dossier *linux-2.6.33-lpc3131x* :

```
make menuconfig
```

La **figure 14** montre l'outil de configuration appelé par cette commande. C'est à travers ce menu que l'on activera par exemple le pilote d'un périphérique USB.
Les touches de direction servent à naviguer parmi les différentes caractéristiques du noyau et les commandes du menu. La touche Entrée ouvre/ferme un sous-menu, ou active une commande. Si d'aventure vous parcourez ce menu, vous tomberez certainement sur un pilote de périphérique que vous connaissez. Nous explorerons cet outil plus avant dans le prochain article.

Sauvegarder la carte

Travailler sous le capot expose le moteur, aussi est-il plus prudent de mettre de côté une copie conforme de la carte SD.
Rappelons que pour trouver le nom attribué par le système d'exploitation à votre carte, vous pouvez lancer `dmesg` après l'avoir connectée au PC.
Par précaution, démontons une nouvelle fois les partitions :

```
umount /dev/sd<lettre>1
umount /dev/sd<lettre>2
```

(remplacez <lettre> par la lettre qui correspond au nom de périphérique de votre carte, p. ex. */dev/sdb1* et */dev/sdb2*, ou */dev/sdh1* et */dev/sdh2*).

Réutilisons la commande dd pour créer une image « physique » de la carte :

```
sudo dd if=/dev/sd<lettre>
of=Copie_Sauvegarde_Carte_SD.img
```

L'opération prend un certain temps. La commande ci-dessous permet de vérifier que copie et original sont de même taille :

```
ls -lh
```

Insérons maintenant dans le lecteur une carte SD de capacité identique. Là encore vous pouvez appeler dmesg pour connaître son nom de périphérique.
La commande suivante écrit le fichier image sur la nouvelle carte :

```
sudo dd if=Copie_Sauvegarde_Carte_SD.img of=/dev/
sd<lettre>
```

Cette fois-ci nous devons attendre un peu plus longtemps avant que le système nous informe du succès de l'écriture. Comme plus haut, nous lançons ensuite sync pour être certains que tous les blocs auront bien été écrits sur la carte SD, carte que nous pouvons tester depuis la carte Linux-Elektor. Inutile avant cela de lancer umount, puisque nous n'avons pas monté le système de fichiers (dd était une écriture physique directe).

Cette procédure de sauvegarde est simple, mais échoue malheureusement lorsque les cartes ont des capacités différentes. Il arrive aussi que l'on ait besoin de créer une carte contenant des partitions et un système de fichiers différents de ceux de la carte originale. Il existe pour cela un installateur graphique, auquel nous consacrerons quelques lignes dans le prochain article.

Perspectives

Nous avons parcouru ensemble une bonne partie du chemin qui mène à l'écriture d'applications Linux pour l'embarqué : l'environnement de développement est posé, noyau et chargeur de démarrage sont compilés, et nous avons même vu, certes sur un exemple modeste, comment écrire et lancer un programme.

Lors de notre prochaine étape, nous verrons comment est structuré un code source écrit pour Linux, avec pour objectif d'être au moins capables d'ajouter un pilote à n'importe quel périphérique. Nous verrons également combien les langages de script simplifient la programmation !

(120180 – version française : Hervé Moreau)

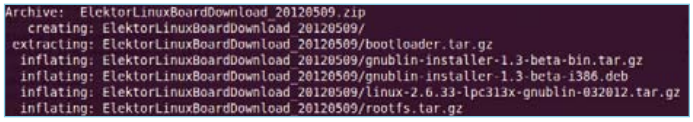


Figure 12. La sortie de la commande unzip.

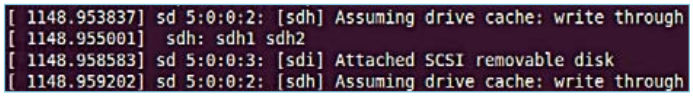


Figure 13. Sur le PC de l'auteur, la carte SD a été identifiée en tant que périphérique /dev/**sdh**.

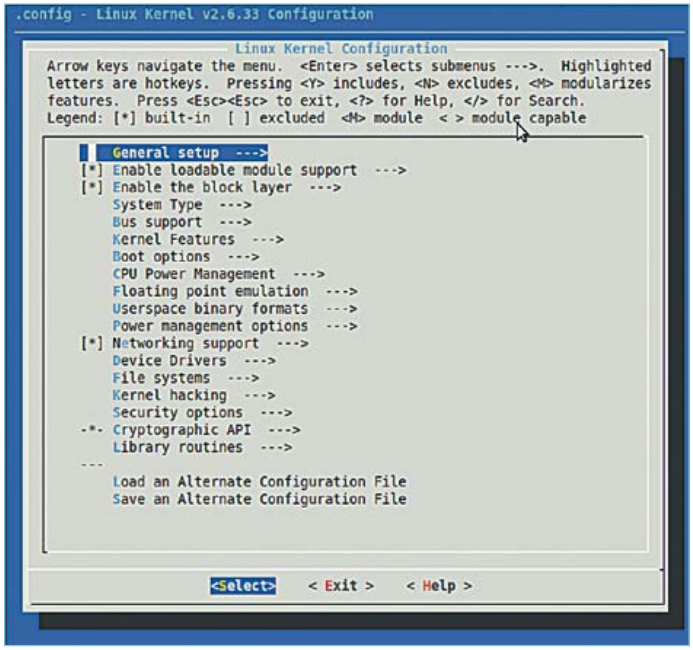


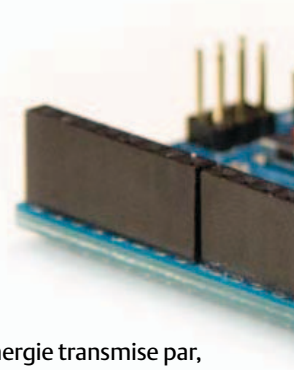
Figure 14. Configuration du noyau Linux.

Liens Internet

[1] sauter@embedded-projects.net
[2] www.ubuntu.com
[3] www.elektor.fr/120180
[4] www.virtualbox.org
[5] www.elektor.fr/120146

mettez le cap sur Arduino

1b: *Andante maestoso*



David Cuartielles (Espagne)

Production de son à 1 bit … gné ?!?

Dans la première partie, nous avons expérimenté la production de son à l'aide du langage Arduino. Cette fois-ci, nous allons triturer le processeur de l'Arduino Uno (un ATmega328) jusque dans ses entrailles, créer un outil capable de stocker du son en provenance d'un fichier WAV dans la mémoire de programme ainsi que le code pour le relire, bien entendu.
La méthode utilisée n'est pas triviale. La clé du problème est de considérer le son du point de vue de son spectre. Ne craignez rien, ce spectre-là ne devrait (normalement) pas vous hanter…

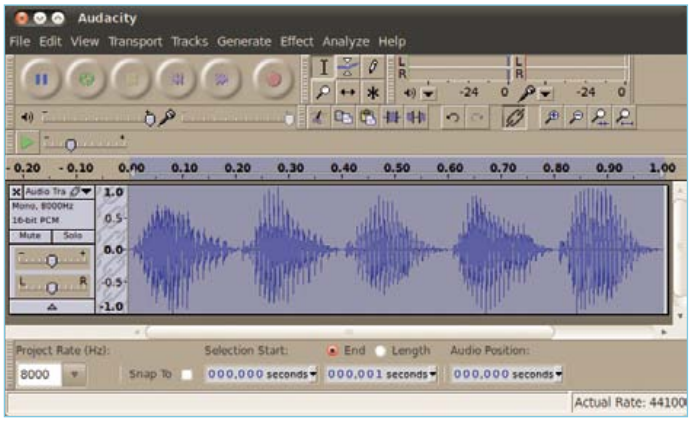


Figure 1. Enregistrement d'un humain disant : "ta-te-ti-to-tu".

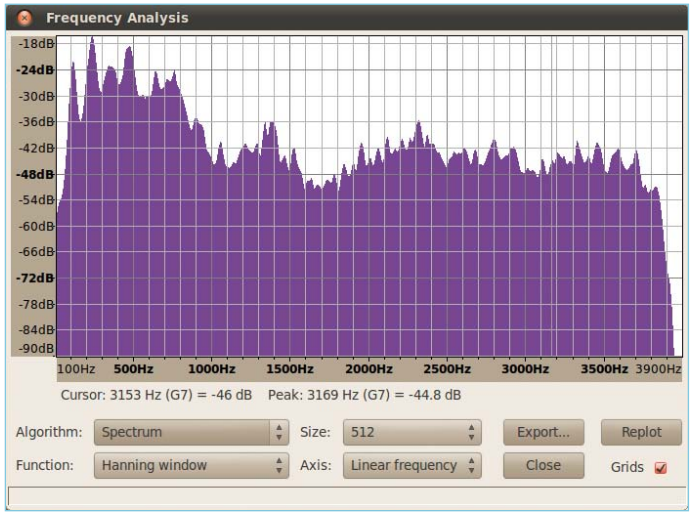


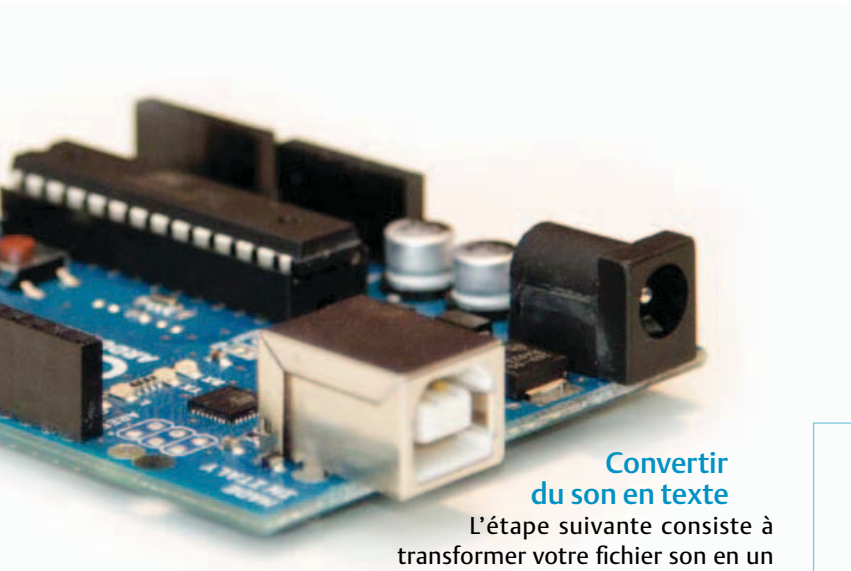
Figure 2. Spectre de l'enregistrement de la figure 1.

