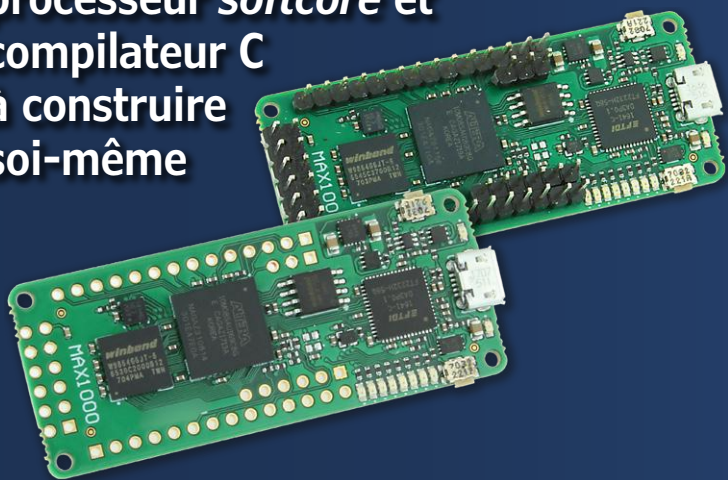


horloge rétro à afficheurs de flipper

Ah... le son apaisant des afficheurs de score des vieux flippers...

projet SCCC

processeur softcore et compilateur C à construire soi-même



surveillance du niveau de pollution de l'air

y a-t-il un fumeur caché dans les toilettes ?



commande de ventilateur (version Arduino et version analogique) ✂ kits à microcontrôleurs pour les nuls ✂ commande de moteur pas-à-pas avec circuit antirésonance ✂ pratique des amplificateurs opérationnels ✂ radar arrière pour vélo ✂ PureBasic, compilateur BASIC multi-plateforme ✂ machine-outil à commande numérique Wolverine ✂ conception matérielle avec (V)HDL (3)



- Plus de 45 ans d'expérience
- Grande disponibilité et livraison garantie
- Plus de 100 000 produits

TECHNIQUE PROFESSIONNELLE POUR L'ATELIER ET LE LABORATOIRE PLEINE PUISSANCE SUR LE LIEU DE TRAVAIL !

Weller®

La WE1010 Education Bundle de Weller

70 watts pour des exigences quasi professionnelles !

La station de soudage la plus puissante de sa catégorie ! Avec ce kit de formation, vous êtes parfaitement armé pour tous les exercices. En plus de la station de soudage WE1010, ce kit de formation comprend une pince coupante 170 mm Weller Xcelite, une bobine d'étain à braser Weller WSW SCN N1 ainsi qu'une panne à souder ETB 2,4 mm.

- Station de soudage ESD, fer à souder et câble silicone résistant à la chaleur
- Les températures stables et le verrouillage thermique protègent la panne à souder et le composant
- Une protection par mot de passe protège tous les réglages

Référence : WELLER WE1010EB

TECHNOLOGIE
TIPP 201,18 (€ 167,65)



Weller Educational Bundle :

- + Pince coupante
- + Étain à braser
- + Panne à souder

★★★★★ (2 commentaires des clients)

Alimentation de laboratoire

0 - 30 V, 0 - 10 A

Grâce à sa conception très maniable et son faible poids, ce bloc d'alimentation à découpage est parfaitement adapté aux applications dans le domaine du développement et de l'éducation.

- Présélection de tension et de courant avec sortie commutable
- Modes pour courant constant et tension constante avec charge connectée
- Arrêt automatique en cas de court-circuit
- Valeurs de sortie réglables finement : 10 mV, 10 mA

PeakTech®



**QUALITÉ POUR LA FORMATION,
LE LABORATOIRE ET LE SERVICE**

Référence : PEAKTECH 6226

97,26 (€ 81,05)

**BEST
SELLER**

Oscilloscope numérique à mémoire pour débutants !

50 MHz, 2 canaux

Oscilloscope numérique polyvalent à mémoire, également idéal pour une utilisation mobile.

- Ecran TFT couleur éclairé
- Fréquence d'échantillonnage : 1 GS/s
- différents types de déclenchement
- incl. bord, vidéo, largeur d'impulsion
- avec Ultra Sigma Software

Référence :
RIGOL DS1052E

302,55 (€ 252,12)

**HÔTE ET
PÉRIPHÉRIQUE USB**



RIGOL
Innovation or nothing

Multimètre compact avec 2 000 comptes

- Mesure de tension : 200 mV - 600 V AC/DC
- Mesure de courant : 200 µA à 200 mA
- Mesure de résistance : 200 Ohm à 20 MOhm
- Tests de continuité (buzzer) et de diodes

Référence : CHAU P01191740Z

50,55 (€ 42,13)

NOUVEAU



**CHAUVIN
ARNOUX**

Prix du jour! Prix à la date du: 31. 1. 2019

Types de paiement :



Les réglementations légales en matière de résiliation sont applicables. Tous les prix sont indiqués en € TVA légale incluse, frais d'envoi pour l'ensemble du panier en sus. Seules nos CGV sont applicables (sur le site <https://rcht.it/CG-FR> ou sur demande). Semblables aux illustrations. Sous réserve de coquilles, d'erreurs et de modifications de prix.
reichelt elektronik GmbH & Co. KG, Elektronikring 1, 26452 Sande (Allemagne), tél. +33 97 518 03 04

www.reichelt.fr

ASSISTANCE TÉLÉPHONIQUE: +33 97 518 03 04

Elektor est édité par :
PUBLITRONIC SARL
c/o Regus Roissy CDG
1, rue de la Haye
BP 12910
FR - 95731 Roissy CDG Cedex

@ : service@elektor.fr

Téléphone : (+33) 01 70 61 27 41
du lundi au vendredi de 10h à 13h

Télécopie : (+33) 01 70 61 27 42

www.elektor.fr | www.elektormagazine.fr

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

Publicité :

Margriet Debeij
Tél. : +49 (0)241 955 09 174
margriet.debeij@elektor.com

DROITS D'AUTEUR :

© 2019 Elektor International Media B.V.

Toute reproduction ou représentation intégrale ou partielle, par quelque procédé que ce soit, des pages publiées dans la présente publication, faite sans l'autorisation de l'éditeur est illicite et constitue une contrefaçon. Seules sont autorisées, d'une part, les reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective, et, d'autre part, les analyses et courtes citations justifiées par le caractère scientifique ou d'information de l'œuvre dans laquelle elles sont incorporées (Loi du 11 mars 1957 -art. 40 et 41 et Code Pénal art. 425).

Certains circuits, dispositifs, composants, etc. décrits dans cette revue peuvent bénéficier de droits propres aux brevets; la Société editrice n'accepte aucune responsabilité du fait de l'absence de mention à ce sujet. Conformément à l'art. 30 de la Loi sur les Brevets, les circuits et schémas publiés dans Elektor ne peuvent être réalisés que dans des buts privés ou scientifiques et non commerciaux. L'utilisation des schémas n'implique aucune responsabilité de la part de la Société editrice. La Société editrice n'est pas tenue de renvoyer des articles qui lui parviennent sans demande de sa part et qu'elle n'accepte pas pour publication. Si la Société editrice accepte pour publication un article qui lui est envoyé, elle est en droit de l'amender et/ou de le faire amender à ses frais; la Société editrice est de même en droit de traduire et/ou de faire traduire un article et de l'utiliser pour ses autres éditions et activités, contre la rémunération en usage chez elle.

Imprimé aux Pays-Bas
par Senefelder Misset – Doetinchem
Distribué en France par M.L.P.
et en Belgique par A.M.P.



ηλεκτρον*



LED, batteries, tubes Nixie ou VFD, transistors, condensateurs... Quel est le point commun entre tous ces composants électroniques ? Leur fonctionnement repose sur des réactions chimiques qui font apparaître des propriétés cachées au sein de la matière. Sans réaction chimique, il n'y aurait ni courant électrique ni émission de lumière...

L'année 2019 est celle de la chimie à plus d'un titre. En effet, l'Unesco a proclamé 2019 « Année internationale du tableau périodique des éléments chimiques » pour fêter le 150^e anniversaire de la publication du tableau de Dmitri Mendeleïev. Vous vous souvenez sûrement de ce grand tableau affiché dans les salles de physique-chimie de votre lycée. Cet événement a été lancé à Paris le 29 janvier dernier.

Ensuite, début juillet, Paris accueillera plus de 3 500 chimistes issus de la recherche académique et industrielle réunis pour le 47^e congrès international de chimie de l'IUPAC (Union internationale de chimie pure et appliquée, installée à Zurich).

Fin juillet, ce sont des élèves de classes de Terminale et de classes préparatoires scientifiques qui se retrouveront à Paris pour les 51^{es} Olympiades internationales de Chimie. Cette compétition fermera l'année scolaire 2018-2019 désignée par le ministère de l'Éducation nationale comme « année de la chimie de l'école à l'université ». Ces événements sont l'occasion de nous remémorer nos cours de chimie. Pour redécouvrir le tableau périodique, rendez-vous sur www.lelementarium.fr. Pour rafraîchir vos connaissances avec le sourire, essayez donc « La chimie en BD » de Larry Gonick et Craig Criddle.

Mariline Thiebaut-Brodier

* Mot grec (êléktôr) qui signifie « ambre », traduit en français par « électron » parce que les Grecs avaient découvert que si l'on frotte de l'ambre, elle est électrisée et attire d'autres objets.

Notre équipe

Rédactrice en chef :

Mariline Thiebaut-Brodier (redaction@elektor.fr)

Rédaction internationale :

Thijs Beckers, Jan Buiting, Jens Nickel

Laboratoire :

Mathias Claussen, Ton Giesberts, Luc Lemmens, Clemens Valens (responsable), Jan Visser

Coordination :

Hedwig Hennekens

Ont coopéré à ce numéro :

Pascal Duchesnes, Pascal Godart, Yves Georges, Robert Grignard, Denis Lafourcade, Jean-Louis Mehren, Denis Meyer, Hervé Moreau, Helmut Müller, Kévin Petit, Xavier Pfaff, Guy Raedersdorf

Service de la clientèle :

Cindy Tijssen

Graphistes :

Giel Dols

Elektor en ligne :

Daniëlle Mertens

- 5 bientôt dans Elektor
- 24 agenda
mars 2019
- 60 vol tous azimuts
l'électronique par monts, maux et merveilles
- 77 agenda
avril 2019
- 104 Rétronique
s'il vous plaît... dessine-moi un bouton !
les dessinateurs d'Elektor,
grands princes du crayon
- 110 questions d'éthique
comment introduire concrètement des
valeurs éthiques dans votre projet ?
- 112 l'e-choppe d'Elektor
- 114 hexadoku
casse-tête pour elektorniciens

en coulisse

- 6 kits à microcontrôleurs pour les nuls
microcontrôleurs ? pas de panique !
- 25 bruits de labo...
- 38 édition 2019 de la NXP Cup
plus intense et plus passionnante que
jamais !
- 48 trucs et astuces
comment remédier élégamment aux dégâts
causés par les fuites des piles alcalines
- 56 la pratique
des amplificateurs opérationnels
partie 2 : amplificateurs opérationnels
à transistors à effet de champ dans les
applications à large bande
- 64 PureBasic
un compilateur BASIC,
beaucoup de plateformes
- 69 projet 2.0
corrections, mises à jour
et courrier des lecteurs
- 72 mesurer, alimenter et imprimer.....
...avec des appareils de l'e-choppe d'Elektor

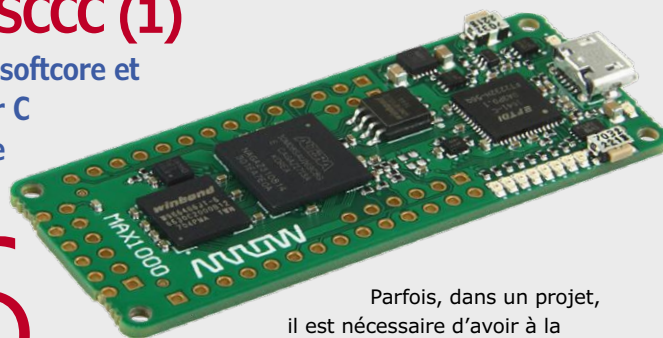
horloge rétro



Ah... le son apaisant des afficheurs de score

projet SCCC (1)

processeur softcore et
compilateur C
à construire
soi-même



26

Parfois, dans un projet, il est nécessaire d'avoir à la fois un circuit FPGA et un processeur (CPU) classique, plus facile à programmer pour le déroulement de programmes complexes. La réalisation d'un processeur en FPGA peut constituer une solution. Il est possible de se procurer un tel processeur logique ou « softcore », mais ce n'est pas très bon marché. Dans ce projet, nous vous proposons de réaliser vous-même un CPU simple à partir d'un FPGA ; grâce au code source ouvert, vous pourrez l'adapter à vos besoins. C'est l'objet du compilateur C DIY (*Do It Yourself* = faites-le vous-même) qui peut, lui aussi, être étendu. Dans cette série d'articles, la théorie sera largement étayée par des exemples pratiques.

- 74 conception matérielle avec (V)HDL (3)
commande d'une matrice de LED WS2812
- 90 sources d'interférences dans
l'électronique automobile
et comment les éliminer !
- 94 lampes à LED et parasites
lampes conformes aux normes de CEM ?
- 99 hors-circuits de R. Lacoste
parlez-vous I/Q ?
phase et quadrature de phase : comprendre
les modulateurs I/Q



104

rétronique

s'il vous plaît...
dessine-moi
un bouton !

à afficheurs de flipper

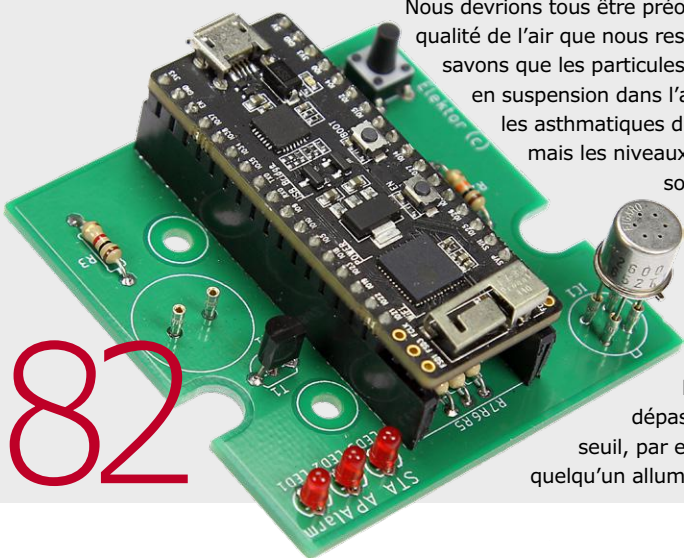
14

des vieux flippers...

surveillance du niveau de pollution de l'air

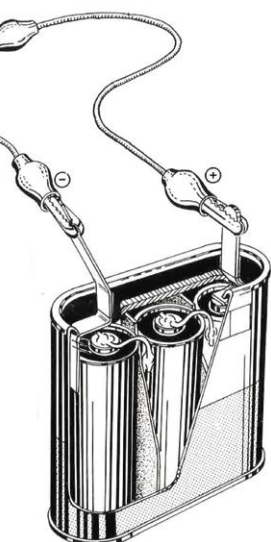
y a-t-il un fumeur caché dans les toilettes ?

Nous devrions tous être préoccupés par la qualité de l'air que nous respirons. Nous savons que les particules de poussière en suspension dans l'air empêchent les asthmatiques de dormir la nuit, mais les niveaux de certains gaz sont tout aussi importants. Ce projet mesure le niveau des gaz qui contaminent l'air et déclenche une alarme quand leur concentration dépasse un certain seuil, par exemple lorsque quelqu'un allume une cigarette...



sur la scène : les projets

- 14 horloge rétro à afficheurs de flipper**
Ah... le son apaisant des afficheurs de score des vieux flippers...
- 26 projet SCCC (1)**
processeur *softcore* et compilateur C à construire soi-même
- 34 commande de ventilateur**
en versions numérique (carte Arduino) et analogique



- 39 commande de moteur pas-à-pas avec circuit antirésonance**
moteurs pas-à-pas : plus de rendement et moins de vibrations
- 44 radar arrière pour vélo**
un montage simple pour améliorer sa cyclo-sécurité
- 50 machine-outil à commande numérique Wolverine**
portique robotisé multifonctionnel – 1^{re} partie : construction mécanique
- 79 e-BoB pour capteur de point de rosée ENS210**
avec mise en œuvre sur Arduino Uno
- 82 surveillance du niveau de pollution de l'air**
y a-t-il un fumeur caché dans les toilettes ?
- 86 sonde pour bus I²C**
données envoyées par USB vers un écran de PC
- 96 règle-gadget Raspberry Pi**
s'amuser avec un capteur de temps de vol

 **bientôt sur ces pages**

Extrait du sommaire du prochain numéro :

- Carte CPLD MAX10
 - Carte de commande à relais, à neuf voies, avec interface PC
 - HAT pour la cryptomonnaie IOTA
 - Mise à jour de la station météo
 - Moniteur de batteries à ESP32
 - Réveil à trois affichages plus précis
 - Réveil avec capteur de température extérieure, animé par un ESP32
 - Thyristor MOSFET à deux anodes
- Etc.

Sous réserve de modification.
Le numéro de mai-juin 2019 paraîtra le 18 avril 2019.

kits à microcontrôleurs pour les nuls

microcontrôleurs ? pas de panique !

Beaucoup d'électroniciens ont peur de se lancer dans la programmation des microcontrôleurs. Mais démarrer n'est pas si difficile, comme nous allons le voir avec quatre cartes bon marché. Elles sont supportées par un grand nombre de programmes de démo, ce qui garantit un succès rapide. Dans ce premier article, Burkhard Kainka, le prolifique collaborateur d'Elektor, a testé une carte Arduino d'un coût particulièrement abordable. Son fils Fabian Kainka a fait ses premiers pas avec la carte NodeMCU, qui se prête avantageusement à la programmation liée à l'internet.

Beaucoup (trop) d'électroniciens craignent de se lancer dans la programmation des microcontrôleurs. L'une des raisons en est sans doute la difficulté de choisir, dans la jungle opaque des kits à microcontrôleurs, le bon modèle, pas trop compliqué, que le fabricant et les revendeurs supportent généreusement en matériel et logiciel, et qui bénéficie d'une communauté d'utilisateurs à laquelle le débutant dans la détresse la plus noire peut demander de l'aide (c'est-à-dire, sans la crainte de poser des questions stupides).

Nous avons sélectionné quatre cartes/kits à contrôleurs appropriés aux programmeurs novices et nous les avons confiés à quatre électroniciens chargés de les déballer et de les essayer. Vous allez voir et lire que l'accès au monde des microcontrôleurs n'est pas si compliqué !

Dans ce numéro, les premiers à s'y mettre sont Burkhard Kainka, le père, et Fabian, son fils. Le premier est bien connu comme auteur d'innombrables circuits pour bricoleurs, publiés dans Elektor, de livres et de didacticiels, et animateur d'un site web pour électroniciens amateurs. Il a pris en mains un clone de la carte Arduino Nano.

Son fils Fabian a mis les pieds dans les pas de son père et s'est spécialisé dans les développements et les publications dans le domaine de l'Internet des Objets. C'est pourquoi « son » kit à microcontrôleur est aussi une carte de développement particulièrement adaptée à ce domaine, le kit NodeMCU.

Dans le prochain numéro d'Elektor, il sera question, entre autres, d'une carte ESP32-WLAN/Bluetooth équipée d'un afficheur OLED intégré.

JOY-iT Nano V3

Burkhard Kainka (Allemagne)

Le JOY-iT Nano V3 est une carte compatible Arduino Nano de la société JOY-iT, disponible dans la boutique Elektor au prix de 12 €. Vous pouvez lui ajouter le *kit complémentaire pour Arduino* qui contient des composants additionnels qui facilitent et simplifient les débuts avec cette carte.

On devrait toujours avoir un Arduino en réserve, vu les services qu'il rend dans un labo d'électronique et pour l'essai de nouvelles idées. Quoique je me serve le plus souvent du modèle standard Arduino Uno, le Nano offre pratiquement le même matériel autour d'un ATmega328 sur une surface sensiblement moindre. Cela signifie de l'espace mémoire à profusion, même pour des grands projets, et plus de puissance de calcul qu'il n'en faut habituellement.

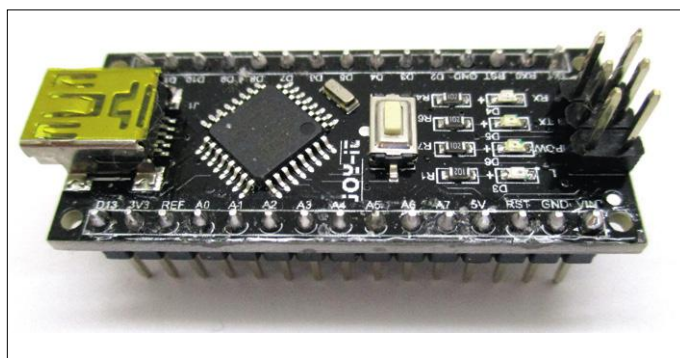


Figure 1. Carte JOY-iT Nano V3.

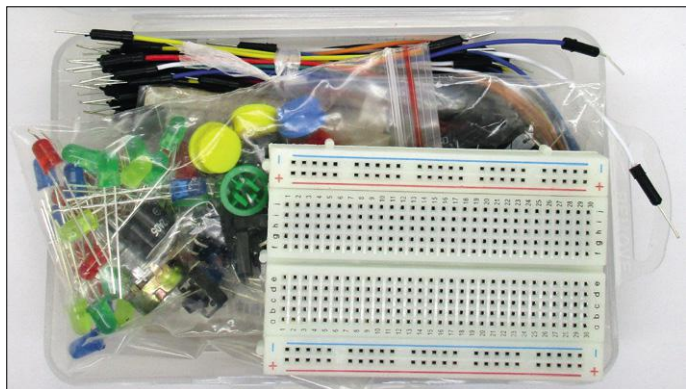


Figure 2. Composants du kit complémentaire pour Arduino.

Premiers essais

Pour voir la différence entre l'Arduino Nano original et le clone Nano de Joy-It (**fig. 1**), il faut retourner la carte. Au verso de la carte se trouve la puce USB, une FT232R sur l'original et une CH340G sur la copie. Les deux ont la même fonction : elles transforment le port série de l'ATmega en un port USB. Lorsqu'on connecte la carte à un PC, un port virtuel est créé. Il est baptisé COM2, COM3 ou COM99, selon le nombre de modules avec un port sériel qui ont été connectés au PC auparavant. À la différence de la puce FT232R, avec laquelle un nouveau numéro COM est créé à chaque connexion, le même numéro est réutilisé pour la CH340 ; on le peut modifier si l'on veut connecter plusieurs cartes à la fois. Si l'on a des difficultés sur un PC avec le pilote USB, il faut télécharger ce pilote sur le web [1] et l'installer manuellement.

Le kit complémentaire pour Arduino (**fig. 2**) contient une plaque d'expérimentation et un grand nombre de composants utiles, du câble au potentiomètre. On dispose de davantage de LED, de résistances, de condensateurs et de boutons-poussoirs qu'on peut en consommer en une semaine.

Passons aux choses sérieuses : les tests. J'ai installé le module sur la plaque d'expérimentation (**fig. 3**) pour que les essais puissent être effectués de manière stable, en toute sécurité et sans risque de courts-circuits. Pour la connexion au PC, il faut un câble mini-USB. La LED verte sur la carte s'allume, ce qui montre qu'elle est bien alimentée. La LED jaune clignote avec de brefs éclats, ce qui indique sur un Arduino qu'aucun programme n'a encore été chargé et que le chargeur est prêt à travailler.

C'est le moment de démarrer l'EDI Arduino. À chaque fois que je teste un nouvel Arduino, je commence par charger le programme de démo **Blink.ino** (**fig. 4**) inclus dans l'EDI. Dans le monde Arduino, un programme s'appelle un croquis. **Blink.ino** fait clignoter la LED jaune à la période de 2 s. Avant de pouvoir charger ce croquis, je dois commencer par choisir la carte (Arduino Nano) dans l'EDI et activer le port effectif (COM2). J'avais prévu qu'il s'agirait de COM2, car j'avais précédemment utilisé un autre contrôleur également équipé d'une CH340, pour lequel j'avais sélectionné COM2.

Maintenant, téléverser et attendre. L'opération se termine avec succès, la LED clignote lentement. Pour en être bien sûr, je règle le délai sur 2000 ms et recharge le croquis à nouveau. Le clignotement est plus lent.

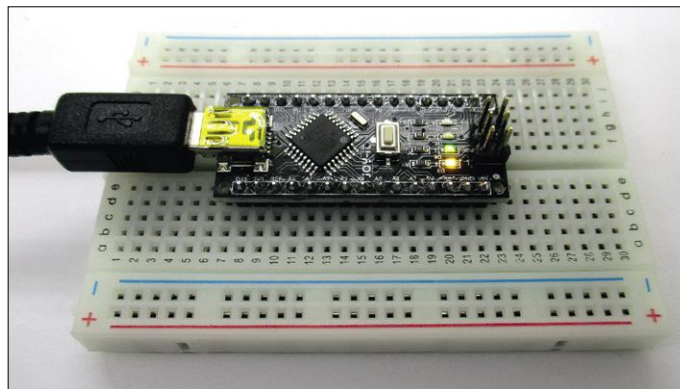


Figure 3. L'Arduino Nano en opération.

Tout fonctionne donc correctement. Alors, que faire de plus ? Mon regard parcourt le matériel du kit. Il y a à un vibreur qui a envie d'être testé. Il s'agit d'un vibreur actif, avec électronique intégrée. Un test avec la sortie à 3,3 V du Nano fonctionne ; sur le 5 V, le bourdonnement devient plus intense. Je décolle l'opercule comme spécifié et le recolle aussitôt, l'intensité du bourdonnement étant devenue franchement désagréable. Et si je l'atténuais avec une résistance ? Sitôt testé, sitôt abandonné : au-delà de 100 Ω , le courant devient trop faible pour démarrer le vibreur. Mais sur une sortie, ça devrait marcher ! Je relie le vibreur à la broche 13, celle qui est connectée à la LED jaune. Comme prévu, on entend un bourdonnement en pointillé. Le vibreur et la LED sont synchrones.

Mais il me vient un soupçon : et s'il ne s'agissait pas d'un vibreur piézo-électrique comme je l'avais supposé, mais d'un vibreur électromagnétique ? J'approche un aimant, l'attraction se fait clairement sentir. OK, ces composants contiennent une bobine avec un oscillateur à transistor et une membrane magnétique. Ils exigent un courant bien plus élevé qu'un vibreur piézo com-

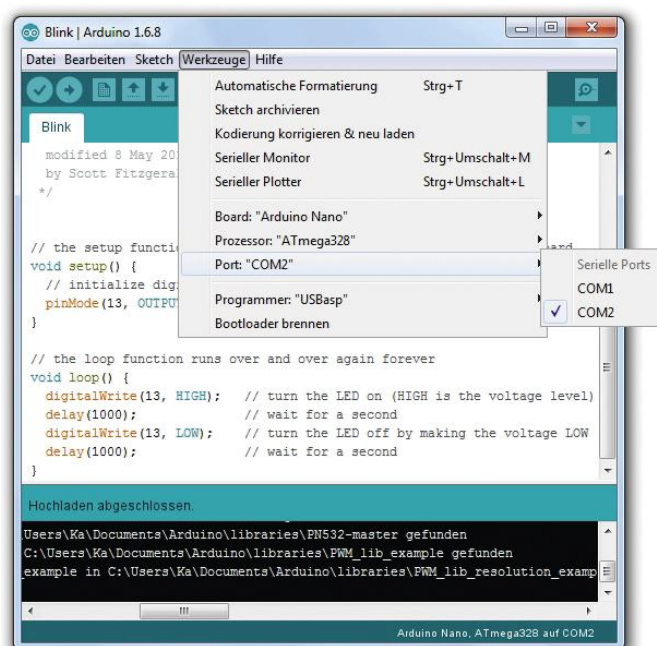


Figure 4. Configuration dans l'EDI et téléversement vers le Nano.

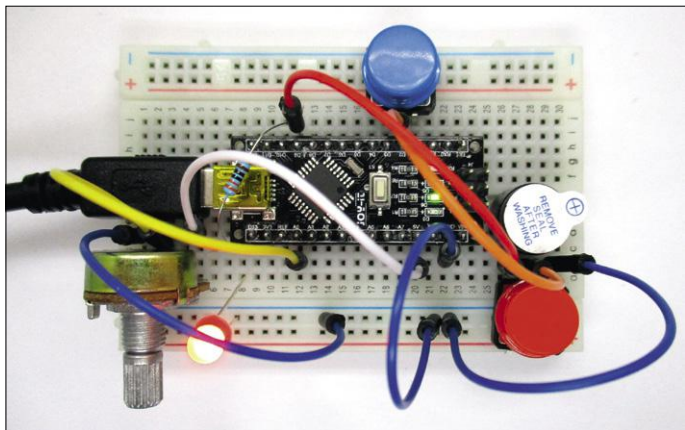


Figure 5. Un montage de test.

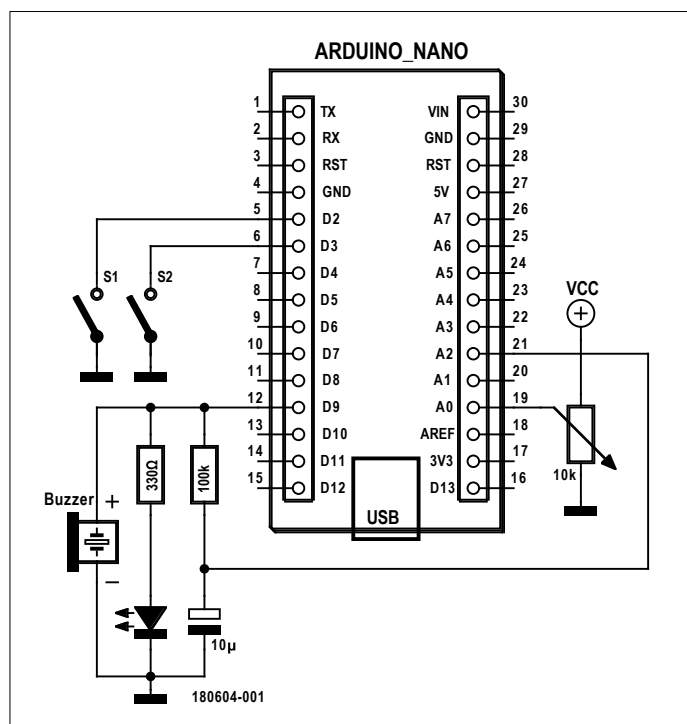


Figure 6. Extension avec un filtre passe-bas de 100 kΩ / 10 μF.

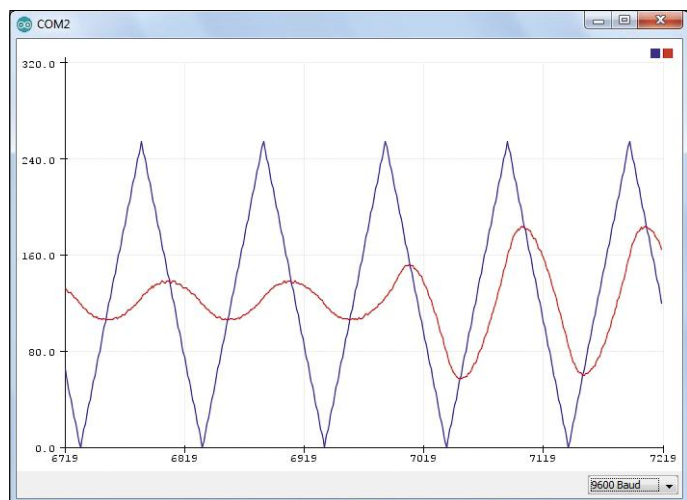


Figure 7. Signal de sortie et tension filtrée.

parable, c'est pourquoi ça n'a pas marché avec la résistance en série. Mais une sortie de l'ATmega fournit un courant suffisant. Ce courant suffit même pour alimenter sans pilote intermédiaire des moteurs à courant continu à très faible consommation.

Vibreux sur sortie MLI

Quoi d'autre peut-on encore essayer avec le matériel disponible ? Je teste une LED rouge avec une résistance série de 330 Ω sur D9 avec le croquis de démo [Fade.ino](#) de l'EDI Arduino. Ça marche immédiatement, la LED fonctionne en « clignotant doux » : son intensité croît et décroît lentement. Tout ça grâce à la MLI, c'est-à-dire la commutation rapide d'une sortie.

Mon instinct ludique s'éveille petit à petit. Et si je connectais le vibreur directement à une sortie MLI ? En effet, ça marche, même si le vibreur n'est en principe pas prévu pour cela. La puissance du son varie en cadence avec la LED, accompagnée d'une faible variation de la fréquence du vibreur. À certains moments, on entend les battements entre la fréquence du vibreur et celle de la MLI.

Et que reste-t-il comme place sur la plaque d'expérimentation ? À première vue, on se sentirait rapidement à l'étroit, mais on finit toujours par trouver un petit coin d'espace. Deux boutons-poussoirs de bonne taille, un potentiomètre, le vibreur, la LED, la résistance et pas mal de fils, tout cela y trouve place (**fig. 5**). Des boutons-poussoirs, seules deux pattes sont embrochées, les deux autres pendent dans le vide.

Ajoutons des fonctions à la démo [Fade.ino](#). Le bouton-poussoir rouge va bloquer la sortie et couper le son. Une pression sur le bouton bleu et ça repart. Le potentiomètre doit servir à régler la vitesse. Et comme toujours, ça ne marche pas au début, à cause d'une faute de frappe ; et comme toujours, il faut que je me relise (parce que j'ai plutôt l'habitude de travailler avec Bascom). Mais tout finit par marcher comme prévu. Certes, ce projet ne vole pas bien haut et finit par devenir assez éternel (c'est pour ça qu'il y a le bouton d'arrêt S1), mais il est plutôt pas mal pour faire ses gammes. Et à la fin, on se sent bien quand tout marche comme on l'a désiré.



@ WWW.ELEKTOR.FR

→ JOY-iT Nano V3, réf. 18615
www.elektor.fr/joy-it-nano-v3

→ Carte Arduino Nano, réf. 17002
www.elektor.fr/arduino-nano-3

→ Kit complémentaire pour Arduino, réf. 17241
www.elektor.fr/arduino-supplement-kit

Livres

→ « 36 Expériences de Physique avec Arduino »
www.elektor.fr/36-experiences-de-physique

→ « Tuto Arduino Uno »
www.elektor.fr/tuto-arduino-uno

→ « Maîtrisez les microcontrôleurs à l'aide d'Arduino (3^e édition) »
www.elektor.fr/maitrisez-les-microcontroleurs

En travaillant avec cette carte, j'ai remarqué un comportement étrange qu'au début j'étais incapable de m'expliquer. Après avoir chargé un programme, je pouvais déconnecter le câble USB du PC et le connecter à un bloc d'alimentation. Cela rend l'application portable. Mais quand je rebranchais le câble au PC, le programme disparaissait ! La LED jaune se remettait à clignoter comme pour un Nano vide. Que s'était-il passé ? Une recherche sur le web m'a donné une réponse : cela vient de l'EDI utilisé, la version 1.6.8. Quand il détecte la connexion d'un nouvel Arduino, il le commute immédiatement dans le mode téléchargement. Pourquoi ne m'en suis-je pas aperçu plus tôt ? Parce que je préfère utiliser la version 1.6.5 moins récente, mais plus rapidement prête à l'emploi.

Tracer des signaux analogiques

Il arrive qu'il faille utiliser la version 1.6.8 ou plus récente, surtout si l'on a besoin du traceur série. Il permet d'afficher des grandeurs variables. Dans la version 1.8.2, le traceur peut même afficher plusieurs canaux. Le programme a en conséquence été étendu pour pouvoir tracer le contenu de la variable `brightness`. De plus, un filtre passe-bas de 100 kΩ et 10 μF a été ajouté pour lisser le signal MLI et le mesurer sur l'entrée analogique A2 (**fig. 6** et **listage 1**). La **figure 7** en montre le résultat. La luminosité est pilotée par un signal triangulaire, mais le filtre RC en fait une sinusoïde approchée. Après quelque temps, une action sur le potentiomètre a ralenti le signal de sortie. La période du signal varie par rapport à la constante de temps fixe du filtre RC. L'amplitude du « signal sinusoïdal » augmente ainsi que sa distorsion. Ce genre d'expérience exige en principe tout un parc d'appareils de mesure, mais ici, accessoirement, le Nano s'occupe de tout quasiment.

Programmation ISP

D'ailleurs, il n'est pas nécessaire que ce soit toujours l'EDI Arduino. En principe, on peut utiliser n'importe quel environnement de développement pour microcontrôleur AVR.

On peut charger le fichier *hex* à travers le connecteur ISP, ce qui rend la petite carte encore plus universelle. Comme déjà dit, je travaille volontiers avec Bascom. Dans la série d'articles sur l'Arduino Uno (« L'atelier du microcontrôleur », à partir d'avril 2014, n°430), on utilise un chargeur d'amorçage Bascom spécialement adapté, qui fonctionne aussi pour le Nano, vu qu'on emploie le même contrôleur ATmega328 à la même fréquence d'horloge de 16 MHz.

J'ai « gravé » le chargeur dans le Nano au moyen d'un STK500 (AVR Flash Micro-

controller Starter Kit). C'est là que j'ai rencontré un problème particulier. Le Nano (de même que son clone) utilise sur la broche de réinitialisation (*reset*) une résistance de rappel particulièrement petite de 1 kΩ, de sorte que le STK500 n'arrivait pas à la tirer suffisamment vers le bas. L'Arduino Uno utilise

Listage 1. Exemple Fade étendu.

```
/*
  Fade
  Cet exemple montre comment faire varier
  l'intensité d'une LED sur la broche 9 en
  utilisant la fonction analogWrite().
  Ce code de démo est dans le domaine public.
  Autres : pot + s1 off + s2 on + AD + traceur série
*/

int led = 9;           // broche connectée à la LED
int brightness = 0;    // luminosité de la LED
int fadeAmount = 5;    // incrément de la luminosité
int s1 = 2;
int s2 = 3;

// fonction setup exécutée quand on appuie sur reset
void setup() {
  // configurer la broche 9 en sortie:
  pinMode(led, OUTPUT);
  pinMode(s1, INPUT_PULLUP);
  pinMode(s2, INPUT_PULLUP);
  Serial.begin(9600);
}

// fonction loop exécutée indéfiniment
void loop() {
  // envoyer la luminosité sur la broche 9:
  analogWrite(led, brightness);
  Serial.print(brightness);
  int U = analogRead(A2);
  Serial.print( " ");
  Serial.print(U/4);
  Serial.println( " ");
  // incrémenter la luminosité:
  brightness = brightness + fadeAmount;

  // en fin de variation de luminosité, changer de sens:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // attendre ... millisecondes pour voir l'effet
  delay(analogRead(A0)/10+10);
  if (digitalRead(s1) == LOW){
    analogWrite(led,0);
    while (digitalRead(s2) == HIGH);
  }
}
```

Liens

- [1] Pilote CH340G : www.wch.cn/download/CH341SER_ZIP.html
- [2] Site web de l'auteur : www.b-kainka.de/

une résistance de rappel de 10 k Ω , pour laquelle il n'y a pas ce problème. Pour le Nano, j'ai trouvé une astuce de dépannage rapide et malpropre : ajouter entre *reset* et la masse une autre résistance de 1 k Ω . On est alors juste à la limite du *reset*, mais ce coup de pouce permet au STK500 d'y arriver.

Ce genre d'ennui peut aussi se produire pour d'autres programmeurs. Donc, si quelque chose ne va pas, examinez la tension sur la broche *reset*.

NodeMCU

Fabian Kainka (Allemagne)

NodeMCU est une carte de développement pour l'Internet des Objets. Elle est basée sur le microcontrôleur à 32 bits ESP8266 du fabricant chinois Espressif.

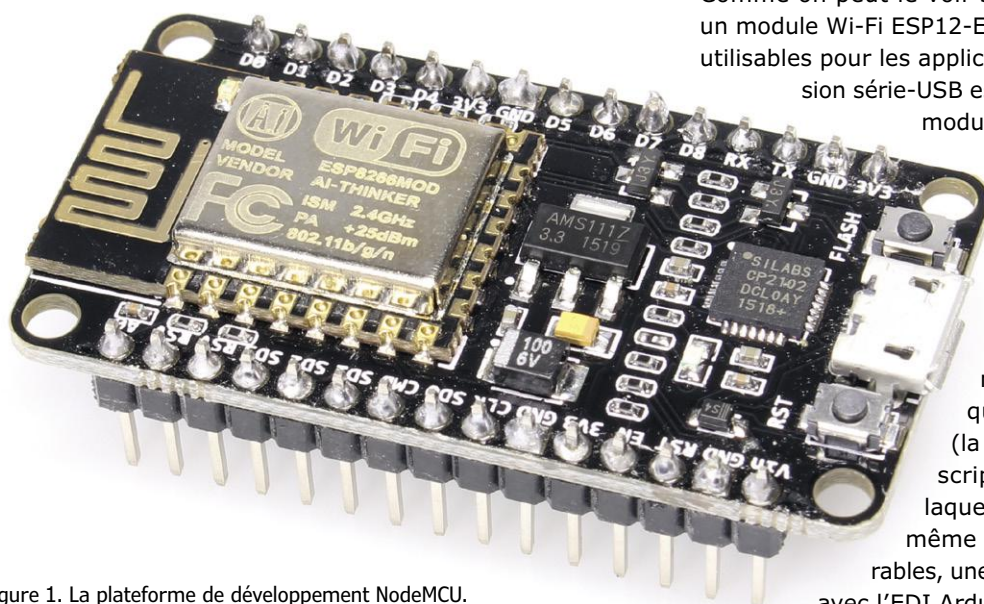


Figure 1. La plateforme de développement NodeMCU.

Caractéristiques techniques de l'ESP8266

- Module ESP12-E
- Microcontrôleur à 32 bits, basse consommation, cadencé à 80 MHz
- 4 Mo de mémoire flash pour les programmes et les données
- Wi-Fi : 802.11 b/g/n jusqu'à 72,2 Mbit/s
- Antenne imprimée
- Tension d'alimentation de 3,3 V
- Programmation et alimentation par connecteur micro-USB
- Convertisseur USB-série : DP2102
- 13 broches GPIO
- CA/N à 10 bits (par approximation successive)
- SPI, I²C, I²S, 2x UART, IRDA, PWM, RTC
- 1 LED et 2 boutons-poussoirs embarqués

Comme on peut le voir sur la **figure 1**, le matériel utilise un module Wi-Fi ESP12-E. On dispose en tout de 13 broches utilisables pour les applications les plus diverses. La conversion série-USB est réalisée par une puce DP2102. Le module embarque aussi une LED et deux boutons-poussoirs (dont un pour la réinitialisation). On dispose donc de tout ce qu'on peut espérer d'une carte de développement ESP8266.

Mais NodeMCU ne désigne pas seulement la carte de développement, mais aussi le microgiciel préinstallé, qui inclut un interpréteur Lua. Lua (la Lune, en portugais), un langage de script, est en fait la vraie raison pour laquelle je m'intéresse à cette carte. De même que bon nombre de cartes comparables, une carte à ESP8266 est programmable avec l'EDI Arduino sans problème et sans équipement complémentaire. Mais ce qui est fascinant avec Lua, c'est que les programmes sont interprétés en cours d'exécution et n'ont pas besoin d'être compilés au préalable. Les programmes ne sont alors rien d'autre que de simples textes compris par l'interpréteur, traduits et exécutés en temps réel. Le tout est comparable à la console de Linux ou au PowerShell de Windows, qui peuvent accepter des commandes à tout moment et exécuter des fichiers script automatiquement. Là aussi, le seul environnement de développement nécessaire n'est rien de plus qu'un simple éditeur de texte. On peut même se simplifier encore un peu la tâche, comme nous le verrons bientôt.

Les premiers pas

Mon premier pas a été de débiller la carte et de la connecter à un port USB de mon PC. Pour la première liaison série, j'ai utilisé le programme d'émulation de terminal PuTTY, mais n'importe quel autre programme de ce type fait l'affaire. Après l'établissement de la connexion à une vitesse de 9600 baud, j'ai pu recevoir les premiers caractères lisibles, qui me révélèrent

que j'avais affaire à la version 0.9.6 du micrologiciel, compilée mi-2015. C'était un peu trop ancien pour moi, et je me suis mis en quête de la version la plus récente, ce qui m'a permis de découvrir, au passage, plusieurs choses passionnantes.

Un nouveau micrologiciel

Mon chemin m'a conduit à la page web concernée du site Github [1] (Github est une plateforme internet où des développeurs du monde entier partagent leurs codes source). Toutes les informations importantes sont disponibles sur ce site, et j'espérais bien pouvoir y télécharger le micrologiciel le plus récent. Mais à ma grande surprise, j'ai découvert qu'il n'existait pas une version plus récente unique, mais que chaque utilisateur pouvait se constituer son propre micrologiciel. Il y a actuellement plus de 65 modules que l'on peut (doit) choisir selon son propre goût pour les intégrer dans son micrologiciel Lua. Mais tous les modules ne peuvent pas prendre place à la fois dans la mémoire flash de 4 Mo, ce serait comme si l'on voulait utiliser toutes les bibliothèques Arduino en même temps. L'utilisateur doit donc décider au préalable de la direction dans laquelle il entend orienter son développement.

Les modules standard Timer, UART, Wi-Fi, GPIO, etc. feront sans doute partie de toutes les configurations, alors que les modules tels que HTTP, MQTT, SNTP ou de différents capteurs comme DHT (météo), BME280 (météo, pression atmosphérique), HMC5883L (magnétomètre à 3 axes) ou TCS34725 (luxmètre, colorimètre) ne seront inclus qu'en cas de besoin. Cela donne l'impression que le nombre de possibilités est grand, mais aussi qu'il faut fournir un effort considérable pour réunir tous ces modules et ensuite les compiler pour obtenir un micrologiciel opérationnel. Ce serait certainement pénible s'il n'existait pas un service de compilation en ligne (**Cloud Build Service**) facile à utiliser (**fig. 2**). Sur la page web [2], il suffit de constituer son propre micrologiciel par simples clics sur les modules nécessaires. Il y a des options additionnelles, comme quelle filière (*branch*) de Github on veut utiliser ou si l'on a besoin du support TLS/SSL ou de FatFS pour lire des cartes SD. Une fois le choix effectué, on saisit son adresse de courriel et on clique sur « Démarrer votre compilation » (**Start your Build**). L'opération dure un moment (chez moi 3 min environ), puis on reçoit par courriel un lien de téléchargement, valable 24 h.

Il faut encore téléverser le micrologiciel compilé vers la carte. Il existe pour cela un outil pratique, le *NodeMCU PyFlasher* (**fig. 3**), disponible au téléchargement sous [3] pour Windows ou MacOS. On obtient un fichier directement exécutable, qui ne requiert pas d'installation. On lui indique le port COM et le nom du fichier contenant le micrologiciel et on lance le téléversement. Chez moi, cela marche sans aucun problème. On trouvera sur la page de documentation [4] une description complète de la compilation et du téléversement du micrologiciel, ainsi que des informations sur les différents modules et des foires aux questions.

Hello World – Test Lua

Sur la page de documentation, j'ai également découvert un lien vers le très utile outil **ESPlorer**. Ce programme (**fig. 4**) n'est pas un simple moniteur série, mais une espèce de cou-
teau suisse pour l'ESP8266 – peu importe qu'on utilise Micro-

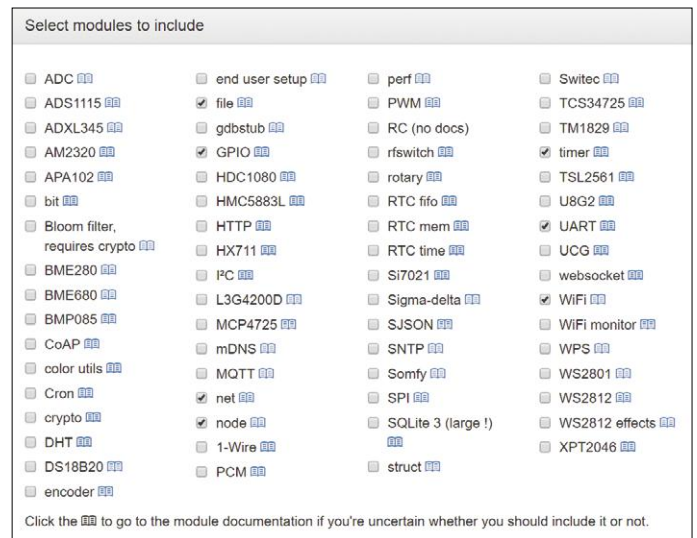


Figure 2. Création du micrologiciel avec tous les modules.

Python, NodeMCU ou un micrologiciel AT.

Après le téléchargement et le démarrage du fichier exécutable **ESPlorer.jar**, j'ai réalisé la connexion série avec la carte. Avec le nouveau progiciel, la vitesse est passée à 115200 baud. Le programme a essayé d'établir un dialogue automatique avec la carte, mais sans succès. C'est seulement après avoir appuyé sur le bouton-poussoir de réinitialisation que j'ai reçu les premières informations sur le micrologiciel actif, y compris tous les modules. C'est donc que la communication fonctionnait.

Le programme **ESPlorer** présente quelques touches, sur le bord de la fenêtre, permettant d'envoyer des commandes automatiques à la carte. C'est ainsi que j'ai pu obtenir l'identification de la puce ou des informations sur le système de fichiers. Bien entendu, il existe aussi une fenêtre pour la saisie directe

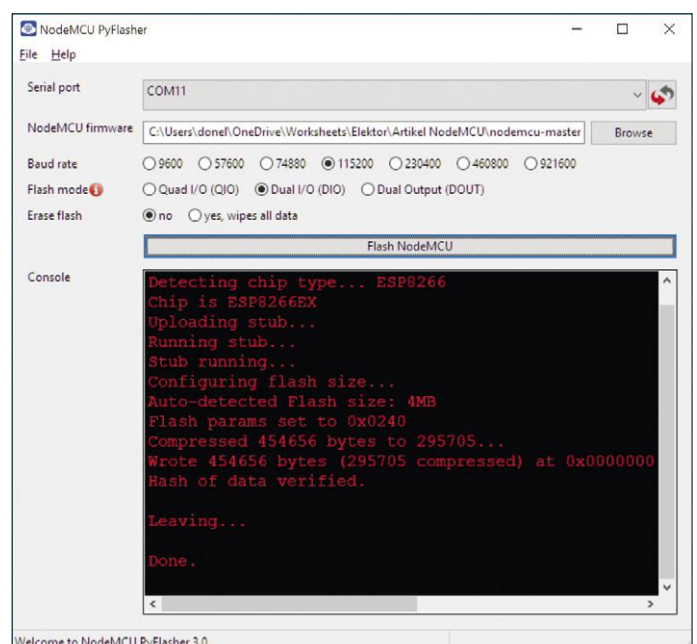


Figure 3. L'outil PyFlasher.

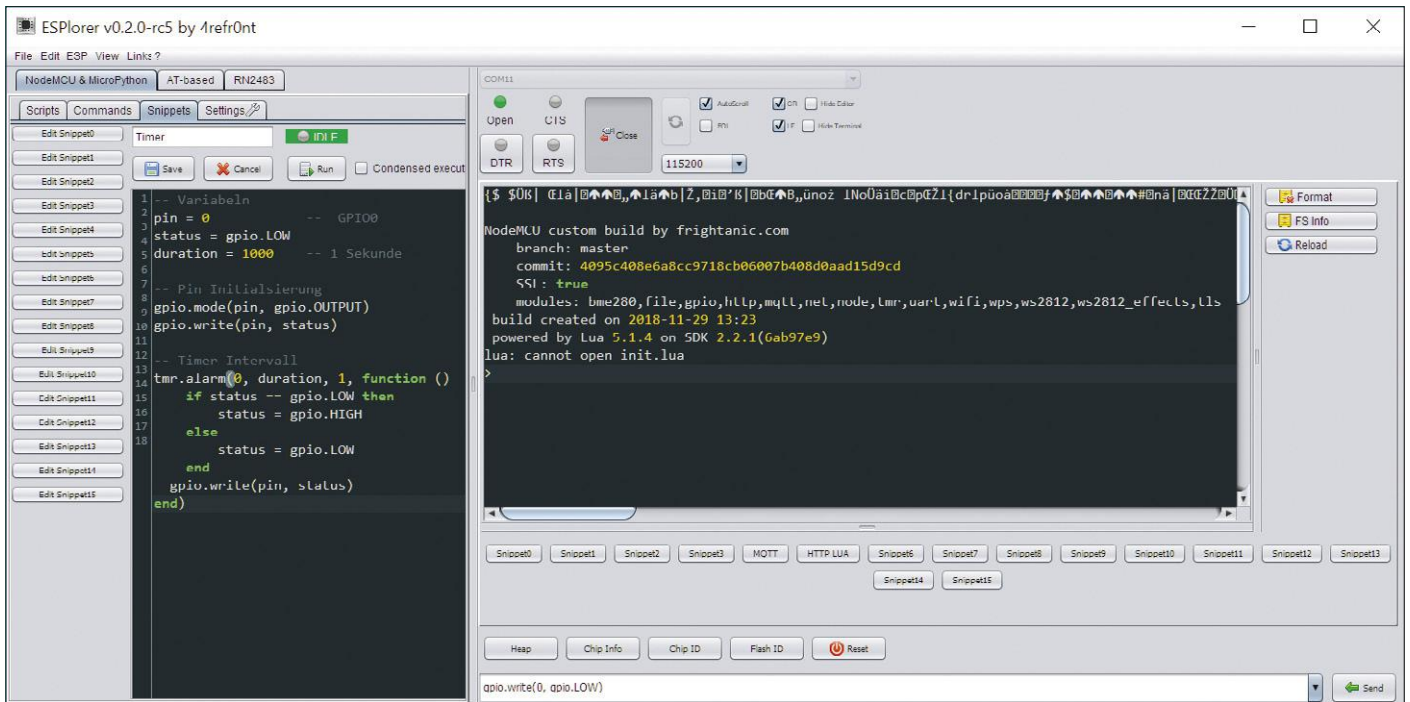


Figure 4. L'outil ESPlorer : à droite la fenêtre de terminal, à gauche les snippets.

des commandes. J'ai commencé par essayer les commandes suivantes :

```
gpio.mode(0, gpio.OUTPUT)
gpio.write(0, gpio.LOW)
```

La syntaxe rappelle le code source Arduino, ce qui rend les commandes faciles à lire. Ici, la broche 0 est configurée en sortie puis mise à la valeur LOW. La broche 0 est connectée à la LED embarquée qui est reliée à VCC et donc s'allume. Voilà qui n'était qu'un premier et simple test de vérification du fonctionnement des commandes sans compilation.

Si l'on ne veut tester qu'une ou deux commandes, la fenêtre de saisie manuelle est très pratique, en particulier parce qu'elle enregistre le fil et facilite considérablement une répétition de la commande. Je voulais réaliser le classique « Hello world » et faire clignoter la LED. Ça demande d'ajouter quelques lignes, mais ESPlorer offre la possibilité de saisir des fragments de code entiers et même de les enregistrer sur le PC. Ces fragments (*snippets*) peuvent ensuite être envoyés automatiquement vers la carte sur simple clic. J'ai saisi le programme de clignotement suivant, que j'ai enregistré dans l'un des 15 *snippets* éditables puis démarré par *Run* :

```
-- Variables
pin = 0          -- GPIO0
status = gpio.LOW
duration = 1000  -- 1 sec
-- Initialisation de broche
gpio.mode(pin, gpio.OUTPUT)
gpio.write(pin, status)
-- Intervalle de tempo (timer)
tmr.alarm(0, duration, 1, function ()
    if status == gpio.LOW then
```

```
        status = gpio.HIGH
    else
        status = gpio.LOW
    end
    gpio.write(pin, status)
end)
```

Il est intéressant de noter qu'une déclaration préalable des variables n'est pas nécessaire. Ceux qui ont déjà programmé en JavaScript n'en seront sans doute pas surpris. Tout aussi intéressante est l'utilisation de la temporisation (*timer*). Ici, on passe au *timer* une fonction complète, entièrement définie dans l'appel, dite fonction anonyme, qu'il doit exécuter toutes les secondes. Faire clignoter la LED n'est donc pas un problème et le programme reste intelligible. Mais ici, il est question d'une carte pour l'Internet des Objets, c'est pourquoi j'ai écrit les *snippets* suivants pour réaliser la connexion avec mon routeur.

```
wifi.setmode(wifi.STATION)
station_cfg={}
station_cfg.ssid="YourSSID"
station_cfg.pwd="YourPassword"
station_cfg.save=true
station_cfg.auto=true
wifi.sta.config(station_cfg)
wifi.sta.connect()

wifi.eventmon.register(wifi.eventmon.STA_CONNECTED,
function(T)
    print("Wi-Fi connected!")
end)
```

Après l'établissement de la connexion avec succès, le programme affiche le message « WiFi connected! ». Pour cela, j'ai

Liens

- [1] Dépôt Git : <https://github.com/nodemcu/nodemcu-firmware>
- [2] Configuration et compilation du micrologiciel : <https://nodemcu-build.com/>
- [3] PyFlasher : <https://github.com/marcelstoer/nodemcu-pyflasher/releases>
- [4] Documentation : <https://nodemcu.readthedocs.io/>
- [5] ESPlorer : <https://esp8266.ru/esplorer/#download>
- [6] Exemple MQTT : <https://nodemcu.readthedocs.io/en/master/en/modules/mqtt/#example>
- [7] Site web de l'auteur : <https://fkainka.de/>

dû créer une nouvelle fonction anonyme exécutée pour l'événement « connexion au Wi-Fi ». Ce genre de programmation pilotée par événement est typique avec Lua.

L'établissement de la connexion a parfaitement fonctionné. En outre, la configuration utilisée ici s'occupe d'enregistrer les données Wi-Fi pour qu'elles soient disponibles après une réinitialisation. De plus, `station_cfg.auto=true` assure que la carte essaie automatiquement d'établir une connexion avec le réseau enregistré. Mais sans programme approprié, le message « Wi-Fi connected! » n'apparaîtra pas, car cette partie ne survivra pas à une réinitialisation.

Après le brillant succès de l'établissement de la connexion, j'ai voulu passer directement au test du module MQTT. La documentation fournit un bon exemple que j'ai directement copié dans un *snippet*, légèrement adapté et testé immédiatement sur la carte [6]. Ici aussi, cela a parfaitement marché, si bien que j'ai décidé de passer au niveau supérieur.

Le pas suivant fut donc d'écrire un programme qui serait stocké à demeure sur la carte et serait automatiquement exécuté au démarrage. Cela s'est avéré très simple, considérant que :

- Le progiciel met à disposition son propre système de fichiers virtuel dans lequel on peut téléverser des fichiers. Ces fichiers peuvent être des scripts exécutables avec la commande `dofile(nom_de_fichier)`, des pages web, des images ou tout autre fichier qu'on veut mettre à disposition sur la carte.
- Au démarrage, la carte teste la présence du fichier `init.lua`. S'il existe, il est automatiquement exécuté.

Concrètement, cela signifie pour moi que dans ESPlorer je suis tout simplement passé de l'onglet *Snippets* à l'onglet *Scripts* qui inclut un éditeur de texte. Là, j'ai saisi les lignes suivantes, que j'ai d'abord enregistrées sur mon PC sous le nom `init.lua`, puis je les ai téléversées sur la carte NodeMCU :

```
wifi.eventmon.register(wifi.eventmon.STA_CONNECTED,
function(T)
    print("Wi-Fi connected!")
    http.get("https://pastebin.com/raw/k4ccGx3T", nil,
function(code, data)
    if (code == 200) then
        if file.open("run.lua","w+") then
            file.write(data)
            file.close()
        end
    end
end
else
```

```
    print("HTTP request failed")
end
```

```
    if file.exists("run.lua") then dofile('run.lua')
else print("run.lua not found") end
end)
end)
```

Ce programme fait quelque chose dont seuls les scripts sont en principe capables. Il commence par attendre l'établissement d'une connexion Wi-Fi. Mais ensuite, en cours d'exécution, il télécharge un programme sur l'internet, l'enregistre sous le nom `run.lua` puis l'exécute. Dans cet exemple, il charge un simple programme de clignotement de LED depuis une page du site pastebin.com. Bien entendu, il est possible d'utiliser n'importe quelle autre source. Ce micrologiciel utilise le module HTTP ainsi que l'option TLS/SSL.

Voilà qui fait de Lua un langage interprété vraiment passionnant. Je m'imagine toute une série de programmes résidants sur un serveur et constamment adaptés et améliorés. Avec HTTP ou MQTT, on choisit son programme et on le charge sur la carte. Plus modulaire et plus simple, il n'y a pas. ◀

(180604-04 – version française : Helmut Müller)



@ WWW.ELEKTOR.FR

→ Carte à microcontrôleur NodeMCU, réf. 17952
www.elektor.fr/nodemcu-microcontroller

→ Livre « Apprendre à programmer avec Python 3 »
www.elektor.fr/apprendre-a-programmer-avec-python-3

→ Livre en anglais 'ESP8266 and MicroPython'
www.elektor.fr/esp8266-and-micropython

→ Livre en anglais 'Python for Microcontrollers: Getting Started with MicroPython'
www.elektor.fr/python-for-microcontrollers

horloge rétro



à afficheurs de flipper

100
1
2

INFOS SUR LE PROJET



flipper retro
ESP12-F ESP8266
eBay



débutant
→ connaisseur
expert



env. 2 h
(hors nettoyage des
rouleaux)



câble FTDI de 3,3 V ou
autre interface USB-UART



75 €
(rouleaux non compris)

Ah... le son apaisant des afficheurs de score des vieux flippers...

Luc Lemmens (labo d'Elektor)

Les afficheurs rétro attirent l'œil par leur charme discret. Celui de cette horloge attirera aussi les oreilles par son charme **bruyant**. Avec elle, ce n'est plus le temps qui s'en va doux-coulant, mais Chronos qui se tape une partie de flipper !

La plupart des afficheurs rétro présentés dans nos pages sont hérissés de tubes Nixie, Numitron ou VFD. Parmi les exceptions figurent un thermomètre recyclant un afficheur électromécanique à rouleaux de flipper [1], ainsi qu'une déclinaison de ce même thermomètre qui utilisait cette fois-ci les rouleaux indiquant les unités de crédit du joueur [2]. L'heure est venue de surfer à nouveau sur ces rouleaux ! Un projet d'horloge n'est peut-être pas d'une originalité décoiffante, certes, mais a pour vertu d'être éminemment pratique, même avec une « bande sonore » qui risque de compromettre la quiétude de votre salon.

En hommage à tous ces antiques afficheurs électromécaniques (EM) qui auront aussi fait l'ambiance des cafés, notre projet cache par ailleurs un jeu de flipper rudimentaire qui, comme il se doit, indique le score avec les rouleaux ! Les schémas d'horloge-flipper ne manquent pas sur la toile mais, pour autant que nous puissions en juger, la nôtre se démarque comme étant l'une des plus étudiées et anachroniques qui soient. Grâce à son ESP8266 notamment, l'heure et la date de notre horloge sont en effet synchronisées par protocole NTP, l'accès au serveur de temps se faisant par Wi-Fi. La configuration est effectuée d'ailleurs elle aussi par Wi-Fi (ordiphone, tablette ou PC), heure et date pouvant cependant également être ajustées manuellement. Une puce DS3231 à RTC (horloge à temps réel) alimentée par pile assure quant à elle une précision de 5 ppm, même lorsque l'horloge est hors tension.

Vous pouvez paramétrer un intervalle horaire durant lequel l'horloge restera silencieuse, cloches et rouleaux étant alors désactivés, mais pour la faire taire vous pouvez aussi gentiment l'éteindre

(ou tirer la fiche de la prise...)

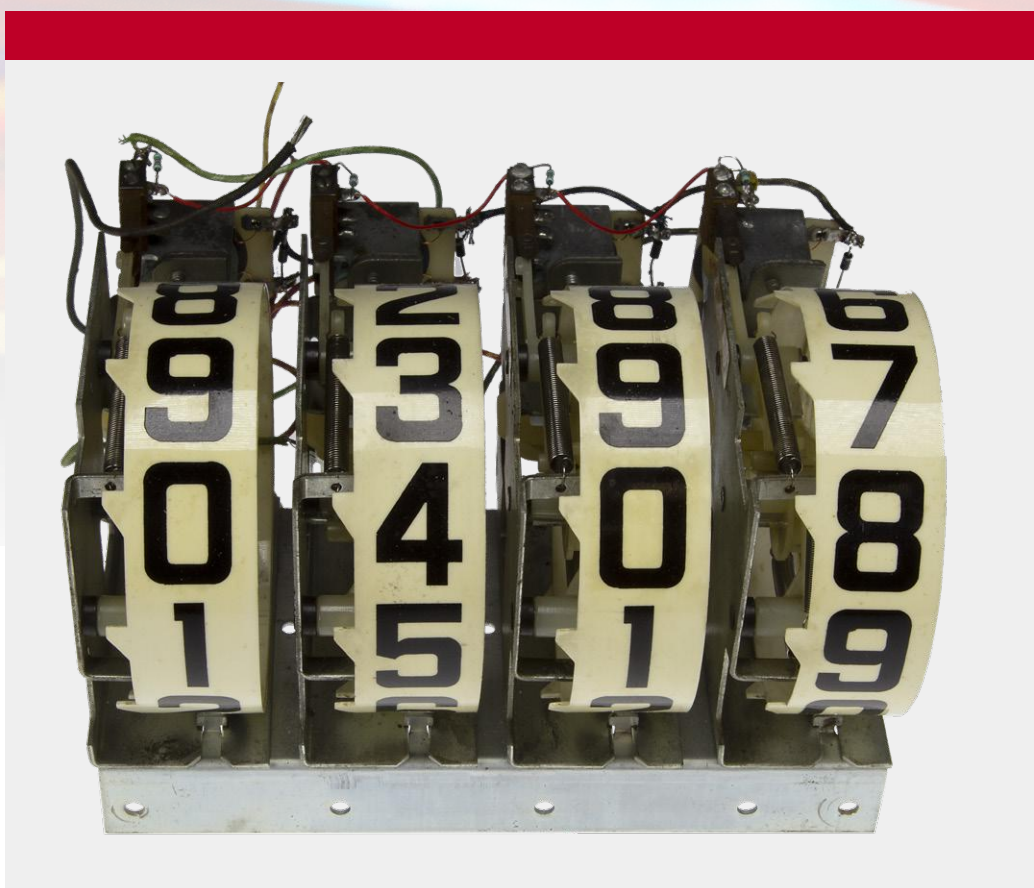
L'heure peut être au format 12 ou 24 h, et la date s'affiche toutes les 30 min (au format européen ou US).

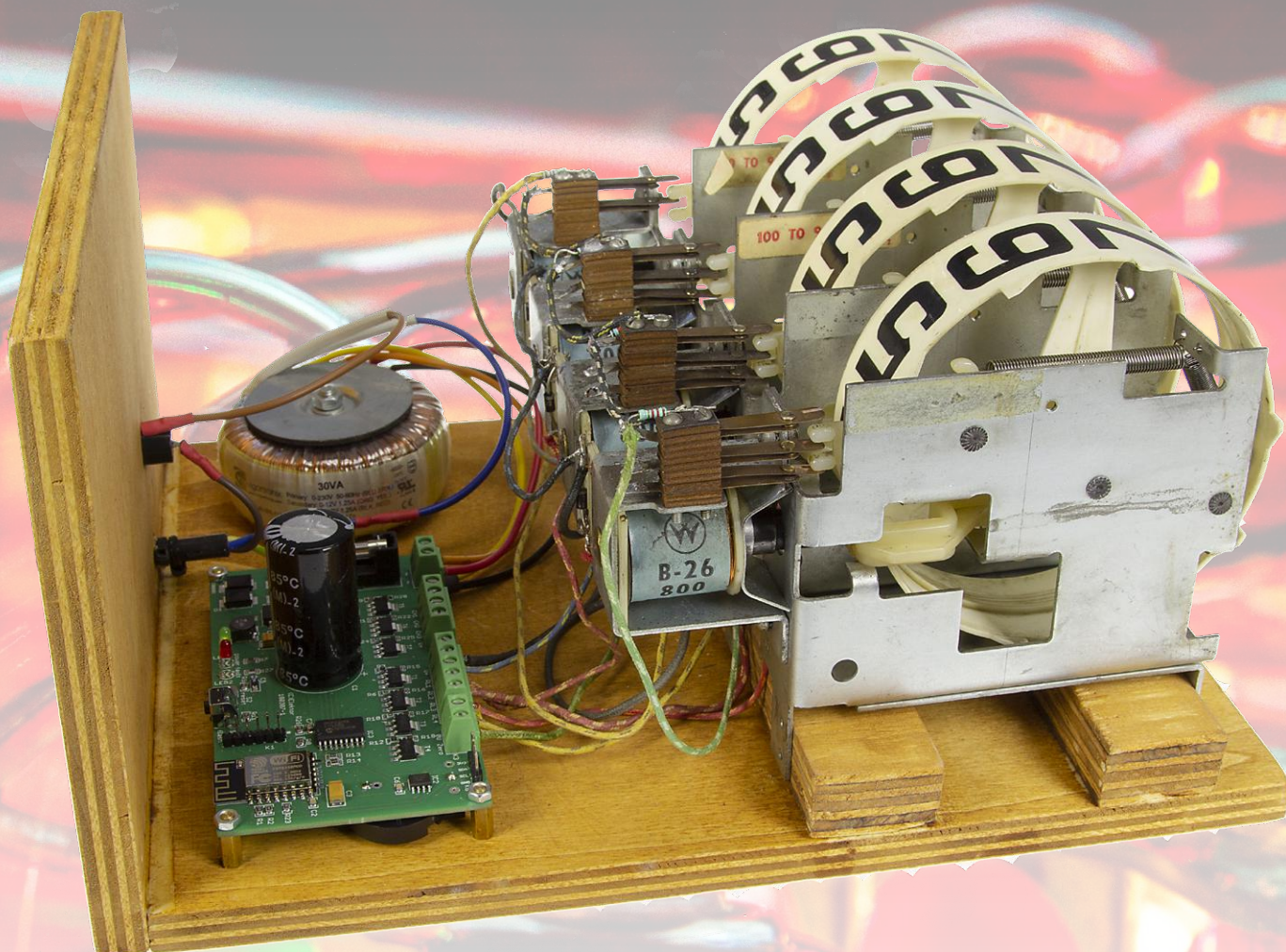
Une ère électromécanique

La plupart des flippers construits entre les années 1950 et le milieu des années 1970 affichaient leur score au moyen de compteurs électromécaniques à rouleaux. L'incrémentation d'un rouleau se faisait par application d'une brève impulsion à une bobine, principe qui n'a guère évolué au fil des ans. Un contact s'ouvrait lorsque le rouleau revenait à

zéro, la remise à zéro du compteur pour une nouvelle partie se faisant par l'intermédiaire de ce contact.

