F : 15,50 €

elektor 487
créer > partager > vendre

JANVIER & FÉVRIER 2021
ELEKTORMAGAZINE.FR

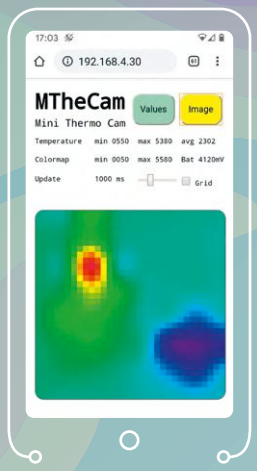
Elektor Helps
L'électronique aide quand ça va mal

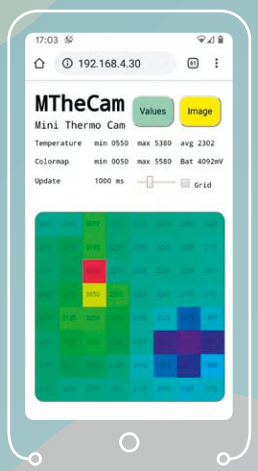
MTheCam

Minicaméra thermique simple pour localisation de points chauds et froids



p. 8







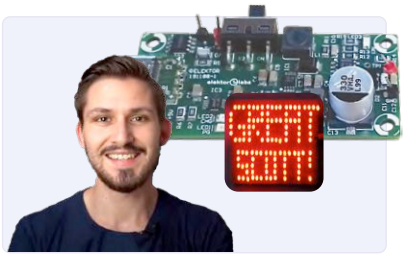
dans ce numéro :

- géné audionumérique I²S de sinus numérique 1 kHz, à 32 bits
- nouveau LCR-mètre 50 Hz - 2 MHz (2)
- connectez votre thermostat avec ESPHome
- analyseur de puissance multicanal
- banc d'essai : station Weller WE 1010
- module de mesure sans fil JOY-iT VAX-1030
- Raspberry Pi aux commandes de la maison
- Raspberry Pi Full Stack RPi et RF24 au cœur d'un réseau de capteurs
- multitâche en pratique avec l'ESP32 (6) Groupes d'événements
- conception de filtres analogiques (3)

et beaucoup d'autres !



analyseur de puissance multicanal
3 canaux avec affichage graphique et alphanumérique de la puissance
p. 66



chargeur & booster de LiPo en kit
Chargeur, booster et protecteur de LiPo proposé par GreatScott & Elektor
p. 6

ISSN 0181-7450

(B) 16,50 € • (CH) 29,00 FS • (CAN) 20.99 $ca • (And) 15,50 €
DOM surface 16,50 € • DOM avion 16,90 € • N Cal/S 2000 cfp

L 19624 - 487 - F: 15,50 € - RD

NOTRE GAMME
PAR DES TECHNICIENS
POUR DES TECHNICIENS

The best part of your project:
www.reichelt.com

Uniquement le meilleur pour vous - provenant de plus de 900 marques.

Nos responsables produits sont employés par Reichelt depuis de nombreuses années et connaissent les exigences de nos clients. Ils rassemblent une large gamme de produits de qualité, à la fois parfaits pour les besoins dans les domaines de la recherche et du développement, la maintenance, l´infrastructure informatique et la production en petites séries et adaptés pour les fabricants.

go-eCharger, boîtier de recharge pour voiture

Installation fixe, 11 kW, Type 2

En plus de fonctions de supplémentaires sécurité telles que la détection de phase, la mesure de mise à la terre et la détection d'adaptateur, le boîtier est équipé de la RFID pour le contrôle d'accès et du WiFi avec une fonction hotspot et peut être personnalisé à l'aide de son application.

- Boîte murale pour installation permanente
- Conception : 3 phases (16 A)
- Modèle : 11 kW
- WiFi, RFID
- Conception : Type 2
- Application contrôlable







PRODUIT RECOMMANDÉ

N° de commande :
GO-E CH-02-18-1

672,50 (560,42)

Vous pouvez trouver de nombreux autres produits pour l'électromobilité en ligne!

Découvrez maintenant ▶ www.reichelt.com/électromobilité









Types de paiement : Sofort. VISA MasterCard Maestro PayPal amazon pay

PRIX DU JOUR! Prix à la date du: 12. 11. 2020

- Excellent rapport qualité prix
- Plus de 110 000 produits sélectionnés
- Livraison fiable - depuis l'Allemagne dans le monde entier

www.reichelt.com

Assistance téléphonique: +33 97 518 03 04

reichelt elektronik – Tirer le meilleur parti de votre projet

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://rch.lt./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

44ème année
n° 487 – janvier/février 2021

ISSN 0181-7450
Dépôt légal : janvier 2021
CPPAP 1120 T 83713
Directeur de la publication : Donatus Akkermans

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

Pour toutes vos questions :
service@elektor.fr

www.elektor.fr | www.elektormagazine.fr

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

Publicité :
Margriet Debeij
Tél. : +31 6 380 780 29
margriet.debeij@elektor.com

DROITS D'AUTEUR :
© 2021 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'oeuvre 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 éditrice 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 éditrice. La Société éditrice 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 éditrice 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 éditrice 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
Pijper Media – Groningen
Distribué en France par M.L.P.
et en Belgique par A.M.P.

Jens Nickel

rédacteur en chef d'Elektor Magazine



concevoir ensemble, plus que jamais !

Les temps sont difficiles pour nous tous, mais chez Elektor mes collègues et moi avons au moins la chance de pouvoir bien travailler à domicile. La durée des trajets boulot-dodo gagnés et la décontraction vestimentaire permettent de consacrer davantage de temps à nos passe-temps même en semaine. Je compte ainsi jeter (enfin !) un coup d'œil sur le nouveau RPi. L'apprentissage machine m'intéresse aussi, j'ai même sur le feu quelques idées pour l'utiliser en musique et en vidéo. J'espère que vous aussi avez maintenant un peu plus de temps pour réactiver des projets endormis. Peut-être avec l'une de ces cartes de prototypage récoltées dans l'un des derniers (vrais) salons ? Ou avec l'un des modules proposés dans notre boutique en ligne ? N'hésitez pas à partager votre expérience avec nous et tous nos lecteurs ! Sur notre plateforme www.elektormagazine.fr/labs, vous pouvez créer encore aujourd'hui un projet puis décrire en quelques lignes l'objectif visé et les résultats déjà obtenus. Ne vous souciez pas trop d'y parler un bon anglais ni de fournir d'emblée une documentation complète, ça s'arrangera plus tard. D'autres elektoriens partageant vos idées vous poseront rapidement leurs questions. Quelqu'un proposera peut-être un projet similaire qu'il a déjà commencé de son côté. Il vous signalera éventuellement des embûches à éviter.

Ensemble, on est plus forts, et par la magie des réseaux cela fonctionne en dépit du confinement ! Nous, rédacteurs, examinons régulièrement ce qui se passe sur la plateforme Elektor Labs, sans négliger aucun des projets. Un de nos objectifs pour 2021 est justement de raccourcir le délai de publication de vos propositions dans notre magazine. Cette réduction de notre *temps de propagation* profitera aussi à vos propositions d'articles envoyées par courriel ! Désormais, un rédacteur de l'équipe vous répondra sans tarder et supervisera la progression de votre proposition.

Continuons de concevoir ensemble et restez en bonne santé !

Bonne lecture.

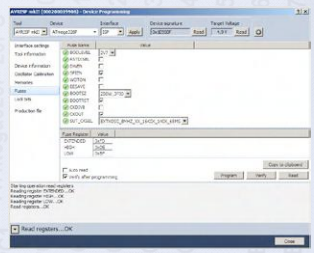
notre équipe

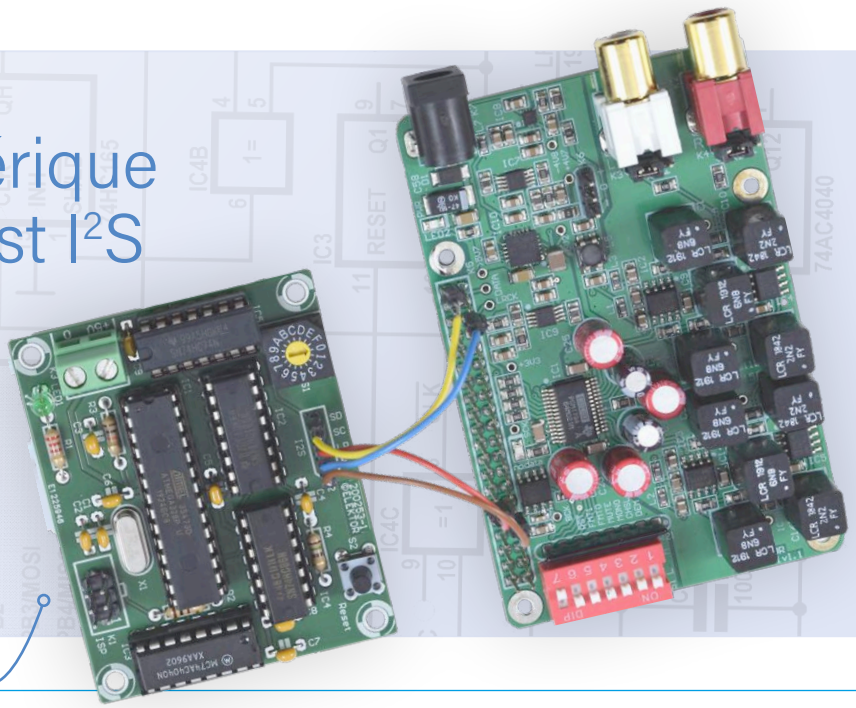
Rédacteur en chef : Jens Nickel
Rédaction : Eric Bogers, Jan Buiting, Rolf Gerstendorf, Denis Meyer (traduction), Thomas Scherer, Clemens Valens
Service aux lecteurs : Ralf Schmiedel
Correcteur technique : Malte Fischer
Laboratoire : Mathias Claussen, Ton Giesberts, Luc Lemmens, Clemens Valens (responsable), Jan Visser
Maquette : Giel Dols, Harmen Heida

DANS CE NUMÉRO

projet de labo :

géné audionumérique de signaux de test I²S





22

rubriques

3 **Colophon**

20 **gagnants du concours electronica Fast Forward 2020**

36 **sur le vif**
Entre ordre et chaos

38 **débuter en électronique… (5)**
…est moins difficile qu'on ne l'imagine !

50 **bureau d'études – Zone D**
D comme développement, comme débrouillardise et dur-à-cuire

53 **elles sont petites mais font de belles choses**
Les pépites d'Elektor

94 **corrections, mises à jour et courrier des lecteurs**

108 **Propeller 2 de Parallax (1)**
Une découverte

114 **Hexadoku**
casse-tête pour elektorniciens

contexte

17 **banc d'essai : Weller WE 1010**
Station de soudage

32 **simule tes circuits en ligne**

62 **multitâche en pratique avec l'ESP32 (6)**
Groupes d'événements

74 **conception de filtres analogiques (3)**
Filtres passifs

83 **Banc d'essai : module de mesure sans fil JOY-iT VAX-1030**

92 **erreurs fécondes**
Conseils sur les régulateurs de tension et sur la conception des circuits imprimés, etc.

96 **Java sur Raspberry Pi Entretien avec Frank Delporte**

100 **analyse de données et intelligence artificielle en Python**
Interpréter les données réelles avec NumPy, pandas et le scikit-learn

110 **observatoire du matériel libre**
Évaluation communautaire du matériel à source ouverte

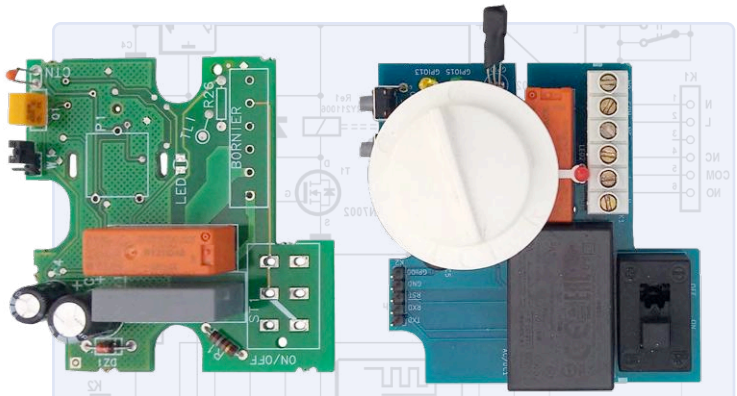
projets

6 **superchargeur & booster de LiPo en kit**
Chargeur, booster et protecteur LiPo proposé par GreatScott & Elektor

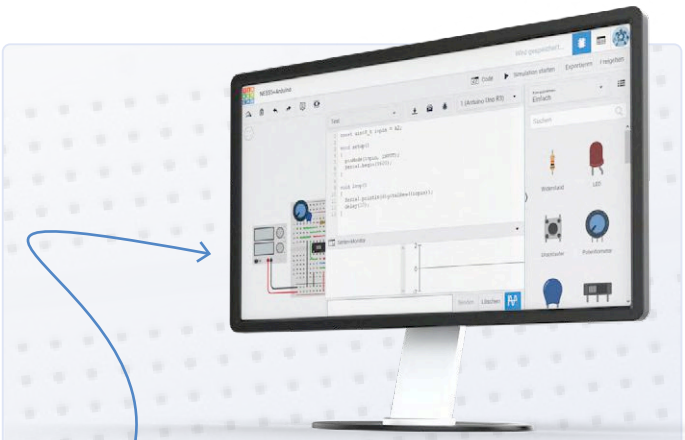
8 **mini caméra thermique MTheCam**
Minicaméra thermique simple pour localisation de points chauds et froids

22 **géné audionumérique de signaux de test I²S**
Sinus numérique 1 kHz, à 32 bits, échantillonné à 192 kHz, réglable de 0 à –110 dB

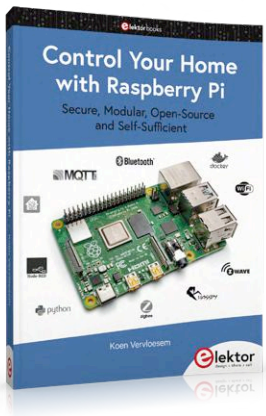
27 **Raspberry Pi aux commandes de la maison**
RPi furète sur 433,92 MHz



connectez votre thermostat avec ESPHome 44



simule tes circuits en ligne 32





RPi furète sur 433,92 MHz 27

41 **ohm suite ohm**
Ni CPU ni ALU,
juste un opérateur logique NOR à deux transistors

44 **connectez votre thermostat avec ESPHome**
Une tentative pour faire de la domotique comme il faut

56 **Raspberry Pi Full Stack**
RPi et RF24 au cœur d'un réseau de capteurs

66 **analyseur de puissance multicanal**
Avec affichage graphique et alphanumérique
de la puissance sur 3 canaux

86 **nouveau LCR-mètre 50 Hz - 2 MHz (2)**
Fonctionnement, étalonnage et microprogramme

bientôt dans ces pages

Le numéro de mars-avril 2021 d'Elektor
Elektor a invité ses amis de SparkFun Electronics à participer à la prochaine édition d'Elektor ! Nous sommes donc heureux d'annoncer ici que les rédacteurs et les ingénieurs de nos deux maisons sont engagés dans une étroite coopération.
Établi au Colorado, États-Unis, SparkFun Electronics est un créateur innovateur de kits et de modules électroniques amusants et intéressants.
Voici quelques-uns des articles et des sujets sur lesquels nous travaillons ensemble pour la prochaine édition d'Elektor (mars/avril) :

- Se lancer avec MicroMod
- Conseils pour la création de produits électroniques personnalisés
- Comment construire vous-même une station de référence GNSS
- Programmation d'un FPGA
- Configuration des circuits imprimés pour le bricolage
- Les indispensables pour votre atelier d'électronique
- ClockClock : Un projet de démonstration de FPGA
- Mise en œuvre de FreeRTOS sur RED-V

Et bien davantage !

Ne manquez pas la prochaine édition spéciale d'Elektor créée par des ingénieurs et des électroniciens d'Europe et d'Amérique ! L'édition mars-avril d'Elektor paraîtra le 11 mars 2021. Nos membres Elektor Gold recevront ce numéro dans leur boîte à lettres. Les titres et le contenu des articles sont susceptibles de changer.

superchargeur LiPo en kit

Chargeur, booster et protecteur de LiPo proposé par GreatScott & Elektor

Mathias Claußen (Elektor)

Avez-vous déjà pensé aux avantages d'une alimentation par batterie LiPo rechargeable, avec sorties de 5 et 12 V ? Avez-vous déjà pensé à vous initier au soudage de CMS ? C'est ce que vous proposent Elektor et le youtubeur GreatScott! Dans cet article, nous présentons à la fois cette alimentation portable pratique et quelques obstacles que nous avons rencontrés au cours de l'étude et de la mise au point du circuit.



Figure 1. Un bel emballage pour un beau kit.

Tout électronicien a au moins une fidèle alimentation stabilisée ; c'est LA source d'énergie de la paillasse de son labo, pour lui le moyen le plus simple, mais pas toujours le plus commode pour alimenter ses protos : les cordons sont parfois gênants surtout si l'appareil en cours d'étude est destiné à un usage portable ou nomade comme on dit maintenant. On voit de plus en plus souvent des grappes de piles scotchées ou fixées avec de la colle thermofusible, associées à un convertisseur DC/DC bon marché pour en faire une sorte de batterie portable.

Pour un proto, c'est bien, mais pas très pratique, surtout si vous avez besoin de 5 ou 12 V pour alimenter votre appareil. « Peut mieux faire ! » ont déclaré d'une seule voix Elektor et le populaire GreatScott après leur rencontre sur le salon *productronica 2019* à Munich ! Ainsi naquit le kit présenté ici, fignolé pour vous.

Pour ceux qui ne connaissent pas GreatScott!, c'est le nom d'une chaîne YouTube [1] qui depuis 2013 présente des projets et des connaissances électroniques à plus d'un million d'abonnés. Certaines vidéos invitent les spectateurs à reproduire les projets variés proposés. D'autres sont conçues comme sources d'inspiration et distillent des solutions à des problèmes de conception. Après quelques vidéos consacrées à des produits d'Elektor, l'idée est venue de présenter un kit réalisable et utilisable par toute personne intéressée, aussi pour élargir ses connaissances et ses compétences.

GreatScott! [2] a esquissé le schéma d'une alimentation LiPo rechargeable, à fabriquer par l'utilisateur avec des composants CMS. À première vue, pas de composants ni d'autres ingrédients trop compliqués : circuit intégré de charge pour les LiPo, convertisseur DC/DC pour 5 V et 12 V, circuit intégré de protection des batteries. Tous les composants sont de la taille 1206, afin que même des débutants puissent implanter ces CMS sur le circuit imprimé. La vocation de ce kit est donc aussi de montrer que le soudage des CMS ne relève pas de la magie. Quelques composants délicats sont déjà implantés sur le circuit imprimé fourni avec le kit. Vous n'aurez donc aucune manipulation délicate.

Le câble de charge sous 5 V pourra être soudé directement, mais pour fournir au chargeur l'énergie nécessaire à la recharge de la batterie,

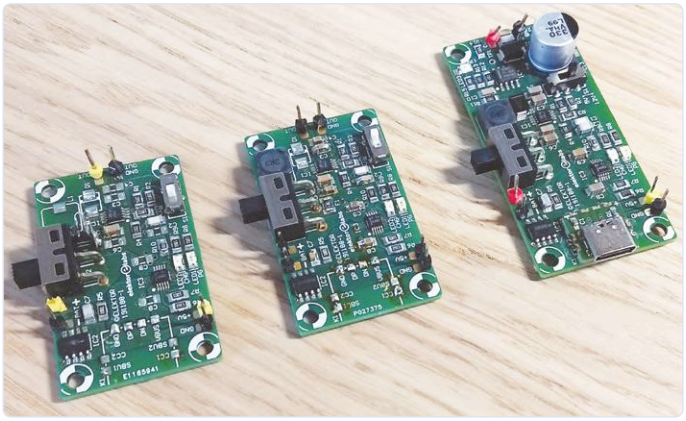


Figure 2. Trois itérations de gauche à droite.

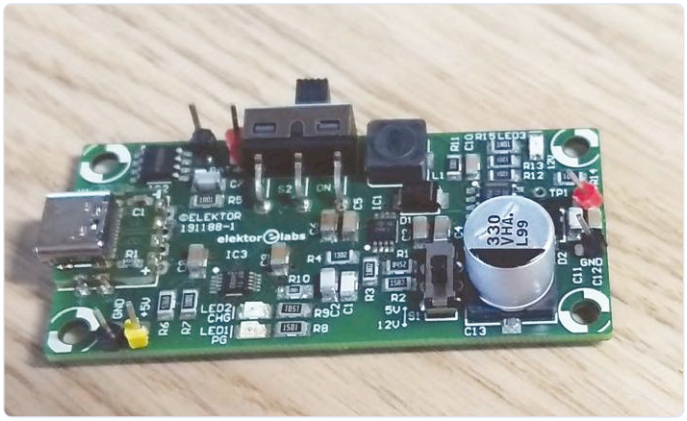


Figure 3. PCB assemblé.

SPÉCIFICATIONS

- Entrée : 5 V +/–10 %
- Sortie : 5 V / 1,5 A ou 12 V /0,75 A
- Pile au lithium à cellule unique

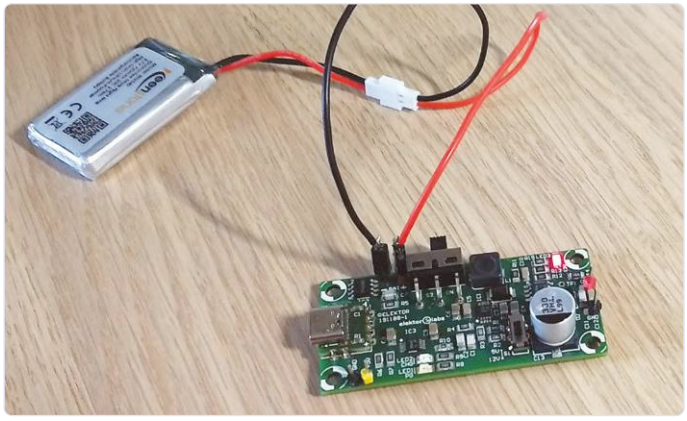


Figure 4. PCB en action.

vous pouvez aussi utiliser le connecteur USB-C sur le circuit auxiliaire. Avec l'USB-C, vous ne risquez pas d'inversion de polarité et, pour une meilleure stabilité, le connecteur est maintenu solidement par quatre trous. Le plaisir et la satisfaction d'utiliser cette alimentation commencent par le fait de la construire soi-même. Un guide d'assemblage pas-à-pas est inclus (**fig. 1**).

Xysemi, fournisseur des trois circuits intégrés de la carte, est bien connu, mais le XB8089D, circuit de protection de la batterie, n'est trouvable que chez les distributeurs spécialisés dans le silicium chinois. Il protège contre les excès de surcharge, de décharge et de courant et contre l'inversion de polarité. Le prix de ce petit boîtier SOP8 sans broches (*exposed pad*) est raisonnable. Même si beaucoup de batteries, comme celles des drones, sont elles-mêmes munies d'une protection, il est préférable de laisser ce circuit en place pour le cas où vous utiliseriez un jour ce chargeur pour une batterie au lithium *sans* protection.

À partir de l'ébauche de GreatScott, Elektor a parfait le schéma et son circuit imprimé. Ce devait être la fin de l'histoire (« on a le PCB, et tout va bien… »), mais ce serait sans compter sur tout ce qui peut se passer au fil des trois itérations habituelles (**fig. 2**). Le proto1 a fonctionné *presque* comme prévu, à part des problèmes « mineurs » : *primo* pas de charge de la batterie, *secundo* désintégration du convertisseur DC/DC en présence d'une charge (trop) forte. Pour ce qui est de *secundo*, le circuit intégré protège (pour ainsi dire) la batterie au lithium, mais si le seuil de courant est de 10 A (soit 37 W), le convertisseur CC/CC se désintègre lui à environ 15 W. Quant au problème de charge (*primo*), il a été vite résolu… une résistance de 10 k était devenue une 100 k. Banale faute de frappe ou mauvaise lecture de la fiche technique ? Le BQ24092 appartient à la ligne BQ2409**x** ; selon ce dernier chiffre de la référence de la puce (x), la résistance est soit de 10 k soit de 100 k ! Il a heureusement été plus facile de changer la résistance que le circuit intégré complet !

La révision finale du circuit imprimé ressemble à ce que vous voyez sur la **figure 3** et la **figure 4** le montre en action. Nous avons d'ailleurs tiré d'autres enseignements de ces trois itérations, des choses à faire ou à ne pas faire. Ce qui nous permet de proposer des kits toujours mieux préparés. Je vous invite à consulter la chaîne YouTube de GreatScott ! Il montre l'assemblage du circuit et son utilisation.

191188-02

LIENS

[1] **Canal YouTube de GreatScott!**: www.youtube.com/c/greatscottlab/

[2] **Entretien avec GreatScott!**: www.elektormagazine.com/news/elektor-magazine-2020-summer-edition

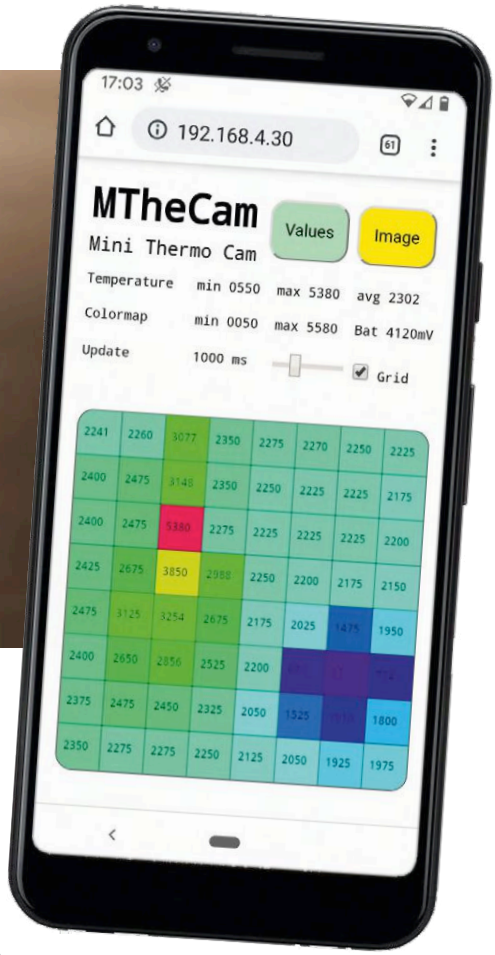
mini caméra thermique MTheCam

Minicaméra thermique simple pour localisation de points chauds et froids

Olaf Mertens

Ai-je soufflé la bougie en partant ? Et la plaque chauffante, est-elle éteinte ? Qui n'a pas déjà quitté son logis avec la crainte d'avoir oublié un appareil susceptible de provoquer de gros dégâts hors surveillance ? Il est désormais possible d'interroger MTheCam par ordiphone. Bien entendu, le projet présenté ici, avec son capteur thermique à 8x8 pixels de *Panasonic*, se prête à bien d'autres applications.