Le spectre sonore

Le spectre sonore est une représentation de l'énergie transmise par, disons, un haut-parleur en fonction de la fréquence. En regardant le spectre d'un son, il est difficile de se le représenter aussi clairement qu'avec la **figure 1**. Le graphique représente la quantité d'énergie pour différentes fréquences sur une certaine durée. On représente habituellement le spectre pour une chanson entière (voir **figure 2**). Mais on peut également jeter un coup d'œil au spectre pour seulement 0,5 s d'une chanson. Plus la fenêtre est petite et mieux le spectre rend compte du son à un instant *i*. Après quelques essais on s'aperçoit vite que deux signaux sonores différents peuvent avoir des spectres très similaires.
L'oreille humaine est stimulée par le contenu énergétique des sons, deux sons possédant des spectres identiques seront perçus comme tels, mais seulement si la résolution temporelle est suffisamment petite. Il s'agit là de la clé de voute des techniques de compression du son : la faculté d'obtenir des signaux qui sont « suffisamment bons » pour que nous comprenions le son, même si celui-ci est objectivement très différent de l'original.
Voici comment fonctionne la production de sons à 1 bit [1]. Il est possible de produire un son à l'aide d'un signal MLI dont le niveau moyen d'énergie est suffisamment similaire à celui du signal original.
Cette astuce mathématique permet de produire du son, certes de qualité moyenne, mais avec un petit microcontrôleur. Les paragraphes suivants vous montreront comment en tirer parti. Nous démarrerons avec un fichier WAV enregistré par exemple avec votre ordinateur. Puis, nous le filtrerons, et le transformerons en un tableau de nombres que nous pourrons stocker dans un Arduino.

Numérisation optimale et filtrage

Beaucoup d'outils différents existent pour enregistrer des sons. Je ne peux que vous recommander l'utilisation d'*Audacity* [2], un outil libre et à sources ouvertes qui fournit la plupart des options nécessaires à la reproduction de son avec des microcontrôleurs (les figures 1 et 2 sont des captures d'écran d'*Audacity*).
Avant d'aller plus loin, vous devriez filtrer le son. J'utilise un filtre passe-bas avec une fréquence de coupure de 4 kHz. Le lecteur de son présenté ici utilise une fréquence d'échantillonnage de 8 kHz (c.-à-d. 8000 échantillons par seconde), ce qui veut dire que les composantes fréquentielles situées au-dessus de 4 kHz dans le fichier d'origine n'auront pour effet que la production d'artefacts dans le son reproduit, théorème de Shannon oblige.
Il est possible, avec des microcontrôleurs, de produire des sons de qualité raisonnable, cependant ces puces ne disposent que de peu de mémoire. Il faudra donc utiliser des formats sonores de faible qualité qui permettront de stocker plusieurs secondes de sons sans recourir à une puce de mémoire externe.
Un très bon exemple est le format 8 bits PCM WAVE (également dénommé *Microsoft unsigned 8 bit*). Avec ce format, la qualité est suffisante pour reproduire, entre autres, la voix humaine. Pour les étapes suivantes, vous devrez exporter votre fichier en : mono, 8-bit, PCM WAVE.



Convertir du son en texte

L'étape suivante consiste à transformer votre fichier son en un fichier d'en-tête que vous pourrez inclure dans votre programme Arduino. L'ATmega328 possède 32 Ko de mémoire flash dont vous pourrez utiliser une partie pour stocker le fichier son. Il est possible de stocker de grandes quantités de données dans des tableaux en mémoire en utilisant la bibliothèque `Progmem` de la chaîne d'outils d'Atmel. Un son codé sur 8 bits n'est rien d'autre qu'un tableau de nombres entre 0 et 255. Vous pouvez le déclarer comme suit :

```
const unsigned char sounddata_data[] PROGMEM = {128,
   128, 128,
[...]
69, 62, 59, 57, 52, 50, 56, 65, 74, 86, 96, 109, 116, };
```

J'ai créé un outil pour l'EDI Arduino qui vous permettra d'incorporer directement des fichiers WAV à votre code. Vous pourrez le télécharger en suivant les liens donnés en référence ; il ne vous restera plus qu'à le décompresser et l'ajouter à votre dossier *sketchbook* ;un répertoire `tools/SoundData` y sera créé.
Après avoir redémarré l'EDI, une nouvelle entrée *SoundData* apparaîtra dans le menu *Tools*. Une fois l'outil lancé, une boîte de dialogue apparaît et vous permet de sélectionner un fichier WAV (**figure 3**). Le deuxième bouton, intitulé `Generate Code` déclenchera l'ouverture du fichier WAV, la vérification de son codage et ajoutera un nouvel onglet `sounddata.h` à votre code. Ce nouveau fichier contient tout ce dont vous avez besoin pour jouer votre fichier WAV sur 1 bit. Il devrait ressembler à ceci (**figure 4**) :

```
// soundData for your Arduino sound project
// automatically generated data
// - original sound file:/development/tmp/matis.wav
// - sampleRate:8000
// - encoding:8 bits
// - channels:1

const int sounddata_length = 7969;

const signed char sounddata_data[] PROGMEM = {127, 127,
   127, 127, 127, 127,
[...]
69, 62, 59, 57, 52, 50, 56, 65, 74, 86, 96, 109, 116, };
```

Avant de vous jeter sur le bouton `Generate Code` vous devriez continuer la lecture, j'ai encore des choses à vous dire !

Le lecteur

La lecture d'un son échantillonné n'est pas évidente : nous allons devoir trifouiller un peu le cœur d'Arduino. Bien sûr, il existe des

Figure 3. Une boîte de dialogue pour importer vos fichiers WAVE sous forme d'en-tête pour programmes Arduino.

bibliothèques pour l'Arduino Uno qui se chargent de faire tout ça pour vous, mais n'est-ce pas là l'occasion pour vous d'effectuer une plongée dans les entrailles d'Arduino et de voir comment cela est réalisé en bas niveau .
L'astuce pour jouer des échantillons sonores réside dans l'utilisation du mode *Fast PWM* des ATMega d'Atmel (il y a des fonctions similaires chez les autres fabricants). Il existe un registre qui permet de faire de la MLI jusqu'à la moitié de la fréquence d'horloge, impressionnant non ? Parmi tout ce que l'on peut faire avec ça, il y a, et c'est ce qui nous intéresse, la lecture de son sur un 1 bit. La seule limitation de ce mode est de ne fonctionner qu'à une résolution de 8 bits, c'est pour cela que nous avons dû convertir le fichier son dans ce format.
Pour faire fonctionner le tout, il faudra utiliser deux des trois temporisateurs internes :
Le premier cadencé à la fréquence d'échantillonnage (SAMPLE_

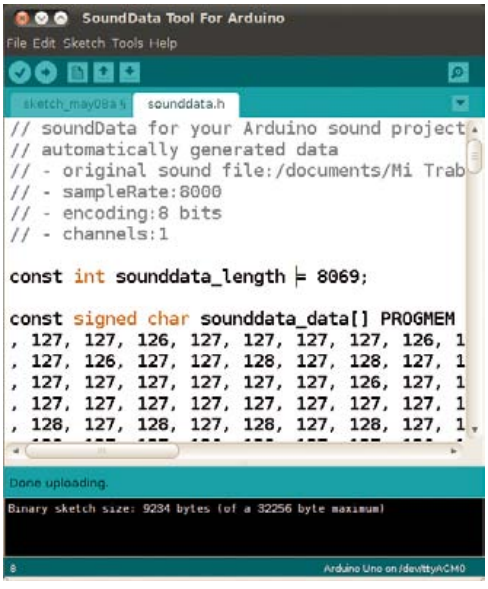


Figure 4. Notre "ta-te-ti-to-tu" dans l'EDI Arduino après l'import avec l'outil SoundData.

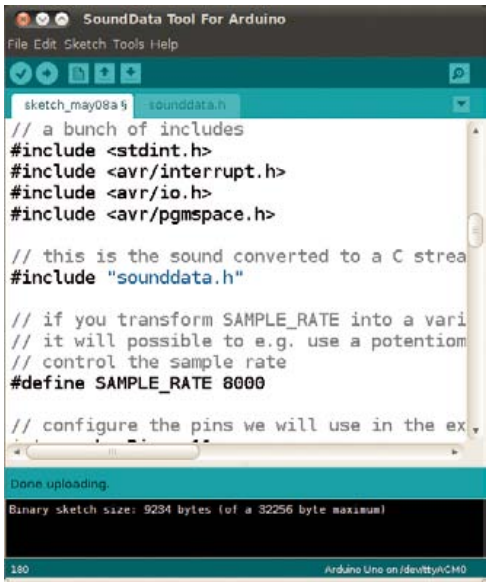


Figure 5. Capture d'écran de l'EDI Arduino après utilisation de *SoundData* ; on peut y voir le code de notre lecteur basique.