Un rouleau de score comporte à vrai dire plusieurs contacts, dont un signalant que la roue a terminé son incrément, et un se fermant en position « 9 » pour faire avancer le rouleau adjacent. Certains rouleaux peuvent même enregistrer leur position comme score à battre ou pour une loterie ayant pour lot une partie gratuite. Ceci dit, nous n'utiliserons ici que la bobine et le contact de détection de la position zéro (appelé contact zéro par la suite). Dans les années 1960, le diamètre des





Où acheter des rouleaux de score, et lesquels ?

Les rouleaux de marques célèbres, telles que Gottlieb, Bally et Williams, conviennent à notre horloge, mais ceux de marques moins connues feront aussi l'affaire. On en trouve sur des sites de petites annonces et d'enchères, mais aussi sur des sites vendant des pièces reconditionnées ou de récupération.

Les tailles et les polices dépendent autant de la marque que de l'âge des rouleaux. Pour l'esthétique de l'horloge, trouvez-en si possible quatre identiques !

Nettoyage

Les rouleaux que vous aurez en main seront probablement crasseux, poussiéreux et couverts de suies. À défaut d'être passionnant, leur nettoyage est donc indispensable. Des pièces mécaniques sales risquent en effet d'entraver leur bon fonctionnement. Ne vous contentez pas de briquer l'extérieur, démontez (entièrement !) les rouleaux et nettoyez-les en profondeur.

Voici quelques-unes des choses à faire et à ne pas faire :

- le plus important : manipulez et nettoyez les rouleaux avec précaution sous peine d'abîmer leurs chiffres, voire de les effacer. Les rouleaux Gottlieb étaient p. ex. (apparemment) imprimés avec de l'encre à base d'eau... En général, il ne faut ni utiliser de nettoyants (agressifs), ni tremper les rouleaux dans l'eau, ni les frotter, ni les gratter.
- Pour le ré-assemblage, aidez-vous des encoches ou marques toujours présentes sur un rouleau pour l'orienter correctement sur son moyeu. Un rouleau mal aligné peut endommager l'unité, ou à tout le moins décaler la position zéro.
- Si un rouleau ne tourne pas correctement, c'est que ses parties mobiles ne sont pas suffisamment propres OU qu'il a été mal remonté.

Quoi qu'il en soit, NE le lubrifiez PAS !

- Les contacts zéro sont assez faciles à identifier : cherchez un contact à lame qui s'ouvre lorsque le rouleau est en position zéro. L'identification est facile sur les rouleaux Bally et Williams plus récents, il s'agit du contact placé au-dessus de l'unité. Les machines Gottlieb sont à cet égard moins faciles à entretenir puisque les contacts sont cachés à l'intérieur de l'unité ; il peut être judicieux de chercher le contact zéro une fois que vous avez démonté l'unité pour la nettoyer. Dans tous les cas, nettoyez les points de contact avec du papier de verre à grain fin, et ajustez les lames si nécessaire.

On peut trouver sur l'internet quelques vidéos expliquant comment démonter, nettoyer et réparer des rouleaux de score (<https://youtu.be/1tYzWID41jc> p. ex.).

rouleaux est passé de 5 à 4 pouces, et plus tard a même été réduit à 3 pouces. L'inertie des compteurs dépendait en effet de leur diamètre, le réduire permettait donc une meilleure synchronisation entre les actions du joueur et l'affichage. C'est d'ailleurs la raison pour laquelle les vieux flippers semblent moins vifs que les modèles récents. Pour reproduire au plus près la cadence sonore des anciens compteurs de score, l'afficheur de notre

horloge est réglable en fonction de la dimension des rouleaux. Les compteurs à rouleaux ne pouvant qu'être **incrémentés**, il faudra par exemple neuf impulsions pour remettre à 0 un rouleau s'il est sur la position 1. Inutile de dire que cette horloge sera la plus bruyante de toutes celles présentées par Elektor. Si à ce tilt-tac vous souhaitez ajouter un système de cloche EM à trois tons pour sonner les quarts, les

demis et les heures, nous avons prévu un connecteur pour cette option.

Toutefois, avant même d'y songer, notez que les cloches de flipper deviennent rares et donc, inévitablement, chères. À moins bien sûr de pouvoir en récupérer sur une vieille machine.

Les bobines des flippers EM sont alimentées en CA, à l'exception des batteurs et des champignons actifs apparus plus tard (les *pop bumpers*). Cependant les bobines CA peuvent aussi être alimentées en CC, mais avec une tension inférieure à leur valeur CA nominale [3]. Notez que la page du lien précédent étudie le cas de relais CA alimentés en permanence par une source CC. Dans notre horloge, de même que dans les flippers, les solénoïdes (du moins la plupart) sont alimentés uniquement par des impulsions brèves, d'où la faible probabilité d'une surchauffe. Un dysfonctionnement de l'horloge entraînant une alimentation permanente des bobines n'étant toutefois pas à exclure, un fusible a été prévu. Dans un flipper, un contact de rouleau collé en position fermé ferait sauter le fusible de 10 A ou plus : l'isolement des enroulements fondrait, transformant la bobine en court-circuit. Bref, ces bobines n'ont jamais été conçues pour être alimentées en permanence !

Machines Williams exceptées (pré-1963 : 50 V_{CA}), la tension des bobines des flippers EM Williams et Gottlieb étaient de 28 V_{CA}, et l'ensemble des Bally utilisaient 50 V_{CA}. Tous les rouleaux fonctionneront correctement sous environ 35 V_{CC} mais les Williams et Gottlieb accepteront aussi une tension inférieure (env. 20 V_{CC}).

Schéma

Le schéma de l'horloge (**fig. 1**) est assez simple, mais j'en détaillerai tout de même toutes les parties.

L'alimentation est un transformateur toroïdal de 2×12 V dont les secondaires en série sont reliés à K4, soit, en théorie, une tension de bobine d'environ 32 V_{CC} sur la borne positive du pont redresseur D1-D4. En pratique, les secondaires du transformateur non chargé afficheront une tension de sortie bien plus élevée et la tension de bobine dépassera 35 V_{CC}. Pour des rouleaux Williams ou Gottlieb, un transformateur de 2×9 V convient.

Le circuit intégré IC1 est un convertisseur abaisseur LM2576 délivrant 3,3 V à la section numérique. Pour ne prendre aucun risque, nous avons opté pour un modèle HT à tension d'entrée maximale

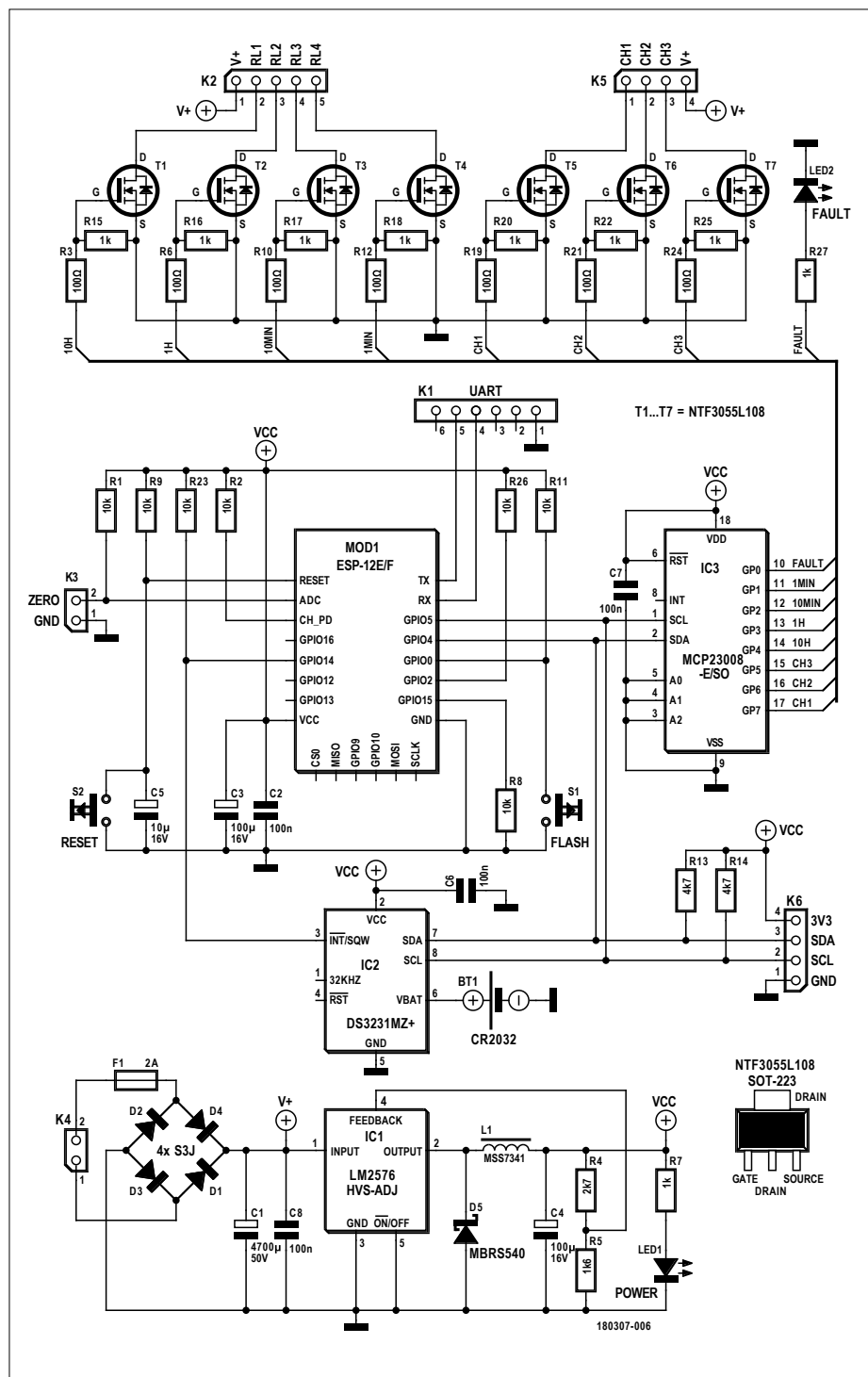


Figure 1. Schéma de l'horloge à afficheur EM. Avec ou sans cloches, son tic-tac est véritablement martelé !

de 60 V, contre 40 V pour les modèles « ordinaires ». LED1 est allumée lorsque l'alimentation de 3,3 V est présente.

Disons un mot sur le condensateur électrolytique C1, l'un des rares traversants de ce projet. Il ressemble peut-être à un vieux moulin au milieu d'éoliennes, mais il s'agit d'un choix délibéré. D'abord parce qu'il est difficile – voire impossible – de trouver de gros électrolytiques CMS de plus de 35 V. Ensuite parce qu'un condensateur traversant offre une meilleure stabilité mécanique et, argument décisif, parce qu'il est possible de se procurer rapidement des 50 V (et même plus).

Les bobines des rouleaux ainsi que celles des cloches sont commandées par des MOSFET NTF3055 à canal N. Leur tension de seuil est suffisamment basse pour qu'ils délivrent fidèlement aux bobines des impulsions avec des sorties logiques de 3,3 V. Dans notre premier prototype, ces transistors étaient directement commandés par les broches GPIO du contrôleur. Assurément une mauvaise idée : l'ESP8266 n'a pas beaucoup d'E/S, et certaines d'entre elles ont des rôles particuliers durant la mise sous tension. Il était dès lors très difficile de conserver des bobines complètement hors tension durant le démarrage. Le recours à l'expansor d'E/S IC3 a facilité la commande du matériel, tout en offrant une sorte de protection/tampon entre l'ESP et les MOSFET.

Au départ, nous avons connecté les quatre contacts zéro au CA/N de l'ESP pour économiser les broches d'E/S. Bien que ce CA/N ne soit pas vraiment précis, cette solution s'est avérée assez fiable pour que nous la gardions. L'ESP détecte uniquement un changement de la valeur donnée par le CA/N ; il peut seulement déterminer si un contact zéro est ouvert après que la bobine d'un rouleau a reçu une impulsion, c.-à-d. lorsque ce rouleau a atteint sa position zéro. Toutes les roues tournent donc lors d'une remise à zéro complète, même si celles-ci étaient déjà en position zéro.

Une résistance de 1,2 k Ω doit être soudée aux bornes des quatre contacts zéro (fig. 2), tous en série et reliés à K3.

Si la valeur donnée par le CA/N reste inchangée après 12 impulsions sur un rouleau, la LED2 (FAULT) s'allume, et l'envoi d'impulsions sur la bobine associée est suspendu jusqu'à ce que l'horloge soit remise à zéro ; avant de procéder à cette initialisation, vérifiez toutefois

les contacts zéro et leur câblage. La mise en défaut précédente a été implantée à la fois pour préserver vos nerfs et protéger la bobine contre toute surchauffe que provoquerait l'envoi d'impulsions cherchant à atteindre une position « zéro » qui ne le serait jamais.

Le connecteur K6 attend un éventuel afficheur OLED I²C montrant les informations d'état de l'horloge, ou tout autre périphérique enrichissant le projet, p. ex. un capteur de température I²C.

S2 est le bouton d'initialisation (reset). Si vous maintenez S1 enfoncé jusqu'au relâchement de S2, l'ESP démarre en mode *bootloader*, ce qui permet ainsi de reprogrammer l'horloge via le connecteur UART K1. S1 permet également d'initialiser l'horloge avec ses paramètres par défaut, ce qui peut s'avérer utile pour la connecter à un réseau Wi-Fi différent. Pour cela, initialisez-la et appuyez sur S1 tant que la LED rouge FAULT reste allumée.

Le circuit IC2 est la très précise (5 ppm !) horloge à temps réel DS3231. Comme elle est alimentée par une pile bouton, l'heure exacte est conservée lorsque l'horloge est mise hors tension. Inutile



donc de la remettre à l'heure à chaque redémarrage si elle n'est pas synchronisée via NTP.

Construction et câblage

L'horloge fonctionne avec quatre rouleaux de score. Soyez certain d'en trouver avant d'acheter les autres composants ! Comme dit plus haut, l'ajout d'une cloche de flipper à trois tons est optionnelle. De toute façon, il y a de fortes chances pour que vos oreilles finissent par vous demander de la déconnecter...

Le soudage des CMS sur le circuit imprimé ne devrait pas poser de difficultés avec un petit fer à souder et du fil fin (de préférence < 1 mm). Soudez d'abord une pastille de chaque compo-

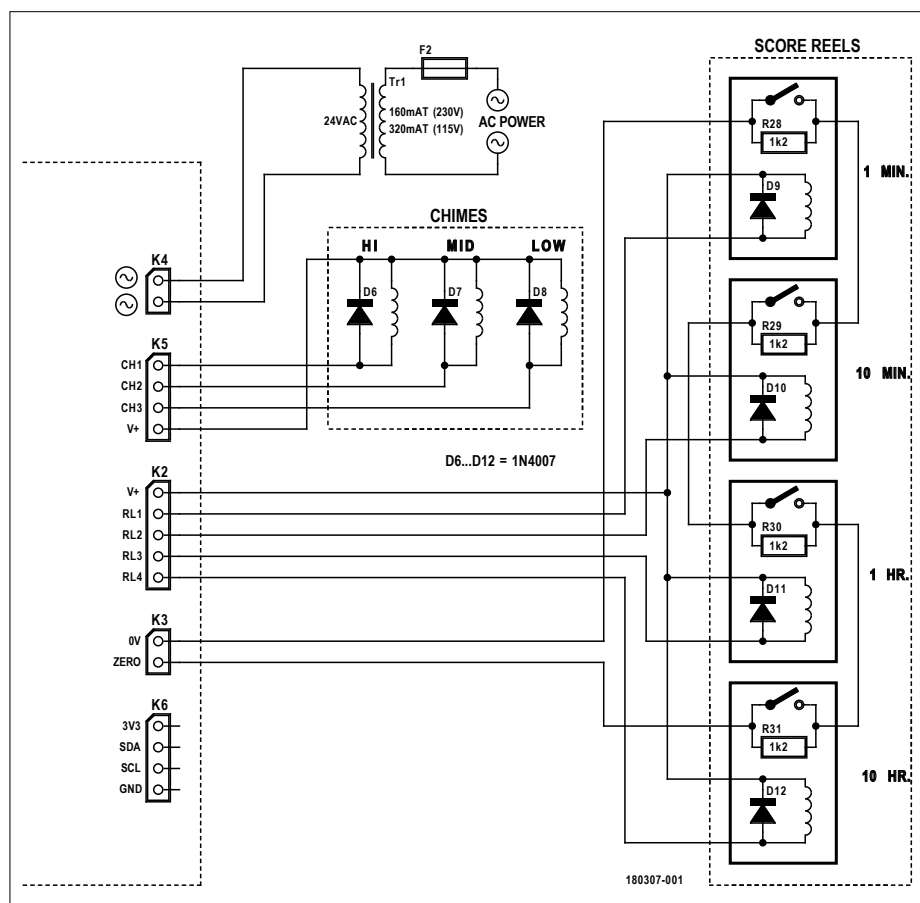


Figure 2. Schéma de câblage de l'horloge.



LISTE DES COMPOSANTS

Rendez-vous sur la plateforme GroupGets pour commander ce projet : www.elektormagazine.com/pinball-clock

Résistances

R1, R2, R8, R9, R11, R23, R26 = 10 k Ω , couche épaisse, 5 %, 0,1 W, 150 V
 R3, R6, R10, R12, R19, R21, R24 = 100 Ω , couche épaisse, 5 %, 0,1 W, 150 V
 R4 = 2,7 k Ω , couche épaisse, 5 %, 0,1 W, 150 V
 R5 = 1k6 Ω , couche épaisse, 1 %, 0,1 W, 150 V
 R7, R15, R16, R17, R18, R20, R22, R25, R27 = 1 k Ω , couche épaisse, 5 %, 0,1 W, 150 V
 R13, R14 = 4,7 k Ω , couche épaisse, 5 %, 0,1 W, 150 V

Condensateurs

C1 = 4700 μ F, 50 V, au pas de 10 mm, 22×41 mm
 C2, C6, C7, C8 = 100 nF, 50 V, X7R, 0805
 C3, C4 = 100 μ F, 16 V, 2312
 C5 = 10 μ F, 16 V, 1206

Inductance

L1 = inductance de puissance 470 μ H (CMS), 570 mA, blindée, 270 mA, série MSS7341T

Semi-conducteurs

D1, D2, D3, D4 = S3J+, 600 V, 3 A
 D5 = MBR540, 40 V, 5 A, V_f =550 mV @ I_F =5 A
 LED1 = verte, 3 mm
 LED2 = rouge, 3 mm
 T1, T2, T3, T4, T5, T6, T7 = MOSFET NTF3055L108T1G, canal N, 3 A, 60 V, 120 m Ω
 IC1 = LM2576HVS-ADJ, régulateur abaisseur, 4-60 V, 3 A
 IC2 = DS3231MZ+ RTC, I²C, 5 ppm
 IC3 = MCP23008-E/SO, extenseur d'E/S à 8 bits, I²C
 MOD1 = ESP-12F, module Wi-Fi à ESP8266

Divers et/ou hors circuit imprimé

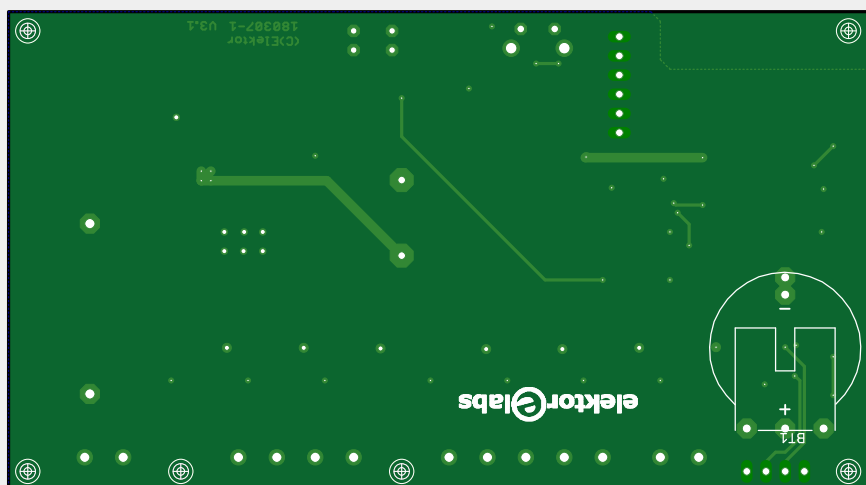
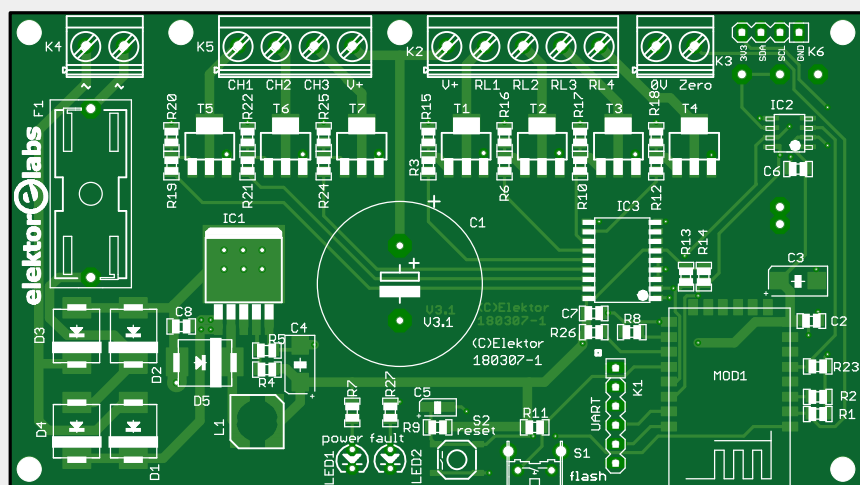
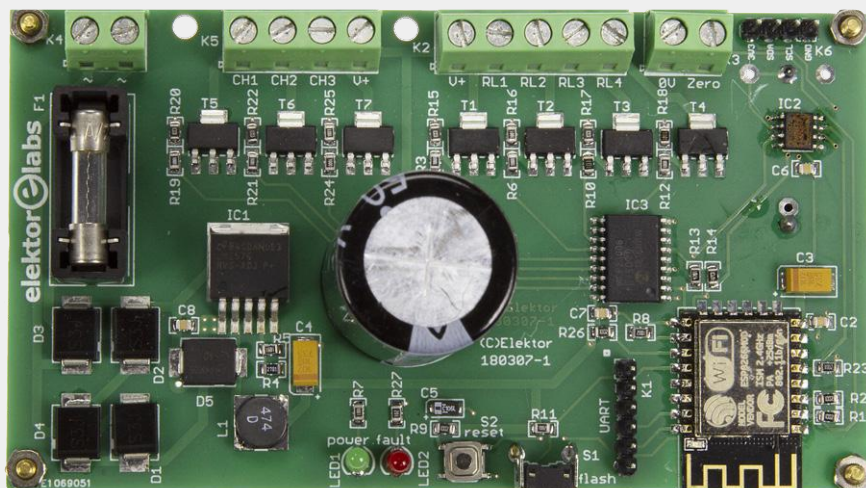
Bt1 = pile CR2032 avec support de pile, p. ex. Keystone 1066
 D6, D7, D8, D9, D10, D11, D12 = 1N4007, 1000 V, 1 A (hors circuit imprimé !)
 F1 = fusible 2 AT, 5×20 mm, support encartable, au pas de 22,6 mm
 F2 = fusible 160 mA, 5×20 mm (secteur 230 V) avec porte-fusible, montage sur panneau, bornes à souder
 F2 = fusible 320 mA, 5×20 mm (secteur 115 V) avec porte-fusible, montage sur panneau, bornes à souder
 K1 = barrette mâle à 6 picots, au pas de 2,54 mm, verticale
 K2 = bornier à 5 vis, au pas de 5,08 mm, 630 V
 K3, K4 = bornier à 2 vis, au pas de 5,08 mm, 630 V
 K5 = bornier à 4 vis, au pas de 5,08 mm, 630 V
 K6 = barrette mâle à 4 picots, au pas de 2,54 mm, verticale
 LCD1 = afficheur de 0,96", LCD OLED, 128×64 pour Arduino (optionnel)

R28, R29, R30, R31 = 1,2 k Ω , couche de carbone, 5 %, 0,25 W, 250 V (hors circuit imprimé !)

S1 = poussoir à retour tactile, p. ex. SKHHLVA010

S2 = poussoir à retour tactile, 12 V, 50 mA, série TM de Multicomp

Tr1 = transformateur, toroïdal, 2×12 V, 30 VA, p. ex. MCTA030/12 de Multicomp
 Connecteur d'entrée secteur, IEC320 C8 + câble
 Circuit imprimé, réf. 180307-1



sant, et vérifiez que toutes les autres connexions sont bien alignées avec le circuit imprimé. Vérifiez deux fois la polarité des semi-conducteurs et des condensateurs électrolytiques avant de souder les autres composants.

Connectez ensuite les éléments qui ne font pas partie du circuit imprimé – référez-vous au schéma de la **figure 2**.

Nettoyez vos rouleaux, et n'oubliez pas de monter une diode de roue libre de type 1N4007 (ou équivalent) aux bornes de chaque bobine (D9 à D12), idem pour celles de l'optionnelle cloche à trois tons (D6 à D8). Reliez à la connexion V+ (de K2 pour les bobines, de K5 pour les cloches) les pattes (languettes) des bobines connectées aux cathodes des diodes. Pour l'autre patte, reliez le rouleau des minutes à RL1 de K2, celui des dizaines à RL2 de K2, etc. Idem pour les trois cloches, celle de la note la plus haute allant en CH1 sur K5, etc. Soudez aux pattes des contacts zéro les résistances R28 à R31. Ensuite, reliez en chaîne ces contacts, les extrémités étant à relier à K3.

Les enroulements du secondaire du transformateur se connectent en K4. N'oubliez pas de protéger le primaire avec un fusible.

Micrologiciel et contenu web

La compilation et le chargement du code dans le module ESP12 de l'horloge nécessitent quelques préalables.

Il vous faudra avant tout disposer de l'EDI Arduino et du paquet pour cartes ESP8266 [4]. L'EDI prêt, installez le greffon `arduino-esp8266fs-plugin` qui servira à charger le contenu web de l'horloge [5]. Téléchargez ensuite le micrologiciel du projet depuis [6].

Pour compiler son code, vous devez avoir installé les bibliothèques suivantes :

- RTCLib d'Adafruit ;
- MCP23008 d'Adafruit ;
- TimeLib de Michael Margolis ;
- Arduino JSON 5.x ;
- WebSockets de Markus Sattler ;
- NtpClientLib [7] (utilisez la branche AsyncUDP) ;
- CRC32 de Christopher Baker.

Pour faciliter l'installation, nous avons inclus notre dossier Arduino « portable » à l'archive du projet [8]. Il contient le croquis complet, toutes les bibliothèques, ainsi que le greffon `esp8266fs-plugin`. Nous n'avons pas inclus le paquet pour cartes ESP8266 afin de réduire la taille de l'archive. Copiez le dossier `portable` dans le dossier d'installation Arduino de votre ordinateur.

- Allumez l'horloge et reliez un câble FTDI de 3,3 V à K1 (le fil noir GND pointant vers IC3) et à un port USB de votre PC. Maintenez enfoncé le bouton S1 (*Flash*), appuyez sur le bouton d'initialisation S2, relâchez-le, puis relâchez S1 : l'ESP démarre en mode *bootloader*.
- Depuis l'EDI Arduino, sélectionnez le port COM auquel est relié le câble FTDI, et choisissez comme carte cible `nodeMCU 1.0 (ESP-12E Module)`.
- Depuis l'EDI Arduino, ouvrez le fichier `PinballClock.INO` situé sous `..\\arduino\\portable\\sketchbook\\PinballClock`, puis compilez et téléversez le micrologiciel sur votre carte.

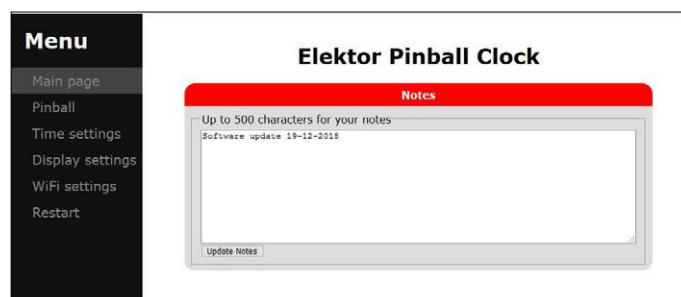


Figure 3. Outre le menu de configuration, la page web de l'horloge contient un bloc-notes rudimentaire.

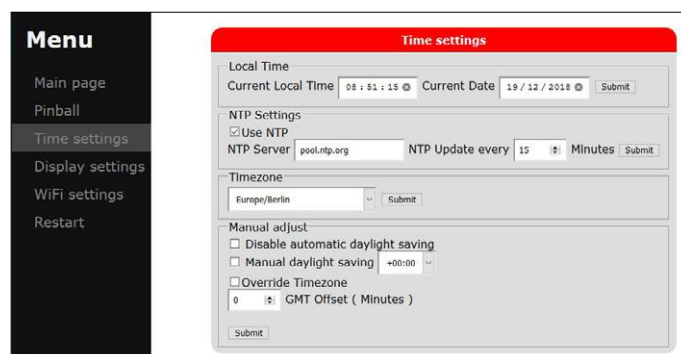


Figure 4. Menu de réglage de l'heure et du serveur NTP.



JACKPOT ! L'horloge Pinball d'Elektor touche hackaday : <https://bit.ly/2Rw9xln>

Pour finir, vous devez charger la page web ESP dont tous les fichiers sont dans le sous-dossier `data` du dossier du croquis Arduino. L'outil *Upload* du *plugin* sélectionnera automatiquement le contenu de ce dossier et le transférera dans la mémoire flash de l'ESP. Vérifiez d'abord que *Flash size* du menu *Outils* est sur *4M (1M SPIFFS)*. Répétez la séquence de pressions de S1/S2 pour mettre l'ESP en mode *bootloader*, puis chargez la page web de l'ESP en sélectionnant *ESP8266 Sketch Data Upload* depuis le menu *Outils*. L'opération prend un certain temps. Elle s'achève avec l'affichage de *SPIFFS Image Uploaded* dans la barre d'état.

Connexion Wi-Fi et réglages

Une fois la page web chargée, l'horloge s'initialise : les rouleaux se mettent sur zéro, puis affichent une heure aléatoire (donnée par la puce RTC non encore synchronisée). Observez la LED rouge (LED2). Si elle est allumée, vérifiez les contacts zéro des rouleaux ainsi que le câblage des résistances ; corrigez ce qui ne va pas et réinitialisez l'horloge avec S2. Si l'horloge n'arrive pas à se connecter à un réseau Wi-Fi (après gravure d'un nouveau *firmware* ou parce que le réseau est indisponible), elle paramètre un point d'accès sans fil ayant pour SSID *Elektor Pinball Clock*. Connectez-y votre PC, tablette ou mobile, et entrez dans la barre d'adresse de votre navigateur l'adresse IP 192.168.4.1. La page web qui s'ouvre (**fig. 3**)

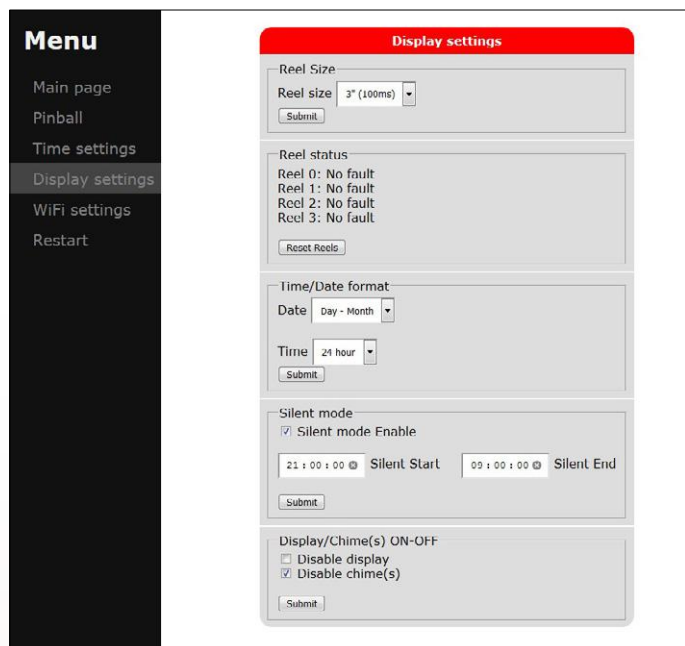


Figure 5. Menu du paramétrage de l'afficheur.

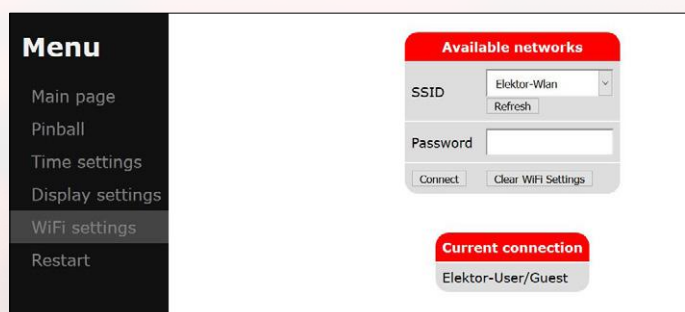
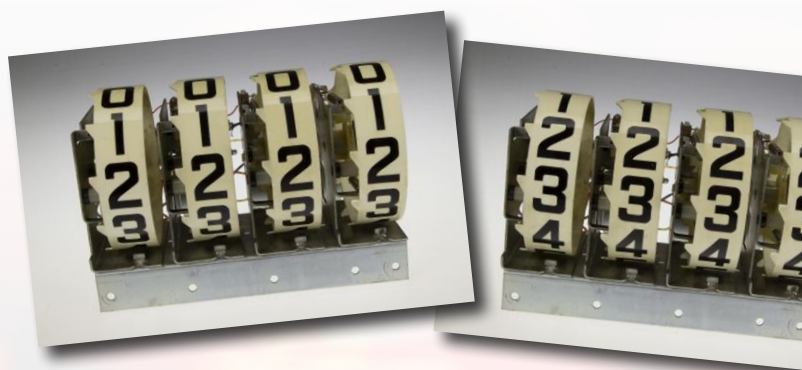


Figure 6. Menu de la connexion Wi-Fi.

contient une zone de texte pouvant servir de bloc-notes. Le menu *Time Settings* permet d'entrer manuellement l'heure et la date (**fig. 4**). Pour synchroniser l'horloge avec un serveur NTP, cherchez le SSID de votre réseau local dans *Wi-Fi Settings*, entrez votre mot de passe et cliquez sur *Submit*. L'ESP se réinitialisera et se connectera à ce réseau avec une adresse IP fournie par le serveur DHCP. Vous pouvez obtenir cette adresse de plusieurs façons :

- le câble FTDI encore branché, ouvrez le *Moniteur série* de l'EDI Arduino pour voir si l'ouverture de la session a réussi et quelle est l'adresse IP de l'horloge ;
- connectez l'optionnel afficheur OLED (LCD1), et après redémarrage listez les paramètres réseau ;
- cherchez dans les *Paramètres réseau* de votre PC (ou utilisez un utilitaire gratuit tel que *Free IP Scanner*) l'adresse IP ajoutée lorsque vous allumez l'horloge.



Son adresse IP connue, vous pouvez maintenant accéder à l'horloge via votre Wi-Fi local. N'oubliez pas de relier votre ordinateur ou mobile à ce même réseau. Le point d'accès de l'horloge est désactivé dès qu'elle se connecte à votre réseau Wi-Fi. Ouvrez à nouveau le menu *Time Settings* et entrez *pool.ntp.org* (ou tout autre serveur NTP de votre choix) ainsi que l'intervalle de synchronisation désiré. Sélectionnez votre fuseau horaire, paramétrez le reste à votre guise, et n'oubliez pas de valider chaque paramètre avec le bouton *Submit* correspondant.

Vous aimez les paramètres ? En voici d'autres :

- **Display settings (fig. 5)** : le paramètre **Reel Size** conditionne la vitesse d'avance des rouleaux. La valeur par défaut de 5" devrait convenir à tous les types et marques. Les petits rouleaux de 3" ont été conçus pour un fonctionnement en pas à pas plus rapide (meilleure synchronisation avec le score) et dans ce cas leur rotation semblera plus naturelle s'ils sont accélérés. Une vitesse d'avance rapide est par contre inadaptée aux plus gros (plus anciens) rouleaux.
- Il n'existe pas de rouleaux de 6" (du moins à notre connaissance), nous n'avons ajouté cette option que pour élargir le spectre des préférences sonores.
- **Reel status** indique le bon fonctionnement des contacts zéro. Utilisez-le pour régler un contact. Le bouton **Reset reels** amène les quatre rouleaux en position 0 et efface tout éventuel message d'erreur (si celle-ci a été corrigée bien sûr).
- **Time/Date display** détermine le format de l'heure (12h ou 24h) et de la date (mm-jj ou jj-mm).

► TILT ! L'horloge Pinball d'Elektor touche l'Italie : <https://bit.ly/2CRGm3O>

@ WWW.ELEKTOR.FR

→ **Circuit imprimé nu, réf. 180307-1**
www.elektor.fr/pinball-clock-180307-1

→ **ESP-12F, module Wi-Fi à ESP8266, réf. 17781**
www.elektor.fr/wi-fi-module-160100-92

→ **Afficheur (optionnel) de 0,96", LCD OLED, 128×64 pour Arduino**
www.elektor.fr/oled-display-for-arduino

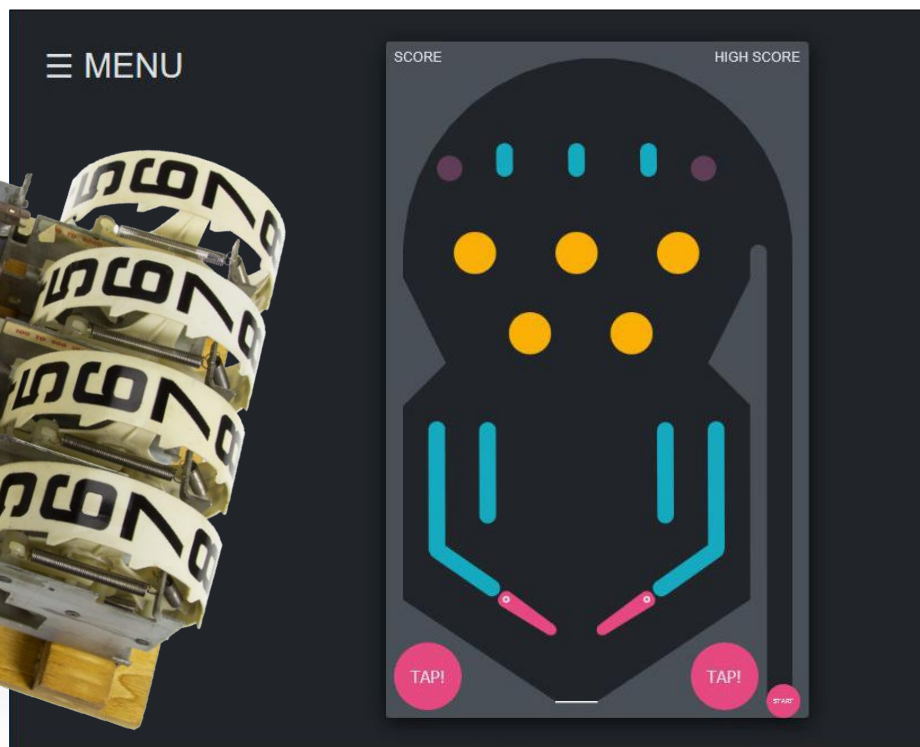
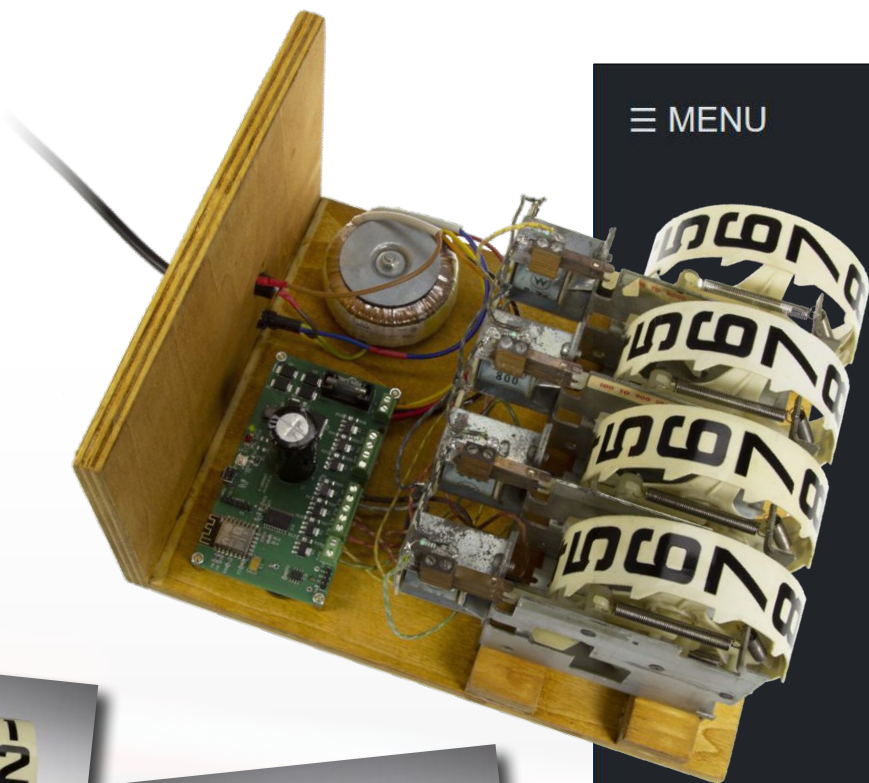


Figure 7. En bonus, ce jeu de flipper virtuel permet de jouer avec un véritable afficheur de score !



▶ EXTRA BALLS ! L'application de l'horloge cache un jeu de flipper gratuit.

- **Silent mode** permet de rendre l'horloge silencieuse (cloches et rouleaux) durant un certain intervalle horaire, que je devine être nocturne. Pour désactiver en permanence les rouleaux et/ou les cloches, servez-vous de l'option **Display/Chime(s) ON/OFF**. Les rouleaux se mettent sur 9999 lorsqu'ils sont désactivés.
- **WiFi settings (fig. 6)** affiche les réseaux disponibles. Le nom du

réseau utilisé est indiqué sous **Current network settings**. Le nom du réseau actif et le mot de passe correspondant sont mémorisés à la mise hors tension.

- **Restart**, vous vous en doutez, redémarre l'horloge. Notez qu'un redémarrage peut modifier l'adresse IP de l'horloge.

Nous avons gardé le plus plaisant pour la fin : le jeu de flipper du menu Pinball

(fig. 7). Lorsque vous le lancez, les rouleaux se mettent à zéro et attendent de marquer le score. Votre entrain sera de surcroît grisé par le son des cloches si vous les avez installées ! Notez que le mode Silent affecte aussi le jeu. Utilisez-le si vous ne voulez pas éveiller les soupçons d'un entourage qui vous imagine studieusement occupé... ◀

(180307-04 – version française : Hervé Moreau)

Liens

- [1] Thermomètre à afficheurs géants Gottlieb® (Elektor, 04/2012) : www.elektormagazine.fr/110673
- [2] Bis repetita plaquette (Elektor, 07-08/2012) : www.elektormagazine.fr/120251
- [3] Bobines CA alimentées en CC : www.ecmweb.com/content/using-ac-coils-dc-power
- [4] Greffon pour cartes ESP8266 : <https://github.com/esp8266/Arduino>
- [5] Greffon Arduino esp8266fs : <https://github.com/esp8266/arduino-esp8266fs-plugin>
- [6] Paquet logiciel (Github) : <https://github.com/ElektorLabs/180307-PinballClock>
- [7] NtpClientLib : <https://github.com/gmag11/NtpClient/issues/60>
- [8] Archive logicielle du projet (labo d'Elektor) : www.elektormagazine.fr/180307-04
- [9] Vidéo de l'horloge Pinball (Elektor TV) : <https://youtu.be/IQL2Vk1Hj-o>



Un événement oublié ?

Vous organisez une conférence, un salon... ou bien vous participez à un séminaire ou tout autre événement qui aurait sa place ici, partagez cette information avec tous les lecteurs.

Envoyez-nous tous les détails à redaction@elektor.fr.

mars 2019

- ◇ **Maker Faire Lille**
01 au 03/03 - Lille
lille.makerfaire.com
- ◇ **Global Industrie**
05 au 08/03 - Lyon
www.global-industrie.com
- ◇ **MIDEST** (sous-traitance industrielle)
05 au 08/03 - Lyon
www.midest.com
- ◇ **Smart Industries** (industrie connectée)
05 au 08/03 - Lyon
www.smart-industries.fr
- ◇ **Rencontres de l'électronique imprimée**
06 au 07/03 - Paris
www.rencontreselectroniqueimprimee.com
- ◇ **JEC World 2019** (matériaux composites)
12 au 14/03 - Paris
www.jec-world.events

- ◇ **Semaine de la presse et des médias dans l'école**
18 au 23/03 - partout en France
www.clemi.fr

Elektor & MagPi
partenaire presse !

- ◇ **Semaine de l'Industrie**
18 au 24/03 - partout en France
www.entreprises.gouv.fr/semaine-industrie

- ◇ **APS Meetings** (fabrication additive, impression 3D)
19 au 20/03 - Lyon
www.apsmeetings.com/index.php/fr

- ◇ **eLearning expo - digital workplace - serious game i-expo & data intelligence forum - documentation**
19 au 21/03 - Paris
www.e-learning-expo.com & www.i-expo.net

- ◇ **Security & Safety meetings**
19 au 21/03 - Cannes
www.security-and-safety-meetings.com

- ◇ **Cloud Computing World Expo IoT World | IAI Congress | Blockchain Corner**
20 au 21/03 - Paris
www.cloudcomputing-world.com & www.iot-world.fr

- ◇ **MtoM & objets connectés - Embedded systems**
20 au 21/03 - Paris
www.embedded-mtom.com



Thème 2019
L'information sans frontières ?

Vous souhaitez publier votre montage dans le magazine ?

Rendez-vous sur la page du labo d'Elektor : www.elektormagazine.fr/labs pour y enregistrer votre projet.

Cliquez sur « Créer un projet ».

Connectez-vous (créez un compte gratuit si vous n'en avez pas encore).

Remplissez les différents champs du formulaire.

Votre proposition de montage sera examinée par l'ensemble des rédacteurs du magazine. Si votre projet est retenu pour sa publication dans le magazine, un rédacteur prendra contact avec vous pour vous accompagner dans la rédaction de l'article.

Labo d'Elektor : www.elektor-labs.com

Découvrir, créer et partager... l'électronique !

ÉTAT DU PROJET
☒ TOUT
☐ PROPOSITION (743)
☐ EN COURS (41)
☐ TERMINÉ (330)
☐ BROUILLON (1)

DATE DE PUBLICATION
☒ TOUT
☐ CETTE SEMAINE (6)
☐ CE MOIS-CI (18)
☐ CETTE ANNÉE (160)
☐ PLUS D'UN AN (1115)

bruits de labo



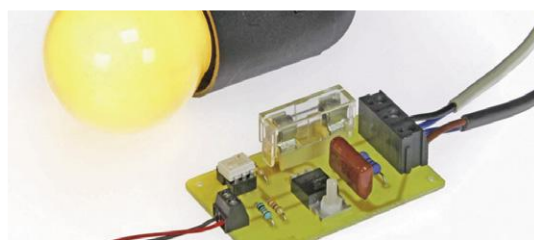
Les minuteries et les horloges sont des sujets populaires chez les électroniciens. L'affichage de l'heure ou d'un compteur est utile, mais permet aussi de donner libre cours à sa créativité.



Devenir riche et célèbre ? Publiez votre projet sur Elektor Labs

Convertisseur LED-ampoule sécurisé

Il est devenu aujourd'hui très courant de remplacer les ampoules incandescentes par des LED. Mais pourquoi ne pas faire l'inverse ? Grâce à ce convertisseur, c'est désormais possible. Il vous suffit de remplacer la LED par un photocoupleur, chargé d'allumer la lampe...



@ Elektor Labs: www.elektor-labs.com/1704

Horloge à LED avec affichage de la température extérieure

Se réveiller et connaître immédiatement la température extérieure ? C'est possible avec cette horloge à LED « maison ». Pour la réaliser, vous n'aurez besoin que d'un RPi comme serveur domotique, d'un ESP32 relié à un capteur DHT22 et d'un deuxième ESP32 destiné à commander un afficheur à LED. Complicé ? En le découvrant, vous pourriez changer d'avis...



@ Elektor Labs: www.elektor-labs.com/1605

Minuteur non programmable (sablier) sans fioritures

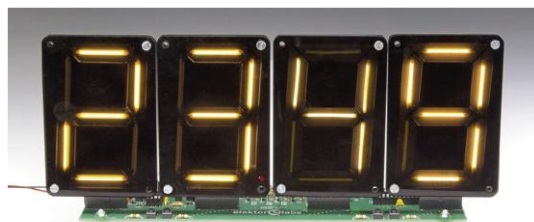
La prolifération des fonctions consiste à multiplier les capacités superflues dans un produit ou un programme : pipettes de sélection de couleurs et options de choix de polices de caractères dans les applications pour ordinateurs, paramètres configurables aussi obscurs les uns que les autres. Aucun risque que ce petit minuteur, dont le fonctionnement s'apparente à celui d'un sablier, ne souffre de ce mal. Quoique...



@ Elektor Labs: www.elektor-labs.com/478

Horloge, tableau d'affichage ou chronomètre avec des chiffres de taille XL

Besoin d'un grand écran pour afficher un score ou un compteur ? Construisez cet afficheur à quatre chiffres à 7 segments de 10 cm de haut, facile à lire à distance. Le projet comprend des modules LEDitron, c.-à-d. des afficheurs à 7 segments construits à partir de filaments de LED comme dans les lampes modernes. Et bien sûr, c'est aussi une horloge ! ◀



@ Elektor Labs: www.elektor-labs.com/1299

(180571-B-04 - version française : Pascal Godart)

projet SCCC (1)

processeur softcore et compilateur C à construire soi-même

Martin Oßmann

Parfois, dans un projet, il est nécessaire d'avoir à la fois un circuit FPGA et un processeur (CPU) classique, plus facile à programmer pour le déroulement de programmes complexes. La réalisation d'un processeur en FPGA peut constituer une solution. Il est possible de se procurer un tel processeur logique ou « softcore », mais ce n'est pas très bon marché. Dans ce projet, nous vous proposons de réaliser vous-même un CPU simple à partir d'un FPGA ; grâce au code source ouvert, vous pourrez l'adapter à vos besoins. C'est l'objet du compilateur C DIY (*Do It Yourself* = faites-le vous-même) qui peut, lui aussi, être étendu. Dans cette série d'articles, la théorie sera largement étayée par des exemples pratiques.



En travaillant avec des FPGA et des CPU standard, nous verrons que les FPGA sont très intéressants lorsqu'il y a un grand besoin de parallélisme et que le *Flow Of Control* (l'organigramme du

programme) est relativement simple dans les projets à FPGA. Lorsque l'organigramme du programme est complexe, les CPU classiques ont un net avantage.

Nous trouvons donc dans de nombreux projets un FPGA et un microcontrôleur. Il est également possible de réaliser en FPGA le CPU lui-même. Les fabricants de FPGA mettent souvent à disposition de tels « softcores » (cœurs logiques). Il s'agit la plupart du temps d'un logiciel maison qui s'avère parfois onéreux. C'est là que notre projet de projet de CPU logique *Small C Compiler CPU Project* (SCCCP) entre en jeu. Nous proposons un CPU simple qui, grâce à son code source ouvert, peut lui-même être adapté à vos propres besoins. Nous aurons ensuite besoin d'un langage pour le programmer. Pour traduire le programme en langage machine, nous utiliserons le *smallC*, un sous-ensemble du langage C qui travaille sur des entiers. Ce compilateur pourra aussi être étendu.

Carte MAX1000

- FPGA Intel MAX®10 10M08SAU169C8G
- Programmeur USB Programmer2 de Arrow
- 64 Mo de SDRAM (bus de données à 16 bits)
- 64 Mo de mémoire flash
- Oscillateur MEMS à 12 MHz
- Oscillateur MEMS optionnel
- 8 LED utilisateur rouges
- 2 LED d'état
- 2 boutons utilisateur
- Capteur d'accélération à 3 axes
- Connecteur enfichable PMOD à 12 contacts (à souder)
- Connecteur enfichable Arduino MKR (à souder)
- Connecteur enfichable JTAG (à souder)
- Connecteur enfichable d'E/S à 3 contacts (à souder)

Matériel

Côté matériel, nous avons choisi pour ce projet la carte MAX1000 d'Arrow (moins de 30 € [1]). Elle est équipée d'un FPGA MAX10 d'Intel/Altera, d'une interface de programmation

USB, de deux boutons utilisateur, de huit LED utilisateur et de différents connecteurs d'extension (voir boîte).

Pour nous familiariser avec la carte et le logiciel Quartus (d'Intel/Altera), nous commencerons par une expérience. Sous le lien [2], vous trouverez le guide d'utilisation *MAX1000 User Guide* qui indique comment installer le logiciel de développement Quartus. Ce dernier crée le fichier de programme qui, après téléchargement dans le FPGA, définit le comportement du module. Ce processus s'appelle aussi synthèse. Le langage de description du matériel utilisé pour ce projet s'appelle Verilog. Un article dans le précédent numéro donne une vue d'ensemble des principaux éléments de ce langage [3].

Expérience 1 : essai

L'expérience 1 est destinée à tester l'installation de Quartus. Pour cela, depuis la page de ce projet [4], téléchargez le logiciel compacté. Il contient différentes démos relatives au projet SCCCP. Décompactez le fichier sur un dossier local créé pour ce projet : `C:/SCCCP/`.

Il suffit maintenant d'ouvrir avec Quartus le fichier QPF (*Quartus Project File*) pour l'expérience 1 (`C:/SCCCP/experiment1/experiment1.qpf`). Ensuite, « synthétisez » le projet et chargez le fichier programme créé dans la carte MAX1000. Si les LED (utilisateur) s'allument alors cycliquement à tour de rôle (chenillard), c'est que tout a bien fonctionné.

Nous avons pour la première fois synthétisé un processeur personnel qui tourne sur le FPGA – le programme C était déjà précompilé dans le dossier de l'expérience 1 (voir **listage 1** pour le code source).

Pour aller plus loin, vous pouvez explorer le dossier de l'expérience 1. Il contient les fichiers `codemem.txt` et `datamem.txt` créés par le compilateur C de l'auteur. En téléchargeant les fichiers de programmation du FPGA, leur contenu est copié dans la mémoire du FPGA. Le contenu de `codemem.txt` définit le contenu de départ de la mémoire du programme du CPU (avec les instructions en langage machine) ; `datamem.txt` définit le contenu de départ de la mémoire des données de notre CPU (voir ci-dessous).

En examinant les fichiers, on peut observer que le code source figure dans chaque fichier, au début. Il sera ainsi toujours possible de retrouver ultérieurement le code source dans les fichiers. Dans le dossier de travail, figure en outre le fichier `sCCCPcpu1v01.v`. C'est notre CPU sous forme d'un fichier Verilog d'environ mille lignes de code.

La **figure 1** représente le diagramme du processus intégrant les différents outils et fichiers.

Le processus de synthèse prend environ 4 min avec l'ordinateur de l'auteur. Cela implique que le débogage est quelque peu pénible et prend beaucoup de temps. Nous verrons qu'il est possible d'améliorer le processus.

Genèse du projet

Concevoir un CPU (et le compilateur y afférent) n'est pas un projet réalisable en quelques heures. C'est pourquoi il vaut mieux adopter dès le départ une architecture correcte. Le projet « small C » de Ron Cain (« A Small C Compiler for the 8080's », Ron Cain, Dr. Dobb's Journal, avril-mai 1980, pp. 5-19) a directement inspiré ce SCCCP.

Ron Cain a développé un sous-ensemble C simple, mais vraiment puissant ; il produit du code pour le 8080 ou le Z80. Le code produit utilise un nombre restreint d'instructions 8080 dif-

Listage 1. Chenillard.

```
setLeds(int pattern){ // output pattern to LED
    pattern ;
#asm
    OUTA 5           // LEDs are attached to channel 5
#endasm
}

delay(int time){
    int k ;
    for(k=0 ; k<time ; k++){
    }
}

main(){
    int k ;
    while(1){
        for(k=0 ; k<8 ; k++){
            setLeds(1 << k) ;
            delay(500000) ;
        }
    }
}
```

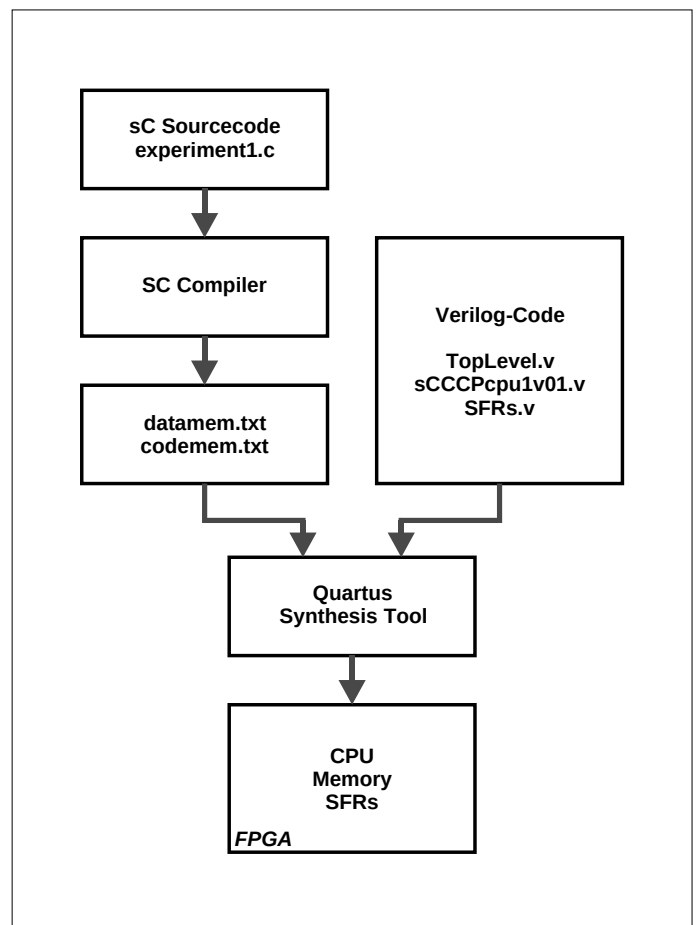


Figure 1. Le compilateur C transpose le programme C en langage machine et produit le fichier texte « `codemem.txt` ». Le code en octal est chargé dans la RAM du FPGA, pour être traité par le CPU logiciel synthétisé.

Listage 2. Chenillard avec gestion d'interruption.

```
#asm
RTCintVector      EQU    8
CPUflagsChannel   EQU    6
LEDsChannel       EQU    5

ORG      RTCintVector
PUSH     R0
PUSH     R1
CALL     RTCinterrupt
POP      R1
POP      R0
RET

#endasm

setCPUflags(int x) {
    x ;
#asm
    OUTA  CPUflagsChannel
#endasm
}

setLeds(int pattern){
    pattern ;
#asm
    OUTA  LEDsChannel
#endasm
}

RTCinterrupt(){
    setLeds(0xFF) ;
}

main(){
    int k ;
    setCPUflags(1) ;
    while(1){
        for(k=0 ; k<8 ; k++){
            setLeds(1 << k) ;
        }
    }
}
```

férentes. Un CPU à « empreinte aussi étroite » devrait pouvoir être transposé en CPU logiciel relativement facilement. Nous avons développé notre concept de CPU SCCCPC pour qu'il puisse exploiter les instructions machine produites lors de la compilation d'un sous-ensemble *smallC*. Ceux qui se souviennent encore de l'époque du 8080 verront rapidement combien notre CPU est proche du 8080.

Architecture du CPU

Notre CPU SCCCPC repose sur une architecture dite *Harvard modifiée*, c.-à-d. que les mémoires de programme et de données sont distinctes. Dans la configuration standard, nous avons assez de place pour loger 4096 instructions et 1024 mots de données. Le CPU dispose en outre d'un compteur de programme PC (*Programm Counter*) et d'un pointeur de pile SP

(*Stack Pointer*). Ces deux registres sont assez larges pour pouvoir adresser directement tous les mots (à 12 bits pour PC ; à 10 bits pour SP).

En outre il y a deux autres registres, de 32 bits de large : le registre primaire R0 ; la plupart des données y sont chargées tour à tour. Le registre secondaire R1 est utilisé lorsque deux opérandes interviennent (dans le *smallC* de Ron Cain, R0 correspond au registre HL et R1 au registre DE).

La mémoire des données fait aussi 32 bits de large, la mémoire de programme en fait 40. Une instruction comprend un mot de 32 bits pour les données et 8 bits pour coder l'instruction elle-même (*opcode*).

Nous pouvons désormais évoquer l'une des instructions du CPU. L'instruction

```
LDI      1234      ; LoadImmediate
```

écrit la valeur 1234 dans le registre R0.

Notre machine ne manipule que des mots de 32 bits et toutes les opérations sont exécutées sur 32 bits. Pour le travail avec des valeurs type *char*, seuls les 8 bits de poids faible sont utilisés. Cela peut paraître dispendieux, mais cela facilite considérablement le développement d'un compilateur.

Périphériques

Notre CPU peut aussi accéder à d'autres valeurs associées au matériel que nous avons conçu, elles font également 32 bits de large.

Par exemple, si on complète le CPU avec un compteur en temps réel RTC (*Real Time Counter*), nous voudrions lire et écrire la valeur RTC ainsi que définir une période. Dans ce cas, il y aurait deux valeurs (*RTCtimer*, *RTCperiod*) provenant du CPU, et une valeur que le CPU doit lire, à savoir le contenu du compteur RTC. Tous les paramètres des périphériques accessibles au CPU en lecture ou en écriture sont affectés d'un numéro, leur adresse SFR (*Special Function Register* = registre de fonction spéciale). Il y a une instruction pour lire :

```
INPA  nn
```

Cette instruction lit le registre *nn* de fonction spéciale et copie la valeur dans R0.

L'instruction

```
OUTA  nn
```

écrit le contenu de R0 dans le registre *nn* de fonction spéciale.

Au cours de notre démarche expérimentale, nous habillerons ce CPU très dépouillé avec d'autres fonctions : interface série, convertisseur A/N, MLI, RTC, interruptions. Le CPU aura accès aux SFR correspondants avec les instructions *INPA* et *OUTA*.

Par exemple, pour initialiser le RTC avec une valeur donnée, ou bien lire son contenu instantané, l'adresse sera toujours la même, 4 par ex et la période de l'horloge RTC sera accessible à l'adresse SFR 7. Ainsi pour régler le RTC sur 256 et définir une période de 500, nous écrivons en assembleur :

```
LDI      256      // charger la constante 256
                        // dans le registre R0
OUTA     4        // et écrire cette valeur à l'adresse
                        //SFR 4 , celle du compteur RTC
LDI      500      // charger la constante 500 dans le
                        //registre R0
OUTA     7        // et écrire cette valeur à l'adresse
                        //SFR 7 , celle de la période du
                        //compteur RTC
```


Les adresses SFR donnent aussi accès aux broches d'E/S (I/O) du FPGA, elles-mêmes reliées aux périphériques de la carte, comme les LED utilisateur.

Débogage de la réalisation du CPU

Si, pour déboguer un CPU en Verilog, nous voulons réaliser la synthèse complète du FPGA à chaque itération, il faudra énormément de temps. Pour rechercher les erreurs dans le code Verilog du CPU (et ses extensions), il est préférable d'utiliser une simulation Verilog. Pour cela nous utiliserons le simulateur gratuit Icarus Verilog [5]. Le lien [6] vous dirigera vers le manuel d'installation.

Avant d'attaquer l'expérience 2, il faut encore modifier les deux fichiers de commande (on dit aussi fichier *batch*) *c.bat* et *r.bat* du dossier de ladite expérience 2 afin d'adapter les appels vers le simulateur Icarus-Verilog. Pour l'auteur, les fichiers exécutables Icarus se trouvent dans le dossier *C:/iverilog/bin/*. C'est pourquoi le fichier *c.bat* ('c' pour « compile ») comprend la commande suivante :

```
c:/iverilog/bin/iverilog.exe -o Vobject TestBenchlv01.v
```

De même, le fichier de commande *r.bat* ('r' pour « run ») contient :

```
c:/iverilog/bin/vvp.exe Vobject
```

Il faut adapter le chemin du dossier d'installation d'Icarus dans les deux fichiers *c.bat* et *r.bat*.

L'environnement SCCC

Dans le dossier de l'expérience 1, nous avons un programme d'application déjà compilé. Il s'agissait du fichier *codemem.txt*, une image de la mémoire de programme. En général, les utilisateurs souhaitent écrire, compiler et charger dans le CPU leur propre programme C. À cet effet, l'auteur a créé un petit environnement de développement rédigé en langage de programmation Java. La ligne de commande ci-dessous permet d'appeler cet environnement :

```
java -jar c:/SCCCP/CompilerAsJar/sCCC.jar -s experiment2.c -i c:/SCCCP/experiment2/experiment2/
```

Avec les paramètres indiqués, la commande chargera l'environnement de développement avec le code source C de l'expérience *experiment2.c*.

Pour vous éviter les fautes de frappe, l'appel est rédigé sous forme d'un fichier de commandes stocké dans le dossier de l'expérience 2 et baptisé *c_compile.bat*.

La **figure 2** illustre l'interface graphique utilisateur (GUI) de cet environnement. Le bouton **MAKE** permet de compiler le programme C. Il y a en outre des options de débogage et de simulation ; nous y reviendrons dans la suite de l'article.

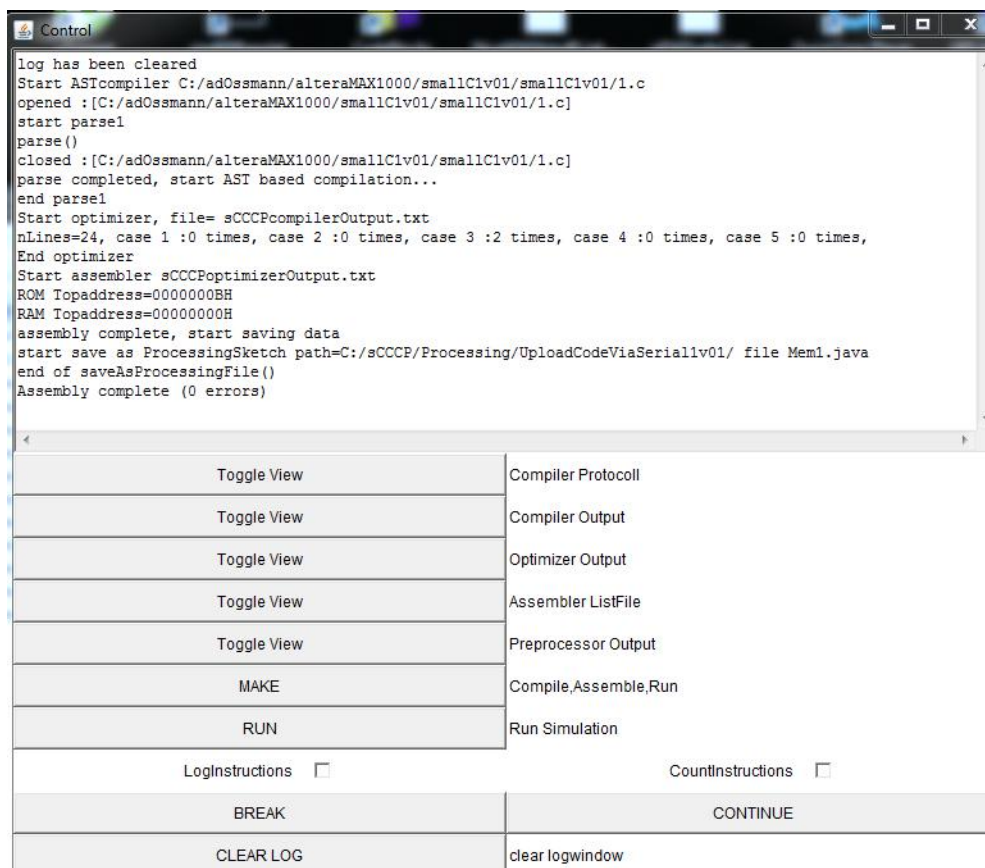


Figure 2. Le petit environnement de développement avec compilateur et simulateur est écrit en Java.