Le mot *hot-spot* (= point chaud) nous fait plutôt penser aux points d'accès publics à l'internet, mais ici ce terme désigne des points dont la température dépasse sensiblement celle de leur environnement (et inversement pour *cold-spots*). Il peut s'agir de flammes, de points de surchauffe, de courts-circuits, ou inversement, de défauts d'isolation thermique. Ces points chauds, s'ils ne se manifestent pas par des flammes ou des braises incandescentes, restent invisibles à l'œil humain. Pour les détecter, il faut un capteur, comme MTheCam. La localisation de points chauds ne se limite certes pas à des choses comme une plaque chauffante oubliée. Dans les montages électroniques, un échauffement anormal de composants surchargés peut annoncer très tôt leur décès prochain. Sur les machines, les frottements de paliers usés ou de surfaces mal lubrifiées se traduise par un échauffement excessif ; une maintenance au bon moment prolongera leur durée de vie. Même les humains peuvent être reconnus comme points chauds, on

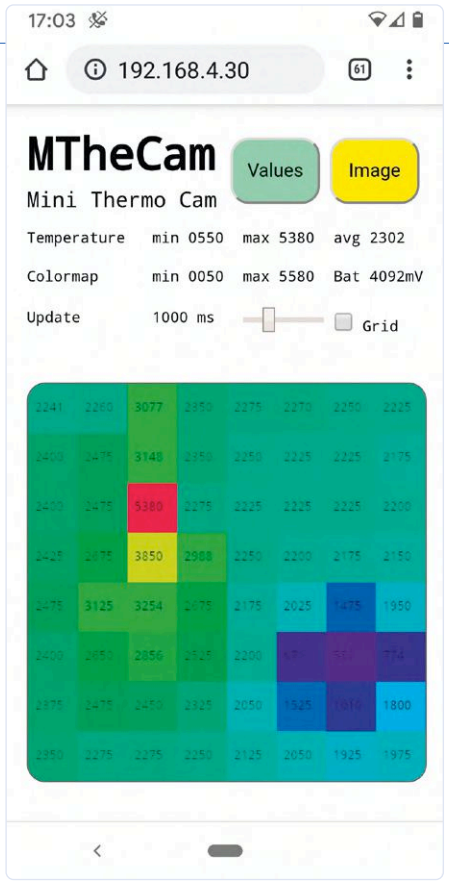


*Figure 1. Saisie d'écran des valeurs de température sur 8*8 pixels.*

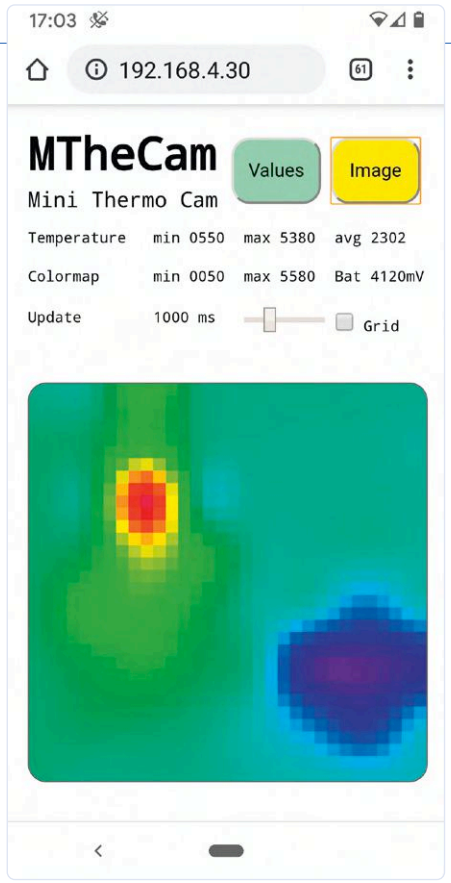


*Figure 2. Saisie d'écran des valeurs interpolées sur 32*32 pixels.*

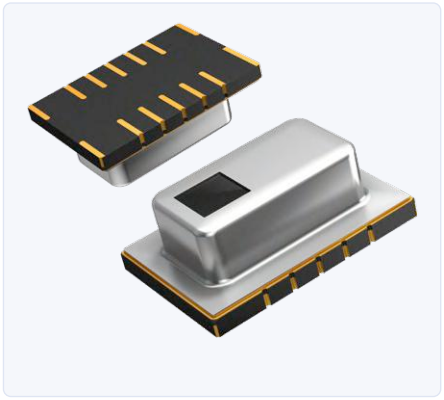


Figure 3. Le capteur AMG88xx de Panasonic.

peut ainsi les compter ou suivre leurs mouvements. Outre les appareils de mesure de température par contact direct, il existe des capteurs sans contact, sensibles au rayonnement infrarouge des objets et en déduisent la température moyenne du champ de mesure. Ils sont basés sur l'effet pyroélectrique qui modifie le potentiel électrique d'un cristal polarisé pris entre deux électrodes quand il est exposé à un rayonnement thermique. Cet effet est évidemment utilisable en électronique [1].

Projet avec MTheCam

MTheCam mesure simultanément 64 points jusqu'à cinq fois par seconde sur huit lignes et colonnes, disposées comme dans une caméra optique avec un angle de vue de 60 °. Chaque point indique la température de 0 à 80 °C (ou de –20 à 100 °C). Aux différentes valeurs de mesure sont attribuées des couleurs d'une palette, de sorte qu'on obtient une image à assez basse résolution, affichable par Wi-Fi sur un ordiphone. Cet amas de pixels colorés n'impressionnera sans doute pas les habitués de la TVHD, mais il affiche les points chauds ou froids au milieu d'un environnement de couleurs nettement différentes. Outre cette image en fausses couleurs, la valeur de mesure de chaque pixel est affichée en °C, ce qui permet une analyse plus précise (**fig. 1**). Les 64 valeurs de mesure sont lisibles sous forme d'un enregistrement de données JSON, utilisable comme source de données pour d'autres applications.

La matrice de 8x8 points est transformée en «fausse» image de 32x32 points de couleur par interpolation bicubique [2], plus agréable à regarder et où les objets sont plus faciles à identifier. Le spectre de l'échelle de couleurs est adapté au domaine des températures mesurées, du minimum au maximum avec un peu de jeu ajouté, ce qui produit un effet de loupe sur les valeurs de température (**fig. 2**).

Capteur de température AMG88xx

Pour la prise d'une image thermique, *Panasonic* commercialise un capteur performant (**fig. 3**) en technologie MEMS (MicroSystème ElectroMécanique) en deux variantes (deux domaines de mesure), qui réunit l'optique, le traducteur thermoélectrique, la mesure analogique et d'autres fonctions de préparation des données dans un boîtier relativement petit [3]. Les valeurs de mesure dans le domaine 0 à 80 °C (AMG8853) ou –20 à 100 °C (AMG8854), qui présentent une erreur maximum de +/–2,5 K (+/–3,0), sont lues via une interface I²C. Dans l'absolu, la précision n'est pas impressionnante, mais suffisante pour l'évaluation qualitative de valeurs relatives. Avec quelques lignes de code, on peut y ajouter un étalonnage qui augmente la précision et diminue le bruit de fond. Le boîtier CMS peu encombrant du capteur *Panasonic* supporte le soudage par refusion. Pour notre application, il trouve place sur une petite carte d'essai maison qui comprend quelques autres composants passifs (au format 0805 soudable à la main) nécessaires au découplage de la tension d'alimentation et au rappel des lignes du bus I²C. Outre la tension d'alimentation (+5 V) et la masse, trois lignes importantes (INT et le bus I²C) sont amenées sur des pastilles au bord de la carte pour une barrette à broches droites ou coudées. La **fig. 4** montre le schéma de cette carte et révèle quelques détails du circuit interne du capteur. Sur la page du projet [9] on peut télécharger gratuitement le tracé des deux faces de la carte et le plan d'implantation des composants. Pour plus de confort, on peut se procurer le module tout équipé chez l'auteur.

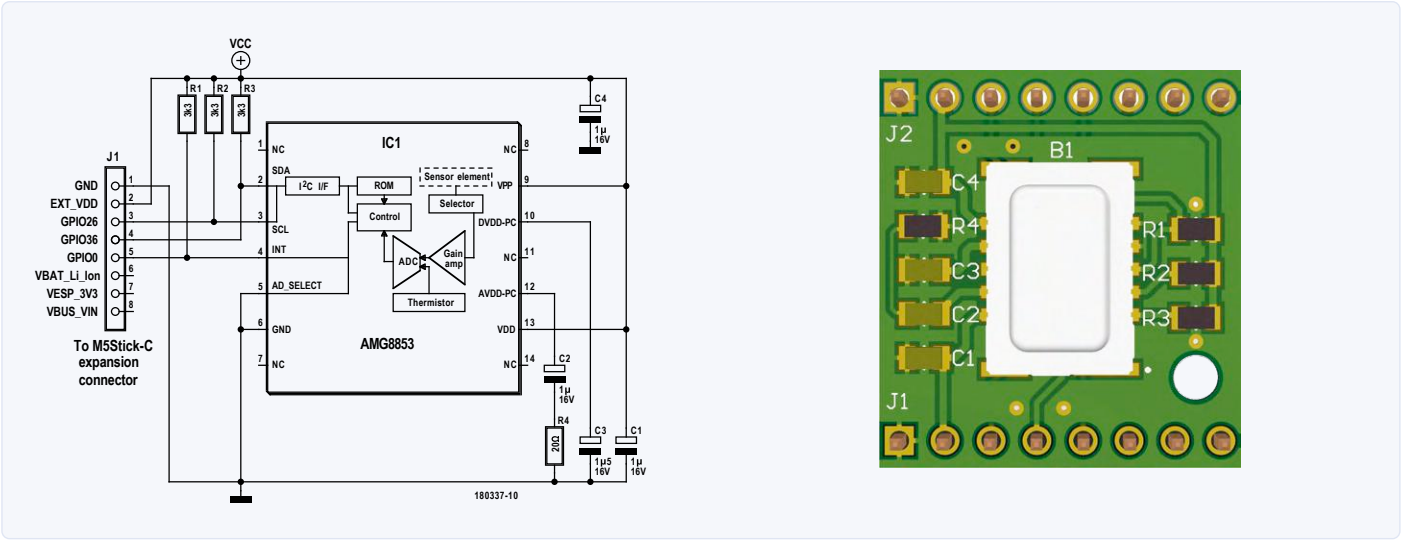


Figure 4. Schéma interne et périphérique du capteur AMG88xx sur la carte d'essai.

M5StickC, circuit à tout faire

Chaque ordiphone est compatible Wi-Fi et possède un écran tactile pour la lecture et les saisies ; c'est l'interface idéale pour notre MTheCam avec laquelle il peut établir une communication sans fil. Pour cela, le capteur doit être équipé d'un transmetteur de données intelligent : un système sur une puce (SoC) avec contrôleur et module Wi-Fi. Notre choix s'est porté sur l'ESP32 d'Espressif, doté de tout ce qui peut séduire un développeur, bon marché, peu gourmand et facile à programmer avec l'EDI Arduino. Outre le capteur et le module ESP, il ne manque que l'alimentation sous 5 V que peut fournir un port USB ou une batterie en opération autonome. Dans la boutique en ligne d'Elektor, mon attention a été attirée par un produit paré d'orange vif, le M5StickC qui offre en sus de l'ESP32 quelques fonctions intéressantes. Ce petit appareil qui ne mesure que 50x26x14 mm en a dans le coffre ! Outre l'ESP-32Pico avec Flash de 4 Mo et SRAM de 520 Ko, il possède un écran de 2,5 cm (0,96") et 80x160 pixels, un accéléromètre à six axes, une horloge en temps réel, une unité de gestion de l'alimentation, une LED rouge, une LED infrarouge pour la télécommande, un microphone MEMS, une batterie de 80 mAh, l'USB-C, un connecteur Grove (alimentation, masse et deux ports I²C) et une barrette à 8 broches femelles avec trois ports et l'alimentation. Voilà de quoi émoustiller l'électronicien le plus blasé !

Avec sa petite taille, son afficheur et son alimentation autonome, le M5StickC peut se faire passer pour une montre chic. Avec son support et son bracelet orange vif, on est certain de se faire remarquer (**fig. 5) !** Après ça, qui voudrait encore mettre à son poignet un quelconque bijou d'un producteur de pommes californien ? Sur lequel on ne peut même pas brancher quoi que ce soit ! Dans ce projet nous n'utiliserons que l'ESP32, l'écran et la batterie, mais les autres fonctions seront certainement les bienvenues dans de futures applications.

En dépit de sa petite taille, l'image sur l'écran du M5StickC n'en est pas moins nette et en couleurs. Pour une image thermique, c'est bien ; pour quelques lignes de valeurs de mesure, aussi. Le défi pour le concepteur sera de faire tenir davantage d'informations sur cette surface minuscule. La bibliothèque graphique fait une grande place aux jeux et aux animations colorées. Elle donne aux inventeurs de nombreuses possibilités d'exercer leur créativité.

Avec ses 80 mAh, la batterie Li-Ion peut alimenter le M5StickC à pleine charge pendant près d'une heure. Elle se recharge via une prise USB-C de 5 V / 500 mA, qui reçoit le câble fourni, à brancher d'autre part sur une banale fiche USB-A. Une pression sur un petit bouton latéral allume le M5StickC, une pression pendant six secondes l'éteint.

Coopération

Selon que la barrette utilisée aura les broches droites ou coudées, la position de la carte du capteur sera parallèle ou perpendiculaire (**fig. 6**) au plan du M5StickC. Avec une imprimante 3D, il est facile de créer de jolis boîtiers pour les deux variantes en fonction de l'axe de visée que vous préférez.

Le connecteur d'extension du M5StickC possède huit broches affectées à la sortie 5 V, à la masse, au signal d'interruption INT (GPIO36) et aux deux lignes I²C de signaux de données et d'horloge (SDA sur GPIO26 et SCL sur GPIO0). Les trois autres broches (BAT, 3V3 et l'entrée 5 V) restent libres. Remarquons que GPIO36 ne peut être utilisée qu'en entrée, ce qui évite bien du débogage.

Pour l'instant, la possibilité d'interruption n'est pas exploitée par le logiciel. Elle permettrait, par ex., de programmer la puce du capteur pour émettre un signal INT lorsqu'un seuil est franchi dans un sens ou dans l'autre, et ce pour chaque pixel individuel ! La puce pourrait alors surveiller toute seule une certaine partie de son champ de vision et ne réveiller l'ESP32 que lorsqu'un seuil est franchi, ce qui économiserait de l'énergie. Le processus de surveillance pourrait être géré ensuite par l'application sur l'ordiphone exploitant l'image entière.

Question de logiciel

Pour le développement du micrologiciel, au lieu de l'environnement propriétaire d'Espressif, on peut utiliser l'écosystème Arduino, très intéressant pour son support très efficace par une communauté mondiale. L'EDI Arduino est excellent pour démarrer mais rapidement insuffisant pour des applications exigeantes. C'est pourquoi l'auteur recommande l'usage de l'éditeur de code libre du Visual Studio de Microsoft, qui possède une extension supportant l'Arduino [6]. *M5Stick-C (esp32)* est ajouté par le gestionnaire des cartes.

L'auteur a utilisé ces outils pour écrire le micrologiciel en C++. Au

fichier source *MTheCam_LT.ino,* s'ajoutent quelques fichiers .h et .cpp (*Mxxx.cpp/h*), ainsi que la bibliothèque *ArduinoJson* hautement recommandée (au moins la V6, la V5 ne fonctionne pas ici !) pour le traitement JSON [7]. La bibliothèque *M5StickC* [8] gère les particularités matérielles sans problème. Elles doivent être ajoutées via le gestionnaire de bibliothèques (*Outils > Gérer les bibliothèques*).
Contrairement à la superbe documentation de l'*ArduinoJson-Lib,* la description de la bibliothèque *M5StickC* laisse beaucoup à désirer. Afin de l'utiliser correctement, une laborieuse lecture du code source s'impose. Cet épineux sujet mériterait un article séparé !
Voyons quelques fragments de code pour différentes fonctions dans l'ordre logique.

Lecture des capteurs
Les valeurs mesurées actuelles sont rafraîchies par le capteur dix fois par seconde dans des registres d'un octet, que le logiciel lit en permanence via l'interface I²C *Wire*. En raison de la résolution de 12 bits, il faut lire deux octets par pixel. C'est ce que fait *Wire* dans `MTC_readReg():`

```
#define BUF_LENRX 128
int reg = 0x80;
byte _rxBuf[BUF_LENRX];
. . .
Wire.beginTransmission(devAddr); // Chip-Address @
   datasheet
Wire.write(reg); // 0x80 -> read 128bytes of 64
   pixels @12bit
Wire.endTransmission();
if (Wire.readTransmission(devAddr, rxBuf, BUF_
   LENRX) == I2C_ERROR_OK) { success = true; }

Wire.endTransmission();
```

En cas d'échec de la lecture, `Wire.readTransmission()`peut éventuellement retourner des codes d'erreur utiles pour le débogage.
Pour la lecture du capteur et le calcul des valeurs de température, on se reportera au **listage 1**.
À partir de deux entrées du tableau `rxBuf[]`, et, moyennant quelques décalages de bits, on arrive à calculer la valeur de la température – ici sous forme d'un nombre entier en centièmes de degré Celsius : pour 21,35 °C on a donc la valeur 2135. Le capteur est initialisé de telle manière qu'il forme en interne la moyenne mobile de deux images et réduit ainsi un bruit un peu fort. Tous les registres sont lus en une seule fois, de sorte que toutes les valeurs sont garanties appartenir à une même fenêtre de mesure. Les autres calculs de la température des pixels et de la puce (non utilisée ici) diffèrent par un facteur (voir la fiche technique [3]). Pour plus de précision de calcul, on utilise la multiplication par 10000, divisée par 100 pour obtenir le 100ᵉ de degré.



Figure 5. Le M5StickC comme montre-bracelet orange vif (source: m5stack.com)



Figure 6. Droit ou coudé : deux manières de connecter le capteur au M5StickC.

Calcul des valeurs de couleur
Pour la représentation des couleurs, on utilise le diagramme HSL [10] (**fig. 7**). Pour cela, la collection M5Stick-C contient une bibliothèque TFT correspondante. Des paramètres teinte (Hue, H), saturation (Saturation, S) et luminosité (Lightness, L), seul le paramètre H est utilisé, S et L sont ignorés (constants). Les valeurs HSL sont converties en valeurs RGB pour un affichage ultérieur.
Comme nous résolvons la plage de 0 à 80 °C en 8000 valeurs mesurées (100ᵉ de Kelvin) pour l'affichage numérique, celles-ci doivent d'abord être normalisées au domaine des valeurs de teinte possibles entre 0 et 359. Comme nous avons l'habitude d'associer le bleu au froid et le rouge au chaud, le grand arc de cercle entre 255 degrés (t_{Min} = bleu-vert) et 315 degrés (t_{Max} = rouge vif) semble être optimal pour l'affichage. Comme nous n'avons pas l'habitude d'associer le violet à une sensation de température, les valeurs de violet sont ignorées. Ainsi, des 360 valeurs de couleur possibles, 300 sont utilisées et 60 ne le sont pas.
La fonction `map()` (**listage 2**) applique les valeurs de température dans l'angle de 300 degrés compris entre 315 et 15 degrés de l'espace des teintes dans le sens antihoraire. Puis elle tourne le tout de 60 ° vers la gauche (angle de –60 °) pour finalement atteindre les bornes 255 et

Listage 1. Lecture du capteur et calcul de la température.

```
. . .
 // read one frame with 64 temp-values à 12bit -> join 2 bytes to 1 int
 // result is int[] with calculated values -> _frame[]
 int  _frame[FRAMESIZE];          // filled by MTC_getFrame()
. . .
 int* MTC_getFrame()
  {
    byte _rxBuf[BUF_LENRX];
    byte dL[FRAME_SIZE];  // low byte
    byte dH[FRAME_SIZE];  // high byte
    int reg = 0x80;       // start with this reg for next 128 bytes to read
    . . .
    // sensor-read 128 bytes - low + high
    boolean ok = MTC_readReg(MTCADDRESS, reg, _rxBuf, 128);  // readings now in _rxBuf
    if (ok)
    {
        int k = 0;
        for (int i = 0; i < FRAME_SIZE; i++)
        {
            int index   = i;
            int d1 = _rxBuf[k++];                // low byte here
            int d2 = _rxBuf[k++];                // high byte next: includes sign!
            // sensor on PCB is upside down, start from the end
            index  = (FRAME_SIZE - i - 1);
            _frame[index] = MTC_calcTemp(d1, d2); // 2 bytes calc'd to one int
        }
     }
    return _frame;
 }
. . .
 // make one int from low/hight bytes d1/d2
 int MTC_calcTemp(uint8_t d1, uint8_t d2)        // return temp in deg Celsius * 100
  {
    int factor = (int) (0.25 * 10000);           // pixel-temp, for device-temp use 0.0625
    if (d2 & 0b00001000) d2 |= 0b11111000;       // handle sign-bit #4 in high-byte
    int valRaw = (d2 << 8) | d1;                 // merge to one int …
    int valCalc = (int)(valRaw * factor / 100);  // … with this factor
    return valCalc;
 }
. . .
```

Listage 2. Calcul des valeurs de couleur.

```
#define MC_COLORWHEEL_LOW  315     // degrees on colorwheel
#define MC_COLORWHEEL_HIGH  15
#define MC_DIFFCW           60     // diff in degrees high to low
. . .
int MC_getColorHSL(int value, int vmin, int vmax, int vavg)
{
    // do not use 255 - 315 deg = 60 deg -> it's not clearly 'felt' as cold or hot
    // vmin/vmax for minimum spread of spectrum -> less color flicker if no spots seen
    if((vavg - vmin) < MC_TMIN_SPREAD2AVG)  { vmin = vavg-MC_TMIN_SPREAD2AVG; }
    if((vmax - vavg) < MC_TMIN_SPREAD2AVG)  { vmax = vavg+MC_TMIN_SPREAD2AVG; }
    // map range vmin -> 315, vmax -> 15 degree 'left turn' - exclude 315...15 = 60 deg
    int colInd = map(value, vmin, vmax, MC_COLORWHEEL_LOW, MC_COLORWHEEL_HIGH); // 315/15
    colInd = colInd - MC_DIFFCW;   // turn - 60: icecold->DEEPBLUE=255, hot->DARKRED=315
    if (colInd < 0) colInd += 360; // no neg. values!
    return colInd;
}
```

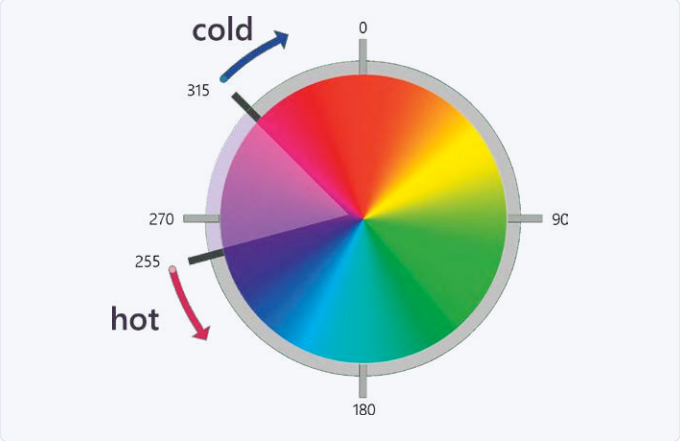


Figure 7. Modèle HSL: disque chromatique et domaine utilisé.

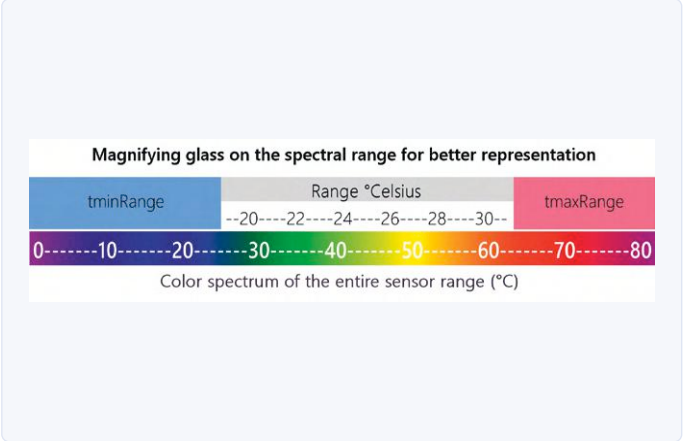


Figure 8. Le domaine de mesure est étendu à la partie souhaitée du spectre.

315. Pour réduire le bruit de fond relativement intense dans les zones de température uniforme sans point chaud ou froid, on introduit un intervalle minimum de 500 unités entre la température moyenne et tMin et TMax, pour que le spectre d'affichage (la «loupe») ait une largeur minimum de 1000 unités, correspondant à 10 °C, afin que chaque pixel bruité ne prenne pas une couleur complètement différente.
La barre de couleurs avec la zone de spectre utilisable est ainsi fixée : la température minimale correspond à 255 degrés du cercle des couleurs, la maximale à 315 degrés, parcourus dans le sens antihoraire.

Représentation sur l'écran

Pour réaliser une sorte de fonction de loupe qui applique le domaine d'affichage effectif dans cette plage spectrale possible, la fonction de détermination des couleurs utilise les valeurs `tminRange` et `tmaxRange` comme paramètres (**fig. 8**). Il s'agit là d'un moyen ultrasimple de créer un affichage couleur attrayant, qu'on pourrait améliorer par l'utilisation d'une palette de couleurs spécialement conçue, un calcul plus astucieux de la loupe de couleur, par ex. en déterminant les valeurs de la gamme de manière adaptative sur plusieurs mesures afin d'éviter les sauts de couleur, ou un traitement séparé de la dynamique du point chaud et de l'arrière-plan afin que ce point apparaisse plus clairement. Les experts en apprentissage machine auront plaisir à utiliser l'IA pour la reconnaissance des motifs, la réduction du bruit et l'analyse des données.
L'écran TFT du M5StickC montre (**fig. 9**) dans sa moitié supérieure l'image thermique originale avec le spectre de couleurs utilisé, et, en dessous, les valeurs mesurées minimum et maximum (tMin et tMax) pour l'ensemble des pixels de l'image, ainsi que la valeur moyenne calculée à partir de celles-ci (tAvg). Ces valeurs sont données en °C * 100, c'est-à-dire sans virgule.

Wi-Fi et serveur web

Jusqu'à présent, tout fonctionne localement sur le M5StickC et sans Wi-Fi. Mais si les valeurs mesurées doivent être affichées ou même traitées sur un appareil mobile distant, le M5StickC doit pouvoir se connecter par Wi-Fi et fonctionner comme serveur web capable de fournir une page web avec les valeurs mesurées.
Dans le logiciel Arduino, les bibliothèques `WiFi.h` et `WiFiClient.h` créent un tel serveur web, qu'il suffit de paramétrer avec les données



Figur 9. Le domaine de température du prototype.

d'accès à votre réseau (`ssid` et `password`). Après une réinitialisation ou une mise sous tension, l'écran affiche la tentative de connexion au réseau. En cas de succès, l'adresse IP reçue est affichée. Sinon : les données d'accès sont-elles correctes ? Un point d'accès est-il suffisamment proche ? D'ailleurs, la portée de ce petit appareil est étonnante, puisqu'elle peut facilement se comparer à celle des ordiphones.
Le serveur web écoute alors les requêtes `/`, `/index.html` et `/frame` sous son adresse IP locale attribuée par le routeur, par ex. `http://192.168.10.1/`.
Une fois la connexion Wi-Fi établie, le serveur web apprend ce qu'il doit faire lorsqu'il reçoit une requête spécifique du navigateur client. La fonction associée est alors appelée - par ex., à `/` (identique à `/index.html`) est associée la fonction `MW_index()`, qui délivre le document au client :

```
webServer.send_P(200, «text/html», _uidoc);
```

Listage 3. Wi-Fi et serveur web.

```
. . .
    const char *ssid     = "YourSSIDHere";      // change…
    const char *password = "YourPasswordHere";  // … to your WLAN
    WebServer webServer(80);                    // listening on port 80 = standard

    WiFi.mode(WIFI_STA);                        // switch to ESP station mode
    WiFi.begin(ssid, password);                 // look for known WLAN & try to connect
    while (WiFi.status() != WL_CONNECTED)       // wait for connection
    {
       delay(500);                              // 500ms delay
       Serial.print(".");                       // still alive
    }
    Serial.println("\n[Setup] WIFI connected!");
    // what to do if browser request is coming in
    webServer.on("/", MW_index);                // call MW_index() for / or /index.html
    webServer.on("/index.html", MW_index);
    webServer.on("/frame", MW_frameJS);         // /frame - output as JSON-Data
    webServer.onNotFound(MW_notFound);
    webServer.begin();                          // webserver starts here
. . .
// Webserver callback
void MW_index()     // send HTML-Doc from literally defined PROGMEM var
 {
    webServer.send_P(200, "text/html", _uidoc); // send_P: use PROGMEM-Var
 }
. . .
// main loop
void loop()
 {
                    // give webserver a chance to handle requests. No delay() in LOOP!
      webServer.handleClient();
     . . .
 }
. . .
```

`Send_P` est utilisé parce que le document HTML `_uidoc` est conservé en mémoire flash en tant que `PROGMEM` pour ne pas encombrer la mémoire RAM. Cela permet à MTheCam d'afficher des images thermiques en couleur sur n'importe quel ordiphone. Le **listage 3** montre et documente les parties de programme du Wi-Fi et du serveur web.