RATE) est utilisé pour aller chercher les échantillons dans la table `sounddata_data` array que nous venons de produire. Cette valeur réglera le rapport cyclique du signal MLI qui sortira sur la patte 11. Le second cadencé à 62500 Hz (16000000 / 256), c.-à-d. bien plus rapidement que la fréquence de lecture (par défaut 8 kHz). Comme expliqué précédemment, il sera chargé de produire un signal numérique dont le spectre sera très proche de celui du fichier son d'origine. En d'autres termes, en augmentant la fréquence MLI, le spectre du signal obtenu se rapprochera de celui de notre son. Dans la plupart des applications, c'est plus qu'assez pour jouer du son à moindre coût.

Important : en ajoutant ce code à votre programme Arduino, vous surchargerez certaines des commandes de base du langage Arduino. Par exemple, les commandes MLI ne fonctionneront plus sur certaines pattes, et la fonction `delay()` ne fonctionnera plus comme prévu. Cela n'affectera bien entendu que ce programme.
Cette technique a été présentée pour la première fois aux Arduineux en 2008 par Michael Smith à travers un exemple qui mimait le son de démarrage des ordinateurs Macintosh à chaque démarrage d'une carte Arduino Diecimila. Le code de Michael s'inspirait du travail de beaucoup d'autres ; jetez un œil aux liens [3].
Pour vous faciliter la vie, l'outil SoundData ne se contentera pas de produire les informations sonores, il inclura également le code d'un lecteur basique dans votre sketch. Assurez-vous toutefois que votre sketch soit vide avant de faire appel à l'outil : il écrasera tout ce qui se trouve dans l'EDI.
Le programme basique (**figure 5**) de lecture de son sur 1 bit utilise beaucoup de commandes de bas niveau pour piloter les temporisateurs. Je vais décrire les fonctions une à une. Commençons par le `TIMER1` :

```
ISR(TIMER1_COMPA_vect) {
    if (sample >= sounddata_length) {
        if (sample == sounddata_length + lastSample) {
            // condition testant s'il s'agit du dernier
    échantillon
            stopPlayback();
        } else {
            // On revient à zéro pour limiter le bruit
    en fin de lecture.
            OCR2A = sounddata_length + lastSample
    - sample;
        }
    }
    else {
        // OCR2A est le registre pour régler le rapport
    cyclique
        // de la MLI haute-fréquence de la patte 11
        // pgm_read_byte lit un octet depuis le tableau
    en mémoire programme
        OCR2A = pgm_read_byte(&sounddata_data[sample]);
    }

    // incrémentation du compteur d'échantillons
    ++sample;
}
```

`ISR` nous indique qu'il s'agit d'une routine de gestion d'interruption. Une interruption est un événement qui dit à la puce d'arrêter tout traitement pour s'occuper immédiatement de quelque chose de plus urgent. Il existe, sur les processeurs, des interruptions internes et/ou externes. En interne, il y a par exemple les temporisateurs ; en externe, un changement d'état d'une patte.
Ici, notre `ISR` s'occupe du traitement des événements du `TIMER1`, interne. À chaque événement du `TIMER1`, cette fonction effectuera les traitements suivants, en vrac :

- incrémenter le compteur utilisé pour parcourir les données sonores
- vérifier si la fin du tableau de données est atteinte ou non
- si non, charger le prochain échantillon
- si oui, effectuer un fondu du son vers zéro.

Les temporisateurs `TIMER1` et `TIMER2` sont tous deux initialisés par la fonction `startPlayback`. Regardons la tête qu'elle a :

```
void startPlayback()
{
    pinMode(speakerPin, OUTPUT);

    // Régler le Timer 2 pour faire de la MLI sur la
    patte du haut-parleur

    // Utilisation de l'horloge interne (p160 de la
    datasheet)
    ASSR &= ~(_BV(EXCLK) | _BV(AS2));
```

```
    // Choix du mode Fast PWM (p.157)
    TCCR2A |= _BV(WGM21) | _BV(WGM20);
    TCCR2B &= ~_BV(WGM22);

    // MLI non-inversée sur la patte OC2A (p.155)
    // Sur l'Arduino, c'est la patte 11.
    TCCR2A = (TCCR2A | _BV(COM2A1)) & ~_BV(COM2A0);
    TCCR2A &= ~(_BV(COM2B1) | _BV(COM2B0));

    // Pas de prédiviseur (p.158)
    TCCR2B = (TCCR2B & ~(_BV(CS12) | _BV(CS11))) |
   _BV(CS10);

    // La largeur d'impulsion initiale viendra du
   premier échantillon
    OCR2A = pgm_read_byte(&sounddata_data[0]);

    // Régler le Timer 1, un échantillon sera envoyé à
   chaque interruption
    cli();

    // Utilisation du mode CTC (Remise à zéro du timer
   sur comparaison) (p.133)
    // OCR1A doit être initialisé *après*, sinon il est
   remis à zéro !
    TCCR1B = (TCCR1B & ~_BV(WGM13)) | _BV(WGM12);
    TCCR1A = TCCR1A & ~(_BV(WGM11) | _BV(WGM10));

    // Pas de prédiviseur (p.134)
    TCCR1B = (TCCR1B & ~(_BV(CS12) | _BV(CS11))) |
   _BV(CS10);

    // Initialisation du registre de comparaison
   (OCR1A).
    // OCR1A est un registre à 16 bits, les
   interruptions doivent donc être
    // désactivées pour avoir l'esprit tranquille
    OCR1A = F_CPU / SAMPLE_RATE;    // 16e6 / 8000 =
   2000

    // Activer l'interruption lorsque TCNT1 == OCR1A
   (p.136)
    TIMSK1 |= _BV(OCIE1A);

    lastSample =
   pgm_read_byte(&sounddata_data[sounddata_length-1]);
    sample = 0;
    sei();
}
```

Bien que cette séquence de commandes de bas niveau soit expliquée dans le code, en voici un petit résumé qui sera sans doute utile aux débutants :

- régler la patte reliée au haut-parleur en sortie

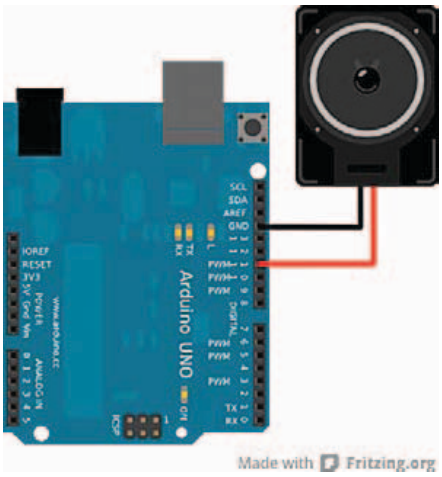


Figure 6. L'Arduino Uno est parfaitement capable d'attaquer directement un petit haut-parleur.

- configurer l'utilisation de l'horloge interne
- initialiser le mode Fast PWM
- configuration du `TIMER2` pour obtenir la MLI qui jouera le son, le rapport cyclique sera réglé en changeant la valeur de `OCR2A`
- chargement du premier échantillon sonore dans `OCR2A`
- arrêt des interruptions pour une seconde – `cli()`– pour configurer `TIMER1` l'esprit tranquille
- configuration de `TIMER1` pour charger les échantillons
- chargement du dernier échantillon sonore

```
void stopPlayback()
{
    // Arrêt de l'interruption pour les échantillons
    TIMSK1 &= ~_BV(OCIE1A);

    // Arrêt complet du temporisateur associé
    TCCR1B &= ~_BV(CS10);

    // Arrêt du temporisateur pour obtenir la MLI
    TCCR2B &= ~_BV(CS10);

    digitalWrite(speakerPin, LOW);
}
```

- réactivation des interruptions – `sei()`.

De la même manière, il va nous falloir une fonction pour arrêter les temporisateurs à la fin de la lecture.

```
void setup()
{
    startPlayback();
}

void loop()
{
    // rien à faire
}
```

Vous voici maintenant armé d'une série de fonctions que vous pourrez appeler n'importe où dans vos programmes pour jouer du son. Dans cet exemple, la fonction `startPlayback()` est appelée lors de la configuration et le son ne sera joué qu'une seule fois.

Le mot de la fin

Nous voici arrivés au terme de cette présentation des différentes manières de produire du son avec un Arduino. Je vous ai tout d'abord montré comment jouer des notes à l'aide des fonctions de base d'Arduino. Puis, je vous ai présenté des bibliothèques permettant de simplifier la lecture de mélodies simples sur des transducteurs piézo à bas-coût. Enfin, nous avons fait une petite plongée dans les techniques de programmation permettant de produire du son avec 1 bit à l'aide de *Fast PWM*.

Tous les bouts de code et outils dont il a été question sont disponibles dans un fichier ZIP [4], qui contient également des fichiers sons au bon format pour que vous puissiez essayer les différents exemples. J'ai également créé un outil pour l'EDI Arduino qui vous aidera à importer des fichiers WAVE de courte durée dans la mémoire programme de l'Arduino. Cet outil vous permettra de charger des sons et les relire, jouer sur la fréquence d'échantillonnage, les jouer à l'envers ou encore les trifouiller. Niveau matériel, tout ce qu'il vous faudra, c'est un haut-parleur relié comme sur la **figure 6**.

Ne vous arrêtez pas là ! Il y a beaucoup à explorer — par exemple utiliser la patte 3 en même temps que la patte 11 pour faire de la stéréo. Ou encore créer un synthétiseur à 8 bits capable de mélanger plusieurs sons sur un seul canal. Pourquoi pas sinon transformer votre carte Arduino en une carte son commandée par MIDI ?