Listage 3. Informations de débogage.

```
VCD info: dumpfile test1.vcd opened for output.
sC reset
sC reset
-- set CPUflags 00000001H
-- out channel 00000005H data = 00000001H
-- out channel 00000005H data = 00000002H
-- out channel 00000005H data = 00000004H
-- out channel 00000005H data = 00000008H
-- out channel 00000005H data = 00000010H
//
Interrupt irq0 CPUflags=00000001H
***** interrupt ACK          selectedIRQ=0001H
      serviceCode at 0008H  simTimer=0000012eH save
      PC=00000028H

-- out channel 00000005H data = 000000ffH
-- out channel 00000005H data = 00000020H
-- out channel 00000005H data = 00000040H
-- out channel 00000005H data = 00000080H
-- out channel 00000005H data = 00000001H
-- out channel 00000005H data = 00000002H
-- out channel 00000005H data = 00000004H
```

Expérience 2 : simulation avec Icarus

Dans l'expérience 2, nous réutilisons le code du chenillard de l'expérience 1, mais nous avons ajouté une gestion des interruptions pour **Interrupt 1 (listage 2)**. Pour commencer, nous appelons l'environnement SCCP en exécutant le fichier `c_compile.bat` et `MAKE` crée le code cible (dans le dossier de travail `C:/SCCP/experiment/experiment2`).

Ensuite, l'exécution de `c.bat` traduit le code en format interne d'Icarus. L'exécution de `r.bat` démarre alors la simulation (cf. résultat dans le **listage 3**).

On voit que les données 01H, 02H, 04H, etc. sortent tour à tour, cela correspond bien à la séquence des LED à allumer. L'accès aux LED correspond à l'adresse SFR 5. Nous voyons ensuite comment l'interruption est traitée. Mais cela ne donne pas beaucoup plus d'information que lors de l'expérience 1. Cela vient du fait que dans le code Verilog de notre CPU, les commandes `define` correspondant au débogage sont restées sous forme de commentaires. Il faut supprimer les caractères `'//'` au début de `sCCCPcpu1v01.v` pour activer les `define`.

```
//`define VERBOSE2
//`define VERBOSE1
//`define VERBOSE3
//`define VERBOSE_INPA1
//`define VERBOSE_OUTAchrOut
```

Listage 4. Informations détaillées.

```
FETCH PC=0000H Time=000002 SP=03ffH OPC=2500000018H R0=00000000H R1=00000000H
decode: CALL    00000018H
pending write, store adr=000003feH val=00000001H

FETCH PC=0018H Time=000004 SP=03feH OPC=2affffffffH R0=00000000H R1=00000000H
decode: ADDSP   ffffffffH

FETCH PC=0019H Time=000006 SP=03fdH OPC=4000000001H R0=00000000H R1=00000000H
decode: PUSHI   00000001H
pending write, store adr=000003fcH val=00000001H

FETCH PC=001aH Time=000008 SP=03fcH OPC=250000000eH R0=00000001H R1=00000000H
decode: CALL    0000000eH
pending write, store adr=000003fbH val=00000001bH
```

Listage 5. Instructions NOP et LDI en Verilog.

```
if( sCstate==sCstateDecode1 ) begin
    if (sCopcode==sCopcNOP) begin
        sCstate<=sCstateFetch ;           // NOP instruction: nothing to do
    end                                   // goto fetch phase of next instruction
    else if (sCopcode==sCopcLDI) begin
        sCregR0 <= sCoperand1 ;           // Data fom instruction goes to R0
        sCstate<=sCstateFetch ;           // goto fetch of next instruction
    end
    else
        ...
end
```

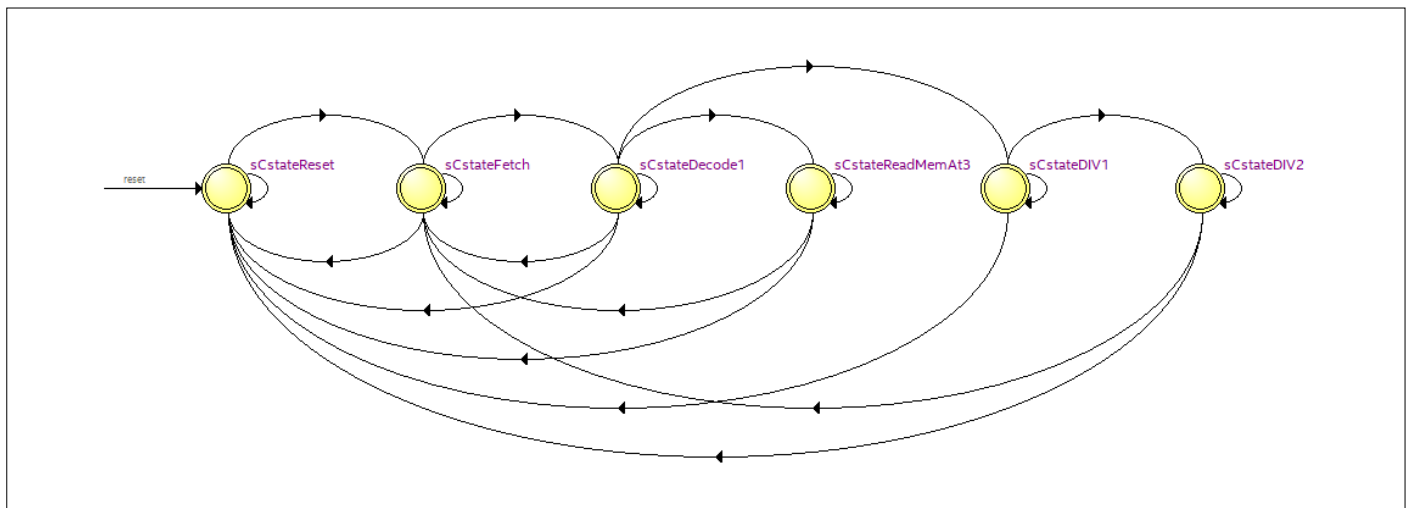



Figure 3. Diagramme d'états de l'automate à états finis pour le traitement des instructions machine.

Après une nouvelle compilation, nous obtenons un listage détaillé de ce qui se produit dans le CPU, voir le **listage 4** pour les quatre premières instructions.

Nous observons que pour chaque instruction, le contenu de chaque registre s'affiche, ainsi que le mnémonique de l'instruction. Les informations obtenues sont donc très détaillées. Nous croulons même sous les détails. Il faut être créatif et limiter les informations affichées à celles réellement nécessaires. Cependant, il est parfois nécessaire de disposer de tous ces éléments pour cerner un bogue. Malheureusement la simulation ne peut pas toujours révéler toutes les facettes du déroulement du programme, pour la bonne raison que dans la réalité, cela peut prendre beaucoup trop de ressources de le faire. Imaginons par ex. que nous analysons le comportement de la synthèse de signaux radio. Il faudrait simuler les signaux radio en respectant rigoureusement la chronologie, mais cela demande en général trop de ressources.

Mais si l'on veut identifier des erreurs par simulation du CPU, le simulateur Verilog Icarus est vraiment intéressant. Il a permis de découvrir diverses erreurs dans notre CPU et d'y remédier.

Programmation du CPU en Verilog

Nous allons maintenant expliquer à l'aide de quelques instructions choisies comment le CPU SCCCPC est réalisé en Verilog. Les instructions sont traitées par une machine à états finis (*finite state machine*) ou automate fini ; la **figure 3** montre le diagramme d'états produit par Quartus. La première phase

s'appelle `sCstateFetch` ; elle se résume à charger l'instruction et incrémenter de un le compteur PC (`sCregPC`). L'état suivant est `sCstateDecode1`. En Verilog, cela s'exprime comme suit :

```
if( sCstate==sCstateFetch ) begin
    sCinstruction <= sCcodeMem[sCregPC] ;
    sCregPC <= sCregPC + 1'b1 ;
    sCstate <= sCstateDecode1 ;
end
```

Dans l'état `sCstateDecode`, l'instruction est décodée ; l'exécution proprement dite débute alors. Pour certaines instructions, c'est pratiquement la fin de l'histoire, voir par exemple le code Verilog des instructions `NOP` et `LDI` dans le **listage 5** (l'état suivant est de nouveau `sCstateFetch` ce qui démarre l'instruction suivante).

Il y a naturellement des instructions plus complexes. Par ex. l'instruction `LDPUSH aa` place la valeur présente à l'adresse `aa` sur la pile et l'envoie dans R0. Dans l'état `sCstateDecode1`, la valeur de `aa` est placée dans la mémoire tampon `readAddress`. L'état suivant est `sCstateReadMemAt3`. La mémoire de données est alors lue et cette valeur est placée dans le registre R0. Simultanément, le processus d'écriture dans la pile est préparé, tandis que la variable `writeAddress` reçoit la valeur du pointeur de pile `sCregSP-1` ; cette valeur définit aussi la nouvelle valeur de `sCregSP`. La valeur lue est utilisée comme valeur à écrire `writeValue`. L'écriture peut maintenant être effectuée, le 4^e état

Liens

- [1] Carte MAX1000 : <https://shop.trenz-electronic.de/de/Produkte/Trenz-Electronic/MAX1000-Intel-MAX10/>
- [2] Manuel de la carte MAX1000 avec manuel d'installation Quartus : <https://bit.ly/2G96oS8>
- [3] « se lancer dans Verilog », Elektor 1-2/2019 : www.elektormagazine.fr/180562-04
- [4] Page du projet avec téléchargement du logiciel : www.elektormagazine.fr/180394-04
- [5] Simulateur Verilog Icarus : <http://iverilog.icarus.com>
- [6] Manuel d'installation d'Icarus : http://iverilog.wikia.com/wiki/Installation_Guide

Vue d'ensemble des instructions du CPU DIY.

instruction set (all operations 32 bit wide):

Name	cycles	meaning	explanation
NOP	2	no Operation	
IDIV	37	signed integer division	$R0 = R1 / R0$
IMOD	37	modulus operation	$R0 = R1 \% R0$ for positive operands
HALT	2	halts execution	
LDIND	3	load indirect	$R0 = \text{memory}[R0]$
LDI n	2	load immediate	$R0 = n$
ADDI n	2	add immediate	$R0 = R0 + n$
ADDSP n	2	add to stack pointer	$SP = SP + n$
JMP adr	2	jump	$PC = \text{adr}$
PCR0	2	load PC with R0	$PC = R0$
JFALSE adr	2	jump if $HL == 0$	if($HL == 0$) $PC = \text{adr}$
LD @adr	3	load	$R0 = \text{memory}[\text{adr}]$
PUSH R0	2+	save R0 on stack	$\text{memory}[--SP] = R0$
PUSH R1	2+	save R1 on stack	$\text{memory}[--SP] = R1$
CALL adr	2+	call subroutine	$\text{memory}[SP--] = PC$, $PC = \text{adr}$
POP R1	3	fetch R1 from stack	$R1 = \text{memory}[SP++]$
POP R0	3	fetch R0 from stack	$R0 = \text{memory}[SP++]$
XTR0	3	xchange top of stack and R0	$R0 = \text{memory}[SP]$, $\text{memory}[SP] = R0$
RET	3	return from subroutine or interrupt	$PC = \text{memory}[SP++]$
IADD	2	integer addition	$R0 = R1 + R0$
IADD SP	2	add stackpointer to R0	$R0 = R0 + SP$
IXOR	2	bitwise XOR	$R0 = R1 \wedge R0$
IOR	2	bitwise OR	$R0 = R1 \mid R0$
IAND	2	bitwise AND	$R0 = R1 \& R0$
ICOM	2	bitwise complement	$R0 = \sim R0$
INEG	2	signed negate	$R0 = -R0$
IMUL	2	signed multiply	$R0 = R1 * R0$
IASR	2	arithmetic shift right	$R0 = R1 \gg R0$
IASL	2	arithmetic shift left	$R0 = R1 \ll R0$
ST @adr	2+	store	$\text{memory}[\text{adr}] = R0$
STIND @R1	2+	store indirect	$\text{memory}[R1] = R0$
OUTA ch	2	output R0 to channel ch	channel $n = R0$
INPA ch	2	input R0 from channel ch	$R0 = \text{channel } n$
CMPEQ	2	compare equal	$R0 = (R1 == R0)$
CMPNE	2	compare not equal	$R0 = (R1 \neq R0)$
CMPLT	2	compare less than	$R0 = (R1 < R0)$
CMPLE	2	compare less equal	$R0 = (R1 \leq R0)$
CMPGT	2	compare greater than	$R0 = (R1 > R0)$
CMPGE	2	compare greater equal	$R0 = (R1 \geq R0)$
CMPULT	2	compare unsigned less than	$R0 = (R1 < R0)$
CMPULE	2	compare unsigned less equal	$R0 = (R1 \leq R0)$
CMPUGT	2	compare unsigned greater than	$R0 = (R1 > R0)$
CMPUGE	2	compare unsigned greater equal	$R0 = (R1 \geq R0)$
The following instructions were added later to increase efficiency (higher speed, smaller code). These instructions are not necessary. They are used by a simple peephole-optimizer. Switching the optimizer off eliminates usage of these instructions.			
PUSHI n	2+	push immediate value	$R0 = n$; $SP--$; $\text{memory}[SP] = R0$
TWOOP n	2	prepare two operands	$R1 = R0$; $R0 = n$;
LDPUSH adr	3+	load var and push	$R0 = \text{memory}[\text{adr}]$; $SP--$; $\text{memory}[SP] = R0$
ADDSPI n	2	load R0 by $SP + \text{offset}$	$R0 = SP + n$
GETLCL n	3	get local variable	$R0 = SP + n$; $R0 = \text{memory}[R0]$

Listage 6. Instruction LDPUSH en Verilog.

```
if( sCstate==sCstateDecode1 )
    readAddress <= sCoperand1 ;
    sCstate<=sCstateReadMemAt3 ;

if( sCstate== sCstateReadMemAt3 )
    sCregR0 <=sCdataMem[readAddress] ;
    writeAddress <= sCregSP-1'b1 ;
    writeValue    <= sCdataMem[readAddress] ;
    writePending <= 1 ;
    sCregSP <= sCregSP-1'b1 ;
    sCstate<=sCstateFetch ;

if( sCstate== sCstateFetch )
    if(writePending) begin
        sCdataMem[writeAddress[sCdataMemAddressWidth-1:0]]<=writeValue ;
    end
```

Listage 7. Instruction LDPUSH dans le simulateur Java.

```
instruction=ram[regPC] ;
opcode=(instruction>>32) & 0xFF ;    // 8 Bit opcode
operand1=instruction & 0xFFFFFFFF ; // 32 Bit immediate operand1

switch(opcode){
    case(ccNOP) :    { return ;}
    case(ccLDI) :    { regR0=operand1 ; return ;}
    case(.ccLDPUSH) : { regR0=readDataMemX(operand1) ;
                        regSP-- ;
                        writeDataMemX(regSP,regR0) ;
                        return ;}
}
```

y pourroit. Cependant, comme notre CPU a une architecture matérielle modifiée, nous pouvons en même temps accéder à la mémoire de données et à la mémoire de programme. L'accès en écriture qui reste encore à effectuer dans la mémoire de données va par conséquent être exécuté concomitamment à la phase **Fetch** suivante. L'automate le détecte grâce au bit **writePending**. Cela raccourcit le temps d'exécution d'un état (trois états au lieu de quatre).

Le **listage 6** déroule l'exécution de l'instruction **LDPUSH**.

Il faut maintenant produire un code semblable pour chaque instruction (voir l'encadré « **Vue d'ensemble des instructions** »). Il y a rien de très sorcier. Mais avant que tout fonctionne correctement, il a fallu plusieurs jours de programmation. Pour ce CPU, l'auteur a intégré dans l'environnement de déve-

loppement SCCC un petit simulateur pour PC. La version Java de l'exécution de l'instruction **LDPUSH** est nettement plus simple, voir le **listage 7**).

La seconde partie de l'article sera consacrée à la simulation du CPU dans l'expérience 3. ◀

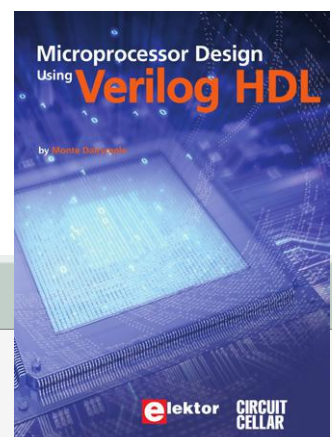
(180394-04 – version française : Yves Georges)



@ WWW.ELEKTOR.FR

→ Livre en anglais, 'Microprocessor Design Using Verilog HDL'
www.elektor.fr/microprocessor-design-using-verilog-hdl-e-book

→ Carte de développement FPGA MAX1000
www.elektor.fr/max1000



commande de ventilateur

en versions numérique (carte Arduino) et analogique

Jac Hettema (Pays-Bas)

Les montages qui dissipent beaucoup d'énergie doivent parfois être refroidis activement, c'est-à-dire à l'aide d'un ventilateur. Pour minimiser l'usure de ce dernier, et surtout son bruit, il convient d'adapter la vitesse de rotation à l'énergie à dissiper. Pour cela un dispositif de commande est nécessaire ; il peut être analogique, « à l'ancienne », ou plus moderne, avec une carte Arduino programmée.



Les circuits électroniques dégagent souvent de la chaleur. Lorsque la quantité à évacuer n'est pas trop élevée, un refroidissement passif est suffisant, par ex. à l'aide d'un simple radiateur. Plus la puissance augmente, plus le radiateur grossit, ce qui conduit parfois à un problème inextricable ; il faut alors recourir à un refroidissement actif, à l'aide d'un ventilateur. Nous vous montrons ci-après comment commander ce ventilateur.

Électronique analogique ou logiciel ?

Notre montage n'est pas très gourmand, et nous avons conçu deux variantes pour la commande : une avec de l'électronique analogique éprouvée, l'autre avec une carte Arduino (Pro Mini) et le programme ad hoc. Le circuit comprend un générateur de signal à modulation de largeur d'impulsion (MLI) pour la commande d'un transistor de puissance en série avec le moteur du

ventilateur. Le transistor de sortie et les composants connexes – inductance, diode et condensateur, font à juste titre penser à une alimentation à découpage et le comportement n'en est pas très différent.

Version analogique

Dans la version analogique (**fig. 1**), on utilise la jonction base-émetteur d'un transistor comme capteur de température (coefficient d'environ $-2,2 \text{ mV/K}$). La tension sur la base, fixée par le diviseur de tension R4/R5 est amplifiée par le transistor et comparée à la tension sur le curseur du potentiomètre R1 par l'ampli op U1A. La tension aux bornes de R1 est égale à 7,5 V, tension fixée par la diode zener D2, moins la tension aux bornes de la LED D1, de couleur verte ou bleue. Ce circuit présente quelques avantages.

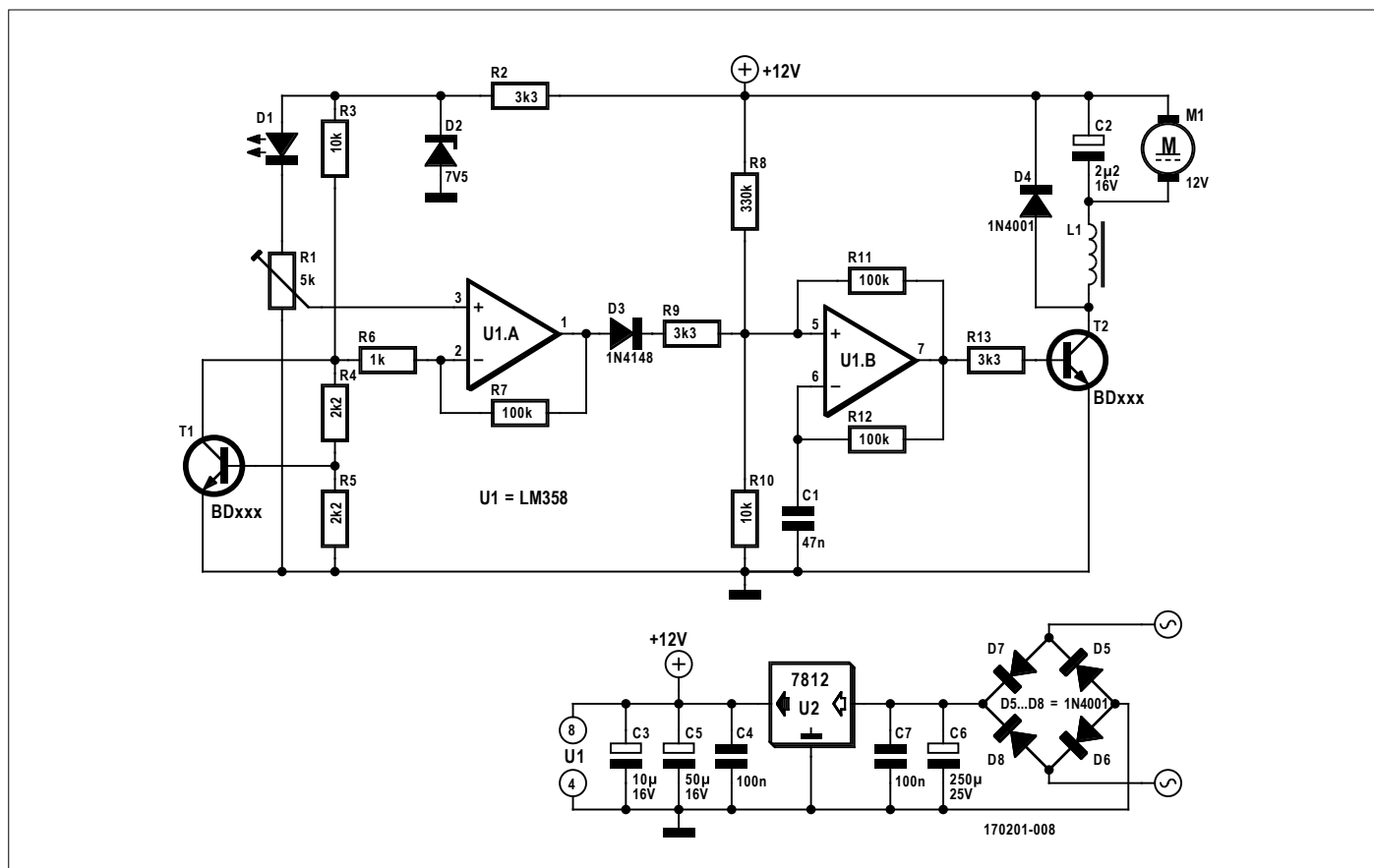


Figure 1. Dans la version analogique de la commande de ventilateur, le LM358 est utilisé en multivibrateur astable.

- Le coefficient de température de la zener est minimal pour la tension choisie de 7,5 V, et celui de la LED est négatif. Le coefficient de température de la tension aux bornes du potentiomètre R1 sera positif, c'est celui de R2, dont la température représente celle de l'environnement et est « mesurée ».
- Le transistor T1, dont le coefficient de température est négatif, mesure la température de l'objet à refroidir. La tension à la sortie d'UA1 dépendra donc de la différence de température entre l'objet cible et l'environnement. Elle sera plus élevée lors d'une augmentation de cette dernière et le ventilateur tournera alors plus vite.
- La LED indique également la mise sous tension du montage.

La tension du capteur de température attaque l'entrée inverseuse d'U1A, la moitié d'un LM358, et est amplifiée d'un facteur 100 (rapport R7/R6). Si un tel gain n'est pas nécessaire, la valeur de R7 peut être diminuée ; R7 peut aussi être remplacée par un potentiomètre de 100 kΩ monté en résistance variable, ce qui permettra de régler le gain entre 1 et 100.

Le second ampli op du LM358, U1B, est configuré en multivibrateur astable. La tension sur l'entrée non-inverseuse est fixée par le diviseur de tension R8/R10 ; cette entrée est attaquée par la sortie d'U1A via D3 et R9. La période du signal dépend de R12 et C1. Pour plus d'information sur le fonctionnement du circuit, voir par ex. [1].

Le rapport cyclique du multivibrateur est déterminé par la tension sur l'entrée non-inverseuse de l'ampli op. Si elle est égale à la moitié de la tension d'alimentation, il sera de 50% ; si elle est plus basse, la sortie restera plus longtemps à l'état haut avant basculement. On peut donc faire varier le rapport cyclique : il sera plus élevé avec une tension plus basse à l'entrée, et inversement.

TINA-TI [2], un logiciel de simulation gratuit basé sur Spice, permet d'évaluer facilement le circuit. La **figure 2** montre le schéma du circuit que nous avons utilisé pour ce faire ; il met

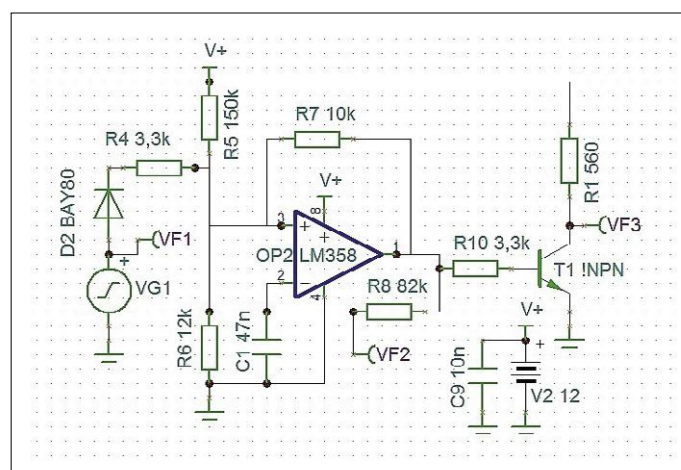


Figure 2. On peut facilement simuler le circuit analogique avec TINA-TI.

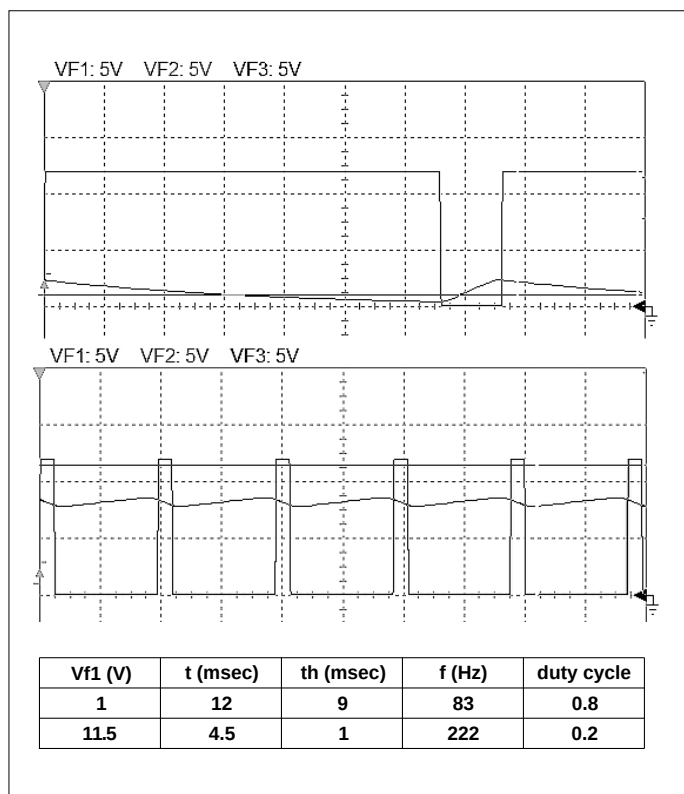


Figure 3. On voit clairement dans ces résultats de la simulation que tant le rapport cyclique que la fréquence varient.

bien en évidence l'influence de la tension d'entrée sur le rapport cyclique.

Le lecteur averti aura déjà noté qu'il n'y a pas que le rapport cyclique qui varie, mais aussi la fréquence du signal ! Cela n'a heureusement pas grande importance dans notre montage. Sur le schéma de la fig. 1, le transistor T1 est un NPN. On peut aussi utiliser un PNP, il suffit d'inverser émetteur et collecteur. Avec des transistors de la série BD, dont le collecteur est relié à la semelle métallique du boîtier, l'utilisation d'un PNP présente un autre avantage : il ne faut en principe pas d'isolation entre le transistor (le collecteur est à la masse) et l'objet dont on mesure la température. Le transfert thermique n'en sera donc que meilleur.

Nous n'avons pas mentionné de valeur pour L1, il faudra la déterminer expérimentalement. Vous pouvez toujours essayer avec ce que vous avez sous la main et vérifier le comportement

du circuit. Son rôle est de filtrer le signal MLI pour un fonctionnement plus stable du moteur du ventilateur, pour lequel nous n'avons pas non plus mentionné de type.

Version numérique

Pour la version à microcontrôleur, nous avons choisi une carte Arduino Pro Mini, parfaite pour notre projet et que l'on trouve à bas prix sur eBay ou d'autres sites. On peut aussi réduire la taille du circuit en utilisant une carte Digispark à base d'ATtiny85, mais ce n'était pas un critère déterminant pour ce projet. Le circuit devrait fonctionner avec toutes les cartes Arduino, mais le plus simple est d'en choisir une avec le même microcontrôleur ; un comparatif des divers modèles est disponible sur le site de l'Arduino [3].

Le circuit, dont le schéma est en figure 4, est plus simple que son homologue analogique. Pour le capteur de température, nous utilisons ici une diode d'usage général et non un transistor. Ce choix est dicté par l'utilisation de la référence de tension interne de l'Arduino, d'une valeur de 1,1 V ; avec un transistor comme capteur, la tension d'entrée serait supérieure à cette référence et impossible à traiter par le microcontrôleur. Le CA/N de l'Arduino fonctionne sur 10 bits, soit une résolution de 1,1 mV par pas. Avec la diode utilisée, ceci correspond à une résolution d'environ 0,5 K, ce qui est suffisamment précis dans cette application.

La référence de tension et les caractéristiques de la diode peuvent varier. Il est conseillé d'effectuer un étalonnage, en mesurant avec l'Arduino la tension aux bornes de la diode, à 0 et 100 °C. Une fois la tension à ces deux températures connue, on peut facilement en déduire la température correspondant aux tensions intermédiaires, et ainsi calculer la valeur que doit avoir le signal MLI en fonction de la température mesurée.

On pourrait bien entendu utiliser un thermomètre numérique DS18B20 de Maxim, mais on n'en a pas toujours un sous la main et c'est bon à savoir qu'on peut se débrouiller avec une simple diode. Le DS18B20 peut remplacer la diode sans autre modification ; il utilise le bus 1-Wire, ce qui ne présente aucune difficulté pour de courtes liaisons.

Le programme doit encore être amélioré. La lecture de température fonctionne correctement, mais la variation du signal MLI en fonction de la température n'a quant à elle pas été vraiment testée.

Commande du moteur et alimentation

Le ventilateur utilisé sur notre prototype est un modèle peu puissant alimenté sous 12 V. La commande de son moteur est confiée à un BD135 (un BD137 ou un BD139 conviennent aussi), dont le collecteur est relié au moteur via L1 et D4.

Un transistor plus puissant permettrait de commander de plus grosses charges, mais il faut alors veiller à ce que la carte Arduino ou le LM358 soient capables de fournir le courant de base nécessaire au bon fonctionnement du transistor.

L'alimentation des deux versions est confiée à un régulateur classique de type 7812. La tension alternative, d'une valeur maximale de 18 V_{eff} avec un radiateur approprié pour le régulateur, est redressée par le pont D5-D8 et filtrée par les condensateurs C7 et C8. C5 et C6 sont quant à eux nécessaires au fonctionnement stable du 7812. C'est le moteur du ventilateur qui consomme le plus. Il convient de prévoir une petite réserve – par ex. de 100 mA – en plus du courant nominal de

@ WWW.ELEKTOR.FR

→ JOY-iT Uno R3, réf. 18613
www.elektor.fr/joy-it-uno-r3

→ Kit Elektor « 35 capteurs pour Arduino », réf. 16843
www.elektor.fr/arduino-sensor-kit

→ Livre « Maîtrisez les microcontrôleurs à l'aide d'Arduino (3^e édition) »
www.elektor.fr/maîtrisez-les-microcontrôleurs-arduino

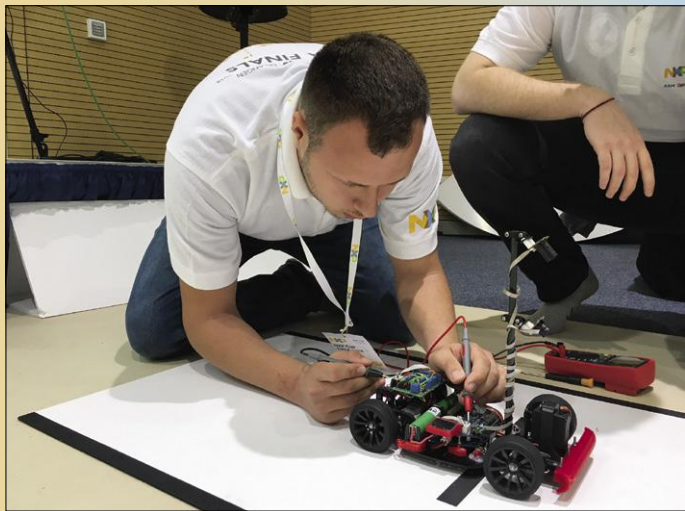
édition 2019 de la NXP Cup

plus intense et plus passionnante que jamais !

Clemens Valens (labo d'Elektor)

La NXP Cup est une compétition mondiale organisée chaque année, qui réunit équipes d'étudiants et clubs de robotique pour construire, programmer et piloter, sur circuit, un modèle réduit de voiture autonome. Et ce n'est pas qu'une affaire de « fun » : l'équipe gagnante s'envolera pour la Silicon Valley !

Le but de la NXP Cup est de construire et piloter en course des modèles réduits de voitures de niveau professionnel, mais sans les budgets d'une entreprise... Pour y parvenir, le matériel et le logiciel doivent être parfaitement conçus et optimisés dans les plus petits détails, et cet aspect est loin d'être simple. Avec la NXP Cup, les élèves ingénieurs ont là un environnement idéal pour se familiariser avec les compétences nécessaires



pour leur future carrière professionnelle. Comme l'a précisé Lars Reger, directeur technologique de NXP Automotive : « La technologie, le soutien des universités et notre extraordinaire écosystème de partenaires donnent aux étudiants l'opportunité d'acquérir les compétences essentielles du travail en équipe et de se familiariser avec les technologies indispensables pour construire des robots autonomes ».

Créer une voiture autonome compétitive, et capable de remporter la course, nécessite :

- Des compétences en programmation de logiciels embarqués et conception de circuits ;
- L'optimisation du matériel et du logiciel de commande des moteurs pour atteindre les vitesses maximales et assurer la précision de pilotage ;
- L'incorporation d'une caméra pour maintenir la voiture sur le circuit en suivant une ligne de guidage.



Pour aider les équipes engagées à atteindre leurs objectifs, les règles de l'édition 2019 de la NXP Cup ont été améliorées en donnant davantage de liberté aux concurrents :

- Capteurs illimités ;
- Liberté de choix et nombre illimité de processeurs NXP ;
- Défis accrus pour gagner des points supplémentaires : contre-la-montre, circuit en forme de 8, évitement d'obstacles et limitations de vitesse.

Outre l'assouplissement des règles et les nouveaux défis, le circuit lui-même a été amélioré et assure une meilleure adhérence, donc des vitesses supérieures. ◀

(180692-04 - version française : Pascal Godart)

Lien

[1] NXP Cup EMEA:

<https://community.nxp.com/groups/tfc-emea>



commande de moteur pas-à-pas avec circuit antirésonance

moteurs pas-à-pas : plus de rendement et moins de vibrations

René Merz (Magnetron.ch)

Les moteurs pas-à-pas sont devenus des composants incontournables de l'électromécanique moderne. On en trouve partout, des imprimantes 3D à tout un tas de machines en passant par les disques durs. Toutefois certaines de leurs caractéristiques ne sont pas idéales, comme les phénomènes de résonance et un rendement qu'on peut considérablement améliorer. Cet article montre comment il est possible de s'y prendre.

Des moteurs pas-à-pas, il y en a de toutes les tailles, avec des amplitudes de pas, des domaines de fréquences et des structures les plus divers, adaptés à chaque utilisation. En l'occurrence, ce sont les moteurs pas-à-pas hybrides et les modèles à pôles à griffes qui sont de plus en plus utilisés sur les imprimantes 3D pour les entraînements linéaires à arbre

fileté. Ils sont alors fréquemment sollicités en dehors des conditions optimales, c'est-à-dire aux hautes fréquences et à faible charge, domaine favorable à la manifestation de phénomènes perturbateurs tels que les résonances mécaniques. Pour obtenir un fonctionnement optimal d'un moteur pas-à-pas, c'est donc dès la conception de sa commande qu'il faut prendre ces phénomènes en compte et, autant que possible, y remédier. Cet article met en pratique une telle commande ; le montage repose sur un Raspberry Pi coiffé d'une carte d'extension (*Hardware on Top*, HAT).

Résonances

Dans le monde des moteurs pas-à-pas, on fait la différence entre les résonances propres et un domaine dit d'instabilité paramétrique. Pour des fréquences de commande de 1 Hz à 300 Hz, on peut atténuer les fréquences propres avec des dispositifs mécaniques (choix des entraînements, amortissements, etc.). Sur cette plage, le rendement d'un moteur pas-à-pas est typiquement limité à 10 %.

Au-delà de 300 Hz (ou pas par seconde), des résonances dites paramétriques apparaissent dans certaines bandes de fréquences, elles prennent la forme d'oscillations du rotor autour de sa position courante (**fig. 1**). À cause de ces résonances, ces oscillations peuvent atteindre de grandes amplitudes, au point de désynchroniser le rotor avec la fréquence de commande, lui faire perdre des pas, ou même le faire décrocher. Sans mesures correctives appropriées, il est impossible d'utiliser entièrement le domaine de puissance (**fig. 2**). La ligne en pointillés sur la **figure 3** montre que, sans mesures correctives, le couple et la fréquence de commande utiles se situent notablement en dessous des valeurs possibles.

Une solution

Pour mesurer les phénomènes à la résonance, il est possible de capter les signaux correspondant aux oscillations au moyen

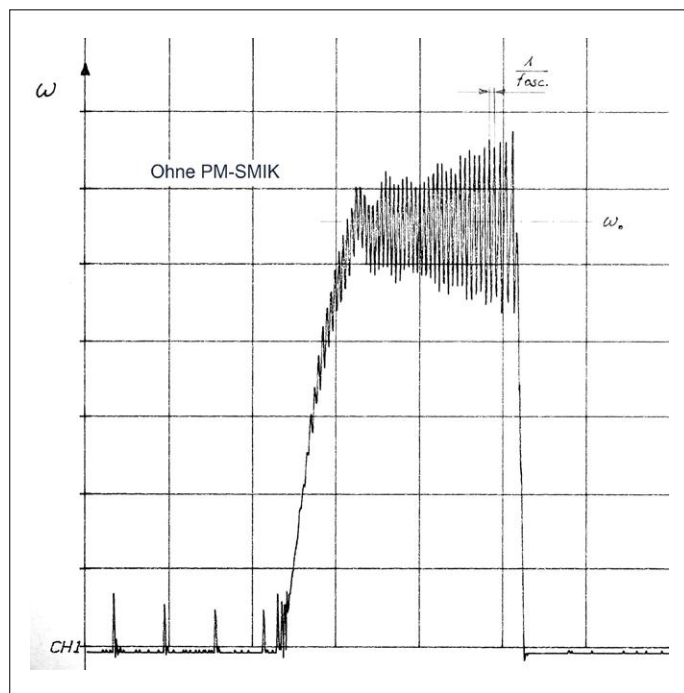


Figure 1. Oscillations typiques du rotor de petits moteurs pas-à-pas, mesurées en fréquence sur un shunt. X = 500 ms/div et Y = 200 mV/div. Sans dispositif d'antirésonance, on observe une fréquence d'oscillation du rotor de 34 Hz. En fin du transitoire, le moteur se bloque.

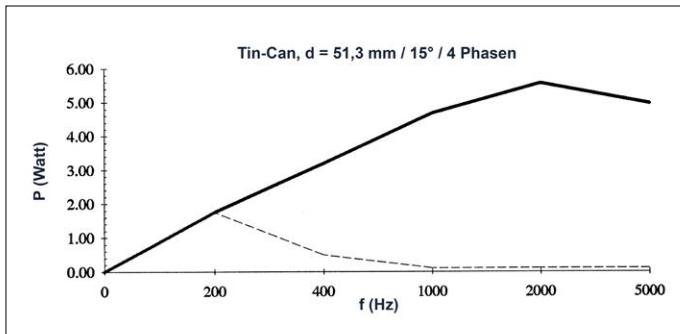


Figure 2. Puissance mécanique en W avec $I = 0,4$ A/phase et $ED = 30\%$. Sans circuit antirésonance, on obtient la courbe en pointillés. La puissance possible peut donc être triplée et la fréquence notablement augmentée.

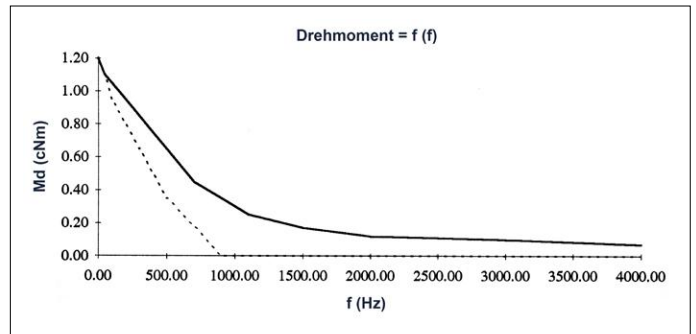


Figure 3. Couple en fonction de la fréquence. La courbe en trait continu montre l'amélioration considérable due au circuit antirésonance.

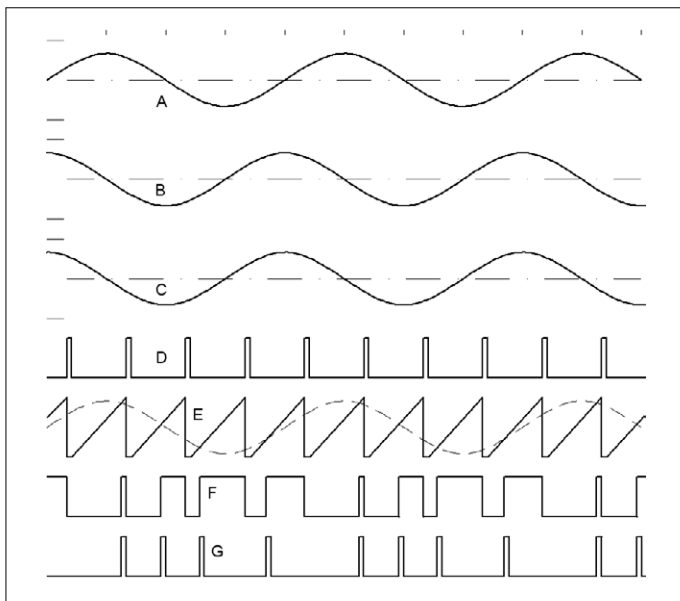


Figure 4. Relations de phases entre les signaux de pilotage du moteur : A : vitesse du rotor ω ; B : angle de charge rotor γ ; C : $U = f(IT)$, en phase avec le signal B ; D : Clock-In (f_0) du générateur de fréquence ; E : signal en dents de scie et U' après filtrage, décalé de 90° ; F : MLI ; G : Clock-PM, alimente le séquenceur.

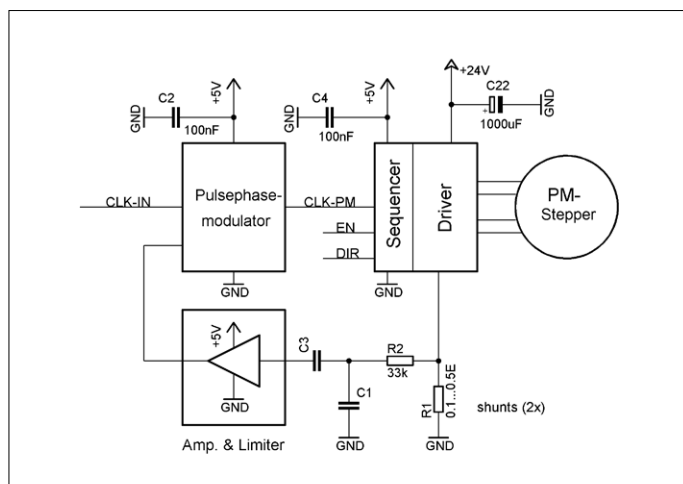


Figure 5. Principe du circuit antirésonance. Le signal d'un shunt est utilisé pour décaler la phase des impulsions de pilotage.

d'un shunt (chute de tension du courant de l'enroulement du rotor). La **figure 4** représente les correspondances de phase idéales entre le rotor et les signaux de commande.

La **figure 5** montre le principe d'une mesure corrective conçue en 1986 par Pierre Boillat [1], dont le brevet a depuis expiré. Sur les petits moteurs à pôles à griffes (*tin-can motors*), les oscillations se produisent typiquement dans le domaine de fréquences de 5 à 80 Hz. Après filtrage et amplification (en bas de la figure), le signal de shunt pilote un modulateur d'impulsions de phase. Son signal de sortie CLK-PM a une position de phase corrigée des impulsions de rotation ; il est appliqué sur l'entrée du séquenceur du pilote du moteur. On utilise un tel modulateur d'impulsions de phase lorsque le générateur d'impulsions (le plus souvent un microcontrôleur) n'est pas conçu pour produire une modulation de phase appropriée (ou n'en est pas capable).

De cette manière, il est possible d'atteindre des vitesses très élevées, jusqu'à 1500 pas/s, ce qui permet également d'améliorer le rendement des moteurs pas-à-pas. Autre avantage : il n'est pas nécessaire de respecter une courbe d'accélération particulière pour franchir les domaines d'instabilité. Enfin, à ne pas négliger, un moteur pas-à-pas ainsi stabilisé est aussi moins bruyant. Bon à savoir : ce procédé de stabilisation n'a rien à voir avec le pilotage MLI incorporé à beaucoup de circuits intégrés pilotes de moteurs pas-à-pas.

Circuit antirésonance

Pour la démonstration pratique du principe, l'auteur a conçu le circuit de la **figure 6**. Il s'agit d'un module d'extension (HAT) directement embrochable sur une carte Raspberry Pi au format A+. Le signal de cadencement fourni par un microcontrôleur externe est connecté à CLK-IN, c'est-à-dire à la borne 3 du bornier à vis X2. Après une mise en forme par l'un des inverseurs de IC7, le signal arrive sur le monostable IC3A, qui se déclenche aussi sur les fronts descendants. La sortie Q fournit des impulsions de largeur constante d'environ 30 μ s, lesquelles déchargent à travers le MOSFET T2 le condensateur C2 du convertisseur de signaux carrés en dents de scie réalisé avec IC1A et T1 à T3. La source de courant pilotée par T1 fait en sorte qu'après sa décharge à travers T2, le condensateur C2 soit rechargé linéairement à sa valeur maximale. Résultat : un signal en dents de scie très propre sur l'entrée positive (broche 3) du comparateur IC2A. Combiné au signal de modulation U' sur l'entrée négative (broche 2), on obtient

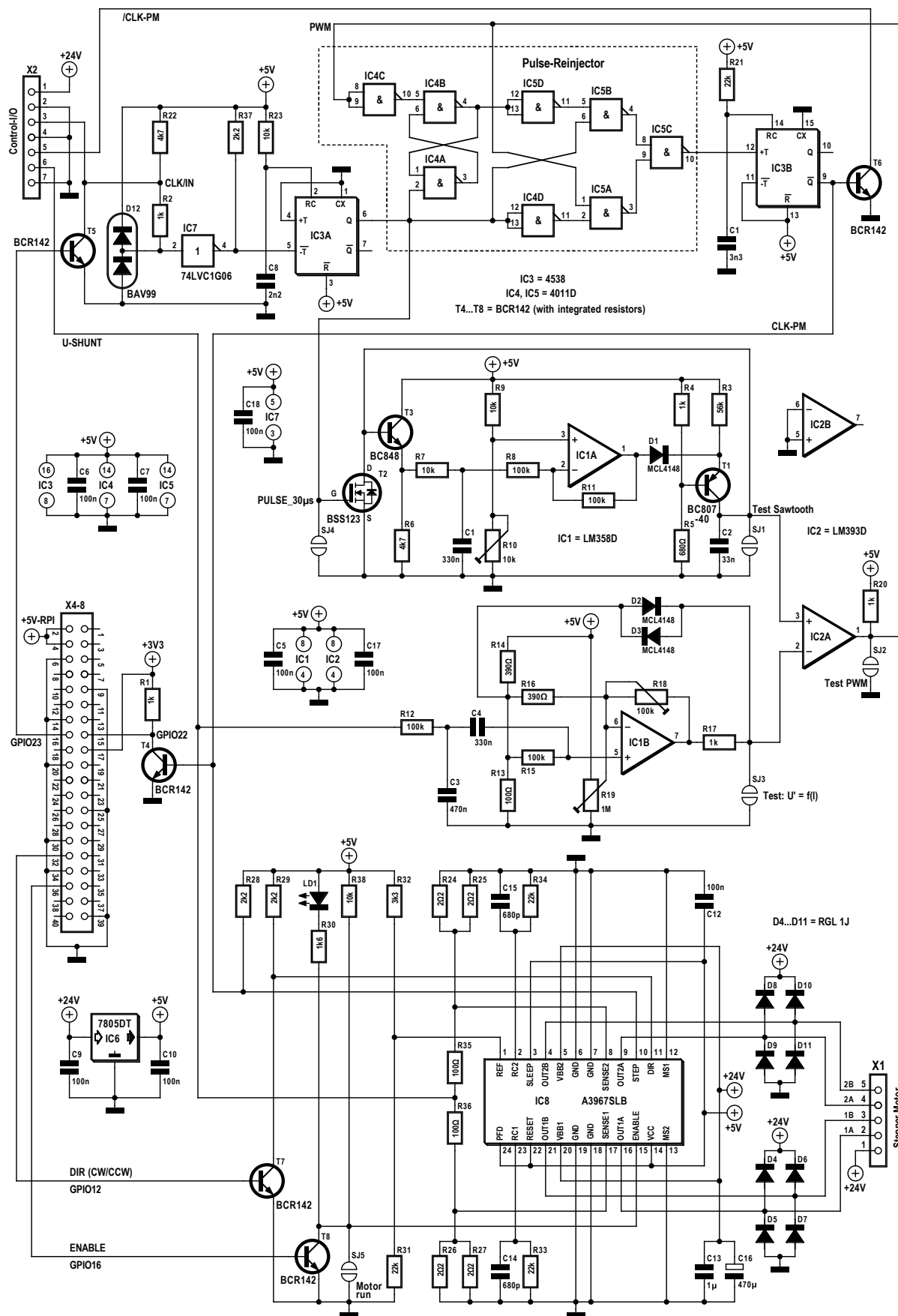


Figure 6. Le schéma concret du circuit antirésonance conçu sous forme d'une extension pour RPi (HAT).



Figure 7. Signaux du circuit antirésonance pour un angle de pas de 15° et 901,75 Hz. C1 : impulsions de 30 µs sur SJ4 ; C2 : dents de scie sur SJ1 ; C3 : signal MLI sur SJ2 ; C4 : signal du shunt, $U = f(IT)$ sur X2-6.

sur la sortie (broche 1) des impulsions avec la modulation de phase désirée (linéaire).

Pour un large domaine de fréquences du modulateur, la dent de scie doit avoir une amplitude indépendante de la fréquence. Pour cela, la source de courant T1 est pilotée par IC1A. Pour obtenir cette constance, on utilise la valeur moyenne de la dent de scie, en fait sa composante continue tamponnée par T3 et filtrée par R7/C1, comme contre-réaction. Le signal de commande est alors injecté à travers la diode D1 dans la source de courant.

Pour un balayage très rapide (par ex. quelques impulsions pour un passage de 0 Hz à 4 kHz en seulement 200 ms), ce réglage est trop lent à cause de la constante de temps R7/C1. L'am-

plitude de la dent de scie s'effondrerait alors, ce qui pourrait conduire à des impulsions manquantes sur le comparateur IC2A en fin de balayage. Voilà qui serait fatal au positionnement ! Pour éviter la disparition d'impulsions, on a introduit dans la boucle le circuit logique composé des portes NAND IC4 et IC5 (réinjecteur d'impulsions).

Le monostable IC3B est nécessaire, car seul le front montant du signal est modulé et certains séquenceurs-pilotes du commerce sont déclenchés sur un front descendant. Grâce au circuit à collecteur ouvert, un circuit en aval peut procéder à une adaptation de niveau au moyen d'une résistance de rappel. La tension de modulation U' est obtenue à partir du signal sur les résistances shunt R24 à R27 du pilote, filtré par un filtre passe-bande puis amplifié par IC1B. Dans de rares cas, le filtre passe-bande doit être adapté en fonction du moment d'inertie du rotor. Le gain de IC1B est réglable de 1 à 250 au moyen de R18. La pratique a montré que des gains de l'ordre de 50 sont le plus souvent suffisants. Les pastilles SJ1 à SJ4 sur le circuit imprimé servent de points de test pour un oscilloscope. Les signaux montrés sur l'oscillogramme de la **figure 7** peuvent être réglés au moyen des potentiomètres d'ajustage R10, R18 et R19. R19 sert au réglage du niveau continu à la sortie de IC1B, lequel devrait valoir sur SJ3 la moitié de la valeur de crête de la dent de scie sur SJ1 et se situer dans le domaine de 1 à 1,3 V.

Tout en fin de chaîne, on a IC8, le pilote intégré de moteur pas-à-pas d'Allegro [2]. Le circuit intégré A3967SLB contient un séquenceur, un pilote à découpage MLI pour le micro pas-à-pas et un étage de sortie à transistors bipolaires fournissant 0,75 A par phase. La tension de référence sur la broche 1 de IC8, fournie par le diviseur de tension R31/R32, peut être adaptée aux spécifications électriques du moteur pas-à-pas utilisé.

HAT RPi

L'extension de type HAT de la **figure 8** vient se loger directement sur une carte RPi de type A+ grâce à son connecteur 2×20 broches situé à l'emplacement approprié. Le RPi peut délivrer le signal de commande du moteur pour toutes sortes d'applications pratiques. Pour cela, les impulsions de cadencement avec un niveau de 3,3 V sont transmises, via sur la broche GPIO23 et le transistor T5, au circuit antirésonance avec ses niveaux de 5 V. Pour son adaptation de niveau, CLK-IN nécessite une résistance de rappel (R22). Les lignes de commande ENABLE (déverrouillage du moteur,

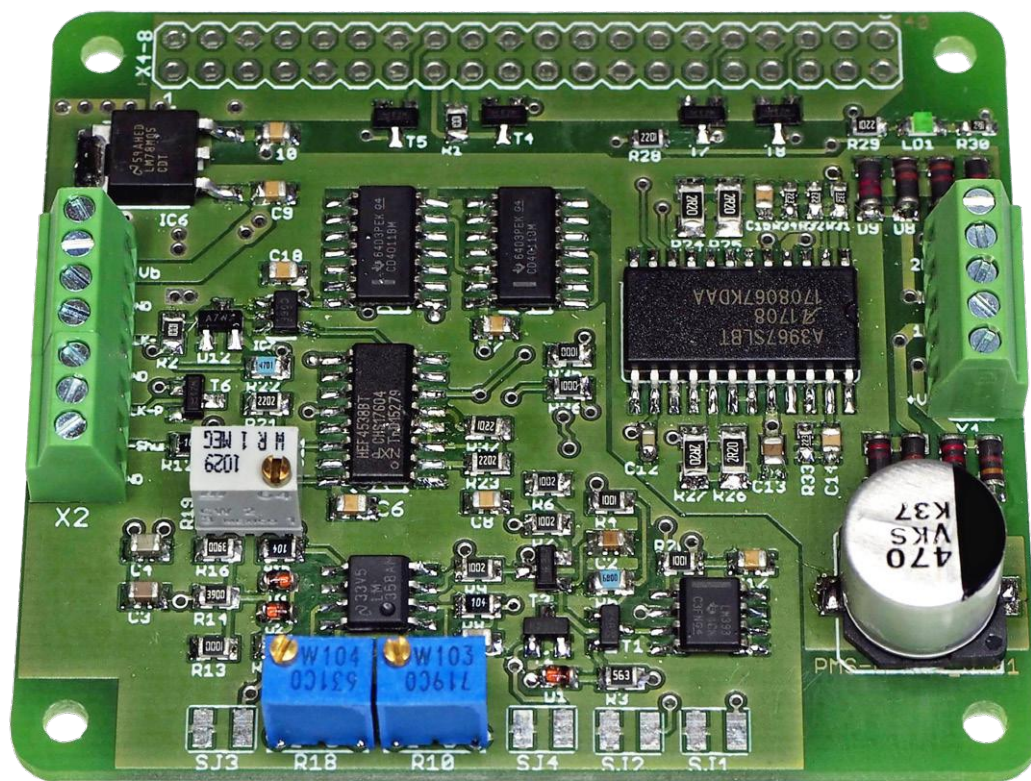


Figure 8. Vue du prototype du circuit de pilotage de moteur pas-à-pas à antirésonance, sous la forme d'un HAT pour le RPi modèle A+.

niveau bas actif) et CW/CCW (sens de rotation) sont adaptées de même. Bien entendu, si le module HAT est monté sur le RPi, il faut prévoir quatre entretoises et des vis pour la tenue mécanique. Les deux cartes doivent être alimentées avec des alimentations séparées, électriquement isolées. Pour le test, tous les petits moteurs pas-à-pas biphasés conviennent, peu importe qu'ils soient de type pôles à griffes ou hybride (NEMA17), bipolaires ou unipolaires, avec des résistances d'enroulement dans le domaine de 5 à 100 Ω /phase et avec des puissances de 1 à 15 W. Pour des résistances $\leq 5 \Omega$ /phase, le réglage du courant doit se faire avec de grandes précautions. Les connexions électriques au moteur ne doivent en aucun cas être interrompues en cours de fonctionnement.

Quoique le circuit IC8 soit conçu pour des moteurs bipolaires, on peut aussi connecter sur le bornier à vis X1 des moteurs unipolaires. Le milieu des enroulements est alors à brancher sur la borne X1-1 (+24 V).

Si vous voulez reproduire la carte HAT, vous trouverez sur la page web du projet sur le site Elektor [3] le montage et les fichiers de la carte. Le tableau ci-contre présente les **caractéristiques techniques**. Il ne vous reste plus qu'à programmer sur le RPi un logiciel de test ou un générateur d'impulsions approprié à votre application (ainsi que l'application elle-même). Mais ça, c'est votre tâche...

Tenants et aboutissants

En 2014, l'auteur a mis en service une imprimante 3D sur laquelle tous les arbres d'entraînement étaient animés par des moteurs pas-à-pas hybrides (4 fils, NEMA17), dont les enroulements sont connectés en série. Malheureusement, les fréquences choisies pour le contrôleur se situaient dans des domaines où l'enceinte de l'imprimante présentait de fortes résonances. On aurait pu remplacer les gros moteurs pas-à-pas hybrides par des modèles plus petits si le système de pilotage avait comporté un circuit antirésonance et une transmission adaptée. Beaucoup d'utilisateurs passent en mode micro-pas pour l'amortissement dans le domaine des résonances propres, mais aussi pour une meilleure résolution. Mais une meilleure résolution est loin de signifier une meilleure précision du positionnement.

Raisons à cela : défauts de géométrie du stator, dispersion dans la qualité des caractéristiques des tôles en fer doux du stator, déviations par rapport au courant de consigne (découpeur MLI, 1/8 à 1/16 de pas) pour des petits courants de phase.

Pour une même fréquence de pilotage sur le séquenceur et huit micro-pas, la vitesse de rotation de l'arbre du moteur est divisée par huit. On voit en effet avec la formule de l'énergie mécanique produite :

$$P_{\text{mech}} = M_d \times \omega$$

que la vitesse ω de l'arbre du moteur joue un rôle aussi important que le couple M_d . La charge devrait donc prendre sa position exacte dans le temps le plus court possible ! Il faudrait donc que le microcontrôleur intègre un ou plusieurs oscillateurs appropriés pour alléger sa charge de travail. Il est donc important que la puce comporte suffisamment de mémoire (NV-RAM ou flash) pour stocker des valeurs de fréquences, de nombres d'impulsions en rafale et de rampes. Pour les systèmes sur puce utilisés sur le RPi, il n'y a guère de problème.

Pour simplifier à l'avenir la mise en œuvre des moteurs pas-à-pas, les fabricants devraient indiquer précisément dans les feuilles de caractéristiques de leurs moteurs le rendement

Caractéristiques techniques

Tension d'alimentation	7 à 24 V
Fréquence de pilotage	1 mHz à 25 kHz
Décalage temporel CLK-In/Out	env. 5 ms à 1 kHz
Courant de phase max.	0,75 A/phase

(ou le rapport entre la puissance électrique consommée et la puissance mécanique restituée) en fonction de la fréquence de pilotage. Pour tirer un maximum des moteurs pas-à-pas, des algorithmes complexes à base de rampes d'accélération et de freinage sont indispensables, ce qui augmente bien entendu l'effort de programmation.

Il est vital d'éviter d'utiliser un moteur dans le domaine des résonances propres. Cela concerne aussi son montage : on doit veiller à ce que sa fixation comporte un dispositif d'amortissement efficace. Concernant l'usage d'un entraînement sans jeu (Harmonic Drive) ou d'un entraînement à courroie à dents, cela exige une évaluation du rapport prix/utilité. Il est possible qu'on dispose bientôt de pilotes à prix intéressant possédant un étage de sortie D qui fournirait directement des signaux de puissance filtrés sinus et cosinus pour les phases des moteurs. On saluerait aussi l'usage futur de matériaux pour le stator qui auraient de faibles pertes d'inversion magnétique et d'hystérésis, car des fréquences de découpage > 18 kHz sont déjà communes aujourd'hui. ◀

(180279-04 – version française : Helmut Müller)

L'auteur

De 1985 à 1990, René Merz a travaillé chez SAIA AG (aujourd'hui Johnson Electric International AG, Murten, Suisse) dans le domaine des bobines de magnétisation et dans le développement de nouveaux rotors à aimants permanents pour moteurs pas-à-pas et synchrones. Depuis 1990, René Merz s'occupe également de systèmes de pilotage de moteurs pas-à-pas. Depuis fin 1992, il est propriétaire de la société Magnetron Labs Merz.

Liens

- [1] 'Restraining the instability of a stepper motor', Pierre Boilat, brevets américains US4673855 et US4683409
- [2] Pilote à micro-pas A3967, Allegro : <https://bit.ly/2UoRnk1>
- [3] Page de l'article : www.elektormagazine.fr/180279-04



@ **WWW.ELEKTOR.FR**

→ 'Motors for Makers', livre en anglais
www.elektor.fr/motors-for-makers

→ Moteur photovoltaïque Mendocino à lévitation magnétique
www.elektor.fr/mendocino-motor-x-8

radar arrière pour vélo

un montage simple pour améliorer sa cyclo-sécurité

Laurent Labbe (Hong Kong)



Habitant en Asie depuis longtemps (Shenzhen en Chine et maintenant Hong Kong), j'enfourche régulièrement mon vélo soit pour aller travailler, soit pour des entraînements de triathlon. C'est pourquoi j'ai conçu ce montage autour d'un petit microcontrôleur à huit pattes ; il me permet d'éviter les torticolis à force de vouloir surveiller les véhicules autour de moi.

Quand j'allais à l'école en vélo, il y a (très) longtemps, et que je traversais des villes, j'avais un petit rétroviseur à droite et un à gauche pour apercevoir les voitures qui me doubleraient. Maintenant, ce genre d'accessoires est plus difficile à trouver. C'est pourquoi j'ai imaginé un système qui puisse m'alerter sans avoir à tourner la tête à gauche ou à droite, donc qui me permet de demeurer concentré sur la route devant moi. En Chine, les voitures et les bus vont dans tous les sens (souvent sans prévenir), il faut donc rester très attentif. C'est même pire à Hong Kong. Les routes sont très étroites, du coup les bus, taxis et minibus vous frôlent en permanence lors des dépassements.

Le système proposé ici est une sorte de radar, composé de deux capteurs à ultrasons, placés à 90°. Ils mesurent à intervalle

régulier la distance entre la selle du vélo (là où est installé le dispositif) et les véhicules s'approchant soit à l'arrière, soit à droite (en cas de conduite à gauche comme à Hong Kong). Il est possible d'ajouter un troisième capteur. Le prototype présenté ici (**fig. 1**) n'en a pas, mais le logiciel à télécharger [1] gère les trois capteurs. La position finale du module dépend du sens de circulation (à gauche ou à droite), donc du pays. Il faudra mettre le module plus vers la droite (cas de conduite à gauche) ou plus vers la gauche (cas de conduite à droite comme en Chine). Si le module est aligné avec l'axe du vélo, avec les deux capteurs à 45° de part et d'autre de l'axe, j'ai constaté que les voitures en stationnement (situées à gauche à Hong Kong et à droite en Chine) déclenchent des alarmes intempestives.

Comment alerter le cycliste ? Son regard ne doit pas quit-

ter la route ! J'ai choisi d'utiliser deux petits vibreurs, placés sous la selle, à gauche et à droite pour indiquer d'où vient le véhicule. Le nombre de vibrations (1, 2 ou 3) donne une idée de la distance. On pourrait aussi imaginer de mettre les deux vibreurs sur ou dans le guidon, à gauche et à droite en tirant des fils plus long. Les vibreurs sont des petits moteurs avec une masselotte, récupérés sur de vieux téléphones mobiles. Enfin il reste la question de l'alimentation. La consommation doit être aussi faible que possible, le montage se limite donc à une seule fonction : le radar. Il est alimenté par une batterie au lithium, rechargée via un circuit intégré spécialisé (ici le MAX1555). Je me suis passé de régulateur. La recharge par énergie solaire reste une option, peut-être sur la prochaine version.

Le circuit

Comme je garde l'idée de pouvoir mesurer la tension de la batterie, il faut un microcontrôleur avec une référence de tension intégrée, mais petit (à 8 broches) pour limiter la consommation. J'ai choisi le 12F1512 qui supporte une tension d'alimentation comprise entre 3 V et 5 V.

Pour les capteurs à ultrasons, il y a pléthore de choix. Cependant, peu d'entre eux peuvent être alimentés avec une tension

comprise entre 3,3 V et 5 V. Les modèles Y401 et US100 le permettent. Les capteurs sont configurés en mesure de distance par largeur de l'impulsion de retour. L'autre option (inutilisée ici) est de recueillir la distance sur une liaison de type série. Le schéma est très simple (**fig. 2**) : deux sorties GP1 et GP2 du μC activent les triggers des deux capteurs à ultrasons et une porte OU constituée de diodes permet de mesurer le temps jusqu'à l'impulsion de retour. Comme l'activation et la mesure de distance sont séquentielles dans le logiciel, la porte OU est suffisante, avec la résistance de rappel vers le bas.

La mesure est effectuée à l'aide du Timer1 qui compte en fonction de l'état de la broche 4 (GP3, **T1GVAL**). Le registre de ce *timer* est incrémenté quand la broche est à '1' ; il ne se passe rien quand la broche est à '0'. Le logiciel comporte une temporisation préliminaire (tant que le signal est à '0') puis mesure la durée pendant laquelle le signal est à '1' ou arrête la mesure si la durée limite réglée (**TIME_LIMIT_WAIT**) est atteinte (au cas où il n'y a pas d'obstacle rencontré par les capteurs à ultrasons). La porte OU à diodes est donc reliée à cette entrée qui déclenche le comptage par le Timer1. La durée est lue dans le registre de ce *timer* lorsque le signal de la porte OU retombe à '0'.



Figure 1. Prototype de l'auteur in situ, avec un capteur à droite et un à l'arrière. On distingue les fils de raccordement des vibreurs de part et d'autre de l'armature de la selle.