HTML et JSON

Les requêtes `/` et `/index.html` créent une page web avec un tableau de 8x8 cellules colorées avec indication numérique de la température. De plus, sont affichées la température minimale (tmin), maximale (tmax) et moyenne (tavg) d'une mesure. Chaque teinte correspond à une température, le gradient de couleur étant ajusté dynamiquement à la plage des valeurs MIN et MAX (`tminrange / tmaxrange`). Le bouton «*Image*» commute l'animation thermique sur une image «haute résolution» en fausses couleurs de valeurs interpolées, le bouton «*Values*» la fait revenir en arrière. Chaque seconde, une nouvelle mesure est affichée; l'intervalle d'affichage peut être modifié à l'aide d'un curseur. La case à cocher «*Grid*» superpose un quadrillage sur l'image.

La requête `/frame` retourne les valeurs mesurées sous forme d'un enregistrement de données JSON pour un traitement ultérieur par d'autres programmes, avec un délai à prévoir d'au moins 200 ms pour laisser le temps à la puce du capteur d'effectuer une nouvelle mesure, ce qui prend environ 100 ms. Un tel enregistrement pourrait être créé au besoin par manipulation de chaînes de caractères, mais la bibliothèque très pratique `ArduinoJson.h` permet d'alléger ce travail et peut même lire de telles données. Le résultat ressemble par ex. à ceci :

```
{"device": "MTheCam","ver": "1.0","rowsize":
   8,"batvolt": 3.39,
"tavg": 2317,"tmin": 2050,"tmax": 2750,"tminrange":
   1817,"tmaxrange": 2817,
"frame": [2075, 2050, 2100, 2275, 2600, 2475, 2100, 2050,
, ..... ]} // 64 values °C * 100 // + 2 arrays with
'HSL'-colors of 8*8 and 32*32 tiles.
```

Le **listage 4** montre les parties intéressantes du texte HTML. Le document avec les balises CSS et HTML est défini comme une chaîne fixe et envoyé au navigateur client. L'image thermique est construite à partir d'une balise `<canvas>` avec un tableau 8*8 (*values*) ou 32*32 (*image*).

Pilotée par une minuterie, une partie du script récupère alors les données du capteur (*/frame*) qui sont ensuite livrées par paquet JSON. Javascript et JSON s'entendent bien, de sorte que les données utilisateur peuvent être extraites du paquet très facilement : *jsonData.frame* ainsi que *jsonData.framecol* fournissent directement sous forme de tableaux les valeurs et les couleurs à partir desquelles la température originale et les valeurs interpolées sont évaluées, les cellules correspondantes colorées et pourvues de leur valeur numérique de

Listage 4. Créer un document HTML avec des valeurs mesurées.

```
. . .
/* Canvas-Area 400 px with 8*8/32*32 tiles à 50/12 px */
<div class="row">
      <div class="column">
        <canvas id="canvas"></canvas>    /* size see resizeCanvas() */
      </div>
</div>
. . .
/* resize canvas for value or image mode */
function resizeCanvas () {
 . . .
     canvasSize = rectSize * gridSize    /* gridSize: 8 or 32 @rectSize: 12 or 50 */
     canvas.width  = canvasSize
     canvas.height = canvasSize
. . .
/* update image,  rate = 200…3000 ms */
function refresh (rate = 1000) {
 . . .
refresher = setTimeout(function () {
        getData()
        }, rate)
. . .
/* request temperature-frame from MTC via /frame - call  */
function getData () {
     fetch(url)
        .then(response => response.json())
        .then(jsonData => {
          if (mode == 1) {
            // tempvalues, colors (interpolated) in framecolinter array
            paint(jsonData.frame, jsonData.framecolinter);
          } else {
            // tempvalues, colors (original) in framecol array
            paint(jsonData.frame, jsonData.framecol)
          }
          infoText(jsonData)             /* show min/max etc. values */
        }).catch(err => {
          errorText.innerHTML = 'Error: $'
      }).then(function () {
        refresh()                        /* start timer for next ride */
     })
}
. . .
/* paint tiles on canvas */
const ctx  = canvas.getContext("2d")

function paint(values, colors) {
  . . .
   ctx.clearRect(0, 0, canvas.width, canvas.height);
   colors.foreach(color, index) ….  {
     . . .   /* set x/y of next tile: x+=rectSize … */ . . .
     /* paint tile */
     ctx.beginPath()
     ctx.rect(x, y, rectSize, rectSize)           /* paint square */
     ctx.fillStyle = 'hsl($, 100%, 50%)'   /* set color of square */
     if (grid) { ctx.stroke() }                    /* show grid optional */
     ctx.fill()                                   /* do fill */
     ctx.closePath()
     if (mode == 0) {                             /* values as text over */
         // show tempvalues @mode=0 -> lowres 8*8 only
         drawText(values[index], x + 7, y + rectSize / 2)
       }  }
 . . .
```

température. L'intervalle de mise à jour de cet affichage peut être réglé de 200 ms à 3 s.

Bonne base pour de nouveaux projets

Dans une version enrichie du micrologiciel, il faut prévoir un point d'accès Wi-Fi privé, géré par ESP32, établissant son propre réseau local et permettant une saisie aisée des données d'accès à d'autres réseaux. Pour un affichage optimal, la palette et la loupe de couleurs seront adaptables, des algorithmes de filtrage réduiront le bruit et le scintillement. Une application Android peut simultanément gérer, configurer et afficher en direct plusieurs appareils MTheCam. Des états et des messages d'alarme peuvent être définis et déclenchés ; Node-Red via le serveur MQTT donnera plus de confort à l'exploitation des données. Cela vaut donc le coup de consulter de temps en temps la page du projet [9]. Les bases d'applications pratiques intelligentes étant jetées, ne manquent plus que les bonnes idées ! Nous remercions tout particulièrement Daniel Zelosko pour le développement du prototype.

180337-03



PRODUITS

> **M5StickC ESP32-PICO Mini – Carte de développement pour l'IdO**
www.elektor.fr/m5stack-m5stickc-esp32-pico-mini-iot-development-board

Votre avis, s'il vous plaît ?
Votre avis et vos commentaires sont bienvenus par courriel adressé en allemand ou en anglais à l'auteur (mtc@micom.de) ou en français à la rédaction (redaction@elektor.fr).

Ont contribué à cet article
Auteur : **Olaf Mertens** (Micom)
Traduction : **Helmut Müller**
Rédaction : **Rolf Gerstendorf**
Maquette : **Giel Dols**

LIENS

[1] **Pyromètre** : https://fr.wikipedia.org/wiki/Pyrom%C3%A8trerg/wiki/Pyrometer
[2] **Interpolation en photographie** : https://bit.ly/interpolation-photo
[3] **AMG88x** : https://eu.industrial.panasonic.com/products/sensors-optical-devices/sensors-automotive-and-industrial-applications/infrared-array
[4] **Documentation ESP** : www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf
[5] **M5StickC** : www.elektor.fr/m5stack-m5stickc-esp32-pico-mini-iot-development-board
[6] **VisualStudio** : https://code.visualstudio.com/
[7] **Arduino-JSON** : https://arduinojson.org/
[8] **Bibliothèque M5StickC** : https://github.com/m5stack/M5StickC
[9] **Page de projet de l'auteur** : www.micom.de/lab/mthecam
[10] **Système de couleurs HSV** : https://fr.wikipedia.org/wiki/Teinte_saturation_lumi%C3%A8re

Station de soudage
Weller WE 1010



Harry Baggen (Elektor)

Le choix de fers à souder et de stations de soudage est énorme. Pour de nombreux électroniciens, quand il s'agit de leur outil principal, la marque Weller vient immédiatement à l'esprit. Leurs stations de soudage sont populaires et les prix y sont pour quelque chose. Dans ce banc d'essai, je m'intéresse à la station de soudage la moins chère de la série professionnelle, la WE 1010. Avec son prix de l'ordre de 150 €, elle peut séduire les amateurs sérieux.

Mon premier fer Weller était une station de soudage WTCP avec le fameux fer à souder Magnastat. La température était contrôlée dans la panne par un aimant qui perdait son attraction magnétique au-dessus d'une certaine température (effet Curie). On trouvait ces pannes pour différentes températures selon les besoins. La date de fabrication de mon exemplaire (**fig. 1**) est 10-1978 sur la plaque signalétique. Après 42 ans, il fonctionne toujours très bien. Le regain de mes activités électroniques à la maison justifie que j'achète une station de soudage moderne. Une avec un écran qui indique la température et un fer plus facile à manipuler que l'antique Magnastat.

On trouve de très belles stations de soudage, pour la plupart fabriquées en Chine, pour une somme modique. Mais dans l'offre de Weller j'ai aussi trouvé quelque chose d'intéressant pour pas trop cher : une version de base dans sa série *Professional* pour environ 150 €, ce qui correspond à ce que j'avais imaginé pour une nouvelle station de soudage. Les avantages espérés d'un outil de marque réputée par rapport aux alternatives chinoises : fiabilité (je croise les doigts), garantie et support à long terme et cordon du fer très flexible et résistant à la chaleur. Je compte bien utiliser ma nouvelle station de soudage encore pendant de nombreuses années.



Figure 1. Mon ancienne station Weller a plus de 40 ans et fonctionne toujours !



Figure 2. Affichage de la température de la panne et, en dessous, de la consigne.



Figure 3. Vous pouvez facilement changer de panne en dévissant le manchon métallique.

Je craque

C'était prévisible, j'ai craqué pour le WE 1010, sa station de base, son afficheur LCD et son fer à souder de 70 W. C'est un fer à souder à thermorégulation passive puisque élément chauffant et capteur ne sont pas placés dans la panne elle-même, mais derrière. Je pense que cela ne pose pas de problème, mais j'y reviendrai. Le carton contient une station de base avec le cordon d'alimentation, un fer à souder WEP 70, un support assez simple monté sur une base moulée, avec une éponge. Rien d'extraordinaire, mais une finition soignée. Sur la station, il y a un bouton marche/arrêt, un afficheur LCD et quelques boutons de commande.

L'afficheur n'est pas rétroéclairé, mais il est heureusement très clair et facile à lire, même en lumière tamisée. Hélas, l'interrupteur n'a pas non plus de témoin lumineux. Ainsi, pas moyen de voir au premier coup d'œil et de loin si la station est allumée ou pas. Pour s'en assurer, il faut regarder l'affichage ou la position de l'interrupteur (ligne rouge quand il est alimenté).

La station de base est assez lourde (presque 2 kg) et fait une impression de solidité. Le cordon du fer est long, 1,5 m, sous gaine en silicone, très souple et le fer tient bien dans la main grâce à son revêtement en mousse pressée. Le câble du fer a une fiche à 5 pôles à insérer dans la prise de la station de base, que l'on verrouille en la tournant. La partie métallique du fer est reliée à la terre du câble d'alimentation. La station n'a pas de fil de mise à la terre séparé.

Fonctionnement et utilisation

Le fonctionnement de la station de soudage est très simple. Deux boutons permettent de régler la température à la hausse ou à la baisse. À la livraison, elle est préréglée à 350 °C. Une touche de menu permet d'accéder aux réglages suivants : temps avant mise en veille, temps avant lequel la température du fer est réduite (max. 99 minutes) ; décalage (*offset*), pour la correction de la température et utile uniquement si vous pouvez mesurer la température de la panne (max. ±40 °C) ; commutation entre °C et °F et enfin la possibilité de verrouiller la station au moyen d'un code d'accès.

La température réelle de la panne est donnée en gros chiffres avec en dessous, en plus petits chiffres, la valeur de consigne. À côté de la consigne apparaît un symbole de chauffage lorsque le fer à souder chauffe. Ces symboles s'affichent même quand le fer est en veille ou que la fonction de verrouillage est utilisée.

Grâce à son poids, le support du fer tient bien sur la table. Il comporte un certain nombre de trous pour accueillir des pannes de rechange. Pour essuyer la panne, il y a bien sûr l'éponge jaune habituelle. Pas de place pour une boule de paille de laiton ou autre. Si pour essuyer votre panne vous préférez la paille métallique à l'éponge humide, il vous faudra un récipient supplémentaire pour y loger la paille.

Le fer à souder lui-même est assez fin et le long câble flexible est de belle facture. Cette flexibilité n'est pas un luxe, lorsqu'on soude, il ne faut qu'on sente ni le poids ni la raideur du câble. Les pannes peuvent être changées en dévissant simplement le manchon métallique. Comme le chauffage de la panne est indirect, les pannes de rechange sont bon marché, même les originales de Weller, disponibles (presque) partout. J'en ai fait une petite réserve, avec différentes formes.



Lumière !

Après la mise sous tension, j'ai attendu environ 35 s avant que la panne atteigne sa température de 350 °C. Pour les 380 °C exigés pour la soudure sans plomb, ce sont 5 s de plus. Ça va, non ? Le soudage avec le fer WE P70 se passe bien. Je n'ai éprouvé aucune difficulté pour souder des composants moyens et des pastilles de cuivre standard. Avec de grandes surfaces de soudure, la température baisse un peu et que la panne retrouve sa température de consigne en quelques secondes (c'est dans ces cas-là qu'un fer à régulation active réagit plus vite). C'est néanmoins très rapide grâce à la puissance de chauffe de 70 W. Pour le soudage de gros composants, il est conseillé d'utiliser une panne courte, large et biseautée pour favoriser l'échange thermique. La panne fournie d'origine est un peu trop étroite pour ça et convient mieux pour les petits composants.

Pour passer en mode de veille, la station surveille la quantité de

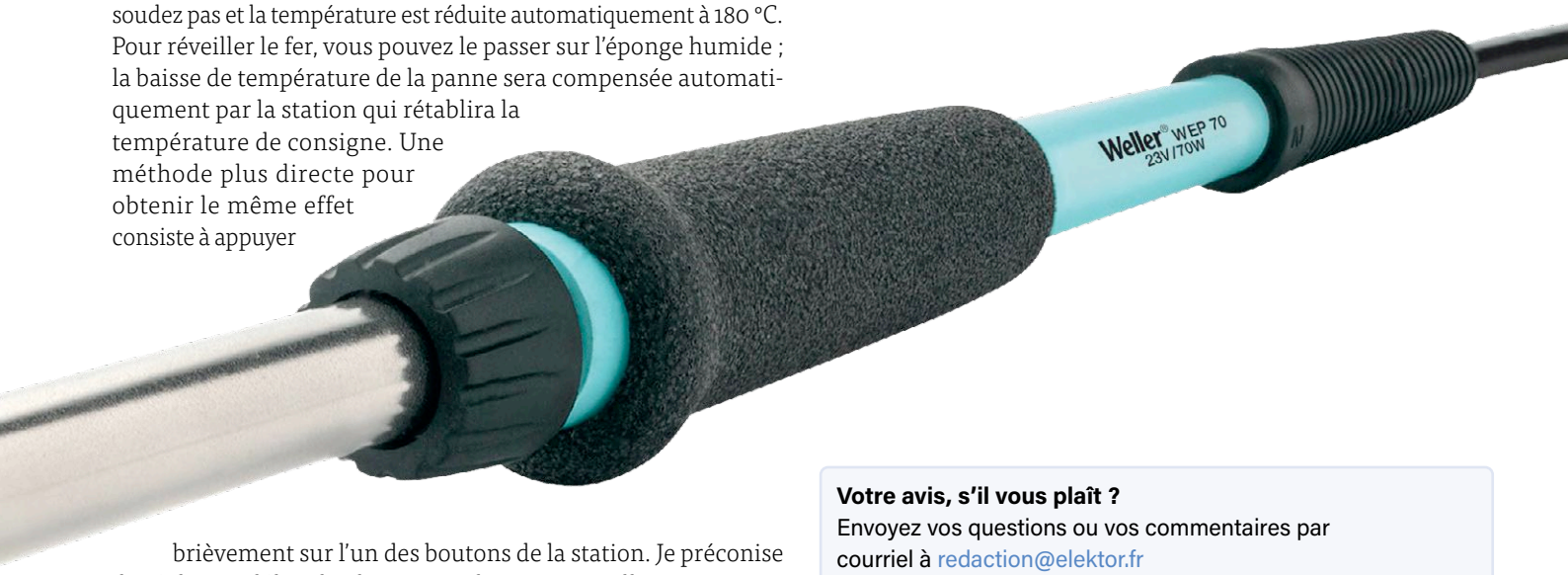


Figure 4. J'ai fait une petite réserve de pannes supplémentaires.



Figure 5. Quand le fer reste inutilisé pendant un certain temps, la station réduit automatiquement la température à 180 °C.

chaleur fournie par le fer plutôt que d'utiliser un détecteur de mouvement dans le fer à souder. Si la puissance calorifique reste constante pendant un certain temps, c'est le signe que vous ne soudez pas et la température est réduite automatiquement à 180 °C. Pour réveiller le fer, vous pouvez le passer sur l'éponge humide ; la baisse de température de la panne sera compensée automatiquement par la station qui rétablira la température de consigne. Une méthode plus directe pour obtenir le même effet consiste à appuyer brièvement sur l'un des boutons de la station. Je préconise de régler un délai plus long avant la mise en veille. 30 minutes par exemple, cela me convient. Vous pouvez désactiver complètement la mise en veille. Jusqu'à présent, je n'ai pas remarqué de dégradation des pannes. Et si cela devait arriver, une panne est si bon marché que ce ne serait pas une catastrophe de la remplacer.



Conclusion

Je suis très satisfait de ma station WE 1010 à laquelle mon ancienne station Weller va définitivement céder sa place. Le fer tient bien dans la main et les soudures sont impeccables. Les réglages sont limités, mais suffisants pour un usage normal. Il ne me manque qu'un témoin de mise sous tension plus voyant. J'ai du mal à croire qu'un interrupteur lumineux ou une LED témoin aurait coûté tellement plus cher ? En tout cas, j'aurais volontiers déboursé un peu plus pour l'avoir. La fonction de veille, on s'en accommode, même si ce n'est pas l'idéal. Ces tout petits défauts ne me font pas regretter mon achat. La WE 1010 est désormais ma station de soudage de qualité, j'en suis très satisfait, prix inclus.

200572-02

Votre avis, s'il vous plaît ?
Envoyez vos questions ou vos commentaires par courriel à redaction@elektor.fr

Ont contribué à cet article
Auteur : **Harry Baggen**
Photos : **Patrick Wielders**
Rédaction : **Eric Bogers**
Mise en page : **Giel Dols**
Traduction : **Alice Coper**

PRODUITS

› **Weller WE 1010 station de soudage (70 W)**
www.elektor.fr/18513

VIRTUAL ELECTRONICA FAST FORWARD 2020

LES GAGNANTS



electronica fast forward
the startup platform

powered by elektor

Clemens Valens (Elektor)

electronica Fast Forward, le concours de création d'entreprise, une initiative d'Elektor, est devenu *virtuel* en 2020. De janvier à octobre, de jeunes entreprises du monde entier spécialisées en électronique nous ont soumis leurs projets, leurs plans d'entreprise et leurs produits. Félicitations à tous les participants talentueux. Voici les gagnants en chair et en ohms.

Ces dernières années, Elektor organise le concours de création d'entreprise *Fast Forward*, conjointement avec les grands salons electronica et productronica. En 2020, les choses ont changé… Au lieu d'annuler l'événement, Elektor et electronica (https://electronica.de/) ont mis le concours "*electronica fast forward*" (e-ffwd) **en ligne**, ce qui a permis à de jeunes entreprises d'Allemagne, des États-Unis, de Hongrie, de France et d'autre pays de présenter leurs projets, leurs plans d'affaires et leurs produits. Le concours **electronica Fast Forward** est ouvert à toute jeune entreprise, mais toutes ne sont pas admises en finale. Seuls les participants qui ont envoyé un plan d'entreprise, un profil d'entreprise et une courte vidéo ont été présentés au jury. Cette vidéo tient lieu de présentation rapide que les participants font normalement en direct devant un public et un jury. La présence du public a été simulée par un vote avec attribution d'étoiles par un public constitué par les visiteurs des profils des candidats.

Gagnants de l'e-ffwd

Après avoir étudié les documents fournis et regardé les vidéos, puis fusionné les votes du public, le jury, présidé par le Pr Rik De Doncker de l'université RWTH d'Aix-la-Chapelle (cf. encadré) a décerné les prix aux trois gagnants en saluant leur potentiel commercial élevé et la qualité de leur équipe, deux critères importants pour qu'une start-up devienne une entreprise prospère. Félicitations à AXS Motionsystem [1], Micropack3D [2], et e-Fásli [3] !

Premier prix : AXS Motionsystem Ltd (Hongrie)

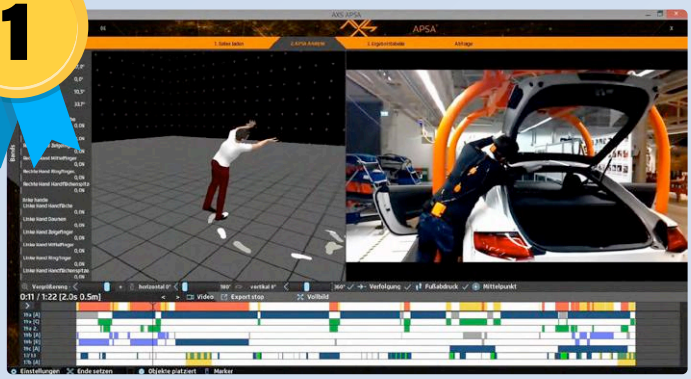
Fondée en 2014, AXS Motionsystem Ltd propose un système expert en ergonomie pour évaluer et qualifier les lieux de travail. Le système AXS capture numériquement les mouvements et les forces manuelles d'un travailleur pendant le déroulement du travail. Sur la base de ces données, l'évaluation se fait presque automatiquement selon différentes méthodes ergonomiques. Pour sa réalisation, AXS Motionsystem a remporté un budget marketing Elektor de 75.000 € et un stand de jeune entreprise sur le salon electronica 2022.

Deuxième prix : Micropack-3D (Allemagne)

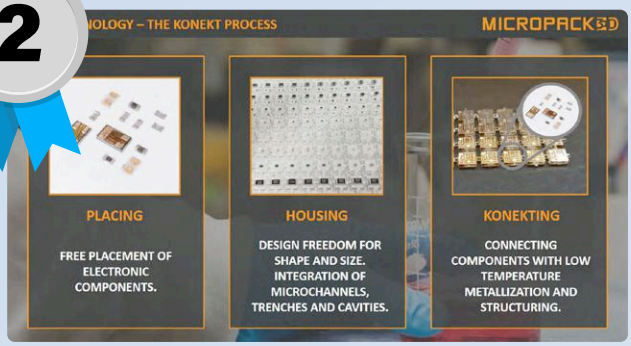
Micropack-3D est une start-up récente qui propose des boîtiers électroniques adaptatifs, pour loger et connecter des circuits. Son objectif est de faciliter la mise au point d'une électronique spécialisée parfaitement adaptée aux besoins, afin de réduire le précieux temps de développement. L'entreprise a remporté le deuxième prix avec un budget marketing Elektor de 50.000 €.

Troisième prix : e-Fásli Ltd (Hongrie)

e-Fásli est le premier appareil d'autoapprentissage de la thermothérapie. Posé sur le corps du patient, il peut maintenir un niveau de température entre 25 °C et 45 °C pendant des heures. Les paramètres et les données relatives au patient sont recueillis par



AXS Motionsystem, lauréat du premier prix avec son système expert en ergonomie pour évaluer et qualifier les lieux de travail.



Le deuxième Micropack3D propose des options de conditionnement innovantes pour l'électronique.



Le 4e prototype de l'appareil d'autoapprentissage de la thermothérapie d'e-Fásli Ltd.

une application mobile, puis des données anonymes sont envoyées au serveur, traitées pour déterminer les paramètres thérapeutiques optimaux et personnalisés. L'appareil est commandé via le WiFi. Pour sa réalisation, e-Fásli Ltd a remporté un budget de marketing Elektor de 25.000 €.

On se voit l'année prochaine ?

Grâce à la participation d'acteurs de haut niveau, le concours virtuel *electronica Fast Forward* 2020 [4] a été un succès. Visitez la page e-ffwd (www.elektormagazine.com/effwd-2020) pour mieux connaître tous les finalistes et leurs projets sur divers sujets :

- charge sans fil
- informatique de pointe
- véhicules électriques
- récolte d'énergie
- technologies d'actionnement
- solutions MEMS à miroir et à balayage laser
- contrôle et surveillance en temps réel
- et plus encore…

Espérons que la prochaine édition se déroulera en 2021 dans des conditions normales.

200584-03

LIENS

[1] **AXS Motionsystem Ltd** : www.elektormagazine.com/labs/axs-motionsystem-ltd

[2] **MicroPack3D** : www.elektormagazine.com/labs/micropack3d

[3] **e-Fásli Ltd** : www.elektormagazine.com/labs/e-fasli-ltd

[4] **Profils des participants au concours e-ffwd 2020** : www.elektormagazine.com/effwd-2020

Pr De Doncker

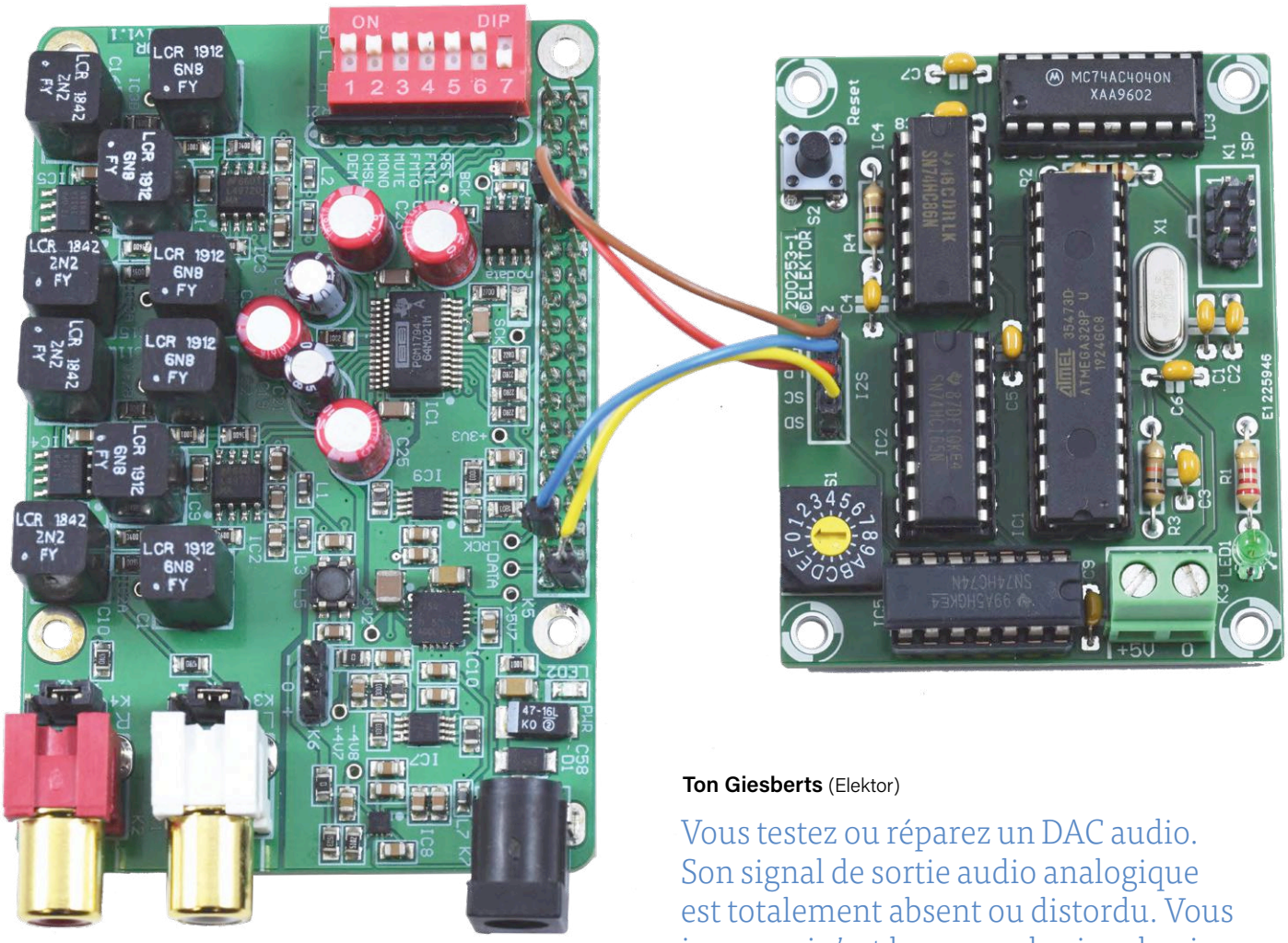


Le jury virtuel du Fast Forward Award était présidé par le professeur Rik W. De Doncker de l'université RWTH d'Aix-la-Chapelle.