Bonne bidouille et au mois prochain !

(120427 – version française : Kévin PETIT)

Liens

[1] **Convertisseurs N/A Sigma Delta à 1 bit :** www.digitalsignallabs.com/presentation.pdf

[2] **Audacity, logiciel libre et à sources ouvertes pour le traitement du son :** audacity.sourceforge.net

[3] **Quelques références tirées de l'exemple de Michael Smith :**
Manuel Arduino sur l'utilisation de la bibliothèque tone : arduino.cc/en/Reference/Tone
L'article d'origine sur Arduino Playground: arduino.cc/playground/Code/PCMAudio
www.uchobby.com/index.php/2007/11/11/arduino-sound-part-1/
www.atmel.com/dyn/resources/prod_documents/doc2542.pdf
www.evilmadscientist.com/article.php/avrdac
http://gonium.net/md/2006/12/27/i-will-think-before-i-code/
http://fly.cc.fer.hr/GDM/articles/sndmus/speaker2.html
www.gamedev.net/reference/articles/article442.asp

[4] **Les fichiers de cet article :** www.elektor.fr/120427

Tubes audio anciens & récents

Répertoire inédit de plus de 1500 marques de tubes, commenté par l'auteur

Il y a plus de cent ans, Lee de Forest eut la merveilleuse idée d'ajouter une troisième électrode, une grille, entre le filament et la plaque d'une lampe détectrice : l'Audion était né, et avec lui, débutait l'ère de l'électronique. Que de progrès depuis ! Et pourtant cette technologie vieille d'un siècle n'est pas encore obsolète : aujourd'hui, à l'heure des nanotechnologies et de la très haute intégration des circuits électroniques, Western-Electric fabrique toujours la triode 300B.

Quelle est donc cette magie des tubes de verre qui nous enchante encore ? Quels secrets les rendent irremplaçables à nos oreilles de mélomanes ? Rien que le savoir-faire d'hommes passionnés, transmis et enrichi sur plus de quatre générations.

Découvrez ici ce qui se cache au cœur des "lampes", à travers une centaine de photos et d'illustrations, un tour d'horizon des productions actuelles et un historique des fabricants anciens les plus connus.

Apprenez à reconnaître et à choisir les tubes NOS les plus réputés, grâce aux codes et à leurs particularités de construction.

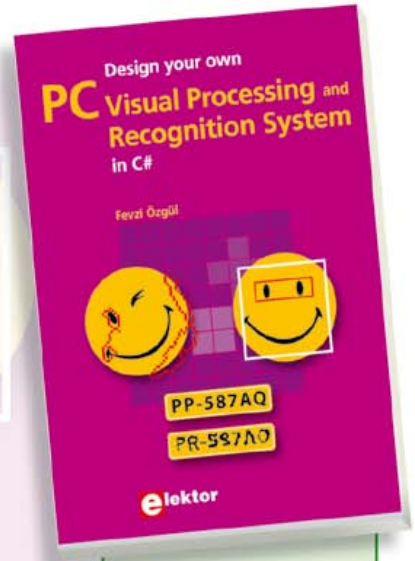
2e édition revue & augmentée

175 pages
Format 17 x 24 cm
ISBN 978-2-86661-185-9
43,50 €

Pour commander en ligne et bénéficier d'une remise spéciale de 5% :

www.elektor.fr/e-choppe

Design your own PC Visual Processing and Recognition System in C#



Nouveau livre d'Elektor en anglais

Reconnaissance d'objet, détection et identification de visage, reconnaissance optique de caractères, détection de présence et de mouvement. Voilà les sujets chauds, liés au traitement de l'image. L'apparition de nombreuses bibliothèques libres (*open source*) de traitement de l'image permet maintenant d'inclure dans les applications.NET des fonctions complexes d'imagerie.

Ce livre s'adresse aux ingénieurs, scientifiques et amateurs éclairés, compétents en programmation et intéressés par les techniques de traitement de l'image sur PC.

Il utilise le langage de programmation orienté objet C# de Microsoft, avec des exemples pratiques et utiles, pour vous permettre de développer des logiciels de traitement de l'image de grande qualité.

L'ouvrage commence par une revue détaillée des principes de base du traitement de l'image. Il se poursuit par la présentation et l'étude de deux bibliothèques open source de traitement de l'image très raffinées : *AForge.NET* et *Emgu.CV* ; ces bibliothèques sont téléchargeables gratuitement et mises en œuvre dans l'environnement Microsoft Visual Studio.

307 pages
Format 17 x 23,5 cm
ISBN 978-1-907920-09-7
39,95 €

Pour commander en ligne et bénéficier d'une remise spéciale de 5% :

www.elektor.fr/e-choppe

concours DesignSpark ChipKIT™ :

DesignSpark chipKIT™ Challenge

En septembre 2011, les ingénieurs du monde entier avaient été conviés à transformer en réalisations leurs idées géniales à partir de la carte de développement chipKIT Max32 de Microchip. Les forums DesignSpark ont offert aux concepteurs éco-responsables une plateforme d'échanges fructueux avant d'affronter le verdict du jury.

Les juges ont étudié toutes les propositions et les ont notées sur des critères de mérite technique, d'originalité, d'optimisation de la conception ainsi que sur la qualité de la carte d'extension qui devait être réalisée avec les outils PCB de *DesignSpark*. Merci à tout ceux qui ont participé à ce concours et bravo à tous les gagnants ! Le moment est venu pour eux d'obtenir la reconnaissance qu'ils méritent.

Premier prix

Dean Boman

États-Unis – d.boman@cox.net

Système de surveillance énergétique

Ce système fournit en temps-réel aux occupants d'une habitation les données de consommation électrique leur permettant de prendre les bonnes décisions. Innovant, il utilise deux cartes d'extension. Un serveur web permet de suivre l'utilisation de l'énergie par circuit. Il peut être interfacé avec un système domotique pour la surveillance à long terme et la collecte de données. Le logiciel a été écrit en C avec l'environnement de développement MPLAB de Microchip et fait appel à leur pile TCP/IP.

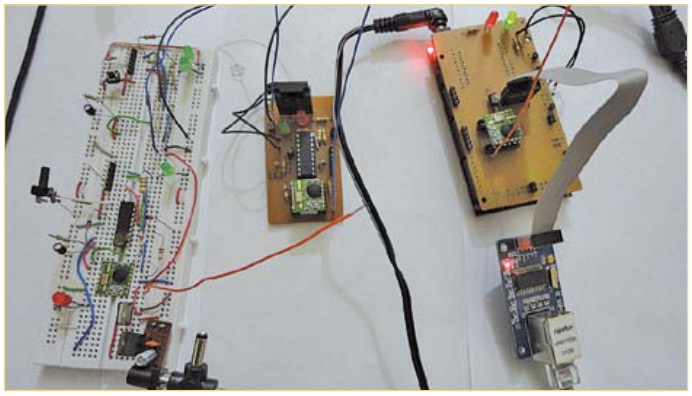
Deuxième prix

Raul Alvarez Torrico

Bolivie – raul-at@hotmail.com

Passerelle énergétique domestique

Cette passerelle énergétique domestique, bien conçue, permet à ses utilisateurs de surveiller leur consommation et commander les appareils de la maison (p.ex. les luminaires) à distance. Une passerelle/serveur web embarquée dans un Max32 communique avec deux types de capteurs intelligents dans la maison : un mesureur intelligent qui surveille la puissance active moyenne et plusieurs prises intelligentes (*Smart Plugs*) formant un réseau sans fil (*Home Area Wireless Network*). Un utilisateur peut surveiller les *Smart Plugs* et y opérer des ajustements via une interface web.



les gagnants

Troisième prix

Graig Pearen

Canada – graig@pearen.ca

SunSeeker (Suiveur de soleil pour panneaux photovoltaïques)

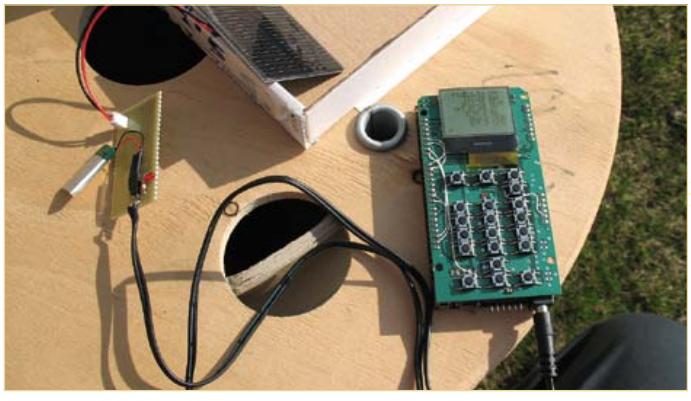
Si vous pouviez orienter vos panneaux solaires afin qu'ils reçoivent en permanence l'ensoleillement maximal, leur rendement serait optimal. C'est précisément ce que permet le *SunSeeker*, conçu pour suivre, surveiller, et ajuster des groupes de panneaux solaires en fonction de la météo et de l'état du ciel. Il décrypte la météo, mesure la température de l'air et des panneaux, calcule les statistiques nécessaires, et communique le tout à un serveur local, ce qui facilite le développement et l'affinement d'algorithmes logiciels. Un logiciel de diagnostic surveille les moteurs de l'appareil et affiche leurs mouvements et positions.