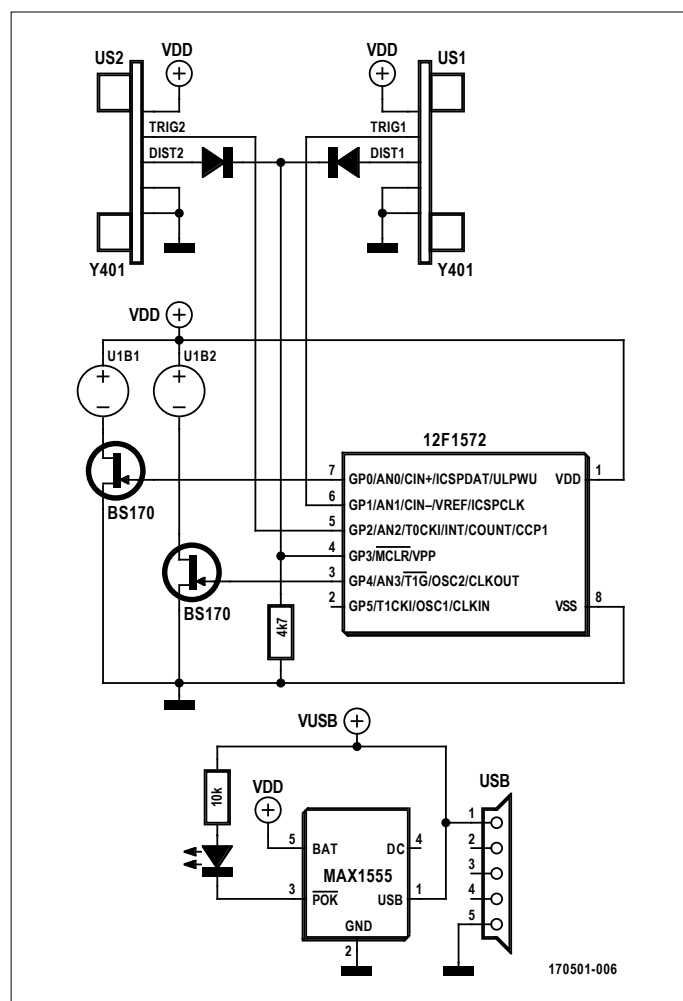


Figure 2. Schéma du radar, simple et efficace.

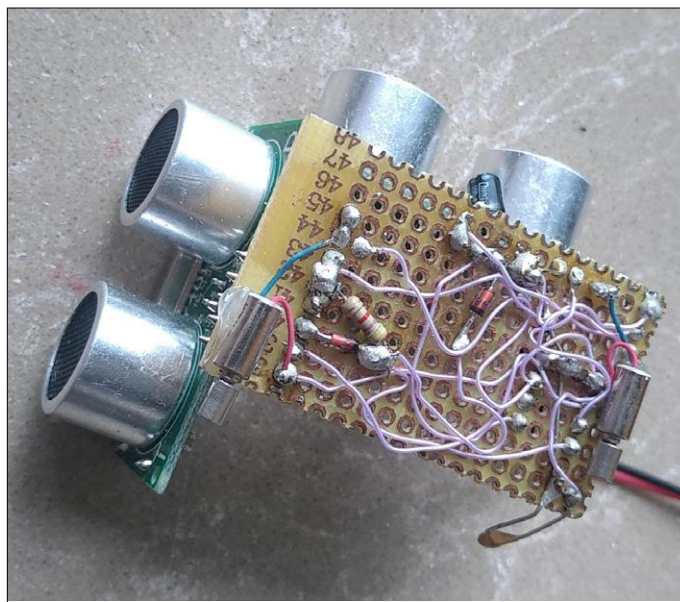
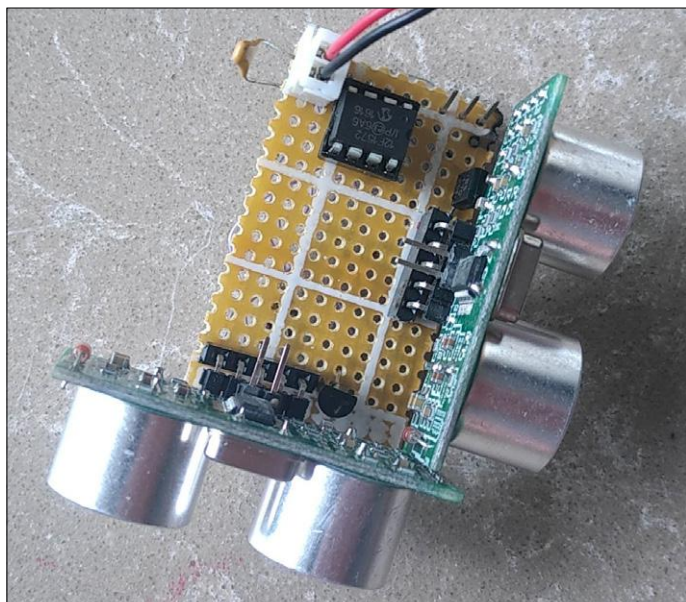


Figure 3. Câblage « double face » maison sur une plaque à trous.

La distance (en cm) est calculée avec la formule suivante :

$$d = t \text{ (durée du signal 'haut' en } \mu\text{s)} * 34 / 200$$

Les vibreurs sont activés via deux transistors FET de type BS170. Comme indiqué précédemment, le module peut être alimenté avec n'importe quelle batterie au lithium ; un circuit intégré MAX1555 sert à la recharge via une prise USB.

Le logiciel

Le logiciel est très simple. Toutes les deux secondes, il active le capteur de gauche, mesure la distance, active le vibreur de gauche en fonction de l'obstacle détecté puis fait la même opération avec le capteur de droite.

Si un troisième capteur est câblé pour le centre (sortie GP5), le logiciel prévoit de déclencher l'alerte sur les deux vibreurs en même temps.

Les distances d'alertes sont prédéfinies dans le logiciel.

Le logiciel a été développé avec la dernière version de MikroC, la seule à prendre en compte le µC 12F1512. La version gratuite suffit, car la taille du programme final est réduite. J'ai utilisé un PICKit3 pour la programmation.

Montage sur le vélo

Cela n'est pas le plus simple... Il faut installer la carte du radar et les vibreurs :

- Le radar est placé sous la selle. Je l'ai fixé avec des colliers de serrage rapide. Il faut veiller à ce qu'il soit bien horizontal afin de ne pas « capter » la route...
- Comme expliqué ci-avant, il faut tourner sensiblement le radar en fonction du sens de circulation dans le pays.
- Pour les vibreurs, la position la plus simple est sous la selle.

Les deux vibreurs sont entourés de gaine thermo et fixés avec un collier de serrage sur les deux tiges métalliques de l'armature de la selle. Plusieurs tests m'ont permis de constater que c'est là qu'il y a le moins de mouvement. Sur le devant de la selle, avec le mouvement des jambes, on ne ressent pas les vibrations. Si ajouter des fils sur le vélo ne vous pose pas de problème, la solution la plus efficace est sur le guidon...

Ce petit « dispositif de sécurité » est perfectible (capteurs, alimentation...). À vous de jouer ! ◀

(170501-01)



Lien

- [1] Page de l'article : www.elektormagazine.fr/170501-01



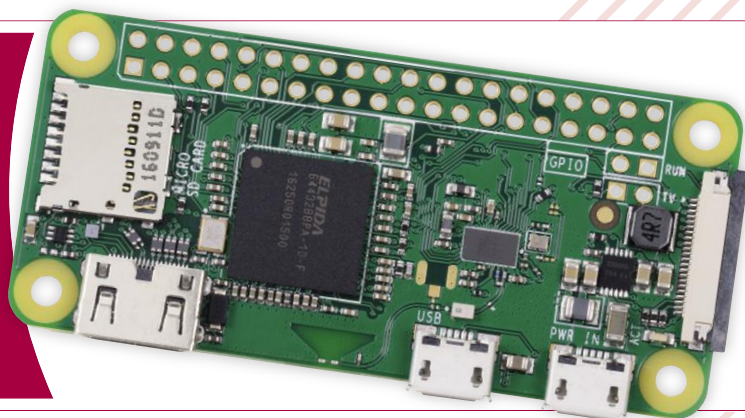
@ WWW.ELEKTOR.FR

→ Livre « Les microcontrôleurs PIC pour les débutants », Rémy Mallard, ISBN 9782866611931, réf. 15959

www.elektor.fr/les-microcontroleurs-pic-pour-les-debutants

ABONNEZ-VOUS ET RECEVEZ

RPI ZERO W GRATUIT



Souscrivez dès maintenant un abonnement d'un an au magazine MagPi, nous vous offrons :

- Six numéros du magazine MagPi
- Une carte Raspberry Pi Zero W
- Un boîtier avec trois couvercles différents
- Un connecteur pour module de caméra
- Un câble HDMI/mini-HDMI et un câble micro-USB/USB OTG

**SEULEMENT
54,95 €
PAR AN
(6 NUMÉROS)**

TOUS LES 2 MOIS, LES DERNIÈRES NOUVELLES DU RASPBERRY PI ET LES MEILLEURS PROJETS !



Vos avantages :

- Prix au numéro réduit
- Chaque numéro directement dans votre boîte aux lettres
- Tous les numéros disponibles sous forme numérique (PDF)
- Cadeau de bienvenue d'une valeur de 22,95 €
- Découverte de chaque nouveau numéro avant sa sortie en kiosque

ABONNEZ-VOUS : WWW.MAGPI.FR



trucs et astuces

les lecteurs écrivent aux lecteurs

Encore des solutions futées qui facilitent la vie des électroniciens.

Comment remédier élégamment aux dégâts causés par les fuites des piles alcalines

Peter Krueger

Alors voilà : on sort du tiroir un appareil parce qu'on en a de nouveau besoin, et rien ne marche plus avec la lampe de poche, le poste de radio ou la brosse à dents électrique de voyage. Piles vides ? Pire : elles ont fui ! Si vous cherchez désespérément le moyen de rétablir un fonctionnement normal des contacts des piles, vous ne manquerez pas d'apprécier le procédé de remise en état présenté ici.

Se contenter de jurer ou de gratter ne sert à rien ou du moins à pas grand-chose. Les piles mangano-alcalines sont primaires non seulement par leur structure (à savoir non rechargeable), mais également par leur capacité de nuisance. Elles ont la propriété scandaleuse et toxique d'évacuer en fin de décharge la pression interne croissante, due aux processus électrochimiques à travers une membrane effrontément nommée « soupape de sécurité », mais qui se perce régulièrement, ce qui provoque l'écoulement de l'électrolyte (un liquide alcalin corrosif) par deux trous de dégazage du côté du pôle négatif de la pile [1]. Les fabricants de piles ne semblent pas être capables ou disposés à produire des piles étanches ! Si cela vous est déjà arrivé et a mis votre appareil hors service, vous serez certainement de mon avis :

*voilà quelque chose qui n'aurait jamais dû être commercialisé !
Mais rien ne sert de se lamenter, aide-toi et misère tu vaincras !*

Situation de départ

La **figure 1** montre dans quel état peut se trouver un feu arrière de vélo après deux ans de stockage et d'oubli. Si ce qui s'est passé vous intéresse : l'hydroxyde de potassium des piles réagit avec le gaz carbonique (CO_2) de l'air pour donner du carbonate de potassium (K_2CO_3) et former des dépôts blancs, cristallins et hygroscopiques. Le résultat est une corrosion de tout ce qui a rapport avec le courant électrique. Cela touche en première ligne les ressorts de contact, surtout leur couche galvanique, mais les pistes conductrices d'un circuit imprimé peuvent aussi bien être rongées. De plus, le carbonate sec est un très mauvais conducteur électrique. Le changement de piles et un soufflage occasionnel n'y suffisent généralement pas.

Tandis que des pistes conductrices abîmées ne posent habituellement pas de problème aux électroniciens (après tout, à quoi servent les petits fils et la soudure ?), des essais de toutes sortes pour nettoyer les contacts, par un procédé mécanique ou autre,



Fig. 1. Un feu arrière de vélo oublié pendant deux ans : les piles alcalines ont fui et les contacts ont été sévèrement rongés.



Fig. 2. Le porte-piles ainsi que les contacts et le circuit imprimé dans le bain rénovant de produit à vaisselle.

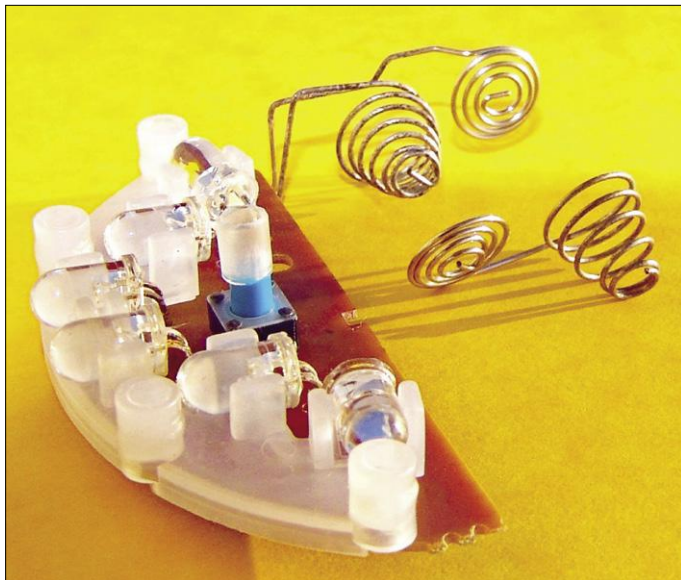


Fig. 3. À peine croyable : les contacts ont l'air comme neufs !

finissent par vous frustrer et par rendre vos appareils peu fiables. De même, les essais de nettoyage au vinaigre ou autre acide des contacts à ressort en acier chromé, nickelé ou même doré provoquent une corrosion supplémentaire qui dégrade encore plus la situation. Oubliez tout cela ! Car il existe de meilleures méthodes.

Suppression des dégâts

En regardant une émission de télévision sur les acides et les bases, j'ai eu l'idée de dissoudre les produits alcalins de fuite de pile dans un liquide alcalin du genre produit à vaisselle ordinaire. La restauration d'appareils endommagés s'avéra en effet ridiculement simple, car le produit à vaisselle fonctionna à merveille. Si ça vous laisse sceptique, essayez, vous verrez bien !

Il faut utiliser un produit contenant quelques pour cent d'un composant tensio-actif. Pour moi, même un savon liquide pour les mains bon marché, appelé « Aloe Vera » a donné de bons résultats. Pour éviter des dégâts supplémentaires dus à la corrosion, le produit doit avoir un pH pas trop élevé, ce qu'on peut tester en le goûtant : il doit avoir un goût neutre et pas acide.

On trempe les contacts démontés ou le porte-piles complet (fig. 2) dans un récipient rempli de liquide à vaisselle ou de rinçage. Concentré, bien entendu, car il contient déjà bien assez d'eau. Après un jour de trempage environ, les dépôts ont été dissous et le liquide s'est épaissi au cours de l'opération. Pour finir, il faut rincer à l'eau, laisser sécher, et voilà : c'est quasiment neuf ! La preuve : la photo de la figure 3. Il n'y a plus qu'à remonter et voilà un feu arrière de vélo resplendissant (fig. 4), prêt à recevoir un nouveau jeu de piles ;-).

Compléments

Prévenir vaut mieux que guérir, chaque fois que les exigences de tension le permettent, je remplace désormais les piles alcalines de base par des accumulateurs NiMH. Je choisis ceux du type « prêt à l'emploi », c'est-à-dire à autodécharge lente ou faible (aussi connus sous la désignation « eneloop »). Ils ne fuient pas et ne se plaignent habituellement pas d'un an d'oubli. Ou alors, on les retrouve défectueux, voilà tout. Ce sort est plus rapide,

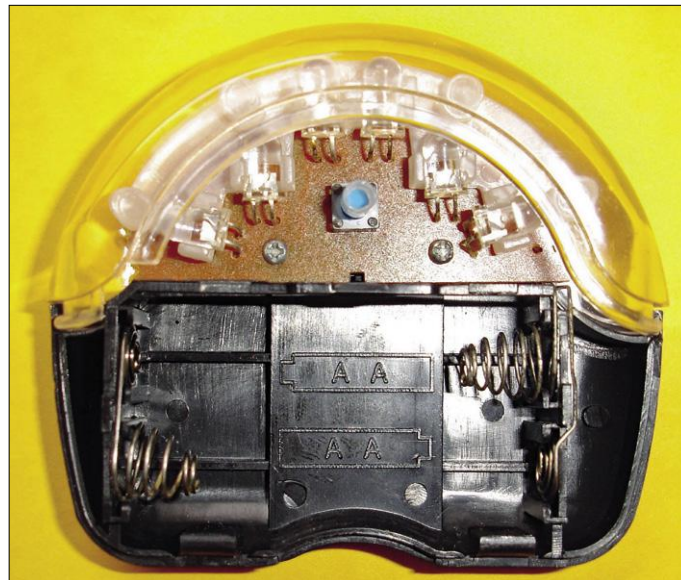


Fig. 4. Le feu arrière de vélo attend ses nouvelles piles. Ou mieux : si on y mettait des batteries ?

L'auteur

Peter Krueger est ingénieur de développement en électronique avec, jusqu'à présent, 62 années d'expérience professionnelle dans le domaine de l'électronique numérique et analogique ainsi que dans le développement de matériel et de logiciel.

donc plus fréquent, chez les accumulateurs NiMH ordinaires.

Il peut arriver que les piles soient collées aux contacts, de sorte qu'on ne puisse plus les sortir du porte-piles sans dommage. Solution : plonger l'ensemble, y compris les piles, dans le bain pendant une journée.

Si les contacts sont soudés directement sur le circuit imprimé, il est préférable de les dessouder au préalable. Quand c'était impraticable, il m'est arrivé de plonger l'ensemble du circuit et des composants électroniques dans le bain sans dommage. Les circuits imprimés en ressortaient particulièrement propres ! Il faut juste prendre garde à ce que les potentiomètres, les interrupteurs ou les touches ne soient pas submergés. Leurs cavités ne sont pas si faciles à vider de restes de détergent et à sécher. À vous d'essayer. Sans risque, pas de plaisir ! ◀

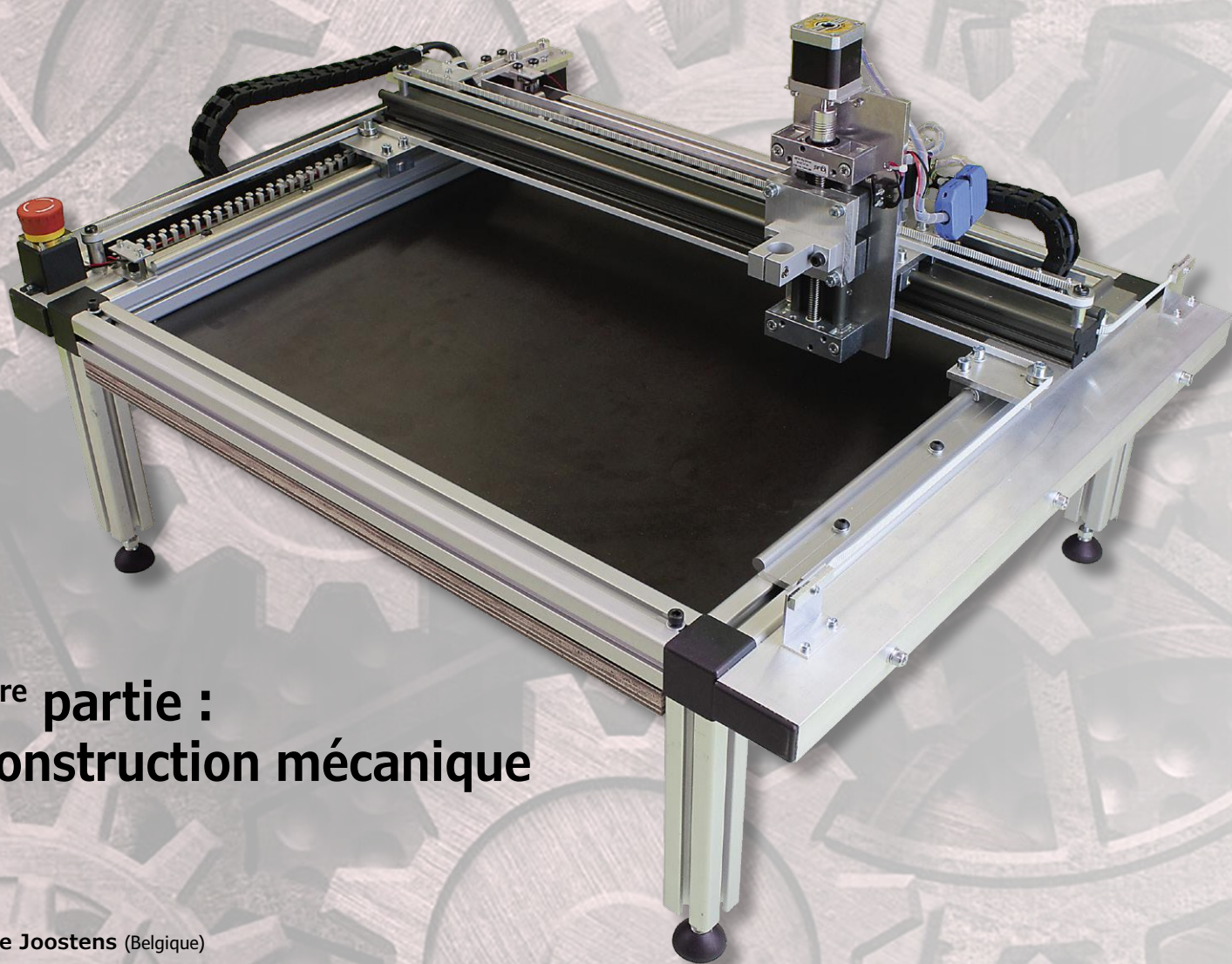
(190088-01 - 160429 -
version française : Helmut Müller)

Liens

- [1] https://fr.wikipedia.org/wiki/Pile_alcaline &
https://fr.wikipedia.org/wiki/Pile_électrique

machine-outil à commande numérique Wolverine

portique robotisé multifonctionnel



1^{re} partie : construction mécanique

Ilse Joostens (Belgique)

De nombreux électroniciens amateurs, chevronnés ou non, et faiseurs ont souvent des soucis avec les aspects mécaniques de leur projet. Dessiner un circuit imprimé ou écrire un programme pour un microcontrôleur n'effraie pas la plupart d'entre nous, mais dès qu'il s'agit de la construction de l'appareil lui-même, l'enthousiasme retombe d'un cran. Et on ne parle même pas de l'usinage de quelques pièces simples, comme un pignon ou un couvercle... L'avènement des imprimantes 3D et d'autres machines-outils à des prix abordables, ainsi que l'apparition de nombreux fournisseurs de services, facilitent la conception d'ensembles mécaniques ; le projet de machine-outil à commande numérique par ordinateur (CNC) décrit ci-après va vous aider à découvrir ce monde fascinant !

Caractéristiques

- surface de travail d'environ 50 × 30 cm²
- rails linéaires avec paliers insalissables en plastique
- vitesse maximale de 1,5 m/min sur les axes X et Y, 75 cm/min sur l'axe Z
- précision d'environ 0,2 mm répétable
- moteurs NEMA 17
- entraînement par courroies renforcées GT2 (axes X et Y) et vis sans fin (extension de l'axe Z)
- extensions de l'axe Z pour forage, fraisage et gravure de bois, plastiques et métaux non ferreux (aluminium, laiton) avec la Proxxon LBS/E (5 à 20 000 tr/min, 100 W)
- contrôleur et logiciels professionnels de chez Eding CNC (CNC310)
- pilotes de commande des moteurs pas-à-pas bâtis autour du contrôleur Trinamic TMC2160, garantissant un fonctionnement souple et silencieux
- logiciel 2,5D extensible à 3D
- concept multifonctionnel, adaptable à d'autres applications
- possibilité de raccorder un pendant pour la commande manuelle (console avec boutons et volant)
- consommation maximale d'environ 100 W (sans la perceuse)
- poids total avec une Proxxon LBS/E montée : 24 kg

Si vous voulez construire vous-même les parties mécaniques d'un appareil, vous avez sans doute songé à utiliser une imprimante 3D. Ces machines existent depuis un certain temps, mais ce n'est qu'avec la mise sur le marché de modèles peu coûteux que la technologie a vraiment pris son essor. La plupart utilisent la technique FDM (*Fused Deposition Modeling*), où un filament de plastique est fondu et extrudé en couches successives, mais on en trouve aussi sur la base de la stéréolithographie (photopolymérisation).

Inconvénients des imprimantes 3D

Les imprimantes 3D sont très populaires, mais elles ont tout de même quelques inconvénients. Les matériaux utilisés – filaments ou résines – sont assez chers et l'impression est relativement lente ; ce n'est donc pas l'outil idéal pour une production en série. L'objet imprimé avec les modèles FDM est aussi relativement rugueux, et il faut parfois encore le polir. Ceci empêche également le contact avec des aliments, car un nettoyage correct est pratiquement impossible.

En comparaison avec la sophistication des imprimantes 3D, les machines à CNC ordinaires ont l'air de perceuses primitives améliorées. Elles sont pourtant capables de produire un objet beaucoup plus rapidement qu'une imprimante 3D ; la finition est aussi meilleure et le choix du matériau plus large. Pour vous donner une idée : la fabrication d'un petit objet dure en moyenne cinq minutes avec une machine à CNC, des heures avec une

imprimante... Nous ne voulons pas critiquer les imprimantes 3D, qui restent le choix idéal pour un prototype ou des objets de formes compliquées.

La popularité des imprimantes 3D a un autre avantage indéniable pour notre projet : de nombreux composants mécaniques, comme des rails, des pignons ou des roulements sont disponibles partout ; il y a dix ans, cela aurait été un véritable cauchemar de les dénicher. Tous les composants de notre machine à CNC sont en principe « standard », et on les trouve aisément sur eBay ou d'autres sites. Nous allons maintenant passer à la partie mécanique du projet.

Châssis et ensemble

L'apparence de notre *Wolverine*, comparée aux machines-outils à CNC traditionnelles, ne manquera pas de frapper. Ces dernières sont en général bâties autour d'une surface de travail fixe et lourde, alors que nous avons choisi un châssis en aluminium. Les profilés de 40×40 mm de section contribuent non seulement à la rigidité de l'ensemble, mais ils permettent également d'envisager d'autres options d'emploi et d'extension de l'appareil.

La surface de travail est constituée d'une plaque de Béton Plex (panneau de contreplaqué recouvert d'une couche

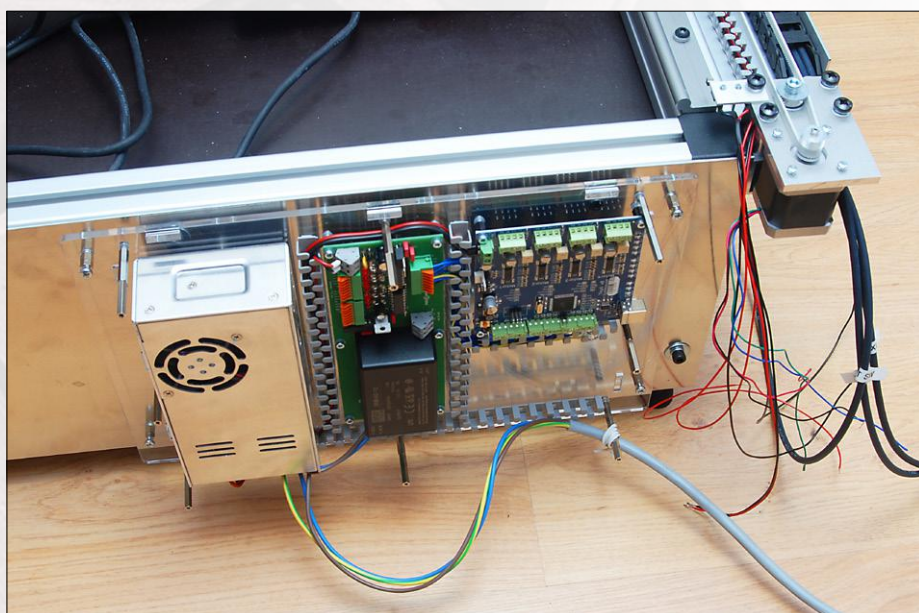


Figure 1. L'électronique trouve sa place au dos de la machine.

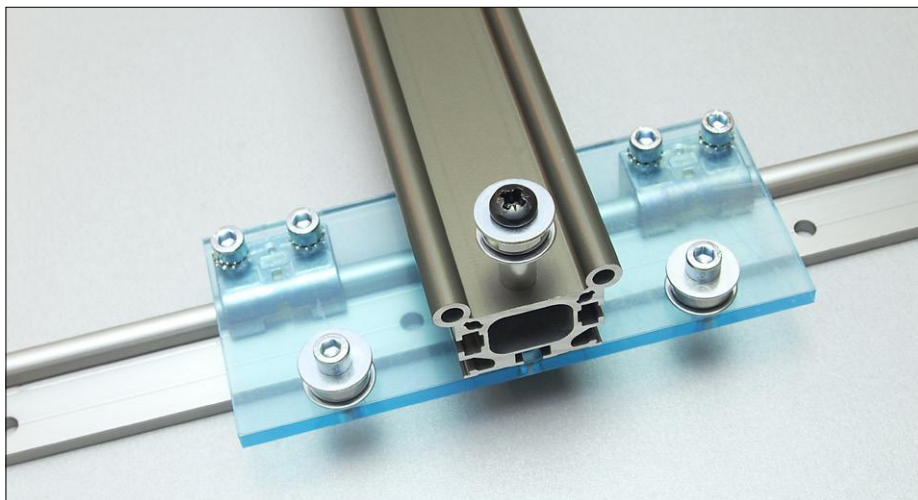


Figure 2. Deux rails simples WS-10 pour l'axe Y.



Figure 3. Un rail double WSX-10-40 pour l'axe X.

d'époxy) de 2 cm d'épaisseur, qui peut être facilement retirée du châssis. Ceci permet de choisir une autre surface de travail si nécessaire, d'en modifier la hauteur, ou même d'utiliser la machine sans surface de travail, par exemple pour usiner un objet de plus grande taille. Nous avons là une idée d'extension possible : une table réglable en hauteur pour la gravure ou la découpe au laser à CO₂, ou pour l'impression 3D.

La machine peut être utilisée en *pick-and-place*, pour le placement de CMS sur des circuits imprimés. Les bobines de composants ou les chargeurs CMS peuvent être fixés au châssis.

L'électronique de commande et l'alimentation sont fixées au dos de la machine (**fig. 1**). Celle-ci n'en est que plus compacte, et l'électronique est aussi mieux protégée des éclats et poussières lors

de l'usinage. Les pieds sont réglables, la machine peut donc être placée sur une surface inégale. On peut aussi s'en passer et mettre la machine directement sur une plaque de Béton Plex. Malgré la complexité de l'ensemble, la construction ne pose pas de difficultés particulières. Il suffit de s'y connaître un peu dans le travail des métaux et de disposer de quelques outils indispensables : une perceuse à colonne ou avec support de perçage, une scie circulaire avec lame pour l'aluminium, un jeu de tarauds, et divers forets.

En option on pourra utiliser comme plan de travail (*spoilboard*) un panneau en MDF avec des rainures en T pour fixer les pièces à usiner avec des brides de serrage Proxxon 24257. Ce plan sera fixé sur la plaque en Béton Plex et facilement remplacé s'il est endommagé. L'usinage

se fera bien entendu avec une fraiseuse fixée verticalement. Les détails – schémas, liste de composants et manuel d'utilisation – sont disponibles sur la page du projet [1].

Axes X et Y

• Rails et paliers

Pour les rails des axes X et Y, nous avons choisi des profilés en aluminium anodisé et des paliers lisses en plastique de la gamme DryLin-W de la firme allemande Igus [2]. Le rapport qualité-prix est excellent, et ces éléments sont solides et peu sensibles à la salissure. La partie en plastique des paliers pourra être remplacée aisément si cela s'avère nécessaire. Un autre avantage est que l'alignement ne doit pas être aussi précis qu'avec des guides à roulement à billes ; l'assemblage de la machine n'en sera que plus facile.

L'axe Y est constitué de deux rails simples de type WS-10 d'une longueur de 50 cm (**fig. 2**). L'axe X est quant à lui bâti autour d'un rail double de type WSX-10-40 d'une longueur de 75 cm (**fig. 3**). La traverse X étant autoportante sur une assez grande longueur, nous avons spécialement sélectionné un profilé surélevé. Si on veut construire une machine plus grande, les dimensions peuvent être adaptées dans certaines limites.

Les paliers de l'axe X peuvent être de différents types, en fonction de l'application : des WJUME-01-10 avec jeu réglable, des paliers flottants WJ200UM-01-10-P40, -P90 ou -P140, ou encore des modèles de précision WJUM-01-10K avec revêtement interne JUMO-01-10K. Ce revêtement doit être utilisé pour l'axe Z, conçu spécifiquement pour des fraiseuses.

Les paliers à jeu réglable et flottants permettent – lorsqu'ils sont correctement ajustés – un fonctionnement plus souple, le moteur devant fournir moins d'efforts. Ceci est idéal pour les applications sans forte contrainte sur l'axe X, par exemple la gravure ou la découpe au laser, l'impression 3D, ou la pose de CMS. Pour le fraisage, il est plutôt conseillé d'utiliser les modèles de précision WJUM-01-10K, la rigidité de l'axe X sera meilleure (moins de jeu). C'est indispensable avec un support de perçage/fraisage sur l'axe Z, surtout si on veut usiner des matériaux assez durs.

Il y a un inconvénient avec ces derniers paliers, un déplacement plus raide sur l'axe X. La puissance des moteurs est

suffisante, grâce à un dispositif de poulies, et ce même à des vitesses élevées. On peut aussi lubrifier légèrement les paliers avec de l'huile pour armes de type Ballistol GunCer, disponible en armurerie ou sur Amazon.

Les paliers de l'axe Y sont des modèles hybrides WJRM-01-10, qui peuvent aussi bien tourner que glisser. La stabilité en est meilleure aux vitesses élevées (> 10 m/min).

• Entraînement et moteurs

Notre machine se veut multifonctionnelle, mais nous voulions avant tout un appareil rapide pour la gravure au laser et le placement de CMS. Avec des vis sans fin c'est difficile à réaliser, nous avons

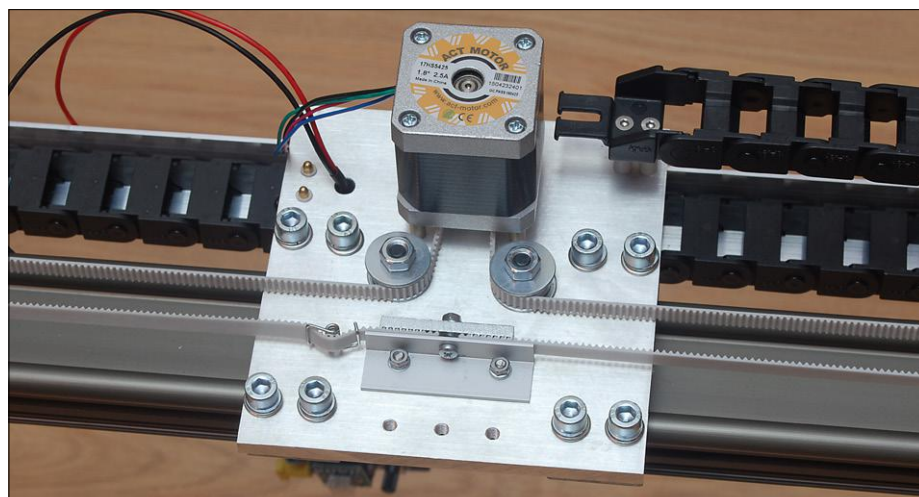


Figure 4. Des poulies permettent de découpler la force des moteurs.

► Un système de poulie permet de doubler la force. Les courroies GT2 renforcées avec des fibres d'acier sont beaucoup plus résistantes que celles en néoprène.

donc choisi des courroies crantées GT2 de 6 mm de large renforcées de fibres d'acier pour l'entraînement des axes X et Y. Elles sont de couleur blanche et très solides en comparaison avec les courroies classiques en néoprène. Le jeu des GT2 est très faible, il faut donc veiller à bien les tendre.

On trouve facilement des courroies GT2 sur eBay ou auprès de revendeurs de pièces détachées pour la fabrication d'imprimantes 3D. Pour le guidage des courroies, nous utilisons des roulements à billes 606 2RS, fixés entre deux rondelles M6 d'un diamètre extérieur de 20 mm

Les moteurs pas à pas sont des NEMA 17, avec un axe de 5 mm de diamètre. Des poulies (**fig. 4**) permettent de doubler leur force, ce qui est plus que suffisant pour des petits travaux de fraisage et les autres applications envisagées pour la machine. Avec un couple de maintien de 0,58 Nm et des poulies d'un diamètre de 12,8 mm, la force théorique en N des moteurs 17HS5425 est de $2 \times (0,58 \text{ Nm} \times 71 \% / 0,0064 \text{ m})$, soit 128,7 N pour de faibles vitesses le long des axes X et Y. Nous avons considéré une valeur théorique du couple dynamique égale à 71 % de la valeur du couple de maintien pour de faibles vitesses, c'est-à-dire avec des micro-pas (*microstepping*). Dans la pratique, nous ne commandons pas les

moteurs avec un courant maximal, et il y a des pertes mécaniques diverses dans le système d'entraînement ; une force de 60 à 70 N est donc plus proche de la réalité. Mais ne vous y trompez pas : si vous bloquez l'axe X, la machine se déplacera toute seule sur le sol !

De plus gros moteurs auraient l'inconvénient d'être plus lourds, et de présenter une plus grande inertie lors des accélérations et décélérations. Il faut aussi des pilotes qui peuvent délivrer un courant plus élevé, et qui sont plus chers.

L'axe Y est entraîné par un seul moteur pas à pas. On voit souvent deux moteurs pour un tel entraînement, mais si la synchronisation n'est pas parfaite – par ex. si un moteur rate quelques pas – la traverse peut se tordre, avec toutes les conséquences imaginables...

Avec les courroies GT2 et des roues à 20 dents, le déplacement le long de l'axe X est d'environ 20 mm par révolution du moteur. Le long de l'axe Y, il était de 20,6 mm sur notre prototype, car l'épaisseur des courroies joue un rôle. Ce sont des valeurs données à titre indicatif, il faudra calibrer votre machine avec un pied à coulisse.

Les supports des moteurs, des paliers et des roulements ont été réalisés avec des plaques T6 d'aluminium 6082 (6 mm d'épaisseur). Elles procurent suffisamment de rigidité à l'appareil dans son

ensemble et sont relativement faciles à usiner. Si vous ne comptez utiliser la machine que pour des travaux légers, vous pouvez utiliser des plaques en stratifié haute pression (HPL), de 6 mm d'épaisseur aussi, au lieu de l'aluminium ; leur usinage sera plus facile.

Axe Z

Le support du moteur de l'axe Z est pourvu de trois trous avec un pas de vis M5 pour le montage de diverses extensions. Notre première utilisation était pour une perceuse-fraiseuse permettant aussi la gravure.

La précision est ici plus importante que la vitesse, et l'entraînement est réalisé par une vis sans fin (**fig. 5**). On peut se procurer l'ensemble du mécanisme chez Igus (code SLW-1040). On peut choisir certains paramètres, dont l'excursion. Nous avons pris 60 mm, ce qui est en principe suffisant. Il est important de ne pas prendre une excursion trop longue, la stabilité en souffrirait. La charge, qui est essentiellement constituée de la perceuse-fraiseuse, est dirigée vers le bas ; il ne faut donc pas prévoir une précharge pour la vis.

Les paliers du dispositif SLW-1040 doivent être des WJUM-01-10K en plastique de chez Igus aussi, sinon le jeu radial risque d'être trop grand.

Nous retrouvons un moteur pas-à-pas

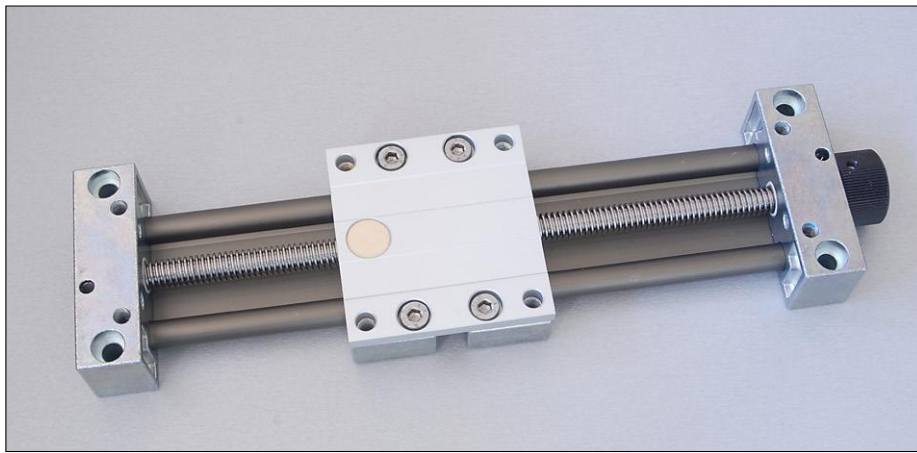


Figure 5. Sur l'axe Z, l'entraînement par vis sans fin (du type SLW-1040) permet une plus grande précision.



Figure 6. D'autres machines que cette Proxxon LBS/E conviennent aussi.

NEMA 17 pour compléter le dispositif via un couplage élastique. Le déplacement est de 2 mm par révolution, la force théorique est donc plus élevée que sur les autres axes. Il faut cependant tenir compte de la charge axiale maximale du SLW-1040, qui est de 700 N en statique, et de la charge que constitue l'axe Z dans son ensemble pour le support du moteur et les paliers de l'axe X. Il est conseillé dans la pratique de limiter la force à environ 20 N, ce qui est suffisant pour la plupart des applications. La

perceuse-fraiseuse est fixée par un support SK20 pour tiges de 20 mm. Nous avons utilisé une Proxxon LBS/E (**fig. 6**), qui convient pour des travaux de fraisage légers. D'autres marques ont des modèles semblables dans leur catalogue, par ex. Kress. Et rien ne vous empêche d'utiliser une véritable fraiseuse, en choisissant un support approprié. Le montage de l'ensemble axe Z sur la traverse de l'axe X se fait à l'aide d'un morceau d'aluminium de section carrée et de quatre vis M5. Ceci permet de

modifier la hauteur de l'axe Z sur cinq positions fixes. La hauteur de la Proxxon LBS/E peut aussi être ajustée, on a donc pas mal de souplesse à ce niveau.

Câblage et interrupteurs de fin de course

Les câbles qui relient les moteurs et les interrupteurs de fin de course à l'électronique sont en mouvement constant, soumis à des flexions avec un petit rayon de courbure, et sujets à la salissure. Des câbles standard ne résisteraient pas longtemps à ce traitement, et nous avons choisi des câbles chainflex® CF9.02.06 (**fig. 7**) prévus pour des sollicitations extrêmes ; ils possèdent six fils conducteurs d'une section de 0,25 mm², juste suffisante pour un courant de 2,5 A pour les NEMA 17. Dans ce type de câble, les fils de cuivre sont enroulés en spirale autour d'un noyau en plastique, ce qui fait que les conducteurs n'ont pas à supporter à eux seuls les flexions du câble. Le noyau de plastique et les gaines des conducteurs et du câble veillent à la résistance à la traction et à la rigidité du câble. La gaine du câble résiste aux produits chimiques courants, aux graisses et aux huiles. Le fabricant garantit un fonctionnement correct pour au moins cinq millions de mouvements aller-retour, et pour une plage de température de -35 à +100 °C.

Il n'est pas conseillé d'utiliser des câbles de plus forte section, ils ne seraient pas assez flexibles. C'est aussi la raison pour laquelle nous avons préféré utiliser trois câbles de six conducteurs plutôt qu'un ou deux câbles avec plus de conducteurs. Les câbles sont maintenus en position par des gaines flexibles, toujours de chez Igus. Elles semblent bien trop longues, mais ça évite des rigoles et manchons additionnels pour soutenir les câbles. Des tuyaux en PVC – par ex. pour l'aspiration, de l'air comprimé ou le refroidissement – peuvent prendre place dans les gaines flexibles. N'y introduisez cependant pas le cordon d'alimentation de la perceuse-fraiseuse, ce n'est pas fait pour cela !

Les interrupteurs de fin de course sont des petits modèles à levier (**fig. 8**), à l'état fermé au repos : il est toujours plus sûr d'avoir à rompre un contact que d'avoir à l'établir. Pour les axes X et Y, on en place un à chaque extrémité. Les interrupteurs à gauche et en dessous ont deux fonctions : limite de course et point de départ (*homing*). Les autres sont des

Liens

- [1] Page de l'article : <http://www.elektormagazine.fr/180246-05>
- [2] Igus : <http://www.igus.com>
- [3] Boutique en ligne Alu-messing-shop : <http://www.alu-messing-shop.de>
- [4] ACT Motor GmbH : <http://de.act-motor.com>
- [5] Proxxon : <http://www.proxxon.com>

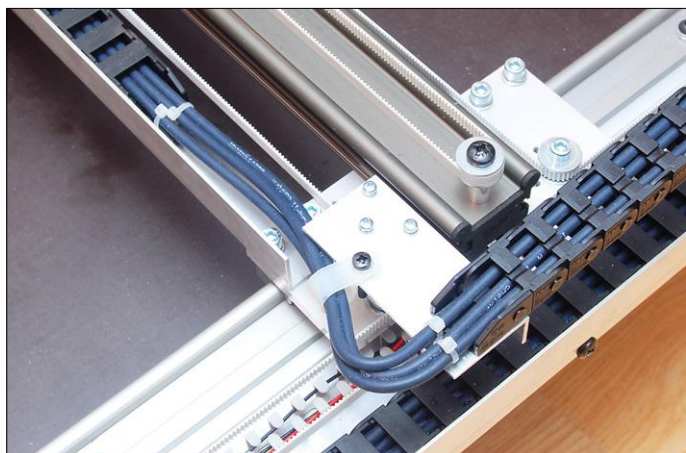


Figure 7. Ces câbles de haute qualité supportent des milliers de flexions.

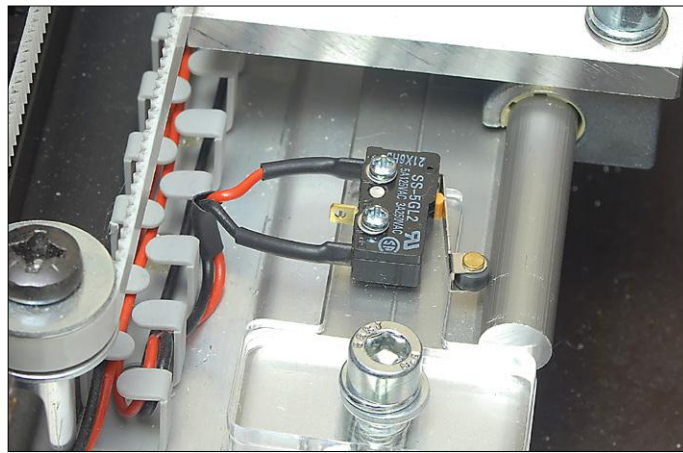


Figure 8. La bonne façon de monter les interrupteurs de fin de course à levier.



Figure 9. Un bouton d'arrêt d'urgence permet d'éviter une issue fatale à un foret en cas de mauvais fonctionnement.

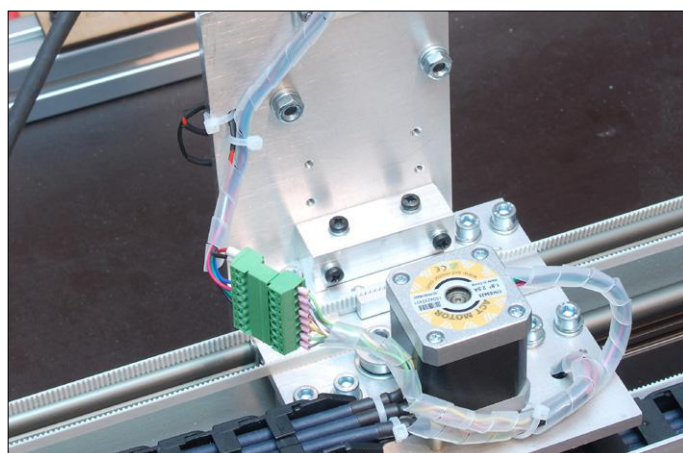


Figure 10. Un connecteur multibroche permet de changer facilement l'axe Z complet.

interrupteurs d'arrêt d'urgence au cas où la machine voudrait continuer au-delà de l'extrémité des axes ; ils sont optionnels et ne sont pas nécessaires au bon fonctionnement de la machine. Même du point de vue de la sécurité, on pourrait s'en passer, car les forces en jeu sont de toute façon limitées.

L'axe Z ne dispose que d'un seul interrupteur à double fonction : *homing* et limite de hauteur maximale (Z_{max}). Il n'est pas nécessaire de prévoir une limite de hauteur minimale, le foret aurait traversé le matériau de part en part avant qu'il soit activé.

Notez que les interrupteurs à double fonction sont montés dans le sens de la longueur par rapport aux axes. Ainsi ils ne constituent pas un obstacle physique et le retour au point de départ peut être plus rapide : lorsque l'interrupteur est activé, il y a encore un peu d'espace pour arrêter le dispositif sans danger d'un contact brusque.

Nous avons également prévu un bouton d'arrêt d'urgence (**fig. 9**). Cela peut paraître insignifiant, mais ça ne coûte pas cher et ça peut éviter le bris d'un foret. L'axe Z est connecté au reste de la machine par un connecteur multibroches (**fig. 10**). On peut dès lors facilement changer l'axe complet, pour par ex. remplacer une perceuse par un laser.

Électronique et logiciel : 2^e partie

Dans la seconde partie de l'article consacré à notre machine à CNC, nous décrirons l'électronique de commande et le logiciel.

Nous parlerons aussi de G-code, de logiciel de FAO, et des différentes étapes menant d'un dessin de CAO à un objet façonné par la machine-outil. ◀

(180246-05 –
version française : Jean-Louis Mehren)



@ WWW.ELEKTOR.FR

→ Livre en anglais '3D Printing and CNC Fabrication with SketchUp'
www.elektor.fr/3d-printing-and-cnc-fabrication-with-sketchup



la pratique des amplificateurs opérationnels

partie 2 : amplificateurs opérationnels à transistors à effet de champ dans les applications à large bande

Burkhard Kainka

Dans de nombreuses applications, il est crucial que les entrées d'un amplificateur opérationnel (ou AOP) aient une impédance extrêmement élevée. Ce n'est pas le cas des AOP standard à transistors bipolaires ; il faut donc faire appel à des AOP à entrées à effet de champ.

Le courant d'entrée d'un amplificateur standard comme le LM358 est de l'ordre de 30 nA. Avec une résistance d'entrée de 1 M Ω , il y a donc déjà une chute de tension de 30 mV, c'est incompatible avec de nombreuses applications. En revanche, un AOP à transistors d'entrée à effet de champ travaille avec un courant d'entrée presque nul.

Avec des J-FET, on atteint un courant d'entrée environ mille fois plus faible qu'avec des entrées bipolaires. Les AOP TL071 (simple), TL072 (double) et TL074 (quadruple) sont des représentants types des AOP à effet de champ. Comme on peut le voir sur la **figure 1**, ces amplificateurs utilisent des transistors J-FET dans les étages d'entrée et des transistors bipolaires dans le reste du circuit interne.

L'isolation est encore meilleure si l'on utilise des transistors MOSFET dans les étages d'entrée. Le CA3140 (**fig. 2**) est un représentant type de cette technologie. L'AOP travaille avec une alimentation double ou simple à partir de 4 V. La tension d'entrée peut atteindre une valeur de 0,5 V au-dessous de la tension d'alimentation négative. Comme les entrées des MOS-

FET sont très sensibles aux surtensions, des diodes Zener de protection sont ajoutées. Le courant d'entrée peut descendre jusqu'à 2 pA. L'étage de sortie du CA3140 est classique avec des transistors bipolaires. C'est pourquoi le constructeur a baptisé cette technologie BiMOS. En plus de la version simple, il y a également le CA3240, un AOP double.

Le CA3160 apporte une autre amélioration avec un étage de sortie à MOS complémentaires (CMOS). Cela permet à la tension de sortie de s'approcher à seulement 10 mV des rails d'alimentation positif et négatif (fonctionnement *rail-to-rail*).

Rampes de tension

Un AOP à entrées à FET est tout indiqué pour produire des formes d'onde à faible pente comme les rampes de tension. La **figure 3** montre un générateur de rampe type qui peut servir pour acquérir automatiquement les courbes caractéristiques de composants. L'AOP est câblé en intégrateur. La pente de la rampe est déterminée par le faible courant de l'entrée inverseuse. Grâce à la haute impédance d'entrée de l'AOP, il

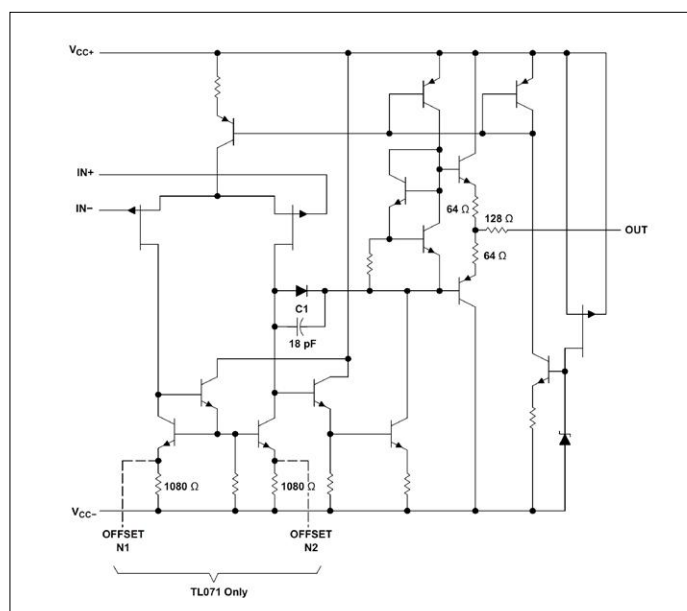


Figure 1. Schéma interne du TL071 (source : Texas Instruments).

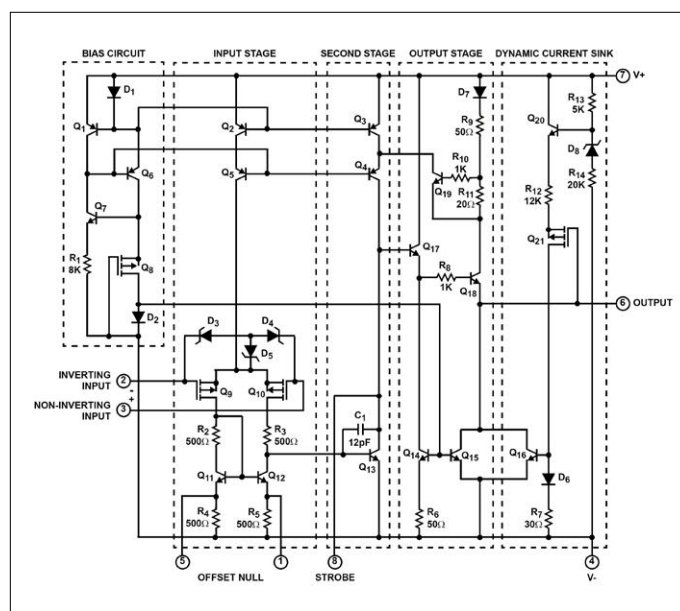


Figure 2. Schéma interne du CA3140 (source : Renesas).

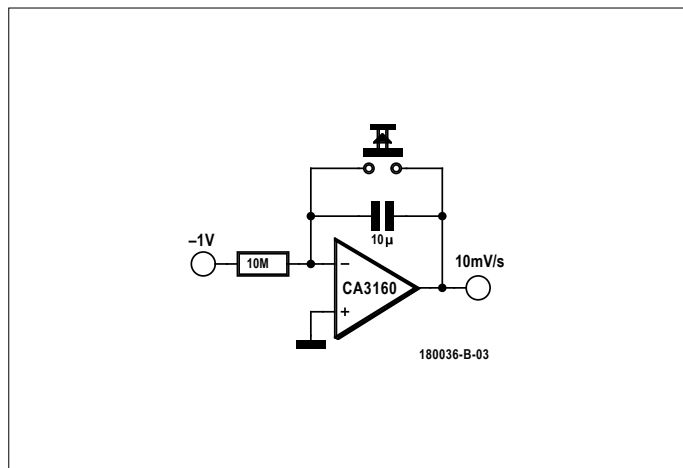


Figure 3. Générateur de rampe.

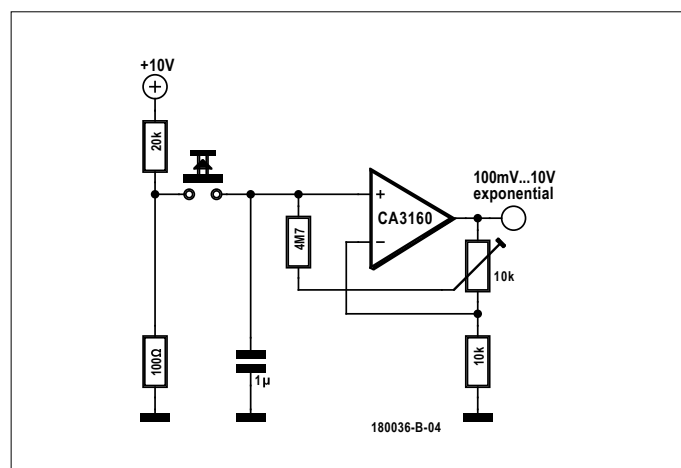


Figure 4. Générateur de rampe exponentielle.

n'y a pas de crainte à avoir quant à une erreur provoquée par le courant d'entrée. Un interrupteur monté en parallèle sur le condensateur de l'intégrateur le décharge et démarre une nouvelle rampe de tension.

Dans certains cas, il est préférable d'avoir une tension à croissance exponentielle et non pas linéaire. Le générateur de rampe exponentielle de la **figure 4** peut par ex. être utilisé pour produire un son destiné à des mesures audio. Le courant de charge du condensateur est déterminé par une tension elle-même proportionnelle à la tension aux bornes du condensateur. Cela conduit à une croissance exponentielle (**fig. 5**). Au départ, il faut une tension fixe donnée à l'entrée du circuit. Avec 50 mV et un gain en tension de deux, la tension de sortie est de 100 mV. Avec un AOP BiMOS à haute impédance d'entrée, il est possible d'adapter le schéma dans de vastes limites. De nombreux AOP modernes sont entièrement en technique CMOS et ne contiennent plus aucun transistor bipolaire. L'AOP CMOS double TLC272 en est un représentant typique avec des courants d'entrée inférieurs au picoampère. La **figure 6** illustre son schéma interne. Certains AOP CMOS travaillent avec des entrées et des sorties dont les tensions peuvent atteindre les tensions d'alimentation.

Les caractéristiques des AOP apparaissant sur le marché sont sans cesse améliorées. Il s'agit soit d'augmenter encore l'impédance d'entrée, soit de réduire la tension de décalage des entrées. Mais il peut aussi s'agir d'obtenir un fonctionnement *rail-to-rail* en entrée comme en sortie avec la possibilité de piloter des charges de sortie à basse impédance. Dans certains domaines, l'AOP idéal est approché de très près. Mais très souvent, d'excellentes caractéristiques dans un domaine sont obtenues au prix de performances moins bonnes ailleurs. C'est ainsi que la plupart des AOP CMOS à haute impédance d'entrée présentent une tension de décalage d'entrée élevée ; en revanche les modèles dont la tension de décalage est réduite sont pénalisés par leur fréquence de coupure plus basse.

Le TS914 est un AOP universel quadruple de nouvelle génération utilisable en *rail-to-rail* en entrée comme en sortie. Le schéma interne (**fig. 7**) montre l'étage d'entrée différentiel complémentaire et un étage de sortie *push-pull* à MOSFET. Ce type d'ampli est un bon choix dans de nombreuses applications dont la SDR (*Software Designed Radio* = radio définie par logiciel) d'Elektor. Avec un courant d'entrée de seulement

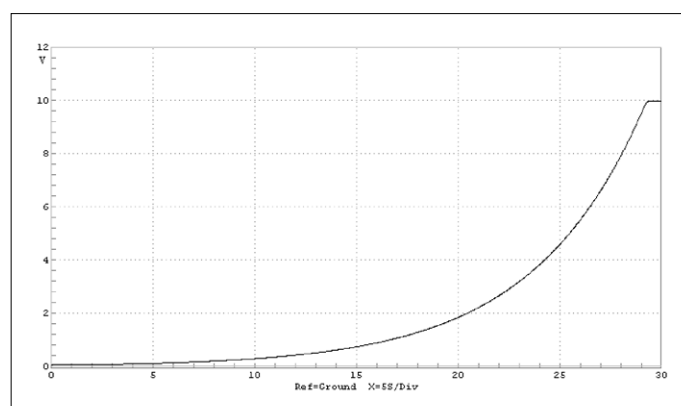


Figure 5. Sortie en tension du générateur de rampe exponentielle.

1 pA, il atteint une tension de décalage de 5 mV seulement ainsi qu'une bande passante opérationnelle de 800 kHz et accepte une charge à basse impédance descendant à 100 Ω. Les AOP à impédance d'entrée extrêmement élevée sont utilisés dans ce qu'on appelle des amplificateurs d'instrumentation. Un ampli d'instrumentation comme celui de la **figure 8** possède des entrées différentielles et une sortie unipolaire. Il

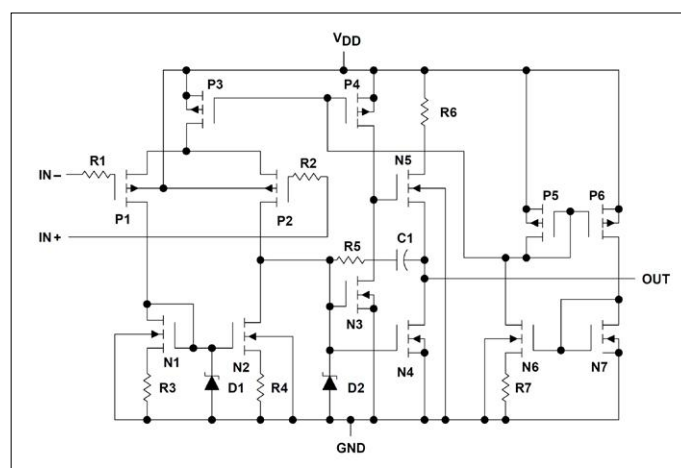


Figure 6. Schéma interne du TLC272 (source : Texas Instruments).

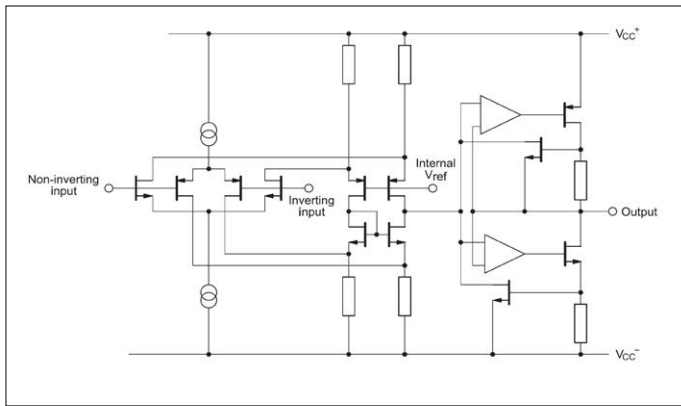


Figure 7. Schéma interne du TS914 (source : STMicroelectronics).

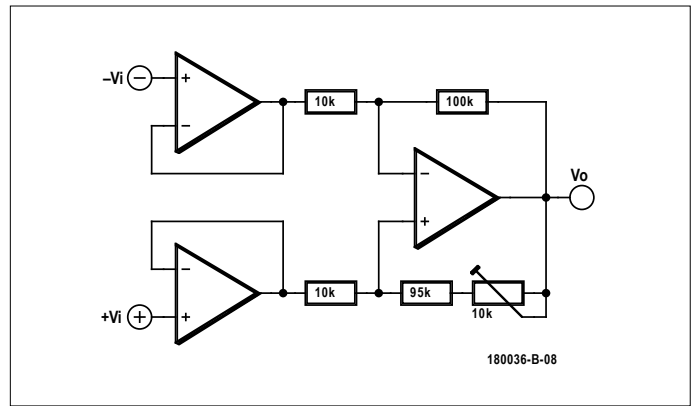


Figure 8. Amplificateur d'instrumentation.

permet d'effectuer des mesures sur des objets non connectés au potentiel de masse. La tension par rapport à la masse doit cependant rester dans les limites permises par les entrées de l'AOP. Le schéma fait appel à un amplificateur différentiel réglable et à deux convertisseurs d'impédance d'entrée afin d'atteindre une impédance d'entrée extrêmement élevée. L'amplificateur différentiel doit être ajusté pour obtenir la meilleure réjection en mode commun et la meilleure stabilité.

Applications à large bande

Bien qu'un AOP soit conçu comme amplificateur de tension continue, il s'accommode très bien de l'amplification de tensions alternatives et permet donc par ex. de réaliser un amplificateur de microphone (fig. 9). Avec une tension d'alimentation unique, une tension intermédiaire, en général égale à la moitié de la tension d'alimentation, sert de masse virtuelle. Le circuit se comporte alors comme s'il possédait deux tensions d'alimentation symétriques, l'une positive, l'autre négative. Un tel circuit amplificateur peut travailler sur une vaste gamme de fréquences, mais on s'étonne parfois de buter rapidement sur les limites d'un AOP. Un amplificateur peut par ex. être calculé pour un gain en tension de 100 et travailler jusqu'à 20 kHz. Le produit $\text{gain} \times \text{bande passante}$ doit alors atteindre $100 \times 20 \text{ kHz} = 2 \text{ MHz}$. De nombreux AOP n'atteignent

que 1 MHz. Pour atteindre 20 kHz, le gain n'est plus suffisant de sorte que la contre-réaction ne fonctionne pas comme souhaité. Il s'ensuit des distorsions, car des étages internes satureront. À l'oscilloscope, une sinusoïde se transforme souvent en signal triangulaire. En effet, en interne, la saturation conduit à des signaux carrés qui, en raison du temps de montée limité des étages finaux, se transforment en sortie en signaux triangulaires.

Pour le *shield* SDR d'Elektor (07/2018), un AOP quadruple TS914 travaille avec un produit $\text{gain} \times \text{bande passante}$ de 0,8 MHz (fig. 10). Pour préserver une bande passante suffisante, l'amplification d'un facteur 100 est partagée en deux étages ayant chacun un gain de 10. Cela permet d'atteindre une bande passante de 80 kHz, suffisante pour cette application. Les signaux aboutissent à l'entrée stéréophonique d'une carte son et sont traités par le logiciel SDR. Selon la carte son, il est possible d'utiliser une bande passante de 24 kHz ou 48 kHz. Le traitement des deux voies stéréo demande une bande passante effective de 48 kHz ou 96 kHz.

Pour une application autonome du récepteur, il faudrait remplacer le traitement du signal par le PC par une solution purement électronique. On peut utiliser pour cela un filtre IQ-CW à 750 Hz (fig. 11). En entrée, se trouve un circuit additionneur pour les signaux I et Q auquel est ajouté un circuit dépha-

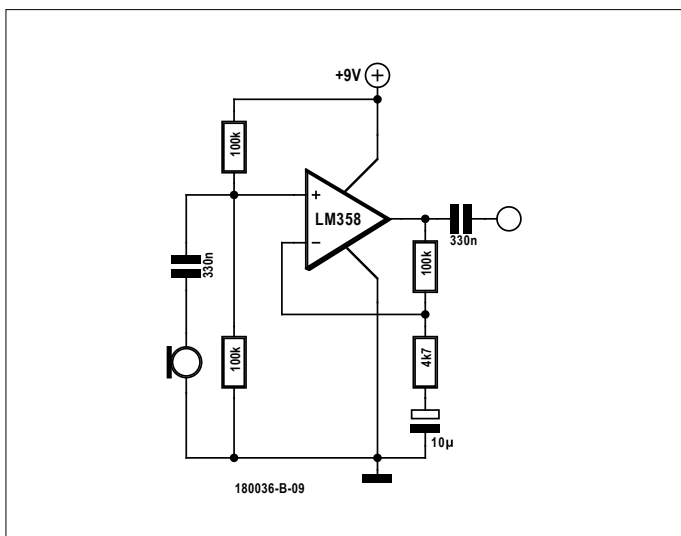


Figure 9. Amplificateur de microphone à AOP.

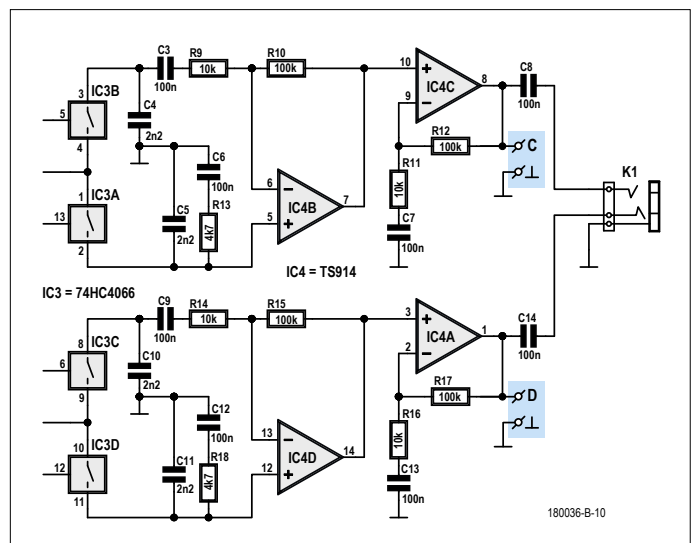


Figure 10. AOP universel quadruple TS914 du *shield* SDR d'Elektor.