Rik W. De Doncker (M'87 SM'99 F'01) a obtenu son diplôme d'ingénieur en 1981 puis son doctorat en génie électrique de la KULeuven, Belgique, en 1986. Il a été nommé professeur associé invité à l'université du Wisconsin, à Madison, en 1987. Il a rejoint le centre de recherche et de développement de GE Corporate, à Schenectady, NY, en 1988. Puis il rejoint la *Silicon Power Corporation* (anciennement GE-SPCO) comme vice-président de la technologie, développant le premier commutateur de transfert statique à moyenne tension au monde. Depuis 1996, il est professeur à l'université RWTH d'Aix-la-Chapelle, en Allemagne, où il dirige l'Institut pour l'électronique de puissance et les entraînements électriques (ISEA). Il a été nommé directeur du centre de recherche sur l'énergie E.ON de l'université RWTH d'Aix-la-Chapelle, où il a également fondé l'Institut pour la production d'électricité et les systèmes de stockage (PGS). Il est directeur du cluster RWTH CAMPUS pour l'énergie durable et dirige le programme de recherche CAMPUS sur les réseaux électriques flexibles du gouvernement fédéral allemand (BMBF). Il est docteur honoris causa de l'Université technique de Riga, en Lettonie.

géné audionumérique
de signaux de test I²S

Sinus numérique 1 kHz, à 32 bits, échantillonné à 192 kHz, réglable de 0 à –110 dB



Ton Giesberts (Elektor)

Vous testez ou réparez un DAC audio. Son signal de sortie audio analogique est totalement absent ou distordu. Vous ignorez si c'est la source du signal qui a un problème de logiciel… ou de matériel. Les deux peut-être. Est-ce le DAC lui-même qui déraille ? Ce générateur peut vous aider à trouver la réponse. Il délivre un signal de test numérique sinusoïdal précis, idéal aussi pour mesurer les performances analogiques de convertisseurs numérique-analogique ambitieux.

INFOS SUR LE PROJET

Mots-clés
audionumérique, Raspberry Pi, DAC, I²S

Niveau
débutant – connaisseur – expert

Durée
environ 4 h

Outils
soudure de composants traversants, AVRISP

Coût
Environ 15 €

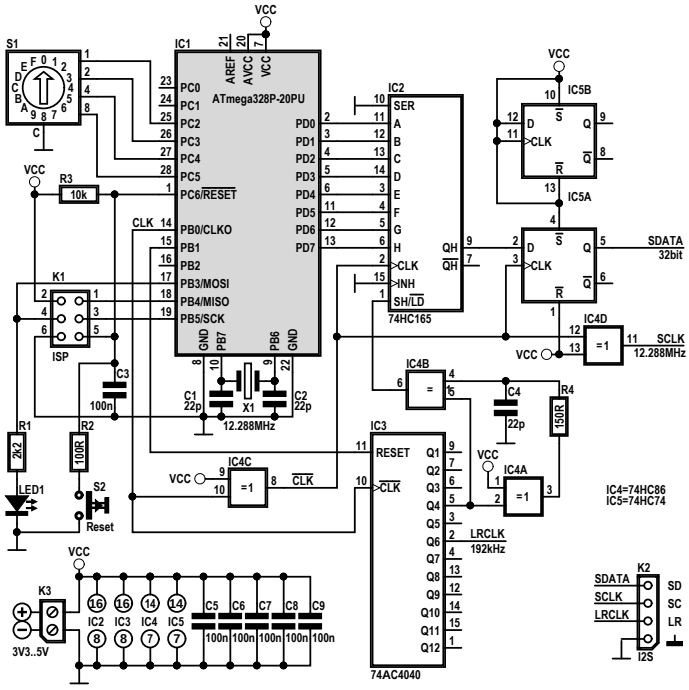


Figure 1. Schéma du générateur de signaux I²S.

Depuis son apparition en 1986, le bus I²S (*Inter-Integrated Circuit Sound*) est la norme de fait pour la transmission sérielle de signaux audionumériques. Au fil de l'étude de mon *Convertisseur numérique-analogique pour le Raspberry Pi* [1], j'ai eu l'idée d'un circuit spécifique qui produise un signal I²S pour tester le CN/A sans avoir besoin de connecter le RPi comme source de signal. Ce circuit peut bien sûr aussi être utilisé pour tester d'autres CN/A audio avec des entrées I²S, à condition qu'ils puissent gérer la fréquence d'échantillonnage de 192 kHz et des données audio codées sur 24 ou 32 bits.

Options de conception

Pour construire un générateur de signaux de test I²S, une option serait d'utiliser un convertisseur analogique-numérique (CA/N) à 24 bits avec des sorties I²S, avec un générateur de signaux (sinusoïdaux) comme entrée. Pour vérifier si les signaux de sortie analogiques du CN/A sont effectivement sans défaut, l'onde sinusoïdale du signal I²S doit être parfaite afin d'obtenir des mesures de distorsion correctes. Le signal de test ne doit en aucun cas être dégradé par une source analogique ou un CA/N médiocre.

Une alternative pour produire le signal I²S serait un microcontrôleur (µC) avec une table d'échantillons de 32 bits, calculés avec précision pour assurer la qualité des données audio. Cela donnerait un signal parfait pour les mesures de distorsion, dans ce cas un sinus de 1 kHz avec une fréquence d'échantillonnage de 192 kHz. Faut-il pour cette tâche un µC avec fonction I²S ou plutôt un µC commun comme l'ATmega328P sans I²S ? C'est un beau défi de construire un générateur de sinus numérique avec sortie I²S en utilisant ce µC et quelques composants, et ce projet montre que c'est possible ! Le micrologiciel de l'ATmega est en BASCOM-AVR.

Un peu de quincaillerie

L'objectif est d'obtenir un signal I²S avec des données de 32 bits à une fréquence d'échantillonnage de 192 kHz, ce qui est proche de la fréquence maximale d'échantillonnage du PCM1794A utilisé dans notre DAC audio RPi. L'horloge série (SCK ou SCLK) doit être de 12,288 MHz (2 canaux * 192 kHz * 32 bits). Comme la fréquence d'horloge maximale du µC est de 20 MHz, la seule façon de sortir des données série (SD ou SDATA) plus rapidement est d'utiliser un registre à décalage externe à entrée parallèle et sortie série et d'utiliser l'horloge du µC pour cadencer ce registre à décalage. PB0 doit être configuré comme CLKO (*Clock Out*) lors de la programmation des fusibles de l'ATmega328. Le schéma complet du générateur de signaux est sur la **figure 1**. Outre le registre à décalage, un compteur

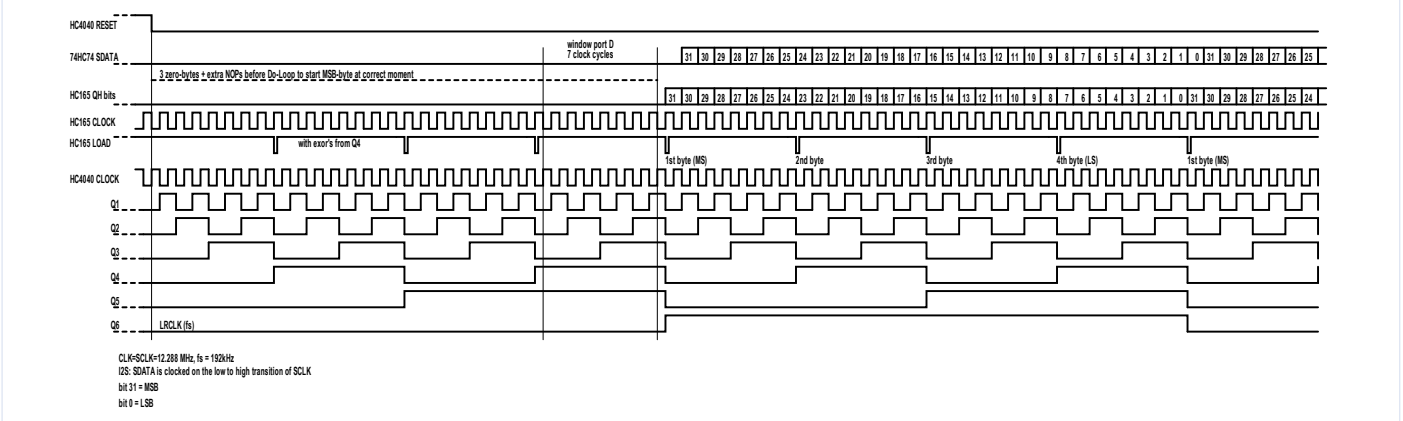


Figure 2. Chronogramme des signaux.

LISTE DES COMPOSANTS

Résistances

R1 = 2,2 kΩ
R2 = 100 Ω
R3 = 10 kΩ
R4 = 150 Ω

Condensateurs

C1,C2,C4 = 22 pF, C0G/NP0, pas 5 mm
C3,C5,C6,C7,C8,C9 = 100 nF, X7R, pas 5 mm

Semi-conducteurs

LED1 = LED, verte, 3 mm
IC1 = ATmega328P-PU, 20 MHz, DIP28
IC2 = 74HC165
IC3 = 74AC4040 – éviter la série HC !
IC4 = 74HC86
IC5 = 74HC74

Autres

K1 = embase à 2x3 broches, vertical, pas 2,54 mm
K2 = embase à 1x4 broches, vertical, pas 2,54 mm
K3 = bornier 5,08 mm, 2 voies, 630 V
S1 = commutateur rotatif à codage hexadécimal réel, THT (p.ex. Nidec Copal Electronics SD-1010)
S2 = poussoir 24 V, 50 mA, 6x6 mm
X1 = quartz 12,288 MHz, charge C 18 pF, 10 ppm, HC-49S
circuit imprimé 200253-1 v1.1

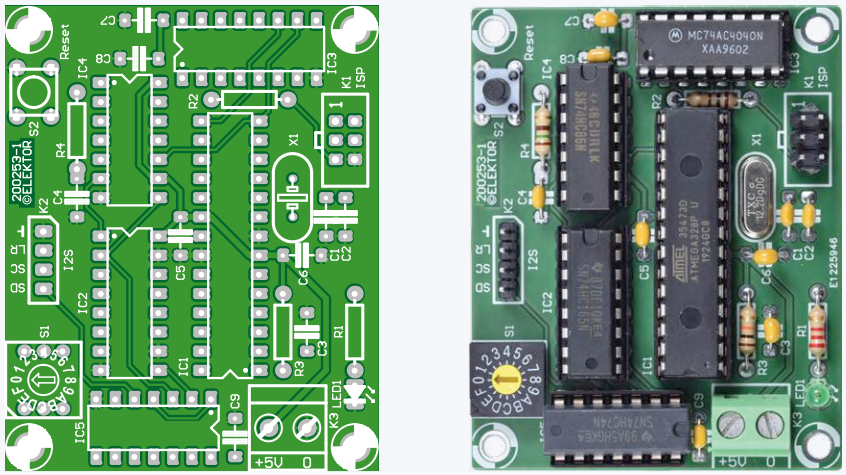


Figure 3. Dessin du circuit imprimé et photo de la carte assemblée.

asynchrone, une bascule et quelques opérateurs OU-exclusif soulagent le µC d'autres tâches (de synchronisation) et lui permettent de ne s'occuper que des octets des échantillons.

Toutes les autres tâches du logiciel, comme le calcul du sinus de 32 bits et la création de la table d'octets d'échantillonnage, doivent être effectuées avant le démarrage de la boucle principale du programme. Les échantillons de 32 bits sont divisés en quatre octets. Cela signifie que la table comportera 4 * 192 = 368 octets. Les voies gauche et droite utilisent le même signal, donc chaque groupe de quatre octets par échantillon doit être dupliqué. Le nombre de lignes occupées par cette partie de la boucle principale peut être calculé ainsi :

2 canaux * 4 octets * 192 échantillons *
4 lignes – 3 = 6141 lignes

Après le dernier octet d'échantillon de la boucle, les trois `NOP` sont omis puisque le redémarrage de la boucle dure trois cycles d'horloge, ce qui explique le «–3» dans ce calcul.

Le matériel, en détail

Le diagramme de synchronisation (**fig. 2**) peut aider à choisir les composants externes. Le bus I²S connaît trois signaux : les données série (SDATA) sont cadencées sur le front montant de l'horloge série (SCLK) et la ligne de sélection de mot (WS ou LRCLK) indique la voie audio (0 pour la gauche, 1 pour la droite). Sa fréquence est égale à la fréquence d'échantillonnage du signal audionumérique (192 kHz) et peut être dérivée de l'horloge série. Les données série et la ligne de sélection de mots doivent toutes deux changer sur le front descendant (HIGH-to-LOW) de l'horloge série. La famille logique HC est assez rapide en principe pour ces composants externes, bien que le temps de propagation des opérateurs doive être corrigé sur un point : un retard de 12 ns représente presque 15 % de la période d'horloge du µC.

Un 74AC4040 (IC3, compteur asynchrone binaire à 12 étages) est utilisé pour aligner correctement les signaux I²S. Ce compteur avance sur le front descendant de l'entrée d'horloge. Il dispose également d'une broche maîtresse de remise à zéro (11) utilisable pour le synchroniser avec le µC. La sixième bascule (broche 2) divise 12,288 MHz par 64 et produit exactement 192 kHz, la fréquence requise pour la ligne de sélection de mots.

Au début du programme, la la broche de r. à z. du compteur est activée (*high*) puis désactivée juste avant le début de la boucle principale (c'est-à-dire environ un quart de seconde après la mise sous tension, soit le temps de créer la table). Ce signal d'initialisation synchronise les données à la sortie du registre à décalage et de la ligne de sélection de mots. Les trois premiers échantillons de valeur 0 et quelques `NOP` sont utilisés de sorte que le premier octet est à la sortie SDATA au bon moment. Il y a une fenêtre de sept cycles d'horloge pour cadencer le premier octet de poids fort dans le registre à décalage. Chaque octet au port D est remplacé par le suivant après huit cycles d'horloge. En d'autres termes, le moment où le premier octet de poids fort du port D peut être chargé pourra varier de sept cycles d'horloge par rapport à l'impulsion de chargement du registre à décalage (voir le diagramme de synchronisation). Les données à la sortie du port D changent quelques nanosecondes après le front montant de CLKO (PB0).

Un registre à décalage de 8 bits à entrée parallèle/sortie série 74HC165 (IC2) est utilisé pour les données série. Il possède une broche de chargement parallèle active au niveau bas (broche 1) et une horloge (broche 2) avec inhibition (broche 15, validation par un niveau bas) qui ont la même fonction (toutes deux connectées en interne à un opérateur logique OU). En fonction de l'emplacement des composants, l'interversion des connexions peut simplifier le routage. Les données sont décalées sur le front montant de l'horloge. L'entrée série (broche 10) inutilisée est connectée à la masse. L'impulsion de charge (LD, active au niveau bas) pour le registre à décalage est dérivée de la sortie du compteur Q4 en utilisant un quadruple

opérateur logique OU exclusif à 2 entrées 74HC86 (IC4). Le signal à la broche 4 de IC4B est inversé et retardé par le temps de propagation de IC4A et supplémentaires dues à quelques nanosecondes par R4/C4. En raison de la fonction OU exclusif, chaque changement de Q4 donne à la sortie de IC4B une courte impulsion active au niveau bas. L'impulsion est suffisamment longue pour charger les nouvelles données dans le registre à décalage, mais suffisamment courte pour être inactive avant le front montant de l'horloge. Pour charger les données du port D dans le registre à décalage au bon moment, l'impulsion doit être active juste après le front montant de l'horloge (broche 1). Cela signifie que l'horloge du registre à décalage doit être inversée, ce que fait IC4C.
Le bit de poids fort des données série du bus I²S arrive une période d'horloge après le changement de sélection de mot. Une bascule D supplémentaire est nécessaire. C'est le 74HC74 (IC5), une double bascule D avec *set* et *reset* et active sur flanc ascendant. Ce faisant, le signal pour SDATA est retardé. Pour compenser cela, le signal d'horloge inversé de IC4C est inversé à nouveau par IC4D pour faire coïncider l'horloge série avec les données série.

Ajuster le niveau de sortie

Au lieu de produire simplement un sinus de 1 kHz avec un niveau fixe, quatre entrées du port C (avec résistance de polarisation interne) sont connectées à un commutateur rotatif à codage hexadécimal (SD-1010) pour ajuster le niveau de sortie. Cette fonction permet de vérifier la linéarité du CN/A. Un micro-interrupteur DIP à quatre voies aurait convenu aussi, mais c'est tout de même plus intuitif avec un sélecteur qui tourne comme un bouton de volume.
Il existe des commutateurs rotatifs à codage dit *réel* ou *complémentaire*. J'utilise le codage *réel*, mais il suffit de modifier le logiciel pour utiliser un autre codage. En position 0, les quatre interrupteurs sont ouverts. Pour modifier le niveau de sortie du signal audionumérique, le µC doit être initialisé pour qu'il recalcule les valeurs de la table d'échantillons ; c'est le rôle de S2 sur lequel on appuiera quand on a modifié le niveau de sortie. À chaque nouveau réglage de niveau, l'instruction `Select-Case` règle le facteur d'échelle U (type Double) correct.
Les valeurs correctes des facteurs d'échelle pour chaque niveau de sortie sont définies comme constantes dans le programme, non seulement pour éviter des calculs supplémentaires, mais aussi par manque de précision des calculs faits avec la fonction `LOG` de BASCOM.

Calcul du sinus

Nous avons vu que le programme calcule la table d'échantillonnage des sinus immédiatement après la mise sous tension ou lors d'une initialisation, en tenant compte du réglage du niveau du commutateur S1. Nous ne nous contenterons pas d'un générateur de signaux I²S quelconque, mais voulions une sinusoïde de test parfaite de 1 kHz avec une résolution de 32 bits. Dans un premier temps, pour les calculs j'ai utilisé l'instruction `SIN(x)` de BASCOM-AVR, mais gare à l'imprécision ! Exemple :

```
DIM Pi,A,X As Single
Pi = 3,14159265358979323846264338
X = Pi / 2
A = NAS(X)
Imprimer "Sin(Pi/2) = " ; A
```

Voici le résultat de ce calcul :
Sin(Pi/2) = 0,99999332 au lieu de 1.
Pour X = Pi/6, le résultat obtenu est 0,499993796, au lieu de 0,5. Ça ne convient pas pour calculer un sinus précis. La seule option est de calculer l'onde sinusoïdale en utilisant le polynôme de Taylor pour `SIN(X)` avec un nombre suffisant de termes :

```
SIN(X) = X-(X^3)/3!+(X^5)/5!-
   (X^7)/7!+(X^9)/9!-
   (X^11)/11!+(X^13)/13!-
   (X^15)/15 !
```

Les calculs dans BASCOM-AVR ne peuvent être effectués que sur deux opérandes ; le polynôme sera donc calculé sur plusieurs lignes de code, comme vous le constaterez dans le code source.
Pour accélérer le calcul, au lieu de factoriser on utilise des constantes (F3 = 6 … F15 = 1307674368000). Toutes les variables des calculs sont de type Double, des nombres binaires signés de 64 bits (8 octets, 5 x 10^-324 à 3,4 x 10^308). Avec le polynôme de Taylor, nous obtenons les résultats suivants, plus précis :

```
Sin(Pi/6) = 500E-3
Sin(Pi/2) = 999.999999993977E-3
```

Pour diviser le résultat du calcul de `SIN(X)` en quatre octets, il doit d'abord être converti en variable de type Long, signée, nombres binaires de 32 bits (–2147483648 à 2147483647). Le résultat du calcul de `SIN(X)` est compris entre –1 et 1. Si nous voulons que les données de 32 bits aient un niveau de pleine échelle, la valeur de `SIN(X)` doit être multipliée par (2^32)/2 – 1 :

```
SINX = SINX * U
```

où U = 2147483647
La conversion en Long peut être faite en une seule ligne, la variable Long prend la valeur d'une variable Double. Le résultat du calcul est alors disponible sous la forme d'une valeur signée de 32 bits :

```
SINXlong = SINX
```

Pour coder cette valeur sur quatre octets, il suffit de décaler les bits de SINXlong et de les stocker dans quatre variables de type Byte. Une routine appelée `SampleX` effectue tous ces calculs pour toute valeur de X. Le calcul n'est vraiment précis que pour X = –π/2 à +π/2. Ainsi, le sinus est d'abord calculé de –π/2 à +π/2, en prenant 97 échantillons (quatre octets par échantillon). Le reste de la table est complété par copie symétrique des éléments de la première table par groupe de quatre octets, en prélevant 95 échantillons supplémentaires. L'échantillon 193 est le même que le premier de la table (quatre octets). D'où repart la boucle `Do`.
J'aurais pu créer une table avec seulement des

LIENS

[1] **Audio DAC for Raspberry Pi** : www.elektormagazine.com/labs/audio-dac-for-rpi-networked-audio-player-using-volumio
[2] **Téléchargement (logiciel et Gerber)** : www.elektormagazine.fr/200253-02
[3] **La page de ce projet sur Elektor Labs** : www.elektormagazine.com/labs/32-bit-i2s-sine-wave-generator-200253

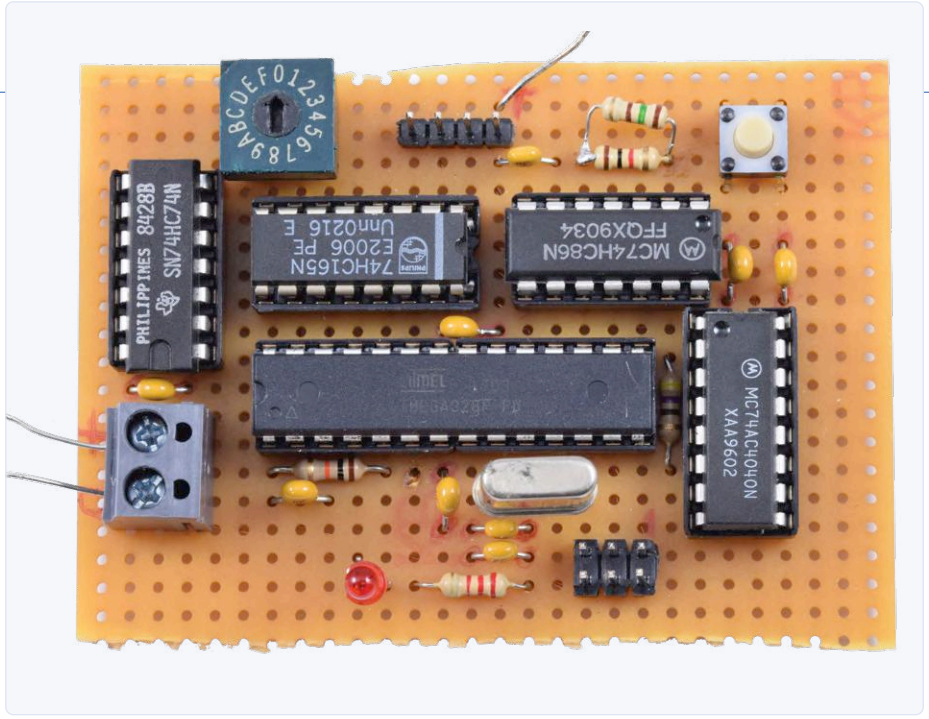


Figure 4. Le prototype du générateur sur plaque d'essai.

échantillons de sinus de l'intervalle de –π/2 à +π/2 puis sélectionner les éléments corrects de la table dans la boucle principale pour compléter les données pour une période complète du sinus sur le port D. Un signal asymétrique plus complexe qu'une onde sinusoïdale aurait pu être intéressant, mais à d'autres fins. Ici l'utilisation d'une table complète donne une boucle principale plus lisible.

Grâce au registre à décalage, le µC dispose de huit cycles d'horloge pour traiter chaque octet. En BASCOM-AVR, dans une boucle `Do`, l'instruction `PORTD = A(n)`, où A(n) est un élément d'une table d'octets, ne prend que cinq cycles d'horloge, ce qui est étonnamment rapide. Pour que chaque octet prenne huit cycles d'horloge, il faut rajouter trois `NOP` après chaque octet. Il faut compter trois cycles d'horloge pour le redémarrage de la boucle `Do`, pile ce qu'il faut pour une période complète des 192 échantillons du sinus de 1 kHz. La boucle principale du programme n'a donc rien d'autre à faire que d'émettre les octets de la table sur le port D.

Assemblage du circuit

Le dessin de circuit imprimé et le plan d'implantation de la **fig. 3** sont disponibles en téléchargement avec les fichiers Gerber pour les PCB. Vous pouvez également utiliser un morceau de circuit d'expérimentation comme moi pour mon prototype (**fig. 4**).

Si vous ne trouvez pas de fournisseur pour le 74AC4040, il est possible de couper la piste entre les broches 13 et 14 de IC4 ; puis, en mettant la broche 13 à la masse avec un fil, vous *pouvez* utiliser un 74**H**C4040, mais la chronologie des signaux devient critique et pourrait poser un problème avec le CN/A à tester. Chez moi, avec le CN/A pour Raspberry Pi, ça a marché. La connexion entre K2 du générateur au CN/A sera aussi courte que possible.

La tension d'alimentation de ce géné doit correspondre à celle du CN/A testé. À VCC = 3,3 V, sa consommation dépasse à peine 20 mA.

Modifier le logiciel...

Le programme occupe 77 % de la mémoire flash. Le calcul du sinus décrit dans cet article représente environ 10 %. Il reste donc un peu de place pour des fonctions supplémentaires. Notez que la version gratuite de BASCOM-AVR *ne* convient *pas* pour compiler le logiciel en raison des limitations de mémoire de programme. Pour modifier le programme, il vous faudra la version complète.

Le logiciel téléchargeable [2] contient le fichier HEX original. Ce circuit peut donc être construit sans licence de la version complète de BASCOM-AVR, il suffit d'une interface de programmation AVR-ISP et d'un logiciel de programmation comme Atmel AVR Studio ou AVRDUDE pour mettre en marche votre ATmega128.

...ou le matériel

Quelques broches d'E/S inutilisées du µC peuvent être mises à contribution pour des fonctions supplémentaires : autres formes d'onde, autres fréquences. Pour cela il faudra modifier le logiciel. Les broches PB2, PC0 et PC1 du µC peuvent être utilisées, mais PB4 et PB5 sont réservées à la programmation *in situ* (ISP). Les composants que vous rajouterez ne doivent pas interférer avec cette interface de programmation.

La LED1 indique que le processeur calcule les données de l'échantillon audio, mais comme cela ne dure qu'un quart de seconde, PB3 pourrait servir aussi pour d'autres fonctions. Dans ce cas, R1 peut être connectée à l'alimentation comme témoin de mise sous tension.

Si vous ajoutez des fonctions à ce générateur de signaux I²S, faites-le-moi savoir. Partagez aussi vos remarques ou suggestions sur la page de ce projet Elektor Labs [3] !