Mention honorable

Jaromir Sukuba

Slovaquie – j.sukuba@gmail.com

EDI portable pour PIC18



Cet EDI portable et autonome pour PIC18 permet de créer, éditer et assembler des fichiers source pour les PIC18 de Microchip. Le binaire produit peut-être programmé dans un PIC18 ou débogué tout en visualisant le code source. Le matériel est simple. Il comprend une interface utilisateur (LCD et clavier), de quoi stocker les données, et une interface de programmation. Cet EDI bien pratique comprend également un interpréteur permettant l'écriture et l'exécution de scripts en langage BF. Il est alimenté par le soleil et possède une batterie Li-Ion pour les jours de pluie.

Mention honorable

John Schuch

États-Unis – hackersbench@gmail.com

Serveur de temps pour réseaux sans fil maillés

Alimenté par le soleil, ce serveur de temps reçoit l'heure et la date via un module GPS et les diffuse sur un réseau sans fil maillé par l'intermédiaire d'un module XBee. Il est alimenté par un groupe de batteries NiCd, rechargées par un panneau solaire. En mode *Normal*, il diffuse l'heure et la date chaque minute. Il est également capable, à la demande d'un appareil du réseau, de passer en mode *Stream* ; les informations sont alors transmises toutes les secondes. En mode *Test*, c'est la tension des batteries qui est mesurée et transmise. Le logiciel, relativement simple, vous permet de programmer dans un langage de haut niveau tout en vous concentrant sur les économies de temps et d'énergie..

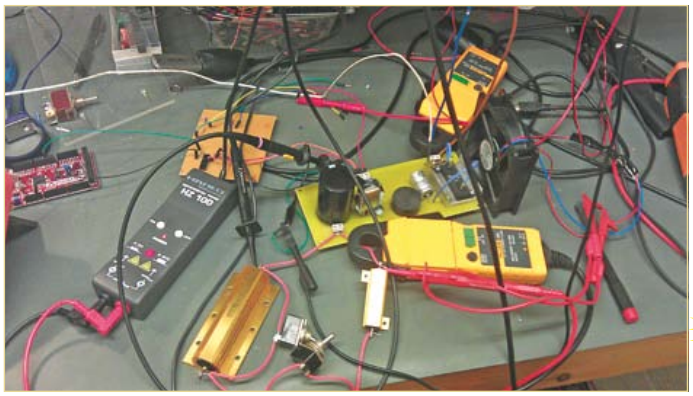
Mention honorable

Ian Johnson, Sajjad Lalji, David Weight

Royaume-Uni – ian.johnson@wattcircuit.com

Convertisseur Élévateur MPPT

Les systèmes de type *Maximum power point trackers* (MPPT) permettent de s'assurer qu'un appareil reçoit la puissance maximale



délivrable par des sources d'énergie renouvelables telles que des générateurs thermoélectriques (TEG), des panneaux solaires, et des systèmes inductifs de transfert d'énergie (IPT). La méthode d'adaptation d'impédance développée rend possible un système de commande numérique en boucle fermée. Il permet de suivre le point de puissance maximale des sources qui possèdent une résistance interne à peu près fixe, comme les TEG et IPT. La méthode MPPT a été analysée et utilisée pour construire ce prototype de convertisseur doté d'un micro gérant les opérations de contrôle. Un modèle modifié pourrait convenir aux sources possédant une résistance interne qui varie avec la température, l'ensoleillement ou les deux.

Mention honorable

Manuel Iglesias Abbatemarco

Venezuela – mhanuel@ieee.org

Centrale domotique verte

Ce système bien conçu se relie à une carte personnalisée chipSolar, de même taille que le chipKIT Max32, où se trouvent deux batteries Li-Ion qui lui fournissent son énergie. La carte possède un chargeur MPPT qui s'accommode du rendement non-linéaire des panneaux solaires. Une carte personnalisée chipWireless, dotée d'un modem GSM/GPRS quadribande, un lien Xbee, un connecteur pour carte SD et un circuit RTCC se chargent de la communication. Le logiciel a été écrit avec MPIDE. La carte SD n'est pour l'instant utilisée que pour enregistrer les données des capteurs.

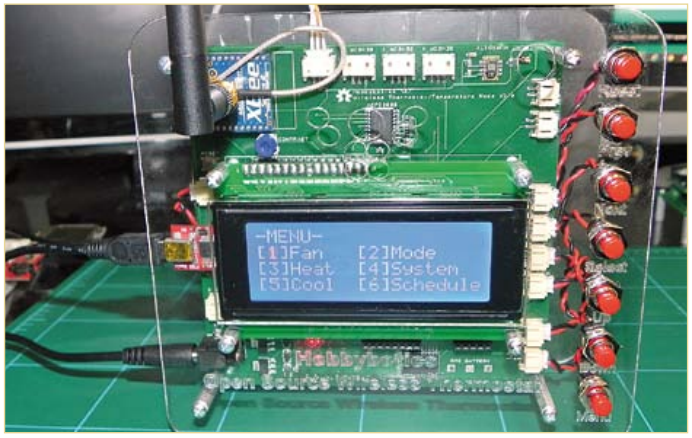
Mention honorable

Curtis Brooks

États-Unis – brooksware2000@gmail.com

Thermostat Internet multi-zones

Ce thermostat multizones offre à ses utilisateurs un contrôle maximal sur la température des bâtiments. Ce système novateur comprend principalement trois parties : un *shield* XBee, une carte « température » sans fil, une carte « sortie » commandée par I²C. Il utilise deux cartes de commande de température sans fil et deux cartes de sortie par I²C. L'accès à l'internet est fourni par un routeur. Le premier nœud comprend un Max32, un *shield* Ethernet Max32, et un *shield* Xbee (PCB). Le deuxième nœud est le thermostat sans fil qui commande un système de chauffage, ventilation et climatisation, et reçoit les données des autres nœuds, disposés dans les pièces. Ces derniers commandent un registre motorisé afin de réguler la température de chacune des pièces.



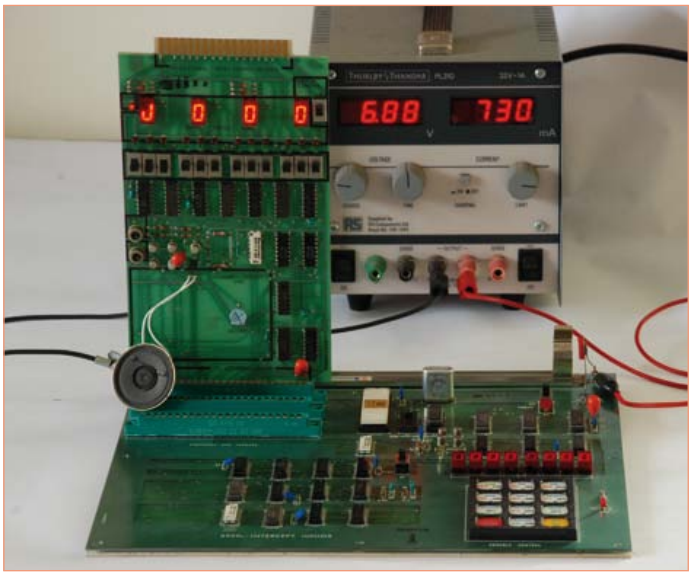
(120420 – version française : Kévin PETIT)

Vous trouverez tous les fichiers du concours DesignSpark/ChipKIT™ sur

www.circuitcellar.com/contests/chipkit2012

Intersil IM6100 : kit de

Geoff Newton (Royaume-Uni)



En 1977, ce kit de développement représentait vraiment l'état de l'art, mais il fallait être passablement pointu pour en tirer quelque chose. Il n'y avait ni assembleur ni compilateur. Vous deviez écrire le programme sur papier, le compiler à la main et charger les données nécessaires dans la mémoire au moyen du clavier — rien de considérable, cette mémoire, avec seulement 256 mots de 12 bits. Le clavier et l'affichage en sept segments se trouvent en bas à gauche de la carte. L'adresse de la case mémoire sélectionnée et son contenu sont affichés par un jeu de huit afficheurs à sept segments. Le clavier a quatre rangées de trois colonnes et la plupart des touches ont plusieurs fonctions.
Du fait du format à 12 bits, vous devez travailler en octal et non dans le système hexadécimal que nous utilisons en général aujourd'hui. Il n'y avait pas de touches montée/descente et pour passer à la case suivante vous deviez vous appuyer tout le cycle de saisie de l'adresse etc. Il y a 1 K mots de 12 bits en ROM qui contiennent un *interpretor* (sic) en microcode qui se charge des commandes du clavier etc. et trois connecteurs en bord de carte pour des cartes d'extension. Pour l'alimentation, on trouve en haut de la carte un clip qui reçoit quatre piles au format A, fourrées dans un tube de carton qui les maintient alignées.
Je n'ai qu'une carte d'extension, avec un haut-parleur et un registre à interrupteurs, une rangée de LED et quatre afficheurs à sept segments.
Il existait aussi une carte d'extension qui apportait rien de moins que 1 K (1024 mots) de RAM, une carte de ROM et une carte d'interface avec UART et port parallèle. Tous les derniers accessoires ! La platine du kit de développement est vraiment très bien réalisée, avec le même aspect que si elle avait été dessinée en CAO.

Mémoire

Ce kit a besoin de trois puces de 256 x 4 bits pour constituer sa mémoire. La version précédente avait 12 puces de 256 x 1 bit. C'était le début des puces de mémoire et cela illustre les problèmes de capacité de mémoire. Un gros système avait couramment 64 K x 16 bits et chaque programmeur avait une partition de 2 K où travailler. S'il avait besoin de plus de place, il devait s'arranger avec des calques *(overlays)* sur bande magnétique.