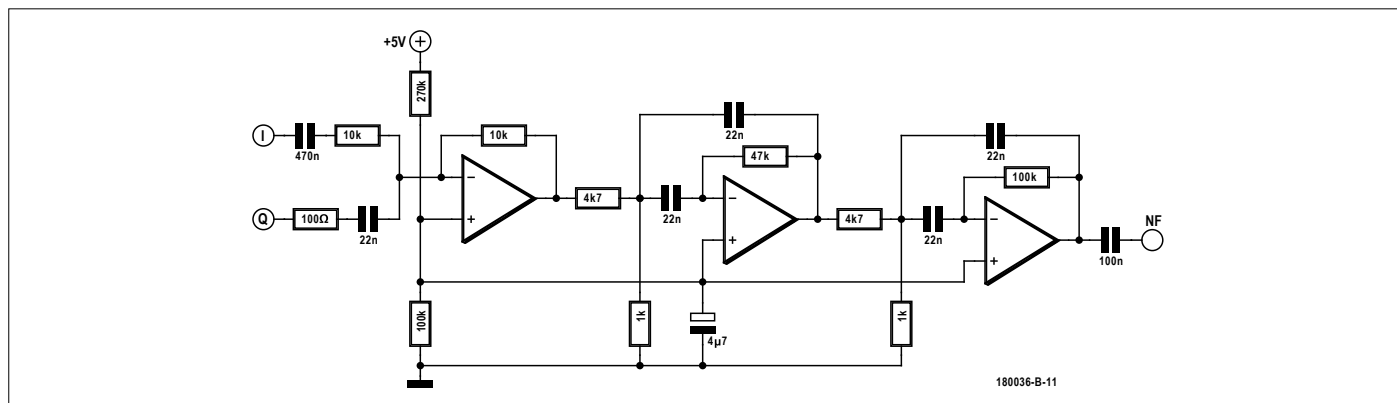


Figure 11. Filtre IQ-CW.

sant de 90° à 750 Hz. Nous trouvons ensuite un filtre à deux étages, passe-bande d'une largeur de 750 Hz environ. Il utilise un double AOP de type LM358.

Les étages du filtre ont été conçus avec le simulateur LTspice. Mais attention, ici aussi, la bande passante limitée de l'AOP joue un rôle. On pourrait en effet penser que tout se passe en BF, et que donc aucun problème ne peut arriver. Nous avons effectivement dans le deuxième étage du filtre un gain de 100. Pour un signal de 800 Hz, nous avons donc besoin d'un produit *gain* $\times$ *bande passante* de 80 kHz seulement. Cependant, le signal peut comporter des fréquences plus élevées qui ne doivent pas produire de distorsion. Et c'est encore plus critique

pour construire un filtre de haute qualité pour les fréquences plus élevées, au-dessus de 10 kHz. ◀

(180036-B-04 – version française : Yves Georges)

@ WWW.ELEKTOR.FR

→ Livre en anglais 'The Circuit Designer's Companion'
www.elektor.fr/the-circuit-designers-companion-3rd-edition

Livre en anglais 'Learning the Art of Electronics'
www.elektor.fr/learning-the-art-of-electronics-3rd-edition

Publicité

Salon de l'innovation en électronique, mesure, vision et optique

L'INNOVATION INDUSTRIELLE S'INVITE À L'OUEST

3-4 AVRIL 2019

CITÉ DES CONGRÈS NANTES

Votre badge gratuit sur www.enova-event.com



vol tous azimuts

l'électronique par monts, maux et merveilles

Clemens Valens (labo d'Elektor)

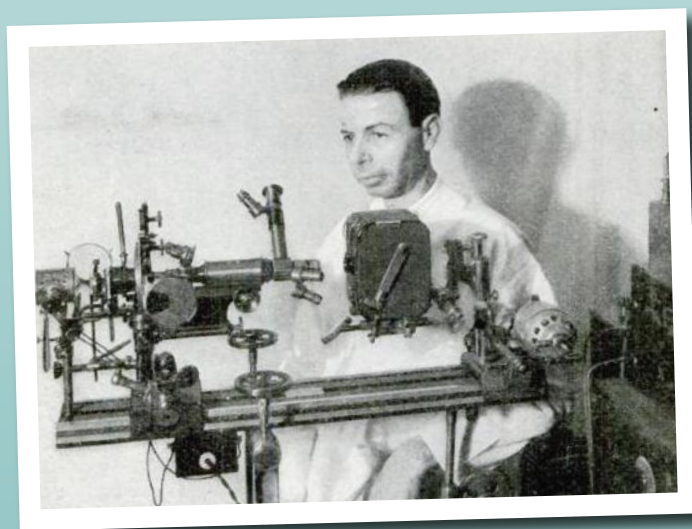


Résonance sympathique — vaudou ondulatoire

L'internet regorge de communautés, produits et technologies qui ne demandent qu'à être découverts. C'est ainsi que de temps à autre nous tombons sur une chose qui, pour une

raison ou une autre, suscite notre intérêt (*). Poussés par la curiosité, explorant lien après lien, nous débouchons alors soudainement au beau milieu d'une communauté dont la vie entière semble tourner autour de cette chose. Voilà comment je me suis un jour retrouvé nez à nez avec la machine de Rife. Je n'en avais jamais entendu parler, et pourtant elle semble avoir pas mal d'adeptes.

Avant de poursuivre, disons clairement que les machines de Rife et la théorie qui les sous-tend sont pour le moins controversées. Le sujet semble flirter avec des pseudosciences telles que la thérapie par champ magnétique ou l'énergie des pyramides. Inventée dans les années 1920 par Royal Raymond Rife (1888-1971), la machine de Rife est un appareil électronique censé détruire les microorganismes malveillants de type bactéries, virus et cellules cancéreuses. Comment ? En les faisant vibrer à leur fréquence propre. Cette fréquence de résonance a été baptisée *Mortal Oscillatory Rate* par Royal Rife, mais est plus communément appelée fréquence de Rife. L'effondrement du pont de Tacoma en 1940 est un exemple bien connu du potentiel destructeur du phénomène de résonance. Le professeur de musique Anthony Holland a repris, et en quelque sorte réinventé, les travaux de Rife il y a une dizaine d'années. Vous trouverez sur YouTube une présentation très claire de son travail.



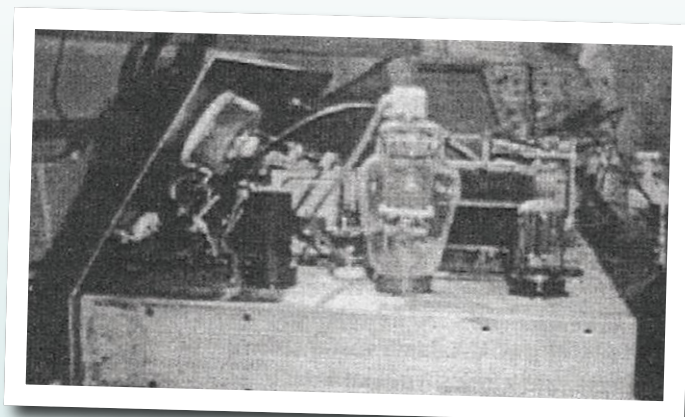
Royal Raymond Rife posant en 1931 devant un de ses microscopes. Il les utilisait pour observer les germes soumis à la résonance sympathique. (Source : Wikipédia)

* Feu le site StumbledUpon permettait ce genre de découvertes en fonction de vos intérêts.

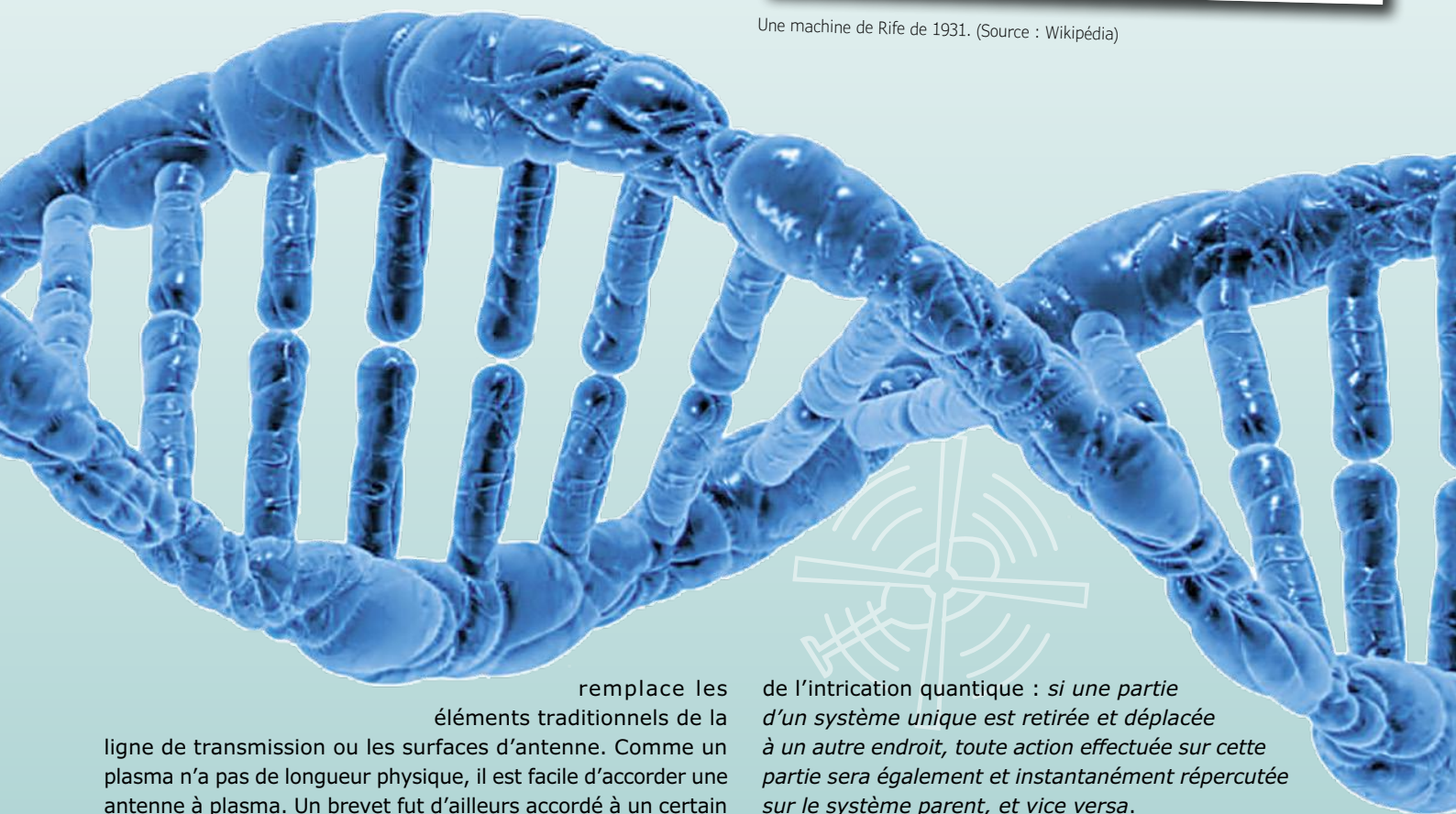
Une machine de Rife utilise un générateur de fonctions ordinaire pour créer des signaux dont les fréquences « mortelles » sont déterminées empiriquement. Lorsque ces signaux sont dirigés vers les pathogènes infectant un organisme, ceux-ci, tel un verre brisé par la voix de la Castafiore, sont détruits par résonance sympathique.

Les signaux peuvent être transmis selon trois modalités : par contact direct, au moyen d'électrodes p. ex. ; sans fil, à l'aide d'antennes à plasma ; enfin à distance, en mêlant espace non local, physique quantique et propriétés radio-réceptrices de l'ADN.

Dans une antenne à plasma fortement ionisé, le plasma, conducteur par nature,



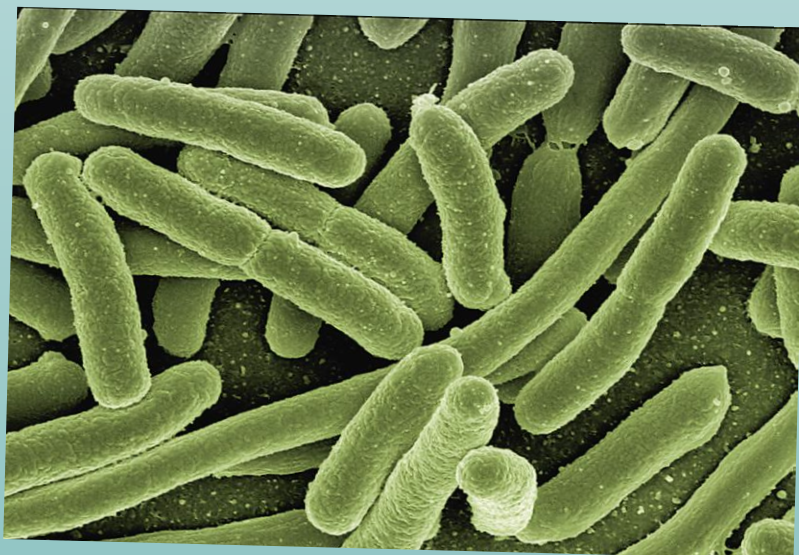
Une machine de Rife de 1931. (Source : Wikipédia)



remplace les éléments traditionnels de la ligne de transmission ou les surfaces d'antenne. Comme un plasma n'a pas de longueur physique, il est facile d'accorder une antenne à plasma. Un brevet fut d'ailleurs accordé à un certain J. Hettinger dès 1919 pour cette technologie. Plus récemment, en 1999 pour être précis, James E. Bare a obtenu le brevet US n° 5,908,441 pour son Appareil thérapeutique à fréquence de résonance à antenne à plasma. Le principal avantage de la méthode reposant sur l'ADN est que la machine peut opérer à distance, même sur des milliers de kilomètres. L'opérateur place une rognure d'ongle du patient dans un petit contenant, puis envoie l'énergie de la fréquence audio à l'ADN contenu dans la rognure. Sur le site de Spooky2 – une machine de Rife bon marché – on peut lire : *L'ADN étant formé d'un enroulement hélicoïdal à deux brins, la molécule agit comme une antenne capable de transmettre et de recevoir via l'espace non-local l'information codée dans les signaux énergétiques. C'est par intrication quantique que l'énergie est supposée être relayée instantanément à l'ADN parent, c.-à-d. à celui du patient (il s'agit d'une technique très précise). Le même site explique ensuite qu'en raison*

de l'intrication quantique : *si une partie d'un système unique est retirée et déplacée à un autre endroit, toute action effectuée sur cette partie sera également et instantanément répercutée sur le système parent, et vice versa.*

Même si le rapprochement pourra sembler excessif, la



United States Patent [19] Bare



US005908441A

[11] Patent Number: 5,908,441
[45] Date of Patent: Jun. 1, 1999

[54] RESONANT FREQUENCY THERAPY DEVICE

[76] Inventor: James E. Bare, 8005 Marble Ave, NE,
Albuquerque, N.M. 87110

[21] Appl. No.: 08/784,794

[22] Filed: Jan. 16, 1997

[51] Int. Cl.⁶ A61N 1/02

[52] U.S. Cl. 607/1; 607/2; 607/156

[58] Field of Search 607/1-2, 145,

607/150, 151, 154, 155, 156, 88, 89, 93,
100, 101, 115; 606/1-2, 333/32-35, 227;
343/762, 772; 340/384.8, 384.2; 367/137-139

References Cited

U.S. PATENT DOCUMENTS

4,119,102 10/1978 LeVeen 607/154
4,315,514 2/1982 Drewes et al.

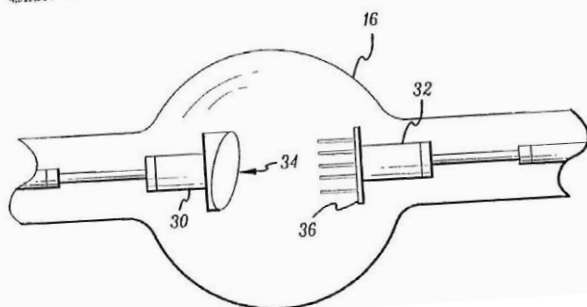
4,336,809 6/1982 Clark 128/665
4,545,368 10/1985 Rand et al. 607/98
4,680,749 7/1987 Englund et al.
4,852,086 7/1989 Eastmond et al.
4,955,083 9/1990 Phillips et al.
5,300,068 4/1994 Ronar et al. 606/34
5,402,782 4/1995 Lodder
5,553,610 9/1996 Lodder
5,676,095 10/1997 DiMino et al. 607/154

Primary Examiner—Kenneth E. Peterson
Assistant Examiner—David M. Ruddy
Attorney, Agent, or Firm—Hoffman, Wasson & Giller

ABSTRACT

[57] A generator of a complex energy wave, having audio, radio and light components, including an audio frequency oscillator, a radio frequency transmitter, a radio frequency amplifier, an antenna tuner, an antenna, tuned coaxial cables and an optional reverberation unit.

10 Claims, 3 Drawing Sheets



La première page d'un brevet de 1999 d'un *Resonant Frequency Therapy Device*.

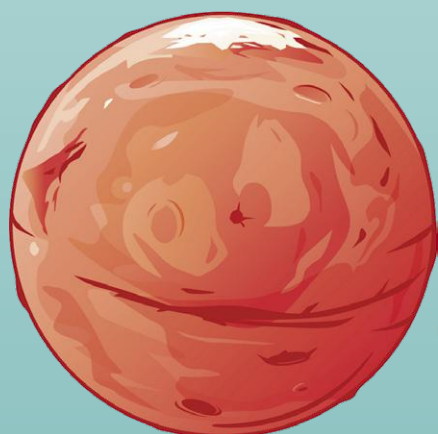
magie dite sympathique ou imitative recourt depuis des lustres à de telles techniques. Ses praticiens croient en effet qu'une personne mise en contact avec un objet lui transfère une partie de son énergie. Placez ledit objet – ou une mèche de cheveux – dans une poupée (vaudou), percez-la d'aiguilles, et la personne connectée énergétiquement devrait au moins se gratter. De toute évidence, l'intrication quantique ne date pas d'hier. Il existe en réalité deux autres façons de diriger les oscillations mortelles vers une partie du corps : par thérapie par laser froid

(LLLT), une technique exploitant des photons non-thermiques – les biophotons étudiés par Fritz-Albert Popp – et par thérapie par champ électromagnétique pulsé (PEMFT). La LLLT et la PEMFT sont couramment utilisées en médecine parallèle, certains indices laissant même à penser qu'elles pourraient avoir un léger effet bénéfique. La *Food and Drug Administration* (FDA) a d'ailleurs autorisé la vente de certains dispositifs de PEMFT en tant qu'appareils de « bien-être ».

Quant aux machines de Rife, leur vente est illégale dans certains pays, aucune institution médicale ou scientifique n'ayant approuvé leur utilité. L'électronicien n'en a cure, c'est le cas de le dire, puisque le générateur de fonctions fait partie de sa panoplie. Tout ce que vous avez à faire est de relier la sortie du générateur à un être vivant sans le mettre en danger. D'où l'intérêt de la machine distante : elle évite tout contact direct entre le générateur et le patient. Connectez simplement la sortie du générateur à une rognure d'ongle fournie par le patient – l'adaptation d'impédance peut être un problème ici – tournez le bouton, et voilà. Inutile de vous soucier de la puissance, l'antenne que forme la double hélice d'ADN est très efficace ; prenez juste garde de ne pas rôti la rognure. Depuis que j'ai installé un tel dispositif dans mon labo, mon générateur balayant continuellement une large plage de fréquences de façon à couvrir plusieurs types de germes, je n'ai pas contracté une seule fois la grippe espagnole.

Si cet article donne lieu à un nombre suffisant de réponses sympathiques, je serai sans doute enclin à en écrire un sur le moteur Keely, un moteur à « sympathie vibratoire ».

Note : nous déclinons toute responsabilité en cas de dommage ou dysfonctionnement causés par la pratique de la résonance et/ou de la magie sympathique sur des êtres vivants ou des objets inanimés.



Écoutez le vent sur Mars :
youtu.be/o3cxuIsEFRM

Trucs & astuces



Comment savoir si une pile est chargée ou à plat ? Facile, faites-la tomber d'une hauteur de 20 cm sur une surface plate, une table p. ex. Si elle rebondit, c'est qu'elle est vide ; si elle fait un plat, c'est qu'elle est chargée. Sceptique ? Pourtant ça marche bel et bien avec des piles alcalines non rechargeables de type AA ou AAA. Vraiment. Nous avons fait le test au labo. L'explication tient à la matière collante interne qui se solidifie lorsque la pile est déchargée : sa capacité à absorber les chocs diminue, d'où le rebond.

Des véhicules électriques trop silencieux ?



Les véhicules électriques sont considérés par beaucoup comme la panacée universelle des transports, pourtant ce ne sera pas le cas. En particulier parce que le silence de leur moteur empêche piétons et cyclistes d'être avertis de leur présence et les met donc potentiellement en danger. L'ampleur du problème est telle que partout dans le monde, des gouvernements planchent sur des mesures législatives relatives à un niveau sonore *minimal* de ces moteurs. Oui, pour aider les malvoyants, piétons et cyclistes à percevoir et localiser les VE, il faut produire plus de pollution sonore ! Heureusement, il existe déjà certaines technologies pour se faire entendre : à l'indienne en klaxonnant en permanence, ou encore façon jeunes mariés en traînant des conserves. Faire du bruit,

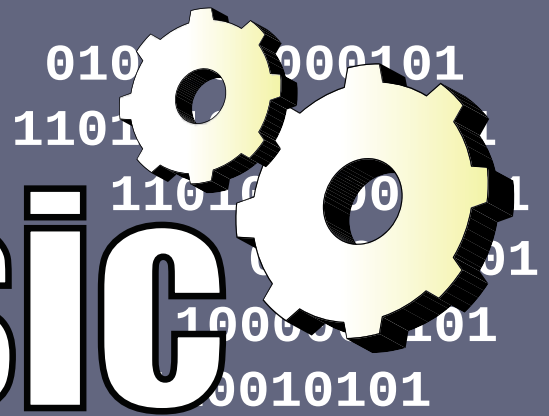
l'humain sait faire. Au labo d'Elektor, nous privilégions cependant les solutions électroniques, ce qui explique notre soutien à la note d'application AN-1552 d'Analog Devices concernant un *Electric Vehicle Warning Sound System* (EVWSS). Un tel système « produit une suite de sons alertant les piétons de la présence d'un VE. Le conducteur peut déclencher des sons d'alerte (similaires à un klaxon, mais moins impératifs), ceux-ci devant toutefois être automatiquement activés à faible vitesse. Leur nature va des tonalités artificielles aux sons réalistes imitant le bruit d'un moteur ou celui de pneus roulant sur du gravier. » (Songeaient-ils à des VE sans roues en écrivant cela ?) La lecture de cette note rappelle la scène du film *Sacré Graal* des Monty Python dans laquelle le valet de pied du roi Arthur frappe deux moitiés de noix de coco vides pour suggérer que son souverain progresse sur sa royale monture.

AD a préparé deux systèmes EVWSS comprenant un son de moteur d'habitacle ainsi qu'un son de moteur extérieur. Tous deux sont produits par le processeur BlackFin ADSP-BF706. Les systèmes d'entrée de gamme peuvent utiliser le processeur SigmaDSP ADAU1450.

(180567-B-04 - version française : Hervé Moreau)

Vous voulez participer ? Envoyez vos commentaires, suggestions, trucs et astuces à labs@elektor.com

PureBasic



un compilateur BASIC, beaucoup de plateformes

Waldemar Pieczek

Il existe un programme assez peu connu pour le développement interplateformes : PureBasic, un compilateur pour le BASIC, un langage de programmation âgé de plusieurs dizaines d'années. Les programmes compilés par PureBasic sont petits, rapides, et peuvent être lancés à partir d'une clé USB sans nécessiter d'installation. Il est donc intéressant d'examiner ce compilateur de plus près.

Caractéristiques

- Réalisation d'applications pour les systèmes à 32 et 64 bits
- Concepteur d'interface pour la réalisation d'interfaces utilisateur (GUI)
- Création de DLL personnelles
- Utilisation de pointeurs
- Accès aux API du système d'exploitation
- Assembleur en ligne
- Compilateur et débogueur intégrés
- Production de fichiers immédiatement exécutables

De nos jours, le développement interplateformes est la tendance dans le domaine des langages de programmation. Il répond à l'idée de ne plus programmer spécifiquement pour un système d'exploitation, mais d'écrire le code une seule fois et de laisser à l'environnement de développement la tâche d'adapter automatiquement le programme au système d'exploitation choisi. Même si l'idée est loin d'être neuve, elle gagne constamment du terrain, car un logiciel efficace doit pouvoir tourner indifféremment sur tous les environnements PC

et de préférence aussi sur ceux de tous les mobiles. Cette façon de procéder est appelée « Développement rapide d'applications (RAD) ». Dans ce cadre, les éditeurs de logiciel, petits ou grands, proposent des solutions pour lesquelles le nombre de plateformes cibles est plus ou moins élevé.

En pratique, il existe deux variantes, avec leurs avantages et leurs inconvénients :

1. Le code n'est écrit qu'une fois, mais doit être compilé sur la plateforme cible. Le fichier exécutable peut rester très petit, car le programme utilise autant que possible les fonctions du système d'exploitation (par exemple la gestion d'un bouton). Le programme peut alors être directement exécutable.
2. Le code n'est compilé qu'une seule fois et est immédiatement exécutable sur toutes les plateformes cibles. Comme les éléments de l'interface graphique utilisateur (par ex. les boutons) ont un aspect qui dépend du système d'exploitation, de même qu'il y a des différences de taille et de largeur des bords, il est nécessaire de vérifier l'interface utilisateur de chaque système cible et de l'adapter au besoin. Pour l'éviter, l'environnement de

développement apporte ses propres éléments indépendants du système d'exploitation, qui doivent être installés sur le système cible pour que le programme devienne exécutable. Cela augmente notablement la taille du fichier compilé sur le disque dur.

PureBasic [1] est un programme de développement interplateformes assez peu connu. C'est un compilateur du langage de programmation BASIC. Malgré son niveau de notoriété assez faible, il peut s'enorgueillir d'un passé de plusieurs dizaines d'années. Selon certaines rumeurs, il serait très apprécié dans le milieu des hackers russes. PureBasic appartient au premier groupe : les programmes compilés sont petits, rapides, et peuvent être démarrés sur une clé USB, sans installation. Les systèmes d'exploitation des mobiles ne sont toutefois pas supportés. La plupart des langages actuels sont orientés objet. PureBasic fait exception : c'est un langage procédural, mais qui peut travailler avec des objets. La programmation procédurale permet une syntaxe plus légère, dont la simplicité est à l'avantage du programmeur amateur, dont le temps d'apprentissage est réduit d'autant. Les électroniciens trouveront particulière-

ment intéressantes les possibilités d'accès aux périphériques comme les imprimantes, les ports série et l'Ethernet, ainsi que les fonctions spéciales comme l'envoi de courriels. Si l'on veut afficher des données de mesures ou des diagrammes, on trouvera dans les forums de nombreux exemples de code. Certains sont toutefois tellement longs qu'ils ne sont pas compilables avec la version de démonstration de PureBasic, qui est limitée à un certain nombre de lignes de code. Il est bon de ne pas négliger les nombreux exemples fournis, qui facilitent l'apprentissage.

Un coup d'œil sur la version Windows

Les auteurs de PureBasic exigent comme système d'exploitation au moins Windows XP. J'ai essayé de l'installer sous Windows XP et, effectivement, ça a marché ! Non seulement PureBasic produit des fichiers .exe qui tournent sans nécessiter de bibliothèques additionnelles à installer, mais l'environnement de développement lui-même n'a pas besoin de fichiers autres que ceux du répertoire des programmes. Dès que PureBasic est installé, on peut copier le répertoire PureBasic issu du répertoire des programmes sur une clé USB et avoir son compilateur BASIC toujours avec soi. Le processus de compilation est simplement un peu plus long à partir de la clé USB.

En principe Linux est supporté aussi, mais avec un petit bémol : PureBasic ne sait pas compiler les programmes destinés à l'ordinateur monocarte Raspberry Pi. Seuls les processeurs compatibles Intel de la gamme X86 sont supportés. À l'origine, PureBasic a été écrit pour l'AmigaOS. On le remarque encore aujourd'hui par le fait qu'à certains endroits, sa syntaxe se distingue nettement des variantes Microsoft du BASIC (QBasic, VB, VBA). À d'autres endroits, on a l'impression d'un afflux d'éléments du langage C.

Quelques différences avec les autres dialectes BASIC :

- Les commentaires sont introduits par un point-virgule.
- Les éléments des fenêtres de l'interface utilisateur (champs de texte, boutons, etc.) sont appelés gadgets.
- L'exponentiation est réalisée par une fonction séparée : `pow(a,n)` correspondant à a^n .

- Le mot clé `Define` permet de déclarer des variables. Par exemple : `Define.s Variable1, Variable2, Variable3` est la déclaration de trois variables de type chaîne de caractères.
- La déclaration directe d'une variable est possible : `Variable4.d`. On fait suivre le nom de la variable de l'abréviation de son type (ici `Double`) séparée par un point.
- Il n'y a pas de différence entre fonctions et procédures ; il n'y a que des procédures.
- Le passage de variables à ou par une procédure s'effectue de la manière suivante :

```
Procedure.d
MyFunction(Number1.d, Number2.d)
    result.d = .....
    ProcedureReturn result
EndProcedure
```

On commence par déclarer `MyFunction`, suivi du type de la valeur retournée (`Double`), puis on passe deux variables doubles à la procédure.

- Les conditionnels `If-Then` : le mot clé `Then` n'est pas utilisé. Un bloc conditionnel a l'aspect suivant :
- ```
If a=1
....
EndIf
```
- `MessageBox` : la lucarne d'information connue dans le monde Microsoft sous le nom de `MessageBox` s'appelle « `MessageRequester` » en PureBasic.
  - Les opérateurs mathématiques peuvent s'écrire de manière abrégée : `A+1` est identique à `A=A+1`, `B*2` est la même chose que `B=B*2` et `C<<1` signifie `C=C<<1`, le décalage à gauche d'un bit.
  - Les boucles `For-Next` ne fonctionnent qu'avec des valeurs entières.
  - Pour sortir d'une boucle, on utilise le mot clé `Break` :
- ```
For i=1 to 10
    If i=5
        Break
    EndIf
Next
```
- `Break` suivi d'un nombre permet de sortir de plusieurs boucles emboîtées. Le nombre indique combien de

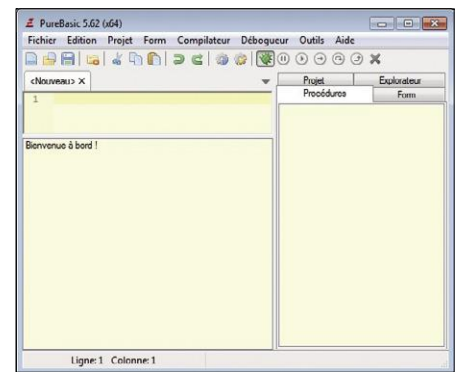


Figure 1. Interface utilisateur de PureBasic.

boucles on veut quitter (ici : 2).

```
For i=1 to 10
    For j=8 to 20
        If i=5
            Break 2
        EndIf
    Next
Next
```

Comme on le voit, il faut donc prendre garde à diverses choses. La lecture approfondie de l'aide en ligne ou du manuel de référence peut paraître ennuyeuse, mais est fortement recommandée.

Mon premier programme

Après le démarrage, l'environnement de développement intégré (EDI) se présente comme sur la **figure 1**. Le grand espace vide est réservé au code source ; en dessous apparaissent les messages système. Dans la zone de gauche, on peut choisir d'afficher la liste de toutes les procédures du programme en cours ou d'un projet, les commandes disponibles pour concevoir des fenêtres utilisateur ou un

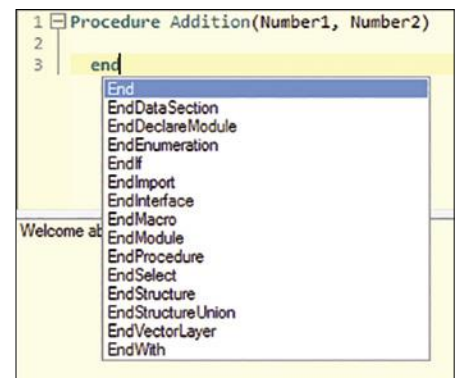


Figure 2. Auto-complétion et coloration syntaxique.

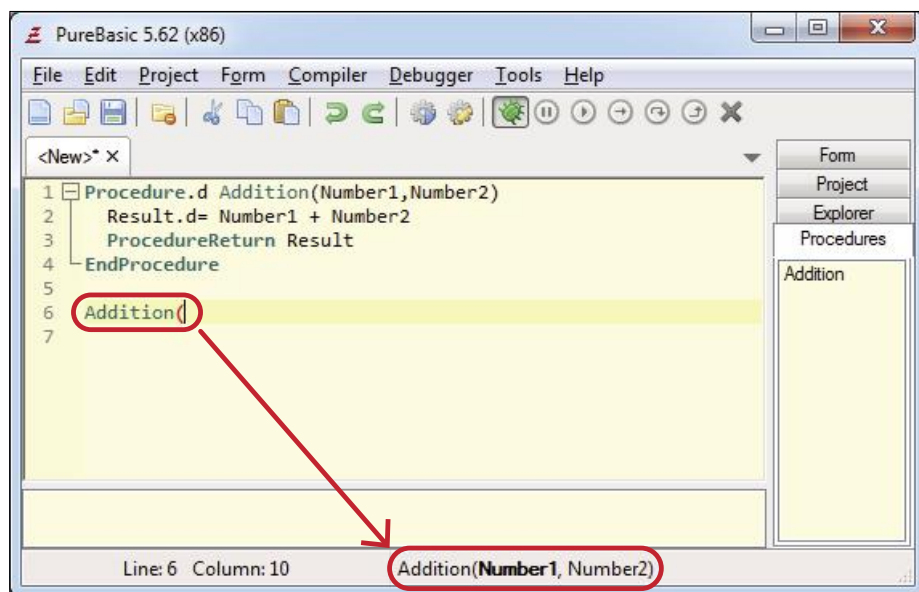


Figure 3. Auto-complétion : affichage des paramètres.

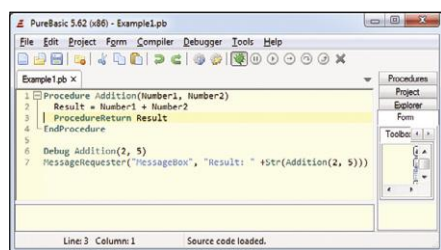


Figure 4. Le programme terminé.

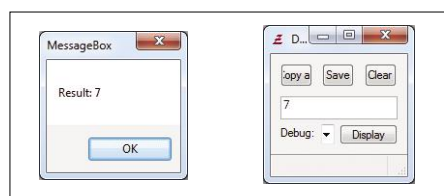


Figure 5. Résultat de l'exécution du programme, affiché dans la lucarne de messages et dans la fenêtre du débogueur.

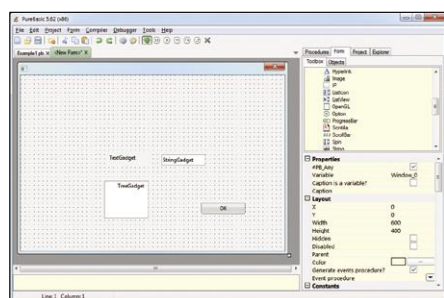


Figure 6. Réalisation d'une interface utilisateur graphique.

petit explorateur de fichiers. Dans la barre d'outils se trouve l'icône activée du scarabée vert du débogueur qui ne se laisse désactiver que dans la version complète, pour obtenir des exécutables encore plus compacts, débarrassés des informations additionnelles à l'usage du débogueur.

Bien entendu, l'éditeur possède les fonctions d'auto-complétion et de coloration syntaxique. Les procédures peuvent être repliées (**fig. 2**) de telle sorte que seule la déclaration soit visible. Notre premier programme ne fait rien d'autre que la somme de deux nombres pour l'afficher :

```
Procedure Addition(Number1,
    Number2)
    Result.d= Number1 + Number2
    ProcedureReturn
EndProcedure
```

Bon, ça ne va pas loin et ça n'a guère d'intérêt, ça ne sert qu'à se frotter un peu à la syntaxe du PureBasic. Si l'on saisit une commande ou une procédure à laquelle on doit passer des paramètres, dès qu'on a tapé la parenthèse d'ouverture, la liste des paramètres attendus apparaît dans la barre d'état. Dès que notre procédure « Addition » a été définie, cette aide apparaît aussi pour elle (**fig. 3**).

La fenêtre de programmation de la **figure 4** montre le programme complet, une procédure « Addition » qui doit retourner la somme de deux nombres.

Comme aucun type de variable n'est indiqué, le type *Integer* (.i) est utilisé par défaut pour toutes les variables. Les deux lignes suivantes présentent deux possibilités d'afficher le résultat. La ligne 7 ouvre une fenêtre de débogage (ne fonctionne que si le débogueur est actif), la ligne 8 ouvre une lucarne de messages. Le programme contient un bogue : comme les deux nombres sont du type *Integer*, leur somme doit pouvoir prendre une valeur plus grande que ce type. Le type *Quad* (.q) semble convenir. Et voilà qu'un piège guette le candide programmeur amateur : dans le choix du type de donnée du résultat, il faut tenir compte du fait que les programmes peuvent être compilés sur des systèmes 32 ou 64 bits. Auriez-vous imaginé que sur les systèmes à 64 bits une variable de type *Integer* occupe 8 octets, autant que le type *Quad* ? Pour le type du résultat de l'addition, *Float* ou *Double* seraient donc mieux indiqués.

Nous n'allons pas corriger ce bogue, mais démarrons immédiatement une exécution de test par « Compileur → Compiler/Exécuter » ou bien en cliquant sur le bouton avec la flèche bleue. PureBasic ne relève pas le bogue, mais convertit le code BASIC en assembleur. Le compilateur-assembleur intégré le transforme en un fichier .exe. Tout cela se passe à l'arrière-plan. Un court instant plus tard apparaissent la lucarne de messages ainsi que la fenêtre de débogage avec le résultat de l'addition (**fig. 5**).

Il ne nous reste qu'à enregistrer le programme avec « Fichier → Enregistrer ». PureBasic ne crée pas automatiquement les extensions de fichier. On a les extensions valides suivantes :

- *.pb – pour le code source
- *.pbi – pour les bibliothèques de fonctions non compilées (PureBasic-Include)
- *.pbf – pour l'interface graphique utilisateur (PureBasic-Forms)
- *.pbp – pour les fichiers de projet (PureBasic-Project)

Un projet est une collection de plusieurs fichiers de programme. Si vous n'arrivez pas à retrouver votre code source, vous l'avez sans doute enregistré dans un fichier sans extension.

L'interface graphique utilisateur
Pour réaliser une interface graphique uti-

lisateur (GUI), on choisit tout simplement « Form → Nouveau Form » dans la barre de menu. Une fenêtre vide apparaît, avec à droite une liste de gadgets avec leurs propriétés. On peut alors placer librement les gadgets dans la fenêtre (fig. 6). Certaines propriétés des gadgets ne sont affichées correctement qu'au moment de l'exécution du programme, par exemple les modifications de polices de caractères. On obtient l'affichage du temps d'exécution en cliquant sur son symbole avec la flèche bleue.

Ce concepteur visuel d'interface crée en arrière-plan du code source BASIC qui peut être affiché avec « Form → Changer vue code/design ». La structure de base constituée de gadgets est liée aux procédures de traitement des événements. Le fichier résultant doit être enregistré avec l'extension .pbf. On placera son code personnel dans un autre fichier, avec l'extension .pb, pour que le concepteur d'interface ne puisse pas l'écraser. Pour des raisons de place, il n'est pas possible de donner un exemple ici, mais il existe un excellent tutoriel vidéo sur YouTube [2], que je recommande vivement à tous les intéressés germanophones.

L'exemple COMTest

Sur la page [3] du site web Elektor se trouve un exemple de programme appelé « COMTest.zip ». Il montre comment programmer les interfaces utilisateur et accéder au port série. Le tout ne devient toutefois vraiment compréhensible qu'après avoir regardé le tutoriel vidéo sur YouTube.

Pour le transfert de données, beaucoup d'appareils de mesure passent par un port série virtuel (port COM). Si vous travaillez encore avec des ports RS232, vous vous demandez toujours, pour chaque nouvel appareil, si les connecteurs du câble sont correctement câblés. Il faut donc s'assurer que la liaison fonctionne correctement avant de se mettre à programmer. C'est dans ce but que « COMTest » a été écrit, pour avoir une utilité au-delà du simple programme de démonstration.

Compilation

- Extraire les fichiers de l'archive ZIP dans un répertoire quelconque
- Démarrer PureBasic et charger le fichier « COMTest.pb »
- Dans le menu, choisir « Compilateur → Créer un exécutable... »

- Après indication du répertoire cible, le programme est compilé. Il est ensuite immédiatement exécutable. La taille du fichier exécutable produit n'est que de 60,5 Ko. Copiez-le sur une clé USB et essayez-le sur différents PC. Vous verrez que ce petit fichier .exe représente une application Windows opérationnelle qui n'inclut pas de gigantesques bibliothèques additionnelles.

Fonctionnement

Après le démarrage, le programme recherche les ports série disponibles et les affiche dans la lucarne « Com Port » (fig. 7). Il faut commencer par saisir les paramètres de communication puis, dans « Data to send », une commande à envoyer à l'appareil de mesure pour lui demander une réponse. « Send » envoie la commande, « Receive » affiche la réponse dans le champ « Received data ».

Par exemple, la commande préprogrammée « :READ? » demande à un multimètre de table 34401A d'Agilent de renvoyer la valeur de mesure courante.

Extensions

PureBasic peut être étendu avec certains programmes (comme le fichier .exe produit dans l'exemple), qui se retrouvent intégrés dans son interface opérateur (fig. 8). On peut ensuite démarrer l'extension à partir du menu Outils. On peut aussi spécifier des déclencheurs (conditions ou événements) qui pro-

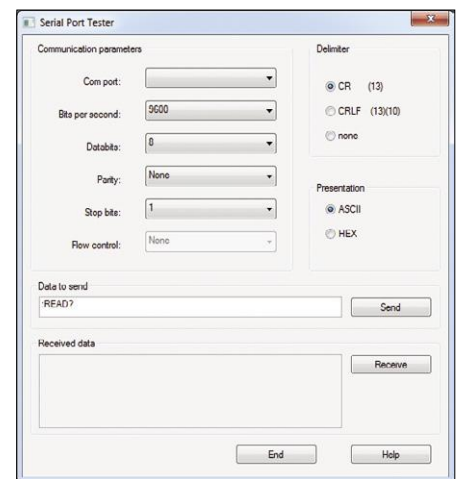


Figure 7. L'interface graphique de COMTest.

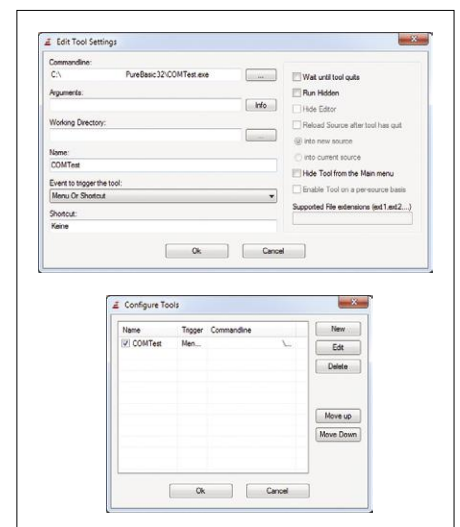


Figure 8. PureBasic peut être étendu avec des outils externes.

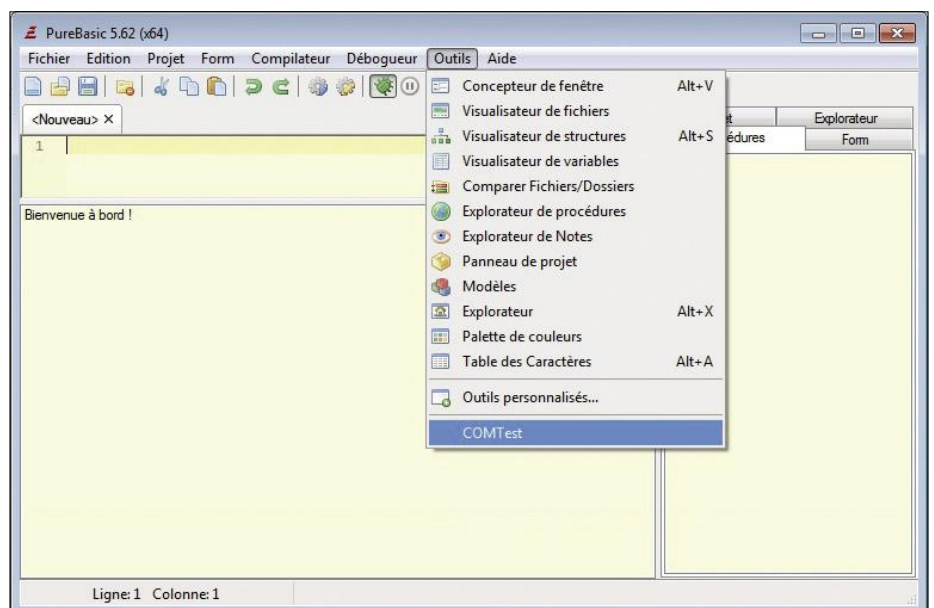


Figure 9. COMTest apparaît comme une rubrique du menu Outils.

Prix et versions

Sur le site web de PureBasic, on peut télécharger une version de démonstration gratuite. Elle est entièrement opérationnelle, mais bridée à 800 lignes de code. De plus, on ne peut ni créer ni utiliser des DLL. La version complète coûte 79 € et comprend des versions pour tous les systèmes d'exploitation : Windows, Linux et MacOS X. Elle inclut également des mises à jour gratuites à vie. La version pour AmigaOS est devenue à code source ouvert et n'est plus maintenue.

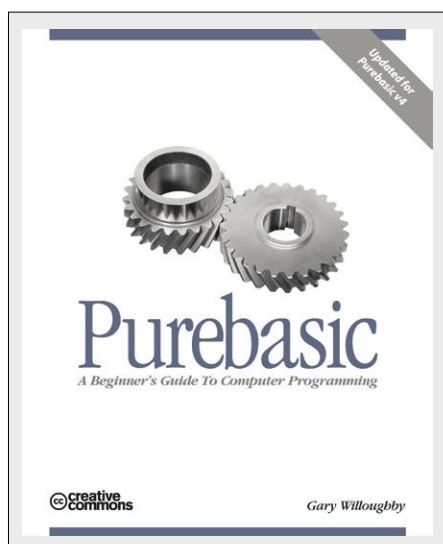
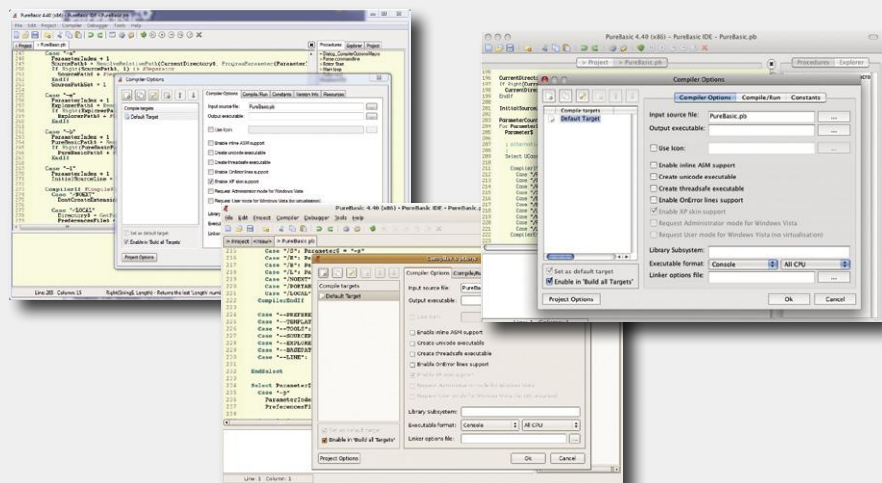


Figure 10. Claire et détaillée : l'introduction de Gary Willoughby.

voqueront le démarrage de l'outil, par exemple celui de l'éditeur.

Déclarer un outil externe :

- Choisir « Outils → Configurer les outils » dans le menu.
- Cliquer sur « Nouveau » pour faire apparaître une nouvelle fenêtre.
- Saisir le chemin et le nom du fichier de l'outil dans le champ « Ligne de commande ».
- Donner un nom à l'outil (« COMTest »).
- Après fermeture des deux fenêtres avec « OK », le nouvel outil apparaît dans le menu Outils (**fig. 9**).

Conclusion

La programmation d'interfaces utilisateur peut paraître un peu laborieuse. Mais PureBasic est un compilateur BASIC très riche, à la syntaxe simple et qui produit du code compact et très rapide. Les programmes compilés ne nécessitent aucune

bibliothèque additionnelle – un BASIC sans surcharge !

Ses possibilités d'accès aux périphériques externes le rendent intéressant pour les électroniciens-développeurs. Si l'on considère que les mises à jour sont gratuites, le prix de la version complète est très raisonnable.

Pour s'informer au préalable, on peut télécharger le manuel de référence (version française 5.60, 2017, 2104 pages) [4] ou bien le livre de Gary Willoughby « Purebasic – A Beginner's Guide To Computer Programming » (version anglaise, 2006, 336 pages) [5] (**fig. 10**).

Une communauté d'utilisateurs très active dans des forums [6] vous aidera dans vos petits ou grands problèmes (mais gare, si vous êtes démasqué comme hacker russe ;).

(180422-02 - 150839 -
version française : Helmut Müller)

Liens

- [1] PureBasic-Homepage en anglais, allemand, français : www.purebasic.com
- [2] Tutoriel du concepteur d'interface (en allemand) : www.youtube.com/watch?v=_Zb8gFIwzpg (partie 1), www.youtube.com/watch?v=ucN0TYIJxcw (partie 2)
- [3] COMTest sur la page du projet : www.elektormagazine.fr/180422-02
- [4] Manuel de référence : www.purebasic.com/french/documentation/PureBasic.pdf
- [5] Introduction : www.purearea.net/pb/download/PureBasicBook.pdf
- [6] PureBasic - site de support en anglais et allemand : www.purearea.net

@ WWW.ELEKTOR.FR

→ Livre en anglais :

Visual Basic for Electronics

Engineering Applications: 5.0, 6.0, Vba, .Net, 2005

www.elektor.fr/vb-engineering-apps



projet 2.0

corrections, mises à jour et courrier des lecteurs

poste de soudage de CMS compact

01-02/2019, p.14 (180348-03)

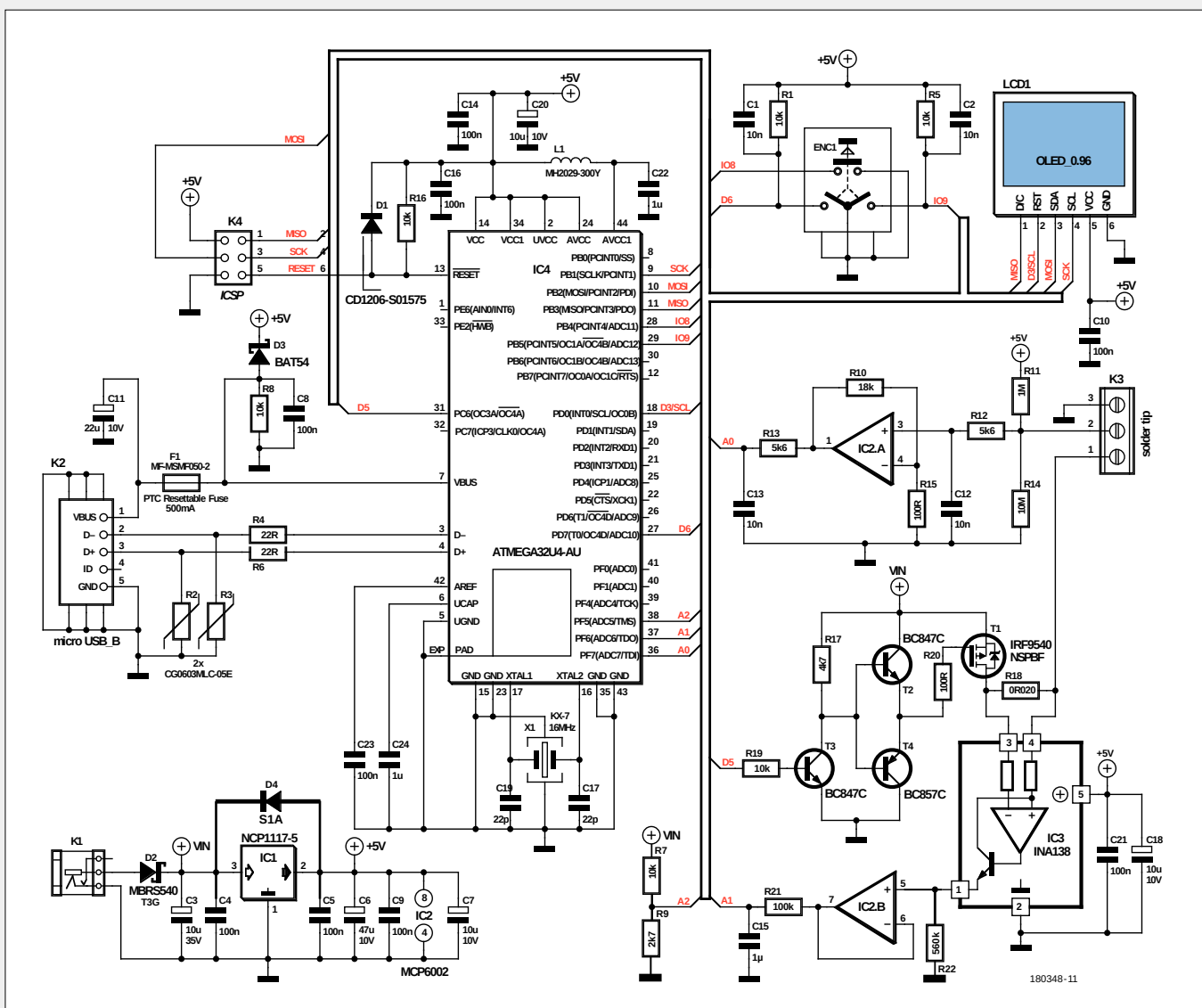
Le schéma comporte quelques erreurs :

- T4 doit être un transistor de type PNP ; son émetteur est connecté à l'émetteur de T2.
- Sur le schéma, il manque R22 (560 kΩ) à la sortie d'IC3, vers le 0 V.
- D4, une diode de roue libre pour IC1, est absente du schéma.
- Les résistances R7 et R9, qui constituent un diviseur de tension pour la mesure de VIN via l'entrée analogique A2 du contrôleur, manquent toutes deux sur le schéma. Actuellement, le logiciel ne s'occupe pas de la tension sur le port A2, mais on peut, en tant qu'utilisateur, ajouter cette fonction. On pourrait alors alimenter le poste de soudage avec une batterie (de voiture) par exemple et éviter une décharge profonde.

Le circuit imprimé est correct.

Un nouveau schéma est disponible gratuitement au téléchargement pour nos lecteurs ; son lien web :

www.elektormagazine.fr/180348-03



détection de coupure de secteur avec alarme SMS

09-10/2018, p.12 (180344-04)

Errata et remarques concernant la liste des composants :

- L'inductance L2 est de type BLM21A-J601SN1D (non pas BLM12...).
- MOD2 : le module GSM de Quectel peut être utilisé dans la version M95FA-03STDN ainsi que la version M95FA-03STN.
- X1 (CSTCC8M00G53-R0, résonateur céramique, Murata 8 MHz) est disponible chez Farnell, TME, Mouser, Distrelec et d'autres distributeurs.
- K2 : le pas du bornier est de 7,62 mm, et non pas de 7,68 mm.



feu de phare 2.0

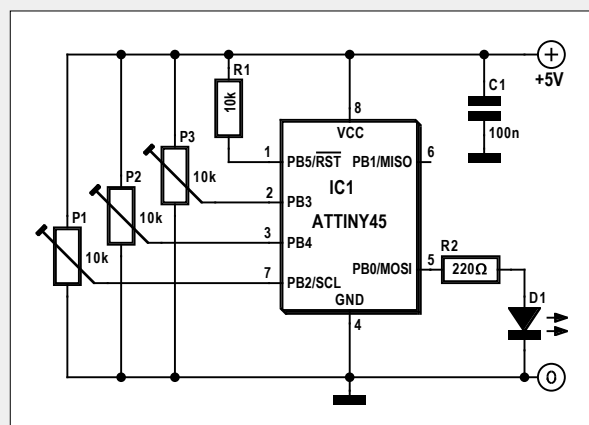
01-02/2019, p.24 (180334-04)

Un test de longue durée a montré qu'il pouvait y avoir des pannes logicielles après une période indéterminée. Un examen approfondi a révélé que le problème est lié à l'acquisition des valeurs du potentiomètre dans la boucle DO-LOOP. Dans une nouvelle version du logiciel [cf. www.elektormagazine.fr/180334-04], un déplacement de la boucle DO-LOOP fait que les valeurs du potentiomètre ne sont demandées qu'une seule fois au démarrage du programme. Le circuit peut, le cas échéant, être doté d'une réinitialisation (reset).

Le réglage s'effectue maintenant comme suit :

on commence par mettre les trois potentiomètres à leur valeur minimale. Il faut d'abord régler avec P1 la longueur d'éclat souhaitée. Ensuite, on règle le nombre d'éclats/clignotements avec P3. La durée du cycle de la signature lumineuse est encore minimale à ce moment, et le programme la prolonge automatiquement de deux secondes de plus que ne l'exigent les éclats et les interruptions qui l'accompagnent. Lorsque le nombre d'éclats et leur durée sont fixés, on peut régler la durée totale de la signature lumineuse avec le potentiomètre P2. Les paramètres du phare sont ainsi réglés. Pour que le réglage effectué soit pris en compte, il faut appuyer sur la touche de réinitialisation pour redémarrer le programme. On peut aussi couper l'alimentation et remettre le circuit sous tension.

Friedrich Lischek (auteur)



mise à jour pour le réveil à triple affichage

11-12/2018, p.88 (170112-B-04)

Dans le numéro de novembre/décembre 2018, j'ai lu dans deux projets comment calculer minutieusement les sinus et cosinus (projets « réveil à triple affichage » et « synthèse numérique directe (DDS) sur Arduino »).

Dans mes programmes, j'utilise souvent la fonction arc tangente, le tout fonctionne sur un petit μC à 8 bits sans débauche de MHz et de mémoire. Pour le calcul, j'utilise l'algorithme CORDIC, qui, mathématiquement, paraît assez compliqué, mais qui est très compréhensible lorsqu'il est codé en C (avec des entiers sur 16 bits).

Le même algorithme peut être utilisé pour sinus/cosinus. Il faut une LUT (Look Up Table, table de correspondance) avec N entrées dans laquelle on précalcule $\arctan(1/2^i)$, $i = 0 \dots N-1$. Pendant N itérations, on fait tourner un vecteur dans le sens horaire ou antihoraire, avec $\arctan(1/2^i)$, jusqu'à ce que le vecteur atteigne approximativement l'angle désiré. La précision et la vitesse dépendent directement de N. Cela ne requiert que des additions et des décalages; tous les nombres (angle, sin, cos, etc.) sont interprétés comme des entiers.

En gros, l'algorithme CORDIC pour le sinus/cosinus ressemble à ceci :

Ce qui manque ici, outre la LUT, ce sont les valeurs de départ pour `sin`, `cos` et `angle`. Cela dépend de la situation concrète.

CORDIC est, soit dit en passant, capable de bien plus ; les informations disponibles sur le web (Wikipedia, etc.) sont relativement mathématiques et (pour les électroniciens) quasi incompréhensibles. Ce que j'ai trouvé de meilleur pour la version sinus/cosinus se trouve ici :

<http://nonagon.org/ExLibris/sites/default/files/pdf/Cordic-Faster-Sin-Cos.pdf>

La version similaire d'`atan2` décrite ici est plus compréhensible :

http://bsvi.ru/uploads/CORDIC--_10EBA/cordic.pdf

Bien sûr, la LUT dans l'article du réveil a aussi des avantages (vitesse). Mais elle requiert beaucoup de mémoire et n'est pas très flexible. En outre, les fonctions `sin()`/`cos()` de la bibliothèque `math.h` sont très lentes et gourmandes en ressources (points flottants). Au passage, merci pour les excellents articles, il y a toujours quelque chose de nouveau à découvrir !

Andreas Dick

```
for (i=0; i
{
    if (angle < 0)
    {
        int temp = sin;
        sin -= cos >> i;
        cos += temp >> i;
        angle += lut[i];
    }
    else
    {
        int temp = sin;
        sin += cos >> i;
        cos -= temp >> i;
        angle -= lut[i];
    }
}
```



testeur de transistors Platino

03/2015, p.62 (130544)

Le testeur de transistors Platino d'Elektor (03/2015) est un projet déjà ancien, mais je pense que rien ne s'oppose à des suggestions visant à améliorer des projets d'un certain âge. En tout état de cause, le circuit imprimé est toujours disponible dans l'e-shoppe d'Elektor et je l'ai acheté il y a près de deux ans. La construction ne présente aucune difficulté majeure, le seul problème semble se trouver dans le logiciel. Si on connecte un transistor, la mesure se fait sans problème. Cependant, si les bornes sont en l'air (sans DUT), il n'y a pas d'information de retour. Il semble, qu'après une pression du bouton, il se passe quelque chose sur la première ligne de l'écran, mais seule une pression permanente du bouton fait apparaître un `No transistor` pâlot.

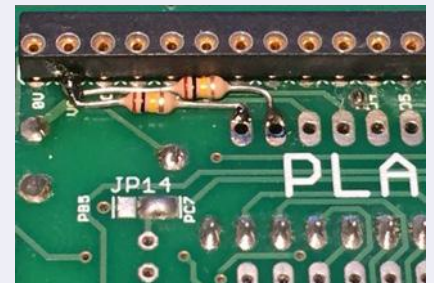
J'ai donc apporté une amélioration avec ma version complète de BASCOM : après la ligne 320 `Lcd "No transistor"`, j'ai inséré la ligne `Waitms 1000` et voici que l'affichage `No transistor` reste lisible. Cependant, les valeurs obtenues lors de la mesure précédente avec connexion d'un transistor s'affichent à nouveau. Il faut donc, après insertion de la nouvelle ligne, supprimer les variables correspondantes :

```
321: Waitms 1000
322: I = 0
323: T_type = 0
324: Cal_pin = 0
```

Super, tout marche !

Comme j'ai découvert dans le code source que les valeurs mesurées sont également envoyées vers l'interface série (malheureusement l'article effleure ce sujet), j'ai eu une idée. J'ai connecté un module USB-série et démarré une fenêtre de terminal. Il m'a fallu un certain temps pour réaliser que la vitesse du port série du terminal était mal réglée ; il manquait simplement dans la ligne 14 du code source un `baud = 9600`. J'ai recompilé. Tout semblait fonctionner, jusqu'à ce que j'enlève le câble entre l'interface série et le PC : un nouveau test du testeur de transistors montre que ce dernier ne réagit plus. Je me suis alors souvenu d'une suggestion de Clemens Valens, qui, lorsqu'il m'a dépanné il y a deux ans, suggérait un « port série flottant ». J'ai donc relié PD0 (TXD) et PD1 (RXD) de l'ATmega32 à VCC, via deux résistances de 100 kΩ. Et voilà : tout fonctionne parfaitement.

Hans Schneider



Note de la rédaction : la version du logiciel de Hans Schneider (V3.1) peut être téléchargée dans les commentaires de la page : www.elektormagazine.fr/labs/130544-1-platino-transistor-tester

(180573-B-04 – version française : Guy Raedersdorf)



mesurer, alimenter et imprimer...

...avec des appareils de l'e-choppe d'Elektor

Multimètre avec module Bluetooth

Qu'est-ce qui est indispensable sur l'établi d'un électronicien (amateur) ? Un multimètre et un fer à souder. La plupart d'entre nous disposent déjà d'un multimètre, mais il faut parfois remplacer un appareil usagé ou en acquérir un second pour effectuer plusieurs mesures simultanées.



Les multimètres, ce n'est pas ça qui manque dans les magasins, du modèle à 5 € chez le quincaillier du coin à l'instrument de précision hors de prix. Toutefois le bon sens nous dicte de ne pas nécessairement acheter le moins cher.

Les caractéristiques du MS8238H de Mastech sont dans la moyenne, avec toutes les fonctions de base que l'on peut s'attendre à trouver sur un tel appareil : mesure de tensions et courants continus et alternatifs, mesure de résistances, test de diode et de continuité (bipeur). On peut aussi mesurer la température, à l'aide d'un thermocouple fourni avec l'appareil, tester des batteries (9 et 1,5 V), et détecter des tensions (> 100 V CA) sans contact en position NCV. Certaines positions du sélecteur rotatif ont plusieurs fonctions, sélectionnées à l'aide du bouton FUNC. HOLD gèle l'affichage, et le bouton RAN permet la sélection manuelle de la gamme de mesure, sélection qui est normalement automatique.

Il y a encore le bouton bleu Link, pour la mise en fonction du module Bluetooth 4 intégré. La valeur mesurée et la position du sélecteur rotatif sont transmises à une application spécifique sur une tablette ou un téléphone portable. Ce type de fonction n'est certainement pas courant sur un appareil de ce prix ! L'affichage se fait sur 3 chiffres $\frac{3}{4}$. Il n'y a malheureusement pas de rétroéclairage, mais l'écran reste raisonnablement lisible avec une faible luminosité ambiante. La précision dans cette gamme de prix est habituellement de 0,5 % pour les mesures de tensions continues. Les tensions et courants alternatifs sont mesurés en valeur efficace vraie (*true RMS*), ce qui est aussi assez rare.



@ WWW.ELEKTOR.FR

→ Multimètre Mastech MS8238H avec module Bluetooth
<https://bit.ly/2H2G8v4>

Alimentation enfichable pour Raspberry Pi

Les modules d'alimentation enfichables pour Raspberry Pi sont légion. Toutefois, l'alimentation StromPi 3 présente quelques caractéristiques uniques. Elle est fournie avec deux entretoises, deux vis et deux écrous, ainsi qu'une notice succincte en allemand et en anglais. Grâce à ces instructions, le montage est loin d'être insurmontable.

La StromPi V3 prend également en charge la *StromPi V3 Battery HAT*, un accu LiFePo4 de 1000 mAh, tension nominale de 3,2 V (3,2 Wh). Cette extension HAT s'enfiche simplement sur la StromPi V3 (hors tension, cela va de soi) et peut alors alimenter le Raspberry Pi.

La StromPi V3 met à disposition du Raspberry Pi plusieurs entrées de tension, l'une ayant une gamme couvrant de 6 à 61 V, un port USB avec un condensateur tampon de 1 F et un connecteur pour l'accu LiFePo4. L'extension HAT fournit en outre une horloge en temps réel, mise à disposition du RPi pendant les coupures de courant et les pertes de réseau. Cela fait de la StromPi V3 le compagnon idéal quand il s'agit d'alimenter le Raspberry au moyen d'un réseau type en 12, 24 ou 48 V CC.



Si une source alimentant la StromPi V3 fait défaut, il est possible de régler dans quel ordre se replier sur les alimentations restantes. Pour empiler d'autres HAT, les connecteurs sont dupliqués ainsi que les ports d'entrée PoE du RPi 3B+. Les prises USB A et les deux micro-USB apparaissent sur le côté. La StromPi V3 peut ainsi alimenter des appareils consommant plus de 500 mA, comme des disques durs. La StromPi V3 est spécifiquement conçue pour le Raspberry Pi, mais grâce à sa construction, elle fonctionne également avec d'autres nano-ordinateurs mono-cartes comme le BeagleBoneBlack. La StromPi V3 est à la fois une alimentation universelle et une alimentation secourue pour Raspberry Pi. Le matériel été très bien conçu.



Imprimante 3D Anycubic Photon

Cette imprimante qui utilise le procédé *Digital Light Processing* (DLP) imprime des objets avec de la résine liquide. Un faisceau ultraviolet projette une image sous le bain transparent de résine et l'objet est construit couche par couche sous une plateforme qui s'élève petit à petit. Si l'on compare avec une imprimante 3D FDM (*Fused Deposition Modeling*), où un extrudeur chauffé dépose un filament de plastique fondu pour obtenir une forme donnée, il apparaît que ce procédé-ci est plus simple. Il y a peu de pièces mobiles et seul l'axe vertical bouge.

La procédure est on ne peut plus simple. Vous chargez l'objet conçu par vos propres soins, ou un fichier STL, dans le logiciel de trancheuse fourni avec l'appareil (*Photon slicer*), puis dans l'imprimante avec une clé USB ou un câble relié à l'ordinateur. Il suffit ensuite de mettre l'imprimante de niveau, de régler la hauteur de la plateforme, de remplir un quart du bac avec de la résine et de lancer l'impression.

Un des inconvénients est que la résine (disponible en différentes couleurs) est assez chère.

Le rapport qualité/prix de cette imprimante en fait sans nul doute un des meilleurs choix possible sur le marché actuel des imprimantes 3D. ◀

180689-B-04

Tested by
elektorlabs



@ WWW.ELEKTOR.FR

→ StromPi 3 – alimentation enfichable pour Raspberry Pi,
www.elektor.fr/strompi-3-power-solution-for-raspberry-pi



@ WWW.ELEKTOR.FR

→ Imprimante 3D Anycubic Photon
www.elektor.fr/anycubic-photon-uv-resin-dlp-3d-printer

conception matérielle avec (V)HDL (3)

commande d'une matrice de LED WS2812

Jörg Zollmann

Dans le précédent article de cette série, nous avons construit une horloge numérique en format binaire. Cette fois-ci, nous allons doter l'horloge d'un afficheur constitué d'une matrice de 64 LED RVB de type WS2812. Par bonheur, le protocole utilisé pour commander ces LED est un protocole asynchrone très simple. Pour notre pilote matériel en VHDL, nous utiliserons, entre autres, un automate à états finis.