200253-02

Votre avis, s'il vous plaît ?
Adressez vos questions ou vos commentaires (en anglais) à l'auteur
t.giesberts@elektor.com

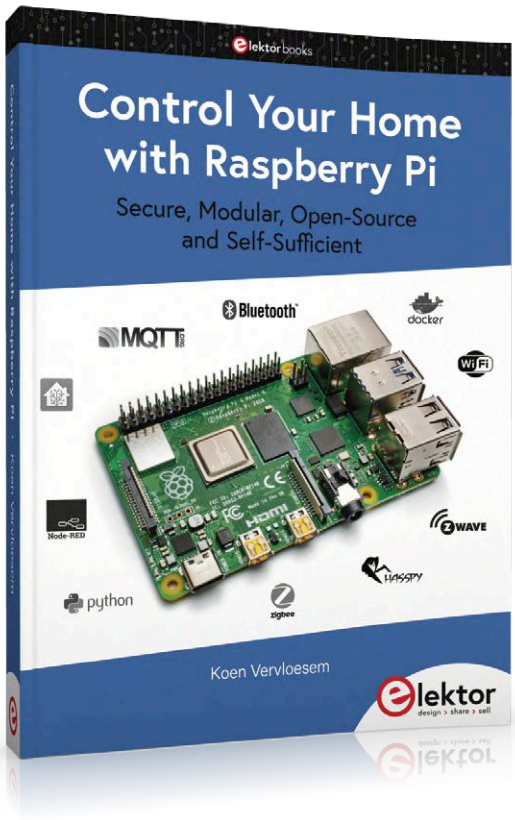
Ont contribué à cet article
Auteur : **Ton Giesberts**
Illustrations:
Ton Giesberts, Patrick Wielders
Rédaction : **Luc Lemmens**
Maquette : **Giel Dols**
Traduction : **Alice Coper**

PRODUITS

› **RPi High End Audio DAC - PCB nu (160198-1)**
www.elektor.fr/rpi-high-end-audio-dac-pcb-160198

› **Raspberry Pi High End Audio DAC – Module (160198-91)**
www.elektor.fr/raspberry-pi-high-end-audio-dac-module-160198-91

Raspberry Pi
aux commandes de la maison
RPi furète sur 433,92 MHz



Koen Vervloesem (Belgique)

Voici un chapitre extrait de l'ouvrage *Control Your Home with Raspberry Pi* de Koen Vervloesem récemment publié par Elektor. Il explique comment transformer votre RPi en un puissant récepteur et décodeur pour la plupart, sinon la totalité, des télécommandes et signaux que vous pouvez capter sur 433,92 MHz, l'une des fréquences ISM (industrielles, scientifiques et médicales) les plus connues et les plus utilisées.

En Europe, où je vis, tous les bricoleurs connaissent la fréquence de 433,92 MHz utilisée par de nombreux appareils sans fil bon marché : ouvre-portes de garage, capteurs météo et sonnettes. En outre, le matériel pour communiquer avec ces appareils est tout aussi peu cher.
La plupart de ces appareils utilisent une communication radio hélas non chiffrée, et s'ils emploient une forme de sécurité, elle est assez faible et/ou avec un algorithme propriétaire qui n'inspire pas confiance. Il y a tellement d'appareils et ils sont tellement peu chers que vous ne pouvez les ignorer. Par sécurité, je n'utilise que des capteurs 433,92 MHz : j'en ai un dans presque toutes les pièces de ma maison, ainsi qu'à l'extérieur. Je ne leur ferais pas confiance pour des tâches critiques.
Je montre ici comment lire les mesures de ces capteurs de température sans fil et comment les transmettre à votre courtier MQTT pour les injecter ensuite dans votre système domotique.
Note : je parle de 433,92 MHz, mais selon l'endroit où vous vivez, ce sera une autre fréquence, par exemple, en Amérique, 915 MHz. Achetez les bons appareils pour votre pays.

Protocoles 433,92 MHz

Les appareils qui utilisent la fréquence 433,92 MHz opèrent dans la bande de fréquence ISM sans licence. La fréquence est une chose, le protocole utilisé en est une autre. Il n'y a pas de protocole standard pour cette fréquence. Il n'y a pas de Z-Wave ou de Zigbee. Toutefois, beaucoup de protocoles de ces appareils ont été décodés par rétro-ingénierie et vous pouvez dialoguer avec eux à condition d'avoir un émetteur-récepteur pour la bande autour de 433,92 MHz et le bon logiciel pour décoder et/ou coder le protocole.

Quelques appareils intéressants :

- capteurs de température, d'humidité et de météo d'Alecto, Cresta, La Crosse, et Oregon Scientific ;
- détecteurs d'ouverture de porte/fenêtre avec capteur à effet Hall ;
- interrupteurs et variateurs d'Energenie, KlikAanKlikUit et LightwaveRF ;
- carillons de porte de Byron et Chacon.

Vous pouvez aussi trouver chez AliGood et BangExpress des appareils encore moins chers qui prennent en charge les mêmes protocoles. Et il y a même de petits circuits imprimés tel que l'émetteur STX882 que vous pouvez connecter à un microcontrôleur ou à une carte Arduino pour créer vos propres cartes de capteurs sans fil. Je mets l'accent ici sur les premiers types d'appareils, les capteurs de température et d'humidité. Vous pouvez en trouver pour moins de 10 € (**fig. 1**), même encore moins chez AliMachin ou BangTruc. Leur portée est assez bonne : je peux recevoir les capteurs partout dans ma maison, y compris dans mon réfrigérateur et mon congélateur, et même sur ma terrasse.



Figure 1. Pour environ 5 €, vous pouvez trouver un capteur de température et d'humidité chez DANIU qui transmet ses valeurs sur 433,92 MHz et les affiche en clair sur un écran.

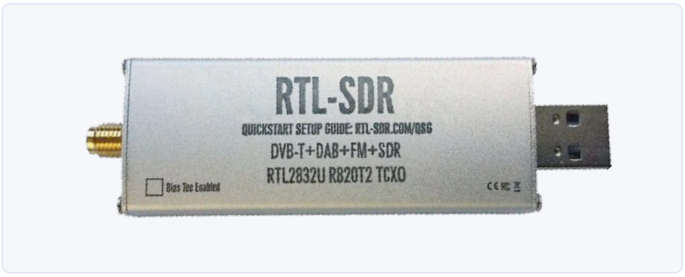


Figure 2. Le RTL-SDR décode de nombreux signaux sans-fil, y compris les stations météo qui émettent en 433,92 MHz.



Figure 3. Avec le trépied, le support de dipôle et les antennes télescopiques du kit RTL-SDR, vous avez tout ce qu'il faut pour recevoir les mesures de tous vos capteurs 433,92 MHz.

Besoins en matériel

Pour que votre RPi puisse recevoir les mesures d'un capteur 433,92 MHz, il vous faut un récepteur et une antenne.

Récepteur

Un type de récepteur répandu pour les projets en 433,92 MHz est une clé DVB basée sur le RTL2832 de Realtek. Oui, vous avez bien lu, DVB comme dans *Digital Video Broadcasting* (= diffusion vidéo numérique). La puce RTL2832 de beaucoup de ces clés peut faire beaucoup plus que de décoder des signaux vidéo numériques : avec le bon logiciel, ça peut faire une véritable radio logicielle (SDR) !

Donc, si vous récupérez une vieille clé DVB, vous pourriez l'utiliser pour recevoir les signaux de vos capteurs météo. Sinon, le RTL-SDR (*https://www.rtl-sdr.com*) est un bon choix (**fig. 2**). Vous trouverez des variantes de cette clé pour 25 € et en kit avec une antenne et d'autres accessoires pour ±45 € [1]. J'ai vu des clé DVB à environ 7 € chez AliAllezy qui fonctionnent très bien pour cet usage, mais ne les ai pas testées.

Antenne

Ce qu'il vous faut ensuite c'est une bonne antenne. Il y a des livres entiers sur ce vaste sujet, mais je ne suis pas spécialiste. Je sais que pour choisir l'antenne il faut connaître sa longueur qui dépend de la longueur d'onde du signal. La longueur d'onde est égale à la vitesse (en m/s) divisée par la fréquence (en Hz), et se mesure en mètres (m). Faisons le calcul pour la communication en 433,92 MHz. Dans l'air, la vitesse de l'onde est pratiquement la vitesse de la lumière. D'où la longueur d'onde :

299 792 458 m/s / 433 920 000 Hz = 0,69 m.

La longueur d'onde entière est donc de 69 cm, la demi-longueur d'onde de 34,5 cm et le quart d'onde de 17,25 cm. Ce sont les longueurs optimales théoriques d'une antenne pour recevoir des transmissions à 433,92 MHz. En pratique, divers facteurs influencent les caractéristiques de l'antenne, dont son emplacement, et une règle empirique est de soustraire 5 % à cette longueur théorique. Ceci n'est pas un livre sur les antennes, je ne m'étendrai pas davantage. De plus, pour lire des capteurs dans votre maison, la qualité de votre antenne est secondaire. Il y a des chances que cela fonctionne juste avec une « antenne 433 MHz » achetée pour quelques euros chez AliLi ou BangLaba. Si vous ne voulez pas prendre de risque, prenez une antenne incluse dans un kit avec le RTL-SDR. Le kit d'antenne officiel (**fig. 3**) comporte des antennes dipôles télescopiques extensibles de 5 cm à 1 m, ce qui couvre les longueurs d'onde optimales pour la fréquence 433,92 MHz [2].

Recevoir des valeurs de capteur avec rtl_433

Du côté logiciel, une solution courante pour lire les signaux 433,92 MHz est « rtl_433 » (*https://github.com/merbanan/rtl_433*), qui malgré son nom est un récepteur de données générique, principalement pour les bandes ISM 433,92 MHz, 868 MHz (SRD), 315 MHz, 345 MHz et 915 Hz.

Toutes les clés DVB à base de RTL2832 devraient fonctionner avec rtl_433, y compris la clé officielle RTL-SDR. J'utilise la clé RTL-SDR avec une antenne dipôle du kit d'antenne SDR-RTL. Connectez juste l'antenne au RTL-SDR et le RTL-SDR à un port USB de votre RPi.

Avertissement : Le RTL-SDR chauffe beaucoup en fonctionnement. Ventilez-le bien.

Installation de rtl_433toMQTT

Le programme rtl_433 est maintenu activement et contient plus de 150 décodeurs de protocole pour divers appareils qui transmettent sur 433,92 MHz. De plus, il peut envoyer et recevoir les valeurs à un courtier MQTT. Par chance, quelqu'un a créé un conteneur Docker avec rtl_433 dans ce but précis (*https://github.com/bademux/rtl_433toMQTT*) [3].

Créez d'abord un répertoire pour le conteneur :

```
mkdir -p /home/pi/containers/rtl433tomqtt
```

Puis il faut ajouter la définition du conteneur à votre fichier *docker-compose.yml* :

```
version: '3.7'
  service:
  mosquitto:
  # mosquitto config
  rtl433tomqtt:
  image: bademux/rtl_433tomqtt:latest
  container_name: rtl433tomqtt
  restart: always
  volumes:
   - ./containers/rtl433tomqtt:/home/user/.config/
rtl_433:ro
  - /etc/localtime:/etc/localtime:ro
  devices:
  - /dev/bus/usb:/dev/bus/usb
```

Entrez la configuration du conteneur Docker pour Mosquitto au lieu de `#mosquitto config`. Le conteneur doit avoir accès au bus USB pour communiquer avec le RTL-SDR. Notez aussi que le répertoire que vous avez créé est monté en tant que volume. Vous n'avez pas à créer un fichier de configuration pour l'instant.
Créez d'abord une règle `udev` pour donner les bonnes autorisations au dispositif USB :

```
sudo nano /etc/udev/rules.d/20.rtl-sdr.rules
```

Saisissez la ligne suivante :

```
SUBSYSTEM==»usb», ATTRS==»0bda»,
   ATTRS==»2838»,OWNER=»pi», MODE=»0660»
```

Enregistrez le fichier avec CTRL+O puis quittez nano avec CTRL+X. Débranchez puis rebranchez le RTL-SDR. Examinez maintenant la liste des dispositifs USB connectés :

```
lsusb
```

Recherchez dans cette liste une ligne comme celle-ci :

```
Bus 001 Device 008: ID 0bda:2838 Realtek
   Semiconductor Corp. RTL2838 DVB-T
```

Notez les numéros de bus et de dispositif : 001 et 008. Dans les fichiers de dispositifs, recherchez à présent le bus 001 :

```
ls -l /dev/bus/usb/001
```



Figure 4. La commande rtl_433 trouve automatiquement le récepteur RTL-SDR et commence à afficher les mesures reçues des capteurs.

Vous devriez voir une ligne comme celle-ci :

```
crw-rw---- 1 pi root 189, 10 May 7 20:22 008
```

Elle montre que le dispositif appartient à l'utilisateur `pi`, qui dispose des droits de lecture et d'écriture.
Après cela, créez le conteneur avec :

```
docker-compose up -d
```

Après la création du conteneur, consultez son journal :

```
docker logs -f rtl433tomqtt
```

Comme la **fig.4** le montre, vous devriez voir quelques messages du programme rtl_433 qui a essayé de trouver un fichier de configuration en divers endroits, enregistré plus de 120 décodeurs de protocoles et trouvé un dispositif de réception. Vous devriez ensuite voir un message « *Tuned to 433,92 MHz* » et si tout va bien vous devriez voir maintenant des valeurs de capteurs arriver. Soyez patient, car beaucoup de ces capteurs n'émettent qu'une fois par minute.

Configurer rtl_433

Au début du journal, vous avez vu que rtl_433 essayait de trouver des fichiers de configuration. Il n'en a pas trouvé et a donc utilisé une configuration par défaut, ce qui était bien pour tester mais il n'a pas utilisé MQTT. Copions maintenant un exemple de fichier de configuration à un endroit parcouru par rtl_433 :

```
docker cp rtl433tomqtt:/usr/local/etc/rtl_433/
   rtl_433.example.conf /home/pi/containers/
   rtl433tomqtt/rtl_433.conf
```

Comme j'ai fait coïncider le répertoire *containers/rtl433tomqtt* avec */home/user/.config/rtl_433* dans votre répertoire d'accueil de l'utilisateur pi, vous pouvez maintenant éditer ce fichier de configura-



Figure 5. La commande rtl_433 peut afficher les valeurs des capteurs sous de nombreux formats, y compris JSON.



Figure 6. La commande rtl_433 peut envoyer les valeurs reçues des capteurs vers votre courtier MQTT.

tion et redémarrer le conteneur pour l'utiliser. Par exemple, vous pouvez désactiver les protocoles dont vous n'avez pas besoin ou activer des protocoles désactivés par défaut.

Le fichier de configuration est abondamment commenté, ce qui devrait vous aider à déterminer quoi changer. De plus, vous trouverez de nombreuses informations dans la documentation en ligne de rtl_433 (*https://triq.org/rtl_433/*), notamment les étapes pour prendre en charge un capteur qui ne l'est pas (encore).
Après avoir modifié votre fichier de configuration, redémarrez le conteneur :

```
docker restart rtl433tomqtt
```

Si tout va bien, le journal devrait montrer que rtl_433 s'arrête de chercher un fichier de configuration après le second fichier et le trouve :

```
Reading conf from «/home/user/.config/rtl_433/
   rtl_433.conf».
```

Avec le fichier de configuration en exemple, rtl_433 montre maintenant la sortie de chaque capteur sous forme de dictionnaire JSON (**fig. 5**).
C'est déjà un grand pas pour l'intégration de vos capteurs 433,92 MHz dans votre installation domotique, mais la dernière étape est la publication de ces valeurs auprès de votre courtier MQTT.

Publier les valeurs du capteur 433,92 MHz dans MQTT

Le programme rtl_433 prend déjà en charge l'émission des valeurs de capteur qu'il reçoit vers un courtier MQTT. Ouvrez le fichier de configuration (*/home/pi/containers/rtl433tomqtt/rtl_433.conf*) et trouvez cette ligne :

```
output json
```

Vous pouvez conserver cette ligne car on peut spécifier plusieurs sorties, ou la modifier en sortie `kv` si vous préférez la sortie par défaut plus conviviale. Ajoutez la ligne suivante pour définir une sortie MQTT :

```
output mqtt://mosquitto:1883,user=home,pass=PASSWORD
```

Assurez-vous d'indiquer un nom d'utilisateur et un mot de passe corrects pour votre courtier MQTT.
Avertissement : Le programme rtl_433 ne prend pas en charge MQTT sur TLS. Si vous exécutez le conteneur `rtl_433toMQTT` sur le même RPi que votre conteneur mosquitto, ce n'est pas un problème qu'ils communiquent en clair, de toutes façons ils sont sur la même machine. Si votre récepteur RTL-SDR est sur une autre machine que votre courtier MQTT (par exemple parce que vous y avez une meilleure couverture), je vous recommande d'exécuter un conteneur `mosquitto` en plus du conteneur `rtl_433toMQTT` et de le configurer comme passerelle vers votre courtier MQTT principal via une connexion chiffrée. Voyez le supplément à la fin du livre pour plus de détails.

Après redémarrage de votre conteneur, vous devriez voir un message MQTT publié sous le sujet principal rtl_433 pour chaque valeur de capteur. Vous pouvez le visualiser avec :

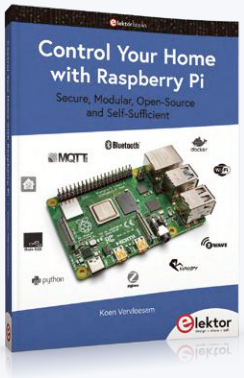
```
mosquitto_sub -t 'rtl_433/#' -v
```

Par exemple, chaque fois que le capteur de température et d'humidité à l'intérieur de mon congélateur transmet un message, je vois des données comme en **fig. 6**.
Vous pouvez maintenant modifier un peu la configuration de MQTT. Pour plus de détails, consultez les commentaires dans l'exemple de fichier de configuration. Par exemple, cette configuration modifie le sujet MQTT en quelque chose de plus court :

```
output
  mqtt://mosquitto:1883,user=home,pass=PASSWORD,
   devices=rtl433/[model]/
  [channel]/[id]
```

Et si vous n'êtes pas intéressé par les métadonnées de bas niveau concernant la connexion radio, comme la modulation, la fréquence, le RSSI (indicateur de niveau du signal reçu), le SNR (rapport signal/bruit) et le bruit, il suffit de supprimer ou de commenter la ligne `report_meta level` dans le fichier de configuration.

Vous y êtes : toutes les mesures de vos capteurs 433,92 MHz sont envoyées à votre courtier MQTT, et vous pouvez souscrire à leurs sujets dans vos scripts Python ou vos autres logiciels de domotique.



LIVRE « CONTROL YOUR HOME WITH RASPBERRY PI »

Ce livre vous explique comment automatiser votre domicile avec un RPi. Vous apprendrez comment utiliser les divers protocoles sans fil pour la domotique, comme le Bluetooth, les ondes radio 433,92 MHz, Z-Wave et Zigbee. Bientôt, vous automatiserez votre domicile avec Python, Node-Red et Home Assistant, et vous serez même capable de parler à votre système domotique. Tout ceci en toute sécurité, avec un système modulaire, à code source totalement ouvert, sans recourir à des services tiers. C'est vous qui serez aux commandes de votre domicile, et personne d'autre.

À la fin du livre, vous pourrez installer et configurer votre RPi comme une passerelle domotique très flexible pour les protocoles de votre choix, et joindre divers services avec MQTT pour en faire votre propre système.

- Transformez votre RPi en une passerelle fiable pour divers protocoles domotiques.
- Rendez votre installation domotique reproductible avec Docker Compose.
- Sécurisez toutes vos communications réseau avec TLS.
- Créez un système de vidéo-surveillance pour votre domicile.
- Automatisez votre domicile avec Python, Node-Red Home Assistant et AppDaemon.
- Accès à distance sécurisé à votre tableau de bord domotique.
- Commande vocale entièrement hors ligne dans votre propre langue.

Le logiciel et les errata du livre sont disponibles sur GitHub : https://bit.ly/2JPXDAg

Livre imprimé :
www.elektor.com/control-your-home-with-raspberry-pi

Livre numérique :
www.elektor.com/control-your-home-with-raspberry-pi-e-book

Résumé et pour aller plus loin

Les appareils sans fil dans la bande de fréquence de 433,92 MHz sont très répandus dans la communauté du bricolage. Ils sont bon marché et facile à trouver un peu partout. Les capteurs de température et d'humidité, utiles pour surveiller votre réfrigérateur, votre congélateur ou votre serre sont particulièrement intéressants.
Vous n'avez besoin que d'une clé DVB bon marché et d'une antenne pour lire ces capteurs. Le programme rtl_433 prend en charge plus de 150 protocoles d'appareils 433,92 MHz et permet de relayer les signaux reçus en messages vers votre courtier MQTT, afin que vos autres logiciels de domotique puissent en faire usage.
Il y a encore beaucoup de sujets intéressants à peine évoqués, comme le choix et le placement optimal de l'antenne pour une meilleure couverture. C'est aussi un exercice passionnant d'essayer d'ajouter la prise en charge d'un nouveau protocole par rtl_433. Le projet dispose d'une documentation détaillée sur la façon de capturer les signaux bruts et d'essayer de faire la rétro-ingénierie du protocole. Vous pouvez aussi essayer d'autres récepteurs, par exemple la famille des dispositifs RFXtrx, qui comprend même un émetteur-récepteur capable d'actionner les volets-roulants Somfy RTS. Il y a même une bibliothèque Python, pyRFXtrx (*https://github.com/Danielhiversen/pyRFXtrx*), pour communiquer avec vos appareils 433,92 MHz avec un émetteur-récepteur RFXtrx. Home Assistant (voir le chap. 10) utilise cette bibliothèque pour prendre en charge les appareils 433,92 MHz. Toutefois, l'émetteur-récepteur RFXtrx coûte bien plus cher qu'un RTL-SDR.
À l'autre extrémité du spectre de prix, si vous voulez expérimenter avec l'équipement le moins cher possible, l'émetteur STX882 et le récepteur SRX887 sont de bonnes bases.

200534-03

Votre avis, s'il vous plaît…
Pour toute question ou commentaire relatifs à cet article, contactez l'auteur koen@vervloesem.eu

Ont contribué à cet article
Auteur : Koen Vervloesem
Illustrations : Koen Vervloesem
Rédaction : Jan Buiting
Maquette : Giel Dols
Traduction : Denis Lafourcade

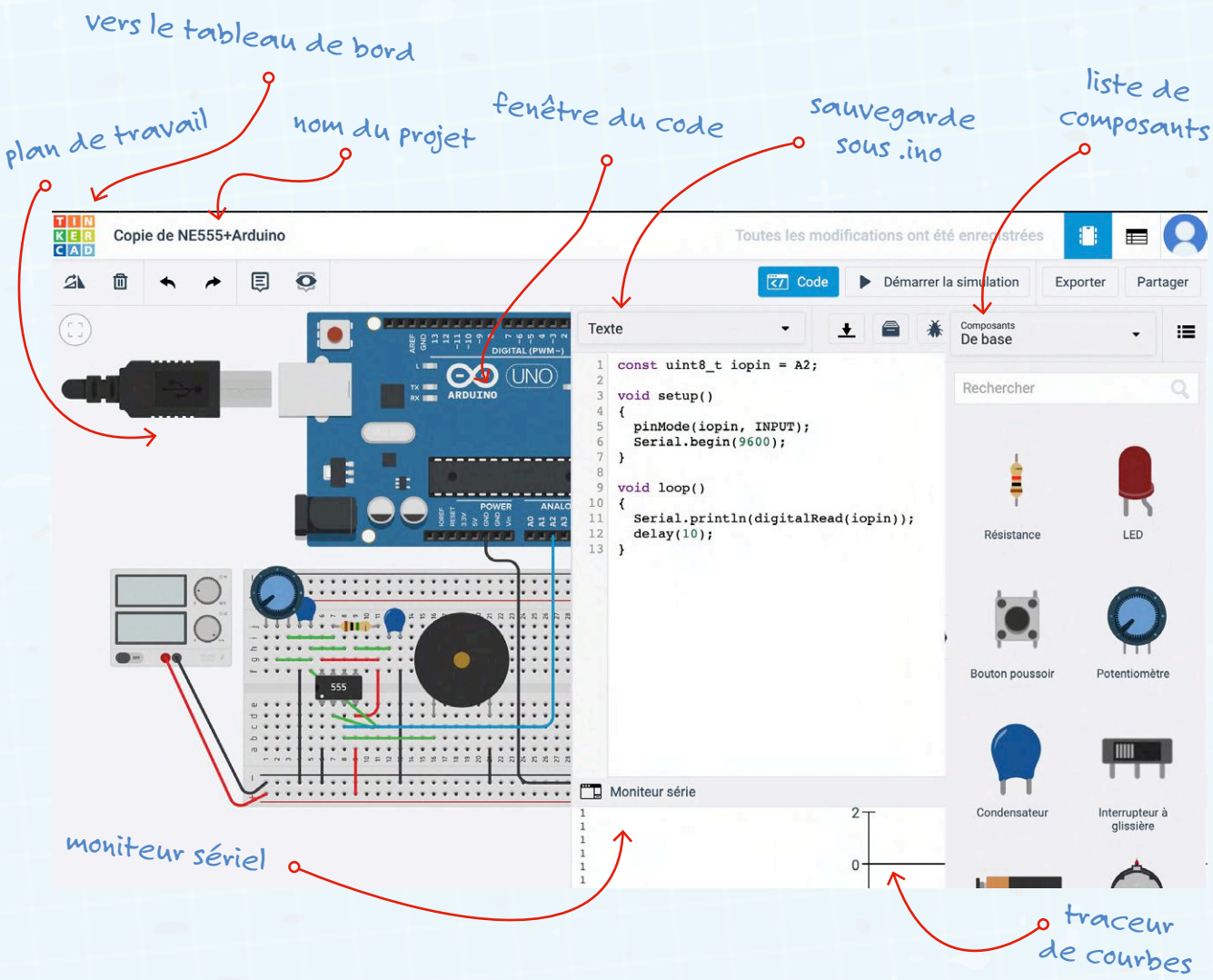
LIENS

[1] **Kit RTL-SDR avec antennes, supports et câble d'extension** : www.elektor.fr/rtl-sdr-software-defined-radio-with-dipole-antenna-kit

[2] **Utilisation d'une antenne dipôle avec RTL-SDR** : https://www.rtl-sdr.com/using-our-new-dipole-antenna-kit/

[3] La fin de la rédaction de ce livre a coïncidé avec la fin du développement actif de cette image Docker (avec plus de 100.000 téléchargements). Il n'y aura plus de nouvelles fonctions. Une autre image prendra-t-elle le relais? À suivre…

[4] Même si vous deviez en trouver une autre, vous n'aurez à faire que quelques modifications mineures pour l'utiliser : le fichier de configuration principal de rtl_433 reste le même.

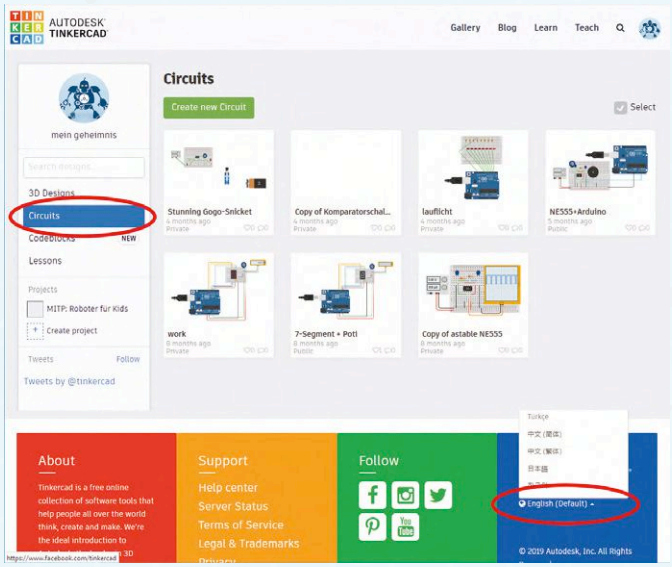


simule tes circuits en ligne

Florian Schäffer

Tu n'as (encore) aucun composant pour tester un circuit que tu as imaginé ou trouvé quelque part, et que tu voudrais maintenant tester. Si tu commençais par le simuler ? Essaye, c'est facile, tu peux même lui faire exécuter du code Arduino !

SPICE (*Simulation Program with Integrated Circuit Emphasis*) est probablement le plus connu des logiciels pour la simulation des circuits électriques analogiques et numériques. Cependant, son utilisation peut paraître lourde quand on débute, et le schéma doit se conformer à certaines exigences. Pour simuler un circuit comme sur une bonne vieille plaque d'essais, il existe heureusement une interface graphique en ligne (aucun logiciel à installer !). C'est *Tinkercad Circuits qui* propose même l'intégration des processeurs ATmega (Arduino Uno) et ATtiny, pour lesquels vous pouvez créer et exécuter des programmes en C/C++ et avec le langage graphique Scratch.