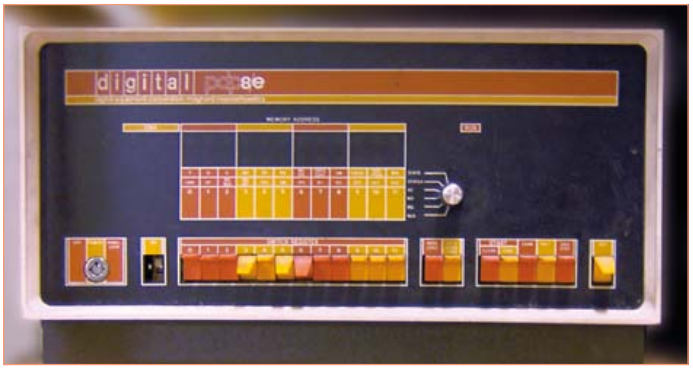
Processeur

Intersil a développé le procédé CMOS en 1972 et le processeur utilisé dans ce *Dev Kit* a suivi en 1975.
Au premier regard, le processeur paraît plutôt bon, mais si on se penche un peu les défauts deviennent visibles. Bien qu'il ait une horloge à 8 MHz, le processeur a besoin de 10 cycles au minimum pour faire quoi que ce soit ; même un NOP *(no operation)*, ce qui veut dire qu'il est désespérément lent. Une instruction de saut conditionnel consomme 22 cycles ! Le quartz de ce kit oscille seulement à 2,4576 MHz, ce qui donne pour l'exécution d'une instruction un temps minimal de plus de six microsecondes.
Il n'y a pas grand-chose au rayon des entrées-sorties ; pas plus de 64 adresses et un adressage sur 12 bits seulement, soit 4 K. Il n'y a que trois registres et pas de pointeur de pile. Pour pouvoir revenir d'un sous-programme, vous devez stocker l'adresse de retour dans la première ligne du sous-programme, la récupérer à la fin et sauter à l'adresse d'origine.

Rétronique est une rubrique mensuelle sur les pages glorieuses et jaunies de l'électronique, avec aussi des montages de légende décrits dans Elektor. Si vous avez des suggestions de sujets à traiter, merci de les adresser à redaction@elektor.fr en indiquant : Rétronique Elektor.

développement d'antan

PIN	SYMBOL	ACTIVE LEVEL	DESCRIPTION
1	V_{cc}		Supply voltage.
2	RUN	H	The signal indicates the runstate of the CPU and may be used to power down the external circuitry
3	DMAGNT	H	Direct Memory Access Grant—DX lines are three-state.
4	DMAREQ	L	Direct Memory Access Request—DMA is granted at the end of the current instruction. Upon DMA grant, the CPU suspends program execution until the DMAREQ line is released.
5	CPREQ	L	Control Panel Request—a dedicated interrupt which bypasses the normal



Il y a une interruption, mais vous devez scruter les organes d'entrée/sortie pour trouver lequel a appelé. Vraie surprise : il y a des lignes pour la requête d'accès direct à la mémoire *(DMA)* et l'accusé de réception.

La comparaison avec le Z80, à peu près contemporain, fait apparaître tout autre chose. Les gens de Zilog ont fait un saut quantique et conçu le Z80 comme un microprocesseur complet avec seulement les lignes de données, d'adresses et de commandes ramenées à l'extérieur par les broches. Par opposition, l'IM6100 ressemble plus à un morceau d'un système à puces partielles *(bit slice)* intégré dans une seule puce. C'est ce qu'il est en réalité.

Héritage du PDP8

L'IMS6100 est une version intégrée du mini-ordinateur légendaire de DEC, le PDP8 des années soixante, visible ici pour votre amusement (version Ē rare ; source *WikiMedia Commons*). Vous en trouverez plus sur Wikipedia [1] et plus encore dans le formidable *Computer History Museum* de Mountain Wiew en Californie [2]. Il a fait les beaux jours de Princeton et de la NASA.

Comme pour plaisanter, quelques broches de l'IMS6100 sont affectées à certaines fonctions telles que la lecture d'un tampon de clavier de 12 bits. Cela nous paraît étrange, mais c'était tout naturel dans le contexte du PDP8. Le PDP8 n'avait pas de système d'exploitation à proprement parler (pour démarrer) et il fallait le lancer en utilisant les interrupteurs de la face avant.

D'autres broches constituent une partie des E/S. Quand un cycle d'E/S démarre, c0, c1, c2 de même que la broche *Skip* ramènent l'information de l'organe d'E/S dans l'unité centrale et déterminent le type d'opération qui doit s'effectuer. L'entrée Skip fait sauter l'instruction suivante. Fascinant ! Une broche de sortie *link* semble être le drapeau *Carry* (retenue).

L'IM6100 utilise le jeu d'instructions du PDP8, ce qui place sa conception à mi-chemin entre la technologie à puces partielles et le processeur (*moderne*) entièrement intégré. Plutôt primitif en réalité, compliqué à programmer, mais le PDP8 était très répandu en son temps et la version mono-puce semblait une bonne idée. L'IMS6100 s'est fait une place dans quelques applications militaires. Il a même connu une version FORTRAN, mais il a été complètement évincé en 1982 quand le PC d'IBM a mis en œuvre des puces Intel 8088.

(120161 – trad. Jean-Paul Brodier)

[1] http://en.wikipedia.org/wiki/PDP-8 et http://fr.wikipedia.org/wiki/PDP-8 (plutôt sommaire)

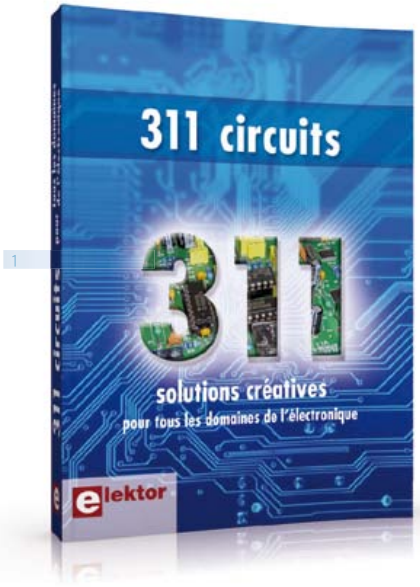
[2] www.computerhistory.org/

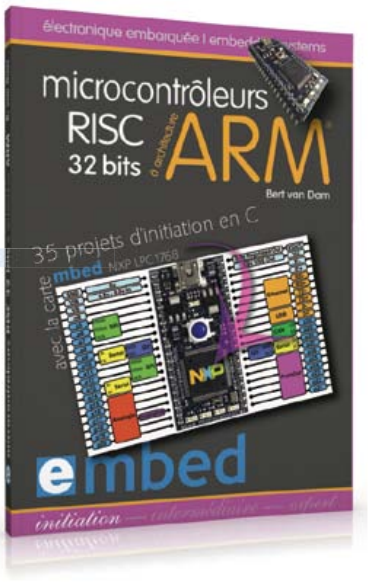
Bit Slice

Dans un système à puces partielles *(bit slice)*, l'unité centrale ou CPU occupe toute une carte à circuit imprimé (15 pouces x 18 pouces, 381 x 457 mm) remplie de circuits intégrés DIL (deux rangées de broches). Chaque élément du processeur est constitué de puces distinctes, ainsi par exemple une ALU (Unité arithmétique et logique) comportait quatre puces partielles à quatre bits 74LS181, de même l'automate fini *(State Machine)* était un assemblage de bascules D LS74 et ainsi de suite. Le circuit imprimé avait un connecteur en bord de carte qui était l'équivalent des broches d'une puce microprocesseur.

D'autres cartes fournissaient la mémoire, les E/S, l'interface du dérouleur de bande etc. Tout le bazar était installé dans une baie pour *rack* 19» avec sa propre alimentation.

Quand les écrans couleurs sont arrivés, ça a donné une autre grande boîte pleine de circuits imprimés et une autre alimentation. J'ai eu un temps un *Eclipse Data General* qui engloutissait 15 A sous 240 V !





Solutions créatives pour tous les domaines de l'électronique

1 311 circuits

Cet ouvrage est un trésor : il réunit 311 schémas d'électronique analogique, logique ou numérique, des programmes, des liens vers des sites internet, des tableaux de caractéristiques de composants et des dessins de circuit imprimé. Il est le onzième volume de la collection « 300 circuits » (301… 302… 303… 304… 305… 306… 307… 308… 309… 310… 311 circuits). Ses deux tables des matières alphabétique et thématique vous permettent de trouver rapidement et facilement parmi les 311 articles pro-posés ceux qui répondront à vos besoins. Ces articles viennent des numéros doubles récents de la revue Elektor, publiés chaque année en été, et appelés numéros Hors-Gabarit, par allusion à leur con-tenu exceptionnellement riche. Ils forment un véritable catalogue d'idées, de trouvailles et d'astuces. C'est une source d'inspiration inépuisable, et à partir de laquelle chacun élaborera ses propres variantes qu'il combinera ensuite à sa guise avec d'autres circuits. Tous les domaines familiers et usuels de l'électronique sont abordés : alimentations, régulateurs et chargeurs • audio & vidéo • communication • hautes fréquences • informatique • jeux & modélisme • maison & automobile • mesure & test • processeur & contrôleur • robots et leurs accessoires.

448 pages • ISBN 978-2-86661-184-2 • 36,00 €

Rémy Mallard présente

2 L'électronique pour les débutants

Par où commencer pour débuter en électronique ? Vais-je m'égarer en explorant l'internet, qui regorge de schémas, mais sont-ils fiables ? Me faut-il un livre avec des montages simples ou plutôt un livre sur les composants ? Après trente ans de pratique, l'auteur de ce livre, resté l'éternel débutant qui réalisait lui-même son premier montage dès l'âge de dix ans, partage ici sa soif toujours vive d'apprendre. Fin pédagogue, il guide les débutants et ré-pond aux questions que trop de livres lais-sent en suspens : « Quel type de fer à souder acheter ? »… « Un multimètre à 5 € peut-il suffire ? »… « Un oscilloscope est-il indispensable ? ».