On peut trouver chez les marchands en ligne une matrice constituée de 64 LED WS2812-RVB à partir de 5 €. Une WS2812 est plus qu'une simple LED RVB. Dans un petit boîtier se trouvent, outre les trois LED RVB, leurs pilotes (sources de courant) et un petit contrôleur. Au moyen d'un protocole sur un fil, on peut donner à ce contrôleur une consigne d'intensité lumineuse pour chaque LED. Le web regorge de projets à microcontrôleurs (et de bibliothèques de logiciel) utilisant des WS2812 [1][2]. Nous allons approfondir la question en « programmant » un pilote en VHDL qui communique avec les contrôleurs intégrés dans les LED.

Comme pour les précédents articles de cette série, le logiciel est disponible au téléchargement [7]. L'organisation du code et la plateforme matérielle utilisée ont été décrites dans le premier article [4].

Le protocole WS2812

Pour créer un pilote matériel en VHDL qui doit communiquer avec une puce numérique externe, on doit être familiarisé avec les détails internes du composant à commander et avoir une connaissance détaillée du protocole utilisé. Pour une LED WS2812, on a la chance que ce soit un protocole asynchrone relativement simple. Les LED sont reliées entre elles sur une chaîne : chacune récupère 24 bits de données sur son port d'entrée DIN et passe les bits restants à la LED suivante, raccordée à son port de sortie DOUT [3].

Pour chacune des trois couleurs, on utilise une valeur sur 8 bits comprise entre 0 et 255. Cette valeur représente l'intensité du courant fourni par le pilote intégré correspondant. Après réception des 24 premiers bits, qui sont stockés localement, les bits suivants sont rafraîchis et émis sur la sortie DOUT. Ce reconditionnement du signal de sortie assure une transmission robuste permettant la mise en série d'une quantité de diodes théoriquement illimitée. La limite ne dépend que de la consommation électrique et de la fréquence de renouvellement de la trame.

Le stockage des valeurs reçues n'intervient qu'après la réception d'un code d'initialisation, à savoir la mise au niveau bas de la ligne pendant plus de 50 µs. Le codage des bits de données est réalisé avec des impulsions de niveaux haut et bas de durées différentes (voir le **tableau 1** et le **figure 1**). La trame pour un bit dure 1,25 µs. Si pendant ce temps une impulsion de niveau haut de 0,35 µs est suivie d'une impulsion au niveau bas de 0,9 µs, cela correspond à un bit à 0. Pour un bit à 1, on permute les durées des deux impulsions. Les données reçues sont codées dans l'ordre vert, rouge, bleu et commencent par les bits de poids forts.

À une fréquence d'horloge de 50 MHz, la résolution temporelle accessible avec la carte CPLD est de 20 ns. Le protocole

Tableau 1. Temps de transfert de données.		
Nom	Description	Durée (TH+TL = 1,25 µs ± 150 ns)
T0H	code 0, durée du niveau haut	0,35 µs
T1H	code 1, durée du niveau haut	0,9 µs
T0L	code 0, durée du niveau bas	0,9 µs
T1L	code 1, durée du niveau bas	0,35 µs
RES	réinitialisation, durée du niveau bas	> 50 µs

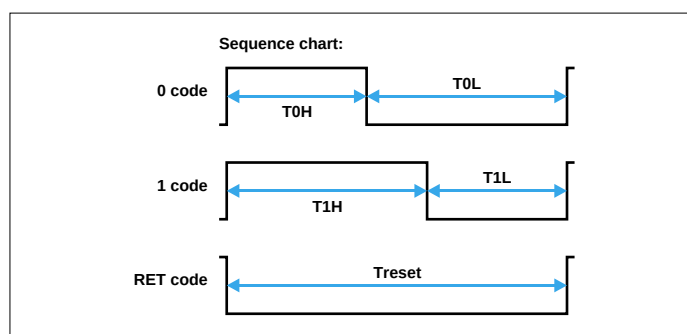


Figure 1. Format des données WS2812 [3].

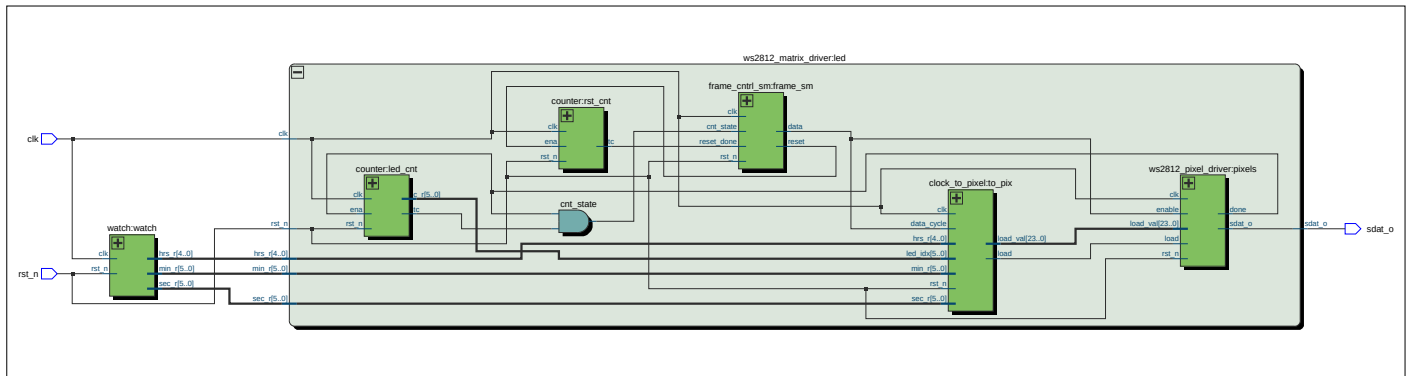


Figure 2. Horloge à WS2812 dans la vue RTL.

WS2812 exige que les durées des impulsions s'écartent des valeurs nominales de moins de ± 150 ns, ce que notre réalisation purement VHDL devrait donc pouvoir respecter. L'expérience avec les LED WS2812 montre que ces LED tolèrent relativement bien des écarts par rapport au protocole. Toutefois, les impulsions de niveau bas ne doivent pas se prolonger. Même celles de durée sensiblement inférieure à 50 μ s ont de bonnes chances d'être prises pour des réinitialisations.

Pilote de pixels

Le pilote de WS2812 d'affichage de l'heure en format binaire présenté ici a une structure hiérarchisée composée de plusieurs modules (**fig. 2**). Le module de niveau supérieur instancie à la fois l'horloge de l'article précédent [8] et le module *ws2812_matrix_driver*. Les signaux d'entrée de ce module sont les données du registre horaire (heures, minutes, secondes) qui doivent apparaître sur l'afficheur 8x8. Le module pilote de matrice contient encore d'autres composants qui exécutent des tâches diverses.

Le composant *ws2812_pixel_driver* est conçu pour pouvoir être utilisé aussi pour des rubans à LED ou des matrices de plus grande taille (ce qui offre un espace pour des expérimentations personnelles). On peut décrire la fonction du pilote de pixel de

la manière suivante : au début d'un pixel, un registre à décalage à 24 bits (*do_reg*) est chargé avec les valeurs de luminosité des trois LED. Pour cela, le signal *load* doit passer à un pendant un top d'horloge pour que le signal d'entrée *load_val* soit copié dans le registre à décalage. Ce contenu doit ensuite être émis sous forme sérielle (**listage 1**).

Le protocole exige toutefois des impulsions plutôt que des valeurs statiques 0 ou 1 sur la ligne de données. Ceci est réalisé par deux processus. Le processus *code_proc* produit en permanence des « codes » pour un 0 ou un 1 (**listage 2**). Le bit de poids le plus fort (bit 23) du registre à décalage, suivi des autres bits, détermine lequel de ces codes apparaît sur la

Listage 1. Registre à décalage.

```
shift_proc: process (rst_n, clk)
begin
    if (rst_n = '0') then
        do_reg <= (others => '0');
    elsif rising_edge(clk) then
        if(enable= '1') then
            if (load = '1') then
                do_reg <= load_val;
            else
                if (bit_done) then
                    do_reg <= do_reg(22 downto 0) & '0';
                end if;
            end if;
        end if;
    end if;
end process;
```

Listage 2. Processus de choix du code Zéro ou Un.

```
code_proc: process (rst_n, clk)
begin
    if (rst_n = '0') then
        zero_one_count <= 0;
        one_code <= '1';
        zero_code <= '1';
    elsif rising_edge(clk) then
        if (enable = '1') then
            if (bit_done) then
                zero_one_count <= 0;
                one_code <= '1';
                zero_code <= '1';
            else
                zero_one_count <= zero_one_count + 1;
                if (zero_one_count >= c_CLKS_T0_H_T1_L)
                then
                    zero_code <= '0';
                end if;
                if (zero_one_count >= c_CLKS_T1_H_T0_L)
                then
                    one_code <= '0';
                end if;
            end if;
        end if;
    end if;
end process;
```

Listage 3. Multiplexeur de sortie `sdato`.

```
mux : process (all) is
begin
  case (out_sel) is --
    when "00" => sdato_o <= '0';
    when "01" => sdato_o <= zero_code;
    when "10" => sdato_o <= '0';
    when "11" => sdato_o <= one_code;
    when others => sdato_o <= '0';
  end case;
end process;
```

sortie. Ce multiplexeur (*Mux*) commute donc la sortie `sdato_o` sur le bon générateur en fonction de la valeur dans le registre à décalage (**listage 3**).

Le reste de la logique dans le pilote de pixels se compose de compteurs, dont nous avons déjà fait la connaissance dans l'article précédent. Les compteurs sont configurés avec des capacités différentes et comptent des événements différents. Il y a un compteur qui indique qu'un bit a été décalé (donc toutes les 1,25 µs), un compteur qui indique qu'un octet a été traité, et enfin un compteur qui avertit de la fin du traitement d'un pixel.

Listage 4. Automate de commande de trame écrit avec la méthode à un processus.

```
-- Build an enumerated type for the state machine
type state_type is (data_cyc, reset_cyc);
-- Register to hold the current state
signal frame_state: state_type;
...
frame_ctrl: process (rst_n, clk)
begin
  if (rst_n = '0') then
    frame_state <= reset_cyc;
  elsif rising_edge(clk) then
    case frame_state is
      when reset_cyc => -- when current state
is reset_cyc
        frame_state <= reset_cyc;
        if(reset_done) then
          frame_state <= data_cyc;
        end if;
      when data_cyc => -- when current state is
data_cyc
        frame_state <= data_cyc;
        if(cnt_state) then
          frame_state <= reset_cyc;
        end if;
      when others => frame_state <= reset_cyc;
    end case;
  end if;
end process;
```

De l'horloge aux pixels

Le pilote `ws2812_matrix_driver` contient encore deux autres compteurs. L'un indique la fin de la durée d'un code de réinitialisation, l'autre compte les pixels à l'instant de leur envoi. Comme les pixels de la matrice sont connectés en série, chacun des 64 pixels est numéroté. La conversion des valeurs des compteurs de l'horloge en valeurs sur 24 bits pour les pixels est effectuée par le module `clock_to_pixel`. Dans le cadre d'un processus synchrone, ce module se charge de remplir le registre à décalage du pilote de pixels d'un flot continu de données. Pour cela, il utilise le numéro du pixel. Si ce numéro n'est pas l'un des numéros des pixels utilisés pour afficher l'heure, la couleur de fond `led_bg` est chargée dans le registre à décalage. L'affichage de l'heure nécessite 4 pixels pour les heures et respectivement 6 pixels pour les minutes et les secondes. Comme la lecture en format binaire exige de toute façon un peu d'entraînement, on a réservé pour chaque affichage une rangée complète de la matrice à LED. Ces rangées sont identifiées par des constantes (exemple : `constant c_hrs_idx: natural := 15;`).

L'affichage des heures commence donc à la troisième rangée de la matrice à LED (deux rangées restent libres). Si l'un des bits de la valeur des heures est à 1, le registre à décalage est chargé avec la couleur jaune (`led_on`), sinon, la LED est éteinte (`led_off`). Les autres LED d'une rangée d'affichage, 5 à 8 ou 7 à 8, sont toujours éteintes. Pour que la comparaison du numéro de pixel fonctionne, il faut encore tenir compte du type des données qui sont comparées. Le signal de sortie du compteur de pixels (`led_cnt`) est de type `unsigned`, les constantes de comparaison sont de type `natural`. Pour que la comparaison soit valide, il faut utiliser la fonction `to_integer` de la bibliothèque `numeric_std` (... `if (to_integer(led_idx) = c_hrs_idx + 0) then` ...) On trouvera une vue d'ensemble des conversions de type intégrées dans VHDL sous [5] et [6].

Automates

Le reste du pilotage est pris en charge par le module `frame_ctrl_sm`, qui représente un automate à états finis. Un tel automate (en anglais : *Finite State Machine*, FSM), est un circuit logique conçu pour piloter des tâches et des séquences récurrentes. Il n'existe guère de matériels numériques de grande taille qui n'utilisent pas un automate à états finis. Un automate se compose d'une partie séquentielle, d'une série de registres qui enregistrent l'état en cours et d'une partie combinatoire qui déduit de l'état en cours l'état suivant ainsi que les valeurs des sorties. Selon l'application, il est souhaitable de synchroniser les entrées au préalable et de tamponner les sorties dans un registre. On distingue deux types d'automates : le Moore et le Mealy. La différence entre les deux tient à leur table de vérité et au passage d'un état à l'autre. Alors que les sorties d'un automate Moore ne dépendent que de son état en cours, celles de l'automate Mealy sont des fonctions de l'état en cours et des valeurs des entrées. On peut passer d'un type à l'autre au moyen de l'algèbre de Boole. En pratique (et selon l'avis de l'auteur), cette distinction n'importe guère, on se concentre plutôt sur le problème concret à résoudre. Par expérience, un automate Moore a davantage d'états (= *mo(o)re*), mais est en général plus facile à comprendre.

Il y a plusieurs méthodes pour décrire un automate en VHDL. La première est celle du manuel, aux deux processus. On y décompose l'automate en deux processus. L'un ne s'occupe que

de mémoriser le nouveau vecteur d'état : c'est la description de la partie séquentielle de l'automate. L'autre est la logique combinatoire nécessaire au calcul de l'état suivant.

Ensuite, il y a la méthode qui consiste à décrire dans un premier processus les parties séquentielle et combinatoire nécessaires au calcul et à la mémorisation de l'état, et dans un second processus la logique résiduelle de gestion des sorties (si nécessaire). Cette méthode a l'avantage d'être d'une écriture plus courte, donc plus compacte et plus facile à comprendre.

Le **listage 4** montre le code source de l'automate qui surveille le déroulement d'une trame dans le pilote. L'automate n'a que deux états qui montrent si un pixel vient juste d'être émis ou s'il s'agit d'un code de réinitialisation. On documente habituellement les états d'un automate au moyen d'un diagramme d'états. Pour les états individuels, on peut créer dans VHDL un type défini par l'utilisateur (*User defined Type*) au moyen du mot-clé **type**. La ligne

```
type state_type is (data_cyc, reset_cyc);
```

crée un nouveau type appelé `state_type` à partir du type standard de VHDL *Enumeration*.

La **figure 3** montre le diagramme d'états de l'automate pour une trame. Modelsim et Quartus sont aussi capables d'afficher un diagramme d'états, pourvu que les automates soient reconnus comme tels. Si pour la synthèse du fichier VHDL avec Modelsim, on utilise l'option (*Command Line Switch*) `+cover`, la plupart des automates sont reconnus comme tels. Chez Quartus, il faut en outre s'en tenir au style de codage préconisé. L'automate du pilote WS2812 n'est toutefois pas reconnu (sans doute parce qu'il ne possède que deux états).

Luminosité

Une recherche sur l'internet indique qu'une seule LED peut consommer jusqu'à 20 mA à pleine puissance. Pour 64×3 LED, on arrive donc à 3,9 A, ce qui suffit pour surcharger un petit bloc d'alimentation. Pour développer le pilote, l'auteur a donc commencé à travailler avec une alimentation séparée, capable de fournir un courant suffisant. Indépendamment du courant beaucoup trop élevé, les LED fonctionnant à pleine puissance

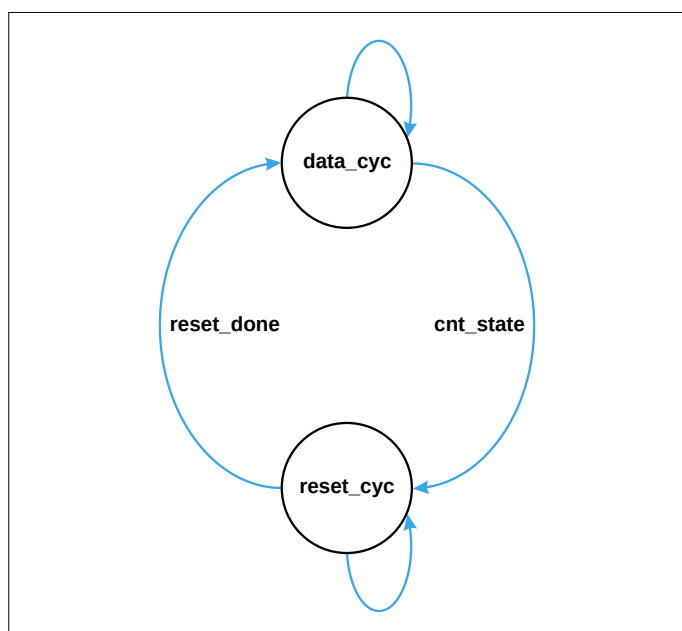


Figure 3. Automate à états finis pour le traitement d'une trame.



Un événement oublié ?

Vous organisez une conférence, un salon... ou bien vous participez à un séminaire ou tout autre événement qui aurait sa place ici, partagez cette information avec tous les lecteurs.

Envoyez-nous tous les détails à redaction@elektor.fr.

avril 2019

◇ Paris Space Week

02 au 03/04 - Paris

www.paris-space-week.com

◇ Rencontres Interrégionales de Sous-Traitance du Sud-Est

02 au 04/04 - Valence

www.rist.org

◇ ENOVA Nantes

03 au 04/04 - Nantes

www.enova-event.com

**Elektor
partenaire presse !**

◇ Salon des seniors

03 au 06/04 - Paris

www.salondesseniors.com

◇ RICV - Rencontres internationales de cerfs-volants

06 au 14/04 - Berck-sur-Mer

www.cerf-volant-berck.com

◇ MIPTV

08 au 11/04 - Cannes

www.miptv.com

◇ 11^e édition de la Journée Française des Tests Logiciels (JFTL) 2019

09/04 - Montrouge

www.cftl.fr/JFTL/accueil

◇ ProDurable (économie durable)

09 au 10/04 - Paris

www.produrable.com

Salon des Acteurs et des Solutions de l'Économie Durable

◇ SIdO - IoT - AI - Robotics

10 au 11/04 - Lyon

www.sido-event.com

◇ Flip Expo 2019

13 au 14/04 - Le Tréport

www.flip-expo.fr

◇ EuroPropre

16 au 18/04 - Paris

www.europropre.com

◇ Passi'Bat (construction passive, positive et durable)

16 au 18/04 - Paris

www.passibat.fr

◇ Concours Lépine

27/04 au 08/05 - Paris

www.concours-lepine.com



sont beaucoup trop lumineuses pour un affichage de l'heure, on a donc réduit leur luminosité. La puissance avec laquelle les LED de chaque couleur éclairent est réglable au moyen de constantes dans le module *clock_to_pixel*. Pour cela, on a créé un type *Array* spécifique peuplé de vecteurs *std_logic*. De cette manière, on peut allouer une constante propre à chaque couleur différente à afficher.

```
type grb_lut is array (0 to 2) of std_logic_vector(7
    downto 0);
constant led_off: grb_lut:= (x"00", x"00", x"00");
    -- éteint
constant led_bg: grb_lut:= (x"00", x"10", x"00");
    -- rouge
constant led_on : grb_lut:= (x"1F", x"1F", x"1F");
    -- jaune
```

de caractéristiques de la matrice de LED, c'est formellement inadmissible puisque la tension minimale du niveau haut doit être de $0,7 * VDD = 3,5 \text{ V}$, ce qui dépasse le niveau de 3,3 V des sorties LVTTTL de 200 mV environ. Selon le type de matrice utilisée, il est donc possible qu'il faille recourir à un adaptateur de niveau.

Perspective

Dans cet article de la série, nous avons considéré pour la première fois un circuit numérique de plus grande taille.

Un coup d'œil au « Fitter Resource Usage Summary » dans la fenêtre du *Quartus Compilation Report* montre que la carte CPLD est occupée à 36 %. Il reste donc un peu de place pour des extensions. Dans le prochain article, nous traiterons brièvement les contraintes de temps (*Timing Constraints*) à prendre en compte pour avoir un circuit qui fonctionne effectivement. Ensuite, nous nous lancerons dans la programmation d'un petit jeu matériel. ◀

(180285-C-04 – version française : Helmut Müller)

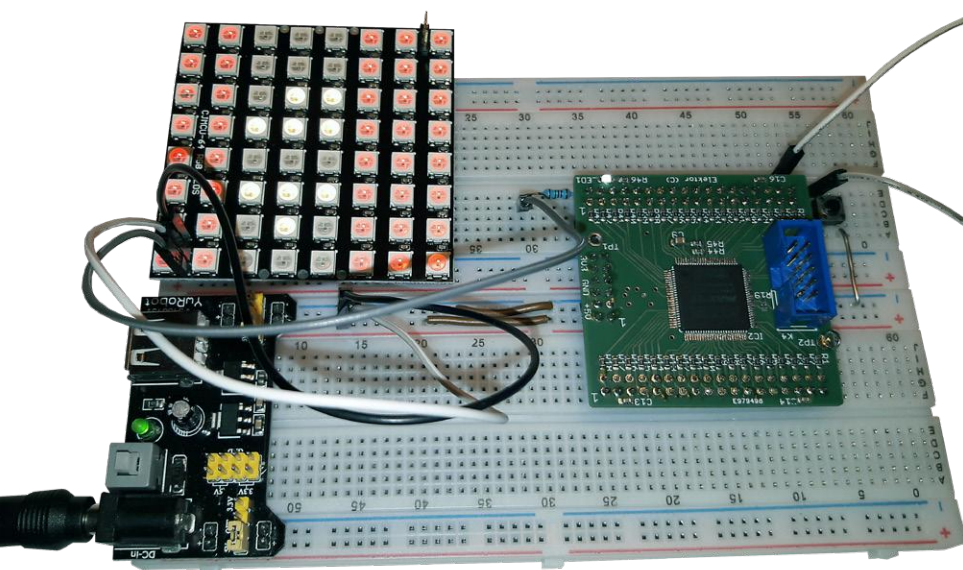


Figure 4. Horloge à affichage 8x8.

La valeur 31 (= 0x1F) choisie comme standard est bien suffisante et permet de se contenter d'un courant de 700 mA fourni par l'alimentation sur la platine d'essai (**fig. 4**). La ligne série de transmission des données aux LED de la matrice est directement connectée à la sortie de la carte CPLD (broche 54 de la carte CPLD = broche 5 sur K1). Selon la feuille



@ WWW.ELEKTOR.FR

→ Carte CPLD au format DIL

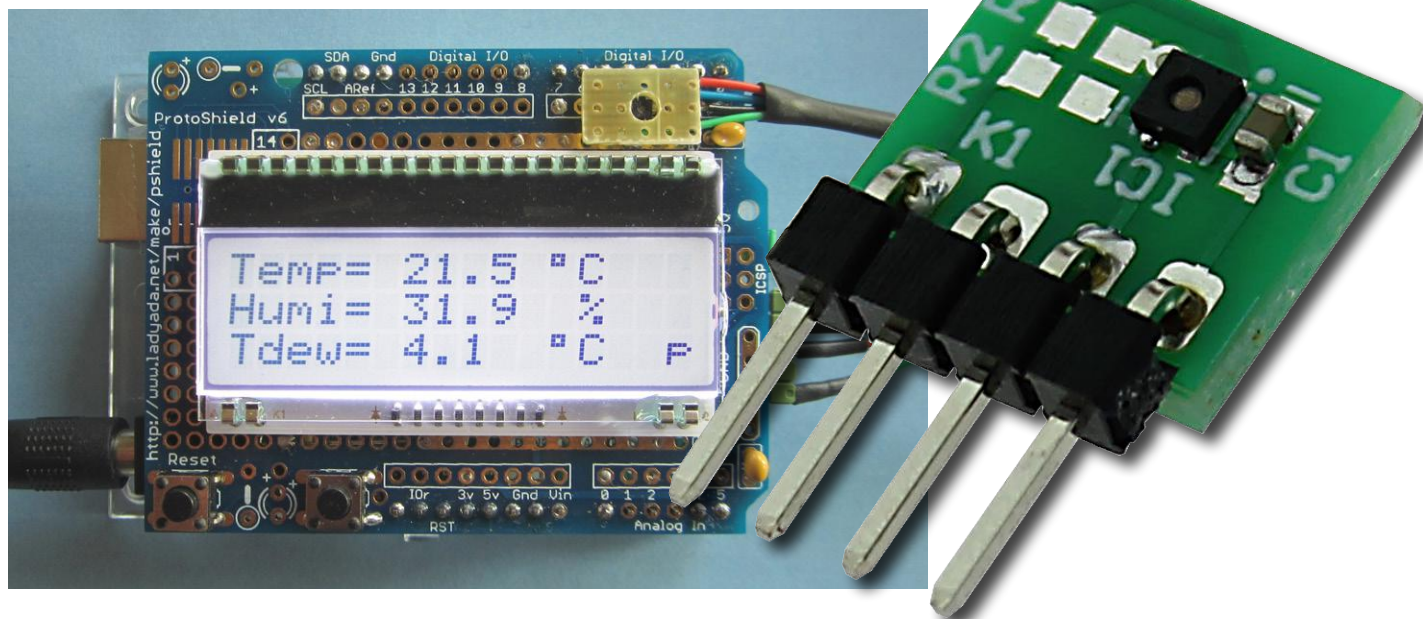
www.elektor.fr/cpld-breakout-board-160425-91

Liens

- [1] Commande de WS2812 : www.mikrocontroller.net/articles/WS2812_Ansteuerung
- [2] « Serveur web ESP8266 pour ruban de LED NeoPixels », Elektor 09-10/2018 : www.elektormagazine.fr/160487
- [3] Feuille de caractéristiques WS2812 : <https://cdn-shop.adafruit.com/datasheets/WS2812.pdf>
- [4] « Conception matérielle avec (V)HDL (1) », Elektor 09-10/2018 : www.elektormagazine.fr/180285-03
- [5] Conversion de types en VHDL : www.nandland.com/vhdl/tips/tip-convert-numeric-std-logic-vector-to-integer.html
- [6] Arithmétique vectorielle en VHDL : www.doulos.com/knowhow/vhdl_designers_guide/numeric_std/
- [7] Page de l'article : www.elektormagazine.fr/180285-C-04
- [8] « Conception matérielle avec (V)HDL (2) », Elektor 11-12/2018 : www.elektormagazine.fr/180285-B-04

e-BoB pour capteur de point de rosée ENS210

avec mise en œuvre sur Arduino Uno



Luc Lemmens (labo d'Elektor) et Roland Stiglmayr (Allemagne)

La valeur mesurée pour le paramètre physique appelé « point de rosée » indique non seulement le risque de formation de moisissures, mais aussi de corrosion d'objets en acier, y compris des outils. Ventiler et chauffer permet un contrôle partiel du point de rosée à un endroit donné. Mais pour le faire de façon pertinente, il faut d'abord déterminer le point de rosée. C'est exactement la fonction de ce montage intelligent.

Caractéristiques

- Carte d'interface (eBoB) simple d'emploi
- Capteur de température et d'humidité, miniature, de grande précision
- Exemple d'application pour Arduino

Cet appareil est avant tout un instrument de mesure très précis. La grande précision de mesure de la température et de l'humidité le rend très utile. Le capteur employé ici, le modèle ENS210 d'ams, est probablement le capteur pré-calibré le plus précis actuellement sur le marché. La température est mesurée avec une précision de $\pm 0,2$ °C dans une plage

de 0 °C à 70 °C sans aucun étalonnage. L'ENS210 peut ainsi servir de référence pour l'étalonnage de capteurs de température et d'humidité.

Qu'est-ce que le « point de rosée » ?

Le point de rosée est calculé à partir des résultats de mesure de température et d'humidité relative. On utilise pour cela l'approximation de « Magnus » pour le point de rosée ; les lecteurs intéressés par les mathématiques et la théorie sont invités à consulter l'article de Wikipedia [1].

L'écart entre le point de rosée et la température courante est aussi calculé, ce qui donne une très bonne indication de

la probabilité de formation de moisissure ou de corrosion. N.d.T. : cet écart est aussi un indicateur fort de la probabilité de brouillard, c'est une valeur très utile des bulletins météo pour l'aéronautique.

Attention : objet lilliputien en approche

Votre tendance à craquer pour les derniers circuits intégrés peut être réfrénée par leur extrême petitesse, souvent imposée par l'industrie pour répondre à la « demande du client ». La puce du capteur ENS210 ne fait pas exception avec son boîtier de seulement 2x2 mm (**fig. 1**). Heureusement, comme on peut le voir sur la photo de la **figure 2**, l'auteur a pu, au prix d'un peu d'effort, sou-

INFOS SUR LE PROJET



ENS210

Point de rosée

e-BoB



débutant

→ connaisseur

expert



env. ½ h



Arduino UNO,
EDI Arduino,
fer à souder



env. 15 €

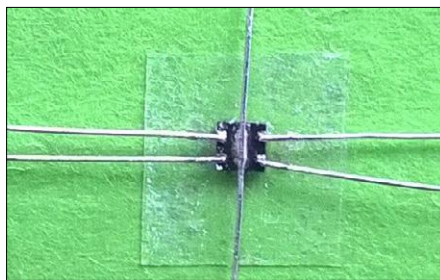


Figure 1. Minuscule, plus, encore plus... le capteur de point de rosée ENS210 d'a m s est vraiment très petit (vue de dessous, photo de l'auteur).

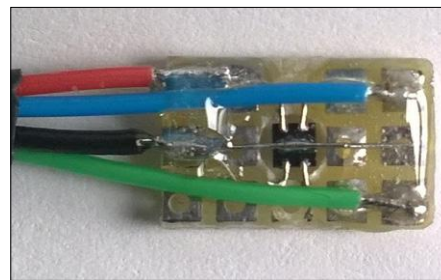


Figure 2. Méthode improvisée, mais parfaitement réussie, pour fixer l'ENS210 sur une plaque d'expérimentation (photo de l'auteur).

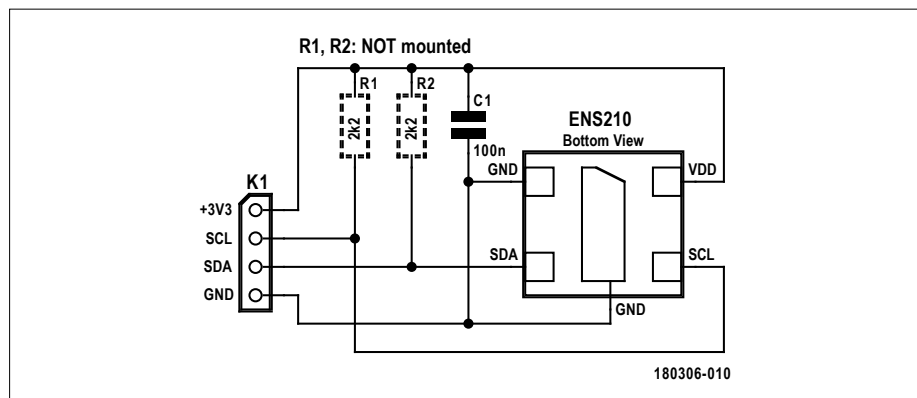


Figure 3. Diagramme de l'eBoB ENS210. Les résistances de tirage de ligne du bus I2C, R1 (SCL) et R2 (SDA), sont optionnelles.

der la minuscule puce sur un morceau de plaque d'expérimentation (plaque à bandes, Veroboard...).

La frustration face à l'incroyable petite taille des circuits intégrés actuels est une chose, le désespoir et la rage de ne pas pouvoir du tout les manipuler sont pires. C'est pourquoi, selon son habitude, le labo d'Elektor a décidé de concevoir une carte d'interface (e-BoB) pour ce magni-

fique capteur. Notre e-BoB évite tous les risques d'endommagement, de brûlure, ou même de perte de l'ENS210 lors des manipulations ou du soudage du dispositif, voire en passant l'aspirateur dans l'atelier, et facilite l'accès aux précieuses données du capteur.

e-BoB ENS210 d'Elektor

Le schéma incroyablement simple ;-) de l'e-BoB ENS210 est représenté sur la **figure 3**. Il suit la configuration recommandée par le constructeur avec rien de plus qu'un rail d'alimentation de 3,3 V découplé (C1) et les résistances de tirage (optionnelles) R1 et R2 sur les lignes SDA

et SCL de l'interface I²C. Les résistances R1 et R2 sont alimentées en 3,3 V, il faut les installer lorsque :

- aucune autre résistance de tirage n'est présente sur les lignes SDA et SCL du système ;
- il faut limiter la tension à cause d'un composant fonctionnant sous 5 V sur le bus I²C, comme un ATmega avec ses résistances de tirage internes, ce qui assure un niveau haut sécurisé de 3,6 V sur les broches SDA et SCL du capteur.

Le capteur ne doit en aucun cas être utilisé sans résistance de tirage sur les



LISTE DES COMPOSANTS

Résistances

R1, R2 = voir texte

Condensateur

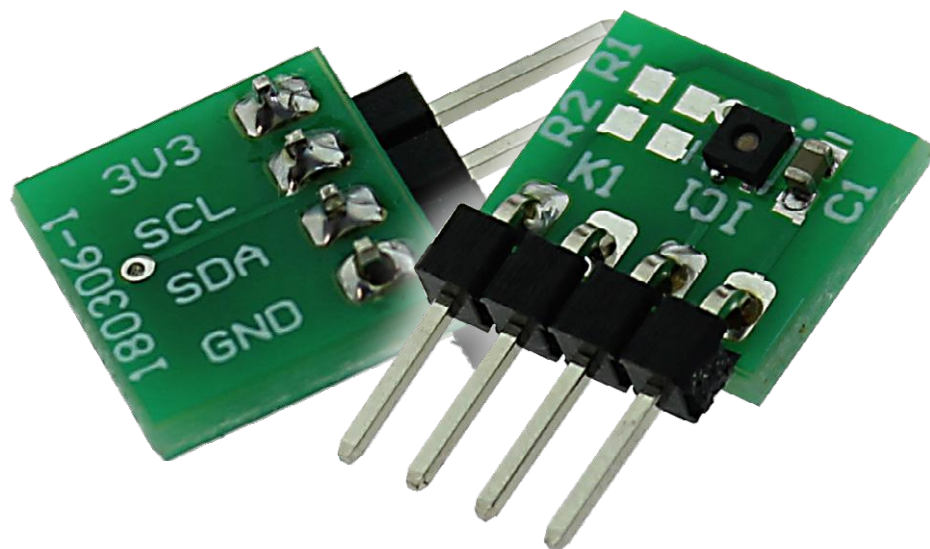
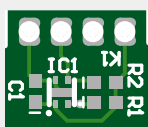
C1 = 100 nF, X7R, 0603

Semi-conducteur

IC1 = ENS210-LQFM (a m s),
par ex. réf. 985-ENS210-LQFM
chez Mouser

Divers

K1 = barrette mâle à 4 broches,
1 rangée au pas de 2,54 mm
Circuit imprimé, réf. 180306-1
(www.elektor.fr)



lignes SDA et SCL.

Enfin, la barrette à 4 broches K1 recueille les données de l'ENS210 et fournit l'alimentation de 3,3 V.

« Arduino connection »

En reliant son capteur ENS210 à un Arduino Uno, l'auteur a créé un instrument de mesure du point de rosée pratique et il l'a aimablement publié sur le site du labo d'Elektor. Grâce à l'utilisation d'un Arduino Uno, le matériel nécessaire pour ce projet est simple, avec peut-être un seul défi : les quelques connexions électriques de l'écran LCD. La solution la plus simple ici aurait pu être de revenir à un *shield* Proto, qui dispose d'une embase pour connecter le capteur via un câble.

La **figure 4** montre le schéma de l'appareil de mesure du point de rosée à lecture directe de l'auteur, avec quelques fioritures, conçu pour fonctionner avec son logiciel fait maison. Le programme qui en résulte, ou « croquis » en langage Arduino, est téléchargeable gratuitement [2]. Notez que ce circuit et le logiciel Arduino ne sont pas associés avec un circuit imprimé conçu par le labo d'Elektor, c'est pourquoi cette partie de l'article est rangée dans la catégorie **Projet de lecteur**.

Comme l'ENS210 dispose d'une interface I²C complète, pratiquement aucun matériel supplémentaire n'est nécessaire. Bien que toutes les mesures soient affichées sur un écran, elles sont aussi envoyées sur le moniteur série ou le traceur série de l'EDI Arduino. Un bouton-poussoir permet de sélectionner le format de sortie série pour le moniteur ou le traceur. On utilise un agréable écran LCD à trois lignes avec un rétroéclairage blanc (du fabricant Electronic Assembly), mais vous pouvez en choisir un autre. Cet écran LCD à trois lignes est pris en charge par le croquis Arduino [ENS210_3L_34](#). Avec juste quelques fonctions supplémentaires, de nombreux éléments de la bibliothèque standard [LiquidCrystal](#) peuvent être utilisés pour étendre ce projet. La bibliothèque elle-même n'a pas besoin d'être modifiée. Il est aussi possible, bien sûr, d'utiliser un *shield* LCD d'Elektor avec le croquis [ENS210_2L_25](#). Pour compléter ce montage, vous pourriez ajouter des boutons pour commander une fonction d'enregistrement des données. Cette fonction stockerait les mesures dans l'EEPROM non volatile.

Enfin, notez que sur le schéma, MOD2

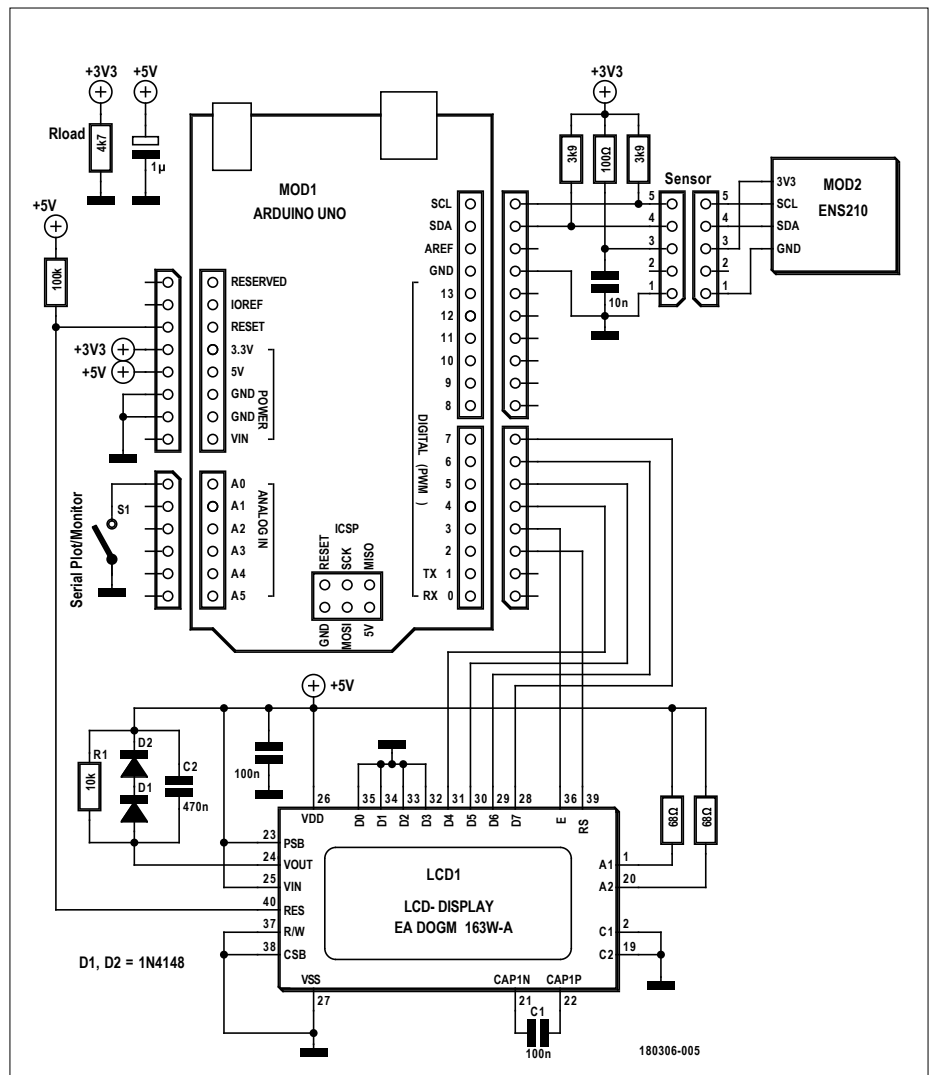


Figure 4. Système « hôte » compatible Arduino pour le capteur de point de rosée ENS210, c'est le montage conçu par l'auteur. Ce système associé à un Arduino et un écran LCD constitue un instrument novateur pour analyser des zones potentiellement affectées par la formation de moisissures et la corrosion. Si la fonction surbrillance de l'écran n'est pas requise, supprimez C1, C2, D1, D2, R1 et reliez les broches 24 et 25.

est le circuit imprimé fait maison par l'auteur pour le capteur (**fig. 2**), donc pas l'eBoB d'Elektor qui a un brochage différent. Un recâblage ne devrait pas être un problème : il suffit de faire cor-

respondre les lignes +3,3 V, SCL, SDA et GND de l'eBoB avec celles du circuit Uno « hôte ».

(180306-04 -

version française : Denis Lafourcade



@ WWW.ELEKTOR.FR

→ Carte assemblée de l'eBoB de mesure du point de rosée, réf. 180306-1
www.elektor.fr/180306-1

Liens

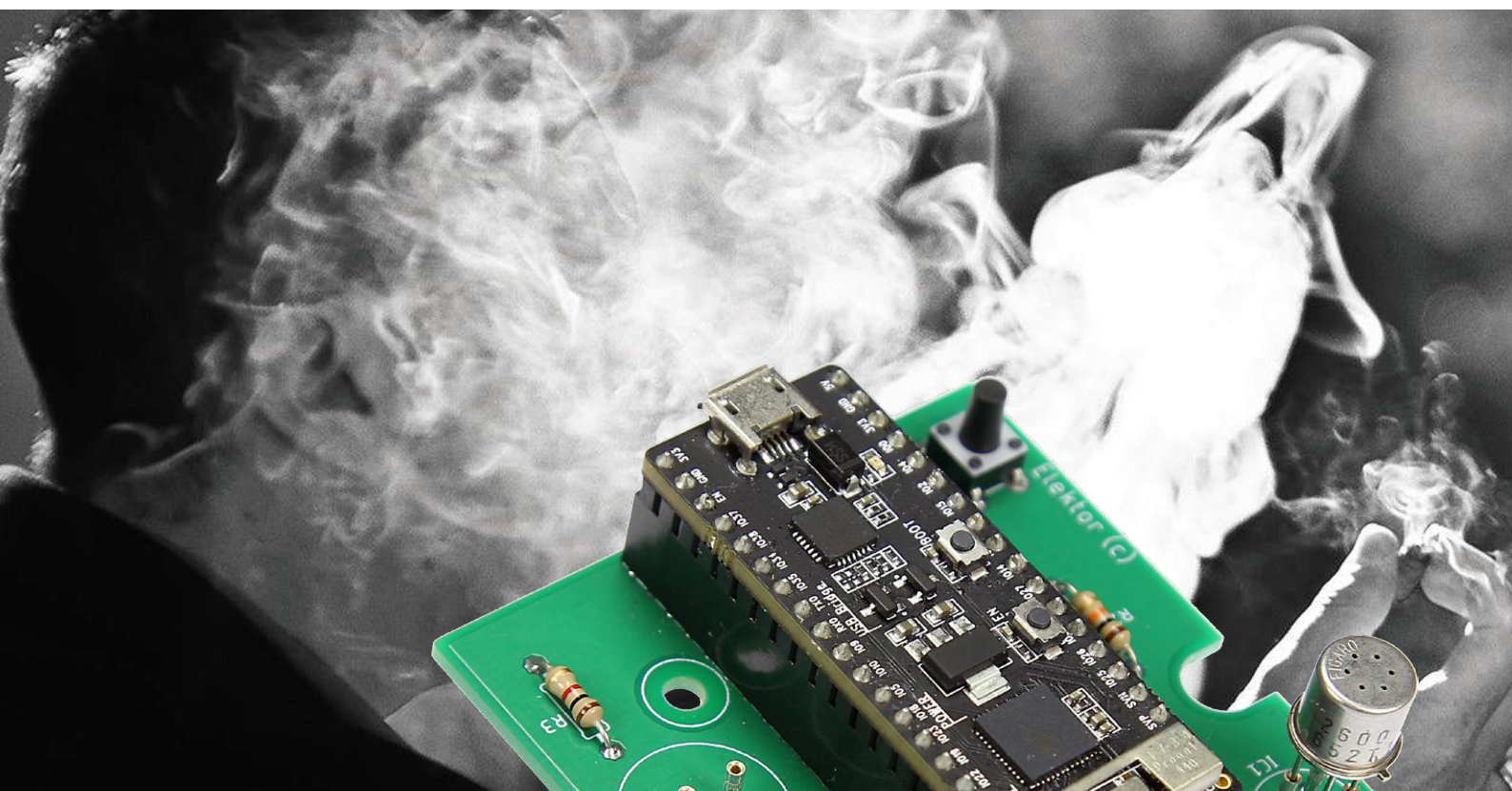
- [1] Théorie du point de rosée : https://fr.wikipedia.org/wiki/Point_de_rosée
- [2] Croquis Arduino : www.elektormagazine.com/180306-01

surveillance du niveau de pollution de l'air

y a-t-il un fumeur caché dans les toilettes ?

Bera Somnath (Inde) et Roy Aarts (labo d'Elektor)

Nous devrions tous être préoccupés par la qualité de l'air que nous respirons. Nous savons que les particules de poussière en suspension dans l'air empêchent les asthmatiques de dormir la nuit, mais les niveaux de certains gaz sont tout aussi importants. Ce projet mesure le niveau des gaz qui contaminent l'air et déclenche une alarme quand leur concentration dépasse un certain seuil, par exemple lorsque quelqu'un allume une cigarette...



Caractéristiques

- Capteur de mesure de la qualité de l'air TGS2600 de Figaro
- Détecte les gaz comme le méthane, le monoxyde de carbone, l'isobutane, l'éthanol, l'hydrogène
- Wi-Fi (basé sur l'ESP32)
- Alimentation de 5 VCC / 0,5 A



Figure 1. Le capteur Figaro de type TGS2600 détecte les gaz comme le méthane, le monoxyde de carbone, l'éthanol, l'hydrogène et l'isobutane.

Le capteur utilisé dans ce projet, le modèle TGS2600 de Figaro (spécialiste japonais des capteurs de gaz), réagit à de faibles concentrations de gaz divers dont le méthane, le monoxyde de carbone, l'éthanol, l'hydrogène et l'isobutane. Ce dernier est souvent utilisé comme propulseur dans les bombes aérosol. Notez que le capteur ne fait pas réellement la distinction entre ces gaz — il se contente de détecter leur effet sur l'air pur. De plus, le TGS2600 n'est pas un détecteur de gaz en soi, mais plutôt un capteur de « gaz présents dans l'air », ce qui signifie qu'il ne fonctionnera pas si le niveau d'oxygène est trop bas. Comme le TGS2600 réagit aux changements de la qualité de l'air, il excelle dans la détection de la fumée de cigarette par exemple. Même si le TGS2600 détecte toutes sortes de gaz, il y en a qu'il n'apprécie pas vraiment de renifler. Il faut éviter les vapeurs de silicone (pas de silicium) et les environnements hautement corrosifs, comme l'eau et surtout l'eau de mer (mais presque rien ne résistera à l'eau de mer. Ni à la pisserie de chat, mais c'est une autre histoire).

À propos du capteur

Le TGS2600, logé dans un petit boîtier métallique (fig. 1), repose sur un élément résistif en dioxyde d'étain (SnO_2). Il faut le chauffer pour lui permettre de réagir aux concentrations de gaz, et justement un chauffage adapté est intégré au dispositif. À cause du chauffage, il faut quelques secondes au capteur après la mise sous tension pour être prêt. À la première mise sous tension ou après une

période d'inactivité de plusieurs semaines ou plus, le démarrage du capteur prend bien quinze à vingt minutes.

L'élément résistif (R_{GAZ} ; R_s) à l'intérieur du capteur constitue simplement la partie « haute » d'un diviseur de tension résistif. Comme sa résistance varie en fonction de la composition de l'air ambiant, le signal de sortie résultant est la tension aux bornes de R_L , la partie « basse » du diviseur de tension, câblée à l'extérieur du capteur. La résistance du capteur dans l'air pur est appelée R_0 , c'est la valeur de référence. La résistance R_s est ainsi inversement proportionnelle à la pollution de l'air. Les niveaux de température et d'humidité relative ambiantes affectent la sensibilité du dispositif au degré près, mais tant qu'on utilise le capteur entre 10 et 50 °C, aucune compensation n'est nécessaire.

Circuit de surveillance de la pollution de l'air

L'utilisation du TGS2600 dans un circuit est simple : il faut un microcontrôleur avec un convertisseur analogique/numé-

INFOS SUR LE PROJET

ESP32

qualité d'air

détecteur de fumée

débutant

connaisseur expert

env. 2 h

EDI Arduino, fer à souder

env. 30 €

rique (CA/N) intégré (fig. 2). Comme nous voulions rendre les données accessibles sur un réseau, nous avons utilisé

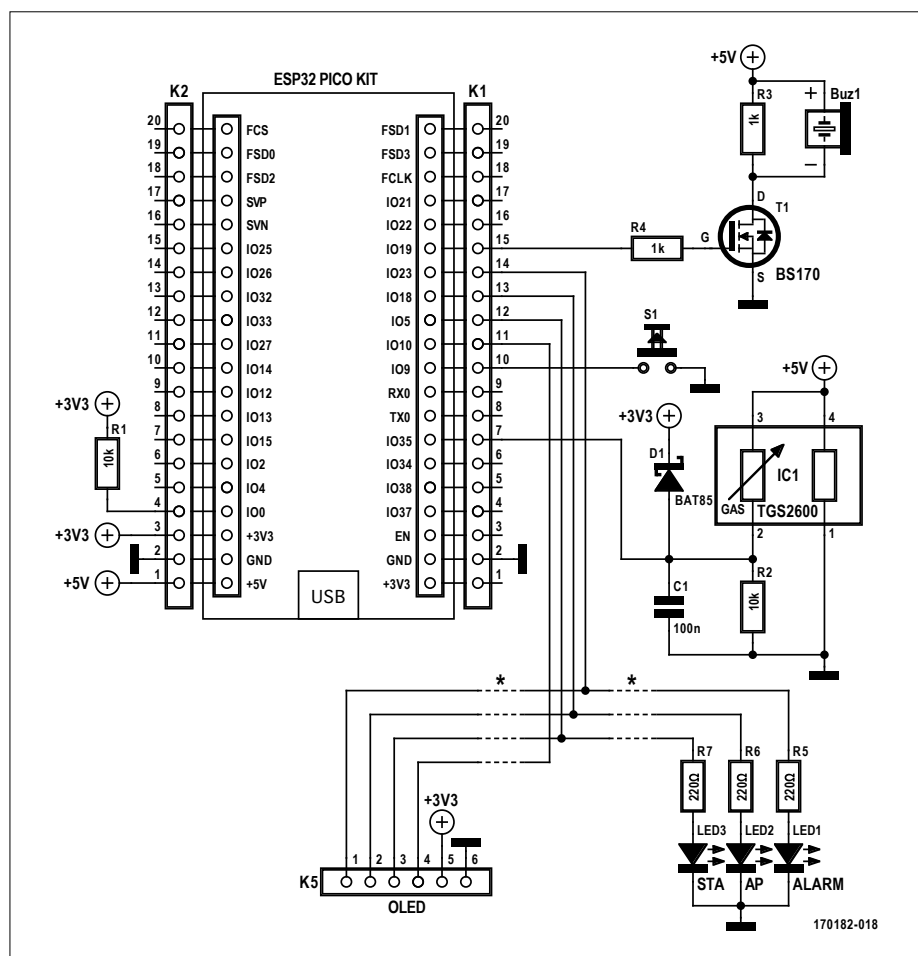


Figure 2. En réalité le TGS2600 effectue le plus gros du travail, c'est pourquoi le schéma du contrôleur de qualité de l'air peut rester simple.

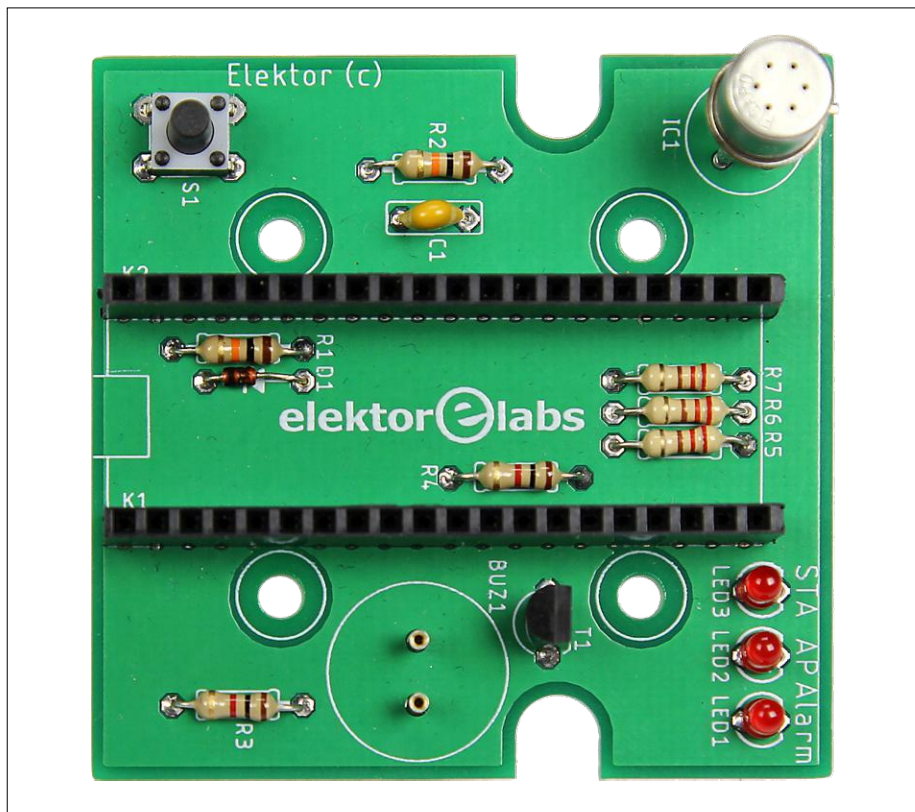


Figure 3. Quelques composants sont cachés sous le module ESP32. Le boîtier métallique en haut à droite de la carte, c'est le capteur TGS2600.

pour cela un module à microcontrôleur à base d'ESP32. En réalité nous avons fait deux versions de ce projet : une sur un bout de plaque de prototypage et une autre sur un circuit imprimé. La première version comporte un module ESP32-PICO-kit, un écran OLED de 2,4 cm, un bouton-poussoir, un buzzer et bien sûr le capteur. La version sur circuit imprimé (CI) est presque identique sauf que pour visualiser les données, l'écran OLED est remplacé par une page web. On a ajouté trois LED pour montrer l'état de la connexion au réseau sans fil (fig. 3). La version CI conçue au labo d'Elektor tient dans un boîtier bon marché Hammond de type 1593k (fig. 4).

Logiciel

Fondamentalement, le logiciel se cantonne à mesurer la tension aux bornes de R2, qui correspond à R_L mentionnée ci-dessus. Comme on connaît la valeur de R2 (10 k Ω comme sur le schéma), on peut calculer la valeur de R_s . Ensuite, une fois R_o connue, on peut calculer le rapport R_s/R_o c.-à-d. le niveau de pollution de l'air. Le problème ici c'est R_o , parce que sa valeur est inconnue. Pour

Liens

- [1] Bibliothèque graphique U8g2lib : <https://github.com/olikraus/u8g2>
- [2] Noyau Arduino ESP32 : <https://github.com/espressif/arduino-esp32>
- [3] Outil de téléchargement SPIFFS : <https://github.com/me-no-dev/arduino-esp32fs-plugin>

ThingSpeak

Les mesures de qualité de l'air peuvent être téléchargées sur la plateforme ouverte de l'IdO nommée ThingSpeak. Pour cela, créez un canal avec un champ de données. Vous obtiendrez alors une clé API d'écriture que vous devez saisir sur la page web de configuration de l'ESP32. Une fois que l'ESP32 est correctement configuré, les mesures sont téléchargées à chaque échéance de l'intervalle de mesure réglé. De plus, la valeur envoyée est la moyenne calculée sur l'intervalle. Pour envoyer les données à ThingSpeak, on envoie d'abord au site web une requête HTTP qui contient la valeur du capteur. ThingSpeak répond avec le nombre d'entrées du canal, ou avec zéro (0) si la requête échoue. Une des raisons de l'échec d'une requête est que la clé d'API est mal saisie, ou que les requêtes sont envoyées trop rapidement l'une derrière l'autre (ThingSpeak limite l'intervalle de téléchargement à 15 s pour les utilisateurs qui profitent de la gratuité). L'intervalle minimum de téléchargement du système de mesure de la pollution de l'air est d'une minute.



LISTE DES COMPOSANTS

Résistances

R1, R2 = 10 k Ω
 R3, R4 = 1 k Ω
 R5, R6, R7 = 220 Ω

Condensateur

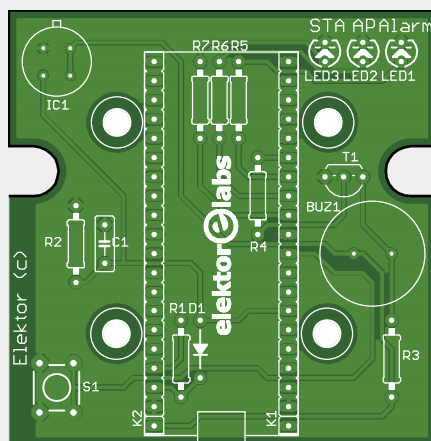
C1 = 100 nF

Semi-conducteurs

D1 = BAT85
 IC1 = TGS2600
 LED1, LED2, LED3 = rouge, diam. = 3 mm
 T1 = BS170

Divers

BUZ1 = PS1420P02CT
 K1, K2 = embase mâle à 17 contacts, au pas de 2,54 mm (pour ESP32-PICO-kit)



S1 = bouton-poussoir, à monter sur circuit imprimé, 6×6 mm
 Circuit imprimé, réf. 170182-1 (www.elektor.fr)
 ESP32-PICO-kit, réf. 18423 (www.elektor.fr)

l'obtenir, on doit faire un étalonnage dans un environnement de référence supposé être de « l'air pur ».

La valeur calculée pour le niveau de pollution de l'air est soit affichée dans un graphe linéaire sur le minuscule écran OLED, soit téléchargée sur la célèbre plateforme ThingSpeak de l'Internet des Objets, où l'on peut visualiser les mesures sur une courbe sur une période plus longue. L'écran OLED n'affiche que les vingt dernières mesures.



Les mesures sont aussi comparées avec une valeur de seuil. Lorsque la pollution de l'air est inférieure à ce seuil, le buzzer de la carte sonne. Selon la fiche technique du capteur, un rapport R_z/R_0 de 0,85 est une valeur de seuil d'alarme raisonnable. Le seuil d'alarme figure sur l'écran sous la forme d'une ligne en pointillés horizontale. Si l'alarme est active, il suffit de presser le bouton pour l'arrêter. En appuyant sur le bouton, vous aurez des informations sur la connexion Wi-Fi actuelle ainsi que sur la valeur du seuil d'alarme. Vous saurez également si ThingSpeak est activé ou non.

Configuration et étalonnage

On aurait pu en rester là, mais alors où serait le plaisir ? Alors, pour compliquer légèrement, mais positivement les choses, on a ajouté un serveur web. Il permet de configurer la surveillance de pollution de l'air au moyen d'une page web adaptée, hébergée par l'ESP32 lui-même. Sur cette page, on peut régler le seuil d'alarme ainsi que les réglages de ThingSpeak – c.-à-d. la clé d'API néces-

Taille unique

Les deux versions de ce projet – plaque de prototypage avec écran OLED, ou circuit imprimé sans écran – utilisent le même croquis Arduino. Pour faire la différence entre les deux versions, on utilise une définition de pré-traitement. C'est la première ligne du code où on lit `#define USEDISPLAY`. Supprimez-la ou mettez-la en commentaire pour la version « circuit imprimé ».

Pour utiliser l'écran, il faut installer une bibliothèque additionnelle nommée U8g2lib [1]. On l'utilise pour tracer le texte et le graphe sur l'écran.

Pour télécharger le logiciel dans l'ESP32, il faut installer le noyau Arduino ESP32 [2]. Après avoir installé le noyau, il faut aussi installer l'outil de téléchargement SPIFFS pour télécharger la page web dans le système de fichiers de l'ESP32. Lorsque tout est correctement réglé, on peut sélectionner l'ESP32-PICO-kit depuis le menu **Cartes** et on peut télécharger la page web et le croquis dans le module ESP32.

Mode « point d'accès »

L'ESP32 doit être en mode « point d'accès » (AP) lors de la première configuration. Ce mode est activé en allumant le module tout en appuyant sur le bouton ; la LED2 « AP » devrait s'allumer. Lorsque c'est le cas, connectez un ordinateur, un ordiphone ou une tablette au réseau de l'ESP32 et naviguez à l'adresse 192.168.4.1 pour activer la page web de configuration du système de mesure de la pollution de l'air. Vous pouvez maintenant régler le réseau Wi-Fi et les autres paramètres mentionnés ci-dessus.

La communication entre la page web et le module ESP32 s'effectue via des « XMLHttpRequests » asynchrones sur la page web. Lorsque la page se charge, une requête http est créée et envoyée à l'ESP32. À réception de cette requête, l'ESP32 envoie une réponse à la page web contenant la configuration courante. La pression du bouton « Submit » sur la page web produit une transaction similaire, mais qui contient maintenant les valeurs saisies par l'utilisateur. ◀

(170182-04 –

version française : Denis Lafourcade)

@ WWW.ELEKTOR.FR

→ Contrôleur du niveau de pollution de l'air, circuit imprimé nu
www.elektor.fr/air-pollution-monitor-170182-1

→ ESP32-PICO-kit
www.elektor.fr/esp32-pico4

→ Capteur de polluants de l'air TGS2600
www.elektor.fr/tgs2600-air-sensor

→ Écran OLED de 2,4 cm, 128×64, 6 broches
www.elektor.fr/oled-display-for-arduino

sonde pour bus I²C

données envoyées par USB vers un écran de PC

Willem den Hollander (Confédération helvétique)

Le bus I²C (pour *Inter-Integrated Circuit*) sert couramment à la communication entre les puces d'un même circuit. Comment percer le contenu de ces échanges discrets ? Le circuit présenté ici parvient à discerner les données I²C qui transitent par le bus, à les décoder et à les transcrire par USB sur un PC sous forme hexadécimale. Il permettrait aussi de filtrer les données pour sélectionner celles qui possèdent des adresses spécifiques.

Le bus I²C s'appuie sur deux lignes de signal : l'horloge SCL et la ligne de données SDA. La transmission de données s'effectue en série. Chaque message se compose de deux à plusieurs octets, chacun d'eux suivi d'un bit d'acquiescement (ACK, *acknowledge* ou accusé de réception). Une trame initiale indique le début d'un message, une trame d'arrêt le clôture. Transférer toutes les informations nécessite d'envoyer deux fois plus d'octets que la substance des données elles-mêmes. En mode standard, la cadence sur le bus I²C est de 100 kHz, ce qui donne un débit de données de 20 Ko/s. En mode rapide (*fast*), il est de 80 Ko/s. Dans l'application qui nous occupe, il

s'agira d'un seul paquet de 64 octets par milliseconde sur le canal USB. C'est plus qu'il n'en faut pour la lecture des données sur un bus I²C en mode standard, mais insuffisant pour le mode rapide. Notre sonde de bus peut malgré tout faire l'affaire en mode *fast* aussi longtemps qu'il y a peu de trafic, ce qui est souvent le cas.

Le matériel

La **figure 1** montre le schéma de la sonde pour bus I²C. On y voit deux microcontrôleurs. J'ai mis en œuvre le PIC16F1455 (U2) pour lire les données I²C et les envoyer par USB au PC.

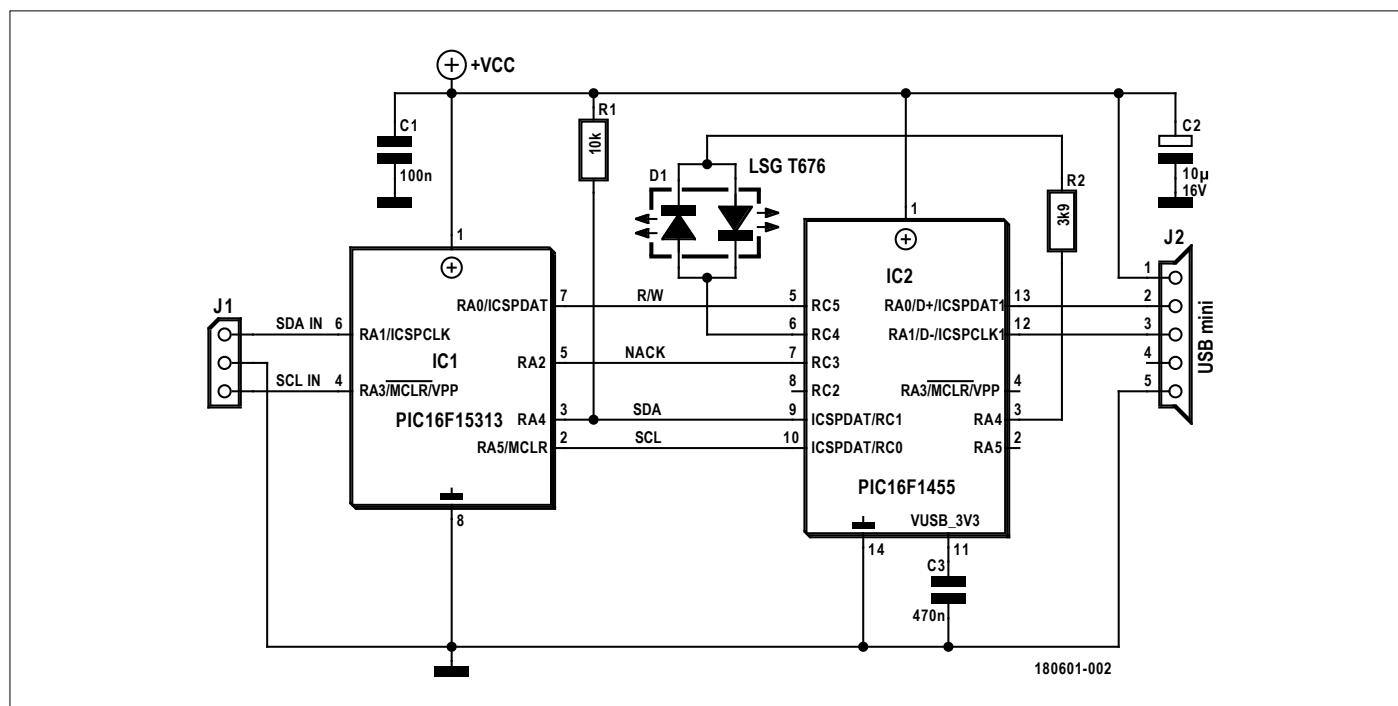


Figure 1. Voici le schéma complet de la sonde pour bus I²C.