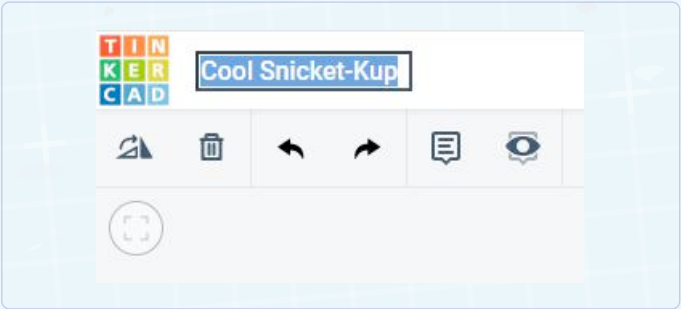
Dans le tableau de bord, choisis la section "Circuits". En bas, tu peux changer de langue si nécessaire (ou si tu as envie de rigoler).

Il faut te connecter sur le site www.tinkercad.com. Soit tu crées un nouveau compte, soit tu te connectes par un compte tiers tel que Google. Quand tu arrives comme nouvel utilisateur dans la section "*Circuits*" du tableau de bord, tu ne verras pas encore de modèles. En bas à droite, tu peux changer de langue si tu veux.

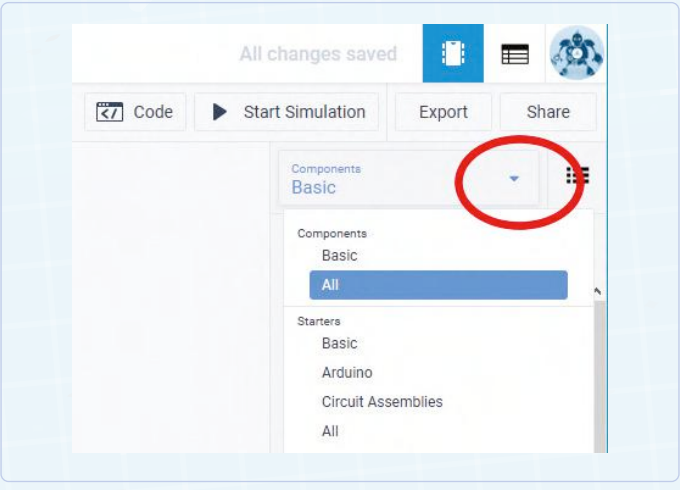
Le projet que je décris ici se trouve en ligne à l'adresse https://www.tinkercad.com/things/az1gBrXUclZ.

1. Clique sur *Créer un nouveau circuit*. La vue de la plaque d'expérimentation ressemble à celle de *Fritzing* (voir l'article à ce sujet dans le précédent numéro d'Elektor).

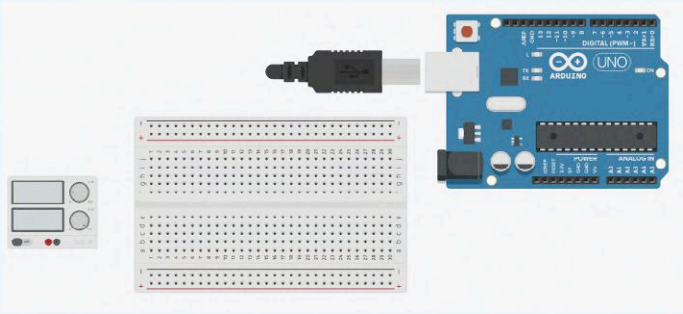
2. Ton dessin ou modèle sera stocké dans le nuage. Dans le coin en haut à gauche, à côté du logo *Tinkercad*, apparaît un nom du projet fantaisiste. Tu peux le garder ou le modifier, il suffit de cliquer sur le texte.



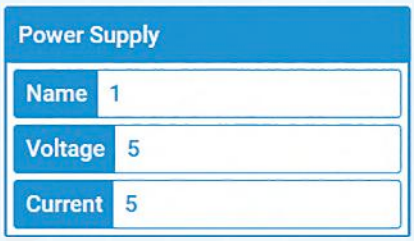
3. À droite de l'écran, tu peux choisir les composants ou des exemples de projets affichés. Dans la liste de sélection sous *Starters* (ou *Démarreurs* en VF – bravo le traducteur !), tu trouveras quelques exemples pour te familiariser. Pour créer un nouveau projet, tu sélectionneras "*All*" pour voir tous les *composants*.



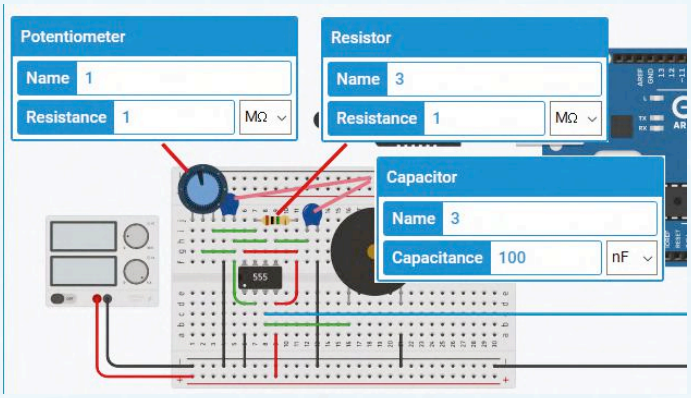
4. Dans la liste de composants, sous *Instruments* se trouve l'alim de labo. Au-dessus, sous *Microcontrollers* il y a l'Arduino Uno R3 et la petite carte enfichable. Clique sur le symbole correspondant et fais glisser le composant dans la zone de travail.



5. Pour beaucoup de composants, quand tu cliques sur le symbole dans l'espace de travail, tu peux définir des paramètres (nom, couleur, type, dimensions, etc.). Une boîte de dialogue propose les options de réglage. Clique sur l'alimentation et saisis "5" pour la tension. Le programme est américain, il utilise le point décimal, pas de virgules.



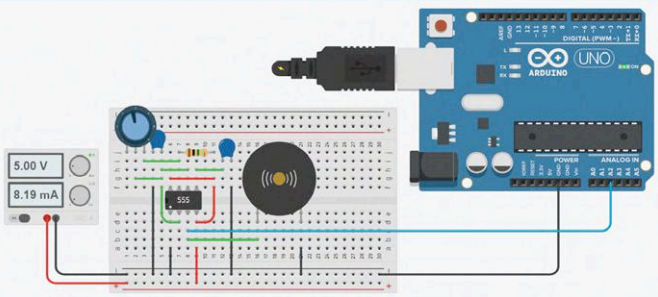
6. Place les composants et les câbles sur la plaque d'expérimentation, puis saisis dans la mesure du possible les numéros et les valeurs de tous les composants.



7. N'oublie pas la connexion de masse de la broche GND de l'Arduino (ligne noire à droite), ni la connexion bleue à la broche A2.

8. Une fois terminée la « construction » du circuit, tu peux tester et simuler le fonctionnement en cliquant sur *Start simulation* en haut à droite. Tu remarqueras la brève animation qui montre l'insertion de la prise USB dans l'Arduino : ce dernier est maintenant alimenté, la LED "ON" s'allume. Le cadran de l'alim de labo affiche la tension et le courant, mais pour l'instant, aucun programme n'est exécuté.

9. Tu peux faire tourner les boutons de l'alim en maintenant enfoncé le bouton de la souris. Le circuit tolère jusqu'à 18 V environ, mais pas l'Arduino, connecté par le fil bleu. Sur un vrai circuit, avec une tension aussi élevée, ton µC serait détruit. Si tu as un HP connecté à ton PC, tu entendras des impulsions sonores émises par l'élément piézo. Avec la souris, tu peux faire tourner le potentiomètre bleu en haut à gauche de la plaque d'expérimentation et changer le son.





10. Clique sur *Stop Simulation* puis sur *Code* pour ouvrir la fenêtre de saisie du code source. Il propose d'abord l'éditeur pour Scratch. Clique sur le triangle de sélection pour passer à l'éditeur de texte. Ne tiens pas compte de l'avertissement, continue.



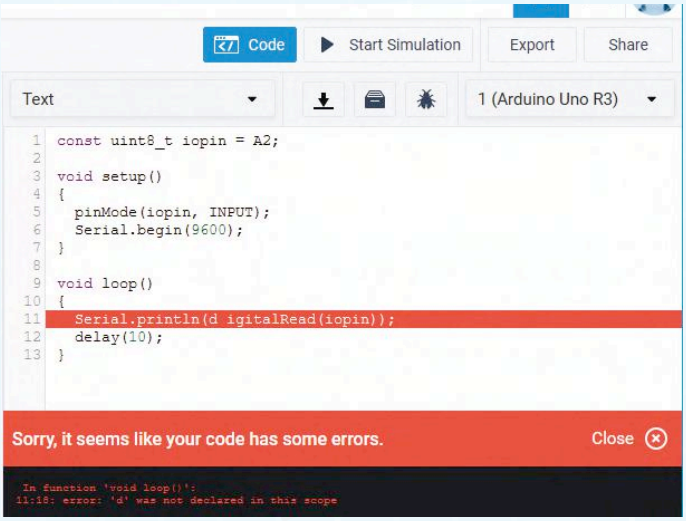
11. Dans la fenêtre de code, il y a déjà un petit programme. Efface-le et remplace-le par les commandes suivantes :

```
const uint8_t iopin = A2;

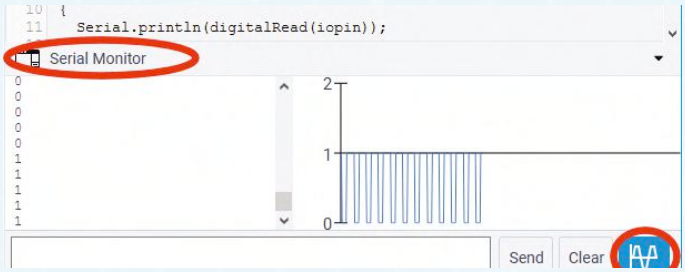
void setup()
{
  pinMode(iopin, INPUT);
  Serial.begin(9600);
}

void loop()
{
  Serial.println(digitalRead(iopin));
  delay(10);
}
```

12. Clique de nouveau sur *Démarrer la simulation* et le nouveau code sera exécuté. Si le programme comporte des erreurs, celles-ci seront signalées comme dans l'EDI Arduino. Avant de poursuivre, il faut les corriger.



13. Dans cet exemple, l'Arduino est censé fonctionner comme un oscilloscope rudimentaire et délivre sur sa broche d'entrée/sortie le niveau du signal qui pilote l'élément piézo. Tu verras la séquence de zéros et de uns si tu cliques sur *Serial monitor* au bas de la vue du code. En bas à droite, clique sur l'icône avec une trace d'oscilloscope et tu verras le rendu graphique.



14. Fais tourner le bouton du potentiomètre pour changer le son et tu verras changer la trace de l'oscilloscope.

Cet article est extrait d'un hors-série d'Elektor Elektor Special: Introduction to Electronics with Arduino
www.elektor.fr/elektor-special-introduction-to-electronics-with-arduino
disponible dans l'e-choppe d'Elektor.

200225-C-02

sur le vif

Entre ordre et chaos

Ilse Joostens (Belgique)

La table de travail de l'électronicien moyen contraste avec le bureau d'un comptable moyen. Je prête à ce dernier un penchant peut-être sadique, pour organiser le monde en le vidant de toute substance. Tandis que nous autres, adeptes de l'électronique aspirons à toujours davantage de substance. Notre poste de travail, de plus en plus encombré semble toujours attendre la visite d'une équipe de spécialistes des matières dangereuses. Comme le bureau de feu Bob Pease, inventeur entre autres de la carte d'expérimentation à barbe [1]. Une profane a du mal à comprendre la différence entre la photo d'un tel bureau et celle d'une zone sinistrée après le passage d'un cyclone.



Compteur de petit-pois, aussi appelé comptable…

Consolons-nous avec l'hirsute Albert von Einstein qui aurait dit : «Si un bureau désordonné est le reflet de la tête en désordre de son occupant, que dire de la tête de l'occupant d'un bureau vide ?»

Le chaos, ce n'est pas seulement une théorie

J'ai encore beaucoup à apprendre de Bob Pease, alias RAP [2], mais j'avoue la honte aux joues que je suis une incorrigible flemmarde. J'entasse obstinément, au point de pousser mes parents, beaucoup d'anciens employeurs et même mon partenaire jusqu'aux cimes du désespoir.

Je pratique les CMS depuis assez longtemps pour savoir, comme tout le monde, que si l'on veut éviter la pagaille il vaut mieux les laisser dans leur emballage d'origine. Évidemment, j'ai laissé les sachets de composants s'amonceler chez moi, puis j'ai peu à peu empilé des boîtes de stockage avec des CMS aussi divers qu'impossibles à distinguer.

Chaque fois que j'avais à produire une petite série de cartes, je passais plus de temps à trier les composants qu'à les implanter. Combien de fois n'ai-je pas commandé des composants que j'avais en réserve, impossibles à retrouver rapidement ! Pire, d'autre fois je n'ai délibérément pas commandé tel composant dont j'avais un besoin urgent, persuadée (à tort) que j'en avais encore assez. Je les avais sans doute, mais dans quel sachet, quelle étagère, quelle boîte ?…

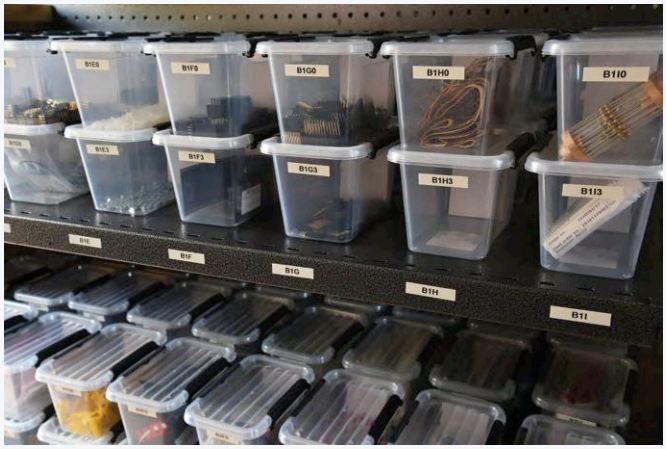
Certains trient leurs composants par type et par valeur dans des armoires à tiroirs, s'efforçant de rassembler des séries aussi complètes que possible, p. ex. toutes les résistances E24. Pas mal pour les amateurs ou même les concepteurs et réparateurs d'appareils, mais pas pour ce qui concerne la production. En fait, j'achète volontiers des composants en vrac, notamment les composants traversants, en fonction de besoins précis pour la fabrication de mes produits. À quoi bon stocker des milliers de résistances de toutes les valeurs de la série E24 puisque vous n'utiliserez sans doute jamais la majorité de ces valeurs? Pas question de les trier minutieusement pour les mettre dans de petits tiroirs.

Espionnage industriel

Un peu d'espionnage industriel lors d'une journée portes ouvertes chez l'un de mes fournisseurs m'a inspiré un meilleur système. J'utilise maintenant des étagères avec des bacs munis chacun de son code indiquant sa position : p. ex. «A5K0». C'est le casier A, de l'étagère 5, à la section K et la position 0. J'utilise un tableur pour m'y retrouver, **les composants n'ont pas besoin d'être triés**. Ainsi, un bac rempli de µC peut-il se retrouver entre deux bacs de vis. Je réarrange facilement les positions vides quand elles sont réutilisées pour d'autres composants.

Hélas le tableur n'est pas idéal, mais je n'ai pas encore trouvé mieux… En attendant, je passe plus de temps à comptabiliser







Ce système fonctionne bien en pratique.

les composants et à manipuler les boîtes qu'à manipuler mes cartes. Bien entendu, je me trompe de temps en temps, et ne m'en rends compte que plus tard, quand la production d'une carte s'enraye pour cause de rupture de stock inopinée. Je me demande parfois si pour moi mon chaos antérieur n'était pas préférable à cette illusion d'ordre. Une chose est sûre, le tableur me facilite l'inventaire de fin d'année pour le comptable.

«Trouver les composants» pour débutants

On a beau parler de la dématérialisation et de la virtualisation de tout, moi pour faire de l'électronique, il me faut des composants et, de préférence, beaucoup de composants différents et en quantité. Ça fait une éternité que je n'ai pas mis les pieds dans un magasin d'électronique ayant pignon sur rue. Je commande tout en ligne chez les fameux vendeurs en gros. Gardez vos distances avec l'oncle Ali pour les composants critiques et les semi-conducteurs. Pour des raisons que j'ignore, l'offre de LED d'éclairage chez les distributeurs connus est tristounette, et je me rabats parfois sur eBay. En temps normal, j'ai toujours quatre ou cinq sites de fournisseurs ouverts sur mon navigateur et je passe d'un onglet à l'autre pour obtenir le composant voulu dans la quantité que je veux au meilleur prix, c'est tout. Rares sont les fournisseurs qui ont tout en stock au meilleur prix. Mes commandes partent chez au moins trois fournisseurs, ce qui se traduit quelques jours plus tard par un défilé de livreurs sur le pas de ma porte. Le gamin de mes voisins croit que c'est Noël ici toutes les semaines. Moi je me demande ce que je vais bien pouvoir faire de ces quelques milliers de transistors MOSFET. Les prix échelonnés sont attrayants, ils vous poussent à acheter plus que prévu. C'est tentant, n'est-ce pas ? Là aussi, il faut garder l'œil ouvert et le bon : j'ai constaté à plusieurs reprises que le prix de tel circuit intégré par 100 pièces était le même que par cinquante pièces. Des composants gratuits donc, oui, ça existe !

200556-02

Votre avis, s'il vous plaît ?
Posez vos questions ou commentez cet article par courriel à l'adresse redaction@elektor.fr

Ont contribué à cet article

Texte et illustrations : **Ilse Joostens (Belgique)**

Rédaction : **Eric Bogers**

Maquette : **Giel Dols**

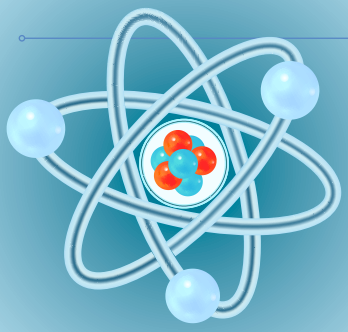
Traduction : **Jean-Pierre Gerné**

LIENS

[1] **Bob Pease** : https://www.electronicdesign.com/technologies/analog/article/21805320/whats-all-this-messy-office-stuff-anyhow

[2] **«Un coup ça marche, un coup ça marche pas» titre français du livre de Robert Pease *Troubleshooting analog circuits*, publié par Elektor** : https://www.amazon.fr/coup-%C3%A7a-marche-pas/dp/2866610903

[3] **Pour compléter votre bibliothèque technique** : http://sylvain.larribe.free.fr/livres/PDF/Marche_MarchePas/Ca-marche_Ca-marche_pas_S_01_I_ocr.pdf



démarer en électronique… (5)

…est moins difficile qu'on ne l'imagine !

Eric Bogers

Comme promis, voici du concret, du tangible.
Des résistances pour être précis.
Il y a résistances et résistances, certaines plus résistantes, et d'autres moins.
C'est simple, mais ça peut se compliquer…

Résistances passives

Par définition, la résistance est passive. Les résistance sont incapables d'amplifier un signal. Les autres composants passifs sont les condensateurs et les bobines ou selfs ; les composants actifs sont les transistors, les triacs et les diodes. Eh oui, les diodes figurent parmi les composants actifs, bien qu'elles n'amplifient pas, mais parce qu'elles appartiennent à la famille des semi-conducteurs avec lesquels on les range.

Voyons à quoi ressemblent les résistances. Dans la **figure 1,** nous avons les variétés de résistances les plus connues. En bas, une résistance à film métallique et au-dessus, une résistance à couche de carbone, toutes deux d'une puissance max. de 0,25 W. C'est la puissance maximale que cette résistance est capable de dissiper sous forme de chaleur sans détérioration de sa couche résistive. La puissance d'une résistance est un paramètre à surveiller !

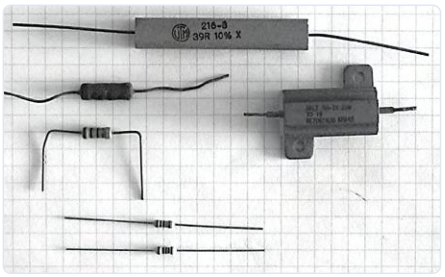


Figure 1. Quelques résistances.

Les résistances présentent non seulement une limite de dissipation de puissance à ne pas dépasser, mais aussi à une limite en tension ; si le seuil de tension de la résistance est dépassé, la surtension peut détruire la résistance (et le circuit dans lequel elle est montée). Pour les résistances courantes de nos projets de loisirs, la tension maximale est généralement de 250 V ; en pratique, nous n'aurons que rarement à nous soucier de tout cela.

Retour à la figure 1 : au milieu, à gauche, deux résistances avec une limite de puissance plus élevée (0,5 W et 1 W), et tout en haut une résistance de 39 Ω cimentée qui supporte jusqu'à 11 W. La résistance de puissance au centre à droite, vissée sur un radiateur, peut dissiper jusqu'à 25 W.

Les résistances ordinaires sont si petites (bas de la figure 1) qu'il est impossible d'y imprimer leur valeur sous une forme lisible. Leur valeur est codée par des anneaux de couleur. Il existe des résistances encore bien plus petites, dépourvues de pattes à souder, appelées CMS, pour composants montés en surface. Comme cette série d'articles est destinée aux débutants en électronique et que les CMS ne conviennent vraiment pas dans le cadre de l'initiation à l'électronique, nous ne nous étendrons pas sur ce sujet. Il y a déjà assez à dire des composants familiers et faciles à manipuler.

Des bagues de couleur

Il existe des résistances à quatre et à cinq bagues de couleur. Les deux ou trois premières bagues indiquent la valeur de la résistance, qui doit encore être multipliée par un facteur (décimal) indiqué par l'avant-dernière bague. Enfin, la dernière bague indique la tolérance. Tolérance ? Oui, les résistances sont imprévisibles : prenons un lot de 1000 résistances et mesurons-en la valeur avec un ohmmètre précis. Nous constaterons qu'il n'y en a pas deux de la même valeur. L'un aura 1001,3 Ω, l'autre 998,6 Ω et ainsi de suite. C'est la faute à d'inévitables petits défauts de fabrication. La tolérance indique entre quelles valeurs extrêmes se situe la valeur réelle d'une résistance. La valeur réelle d'une résistance de 1000 Ω avec une tolérance de 5 % (donc une résistance à couche de carbone normale), se situera entre 950 Ω et 1050 Ω ; si la tolérance est de 1 %, c'est une résistance à couche métallique, la valeur réelle se situe entre 990 Ω et 1010 Ω. Il arrive, mais rarement que l'on tombe sur des valeurs aberrantes, en dehors des limites de tolérance.

Pour la plupart des applications, des résistances à couche de carbone de 5 % conviennent ; nous n'utiliserons les résistances à couche métallique (plus coûteuses) que si la précision est critique, dans les filtres par exemple et/ou dans un circuit d'amplificateur audio de haut de gamme dont le bruit doit être aussi réduit que possible.

Le **tableau 1** indique le code de couleur des résistances. Une résistance à film métallique de 1% de 1 kΩ porte le code brun, noir, noir, brun, brun. Malheureusement, dans ce cas, faute de savoir dans quel sens le code doit être lu, il pourrait s'agir d'une résistance de

Tableau 1. Code de couleur des résistances.

					tolérance de la valeur
4 bagues	**1ère bague**	**2e bague**	**–**	**3e bague**	**4e bague**
5 bagues	**1ère bague**	**2e bague**	**3e bague**	**4e bague**	**5e bague**
noir	–	0	0	1	
marron	1	1	1	10	±1%
rouge	2	2	2	100	±2%
orange	3	3	3	1 k	
jaune	4	4	4	10 k	
vert	5	5	5	100 k	±0,5%
bleu	6	6	6	1 M	
violet	7	7	7	10 M	
gris	8	8	8	100 M	
blanc	9	9	9	1 G	
or	–	–	–	0,1	±5%
argent	–	–	–	0,01	±10%

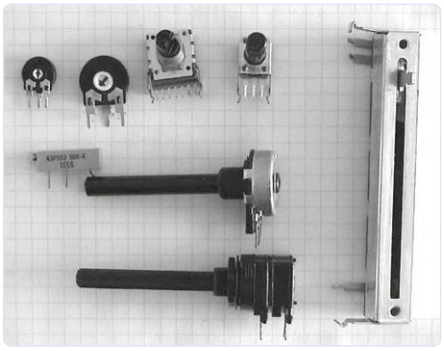


Figure 2 : Potentiomètres et résistances ajustables.

110 Ω... C'est un cas assez exceptionnel. En cas de doute, on vérifiera la valeur exacte avec un ohmmètre.

Un autre inconvénient des bagues de couleur est la confusion possible entre rouge et orange. Là aussi, en cas de doute, c'est l'ohmmètre qui tranche.

Les résistances sont produites en séries de valeurs standardisées, choisies de telle sorte que la tolérance positive maximale d'une valeur corresponde à la tolérance négative maximale de la voisine supérieure. Les plus courantes sont les séries dites E (voir **tableau 2**) produites à leur tour en différentes décades (par exemple 2,2 Ω, 22 Ω, 220 Ω, 2,2 kΩ, 22 kΩ, 220 kΩ et 2,2 MΩ). Il existe aussi des séries R, mais vos chances de les rencontrer un jour sont faibles. On ne peut pas tout savoir.

Si vous avez un jour besoin d'une valeur précise, mais à ce point tordue qu'elle n'existe dans aucune des séries E), elle peut être obtenue par un montage en parallèle et/ou en série de résistances E24 choisies de manière appropriée. Ou, bien sûr, vous pouvez utiliser une résistance variable...

Résistances variables

Le propre des potentiomètres et des résistances ajustables est de présenter une résistance que l'on peut faire varier. Les potentiomètres sont équipés d'un axe tournant sur lui-même que l'on peut munir d'un bouton. Quand il dépasse du boîtier d'un appareil, il peut être actionné par l'utilisateur. L'exemple le plus connu est bien sûr le contrôle du volume d'un amplificateur classique (l'organe de réglage de volume des appareils numériques actuels est un générateur d'impulsions (codeur)). Les résistances ajustables sont implantées sur le circuit imprimé et ne sont généralement pas accessibles de l'extérieur. On les règle une fois pour toutes avec un mini-tournevis lors de l'étalonnage du circuit. La **figure 2** montre quelques exemples de potentiomètres : à droite, le curseur rectiligne utilisé sur les consoles de mixage, au milieu, deux potentiomètres ordinaires, en haut un spécimen monophonique et en bas un stéréophonique. Ce dernier se compose en fait de deux potentiomètres actionnés par un axe commun.

Dans un potentiomètre rotatif classique, un contact de traînée se déplace sur une piste de carbone. La qualité de ce contact électromécanique se dégrade à l'usage, et finit par produire des parasites gênants, surtout avec les amplificateurs. Quand un tel potentiomètre n'est pas hermétiquement clos, un aérosol spécial pour contacts électriques peut résoudre le problème, pour un temps au moins. Parfois on obtient une certaine amélioration (non moins temporaire) en faisant faire à l'axe de nombreuses allées et venues rapides. Le mieux est de changer de potentiomètre.

Les potentiomètres dits *cermet* (plus chers) sont moins sensibles à ces conséquences du vieillissement ; sur la fig. 2, vous en voyez deux exemplaires (mono à droite, stéréo à gauche). Enfin, dans le coin supérieur gauche, il y a deux résistances ajustables en version fermée, réputées moins sensibles à la poussière.

Tableau 2.
Valeurs de résistance normalisées

E3	E6	E12	E24
1	1	1	1
			1,1
		1,2	1,2
			1,3
	1,5	1,5	1,5
			1,6
		1,8	1,8
			2,0
2,2	2,2	2,2	2,2
			2,4
		2,7	2,7
			3,0
	3,3	3,3	3,3
			3,6
		3,9	3,9
			4,3
4,7	4,7	4,7	4,7
			5,1
		5,6	5,6
			6,2
	6,8	6,8	6,8
			7,5
		8,2	8,2
			9,1

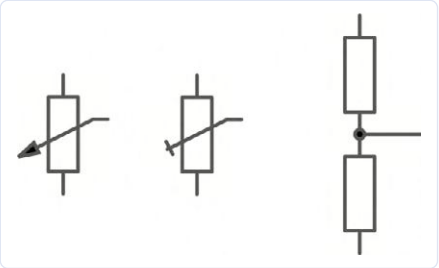


Figure 3 : Symboles de potentiomètres.