317 pages • ISBN 978-2-86661-180-4 • 40,00 €

35 projets d'initiation en C avec la carte mbed NXP LPC 17688

3 Microcontrôleurs RISC 32 bits à architecture ARM

La plate-forme mbed et son microcontrôleur ARM, le NXP LPC1768, sont conçus pour l'informatique en nuage ou cloud computing qui révolutionne le développement de logiciel : aucune installation de logiciel spécifique, il suffit d'un navigateur et d'un port USB. Vous programmerez et stockerez vos résultats sur l'internet, et y accéderez depuis n'importe quel PC, où que vous vous trouviez. Dans ce livre, il est question aussi bien du langage C, des bibliothèques mbed, d'exemples de programmes que du traitement de signaux analogiques, de capteurs, de moteurs etc.

232 pages • ISBN 978-2-86661-178-1 • 40,00 €

sécurité – confort – économies

4 Domotique

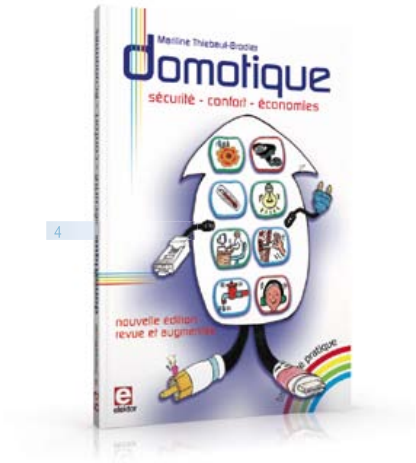
La domotique, c'est l'électronique et l'informatique appliquées au logement. Elle améliore la vie quotidienne au moyen de dispositifs électriques et électroniques. Il peut s'agir aussi bien de motoriser et télécommander par exemple la porte du garage ou bien les volets, que réguler le chauffage, programmer les appareils électroménagers, simuler une présence, commander l'alarme à distance, arroser automatiquement, exploiter un réseau multimédia, etc. Ce livre montre comment la domotique gère le fonctionnement des appareils et dispositifs électriques de la maison.

256 pages • ISBN 978-2-86661-182-8 • 33,00 €

Une carte compacte et bon marché qui vous initiera tout en douceur !

5 Embarquez Linux !

Linux est partout, même dans certaines machines à café. Souvent, l'électronicien tenté d'adopter ce



4

5

6

7

8

9

système d'exploitation est arrêté par sa complexité et par le prix des cartes de développement. Voici Linux pour les électroniciens, sous la forme d'une carte compacte et bon marché qui vous initiera tout en douceur !

Carte Linux Elektor (montée et testée)
Réf. : 120026-91 • 64,95 €

Tous les articles de 2011 sur DVD-ROM

6 DVD Elektor 2011

Ce DVD-ROM réunit tous les articles d'ELEKTOR, le mensuel d'électronique et de micro-informatique appliquées, parus au cours de l'année 2011. Il contient non seulement le texte des articles ainsi que les schémas, mais aussi tous les dessins des circuits imprimés, sous forme de fichiers à haute résolution. Ceci permet à l'utilisateur de modifier à sa guise les dessins existants à l'aide d'un programme adéquat. Dès lors, rien ne s'oppose plus à l'exportation des documents vers un autre format à la convenance de l'utilisateur.

ISBN 978-90-5381-276-1 • 27,50 €

Initiation et maîtrise par l'expérimentation

7 50 applications des microcontrôleurs PIC

Voici 50 projets instructifs et utiles pour vous initier au langage de programmation JAL et maîtriser les microcontrôleurs PIC16 et 18, avec des techniques universelles comme la commande de relais, ou le traitement des signaux émis par divers capteurs (y compris par exemple un codeur rotatif), la communication avec les bus I²C, SPI, RS232, USB, les afficheurs à 7 segments et même le bus CAN. C'est un ouvrage récréatif et pédagogique : assemblez et utilisez les projets proposés. Les explications claires, les schémas et les photographies vous feront découvrir une activité enrichissante.

394 pages • ISBN 978-2-86661-177-4 • 45,00 €

23 projets ludiques et instructifs à construire soi-même

8 Intelligence artificielle

Ce livre ne traite pas de théories abstraites, mais de pratique. Il s'adresse à vous, passionné d'électronique et de micro-informatique appliquée, et propose des circuits et des programmes simples. Expérimentez le biomimétisme sur vos propres robots mobiles, construits avec des pièces de Lego ! Il y est donc question de microcontrôleurs PIC, de programmation, de capteurs, de moteurs… mais aussi de morpions, de fourmis, de gnous, de vers, et d'autres bestioles « naturellement géniales ».

238 pages • ISBN 978-2-86661-179-8 • 43,50 €

Kit (circuits imprimés & composants)

9 TAPIR – Détecteur ultrasensible d'électrosmog

Fin limier de la pollution électromagnétique qu'il traque et rend audible sur casque, le TAPIR est (aussi) un beau projet à construire, un kit avec tout ce qu'il faut, même le boîtier, qui est fait de quatre circuits imprimés ingénieusement assemblés. Le TAPIR, sigle de Totally Archaic but Practical Interceptor of Radiation (totalement archaïque mais pratique intercepteur de radiations), détecte et localise (en la faisant entendre) toute source de champ électrique E ou, avec l'antenne appropriée, de champ magnétique H.

Réf. : 120354-71 • 14,95 €

Informations complémentaires et gamme complète sur :

www.elektor.fr/e-choppe

Elektor/Publitronic SARL
1, rue de la Haye
BP 12910 – 95731 Roissy CDG Cedex
Tél. : +33(0)1.49.19.26.19
Fax : +33(0)1.49.19.22.37
@ : ventes@elektor.fr

elektor

détecteur d'éclair

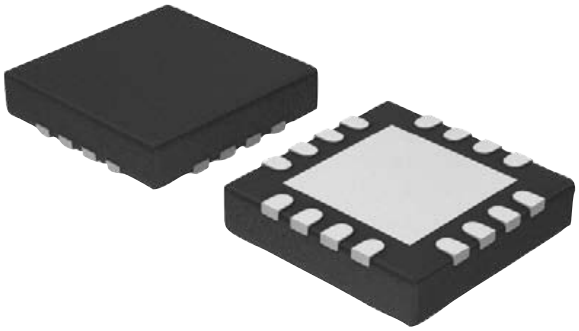
Raymond Vermeulen (Elektor)

Arrivé comme un éclair sur mon bureau, voici un circuit intégré détecteur d'éclair de seulement 4 x 4 mm, une suggestion de mon collègue Luc Lemmens. Si on lui connecte une antenne adaptée, ce CI permet à un microcontrôleur, via SPI ou I2C, de calculer la distance jusqu'à la lisière de l'orage. Vous avez bien lu : non pas la distance de la foudre, mais la distance de l'orage. La portée va de 5 à 40 km.

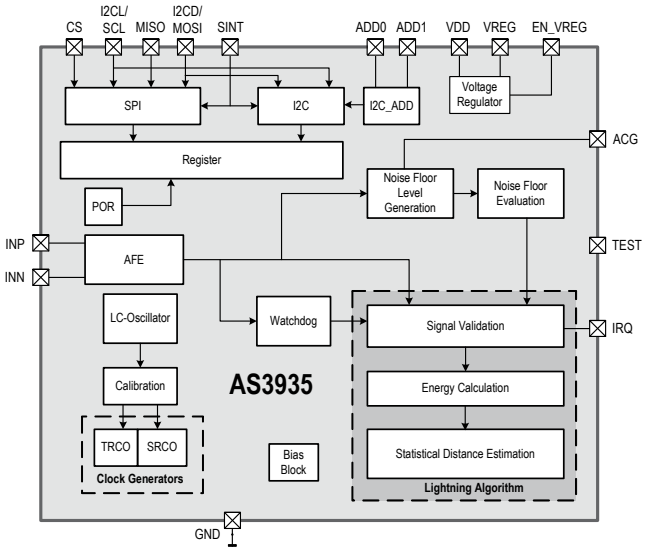
(120405 – version française : Jean-Paul Brodier)

AS3935

Le microcontrôleur associé à l'AS3935 peut de plus configurer une foule de choses dans cette puce. Je trouve très particulier l'algorithme qui permet de déterminer si les signaux détectés sont d'origine humaine ou « naturelle ». Cela signifie qu'on peut l'utiliser aussi à l'intérieur. Encore plus fort, le niveau plancher du bruit peut être consigné dans un registre, ce qui permet l'utilisation dans un environnement bruyant. Le microcontrôleur reçoit une demande d'interruption si le niveau du bruit reçu est supérieur au plancher, afin qu'il puisse éventuellement hausser le niveau du plancher. Le gain de l'amplificateur intégré a aussi un mode spécial pour l'utilisation à l'extérieur. Les sources de parasites artificiels peuvent influer négativement sur la détection des éclairs, c'est pourquoi le seuil de détection peut aussi être fixé dans un registre (cela signifie que la détection devient moins efficace).