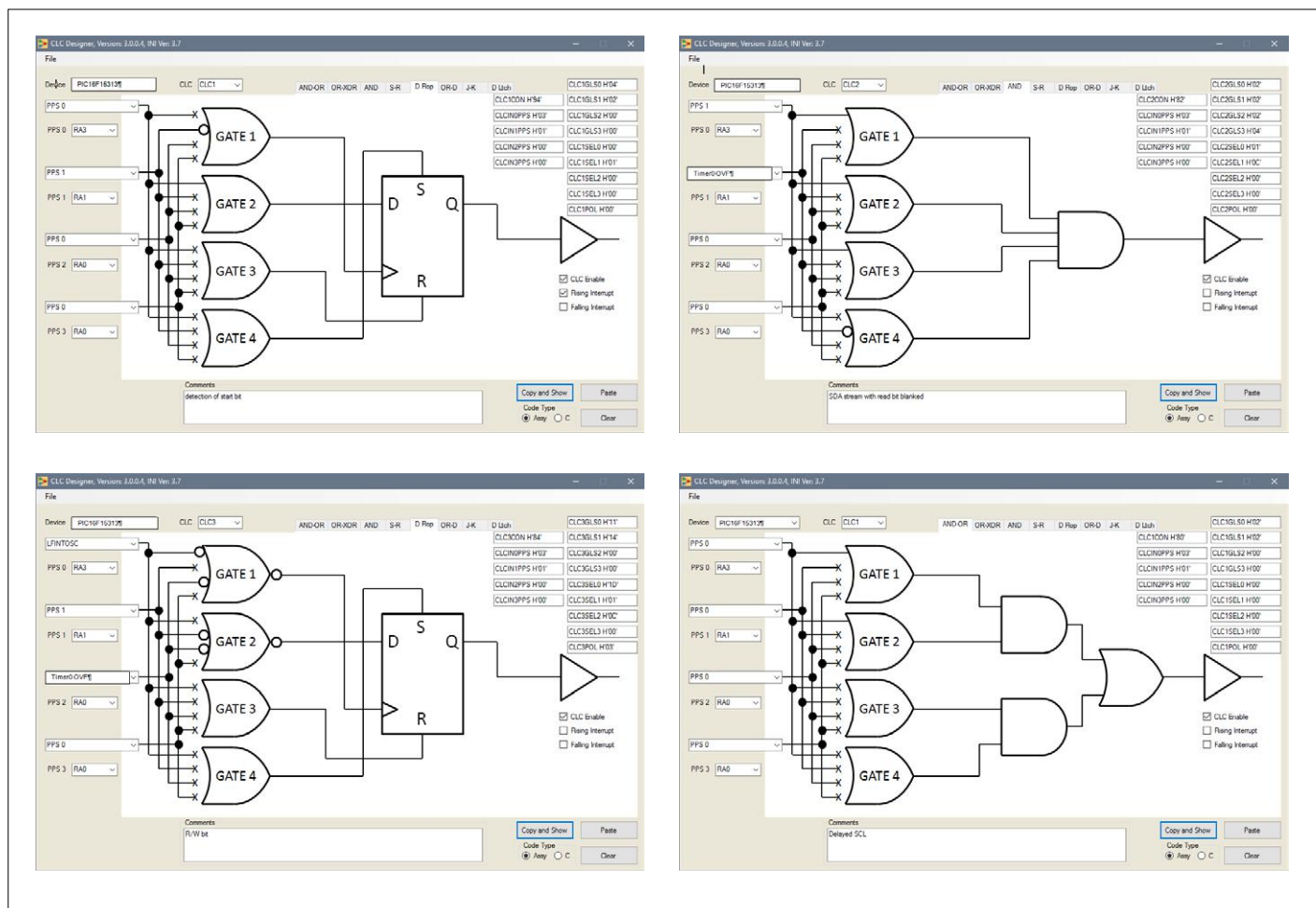


Figure 2. Les cellules logiques à configurer (CLC) dans U1 servent à configurer à leur tour la sonde pour bus I²C.

Le module I²C est configuré en récepteur esclave pour collecter toutes les données à n'importe quelle adresse, pour autant que le bit **d'écriture** des données entrantes soit présent. Si le bit de **lecture** est activé, toutes les données sont ignorées. De plus, on ne peut pas lire les informations d'acquiescement, puisque c'est le module I²C lui-même qui les fournit.

C'est pourquoi j'ai ajouté un préprocesseur, un PIC16F15313 (U1) qui transforme chaque bit de lecture en bit d'écriture. Les bits de lecture et d'acquiescement sont extraits du flux de données et transmis à part au PIC16F1455 (U2), lequel incorpore ces informations dans les données I²C reçues et les envoie par USB au PC.

La puce U1 est un microcontrôleur PIC équipé de quatre cellules à configurer en logique CLC (**C**ellule **L**ogique à **C**onfigurer). La **figure 2** montre la conformation retenue pour ce projet. Le microcontrôleur PIC16F1455 dispose d'une interface USB complète et son oscillateur interne a une précision intrinsèque suffisante pour se passer d'un quartz externe. Le microcontrôleur range tous les octets reçus à la bonne place en mémoire et les transmet au PC dès qu'il en a reçu suffisamment. Le micrologiciel des deux processeurs a été rédigé en assembleur. C'est ce qui permet de maîtriser avec précision une chronologie critique. La configuration du module USB est aussi écrite en assembleur. Cette partie du micrologiciel provient à l'origine de Microchip et, après quelques adaptations et débogages, elle est actuellement en service depuis une douzaine d'années.

Le circuit se satisfait de peu de courant, son alimentation lui vient simplement de la prise USB du PC. J'ai construit le circuit avec des CMS sur une petite carte à double face de 14 sur 33 mm (cf. **fig. 3**).

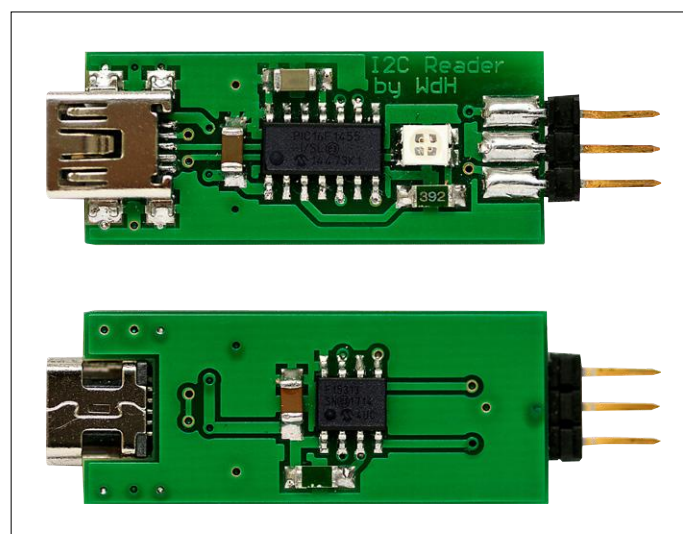


Figure 3. Ce sont les deux faces du circuit imprimé que j'ai conçu pour y installer les CMS de mon projet de sonde de bus I²C.

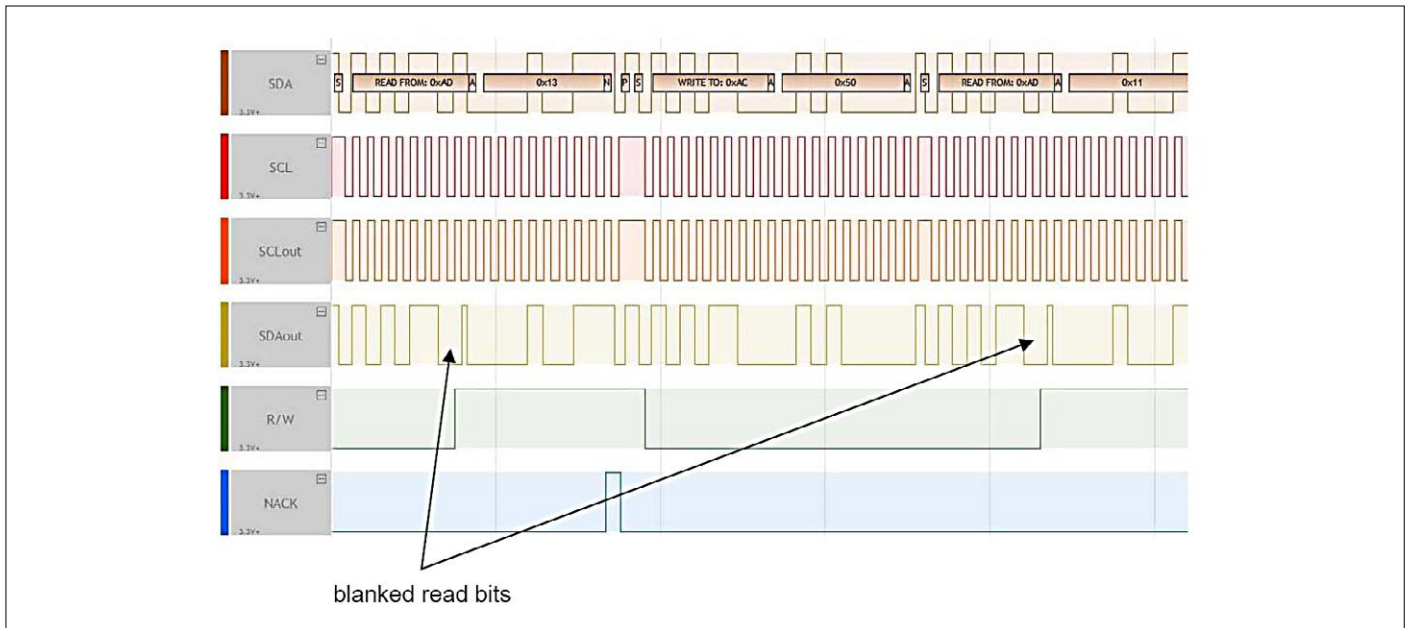


Figure 4. Diagramme synoptique des signaux traités par le PIC16F15313 (U1) avec indication de l'emplacement du bit de lecture à éliminer.

Micrologiciel et programme

Le temporisateur *Timer0* de U1 ouvre une fenêtre de la largeur d'un bit qui vide le bit de lecture, voir aussi la **figure 4**. Le **listage 1** montre comment CLC2 met à zéro le bit de lecture. On voit en haut, à droite de la figure 2, le signal SDA modifié par la suppression, la sortie de CLC2 est connectée à la sortie RA4. RA4 est une sortie à drain ouvert avec une résistance externe (R1) de rappel vers le haut. C'est une nécessité, parce que U2 renvoie toujours un acquiescement (ACK) après avoir reçu un octet du préprocesseur. L'entrée de U2 devient donc une sortie pendant la durée d'un bit. Les informations lues sont disponibles à la sortie de CLC3 (cf. en bas à gauche dans

la figure 2) et sont acheminées vers la sortie RA0. Comme CLC3 retarde quelque peu le signal SDA, il faut différer d'autant le signal SCL pour rétablir la concomitance. C'est CLC4 qui s'en charge. Un petit bout de programme règle le temporisateur et passe le bit d'ACK à RA2. La **figure 5** reproduit l'interface graphique (GUI) du logiciel correspondant pour PC. La ligne d'état en bas indique si la sonde de bus I²C est connectée et reconnue ou pas. Le bouton *Start Acquisition* lance la lecture des données sur le bus. La LED de la sonde s'éclaire alors en vert. Le bouton de démarrage prend dès lors la fonction d'arrêt. Les données du bus s'affichent dans la fenêtre où les signes suivants sont en vigueur : S pour

Listage 1

Ce fragment de code montre la configuration de CLC2 (figure 2b). On y met à zéro le bit de lecture dans le flux de données de SDA. *Timer0* est lancé quand le bit de départ est détecté.

```

banksel    CLC2GLS0 ; TMR0 ANDed with SDA clears bit 8
movlw      2
movwf      CLC2GLS0
movwf      CLC2GLS1
movwf      CLC2GLS2
movlw      4
movwf      CLC2GLS3
movlw      1          ; SDA input signal
movwf      CLC2SEL0
movlw      0ch        ; TMR0 overflow
movwf      CLC2SEL1
clrf       CLC2SEL2
clrf       CLC2SEL3
clrf       CLC2POL
movlw      82h
movwf      CLC2CON

```

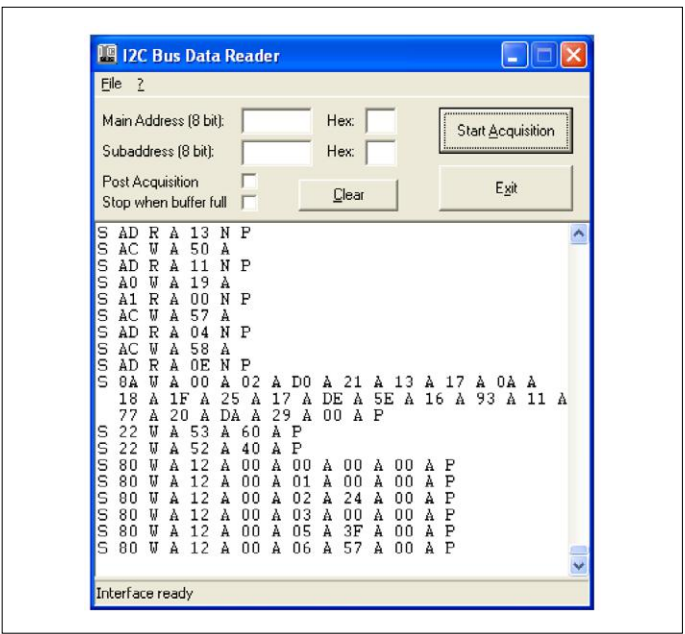


Figure 5. L'interface graphique utilisateur est très claire et offre en plus quelques commodités.

Start, R pour Read (lecture), W pour Write (écriture), A pour ACK (acquiescement), N pour NACK (refus) et P pour stop. Les octets de données s'affichent au format hexadécimal.

Le logiciel permet un filtrage sur les deux premiers octets sous forme hexadécimale ou binaire. Ces deux octets une fois remplis, toutes les données qui commencent par eux sont reproduites. Si vous avez coché *Post Acquisition*, la collecte des données ne débute qu'après l'arrivée des octets indiqués. Lors d'un filtrage en binaire, on peut définir un état « don't care » (pas d'importance), codé par un X.

La capacité du tampon (*buffer*) est d'environ 40 Ko. En cas de dépassement, les données du début sont écrasées. Pour l'éviter, on coche « Stop when buffer full », ce qui arrête la lecture à temps et éteint la LED verte.

S'il y a un surplus de données, leur collecte s'arrête ; du coup, la LED devient rouge et le logiciel boucle automatiquement vers le départ. Les données à l'écran y restent jusqu'à l'effacement ou l'écrasement d'une partie du tampon. Dans le menu *File*, vous pouvez copier sur disque dur ou imprimer tout ou partie des données à l'écran.

J'ai remarqué que l'afflux permanent de données sur le PC accablait lourdement le SE Windows. Mieux vaut donc éviter l'exécution d'autres programmes en même temps.

À l'usage

Avant la première utilisation de la sonde de bus I²C, il faut installer le pilote USB. Quand vous branchez le câble USB de la sonde sur le PC, celui-ci détecte un nouvel appareil. Le téléchargement lié à cet article [1] contient un fichier *.inf* dans le répertoire du pilote. Sélectionnez-le quand le PC demande un pilote, il s'installera automatiquement. D'un double clic sur le programme *Setup.exe*, présent également dans le téléchargement [1], vous installez l'interface utilisateur et puis il suffit de suivre les instructions à l'écran.

Conclusion

La particularité de la sonde de bus I²C décrite ici est qu'elle donne accès aux données recueillies aussi longtemps que nécessaire, vous pourrez ainsi détecter à votre aise les erreurs, même les moins fréquentes. Remanier le programme PC permet d'enregistrer les données reçues sur disque dur pour une analyse ultérieure. ◀

(180601-03 – version française : Robert Grignard)

Lien

[1] Page de l'article : www.elektormagazine.fr/180601-03



@ WWW.ELEKTOR.FR

→ Livre en anglais, 'Elementary Course BASCOM-AVR'
www.elektor.fr/elementary-course-bascom-avr

→ Livre en anglais, 'MAKE AVR Programming'
www.elektor.fr/make-avr-programming



MOUSER
ELECTRONICS



SamacSys
ELECTRONIC COMPONENT LIBRARY SOLUTIONS

Télécharger GRATUITEMENT

- Symboles schématiques
- Empreintes de PCB
- Modèles 3D

Cliquez et déposez dans
votre CAO électronique

Les dernières nouveautés pour vos
conceptions les plus récentes™

mouser.fr/ecad

sources d'interférences dans l'électronique automobile

et comment les éliminer !

Tam Hanna

Si vous baignez dans le monde des calculateurs industriels classiques et que vous devez un jour ou l'autre concevoir un circuit pour un véhicule, vous pouvez facilement tomber dans le piège de l'électronique automobile. L'objet de cet article est de décrire ce qui peut mal tourner, et les risques encourus.

Un mot d'introduction avant d'aller à l'essentiel : l'auteur a connaissance d'un bon nombre de petites séries, dont certaines sont utilisées depuis des années qui n'incluent que peu ou

pas du tout les mesures de protection mentionnées ci-après. Cela est dû, entre autres, à la nécessité de penser un système automobile comme un réseau de termes e^{-t} (voir **figure 1**). Certains circuits sont « sur le grill » et soumis au gros des interférences, alors que d'autres restent relativement peu perturbés. Les mesures nécessaires à la protection adéquate d'un projet ne coûtent pas les yeux de la tête, et sont donc de toute façon économiquement pertinentes.

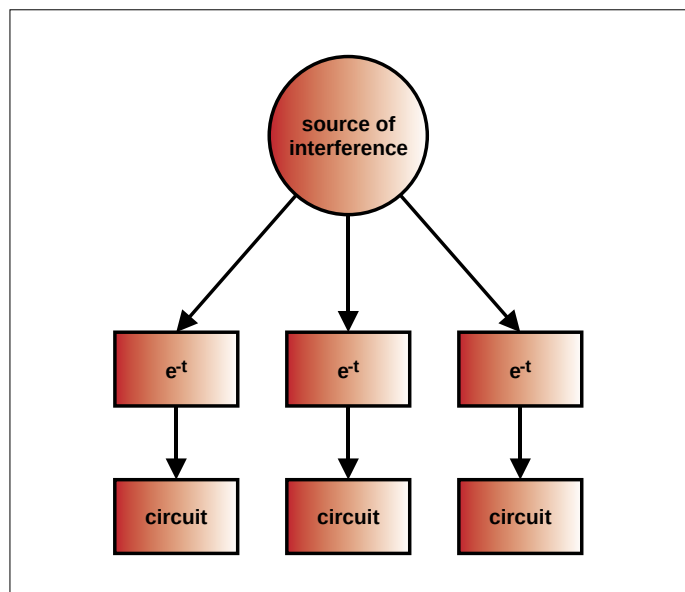


Figure 1. Les circuits sont exposés à diverses sources d'interférences.

Générateur brutal

Commençons par les interférences les plus vicieuses connues dans l'industrie sous le nom de délestage (en anglais : *load shed*). À cet effet, imaginons un quad ou tout véhicule similaire caracolant sur un chemin de terre. La stabilité de la connexion de l'alternateur à la batterie est soumise à rude épreuve. Le fabricant a fait des économies, le mécanicien a été distrait, ou la corrosion s'est infiltrée dans les éléments de la batterie. Peu importe, au passage sur une pierre particulièrement anguleuse, arrive ce qui devait arriver : la connexion entre batterie et alternateur est brièvement interrompue. Mais l'alternateur veut, ou plutôt doit, décharger son énergie excédentaire d'une manière ou d'une autre. Heureusement le phénomène est fort rare. À ce stade, on rencontre pour la première fois un obstacle qui s'avère courant en électronique automobile : les constructeurs concurrents spécifient plus ou moins à peu près tout, en

se basant sur les avis de divers organismes de normalisation concurrents. C'est ce qu'on appelle une jungle !

Retour au délestage. Il existe deux spécifications du groupe ISO : premièrement, la norme ISO-7637-2, valable jusqu'en 2010, encore utilisée par certains constructeurs automobiles (tel que Ford) et qui peut être téléchargée sous [1]. Et deuxièmement, la version ISO-16750-2 actuellement en vigueur, mais qui n'est disponible qu'après paiement. Le fabricant de semi-conducteurs Vishay en fournit un aperçu [2] qui résume les exigences de la norme actuelle. Texas Instruments révèle en [3] la structure de l'impulsion (voir **fig. 2**).

Tout compte fait, on constate que dans le cas d'un système à 12 V, il faut appliquer une impulsion pendant 400 ms, jusqu'à 101 V, avec une résistance interne minimale de 0,5 Ω et une valeur maximale de 4 Ω . Pour un système à 24 V, la durée maximale est de 350 ms, la tension est comprise entre 151 et 202 V, la résistance interne est comprise entre 1 Ω et 8 Ω . Selon Adam Riese, cela se traduit pour un système à 12 V par une puissance momentanée à dissiper de 20.400 W, voire même 40.800 W pour un système à 24 V. Ce qui est fâcheux dans ce contexte c'est le temps de montée (*rise time*) particulièrement rapide d'environ 10 ms. Le cas échéant, les circuits doivent réagir très rapidement, ce qui exclut d'emblée de tout développement les composants tels que les relais.

Tout aussi fâcheux : la nouvelle norme impose aussi au circuit de supporter dix impulsions de ce type en moins de dix minutes. Selon l'ancien standard, il aurait suffi que le système survive à une seule de ces impulsions. Cependant, cette exigence nouvelle est pertinente, car un véhicule avec des bornes de batterie corrodées peut produire des délestages en quantité. Un moyen classique de « protection » contre de telles impulsions consiste, outre le montage en amont d'une capacité énorme et d'une résistance, à insérer un « disjoncteur » à réaction très rapide ; le circuit à sécuriser est alors déconnecté du système électrique en cas d'urgence (**fig. 3**).

Heureusement, l'industrie des semi-conducteurs est bien consciente de la gêne occasionnée par ce problème et propose donc divers composants destinés à combattre les transitoires. La société Littelfuse s'est montrée particulièrement efficace à cet égard. Les diodes dénommées TVS (*Transient Voltage Voltage Suppressor Diode*) et brièvement présentées sous [4] agissent comme une sorte de levier qui limite la tension en cas de doute. STM n'est pas restée inactive non plus. La gamme de diodes de protection appelée « Transil » par STM est subdivisée selon les différentes impulsions de test ISO, ce qui facilite la tâche des développeurs pour la sélection de composants appropriés (**fig. 4**). Vishay appelle ses diodes de protection contre les décharges (ESD) « TransZorb ».

Notez que les diodes Transil sont presque toujours proposées en versions unidirectionnelle et bidirectionnelle. Les diodes Transil unidirectionnelles se comportent comme des diodes zener normales : elles commutent immédiatement si l'impulsion vient de la mauvaise direction. Les diodes Transil bidirectionnelles nécessitent une certaine tension minimale dans les deux sens avant leur activation.

L'auteur utilise volontiers le régulateur à découpage de moyenne tension LM2576HV de la série « Simple Switcher » de TI (qui cependant ne brille pas vraiment en termes de tension de démarrage à froid) afin de faciliter le travail de la diode dans la mesure où elle est polarisée à 35 V par ex., de sorte que la baisse de puissance ne s'avère pas si extrême.

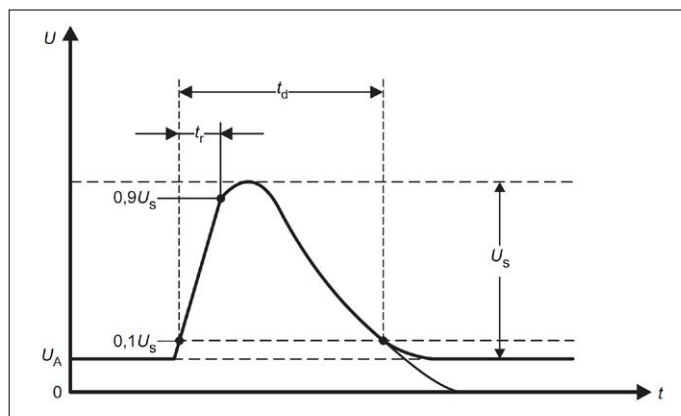


Figure 2. Il n'est pas bon de rencontrer ce pic (source : TI [3]).

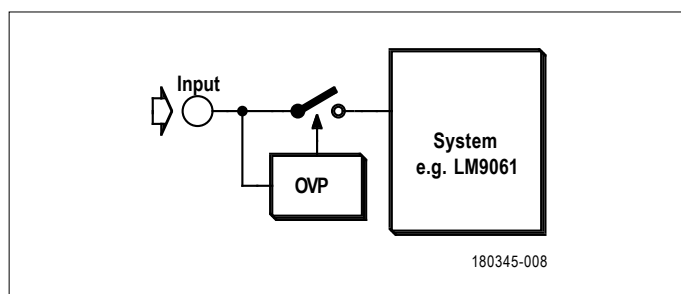


Figure 3. Être un peu lâche peut porter des fruits !

Les diodes TVS ne sont pas les seuls composants qui méritent d'être utilisés pour lutter contre les divers pics de tension. Les varistances (VDR) sont par exemple des résistances bidirectionnelles dépendantes de la tension, qui présentent en principe un comportement semblable à celui d'une diode. Elles diffèrent cependant des TVS par une capacité énergétique beaucoup plus élevée, mais commutent plus lentement et se dégradent avec le temps (bien que Vishay parle de « dégradation négligeable dans la SOA » sous [6] ; SOA = *Safe Operation Area*). À cet égard, la tendance aux pannes dans les varistances à oxyde métallique (MOV) est aussi un problème. L'auteur a connaissance de plus d'un système dont un ou plusieurs rails

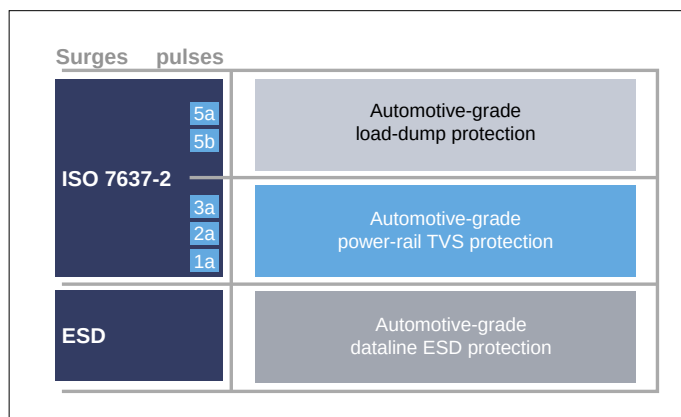


Figure 4. STM subdivise sa famille de Transil selon leur résistance aux décharges électrostatiques (source : STM [5]).

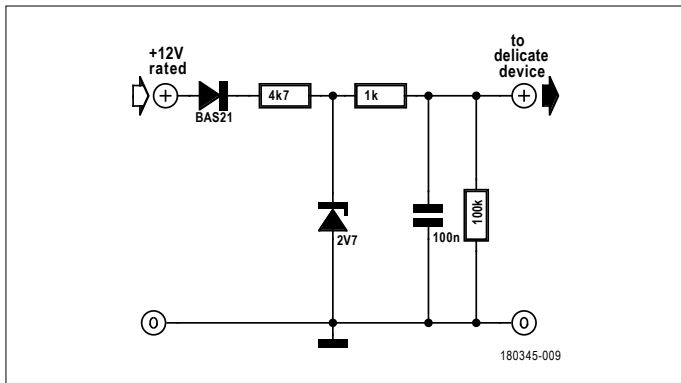


Figure 5. Simple et efficace : un limiteur de tension à diode zener.

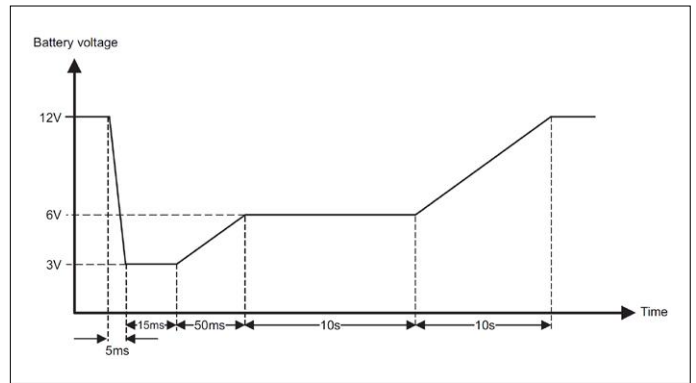


Figure 6. Le *Cold Cranking* peut avoir des conséquences fâcheuses (source : TI).

d'alimentation ont été mis en court-circuit en raison d'une MOV vieillissante.

Par ailleurs, en pratique on observe toujours un couple composé d'une MOV et d'une diode TVS avec une tension de claquage légèrement supérieure. L'idée est de combiner la commutation rapide de la diode TVS et la capacité thermique supérieure de la MOV. Notez que la norme IEC 60950-1 préconise l'utilisation de MOV comme protection contre les surtensions, mais celle de Transil n'est pas autorisée !

Attention tensions

En plus de sécuriser la source de tension principale, vous devriez également protéger de manière exhaustive les composants sensibles ou les entrées, à l'aide de limiteurs de tension. Le classique circuit à diode zener de la **figure 5** n'est certes pas très original, mais parfaitement efficace. Avec cette protection contre les surtensions, il est assez ironique de constater qu'il suffit de s'assurer que la résistance et la diode zener sont capables d'absorber suffisamment de puissance pour littéralement griller les surtensions éventuelles. La résistance doit limiter le courant zener à une valeur sûre, mais ce faisant ne doit pas griller.

En théorie, pour un système à 12 V, on doit s'attendre à un maximum de 14 V, voire à 15 V. Malheureusement, en pratique ce n'est pas le cas, et souvent le problème est lié au « wetware », c'est-à-dire à l'utilisateur. Un grand classique : laisser la batterie de son véhicule se décharger complètement – que ce soit par négligence ou stupidité, puis requérir l'aide au démarrage à partir d'un autre véhicule. Étonnamment, il arrive souvent qu'un camion (avec sa tension de 24 V) fournisse cette aide au démarrage à une voiture sous 12 V. En région anglo-saxonne, il est également très courant d'assurer l'aide au démarrage à l'aide de deux batteries connectées en série. Quoi qu'il en soit, en cours de développement, assurez-vous que vos circuits de protection ne considèrent pas la tension de 24 V du véhicule comme une impulsion « à bloquer » et essayent de la court-circuiter. L'application pendant un temps très court de 24 V ne devrait pas être un problème trop sérieux pour un régulateur à découpage, pour peu qu'il ait été conçu avec un minimum d'intelligence.

Tension : méfiance !

La « neutralisation » des surtensions n'est en fait que la moitié de la solution. Ce ne sont pas seulement les impulsions

parasites qui nous dérangent, mais aussi les chutes de tension pouvant survenir pour diverses raisons. En plus de la mise en marche soudaine d'un « gros » consommateur, il s'agit aussi du démarrage du moteur à combustion par temps froid. Cet état appelé *Cold Crank* en anglais est plutôt à craindre, car il peut entraîner une évolution de la tension du système électrique du véhicule comme illustré à la **figure 6**, à cause de la batterie principale dégradée par le froid.

À première vue, les concepteurs ont tendance à ignorer le démarrage à froid. Le régulateur de tension (tel que le LM2576 mentionné ci-dessus) est prévu pour fonctionner aussi à des tensions d'environ 8 V. Cependant, selon le niveau de criticité du système, l'ISO spécifie des tensions pouvant aller jusqu'à 3 V. Plusieurs méthodes pour lutter contre ce problème : tout d'abord, l'utilisation d'un condensateur de capacité la plus grande possible, qui par ailleurs amortit aussi la courbe de montée en cas d'augmentation rapide de la tension. La deuxième option consiste à concevoir votre propre système de manière à ce qu'il « se bloque » simplement en cas de *Cold Crank*, puis commence à être actif dès que la tension est suffisante. Cependant cette approche n'est pas recommandée, car en cas de calculateurs de type Unix, un temps long de démarrage entraîne des problèmes considérables.

Une autre solution non moins intéressante est l'utilisation de la technologie dite *Buck Boost*. Il s'agit de régulateurs à découpage qui peuvent fonctionner à la fois en élévateurs et abaisseurs. Texas Instruments fournit des informations complémentaires dans sa note d'application 728 [7]. Et vous n'avez plus à vous soucier du démarrage à froid si le régulateur est réellement capable de fonctionner avec une tension d'entrée de 3 V.

La bataille des impulsions négatives

Même si en tant que concepteur cela peut théoriquement vous faire rire, vous ne rirez plus si un utilisateur ou un mécanicien connecte par erreur la batterie dans le mauvais sens, cela arrive ! Heureusement, il n'est pas vraiment difficile de se protéger contre la stupidité, pas besoin d'être un génie pour protéger son système à l'aide d'une diode, comme l'illustre la **figure 7**. Cependant, dans la pratique, il faut tenir compte du fait que les exigences imposées à cette « diode de protection » sont loin d'être minimales. Un véhicule est en fait non seulement une connexion en série de termes en e^{-t} , mais également un ensemble de différentes charges inductives qui produisent divers pics lors de l'allumage et de l'extinction.

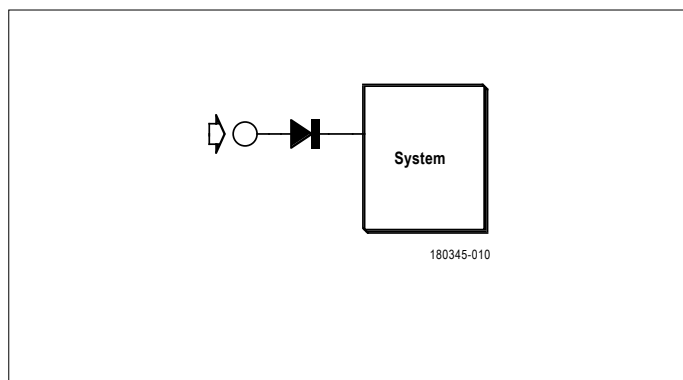


Figure 7. La solution peut être simple !

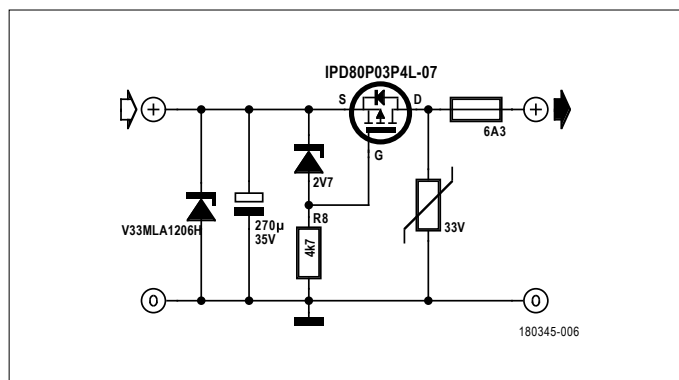


Figure 8. Une manière originale d'utiliser un MOSFET !

Outre les pics positifs pour lesquels des diodes TVS sont utilisées, il existe des pics pouvant atteindre jusqu'à -200 V (!) sur l'entrée du circuit électronique. Il faut donc être extrêmement prudent dans le choix de la diode. Pour l'alimentation principale, l'auteur a fait une bonne expérience avec les S5BC-13-F de Diodes Incorporated, tandis que la BAS20 s'est avérée très utile pour la protection des atténuateurs de tension et circuits similaires.

Les diodes classiques n'ont depuis longtemps plus le monopole de ce scénario d'application. En raison de leur tension de seuil élevée, il est souhaitable pour des courants élevés de se tourner vers des composants plus efficaces. Les concepteurs de circuits à transistors ont trouvé là un nouveau champ d'applications pour les transistors à effet de champ à canaux P et N. La **figure 8** montre un schéma de tels circuits.

Par ailleurs...

Cet article se limite aux systèmes à 12 ou 24 V. En raison de la complexité croissante de l'électronique des véhicules, certains sont désormais équipés d'un circuit électrique à 48 V. Les spécifications ne sont pas encore complètement fixées ; en cas de doute, contactez le fabricant du véhicule.

L'électronique automobile n'est pas vraiment un marché de niche : ne vous étonnez donc pas de la profusion de publications sur le sujet. Les normes ISO mentionnées dans l'introduction sont certes les sources traditionnelles, mais sont difficiles à obtenir. En Autriche, vous pouvez accéder à certaines

d'entre elles à l'Institut de normalisation, mais vous devrez tenir compte du long voyage vers la capitale. En Allemagne, on trouve un grand nombre de points d'information sur les normes, principalement dans les universités, mais aussi dans les administrations et instituts. En France, il est possible de consulter gratuitement certaines normes ISO, EN et NF dans les délégations régionales et les centres partenaires de l'AFNOR [8]. Une alternative intéressante aux normes standard est proposée par le constructeur américain Ford sous [9]. C'est un cahier des charges destiné aux fournisseurs souhaitant produire des composants pour le marché américain. Mais il est également intéressant pour les concepteurs en ce sens qu'il est pratiquement complet et autonome, et donc rend superflue la lecture de l'un ou l'autre des standards.

Comme dit dans l'introduction : il existe des milliers de systèmes qui, tout en fonctionnant sans problème, ignorent totalement certaines, voire la totalité des règles discutées ici. En revanche vous assurerez la sécurité de votre projet en le protégeant à l'aide une diode TVS et quelques autres composants. **■**

(180345-04 – version française : Xavier Pfaff)

Liens

- [1] Norme ISO 7637-2 : www.smd.ru/upload/medialibrary/a3d/iso_7637_2.pdf
- [2] Protection contre la perte de charge : www.vishay.com/docs/49748/49748.pdf
- [3] Protection contre la perte de charge et démarrage à froid : www.ti.com/lit/an/snva681a/snva681a.pdf
- [4] Diodes TVS de Littelfuse Automotive : www.littelfuse.com/~media/electronics/application_notes/littelfuse_tvs_diode_automotive_circuit_protection_using_automotive_tvs_diodes_application_note.pdf.pdf
- [5] Composants de protection dans l'automobile : www.st.com/content/st_com/en/products/protection-devices/automotive-protection-devices.html
- [6] 'Transient Suppressors, a Competitive Look', note d'application : www.vishay.com/docs/88444/tvscomp.pdf
- [7] 'Output Voltage Regulation During Automotive Cold-Crank', rapport d'application : www.ti.com/lit/an/snva728/snva728.pdf
- [8] AFNOR : www.afnor.org
- [9] 'Electromagnetic Compatibility Specification', Ford : www.fordemc.com/docs/download/FMC1278.pdf

lampes à LED et parasites

lampes conformes aux normes de CEM ?

Johan Jongbloed (Pays-Bas)

Championnes de l'économie d'énergie, les LED illuminent actuellement notre monde. Or, en septembre 2017, l'association des radioamateurs allemands a indiqué que les LED seraient la principale cause des interférences croissantes dans les réceptions radio. Dès mars 2018, Alfred Rosenkränzer attirait l'attention de nos lecteurs sur la « pollution électromagnétique causée par les lampes à LED » avec les résultats de son enquête ainsi qu'une procédure de test CEM facile. Elektor se devait d'approfondir la question.



En avril 2018, nous avons demandé à nos lecteurs d'envoyer à la rédaction des lampes à LED inconfortables aux fins d'investigation. Notre appel a été entendu. Pour tester les lampes à LED reçues conformément à la réglementation européenne, le comité EMC-EMF de VERON (association néerlandaise de recherche expérimentale sur la radio) a proposé de soumettre ces lampes à un test préalable de conformité CEM (*pre-compliance*).

Mais d'abord, directive et normes de CEM, de quoi s'agit-il ? La directive sur la **compatibilité électromagnétique** (CEM 2014/30/UE) prescrit d'une part que les appareils ne doivent pas être perturbés outre mesure (devoir de résilience) par l'environnement dans lequel ils sont placés. Elle définit d'autre part la règle par rapport aux **émissions** comme suit :

Les perturbations électromagnétiques produites par un dispositif doivent être limitées à un niveau permettant aux autres appareils de fonctionner comme prévu.

Comment faut-il l'interpréter ? Si une radio ne fonctionne plus convenablement au sens de la directive, si elle ne peut plus capter les stations voulues, il y a brouillage, elle est clairement perturbée. Mais faut-il dans ce cas tenir compte d'une distance minimale entre la source de l'interférence et la radio ?

Les limites et les méthodes de mesure ont été déterminées dans les années 70 en pensant aux seules sources de parasites de l'époque, comme les moteurs à collecteur (aspirateurs et moulins à café) et l'allumage des moteurs à combustion interne. En ce temps-là, il n'y avait pas d'ordinateur, de téléphone mobile, d'alimentation à découpage, d'éclairage à LED ni d'installations domestiques à panneaux solaires. On se fondait sur le principe que la source était chez le voisin ou en tout cas hors d'atteinte et qu'il n'existait aucun moyen de la réduire au silence. On a donc décidé dans les années 70 d'une distance de 10 m entre un appareil et la source de brouillage.

Envisager une approche évolutive ?

Cela n'a pas été vraiment le cas. Les temps changent, mais les normes stagnent depuis cette époque. Or, il y a actuellement tellement de sources de parasites sous un même toit qu'un radioamateur a bien du mal à communiquer sur les ondes, même avec une bonne antenne extérieure. Ses propres sources engendrent déjà environ 70 % des griefs sur les perturbations. Mais celles-là, vous pouvez vous en débarrasser vous-même, celles du voisinage pas ! De plus, les sources d'interférences actuelles n'agissent pas momentanément, comme le moulin à café ou l'aspirateur, mais sévissent souvent en continu, surtout quand on a oublié de les éteindre !

Que font les comités des normes CEM ?

Les comités sur la CEM n'ont jusqu'à présent pas anticipé la progression des perturbations dans les communications radio aussi vite que la progression actuelle, laquelle est arrivée insensiblement, tout en passant inaperçue. Ce sont d'abord les radioamateurs qui ont tiré la sonnette d'alarme. Ils opèrent souvent dans le bruit ou juste sous son niveau et sont donc au premier rang à s'inquiéter de l'état du spectre de fréquences. La réglementation sur la CEM présume aussi que tout produit doit être conforme aux normes, alors que plusieurs produits (trublions) agissent de concert. Et alors, tout comme dans un café animé, le bruit de fond augmente notablement.

Il serait vraiment temps de penser à une norme plus judicieuse qui, par exemple, tiendrait compte de la densité volumique d'utilisation pour abaisser d'un certain nombre de décibels les limites acceptables pour chaque produit.

Pourquoi un test préalable de conformité CEM ?

Pour les fabricants d'appareils, le test préalable de conformité est financièrement avantageux pendant la phase de déve-

loppement, du fait qu'un test complet ou un test formel est nettement plus cher (de 1 000 à 4 000 €). Le *pre-compliance test* permet en outre de détecter à un stade précoce tout manquement éventuel et donne le temps de trouver une solution avant le test complet, le *full compliance test*, ce qui améliore les chances de succès.

Un test préalable de conformité en dit déjà long sur la propension au rayonnement d'un appareil électrique et suffit dès lors à spécifier le niveau de radiation des lampes à LED que nous avons envoyées.

Formes de rayonnement

Nous retenons de ces essais deux formes de puissance irradiée : l'émission par **conduction** (EC) et l'émission par **radiation** (ER). La première est l'énergie électromagnétique produite à l'intérieur de l'appareil, celle qui s'échappe par les câbles du réseau 230 V. L'installation électrique de la maison se comporte alors comme une longue antenne filaire en état d'émettre ces signaux. L'énergie rayonnée (ER) est électromagnétique, souvent produite à l'intérieur de l'appareil et qui se transmet par voie aérienne. Elle provient souvent d'appareils mal conçus ou présentant des défauts. Ceux qui émettent cette énergie peuvent occasionner des embarras à d'autres appareils dans le voisinage immédiat.

La norme EN 55 015, qui est importante pour la puissance radiante, mentionne une autre méthode pour mesurer les émissions rayonnées, elle s'opère alors à l'aide d'un réseau de découplage. Nous l'appelons ici la mesure des émissions par conduction.

Résultats

La mesure de l'énergie émise par conduction (EC) est représentée dans deux domaines du spectre : de 9 kHz à 30 MHz et de 30 à 300 MHz. À chaque bande de fréquence correspond une méthode de mesure particulière. On mesure l'énergie de rayonnement (ER) avec une antenne à distance de 3 ou 10 m de la source de brouillage dans la bande de 30 à 300 MHz.

Comme les perturbations ou les nuisances ne s'occupent pas des frontières théoriques, nous avons spécifié quatre subdivisions. Elles ne découlent d'aucune norme, elles ne font que clarifier la situation. Plutôt que d'indiquer si les lampes sont conformes ou non à la réglementation, nous les classons en trois groupes selon leurs divergences à la norme. Le **tableau 1** donne les résultats des mesures EC.

C'est avec une certaine surprise que nous avons remarqué que le rayonnement de toutes les lampes à LED était inférieur à la limite pour les mesures EC. Dans la gamme de 9 kHz à 30 MHz, une grande partie approchait bien ladite limite, seulement 0 à 5 dB en dessous. Mais pour plus de la moitié, leur rayonnement reculait de plus de 10 dB. En test EC dans la gamme de 30 à 300 MHz, seule une petite partie approche la limite, mais aucune lampe ne la dépasse.

La **figure 1** montre la mesure de l'une des deux lampes qui excèdent la limite en énergie émise par radiation ER. La ligne rouge balise la limite, la verte indique la puissance d'irradiation selon la mesure ER. Dans ce cas-ci, la valeur de pointe du dépassement est supérieure de plus de 10 dB à la valeur maximale admise. C'est en fait l'exception qui confirme la règle : seules deux lampes outrepassent la démarcation et l'une arrive sous les 5 dB plus bas, les autres se comportent si bien qu'il n'y avait finalement aucune raison de les mesurer.

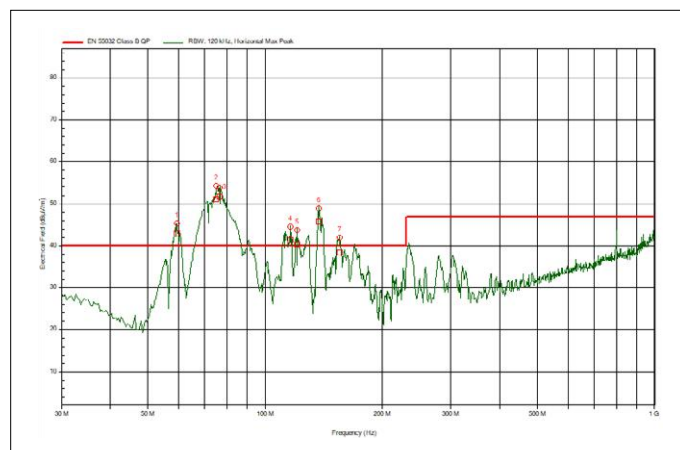


Figure 1. Mesure d'énergie rayonnée ER sur une lampe à LED dont il est clair qu'elle outrepassa la norme.

Tableau 1. Mesures (énergie émise par conduction) sur les lampes à LED

CE 0,009 - 30 MHz	37,5%	0-5 dB sous la limite
CE 0,009 - 30 MHz	6,25%	5-10 dB sous la limite
CE 0,009 - 30 MHz	25%	10-15 dB sous la limite
CE 0,009 - 30 MHz	31,25%	>15 dB sous la limite
CE 30 - 300 MHz	6,25%	0-5 dB sous la limite
CE 30 - 300 MHz	12,5 %	5-10 dB sous la limite
CE 30 - 300 MHz	12,5 %	10-15 dB sous la limite
CE 30 - 300 MHz	68,75%	>15 dB sous la limite

rer. Un certain nombre de lampes n'ont même pas révélé la moindre augmentation du bruit de fond.

Conclusion

Bien que l'épreuve demandée se limite à la conformité CEM préalable, il nous semble bien que la majorité des lampes à LED répondent aux exigences. Mais alors, d'où provient le brouillage ? S'agit-il de circonstances particulières, comme un renforcement de l'action comme antenne des câbles à 230 V ? Est-ce dû à un rassemblement de nombreuses lampes à LED, ce qui multiplie les parasites ? Faut-il incriminer une trop grande sensibilité des appareils en certaines circonstances ? Ou bien les exemplaires fournis n'étaient-ils pas assez nombreux ou représentatifs ? Si vous le savez, vous pouvez le communiquer à redaction@elektor.fr. ◀

(180608-04 – version française : Robert Grignard)

Lien

Page de l'article : www.elektormagazine.fr/180608-04



@ **WWW.ELEKTOR.FR**

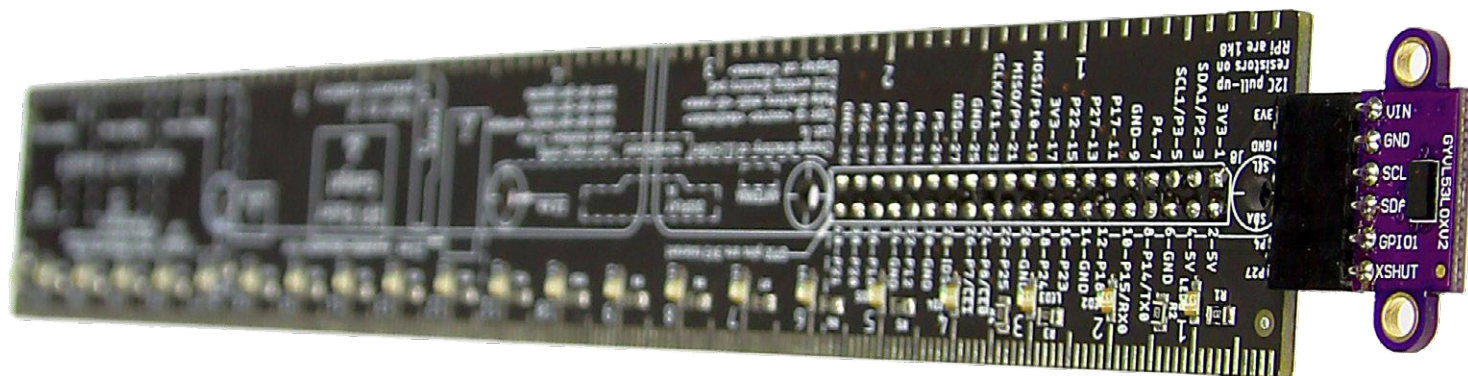
→ Livre en anglais

'EMC for Product Designers (5th Edition)'

www.elektor.fr/emc-for-product-designers-5th-edition

règle-gadget Raspberry Pi

s'amuser avec un capteur de temps de vol



Clemens Valens (labo d'Elektor)

Cette règle est un gadget destiné à être offert sur les salons et autres événements similaires auxquels participe Elektor. Toutefois elle peut mesurer électroniquement des distances allant jusqu'à deux mètres grâce à un capteur de temps de vol (ToF) VL53L0X.

Caractéristiques

- Beaucoup de détails pratiques sur le RPi
- Capteur de temps de vol (ToF) VL53L0X
- 22 LED
- Connecteur I2C
- Règle (version simple) offerte aux visiteurs des stands Elektor

Si vous avez visité notre stand sur le salon *electronica* à Munich (Allemagne) ou *World of Technology and Science* à Utrecht (Pays-Bas), vous êtes peut-être reparti avec un exemplaire de cette règle et vous aurez remarqué qu'elle comporte des pastilles de soudure ainsi que de la place pour deux connecteurs. C'est parce qu'il est possible d'y installer 22 LED (en boîtier CMS 0805) avec des résistances de limitation du courant (boîtier 0805 ; 1 kΩ conviendra), une embase d'extension Raspberry Pi à 40 contacts (d'au moins 16 mm de haut pour un RPi « normal »), et une barrette mâle ou femelle I2C à 6 contacts (ou des fils, si vous préférez) – voir la **figure 1**.

Telle quelle, la règle RPi a au mieux une utilité limitée. Elle fournit néanmoins de nombreuses informations mécaniques et électriques sur le Raspberry Pi, et elle sert bien de règle. Vous pouvez aussi la transformer en un véritable gadget de haute-technologie si vous installez tous les composants, câblez un capteur de temps de vol (ToF) VL53L0X sur le connecteur I2C, et bien sûr enfichez la règle sur votre RPi.

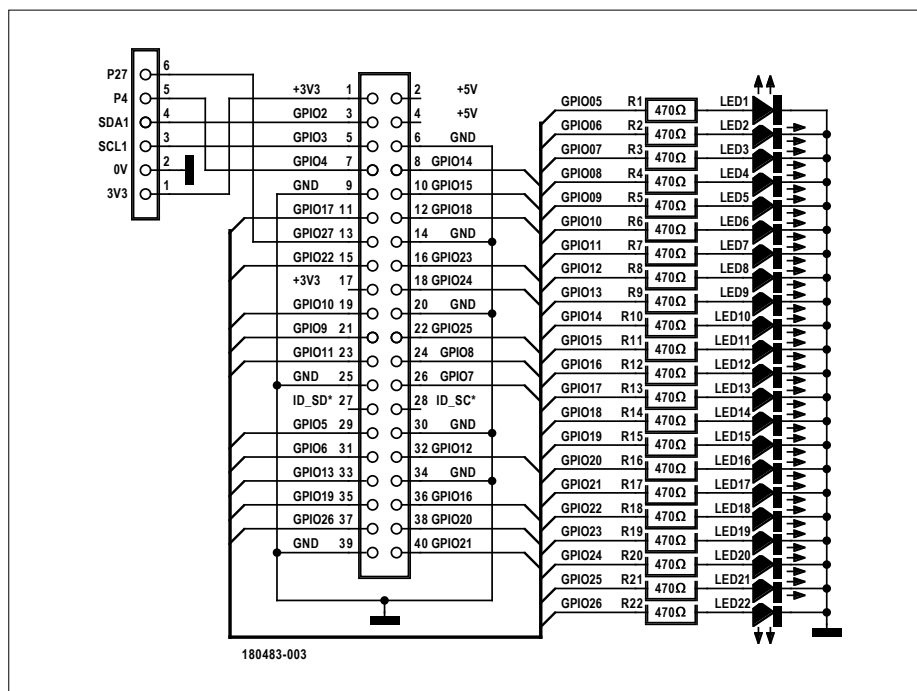
Capteur de temps de vol (ToF) VL53L0X

Le capteur VL53L0X fabriqué par ST, utilisé dans ce projet, est assez étonnant. Sa fiche technique indique que le module de télémétrie laser est logé dans le plus petit boîtier du marché (c'était en 2018). Il permet une mesure précise de distance indépendamment de l'indice de réflexion de la cible, et peut mesurer des distances absolues jusqu'à deux mètres. Le capteur n'estime pas la distance de l'objet par la mesure de la lumière qu'il réfléchit – cette quantité dépend fortement de la couleur et de la surface de l'objet – mais par la mesure précise du temps mis par la lumière pour atteindre l'objet et revenir (le temps de vol en quelque sorte).

Diode à avalanche à photon unique

La lumière réfléchie est détectée par une diode à avalanche à photon unique ou SPAD (*Single-Photon Avalanche Diode*). C'est un type de diode similaire à une photodiode à avalanche (APD) « normale », sauf qu'un seul photon incident suffit à la déclencher (avalanche) et produire un courant de quelques milliampères. Lorsque l'avalanche s'arrête, la SPAD peut se rétablir et détecter un second photon. La principale différence entre une SPAD et une APD est que cette dernière est un amplificateur linéaire pour les signaux optiques alors que la première est un dispositif de déclenchement.

Deux mètres, c'est le mieux que puisse détecter le VL53L0X, et dans les applications réelles la distance maximale dépend des conditions de luminosité ambiante et de la surface réfléchissante sur laquelle il est pointé. En intérieur, on peut facilement atteindre un mètre ; peut-être un peu moins en extérieur. La précision du capteur est supérieure à 10% dans la plupart des cas, mais encore une fois, cela dépend des conditions d'utilisation. Les données produites par le VL53L0X



INFOS SUR LE PROJET

Raspberry Pi

VL53L0X

ToF capteur

débutant

connaisseur expert

env. 1 h.

soudage (CMS), Raspberry Pi, Internet

env. 25 €

Figure 1. Le schéma de la règle RPi n'est constitué que de 22 LED avec des résistances de limitation de courant et de deux connecteurs.

sont accessibles au travers d'une interface I2C (c'est confortable).

Carte de liaison pratique

Cet intéressant module a rapidement été ciblé par les fournisseurs (Chinois ?) de cartes de liaison et se trouve aisément sur l'internet dans un format pratique (**fig. 2**). Le fabricant a consacré beaucoup d'efforts à l'écriture du logiciel pour le capteur, ce qui, en association avec les cartes de liaison, a permis la création de plusieurs bibliothèques de pilotes à code source ouvert pour Arduino et Raspberry Pi.

Règle RPi électronique

Dans ce projet, les LED de notre règle RPi nous serviront à afficher la distance. Comme il n'y a que 22 LED sur la règle, la lecture est limitée à 22 cm. Heureusement, on peut concevoir des formats d'affichage plus évolués avec le multiplexage des LED. C'est pourquoi j'ai monté des LED rouges pour les valeurs 10 et 20, et des LED vertes pour le reste. Notez que la distance en millimètres est aussi affichée sur un écran relié au port HDMI du RPi. Enfin ce projet peut être construit sans la règle, simplement en connectant le VL53L0X au port I2C du RPi.

Le logiciel a été écrit sur un Raspberry Pi 3 modèle B V1.2, mais il devrait fonctionner sur d'autres RPi avec connecteur d'exten-

sion à 40 broches. Enfichez la règle sur le RPi avant la mise sous tension.

Installation du logiciel sur le RPi

Le pilote du VL53L0X a été écrit en C++, c'est pourquoi le logiciel pour ce projet est aussi en C++. Des bibliothèques Python existent aussi pour le capteur, donc si vous préférez continuer en Python, vous pouvez. Juste pour être clair, les étapes suivantes sont effectuées sur notre Raspberry Pi, pas sur un ordinateur Windows, Linux ou macOS.

- Téléchargez et installez **wiringPi** [1]. Cette bibliothèque sert à accéder aux broches GPIO du RPi en C++, on peut donc commander les LED.
- Téléchargez et décompressez le pilote **STSW-IMG005** du VL53L0X depuis ST, dans la section « Get Software » de [2].
- Téléchargez et décompressez l'adaptateur **VL53L0X_rasp** pour Raspberry Pi de Cassou (Mathieu Cassard) [3].
- Téléchargez le fichier **Makefile** depuis [4] et copiez-le dans le dossier [VL53L0X_rasp](#) (remplacez le fichier existant).
- Téléchargez le fichier [rpi_ruler.c](#) depuis [4] et copiez-le dans le dossier [VL53L0X_rasp/examples](#).

En supposant que vous avez suivi ces étapes et que vous travaillez dans votre répertoire d'accueil, saisissez ceci :

```
cd VL53L0X_rasp
make API_DIR=~/.VL53L0X_1.0.2/
make examples API_DIR=~/.VL53L0X_1.0.2/
```

Il ne devrait y avoir ni avertissements si erreurs.

Pour lancer le programme de la règle RPi, tapez :

```
./bin/rpi_ruler
```

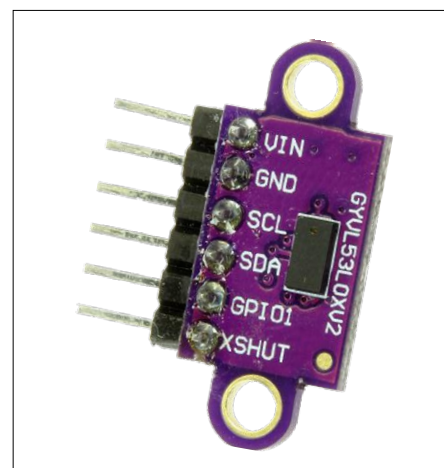


Figure 2. Une carte de liaison (eBoB) particulièrement répandue pour le capteur de temps de vol VL53L0X.

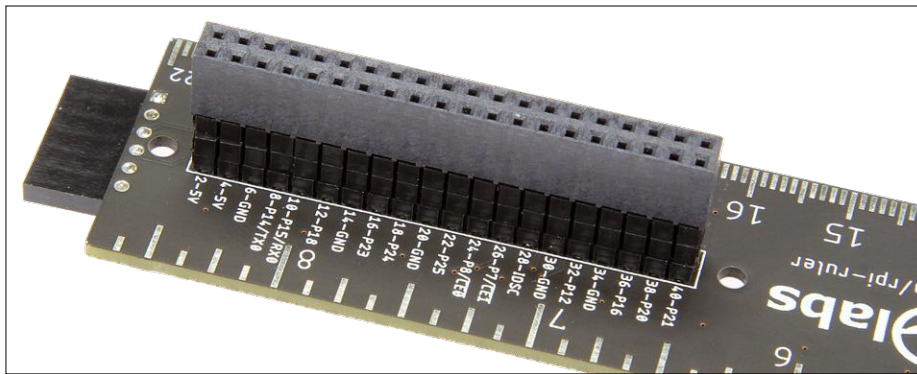


Figure 3. Pour un branchement sur un Raspberry Pi « normal », ce connecteur doit faire au moins 16 mm de haut.

Affichage des résultats

Avec un écran connecté au RPi, vous devriez voir défiler les mesures. Agiter votre main ou un objet devant le capteur devrait produire une distance plausible en millimètres. S'il n'y a rien devant le capteur, la valeur affichée est 8192 (c.-à-d. hors de portée). Si un objet se trouve à moins de 22 cm du capteur, la distance est indiquée par les LED.

Dans ce cas, comme la distance est rarement un nombre entier de centimètres, deux LED peuvent être allumées en même temps. Leur luminosité est fonc-

tion de la valeur décimale, la LED la plus proche de la valeur entière étant la plus brillante. Exemple : avec une distance mesurée de 12,4 cm, la LED12 aura une luminosité de 60% et la LED13 40% (bien sûr, pour une lisibilité optimale, il faut utiliser une échelle logarithmique au lieu de linéaire). La luminosité est commandée par la fonction `softPwmWrite` disponible dans `wiringPi`.

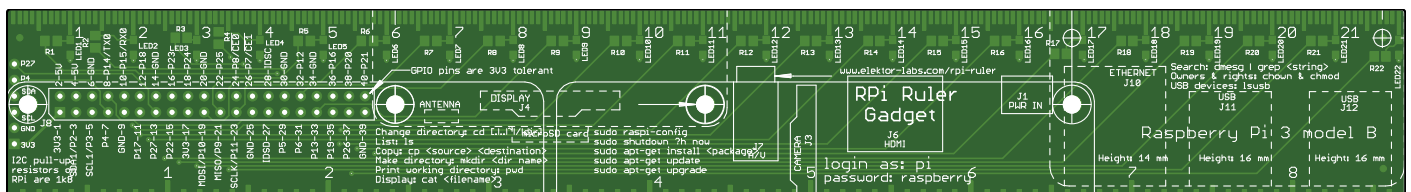
Notes

Pour de meilleurs résultats et un bon confort d'utilisation, veuillez lire attentivement ce qui suit :

- Utilisez une embase de 16 mm de haut (ou plus) comme connecteur d'extension pour le Raspberry Pi, car la règle doit passer par-dessus le connecteur USB du RPi (**fig. 3**). Cette remarque ne concerne pas le Raspberry Pi Zero qui n'a pas de connecteurs hauts.
- On peut alimenter le capteur VL53L0X avec une alimentation de 2,6 à 3,5 V. Toutefois, les cartes de liaison pour VL53L0X intègrent souvent un régulateur de tension à faible perte (LDO) qui permet l'utilisation du classique 5 V. Le connecteur I2C de la règle ne fournit que du 3,3 V. Ce n'est pas un problème, car beaucoup de LDO ne font que transférer la tension lorsqu'elle passe en dessous du minimum de leur tension d'entrée. Si pour une raison quelconque vous aviez vraiment besoin de 5 V, prélevez-les simplement sur le connecteur d'extension du RPi.
- Votre capteur VL53L0X peut avoir un film protecteur collé dessus (souvent jaunâtre). Si tel est le cas, retirez-le avant de vous servir du capteur. ◀

(180483-03 -

version française : Denis Lafourcade)



LISTE DES COMPOSANTS

Résistances

R1-R22 = 1 kΩ, 0805

Semi-conducteurs

LED1-LED22 = vertes, 0805

Divers

Barrette mâle à 40 contacts (2×20)
au pas de 2,54 mm, 16 mm de haut
(par ex. Samtec ESQ-120-33-T-D)
Barrette femelle à 6 contacts,
au pas de 2,54 mm
Module capteur ToF VL53L0X
Circuit imprimé, réf. 180483-1
(www.elektor.fr)



@ WWW.ELEKTOR.FR

→ Règle-gadget RPi, réf. 180483-1
www.elektor.fr/rpi-ruler-kit

→ Kit de composants, réf. 180483-71
www.elektor.fr/rpi-kit-180483-71

→ Capteur de temps de vol (ToF), réf. 18713
www.elektor.fr/tof-distance-sensor

Liens

- [1] Bibliothèque wiringPi : <http://wiringpi.com/>
- [2] Pilote VL53L0X officiel : www.st.com/en/embedded-software/stsw-img005.html
- [3] Pilote VL53L0X pour RPi : https://github.com/cassou/VL53L0X_rasp
- [4] Makefile et programme principal : www.elektormagazine.fr/labs/rpi-ruler

parlez-vous I/Q ?

phase et quadrature de phase : comprendre les modulateurs I/Q

Robert Lacoste
(Chaville)

Dans ce nouvel article de la série « Hors Circuits », Robert Lacoste explique sans d'obscures formules mathématiques le fonctionnement du mélangeur en quadrature, appelé aussi démodulateur I/Q. Il aborde également ses avantages et ses inconvénients.

Dans le précédent article, je vous ai parlé de mélangeurs de fréquences, c'est-à-dire de circuits qui permettent de décaler la fréquence d'un signal vers le haut ou le bas, et ce sans modifier les caractéristiques du signal, notamment sa modulation. Aujourd'hui, je continue dans le même esprit en vous expliquant ce qu'est un « mélangeur I/Q ». Pour vous convaincre de l'importance du sujet, ouvrez n'importe quel ouvrage récent sur le traitement du signal ou l'architecture radio, vous trouverez ces deux lettres 'I' et 'Q' à toutes les pages. Malheureusement, même certains électroniciens expérimentés sont réfractaires à ces notions. Pourquoi ? Probablement parce qu'elles sont en général expliquées avec des expressions mathématiques rébarbatives : nombres complexes, théorème d'Euler, transformée de Fourier complexe, etc. Mon objectif est donc de vous expliquer ce qu'est une représentation I/Q sans maths compliquées, et surtout de vous montrer pourquoi c'est si utile. Promis, les seules expressions mathématiques que j'utiliserai sont du niveau « collège ». Alors pas de stress, et suivez-moi dans cette aventure dans le monde des I/Q, ça vaut le coup !

Petits rappels sur les mélangeurs

Vous vous rappelez ce qu'est un mélangeur de fréquence ? Fondamentalement, c'est un multiplicateur de tension. Prenons l'exemple d'un mélangeur abaisseur de fréquence, utilisé dans un récepteur. Il multiplie deux signaux : le signal radiofréquence

(RF) venant de l'antenne et un signal sinusoïdal produit par un oscillateur local (OL). Le signal de sortie est baptisé « fréquence intermédiaire » (FI). Comme expliqué dans le précé-

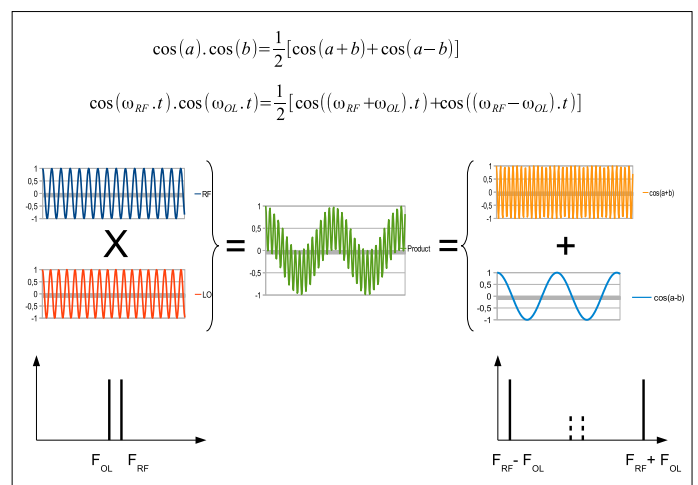


Figure 1. Un mélangeur fonctionne en multipliant deux signaux sinusoïdaux de fréquences F_{RF} et F_{OL} . Grâce à la formule trigonométrique bien connue rappelée ici, le produit est la somme de deux autres signaux sinusoïdaux, respectivement aux fréquences $F_{RF} + F_{OL}$ et $F_{RF} - F_{OL}$.

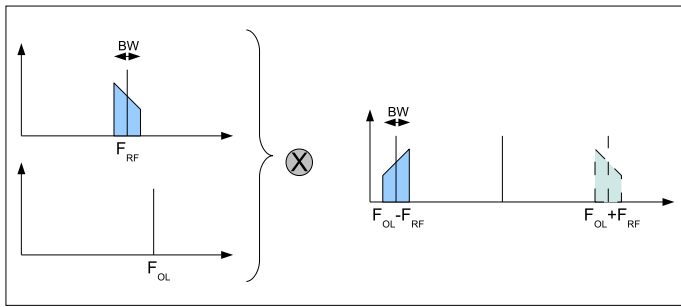


Figure 2. Un mélangeur classique, lorsqu'il est utilisé comme abaisseur de fréquence, produit deux copies du spectre d'entrée, dont une doit être éliminée par filtrage.

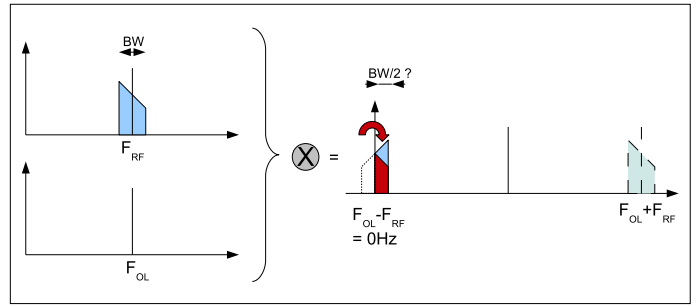


Figure 3. Avec un mélangeur standard, des problèmes surviennent lorsque la fréquence LO est trop proche de la fréquence RF. Le signal de sortie franchit alors la limite de 0 Hz et son spectre est replié du continu à la moitié de la bande passante, ce qui écrase son contenu.

dent article, la magie se cache derrière une simple formule trigonométrique : le produit de deux signaux de fréquences F_1 et F_2 est la somme de deux autres signaux, de fréquences respectives $F_1 - F_2$ et $F_1 + F_2$. Un filtre permet d'isoler l'une ou l'autre de ces fréquences. La **figure 1** illustre clairement ce qui se passe, examinez-la attentivement et relisez mon précédent article si ce n'est pas clair. Bien sûr, dans la vraie vie, un mélangeur est un peu plus complexe que ça, mais cette description suffira pour ce que je veux vous expliquer ici.