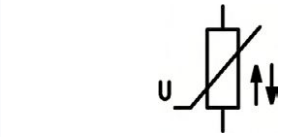


Figure 4 : Résistance commandée en tension (VDR).

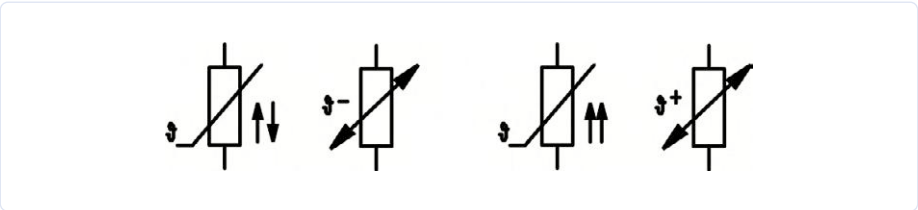


Figure 5 : Thermistances NTC et PTC.

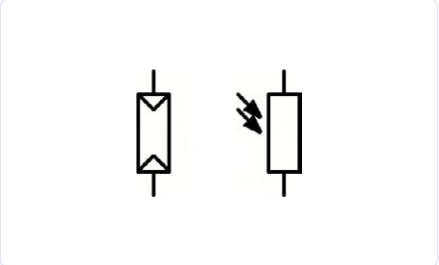


Figure 6 : Photorésistance (LDR).

Linéaire ou logarithmique ?

Dans les listes de composants, l'abréviation *log* ou *lin* indique la caractéristique de la courbe des potentiomètres : logarithmique ou linéaire, selon la progression de la valeur de la résistance d'un bout à l'autre de la piste. Les potentiomètres à courbe logarithmique sont utilisés principalement pour la commande de volume des amplificateurs, car notre sensibilité auditive présente elle-même une caractéristique (plus ou moins) logarithmique.

La **figure 3** montre des symboles de potentiomètres : à gauche un potentiomètre, au milieu une résistance ajustable et à droite le schéma équivalent. La valeur des deux résistances partielles varie selon la position du curseur (c'est-à-dire la connexion centrale dans le symbole du schéma) sur la piste résistive.

Comme celle des résistances normales, la puissance des potentiomètres est limitée.

Résistances spéciales

Il existe des résistances dont la valeur varie en fonction de phénomènes physiques. Voici les plus courantes.

La **figure 4** montre le symbole d'une *résistance commandée en tension* (VDR). Dès qu'un certain seuil de tension est dépassé, une VDR devient très faiblement ohmique ; ce composant est utile pour protéger d'autres composants contre les surtensions. Les VDR sont souvent utilisées pour protéger les transformateurs de réseau contre les surtensions (qui peuvent se produire quand un conducteur neutre est mal connecté). Dans ce cas, la VDR est traversé par un courant d'une telle intensité que le fusible secteur saute : le danger disparaît. Généralement la VDR doit être remplacée par la suite.

Sur la **figure 5,** on trouve des thermistances, ou «résistance sensible à la température». À gauche NTC, à coefficient de température négatif, à droite PTC, à coefficient de température positif. Sur une résistance NTC, la valeur de la résistance diminue à mesure que la résistance se réchauffe ; sur une résistance PTC, c'est exactement l'inverse. Les résistances NTC sont parfois utilisées pour limiter l'intensité des courants d'appel : les amplificateurs de puissance classiques contiennent généralement de lourds transformateurs secteur et de volumineux condensateurs de lissage qui ne sont pas encore chargés au moment de la mise sous tension. L'intensité du courant immédiatement après la mise en marche est donc si élevée que dans le pire des cas cela peut faire sauter le fusible du secteur. Une NTC (froide) peut résoudre ce problème en limitant l'intensité du courant dans un premier temps ; à mesure que la NTC est réchauffée par le courant qui la traverse, l'intensité du courant augmente moins brutalement avant d'atteindre sa valeur nominale sans avoir fait sauter de fusible. Habituellement, après quelques secondes, ces résistances de limitation de courant sont pontées par un relais (électromécanique) afin d'éviter toute perte de puissance dans la résistance NTC.

Enfin, la **figure 6** montre la photorésistance dont la résistance varie selon l'éclairement (LDR). Plus la LDR est éclairée, plus sa résistance diminue. Naguère, ces résistances étaient souvent présentes dans les photocapteurs, mais leur usage était restreint par la lenteur de leur réaction aux changements de luminosité. De nos jours, on préfère les photodiodes ou les phototransistors.

La prochaine fois, nous parlerons de condensateurs.

200551-02

Votre avis, s'il vous plaît
Nous vous invitons à donner votre avis sur cet article en écrivant à l'adresse redaction@elektor.fr

Ont contribué à cet article
Auteur : **Michael Ebner**
Rédaction : **Eric Bogers**
Maquette : **Giel Dols**
Traduction : **Louis Pinck**

LIVRES

- **Basic Electronics for Beginners**
  www.elektor.fr/basic-electronics-for-beginners-e-book
- **L'électronique pour les débutants**
  https://bit.ly/339BBAv

ohm suite ohm

Ni CPU ni ALU, juste un opérateur logique NOR à deux transistors

Dennis Kuschel (Allemagne)

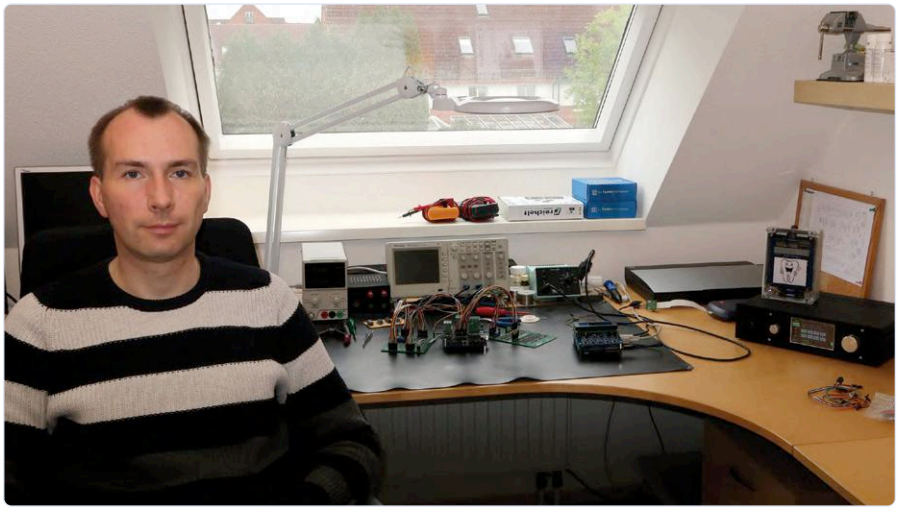


Figure 1. Dennis Kuschel dans son laboratoire. Une installation apparemment spartiate mais elle foisonne d'idées découaphantes.

Rien de tel qu'un microcontrôleur pour désintégrer massivement la quincaillerie électronique tout en multipliant les fonctions dans un volume toujours plus restreint. Un seul µC moderne remplace un tiroir entier de composants dits discrets, mais bien encombrants. Chaque revers a sa médaille : ces millepattes-là il faut les programmer, dans un langage de programmation plus ou moins évolué. Et c'est ainsi qu'on perd le contact avec ce qui se passe matériellement dans un tel contrôleur.
Retour aux sources avec quelqu'un qui connait son sujet !

Un macrocontrôleur

« La photo (**fig. 1**) de mon lieu de travail à la maison montre que mon équipement de mesure se limite à un oscilloscope et deux multimètres numériques. Je n'ai besoin de rien d'autre. Sur ma table, un exemplaire de l'ordinateur MyNOR (jeu de mots sur *minor* = (plus) petit) dont 10 circuits intégrés ont été remplacés par des transistors discrets (on voit surtout les fils de connexion colorés). »
« Pour Noël 1989, j'ai reçu mon premier ordinateur à moi, un Commodore 64. Les jeux ne m'ont pas satisfait longtemps, j'ai voulu écrire mes propres programmes et savoir comment cela fonctionne. Quatre ans plus tard (j'avais 17 ans), j'ai construit un nouvel ordinateur avec les pièces d'un C64 mis au rebut, et je l'ai programmé en assembleur. Pendant mes études, dans les années 90, j'ai eu l'idée de construire une unité centrale avec des opérateurs logiques simples et des circuits intégrés. Quelques années plus tard, il en est sorti MyCPU [1], un ordinateur construit à partir de plusieurs dizaines de circuits intégrés logiques de la série 74-CMOS - fonctionnant avec une version C64-Basic. »
« Partout où je l'ai présenté, ce MyCPU a suscité l'intérêt, mais aussi la consternation car, avec sa périphérie, il m'avait coûté ~1000 €. Vingt ans après, j'ai entrepris de construire un ordinateur entièrement discret, mais aussi simple et bon marché que possible, avec deux conditions restrictives : pas d'ALU (*Arithmetic Logic Unit*), car la puce 74LC181 de l'ALU, jusqu'alors largement utilisée, n'est plus disponible et que les alternatives sont trop compliquées. Et d'autre part un seul circuit intégré de mémoire programmable (EPROM). »

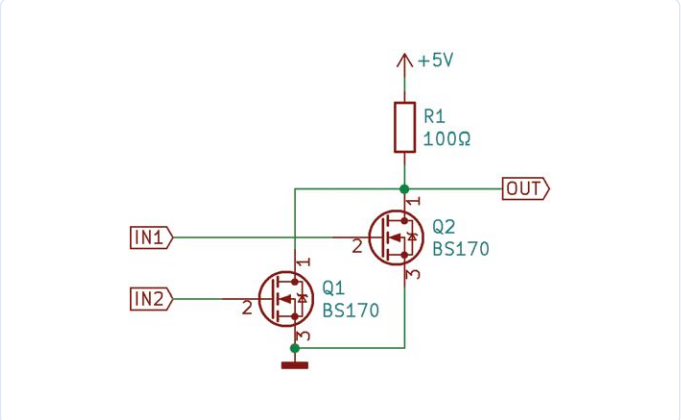


Figure 2. Au cœur de l'ordinateur MyNOR un unique opérateur NOR discret.

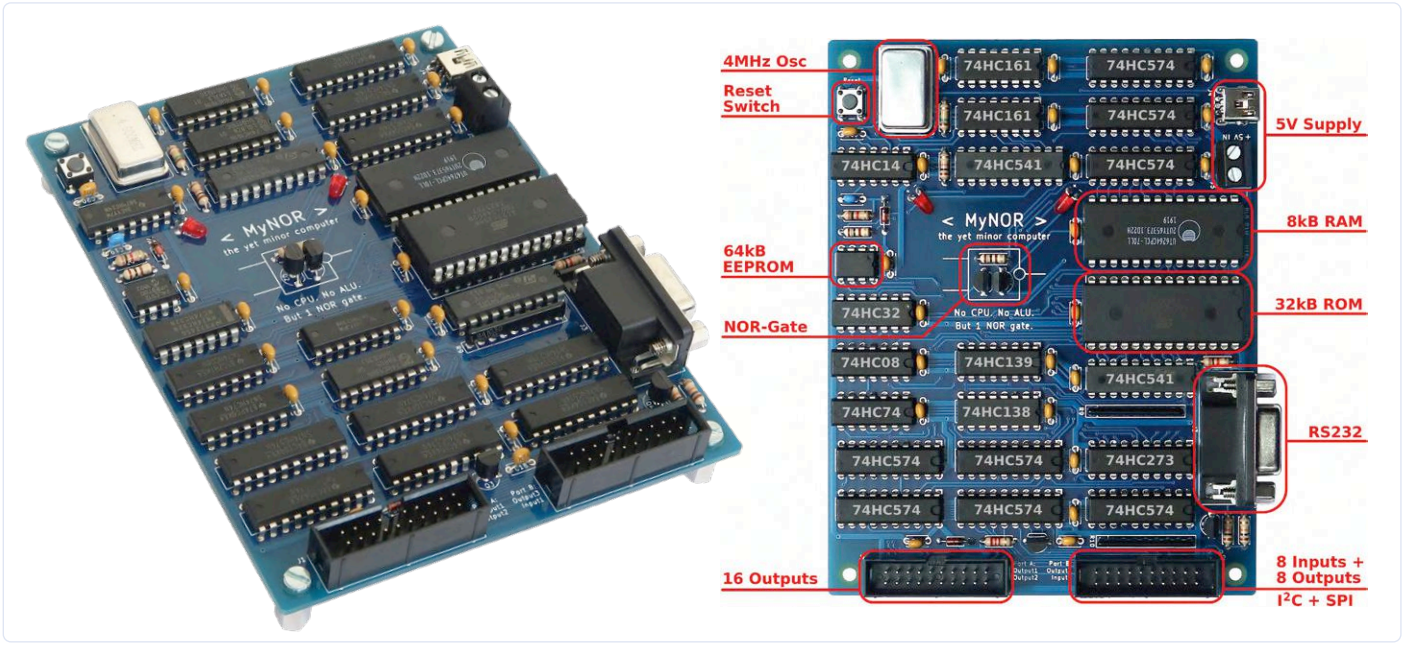


Figure 3. La version finale de MyNOR. La carte de droite montre que même la périphérie nécessaire est disponible.

« Mon nouvel ordinateur n'utilise qu'un seul opérateur NON-OU (NOR) pour les calculs - choisi parce qu'il est très facile à réaliser avec deux transistors MOSFET et une résistance (**fig. 2**). Toutes les opérations logiques et arithmétiques (AND, OR, EXOR, addition et soustraction) sont compilées par un logiciel à partir de nombreuses opérations NOR distinctes. »

« Il est donc logique que je l'aie baptisé MyNOR [2]. Le nombre total de composants est si petit qu'il tient sur une carte de 10 x 13 cm. La **figure 3** montre la carte MyNOR telle qu'on la voit (à gauche) et avec des légendes (à droite). Elle contient également la périphérie nécessaire : une EEPROM de 64 Ko comme «mémoire de masse», tandis que 8 entrées numériques et 24 sorties numériques assurent la communication avec le monde extérieur. Ces 32 E/S permettent également de réaliser les interfaces habituelles (basées sur des protocoles logiciels) : RS232, I²C et SPI. »



Figure 4. À gauche la calculette de poche et à droite le MyNOR avec le(s) clavier(s) et l'écran.

Virgule flottante avec une seule porte NOR

« C'est incroyable ce qu'un tel petit ordinateur à carte unique sait faire ! J'ai par exemple intégré dans le système d'exploitation une simple calculatrice utilisable via l'interface RS232. La fonction principale de ce calculateur est en fait de prouver que les calculs en virgule flottante peuvent être effectués avec un seul opérateur logique NOR ! En outre, le système de commande contient un programme de terminal pour saisir simplement des instructions en assembleur. On peut donc programmer en assembleur directement dans MyNOR et les enregistrer dans l'EEPROM (un peu comme sur le C64). Bien sûr, il est également possible de télécharger un programme complet. Sous la forme d'un fichier texte qui contient le programme binaire, dans un format spécial. Il faut un peu de patience, le débit de la connexion est de 2400 bauds. »

« C'est quand on coupe le cordon ombilical avec le PC que notre ordinateur à carte unique devient vraiment passionnant. C'est pourquoi j'ai conçu deux cartes d'extension (**fig. 4**). La première ne comporte que 20 boutons, une LED, huit afficheurs à 7 segments avec leurs transistors de commande. Les afficheurs et les boutons sont multiplexés comme d'habitude. Tous les jours, au travail, j'utilise une «calculette de poche» construite de cette manière. »

« La deuxième carte d'extension fait de MyNOR un très petit ordinateur. En plus des poussoirs habituels, elle comporte un afficheur LCD 4x20, un haut-parleur, une horloge en temps réel à piles et un capteur de température. Une fois qu'il est flanqué de cette carte, les possibilités de MyNOR sont infinies : j'ai déjà programmé quelques jeux (dragueur de mines, Tetris), une minuterie de cuisine, une boîte à musique et un scanner de bus I²C. »

« Ma première version de MyNOR en grandeur nature était imparfaite, et pourtant cet exemplaire (y compris l'affichage à 7 segments) travaille encore deux fois par jour : j'ai converti MyNOR v1.0 en minuterie de brossage de dents pour mes enfants (**fig. 5**) ».



Figure 5. Minuterie de brossage de dents avec MyNOR v1.0 plus l'affichage à 7 segments v1.0. L'appareil sur lequel est posée la minuterie est un récepteur haut de gamme (FM et DAB+) avec égaliseur à 11 bandes intégré et commande tactile, construit par Dennis l'an dernier.

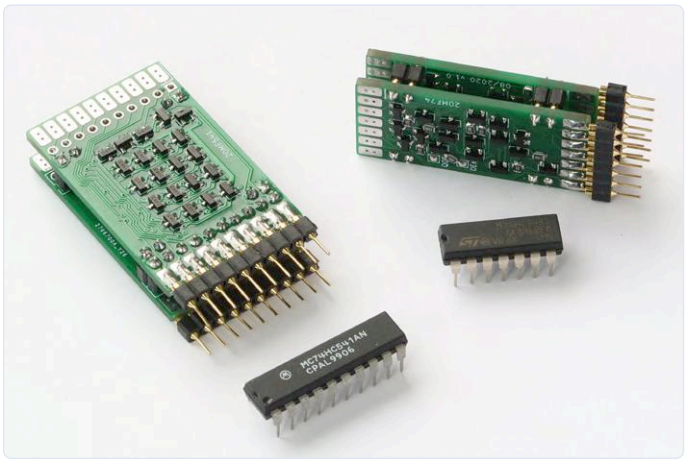


Figure 6. Deux circuits intégrés reconstitués au moyen de transistors. Les originaux sont là pour la comparaison (74HC541 et 74HC74).

« Je travaille actuellement sur une version encore plus discrète (c'est-à-dire encore moins intégrée) de l'ordinateur MyNOR ; c'est un grand défi que de vouloir remplacer 19 des seulement 22 circuits intégrés qu'il comporte par des circuits équivalents à transistors discrets. Mes premiers essais sont prometteurs. Sur la **fig. 6,** on retrouve deux des circuits intégrés reconstitués à l'aide de transistors discrets. »

« Au fait, je ne suis pas le seul à me passionner pour les unités centrales désintégrées à la maison. Nous sommes nombreux à partager notre passion sur le *Homebrew CPU Webring* ; si le sujet vous inspire, une visite du site [3] vaut la peine ».

200552-02

À SUIVRE ?

Si vous êtes un adepte des unités centrales faites maison et que vous pensez avoir de la matière pour prolonger cet article, veuillez nous faire part de votre expérience en écrivant à redaction@elektor.fr

Votre avis, s'il vous plaît

Commentaires ou questions par courriel à redaction@elektor.fr

Ont contribué à cet article

Auteur : **Dennis Kuschel (Allemagne)**
Rédaction : **Eric Bogers**
Maquette : **Giel Dols**
Traduction : **Émile Bodo**

À propos de l'auteur

Dennis Kuschel est entré en contact avec l'électronique à l'âge de six ans, dans la cour de l'école, en échangeant un sachet de bonbons contre une radio à transistors défectueuse. Peu de temps après, à la stupéfaction de ses parents, il a réussi à remettre cette radio en marche. Dennis travaille actuellement comme concepteur de matériel pour un fabricant allemand de caméras. Il y conçoit des circuits et des cartes à 10 couches Starrflex-HDI et programme des noyaux IP complexes pour les FPGA. En privé, il aime aller à contresens : «Keep it simple» est son credo - moins il y a d'électronique, moins il y a de risques de panne…

LIENS

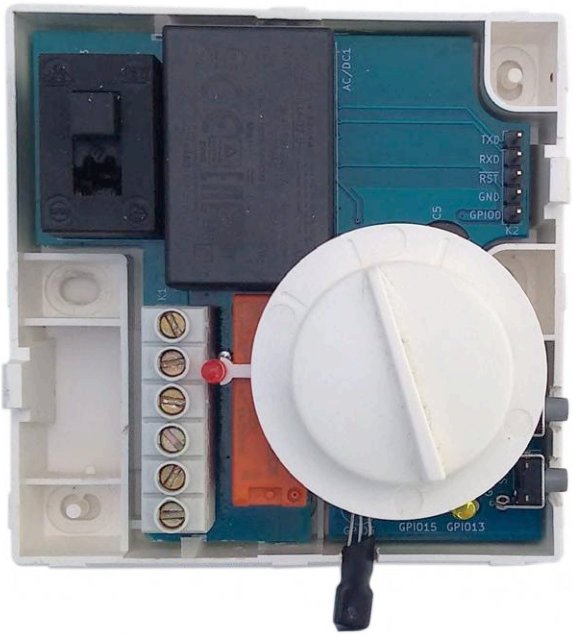
[1] **MyCPU** : www.mycpu.eu

[2] **MyNOR** : www.mynor.org

[3] **Cercle des unités centrales désintégrées** : www.homebrewcpuring.org

connectez votre thermostat avec ESPHome

Une tentative pour faire de la domotique comme il faut





Clemens Valens (Elektor)

La domotique bien faite est comme une main invisible. Elle vous pousse en douceur au sommet de la colline. Sans elle vous graviriez aussi cette colline, avec elle la vie est un peu plus confortable. Cet article porte sur la conception d'un thermostat pour un tel système domotique. Automatisé ou pas, son interface utilisateur classique vous permet de garder la maîtrise de la température de la pièce.

Il y a environ un an, je décidais de me lancer dans la domotique. Mon premier pas a été l'automatisation du thermostat de notre salon. Je l'ai fait en remplaçant le thermostat mural existant par le thermostat de bureau WiFi (projet Elektor 160269, publié dans l'édition de janvier/février 2018 [1][2]) reprogrammé avec un micrologiciel basé sur ESPHome. Le nouveau micrologiciel donnait accès à tous les organes du thermostat de bureau (c.-à-d. un relais, deux boutons-poussoirs et trois LED), ce qui permettait de les automatiser avec un gestionnaire domotique comme *Home Assistant*.

Dans cette configuration, le rôle du thermostat est joué par *Home Assistant* qui s'exécute sur un RPi et qui décide quand allumer ou éteindre le chauffage. Le thermostat de bureau lui-même est devenu un simple relais télécommandé avec quelques LED.

Sympa, mais...

Le système fonctionnait bien et nous a permis de passer confortablement l'hiver 2019-2020. Non sans quelques inconvénients :

- Il fallait un réseau WiFi ;
- Il fallait Home Assistant ;
- Il était moche.

Conçu pour l'avenir

Au moment de faire ces expériences, le premier problème n'en était pas vraiment un, car la plupart d'entre nous ont un réseau WiFi qui fonctionne. Lorsqu'il s'interrompt, il suffit en général de redémarrer le routeur ou son équivalent. Toutefois, on sait tous que les technologies vont et viennent, et nul ne sait si dans trente ans il y aura encore un réseau WiFi pas loin… mais nos maisons seront probablement toujours là. En d'autres termes, il nous faut une solution à l'épreuve du temps.

Compatible avec le passé

Le second point est en rapport avec le premier, car *Home Assistant* et RPi peuvent aussi disparaître un jour. Mais il y a plus. Je pense savoir comment me débrouiller avec *Home Assistant* sur un RPi, ce qui n'est pas le cas de la plupart des gens que je connais. Pour que d'autres personnes puissent utiliser mon thermostat automatisé, il doit donc être « compatible avec le passé ». Il doit ressembler à un classique thermostat mural et se comporter comme tel. L'automatisation ne doit pas être imposée, elle sera plutôt une option discrète. Elle est là pour ceux qui veulent l'utiliser, sans gêner qui n'en veut pas.

Question de goût

Le troisième problème est plutôt personnel. Mon thermostat de bureau automatisé a fini pendu au câble sortant d'un trou dans le mur où se trouvait l'ancien thermostat (**fig.1**). Quelques câbles secteur inutilisés qui sortaient du mur avaient été mis à l'abri des êtres humains stupides avec du chatterton. Et comme le thermostat de bureau a besoin d'une alimentation en 5 V avec un connecteur USB, il était relié à un chargeur de téléphone branché sur une prise multiple voisine. De sorte qu'on voyait plusieurs câbles allant vers l'appareil, là où le thermostat d'origine n'en avait pas. C'est un excellent moyen d'engager la conversation, mais la plupart des visiteurs ne trouvaient pas ça joli, même plutôt « moche », « bizarre » et « dangereux ».

Retour à la planche à dessin

Les défauts de la configuration initiale du thermostat de bureau m'ont convaincu de le reconcevoir pour résoudre certains, voire tous les inconvénients. Ceci conduisit aux spécifications suivantes :

- Réglage local de la température de consigne, c.-à-d. sur l'appareil ;
- La commande locale a priorité sur l'automatisation ;
- Alimentation par le secteur ;
- Connexions adaptées et boîtier d'aspect professionnel.

Le second item est relatif au logiciel, les trois autres concernent l'électronique du nouveau thermostat.
La spécification 1 implique une interface utilisateur intuitive pour régler la température de consigne souhaitée. Le thermostat de bureau a deux boutons qui permettent d'augmenter ou diminuer la température de consigne. En revanche, il n'y a pas d'écran pour l'afficher. Ajouter un écran avec le peu de broches d'E/S disponibles sur le module WiFi est compliqué. Utiliser un potentiomètre avec une échelle étalonnée comme le thermostat d'origine paraissait plus simple et possible grâce à l'entrée analogique du module WiFi.
La troisième spécification demande aussi réflexion. Il faut environ 1 W de puissance pour une connexion WiFi et pour alimenter le relais. On peut fabriquer une petite alimentation avec un transformateur, mais difficile de la faire suffisamment petite ; nous ne voulons pas d'une énorme boîte sur le mur. Le thermostat d'origine a une alimentation sans transformateur, ce qui est parfait pour des charges relativement constantes, mais je ne suis pas sûr que cela fonctionne bien avec une charge qui fait du WiFi. Mon intention était donc d'utiliser à la place un petit module convertisseur CA/CC.

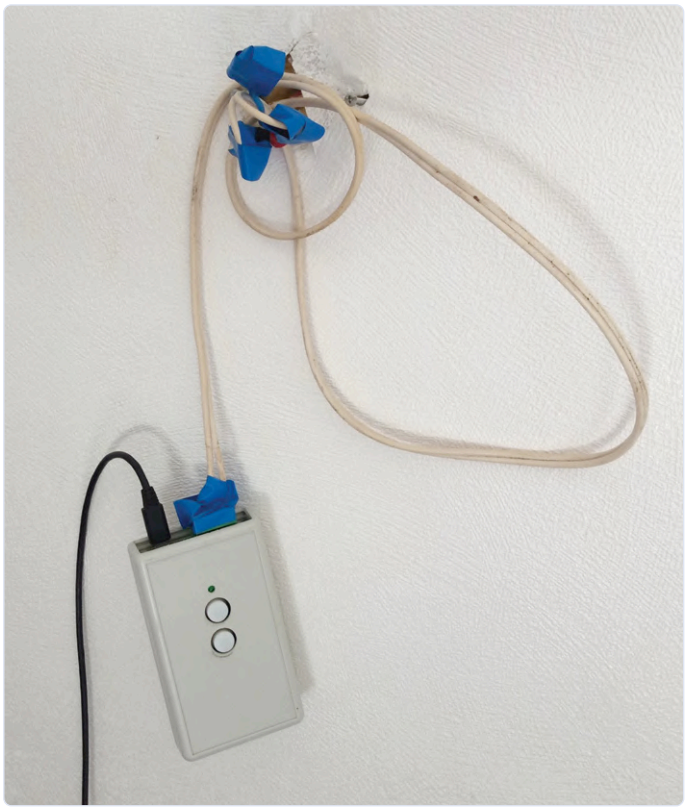


Figure 1. Certaines personnes ont trouvé « inesthétique » cette façon d'installer un thermostat intelligent dans un salon.