L'antenne recommandée est une boucle constituée d'un circuit résonant LC parallèle avec un facteur de qualité Q d'environ 15, accordé sur 500 kHz. Comme il est plutôt difficile de réaliser une antenne parfaite, le circuit intégré permet d'accorder l'antenne. Un registre peut indiquer si la fréquence de résonance doit être divisée par un facteur réglable. Le microcontrôleur peut alors fixer dans un registre une capacité supplémentaire, de 0 à 120 pF par pas de 8 pF. Une fois l'antenne accordée, les autres oscillateurs internes peuvent l'être aussi. Lors de chaque interruption, la broche IRQ passe au niveau haut et le microcontrôleur peut lire dans un registre la situation spécifique qui l'a provoquée. Il est possible aussi de configurer un nombre minimal de décharges en quinze minutes si on ne s'intéresse pas à un unique éclair sporadique.



Comme nos lecteurs s'intéressent à tous les circuits qui touchent à la météorologie, je ne serais pas étonné de voir Elektor publier bientôt un circuit basé sur cette puce.

Description	Condition	Valeur
Plage de tension d'alimentation	EN_VREG = VDD	2,4 à 5,5 V
Plage de tension d'alimentation	EN_VREG = GND	2,4 à 3,6 V
Courant en mode veille	VREG = OFF	1 µA
Courant en mode écoute	VREG = OFF	8 µA
Courant en mode vérification du signal		350 µA

Feuille de caractéristiques AS3935 : www1.futureelectronics.com/doc/AUSTRIAMICROSYSTEMS/AS3935.pdf

Note d'application du kit d'évaluation :
http://media.digikey.com/pdf/Data%20Sheets/Austriamicrosystems%20PDFs/AS3935_EvalManual_AN.pdf

Hexadoku

Casse-tête pour électroniciens

Adieu le monstre à cinq têtes du numéro d'été (que vous avez sans doute terrassé) !
Votre rubrique de récréation mensuelle retrouve ici son format habituel. Ce n'est pas que ce numéro d'Elektor manque de sujets de lecture intéressants, mais il viendra bien un moment où vous aurez envie de ne penser à rien d'autre qu'aux jeux relaxants de l'arithmétique hexadécimale. Pour gagner, peut-être, l'un des quatre chèques-cadeaux Elektor mis en jeu, il suffira alors de remplir la grille et de nous envoyer votre solution.

Les instructions de ce jeu sont simples. Une grille Hexadoku est composée de chiffres du système hexadécimal, de 0 à F. Du tout cuit pour les électroniciens et les programmeurs ! Remplissez le diagramme de 16 x 16 cases de telle façon que **tous** les chiffres hexadécimaux de 0 à F (0 à 9 et A à F) n'apparaissent **qu'une seule et unique fois** dans chaque rangée, colonne et carré de 4 x 4 cases (délimités par un filet gras). Certains chiffres, déjà placés dans la grille, en définissent la situation de départ. Si vous trouvez la solution de ce casse-tête, vous pouvez gagner un chèque-cadeau. Inutile de nous envoyer toute la grille, il suffit de nous envoyer **la série de chiffres** sur fond grisé.

Participez et gagnez !

Nous tirerons au sort l'une des réponses internationales correctes qui nous seront parvenues dans les délais ; son auteur recevra un chèque-cadeau Elektor d'une valeur de 100 €. Nous offrons en outre 3 chèques-cadeaux Elektor d'une valeur de 50 € chacun.

À vos crayons !

Où envoyer ?

Envoyez votre réponse (les chiffres sur fond grisé) avec vos coordonnées par courriel, télécopie ou courrier avant le 30 **septembre 2012** à :

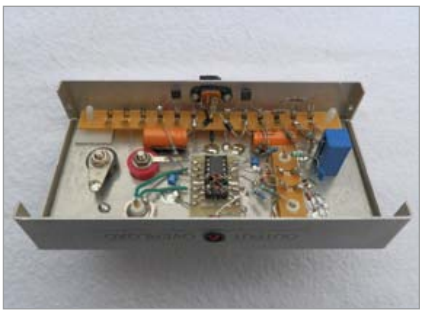
Elektor c/o Regus Roissy CDG – Le Dôme – 1, rue de la Haye
BP 12910 – 95731 Roissy CDG
Courriel : hexadoku@elektor.fr

Les gagnants

La solution de l'Hexadoku du nº 408 (juin) est : **7924A**
Le gagnant du **chèque-cadeau Elektor** d'une valeur de **100 €** est : Thomas Raith (Allemagne).
Les **3 chèques-cadeaux Elektor** d'une valeur de **50 €** chacun vont à : Reino Anttila (Finlande), Michael Evans (Royaume-Uni) et Nuno Tavares (Portugal).
Bravo à tous et félicitations aux gagnants !

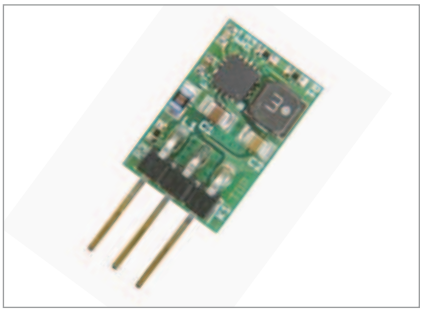
		C		9							E		6		
7			B	C	0					A	F	3			5
A	6					4			0					7	C
E	F			2		8			3		D			A	B
		3		B	2		F	4		E	1		A		
			F	3	4	0			6	B	A	E			
2	5		D		7	A			9	0		8		B	4
	A													9	
	8													0	
F	2		9		A	C			B	8		6		E	D
			E	5	8	2			A	4	0	F			
		5		F	1		6	9		D	C		2		
5	C			8		6			F		4			2	E
B	0					3			2					8	1
8			2	A	E					5	7	C			0
		4		7							8		5		

7	8	C	5	B	F	0	1	E	9	3	6	2	D	4	A
3	A	E	0	7	9	2	8	C	D	B	4	5	6	F	1
4	D	9	B	5	6	C	E	A	2	F	1	0	3	7	8
1	F	2	6	A	D	3	4	0	5	7	8	E	C	9	B
5	C	6	1	F	A	D	0	2	7	8	9	3	4	B	E
8	0	D	4	C	B	1	5	3	E	6	F	9	A	2	7
9	2	A	7	E	3	4	6	B	C	1	5	F	0	8	D
B	E	F	3	8	7	9	2	4	A	D	0	C	1	6	5
E	4	0	8	2	C	F	A	5	B	9	D	6	7	1	3
A	3	5	2	D	E	6	B	F	1	0	7	4	8	C	9
C	7	1	F	0	4	8	9	6	3	A	E	D	B	5	2
D	6	B	9	1	5	7	3	8	4	C	2	A	E	0	F
F	5	8	C	9	0	A	D	7	6	E	B	1	2	3	4
2	1	4	E	3	8	B	C	9	0	5	A	7	F	D	6
6	B	3	A	4	1	5	7	D	F	2	C	8	9	E	0
0	9	7	D	6	2	E	F	1	8	4	3	B	5	A	C



amplificateur à transconductance

L'amplificateur opérationnel AD844 d'*Analog Devices* a été conçu principalement pour les applications de réinjection de courant, mais, de par sa structure particulière, il convient aussi pour d'autres applications dans le domaine notamment de la commande sources. C'est ainsi qu'on peut l'utiliser en métrologie pour commander en tension une source de courant. Nous préparons pour vous un article sur le fonctionnement de l'AD844 et proposerons un amplificateur de mesure autour de cet ampli op.



« … il n'y a pas que le 7805 dans la vie d'un électronicien… »

Le 7805 est sans conteste l'un des circuits intégrés les plus utilisés, certainement dans le domaine des régulateurs de tension. Avez-vous jamais songé à la quantité astronomique d'énergie dissipée en chaleur par tous ces soutiers de la régulation ? Que personne n'ait encore songé à le remplacer par un régulateur à découpage compatible broche à broche ! Au labo d'Elektor nous avons conçu un petit circuit imprimé avec un convertisseur TPS62150 dont la fonction de hacheur série procure un bien meilleur rendement ; alimenté entre 5,5 et 17 V, il délivre jusqu'à 1 A sous 5 V comme il se doit pour se substituer au 7805. Il suffira de changer quelques valeurs de résistances pour obtenir d'autres valeurs de tension régulée.

détecteur de rayonnement

Le détecteur de rayonnement présenté dans le numéro de novembre 2011 d'Elektor a trouvé de nombreux adeptes de par le monde. Il est beaucoup utilisé, semble-t-il, pour les matériaux faiblement radioactifs et pour des mesures prolongées. Le microcontrôleur de ce circuit est doté d'un chargeur d'amorce (*bootloader*), de sorte qu'il est facile d'en modifier le programme. Nous reviendrons donc sur diverses applications possibles, mais aussi des modifications possibles du circuit afin d'étendre la plage de mesure.

Sous réserve de modifications. Le numéro d'octobre paraîtra le 18 septembre

Elektor en ligne

Sur le site d'Elektor, vous trouvez tous les articles publiés depuis 2000, sous forme de fichiers PDF téléchargeables individuellement, certains gratuitement, d'autres moyennant un modeste paiement forfaitaire. Un résumé de l'article donne une idée du contenu avant de le télécharger, de même que la liste des composants (le cas échéant). Le site propose également les autres ressources liées à chaque article : code source, liens, circuits imprimés, et les corrections ainsi que les mises à jour s'il y en a. L'e-choppe d'Elektor propose de nombreux produits : CD-ROM, DVD, kits, modules assemblés, appareils & instruments, E-blocks, livres. Sans oublier l'indispensable et puissante fonction de recherche.

Également sur le site d'Elektor :

- des nouvelles sur le monde de l'électronique
- un forum des lecteurs
- téléchargement de logiciel et de circuits imprimés
- des offres temporaires très avantageuses
- Foire Aux Questions

208 pages

format 17 x 23,5 cm

ISBN 978-2-86661-187-3

33,50 €