Et avec des signaux modulés ?

Passons maintenant aux signaux modulés. Supposons que le signal RF n'est pas une simple onde sinusoïdale, mais un signal qui occupe un canal de largeur BW Hz. La bande utile s'étale alors autour de la fréquence centrale F_{RF} du signal, plus exactement de $F_{RF} - BW/2$ à $F_{RF} + BW/2$. Allez, supposons qu'il s'agit d'un signal Wi-Fi de type 802.11g, émis sur le canal 6. Nous aurons alors $F_{RF} = 2\,437$ MHz (croyez-moi, c'est la fréquence centrale du canal 6 du Wi-Fi) et $BW = 20$ MHz (la largeur de modulation en 802.11g). Donc la bande occupée sera égale à $2\,437$ MHz ± 10 MHz.

Supposons maintenant que nous voulions translater ce signal vers une fréquence FI plus basse pour le traiter plus facilement, par ex. à la fréquence $F_{FI} = 50$ MHz. Il suffirait de mélanger le signal RF avec un oscillateur local réglé à une fréquence $F_{LO} = 2\,437 + 50 = 2\,487$ MHz. La sortie du mélangeur comprendra deux copies du spectre modulé, une centrée sur $F_{LO} - F_{RF} = 50$ MHz, qui est bien ce que l'on cherche, et une centrée sur $F_{LO} + F_{RF} = 4\,924$ MHz (**fig. 2**). Un filtre passe-bas enlèvera facilement la seconde. La largeur de bande occupée du signal n'est pas modifiée par le mélangeur : le signal de fréquence intermédiaire occupera toujours 20 MHz autour de la fréquence intermédiaire de 50 MHz, c'est-à-dire de 40 à 60 MHz.

Zéro FI ?

Maintenant imaginez que vous avez connecté un analyseur de spectre sur la sortie FI pour visualiser ce signal large de 20 MHz autour de 50 MHz. Mettez la main sur le bouton de réglage de fréquence de l'oscillateur local. Tournez alors doucement le bouton et réduisez la fréquence LO. Que se passe-t-il ? Regardez à nouveau la figure 2. Plus F_{LO} se rapproche de F_{RF} , plus F_{FI} sera proche de 0 Hz. Théoriquement, tout devrait rester en ordre jusqu'à ce que la fréquence de l'oscillateur local soit de $F_{RF} + 10$ MHz. À cet instant, le signal FI occupera une bande de 20 MHz centrée sur 10 MHz, soit exactement de 0 Hz à 20 MHz.

Que se passe-t-il si vous continuez à réduire la fréquence LO, en la rapprochant encore de F_{RF} ? Une partie du spectre FI passera dans des fréquences... inférieures à 0 Hz. Malheureusement, les fréquences négatives n'existent pas, donc ce spectre sera « replié » du côté des fréquences positives. Continuez et réglez F_{LO} exactement à la fréquence F_{RF} . Le signal FI sera alors théoriquement centré sur $F_{FI} = F_{LO} - F_{RF} = 0$ Hz. La bande occupée sera de 0 Hz à $BW/2 = 10$ MHz... et le signal sera complètement inexploitable puisque les deux parties du spectre seront repliées l'une sur l'autre (**fig. 3**). Donc tout ça ne sert à rien ? Détrompez-vous ! Il y a même un nom pour une telle architecture où l'oscillateur local est calé sur la fréquence du signal RF : on parle d'architecture « zero-IF ». Il doit donc y avoir une astuce pour éviter ce problème de repliement de spectre...

I/Q, au secours !

Évidemment, la réponse est oui : il faut utiliser un mélangeur en quadrature, appelé aussi démodulateur I/Q. Dans cette dénomination, le 'I' signifie 'en phase' et le 'Q' signifie 'en quadrature de phase', c'est-à-dire avec un déphasage de 90° . Le concept peut être expliqué de plusieurs façons. Pour ceux qui apprécient un peu de maths, il suffit de lire l'excellent article de Richard Lyons disponible sur le web (voir [1]). Je vais tenter ici une approche plus intuitive. Le problème est que les fréquences négatives n'existent pas. Cependant, c'est parce que nous n'examinons le signal que sous un seul angle. Que veut dire ce charabia ? Prenons un exemple physique : une façon de produire un sinus est de planter une épingle sur une roue et de faire tourner la roue sur elle-même à vitesse constante. Appelons I et Q les axes verticaux et horizontaux. Supposons que l'on tourne la roue dans le sens inverse des aiguilles d'une montre (**fig. 4a**), la vitesse de rotation étant la fréquence du signal. Pendant cette rotation, déplaçons la roue à une vitesse constante sur son axe, qui représente la ligne de temps. L'épingle décrit une hélice dans l'espace. Maintenant, mesurons la position de l'épingle sur l'axe I au cours du temps : on obtient bien sûr une fonction sinusoïdale (**fig. 4b**). De même, si nous mesurons sa position sur l'axe Q, nous obtenons une autre fonction sinusoïdale, mais décalée de $+90^\circ$ (ces deux mesures sont en effet respectivement le cosinus et le sinus de l'angle de rotation).

Maintenant, un peu de magie. Que se passe-t-il si la fréquence est « négative » ? Invertissons celle-ci en tournant maintenant la roue dans le sens horaire (**fig. 4c**). La courbe I est inchangée, ce qui explique le phénomène de repliement de spectre.

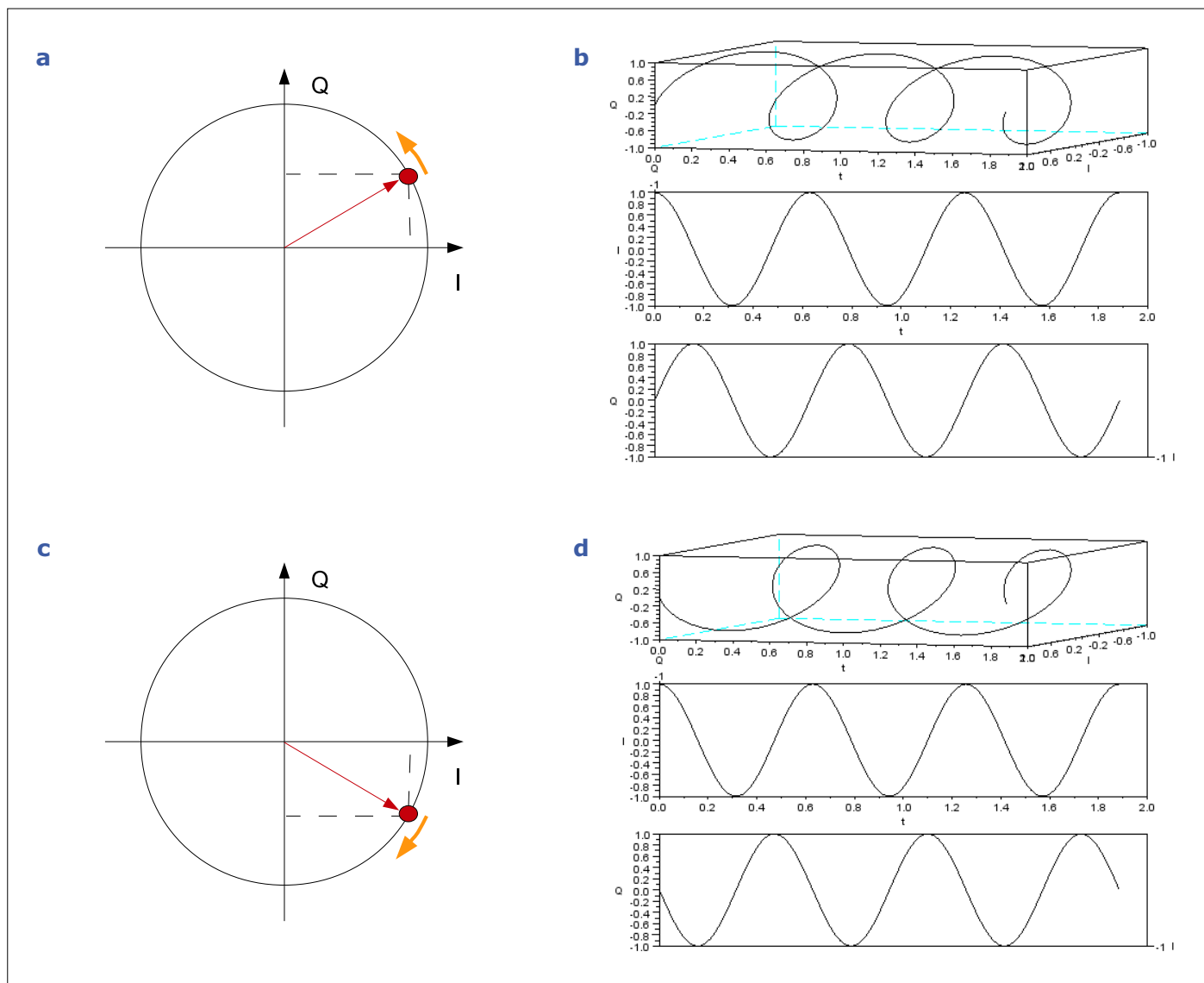


Figure 4. a) Une roue tournant dans le sens antihoraire et b) sa projection sur les axes horizontaux et verticaux. c) La même roue tourne ici dans le sens horaire, simulant une fréquence négative. d) Sa projection sur l'axe horizontal est inchangée, mais le signe de la seconde projection est inversé, ce qui permet de la différencier du premier cas.

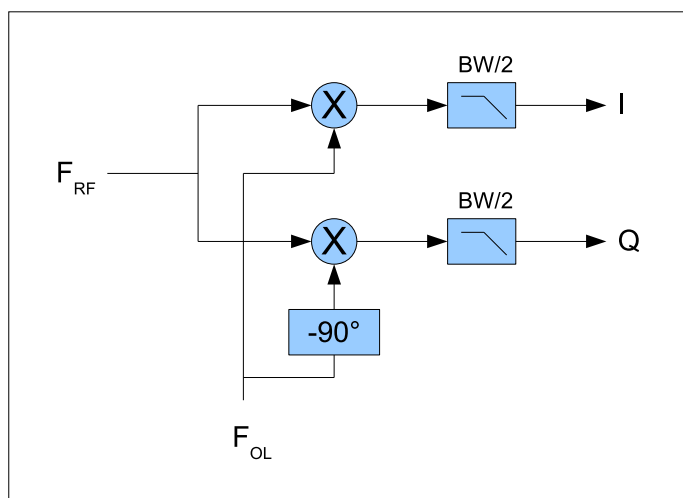


Figure 5. Un démodulateur I/Q comprend deux mélangeurs pilotés par deux copies de l'oscillateur local, l'une décalée de 90° . Les deux sorties, nommées I et Q, sont les signaux en bande de base, chacun allant du continu à la moitié de la largeur de bande du signal entrant.

De même, la courbe Q est toujours sinusoïdale, mais... attendez, son déphasage est maintenant de -90° et non $+90^\circ$ par rapport à I (**fig. 4d**) ! Ainsi, en utilisant simultanément deux projections sur deux axes à 90° , nous avons un moyen de distinguer les fréquences positives et négatives. Vous avez compris tout l'intérêt des architectures dites I/Q.

Bien sûr mon explication n'est qu'intuitive et ne vaut pas une belle démonstration, mais l'idée est qu'utiliser un simple mélangeur ne permet pas de distinguer les fréquences positives et les négatives. Par contre, si nous utilisons deux mélangeurs, chacun alimenté par le même oscillateur local, mais l'un d'eux décalé de 90° , nous aurons assez d'information pour récupérer le contenu du signal. C'est exactement ce qu'on appelle un démodulateur I/Q (**fig. 5**). Les signaux I et Q occupent chacun une plage de fréquence de 0 Hz à la moitié de la largeur de bande ($BW/2$), et sont appelés signaux en « bande de base ».

Un autre point de vue

Il est aussi utile de voir la représentation I/Q du point de vue, disons de la théorie de l'information : le signal RF a une bande

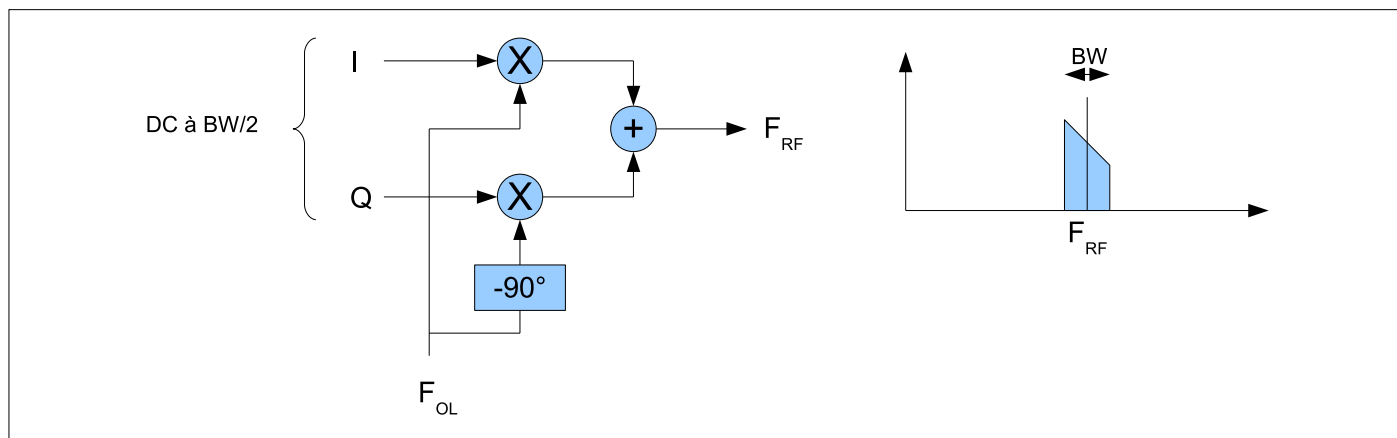


Figure 6. L'architecture d'un modulateur I/Q est très proche de celle d'un démodulateur.

passante de BW Hz. Lorsque nous avons essayé d'utiliser un simple mélangeur en ramenant la fréquence FI à 0 Hz, le signal de sortie occupait la plage de 0 Hz à $BW/2$, donc la moitié de l'information a certainement été perdue. Avec un démodulateur I/Q, nous avons deux signaux, I et Q, chacun ayant une bande passante de 0 Hz à $BW/2$. D'où au total $BW/2 + BW/2 = BW$ Hz, ce qui correspond bien à la largeur, donc à la quantité d'information, du signal de départ, ce qui est rassurant...

Voulez-vous encore un autre point de vue ? Pour ceux qui sont friands d'un peu plus de théorie, les fonctions sinus et cosinus forment une base orthogonale. Chaque signal, projeté sur ces deux axes, donne deux composantes I et Q qui permettent de reconstruire le signal. Si vous regardez à nouveau la figure 4a, vous ne devriez pas être surpris de savoir que l'amplitude du signal RF est la racine carrée de $I^2 + Q^2$, alors que la phase entre le signal OL et le signal RF est $\arctan(I/Q)$. Rien de plus que le théorème de Pythagore et de la trigonométrie de niveau 3^e, je vous l'avais promis !

Enfin, certains d'entre vous savent sûrement ce que sont les Transformées de Fourier (vous pouvez aussi vous référer à l'un de mes anciens articles sur le filtre numérique [2]). Si c'est le cas, vous vous rappelez peut-être que la transformée de Fourier d'un signal sinusoïdal réel a une composante dans les fréquences positives, et son conjugué complexe dans les fréquences négatives, alors que la transformée de Fourier d'un

sinus complexe (exponentielle complexe) a une seule composante. Si vous y réfléchissez à deux fois, ou si vous lisez l'article de Richard Lyons, vous comprendrez que cela est étroitement lié à la magie de l'approche I/Q...

Les modulateurs I/Q

Les signaux en quadrature sont également largement utilisés pour les applications de conversion montante, par ex. pour réaliser un émetteur. On utilise alors un modulateur I/Q, dont l'architecture est illustrée à la **figure 6**. Fondamentalement, c'est la même chose qu'un démodulateur I/Q, mais utilisé en sens inverse. Ici, deux signaux en bande de base I et Q sont fournis en entrée du modulateur, souvent par un processeur de traitement du signal. Ce sont deux signaux bien réels, chacun avec une bande passante du continu à $BW/2$. Le modulateur I/Q les traduit en un spectre RF de largeur de bande BW centré autour de la fréquence de l'OL.

Quels sont les avantages d'un tel modulateur I/Q par rapport à une architecture plus traditionnelle comme un émetteur hétérodyne à plusieurs étages ? Tout d'abord, cela simplifie considérablement le filtrage. Pourquoi ? Imaginons que nous ayons, comme dans l'exemple précédent, un signal FI centré à 50 MHz, et que nous voulions le translaté à 2,4 GHz en utilisant un mélangeur classique. Nous aurions besoin d'utiliser par ex. un oscillateur local à 2 450 MHz. Sa sortie délivre-

Sources et liens

- [1] 'Quadrature Signals: Complex, But Not Complicated', Richard Lyons: www.ieee.li/pdf/essay/quadrature_signals.pdf
- [2] « Hors-Circuits - Le filtrage numérique sans stress – Les filtres FIR », Elektor 01-02/2016: www.elektormagazine.fr/150635
- [3] « Hors Circuits - Synthèse numérique directe », Elektor 07/2015: www.elektormagazine.fr/150315
- [4] 'I/Q Data for Dummies', Mikael Q Kuisma: <http://whiteboard.ping.se/SDR/IQ>
- [5] 'I/Q experimentation applet', Brent Locher: www.fourier-series.com/IQMod/flashprograms/IQFFT.html
- [6] Simulateur Scilab: <http://www.scilab.org>
- [7] Tableur Calc de Libreoffice: www.libreoffice.org/
- [8] Modulateur analogique ADL5375, '400 MHz to 6 GHz Broadband Quadrature Modulator': www.analog.com/en/rfif-components/modulatorsdemodulators/adl5375/products/product.html
- [9] Modulateur numérique AD9957, '1 GSPS Quadrature Digital Upconverter with 18-Bit IQ Data Path and 14-Bit DAC': www.analog.com/en/rfif-components/direct-digital-synthesis-dds/ad9957/products/product.html
- [10] Page de l'article: www.elektormagazine.fr/190003-01

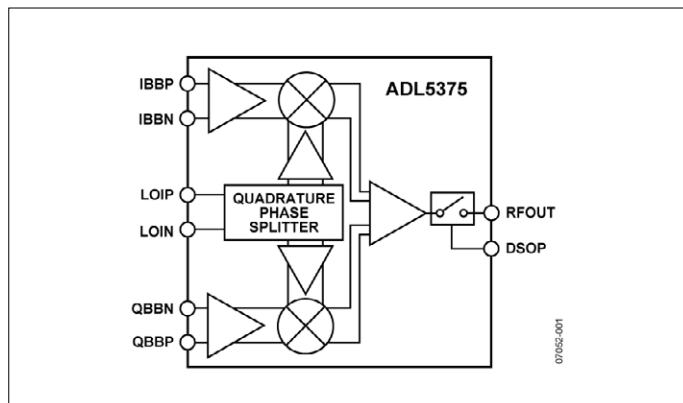


Figure 7. Exemple d'un modulateur analogique I/Q, l'ADL5375 d'Analog Devices

rait le signal à 2 400 MHz désiré ($2420-50$), mais aussi une seconde copie du signal centrée sur $2\,500\text{ MHz}$ ($2450+50$), appelée fréquence image. Un filtre passe-bas très raide serait nécessaire pour éliminer le second signal. Au contraire, un modulateur I/Q ne produit aucune fréquence image, du moins théoriquement : si nous avons deux signaux sinusoïdaux I/Q déphasés de 90° , et qu'on les fait passer à travers un modulateur I/Q avec un oscillateur local à 2,4 GHz, on obtient une seule raie sur le spectre, en dessus ou en dessous de 2,4 GHz selon le signe de la phase et c'est tout.

Cela semble merveilleux, mais évidemment les architectures I/Q présentent aussi leurs inconvénients. Il y en a deux principaux. Le premier, c'est que les composants traitant respectivement les signaux I et Q doivent être très bien appariés en gain comme en déphasage. Si l'une des deux branches a des caractéristiques légèrement différentes de l'autre, la théorie ne se comporte plus aussi bien. Des distorsions et des fréquences images réapparaissent. Le deuxième problème est qu'il y a un petit souci avec une fréquence particulière, 0 Hz. Tout décalage de tension continue ou bruit à basse fréquence sur les signaux I et Q se transforme en un souci à la fréquence centrale du signal RF. Ces deux problèmes ont retardé la généralisation des architectures I/Q pendant des années, jusqu'à ce que les performances du silicium soient suffisantes et que des algorithmes de traitement du signal permettant de compenser ces défauts soient mis au point.

Et les circuits intégrés ?

De nos jours, il existe bien sûr de nombreux circuits intégrés qui facilitent la vie des concepteurs de chaînes I/Q. Ils se déclinent comme toujours en deux versions : analogique ou numérique. La **figure 7** vous montre un exemple de modulateur I/Q analogique, l'ADL5375 d'Analog Devices. Son utilisation se passe quasiment de commentaires : injectez des signaux I, Q et OL sur ses entrées (ici chacun sous forme d'un signal différentiel), et vous obtiendrez sur sa sortie un signal RF, à une fréquence quelconque entre 400 MHz et 6 GHz. Certaines puces ont même un oscillateur local intégré si vous préférez.

La deuxième approche fréquemment utilisée est de concevoir un modulateur I/Q sous une forme entièrement numérique. La **figure 8** vous en montre un exemple avec l'AD9957 de la même société (Analog Devices). Ici, les signaux d'entrée sont des flux numériques I et Q, chacun codé sur 18 bits et avec

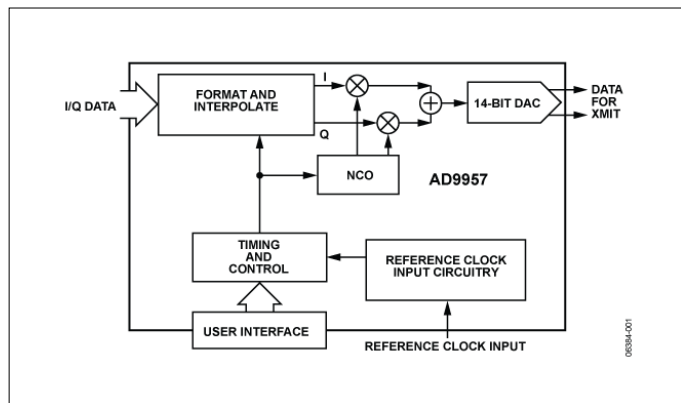


Figure 8. Exemple d'une version numérique avec l'AD9957 d'Analog Devices

un débit pouvant grimper à $2 \times 250\text{ Méc/s}$. La puce comprend un oscillateur local numérique avec sorties en quadrature, qui est ni plus ni moins qu'un oscillateur à commande numérique (NCO, cf. [3]). Le circuit intégré multiplie numériquement le signal OL avec les signaux d'entrées, et convertit le résultat en signal analogique grâce à un CN/A interne rapide à 1 Géc/s et 14 bits. Le résultat ? Un moyen diablement efficace de créer n'importe quel signal de 0 à 400 MHz avec une bande passante de 200 MHz. Quand même impressionnant, non ?

Pour conclure

Nous y voilà. Cet article avait pour objectif de vous donner des bases pour comprendre ce que sont les signaux I/Q et leur intérêt. Vous en utilisez tous les jours puisque tous les téléphones portables et quasiment tous les systèmes sans fil reposent maintenant sur des architectures I/Q, pour des raisons de réduction des coûts, mais aussi de flexibilité.

Ne nous leurrons quand même pas ! Je n'ai fait qu'effleurer le sujet. Je vous encourage chaudement à relire plusieurs fois cet article si vous en sentez le besoin, et surtout à expérimenter par vous-même pour vous convaincre que ça marche vraiment. Faites des petites simulations sur une feuille de calcul ou avec un outil numérique comme Scilab et reproduisez les illustrations de cet article. Produisez deux sinus avec une phase variable et multipliez-les... Vous trouverez également sur le web des animations très intéressantes qui vous aideront à mieux comprendre (voir références et sources). Enfin, je vous invite à lire des articles plus détaillés pour assimiler plus en profondeur, avec un poil plus de mathématiques si vous en avez le courage. Une fois de plus, l'article de Richard Lyons est un incontournable. Et amusez-vous bien !

Ah, avant d'oublier de vous le dire : cette rubrique « Hors Circuits » est aussi la vôtre. Si vous avez des sujets qui vous préoccupent, des réactions ou interrogations, n'hésitez pas à me contacter via la rédaction (redaction@elektor.fr). Je tenterai de vous répondre soit directement soit dans un prochain « Hors Circuits ». ◀

(190003-01)

Cet article a été publié dans la revue Circuit Cellar (n°293, décembre 2014).

s'il vous plaît... dessine-moi un bouton !

les dessinateurs d'Elektor, grands princes du crayon

Jan Buiting, rédacteur de la rubrique Rétronique

De nos jours, qu'ils soient imprimés sur papier ou affichés sur écran, illustrations et textes ont quasiment tous une origine numérique. L'intelligence reste humaine, mais la réalisation est informatique. L'homme de l'art voûté sur sa table à dessin, jonglant entre stylos Rotring, règle et ruban adhésif pour dessiner des circuits imprimés, évolue désormais dans le cliquodrome des logiciels de CAO. Pour des raisons pas seulement nostalgiques, nous avons retrouvé des dessins datant de l'époque où la virtuosité de nos graphistes était encore purement manuelle. Je suis sûr que vous serez d'accord avec moi, certains méritaient bien d'être reproduits ici !

L'an dernier, suite à une nouvelle « réaffectation » des équipes, fonctions et bureaux d'Elektor, il a à nouveau fallu vider quelques placards. C'est dans ceux de l'ancien Service Dessin que mon collègue Patrick Wielders a retrouvé trois lourds classeurs contenant des œuvres d'une époque révolue. Ces classeurs avaient miraculeusement survécu à 25 ans de déménagements, réorganisations et restructurations du Service Dessin, et même échappé à la suppression en 2005 de toutes les archives papier de ce service.

Destins et dessins

Peu après sa naissance en 1961, Elektor embaucha une équipe en charge des dessins « techniques » du magazine. De formation supérieure en électronique, ces jeunes gens (dont quelques filles à l'occasion) apprenaient en interne à respecter les normes touchant à l'aspect pédagogique des schémas. Composé au départ de deux ou trois personnes, le Service Dessin grossit régulièrement

jusqu'à atteindre une dizaine d'employés 15 ans plus tard. C'était vers 1980, et à l'époque personne n'avait droit à son propre bureau, et encore moins à une table à dessin ! Cette équipe comprenait également des maquettistes, un artiste chargé de la couverture, ainsi que des dessinateurs affectés à des éléments plus illustratifs tels des vues éclatées et, reconnaissons-le, à des choses plus frivoles aussi. Leur héritage est le cœur de cet article.

Le style des schémas Elektor s'est imposé de façon naturelle, mais a plus ou moins été formalisé vers 1970. C'est en effet à cette époque que nous avons créé nos désormais célèbres symboles de circuit, dont les formes des pistes et des pastilles. Ces motifs graphiques étaient reportés sur des planches de transfert, et ils sont aujourd'hui aussi motifs de jalousie chez bon nombre de nos concurrents. Chaque dessinateur utilisait ces planches pour transformer un schéma griffonné en un schéma estampillé Elektor prêt à l'impression. Le dessinateur opérait par

transferts de symboles Letraset représentant composants et lignes de connexion. Les nouveaux composants étaient dessinés à la volée et collés manuellement sur la planche. Tout ce travail se faisait habituellement à une échelle de 200 %. Lorsqu'il était temps pour lui de s'atteler à l'écriture d'un article, le rédacteur allait récupérer le gribouillis de schéma laissé par un membre du labo près de son fer à souder, puis le comparait avec « l'œuvre d'art pédagogique », le schéma que par bonté le Service Dessin lui avait livré à l'échelle 100 %. Membres du labo et de la rédaction débusquaient ensuite (ou non) les erreurs et les corrigeaient. Le processus de mise en page reposait sur un système maison de reproduction par lithographie : les bandelettes de texte et les illustrations encollées avec de la cire d'abeille chaude étaient montées sur les différentes pages. La colle qui refroidissait instantanément restait à la fois collante et amovible pour les corrections (des Post-its avant l'heure). Enfin le tout était immobilisé avec un rouleau compresseur.

En 1985, après une réorganisation de la maison, la plupart des employés chargés de la mise en page furent mutés chez *Grafisch Bureau Sinke* (GBS), une société située à deux pas de là et dont Elektor était le client principal. Seuls restèrent

EST^D 2004www.elektor.tv

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

les dessinateurs et l'artiste chargé de la couverture.

Desseins humoristiques

Les trois classeurs découverts par Patrick contenaient des centaines de dessins originaux à l'échelle 200 % qui, à mes yeux, représentent le meilleur de ce qui a été produit entre 1980 et 1990. Leur but était d'enrichir le magazine, et aussi d'apporter une touche légère à un contenu autrement fastidieux et difficile à comprendre pour un débutant. Les symboles de circuit et quelques « vues 3D » des composants et des montages étaient destinés à toutes les publications multilingues d'Elektor, c.-à-d. aux livres et

aux magazines. Leur utilisation était toutefois laissée à la discrétion du rédacteur en chef concerné. Le gros des dessins au crayon représentant des « vues d'artiste » allait à Elex, un magazine publié entre 1980 et 1992, et dont le public visé était les jeunes Hollandais et, plus tard, les jeunes Français. Pour satisfaire ce lectorat à la cervelle folâtre, la rédaction d'Elex avait besoin d'une approche plus légère que celle utilisée pour Elektor, magazine dont le lectorat était supposé plus « cérébral » (selon les pays). De tous les dessins non sérieux, et sur une période couvrant une vingtaine d'années, les dessins humoristiques auront été ceux ayant eu le plus d'impact.

À l'exception de Laurent Martin pour les vues en 3D des composants et des méthodes d'assemblage, et de Jaap Kuiper pour les dessins humoristiques, je ne saurais identifier avec certitude les auteurs de ces merveilleuses illustrations. Ce que je sais, c'est que toutes ont été réalisées à la main et en noir et blanc avec des stylos Rotring. J'entends d'ailleurs encore le cliquetis caractéristique de ces stylos secoués par l'artiste pour en liquéfier l'encre. Sur certains dessins, vous pourrez entrevoir les signatures LM et JAAP.

Avec nos compliments, messieurs ! 

(180574-B-04 -

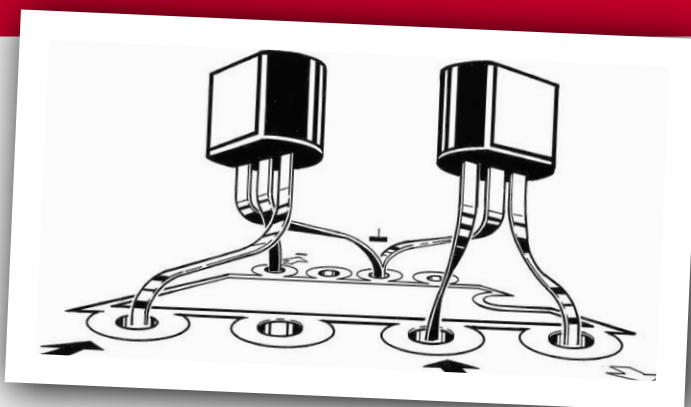
version française : Hervé Moreau)

Les composants

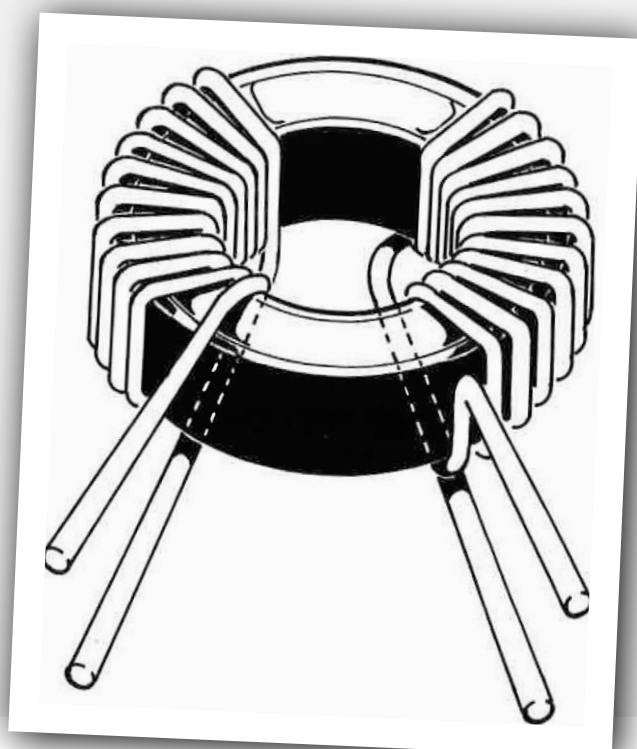


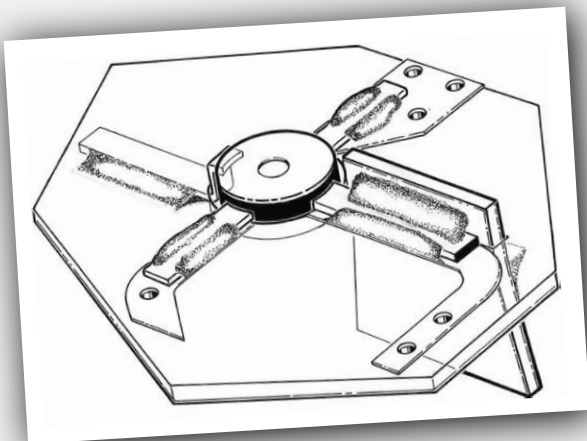
De temps à autre, Elektor se devait d'avertir ses lecteurs des risques de mort et autres désagréments toujours possibles en présence de haute tension.

Le bobinage d'inductances et le sens de leur enroulement relevaient de la magie noire aux yeux des nombreux lecteurs plus à l'aise avec le CC que les radioamateurs qui, eux, aimaient le « L ». L'apparition des tores de ferrite à bobiner soi-même inspira ce joli dessin qui, justement, leur donnait du sens.

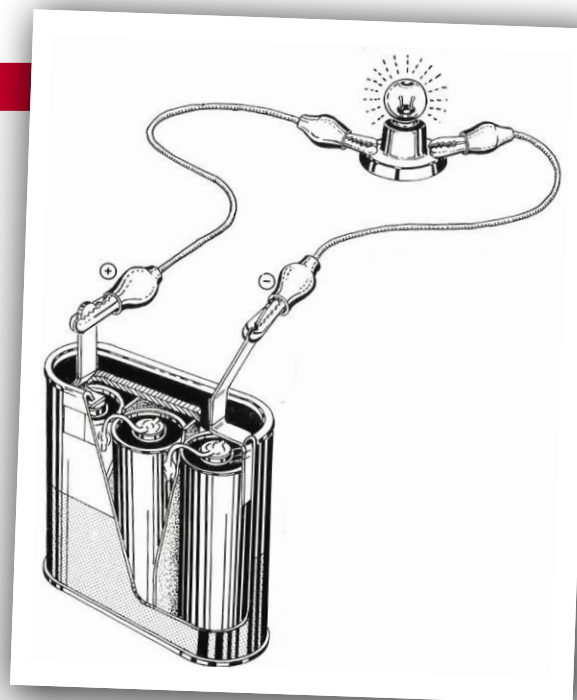


Éléphants comme un couple de danseurs, non ? Comment monter deux TUN (ou TUP ?) sur non pas une piste de danse, mais sur celle d'un circuit imprimé. Est-ce là une source de courant ? Un miroir de courant ? Un Darlington du pauvre ? L'enquête le dira !

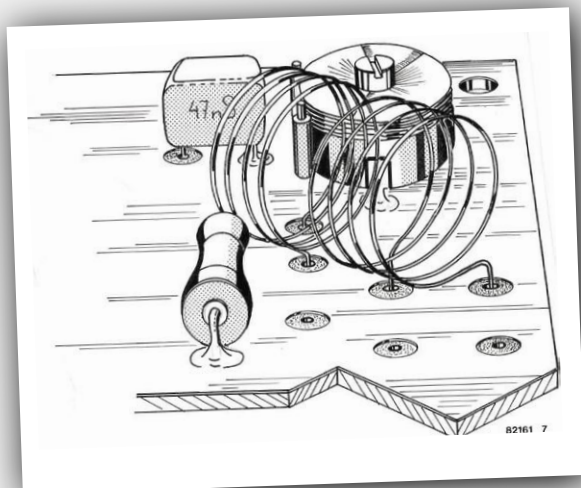




Les MOSFET BF9xx à deux grilles et leur montage « enfoncé dans un trou » avaient de quoi déconcerter. Pour ajouter à la confusion, Elektor prescrivait de monter un condensateur de découplage céramique, sans patte, de type « cercueil », dans une fente de la carte, directement sur la source du FET. Au secours ! Ce dessin aura épargné bien des phrases d'explication au rédacteur, mais quid des lecteurs ?



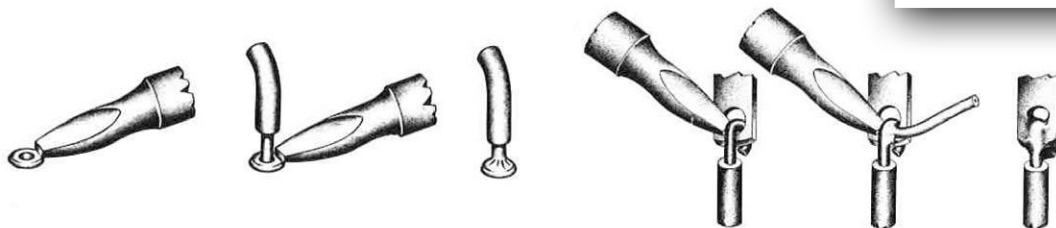
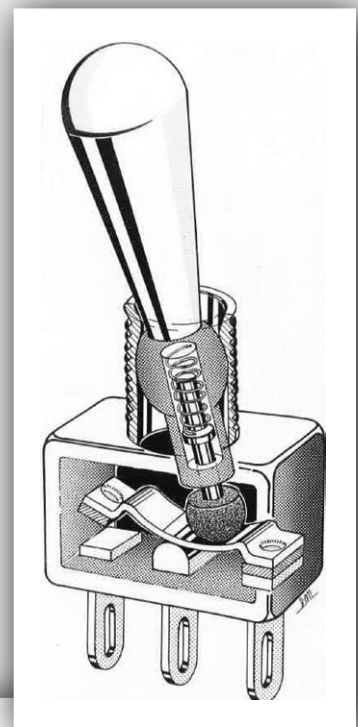
L'icône des débuts de nombreux électroniciens : une ampoule de 6 V alimentée par une pile de 4,5 V. Notez les pinces crocodile et la base de la lampe ! Ce bijou d'illustration est tiré du chapitre 1 de la série à succès « l'électronique ? pas de panique ! » publiée par Elektor dans les années 70 et 80. À mon avis, tous les PDG et directeurs financiers de l'industrie électronique devraient tenter ce montage.



Parlez à un amateur de condensateur variable de 8 pF à PTFE (Téflon), d'inductance de liaison et de condensateur de découplage, vous pouvez être sûr qu'il ou elle préférera un dessin à mille mots.

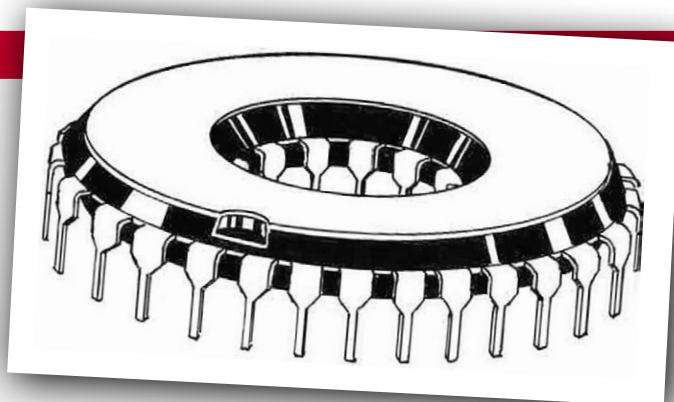
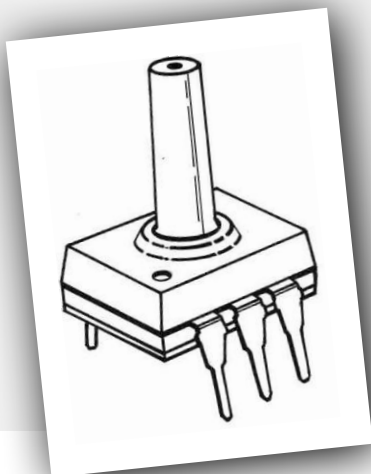
Diverses Bonnes et Mauvaises Pratiques de soudage d'un fil sur une borne (selon Elektor). Pas sûr que l'industrie ait approuvé !

Les mécanismes secrets de l'interrupteur à bascule dévoilés. Comme vous l'ont sans doute appris les joies et les peines du câblage, les contacts fermés ne sont **pas** du côté vers lequel le levier est basculé. Surprenant et contre-intuitif avant, logique maintenant.



Ont-ils vraiment existé ?

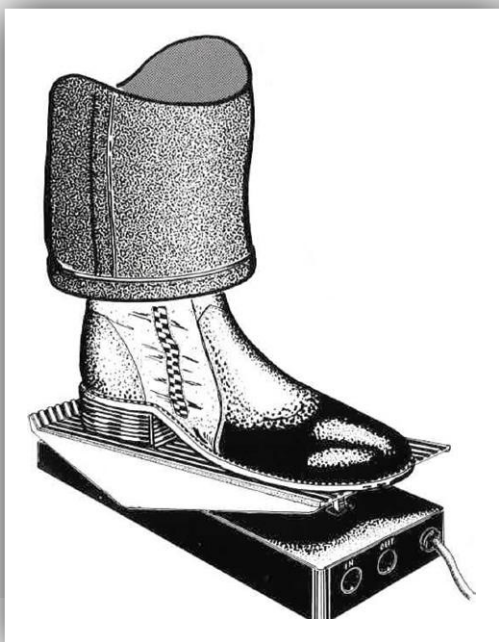
Lorsqu'est arrivé le son *surround*, littéralement « le son qui encercle », beaucoup ont voulu construire leur propre système. Les circuits intégrés prescrits étant souvent introuvables, un dessinateur suggéra celui-ci aux plus... imaginatifs.



Elektor s'intéressa soudain aux capteurs de pression sous forme de circuit intégré, et celui-ci pourrait bien avoir été un des premiers. L'Histoire jugera. En attendant, qui oserait contester la valeur artistique de la canule dressée sur le boîtier ? Ce dessin inspira plus tard un photomontage du *Générateur d'odeurs programmable avec le CD4711*, projet publié dans le numéro d'été de 1986 et un des grands succès des faux-circuits Elektor.

Blanches et noires en noir et blanc

Les circuits et projets liés à la (re)production sonore d'instruments de musique sont depuis des décennies des sujets fer-de-lance d'Elektor. Pour attirer des « étrangers » dans un magazine porté sur l'électronique pure et dure, il faut les « appâter » avec des images bien choisies. Au moins pour Elektor, ce sont les amplificateurs audio, et la Hi-Fi en particulier, qui auront le plus attiré les nouveaux venus. Hélas, le filet remonta aussi pas mal d'idiophiles.



Dans les années 1980, tout le monde voulait construire sa pédale d'effet pour guitare, probablement pour impressionner les filles ou pour masquer une impressionnante inaptitude musicale. Ici un pied sur ce qui me semble être une pédale wah-wah.

Les dessins humoristiques

Je sais, expliquer un dessin humoristique relève du sacrilège, mais comme la plupart de ceux reproduits ci-dessous étaient destinés au lectorat hollandais d'Elex, et que de surcroît il manque le contexte, une légende s'imposait. Avertissement : toute ressemblance entre des personnes représentées dans ces dessins et le personnel d'Elektor ne saurait être que fortuite.

Voyons... comment jouer cette partition sur cet instrument électronique ?

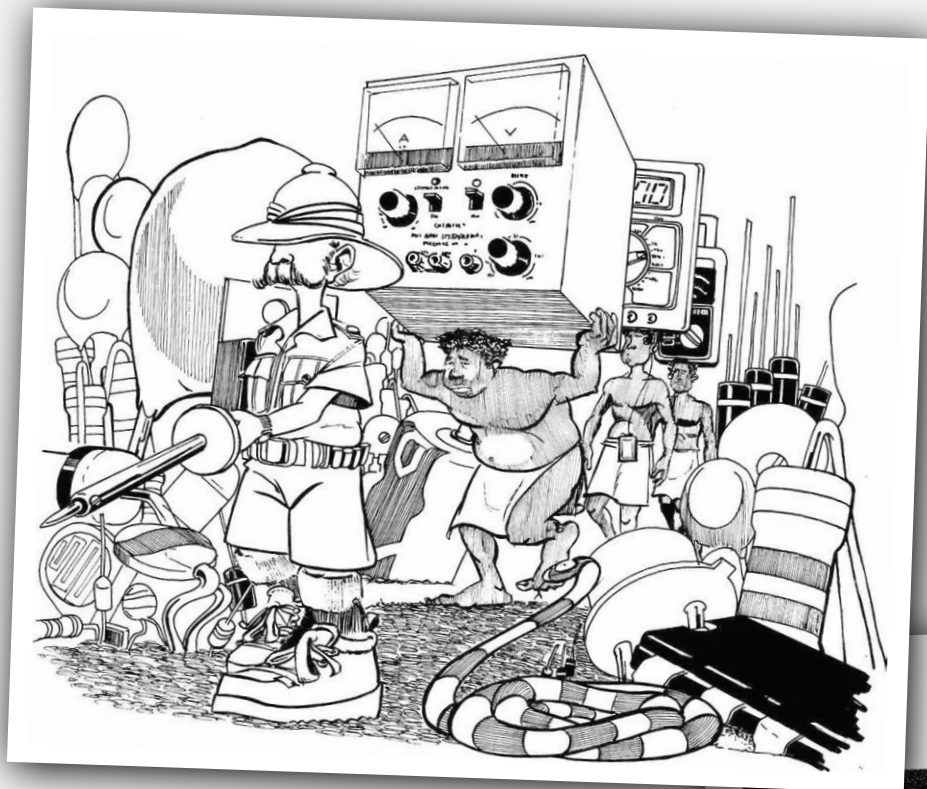


Le séparateur VGA !



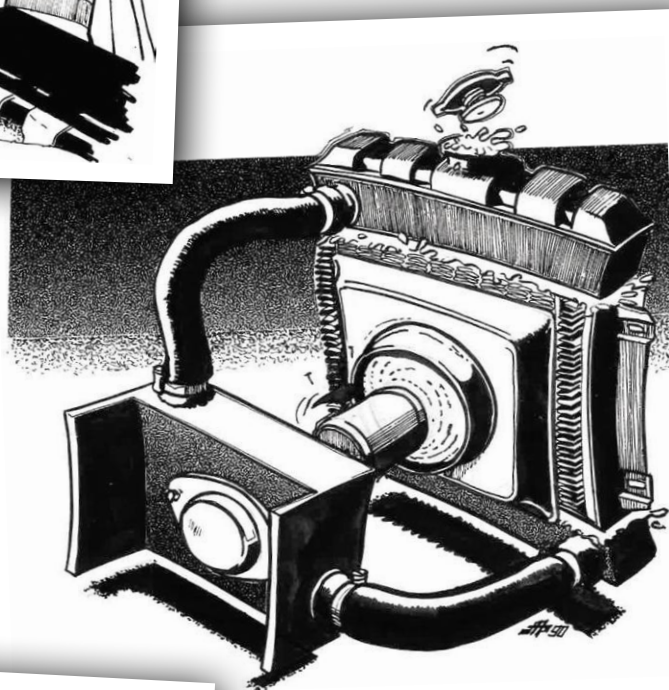
Un chenillard !
(AAN = ON en néerlandais,
UIT = OFF).





Il en faut du matériel et du courage pour explorer et conquérir le monde de l'électronique !

Refroidissement d'un semi-conducteur de puissance par dissipateur thermique !



Frir, griller ou faire sauter des composants et du matériel est... parfois amusant, mais la plupart du temps déconcertant et non souhaitable. Par exemple, ne mettez jamais ce pauvre multimètre sur « ohms » lorsque vous mesurez une source de courant fort !



comment introduire concrètement des valeurs éthiques dans votre projet ?

Tessel Renzenbrink (Pays-Bas)

C'est une très bonne chose que l'éthique soit enfin prise en compte dans la technologie, mais comment introduire des valeurs éthiques dans un projet ? Cette question était au centre des débats lors d'un congrès à Amsterdam (Pays-Bas), où trois orateurs ont partagé leur expérience en la matière. Gijs Boerwinkel, de l'organisation Waag, a montré les possibilités de déterminer des valeurs éthiques pour un groupe suffisamment grand et varié. Douwe Schmidt, administrateur de Tada, a abordé la mise en œuvre des valeurs ; le manifeste de Tada relatif à l'emploi responsable des données a été repris dans l'accord de coalition du conseil municipal d'Amsterdam. Enfin, Mart van Santen, directeur de la technologie chez Greenhost, a rappelé que le fait d'adopter certains principes permettait non seulement d'avoir un œil critique sur sa propre entreprise, mais aussi d'engager le dialogue avec ses partenaires.

Comment déterminer des valeurs éthiques ?

Pour introduire des valeurs éthiques dans un projet, il faut tout d'abord déterminer ces valeurs. Une méthode possible est d'écouter les gens, et autant de gens que possible ! C'est ce qu'a fait le Laboratoire d'identité numérique (IDN) de Waag [1], opérationnel depuis quelque temps déjà et qui implique citoyens et experts dans le développement des nouvelles technologies.

Gijs Boerwinkel, animateur de communauté dans l'organisation Waag, nous a entretenus des recherches du labo sur l'IDN, qui joue un rôle de plus en plus important dans notre vie, que ce soit pour envoyer des messages à nos proches ou communiquer avec les autorités (DigiD aux Pays-Bas). Tout cela constitue le contenu de notre IDN, et nous avons actuellement très peu de prise sur ces données, qui sont éparpillées dans les bases de données de services publics ou d'entreprises privées. Nous ne savons par ex. pas avec qui Facebook partage nos données, ni ce que les sociétés de transport public font des données relatives à nos trajets.

Le Ministère de l'Intérieur des Pays-Bas et l'Association des municipalités néerlandaises ont donné pour mission au labo

de rechercher des méthodes et des applications qui nous permettraient de mieux contrôler nos données privées. Pensez par ex. à une application pour prouver son identité en ligne sans avoir besoin de scanner son passeport ou d'introduire sa carte d'identité dans un lecteur de carte. Le labo souhaite savoir si les applis répondent aux besoins et souhaits des utilisateurs, mais il faut bien entendu connaître ceux-ci. Des réunions ont été organisées en divers endroits du pays, avec des experts et des développeurs de logiciel, mais aussi avec de simples citoyens qui devront utiliser les applis ; des passants ont également été interrogés. Ceci permet d'avoir un échantillon représentatif et d'identifier les valeurs importantes pour la population quand il s'agit de l'IDN. Les deux valeurs jugées les plus importantes sont la transparence (contenu de l'IDN) et le contrôle (avoir droit au chapitre) ; ce sont des valeurs évidentes dans ce contexte. Une autre valeur souvent mise en avant est la facilité d'utilisation, un aspect qu'un groupe d'experts en conception d'applis ou de militants du respect de la vie privée auraient peut-être tendance à négliger. Toutes les parties prenantes ont été impliquées dans la formulation des valeurs, afin d'être sûr de ne rien oublier.

Pour en revenir à notre question : comment déterminer des valeurs éthiques pour un projet ? En y réfléchissant, et surtout en écoutant les autres !

Comment mettre en œuvre ces valeurs ?

La mise en œuvre des valeurs que l'on a définies a fait l'objet d'une discussion du projet Tada [2], qui concerne l'utilisation de données diverses dans une ville. Les données prennent une place de plus en plus importante dans le cadre du développement d'une ville intelligente, qui – selon les fondateurs de Tada – présente de nombreux avantages pour les habitants : cadre de vie plus agréable, propreté améliorée, services plus efficaces, ou encore démocratie effective. L'utilisation des données nécessaires à ce développement doit bien entendu être basée sur certaines valeurs éthiques.

Tada est une initiative qui regroupe une cinquantaine d'organisations, dont le conseil économique d'Amsterdam et l'organisation Waag. Douwe Schmidt, un des administrateurs, explique que Tada a aussi organisé des rencontres pour définir des valeurs éthiques. Ceci a permis de publier un manifeste avec six principes auxquels l'utilisation des données dans une ville doit satisfaire. À titre d'exemple,

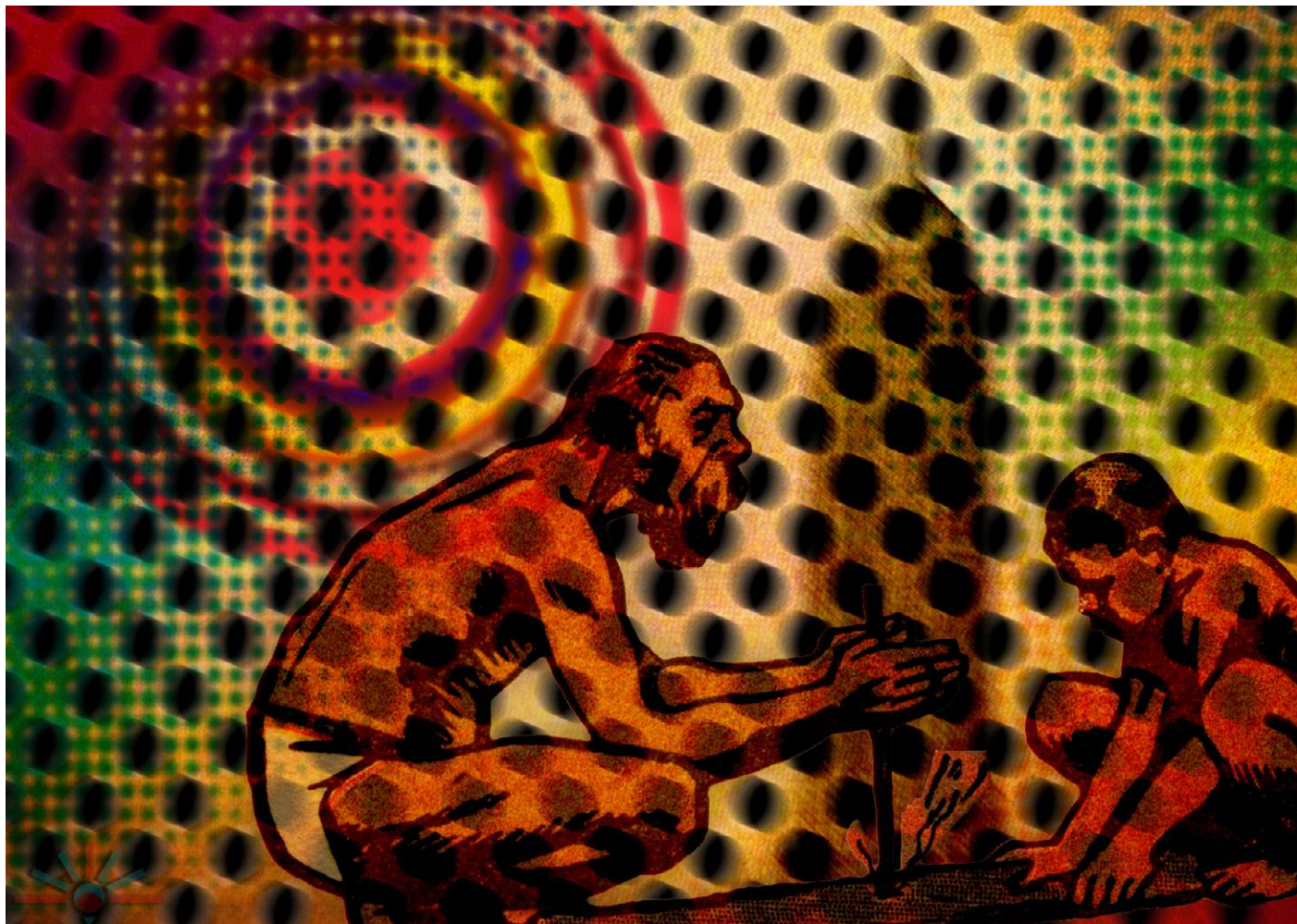


Illustration : Dawn (A.K. Rockefeller). License Creative Commons BY-SA 2.0 / Flickr: <https://is.gd/Gbbqwe>.

citons l'inclusion sociale : l'utilisation des données ne peut pas conduire à l'exclusion de certains groupes d'individus. L'étape suivante est la mise en œuvre des valeurs, et Schimdt tente de nous expliquer l'approche de Tada : « Comment faire en sorte que le manifeste ne passe pas à la trappe ? C'est pourquoi nous avons lancé une campagne pour sa mise en œuvre effective, et nous demandons à tous – entreprises, citoyens, services publics, etc. – de le signer. Les données dans une ville ne concernent en effet pas que l'administration municipale, mais bien tout un chacun, et c'est pourquoi nous organisons aussi des séminaires pour les conseillers municipaux ». À Amsterdam en tout cas, c'est un succès : le conseil municipal qui est entré en fonction l'année passée a adopté le manifeste dans son accord de coalition.

Comment donner de la valeur aux valeurs ?

Mart van Santen, directeur de la technologie chez Greenhost, un hébergeur web [3], a expliqué l'approche de sa

société pour que les valeurs éthiques fassent partie intégrante d'un projet. La durabilité est un principe essentiel chez Greenhost : les technologies de l'information et de la communication consomment beaucoup d'énergie et il faut limiter l'impact sur l'environnement, donc choisir une électricité verte et des appareils peu gourmands. Ce n'est pas toujours simple : un serveur doit par ex. aussi répondre à certaines exigences techniques. C'est donc une affaire de compromis, renchérit Mart.

La gestion responsable et éthique d'une entreprise implique de ne pas seulement se pencher sur ses processus propres, mais aussi de surveiller toute la chaîne en amont. Greenhost fait appel à un centre de données qui affirme n'utiliser que de l'électricité verte, mais on a constaté

qu'une partie était ce qu'on appelle aux Pays-Bas de « l'électricité de la triche », une électricité qui peut légalement se qualifier de verte, mais qui en fait ne contribue pas à l'augmentation de la production d'énergie renouvelable. Greenhost a interpellé la direction du centre de données, qui était semble-t-il déjà en train de rectifier le tir. Ils utilisent désormais de l'énergie éolienne produite en Europe, aux Pays-Bas pour la quote-part de Greenhost.

Ce congrès sur « les principes en technologie » s'est tenu à Amsterdam en décembre dernier. Les organisateurs en étaient le Laboratoire d'IDN, Tada et Greenhost. ◀

(180572-B-04 –

version française : Jean-Louis Mehren)

Liens

- [1] Digital Identity Lab : <https://waag.org/en/project/digital-identity-lab>
- [2] Tada : <https://tada.city/en/>
- [3] Greenhost : <https://greenhost.nl/>



bienvenue dans votre e-choppe

la rédaction recommande



MonoDAQ-U-X – acquisition de données de qualité à petit prix

Avec son port USB-C à isolation galvanique et son boîtier en aluminium, le MonoDAQ-U-X est un « dispositif multifonction et polyvalent d'acquisition de données ». Un même ordinateur peut se connecter à plusieurs MonoDAQ. Le MonoDAQ-U-X dispose de huit ports configurables en E/S analogique ou numérique. Le connecteur à 14 broches donne également accès à un port numérique bifilaire (compatible avec le protocole 1-Wire), une entrée de courant et une sortie de tension. Cette dernière délivre une tension d'excitation à faible bruit (2 à 24 V) ou une tension d'alimentation réglable (1,2 à 24 V) avec un courant de sortie jusqu'à 200 mA (mais pas plus de 1 W). Le MonoDAQ-U-X contient également un capteur d'humidité et de température (à activer dans le paramétrage des dispositifs du logiciel DEWESoft) et une centrale inertielle à 6 axes.

Clemens Valens (labo d'Elektor)



www.elektor.fr/monodaq

Vos favoris :

1. Fer à souder Miniware TS100
www.elektor.fr/miniware-ts100



2. Android App Development for Electronics Designers
(livre en anglais)
www.elektor.fr/android-app-development

3. Testeur intelligent de CMS
Mastech MS8911
www.elektor.fr/mastech-ms8911

4. Raspberry Pi 3A+
www.elektor.fr/rpi-3a-plus

5. HAT PoE pour Raspberry Pi 3B+
www.elektor.fr/poe-hat-rpi-3-plus

6. Testeur USB Bakeey UM25C
www.elektor.fr/bakeey-um25c

7. Raspberry Pi 3B+
www.elektor.fr/rpi-3-plus

Station de soudage à affichage numérique Weller WT 1014 (90 W)



La station de soudage Weller WT 1014 comprend l'unité d'alimentation WT 1, le fer à souder robuste WSP 80 et le support de sécurité WSR 201. L'ensemble est empilable, ce qui libère de la place sur la paillasse. Grâce au capteur intégré, le fer s'arrête automatiquement.

Prix (membres) : 350,10 €

www.elektor.fr/weller-wt-1014

Trousse d'accessoires de mesure PeakTech 8200



Avec cette trousse d'accessoires de mesure pour l'électronique, vous aurez toujours l'adaptateur le mieux adapté à votre multimètre numérique. Les cordons (fiche banane, pointe de touche), les griffe-fils et autres pinces sont fabriqués selon les normes de sécurité les plus récentes. Le matériau utilisé est un plastique souple et antidérapant qui peut supporter les conditions d'utilisation les plus difficiles.

Prix (membres) : 40,46 €

www.elektor.fr/peaktech-8200

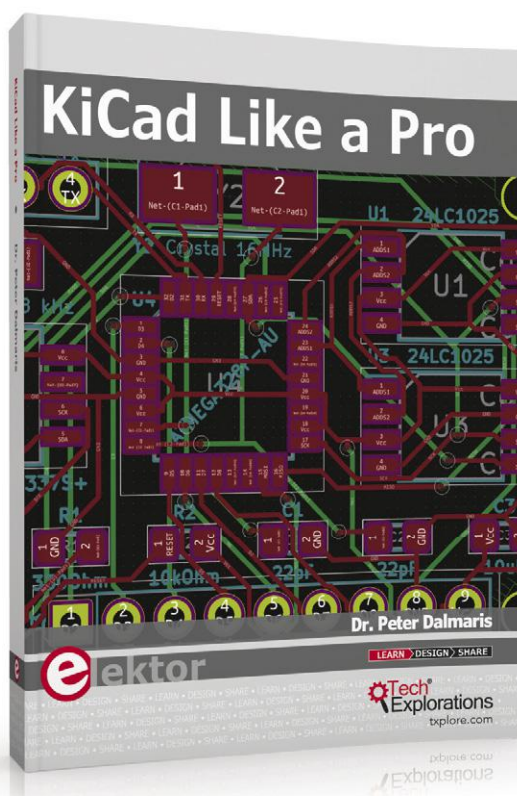
JoyPi – kit d'expérimentation pour Raspberry Pi



Le JoyPi est un boîtier d'expérimentation animé par un Raspberry Pi. Il est idéal pour s'initier à l'électronique et la programmation. Le système de boîtier offre un environnement tout-en-un parfait, qui permet d'éviter la pagaille des composants et des câbles éparpillés sur la paillasse. Le JoyPi dispose d'un grand nombre de capteurs et de modules accessibles aux débutants.

Prix (membres) : 224,10 €

www.elektor.fr/joypi

**nouveau**

KiCad Like a Pro

(livre en anglais)

Ce livre (en anglais) vous apprendra à utiliser KiCad. Que vous soyez amateur ou ingénieur en électronique, grâce à ce livre, vous serez rapidement en mesure de dessiner vos schémas électroniques et de concevoir vos circuits imprimés. L'approche pratique de cet apprentissage repose sur quatre projets de difficulté progressive.

Ces projets permettent de découvrir les fonctions élémentaires et avancées de KiCad. Si vous n'avez aucune expérience de la conception de circuits imprimés, le projet d'introduction vous enseignera les bases. Ensuite vous pourrez continuer avec le reste des projets, à savoir une carte d'alimentation pour une plaque d'expérimentation, un minuscule Raspberry Pi HAT, et un clone Arduino avec mémoire et horloge.

**Membre : 35,96 €****www.elektor.fr/kicad-like-a-pro**

Alimentation de laboratoire programmable JOY-iT DPS5005



Ce bloc d'alimentation réunit dans un même boîtier technologie analogique et fonctions de commande numériques. Il offre une précision exceptionnelle. La sortie réglable peut atteindre 50 V ou 5 A et peut être configurée avec précision par pas de 10 mV ou 1 mA. La manipulation de l'appareil a été optimisée ce qui rend son utilisation particulièrement simple. L'écran couleur offre une présentation très détaillée et complète des informations.

**Prix (membres) : 38,66 €****www.elektor.fr/dps5005**

Electronics for Space *(livre en anglais)*



L'industrie spatiale et le marché des petits satellites en particulier sont en pleine expansion. Bien entendu l'innovation est au cœur de toute cette industrie. Les bureaux d'études ont du mal à trouver des professionnels avec des connaissances en conception électronique spatiale. Le but de ce livre (en anglais) est d'encourager les ingénieurs en électronique à entrer dans le domaine fascinant de l'électronique spatiale.

**Prix (membres) : 22,46 €****www.elektor.fr/electronics-for-space**

Horloge Nixie à 4 chiffres NixieCron-M4



L'horloge NixieCron M4 comprend une puce de montre de haute précision, compensée en température avec une dérive de seulement quelques secondes par mois. Elle est livrée avec un module de son, qui sonne les heures et annonce l'heure actuelle sur simple pression d'un bouton. Il est également possible de connecter un capteur de température DS18B20 et de le configurer à l'aide d'un bouton rotatif.

**Prix (membres) : 134,10 €****www.elektor.fr/nixiecron-m4**

Hexadoku casse-tête pour elektorniciens

Votre magazine se termine toujours et encore par une grille de sudoku particulière. Éteignez le fer à souder, oubliez les écrans de PC et d'oscilloscope, rangez les pinces et les fils. Faites travailler votre matière grise d'une autre façon : attrapez un crayon pour remplir la grille d'hexadoku.

N'oubliez pas d'envoyer votre réponse pour tenter de gagner un bon d'achat.

Une grille hexadoku est composée de chiffres du système hexadécimal, de 0 à F. 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.

Pour participer, inutile de nous envoyer toute la grille, il suffit de nous donner **la série de chiffres** sur fond grisé.



Participez et gagnez !

Nous tirons au sort **cinq** des réponses internationales correctes reçues dans les délais ; leurs auteurs recevront chacun un bon d'achat Elektor d'une valeur de **50 €**. À vos crayons !

Où envoyer votre réponse ?

Envoyez votre réponse (les chiffres sur fond grisé) avec vos coordonnées par courriel, avant le **18 mars 2019** à l'adresse **hexadoku@elektor.fr**

Les gagnants

La solution de la grille du numéro de janvier/février 2019 est **6A258**.

Les cinq bons Elektor d'une valeur de **50 €** vont à :

Mehmet **Can Cakar** (Turquie), Siegfried **Keep** (Allemagne), Vladimir **Koci** (République tchèque), Per **Troelsen** (Danemark), Lucien **Zadikian** (France)

Bravo à tous les participants et félicitations aux gagnants !

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

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

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



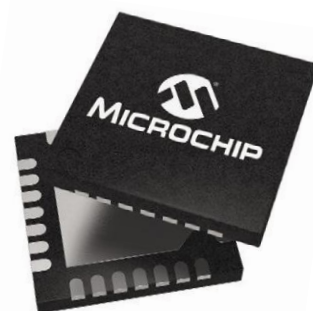
Découvrez votre puissance

La flexibilité de choisir la solution d'alimentation désirée

En tant que fournisseur leader doté d'un portefeuille complet de solutions de gestion et de surveillance de l'alimentation, Microchip vous offre puissance et flexibilité, ainsi que l'assurance de choisir la bonne solution pour votre système.

La gestion de l'utilisation de l'énergie de votre système est capitale pour atteindre les performances exigées par ce dernier. Notre portefeuille de composants de surveillance de l'alimentation vous permet de mesurer avec précision la puissance active, réactive et apparente, le courant et la tension efficaces, la fréquence de ligne et le facteur de puissance. Notre large sélection de composants de gestion de l'alimentation, incluant des contrôleurs et régulateurs DC-DC, des transistors MOSFET et pilotes de MOSFET, des superviseurs et références de tension ainsi que des modules d'alimentation, vous permet de concevoir des solutions efficaces pour gérer les besoins en alimentation de votre système.

Depuis les systèmes de référence jusqu'aux cartes d'évaluation en passant par les outils de simulation, vous réduirez le temps passé à la conception et minimiserez les risques grâce au support technique complet offert par Microchip.



Découvrez votre puissance sur
www.microchip.com/PowerSolutions





productronica
fast forward
the start-up platform
powered by Elektor

PARTICIPEZ AU CONCOURS POUR
LANCER VOTRE STARTUP

SUR LE SALON
PRODUCTRONICA 2019

● Participez à l'édition 2019!
12 au 15 novembre 2019 à Munich

informations détaillées :
www.elektormagazine.fr/p-ffwd

Productronica Fast Forward is brought to you by



productronica