Je n'ai rien trouvé qui convienne en cherchant dans les catalogues de boîtiers. Fabriquer un joli boîtier sur mesure n'est pas mon fort, mais aujourd'hui avec les FabLabs, l'impression 3D et la découpe au laser accessibles même dans les endroits les plus reculés, cela pourrait ne pas être trop difficile. Je n'ai pourtant pas choisi cette voie. J'ai préféré essayer de réutiliser le boîtier du thermostat d'origine qui avait déjà tout ce que je voulais : un potentiomètre avec une échelle, une LED et un interrupteur. De plus, il avait des trous de montage bien placés et un bon moyen de connecter un circuit imprimé au secteur en traversant le fond pour cacher tous les câbles.
Mon effort de conception se réduisait maintenant à tasser le circuit du thermostat de bureau redessiné sur un circuit imprimé de sorte qu'il puisse contenir dans le boîtier existant avec le potentiomètre, la LED, l'interrupteur d'alimentation et la connexion au secteur, dans exactement les mêmes positions que dans le thermostat d'origine.
Reconcevoir le circuit du thermostat de bureau fut assez simple (**fig.2**). J'ai remplacé l'alimentation USB par un module 5 V CA/CC, et ajouté un potentiomètre avec une résistance de limitation de tension, car

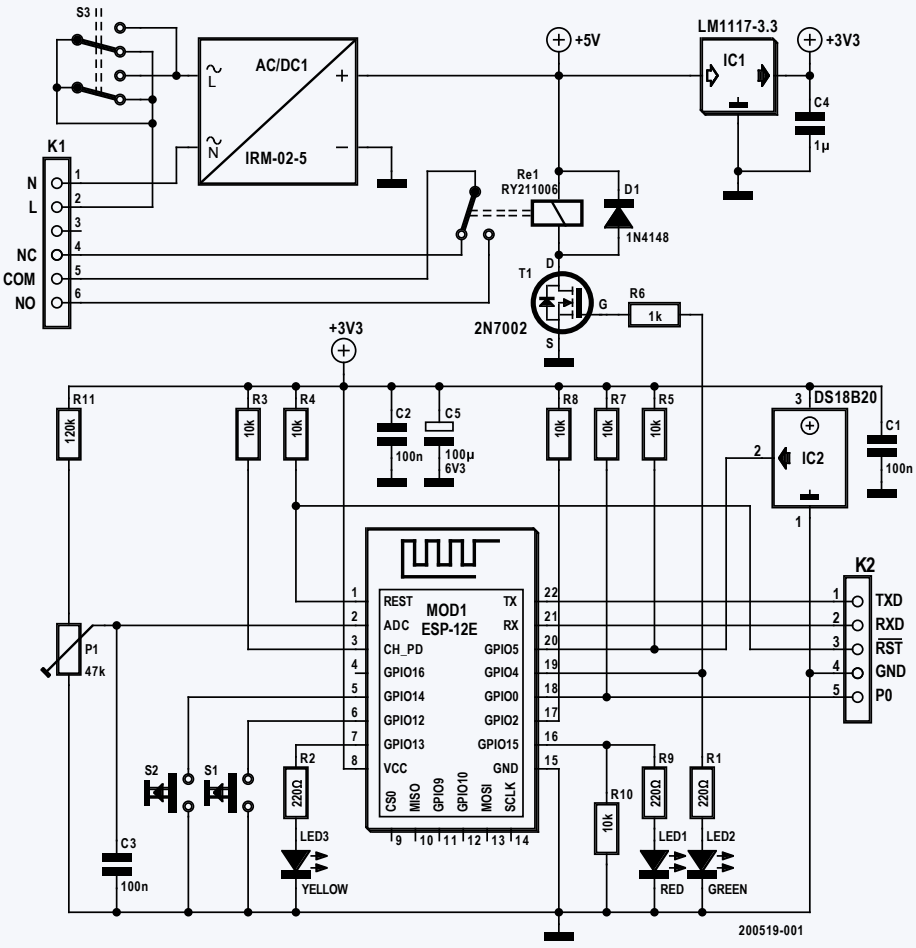


Figure 2. À la base, le nouveau thermostat discrètement connecté est le Thermostat de Bureau d'Elektor de 2018 avec une nouvelle alimentation. Un potentiomètre règle la température de consigne.

le module WiFi ne peut pas gérer de tensions supérieures à 1,1 V. J'ai gardé les deux boutons-poussoirs et les trois LED qui pourraient être utiles tôt ou tard.

Par une heureuse coïncidence, le module CA/CC s'est révélé juste assez petit pour tenir sous le support du bouton en plastique du potentiomètre. Le potentiomètre, l'interrupteur et le bornier du secteur ont été récupérés sur le vieux thermostat (**fig.3**). Restait à remplacer le relais 48 V par un modèle de 5 V. Quelle chance, une fois de plus, le relais du vieux thermostat était d'un type issu d'une famille au standard industriel toujours disponible et qui existe en 5 V !

Il a fallu beaucoup mesurer pour tout mettre sur un circuit imprimé qui s'adapte au boîtier d'origine, mais ça y est. Tous les CMS y compris le module WiFi sont sous la carte (**fig.4**), tandis que tous les composants traversants sont dessus. Une petite surface de carte supplémentaire a été gagnée en obstruant une paire de trous inutilisés du boîtier. Je confesse que je n'ai pu router toutes les pistes que par une approche inévitablement assouplie des normes d'isolation.

Logiciel

Je devais aussi repenser le micrologiciel ESPHome que j'avais compilé pour ma première configuration. Au lieu de juste donner accès à tous les capteurs et organes de commande du thermostat et laisser *Home Assistant* s'occuper du reste, je devais maintenant réaliser les automatismes dans ESPHome. Programmation et configuration sont faites dans le fichier YAML du projet ESPHome du thermostat (voir « la domotique c'est facile avec… » [3]).

Mesurer la température de la pièce

Déclarez d'abord le capteur de température qui mesure la température de la pièce. Comme le capteur utilisé est un DS18B20 (de Dallas à l'origine, maintenant de Maxim ou même Analog Device) connecté à la broche 5 du GPIO, et comme ESPHome a un composant spécifique pour Dallas, cela se traduit ainsi :

```
dallas:
  - pin: GPIO5

sensor:
 - platform: dallas
   address: 0x6D00000C24013928
   name: "Measured temperature"
   id: t_room
   filters:
    - offset: 0.0
```

La première ligne indique à ESPHome d'inclure son module de communication Dallas 1-wire et d'y connecter le GPIO5. Le capteur est ensuite issu de la plateforme *dallas*. L'*adresse* est optionnelle. Toutefois, si vous la spécifiez, elle doit être correcte ; vous pouvez l'obtenir à partir du journal d'ESPHome (n'utilisez pas la mienne, elle est unique). Ici il faut spécifier un *id* (*t_room*), car nous devrons nous référer au capteur depuis un autre endroit dans ce fichier YAML (cf. ci-dessous). L'ajout de filtres permet si nécessaire d'effectuer certaines corrections sur la température mesurée.

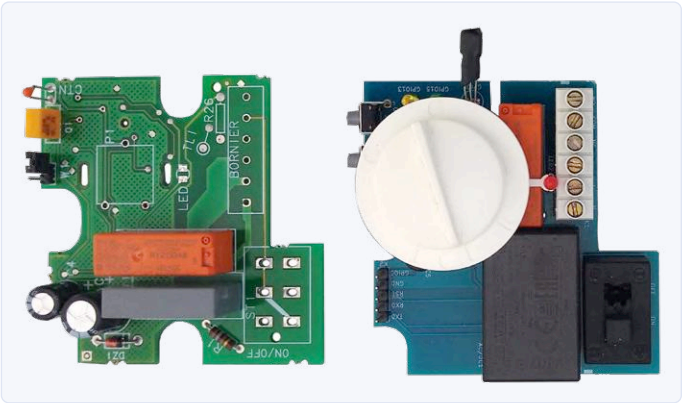


Figure 3. Le nouveau projet réutilise quelques morceaux de l'ancien thermostat, notamment le boîtier, ce qui explique la forme particulière du nouveau circuit imprimé. D'où résultent aussi quelques contraintes sur la position de certains composants.



Figure 5. *Après quelques ajustements mineurs du boîtier de l'ancien thermostat (un peu de plastique coupé), la nouvelle carte s'adapte parfaitement.*

Température de consigne

On règle la température de consigne avec le potentiomètre. Comme la valeur est une tension, ajoutez-le à la section ***sensor*** du fichier YAML avec plateforme ***adc*** comme type de capteur. Les « ***sensor filters*** » permettent de convertir la tension d'entrée en une température qui corresponde à l'échelle imprimée sur le boîtier. L'équation $T_{target} = 25 \cdot V_{in} + 6{,}75$ convient assez bien dans mon cas. Ceci se traduit par un filtre ***multiply*** avec une valeur de 25 et un filtre ***offset*** avec une valeur de 6,75. En spécifiant les unités en °C, un gestionnaire domotique comme *Home Assistant* traitera la donnée en tant que température.

```
- platform: adc
  name: "Target temperature"
  id: t_target
  icon: "mdi:temperature-celsius"
  pin: A0
  update_interval: 5s
  # Convert potentiometer scale to °C (min=6.75°C,
max=31.75°C)
  filters:
   - multiply: 25.0
   - offset: 6.75
```

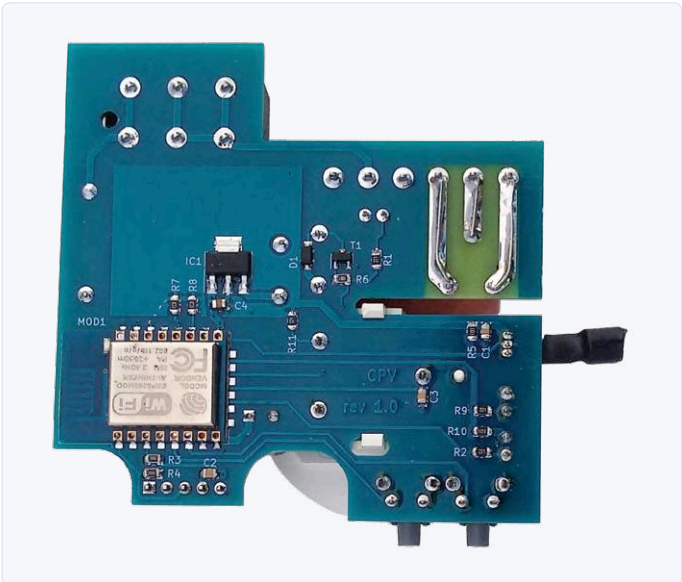


Figure 4. Tous les CMS sont sous la carte. Les pistes connectées au chauffage (en haut à droite) ont été renforcées, avec plus de soudure, pour leur permettre de conduire plus de courant sans surchauffer. La découpe juste en dessous d'elles fait office d'isolation galvanique entre les parties sous haute et basse tension. La protubérance noire à droite est le capteur de température, protégé par une gaine thermorétractable. Tout ce qui produit de la chaleur est à gauche de la carte qui est censée pointer vers le haut lorsque le thermostat est en place contre le mur.



Figure 6. Assemblage terminé. Les deux boutons-poussoirs sont accessibles par deux découpes (à gauche) et le capteur de température dépasse en dessous. Les LED verte et jaune ne sont visibles qu'à travers les fentes d'aération pour éviter d'éclairer la pièce la nuit. Elles sont de couleur différente pour qu'on voie bien laquelle des deux est allumée.

```
    unit_of_measurement: "°C"
    on_value:
     then:
     - lambda: |-
       auto call = id(t_controller).make_call();
       call.set_target_temperature(x);
       call.perform();
```

La partie à la fin après *on_value:* est un automatisme et sera expliquée après l'introduction du composant « climate » ci-dessous. Mais d'abord nous examinerons le relais.

Commuter le chauffage

C'est très simple, car le relais est juste un interrupteur connecté à la broche 4 du GPIO et fait donc partie de la plateforme *gpio*. Comme le capteur de température, il faut un *id (heater)* pour le rendre accessible depuis les autres sections du fichier YAML.

```
switch:
   - platform: gpio
     pin: GPIO4
     name: "Heater"
     id: heater
```

Climatisation

Bon, nous avons maintenant un capteur pour mesurer la température de la pièce (*t_room*), un potentiomètre pour régler la température de consigne (*t_target*) et un relais pour commuter le chauffage (*heater*).

ESPHome dispose d'un composant *climate* pour commander les appareils de chauffage et de rafraichissement. Un thermostat est donc un composant *climate*. L'avantage d'utiliser un composant intégré est de vous économiser du travail. On a aussi la garantie d'avoir un joli composant graphique de commande dans l'interface utilisateur de *Home Assistant*.

```
climate:
   - platform: thermostat
     name: "Thermostat"
     id: t_controller
     sensor: t_room
     default_target_temperature_low: 20 °C
     heat_action:
      - switch.turn_on: heater
     idle_action:
      - switch.turn_off: heater
     hysteresis: 0.5
     away_config:
      default_target_temperature_low: 15 °C
```

Notre thermostat ne rafraîchit pas ; il se contente de chauffer. Cela fait de lui un membre de la plateforme *thermostat* d'ESPHome. Il a besoin d'un *id* pour pouvoir être commandé par d'autres choses dans le fichier YAML. J'ai mis l'ID à *t_controller* parce que la plateforme s'appelle déjà *thermostat* et qu'il ne faut pas tout mélanger.
Le composant *climate* dispose d'une entrée capteur de température, que nous connectons à *t_room*. Il doit aussi être connecté à un chauffage, ce que nous pouvons faire grâce au commutateur *heater* défini plus tôt.

Comme moi, vous vous attendriez à ce qu'un composant *climate* ait aussi une entrée pour la température de consigne, mais il n'en a pas. (Peut-être sera-t-elle ajoutée dans une prochaine version d'ESPHome ?). À la place, il a une commande intégrée de réglage de la température de consigne, un peu comme le potentiomètre du thermostat. Heureusement, on peut contourner cette limitation par l'utilisation de blocs nommés « lambda ».

Blocs Lambda

Le concept est à la fois remarquable et horrible. Remarquables, car ils vous laissent faire tout ce que vous voulez ou presque, et affreux parce qu'ils défient tous les principes de configuration d'un appareil par le biais d'un simple fichier YAML sans avoir à connaître le C++.
En clair, un bloc lambda est du code C++ injecté tel quel dans le projet ESPHome. Pour pouvoir faire cela, les développeurs ont dû se démener pour parvenir à une interface utilisable entre YAML et C++. Le résultat est un code lambda plus compliqué qu'il ne l'aurait été s'il s'était agi de simple C++.
En fait, dès que vous vous retrouverez à utiliser les blocs lambda, vous devriez commencer à penser à la création d'un composant personnalisé. ESPHome prend en charge les composants personnalisés pour quasiment tout. J'étais sur le point d'écrire un composant *climate* personnalisé pour résoudre mes problèmes quand j'ai décidé de garder ce sujet pour un autre article.
Retour au réglage de la température de consigne du composant *climate*. Son interface C++ a une fonction pour fixer la température de consigne. Le bloc lambda de la section d'automatisation *on_value* du capteur potentiomètre précédent (reproduite ci-dessous) montre comment l'utiliser. Chaque fois qu'une nouvelle valeur est disponible, la méthode *set_target_temperature* du composant *climate t_controller* est appelée, comme ceci :

```
  on_value:
     then:
     - lambda: |-
       auto call = id(t_controller).make_call();
       call.set_target_temperature(x);
       call.perform();
```

En C++ normal, cela donnerait quelque chose comme cela :

```
t_controller.set_target_temperature(x);
```

Si le composant *climate* avait eu une entrée de température de consigne, les choses auraient été encore plus simples. Par exemple :

```
climate:
   - platform: thermostat
     name: "Thermostat"
     id: t_controller
     sensor: t_room
     target: t_target
     …
```

Passons.
Malheureusement, cette façon d'automatiser *on_value* est trop simpliste, elle outrepasse la commande à distance de la température de consigne (par ex. avec *Home Assis*tant). Pour résoudre cela, l'automatisation ne devrait pouvoir fonctionner que lorsque t_target change (c.-à-d. lorsque quelqu'un tourne le potentiomètre). Comme ESPHome n'a pas

d'automatisation *on_value_changed* pour les capteurs, nous pouvons gérer cela nous-mêmes dans le bloc lambda, comme ceci :

```
on_value:
   then:
   - lambda: |-
     static float x_last = 0.0;
     if (x<x_last-0.1 || x>x_last+0.1)
     {
      x_last = x;
      auto call = id(t_controller).make_call();
      call.set_target_temperature(x);
      call.perform();
     }
```

La faible hystérésis sur x (±0,1 dans ce cas) améliore l'immunité de l'automatisme au bruit du capteur.

Le filtre delta

Une autre solution – plus élégante – consiste à utiliser le filtre delta sur *t_target*. Ce filtre ne transmet une nouvelle valeur que si elle diffère de la précédente de plus ou moins delta. Ainsi, si delta = 1 et que la dernière valeur était égale à 20, alors la prochaine valeur ne sera transmise que si elle est soit inférieure à 19 soit supérieure à 21.

```
- platform: adc
   name: "Target temperature"
   id: t_target
   …
   filters:
    - multiply: 25.0
    - offset: 6.75
    - delta: 0.2
   …
```

Les filtres sont exécutés dans l'ordre où ils apparaissent, c.-à-d. que le filtre delta travaille sur la valeur convertie en degrés Celsius et pas directement sur la tension en entrée. Sa valeur doit être faible, pour pouvoir facilement ajuster un peu le thermostat vers le haut ou vers le bas, ce qui fait toute la différence pour le confort de l'utilisateur.

Terminer le dispositif

Une fois le fichier YAML prêt, on peut télécharger le micrologiciel sur le module WiFi. La première fois (c.-à-d. avec un module qui ne fonctionne pas déjà avec un logiciel compatible avec ESPHome), on doit utiliser le port série pour cela (disponible sur le connecteur K2). Retrouvez la procédure exacte en [3]. Une fois que le dispositif fonctionne avec ESPHome et que la programmation *Over-the-Air* (OTA) est activée (lorsque le fichier YAML contient la ligne *ota*:), on peut reprogrammer le thermostat sans être physiquement connecté au système de développement. En d'autres termes, on peut l'installer à la place du thermostat existant.
La disposition du thermostat est telle que le capteur de température n'est pas réchauffé par le module WiFi ni l'alimentation.

Conclusion

Cet article a montré comment on peut remplacer un banal thermostat par un autre, fait-maison, intelligent et connecté, compatible avec la domotique. Bricoler un prototype qui fasse à peu près ce qu'il est censé faire n'a pas été trop difficile. Ce qui demande plus d'efforts, c'est de le rendre utilisable par tout le monde, à tout moment, et sans malmener l'esthétique. Nul doute qu'il reste à peaufiner certains réglages du nouveau thermostat, mais comme je pourrai le reprogrammer à distance (sur le terrain comme on dit), ce sera facile.

200519-02

Votre avis, s'il vous plaît ?
Veuillez adresser vos questions et vos commentaires par courriel à clemens.valens@elektor.com ou à redaction@elektor.fr.

Ont contribué à cet article

Auteur : **Clemens Valens**
Schémas : **Patrick Wielders**
Rédaction : **C. J. Abate**
Maquette : **Giel Dols**
Traduction : **Denis Lafourcade**

PRODUITS

- **ESP-12F, ESP8266-based Wi-Fi module**
www.elektor.fr/esp-12f-esp8266-based-wi-fi-module-160100-92
- **Thermostat de bureau WiFi – circuit imprimé nu (160269-1)**
www.elektor.fr/wi-fi-desktop-thermostat-bare-pcb-160269-1
- **Carte à microcontrôleur NodeMCU ESP8266**
www.elektor.fr/nodemcu-microcontroller-board-with-esp8266-and-lua
- **H. Henrik Skovgaard, *IoT Home Hacks with ESP8266* (Elektor, 2020)**
www.elektor.fr/iot-home-hacks-with-esp8266

LIENS

[1] **Roy Aarts & Clemens Valens, " thermostat de bureau wifi", Elektor, janv.-fév. 2018** : www.elektormagazine.fr/magazine/elektor-201801/41300
[2] **Clemens Valens, «Wi-Fi Thermostat at Elektor Labs,» Elektor Labs, 2018** : http://bit.ly/wifi-thermostat
[3] **Clemens Valens, «la domotique, c'est facile avec…», sept.-oct. 2020** : www.elektormagazine.com/200019-02

bureau d'études – Zone D

D comme développement, comme débrouillardise et dur-à-cuire

Astuces, bonnes pratiques et autres informations pertinentes

Clemens Valens (Elektor Labs)

Commutation positive ou commutation négative ?

Dans les circuits à commutation positive, le commutateur est inséré entre l'alimentation positive et la charge. La commutation négative est en revanche reliée entre charge et masse (**fig. 1**). Les principes de fonctionnement sont simples dans les deux cas, mais pourquoi choisir une méthode plutôt que l'autre ? Tout dépend de l'application.

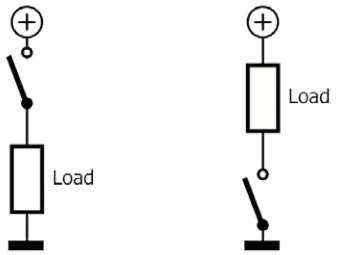


Figure 1. Commuter une charge sur le rail d'alimentation (gauche) ou sur la ligne de masse (droite) ?

Éviter les situations dangereuses

On privilégiera la commutation positive (**fig. 2**) dans les situations où un court-circuit à la masse est plus probable qu'un court-circuit à la ligne d'alimentation positive. C'est le cas des voitures ou des machines dont l'essentiel de la structure est relié à la masse ou au châssis. Dans de tels cas, il est plus prudent de déconnecter la charge de la batterie, plutôt que de la masse. En outre, dans les environnements humides, la corrosion du connecteur sera généralement moindre, du fait de l'absence de tout potentiel sur la charge une fois le circuit coupé.

La commutation de puissance est meilleure avec des transistors de type N

Les transistors de type N supportent généralement plus de courant que ceux de type P, ils sont donc conseillés pour commuter de lourdes charges. La commutation négative avec transistors de type N est plus simple, et peut d'ailleurs souvent être assurée directement par les ports du µC, sans étage de commutation de puissance. Il est possible de commuter dans la branche positive avec un transistor de type N, mais au prix d'une tension de commande plus élevée que la tension de la charge reliée à sa source ou à l'émetteur. Pour porter le collecteur ou la base du transistor à un potentiel supérieur à sa tension de source ou d'émetteur, il faut recourir à une sorte de pompe de charge, ou à une alimentation auxiliaire. Cela complique le circuit, en alourdit le coût, et augmente sa sensibilité au bruit et aux interférences.

Si vous commandez par modulation de largeur d'impulsion (MLI ou PWM) p.ex. la vitesse d'un moteur ou la luminosité d'une LED

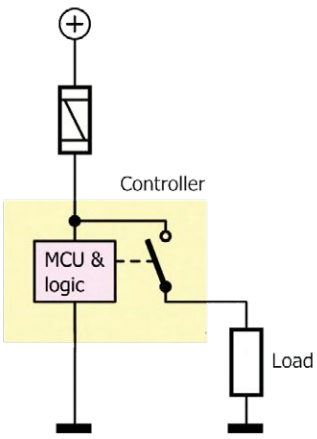


Figure 2. La commutation positive est recommandée quand le risque de courts-circuits à la masse est le plus probable.

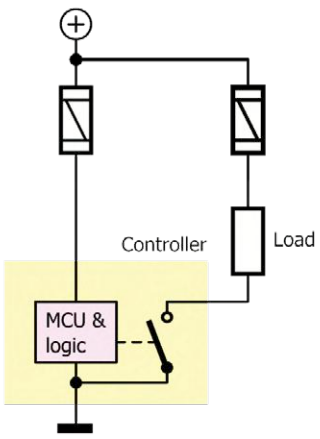


Figure 3. La commutation négative, illustrée ici à gauche est simple et économique. Si l'éloignement entre charge et commande (à droite) impose la présence de deux fusibles, cette configuration peut devenir plus onéreuse que la commutation positive.

par commutation de la branche positive, la pompe de charge risque de poser des problèmes.

Un fusible fait toute la différence

La commutation négative est donc moins coûteuse que la commutation positive. Si la charge et son circuit de commande sont éloignés l'un de l'autre, les règles de l'art du câblage imposent toutefois deux fusibles distincts pour protéger ces deux éléments, plutôt qu'un seul (**fig. 3**). La commutation positive se contente d'un seul. Cela peut paraître anecdotique, mais à considérer le câblage et le travail supplémentaires pour rendre le fusible accessible dans un porte-fusible, cette adjonction pourrait annihiler le gain financier de la commutation négative.

Rien de tel qu'une bonne masse

La commutation négative comporte un potentiel de masse commun à la charge et à son contrôleur. Cela évite les différences de potentiel de masse (masse flottante) en présence de courants de forte intensité et un grand éloignement entre charge et circuit. La commutation négative est ainsi moins sensible aux interférences par la masse que la commutation positive.
D'autre part, même fermé, un commutateur introduit une certaine résistance en série avec la charge. Celle-ci se traduit par une petite chute de tension, de sorte que le potentiel de la masse de la charge est légèrement supérieur au potentiel de masse du circuit de commande.

Alors, que faire ?

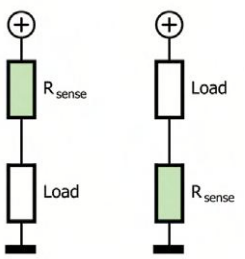
Dans les cas où il suffit d'activer ou de couper une charge (lourde), la commutation positive est préférable. Quand l'alimentation d'une charge doit être commandée par un signal PWM (relativement) rapide, dans un gradateur d'éclairage ou de chauffage par exemple, on préconisera la commutation négative.
Il existe aussi les demi-ponts en H qui combinent les commutations positive et négative… et la commande MLI … Avant de faire son choix, il faut peser le pour et le contre en fonction de l'application.

190369-F-02

DÉTECTER LE COURANT CÔTÉ TENSION POSITIVE, OU NÉGATIVE ?

Plus ou moins : quel côté choisir ?

De même que pour la commutation, se pose le problème de la détection du courant. Il existe plusieurs techniques pour cela, la plus connue étant certainement celle qui consiste à placer une petite résistance d'une valeur donnée, en série à la masse. Comme l'intensité du courant à travers la charge est la même que celle du courant à travers la résistance en série, la loi d'Ohm permet de calculer le courant en mesurant la chute de tension sur la résistance qu'il suffit de diviser par la valeur de la résistance. Où placer cette résistance, avant ou après la charge ? Autrement dit, doit-on détecter le courant côté plus ou côté moins ?



Détection de courant côté moins (à gauche) et côté plus (à droite).

Détection moins chère côté moins

Lors d'une détection côté moins, la tension généralement basse sur la résistance, est référencée à la masse et peut être amplifiée facilement par des amplificateurs opérationnels basse tension peu coûteux, avant d'être numérisée et traitée par un microcontrôleur, par exemple. Néanmoins, comme le commutateur lors d'une commutation négative, la résistance de détection (aussi appelée *shunt*) cause une (légère) baisse de tension qui relève le potentiel de la charge par rapport à celui de la masse, susceptible donc de provoquer des potentiels parasites sur la masse (*ground bounce*).
Par ailleurs, la détection côté moins ne permet pas de savoir que la charge est court-circuitée à la masse, qu'elle le soit en interne ou qu'elle le soit à l'un de ses terminaux.

Gérer les tensions en mode commun élevées

La détection de courant côté plus ne souffre pas de ces problèmes-là, elle en a d'autres. Par exemple, la référence de la tension sur la résistance de détection n'est pas la masse. Pour la rapporter au potentiel de la masse, il faut des circuits supplémentaires, qui compliquent les choses et font monter le prix.
De plus, la tension de mode commun que doit supporter l'amplificateur doit être égale à l'alimentation positive (et même plus élevée en cas de transitoires), soit des centaines de volts dans certaines applications. Par conséquent, l'amplificateur doit soit supporter une tension d'alimentation élevée, soit disposer d'entrées capables de gérer des tensions de mode commun aussi élevées. Enfin, pour contenir les erreurs de tension en mode commun, il faut utiliser des composants de précision, plus coûteux.
Pour faciliter un peu la vie du concepteur, plusieurs fabricants de semi-conducteurs proposent des circuits intégrés de détection de courant côté plus.

Conclusion ?

La méthode de détection de courant à choisir dépend de l'application (*bis*). La détection de courant côté plus peut

détecter si la charge est court-circuitée ou ouverte, et la charge reste référencée à la masse. Cependant, en raison des exigences du mode commun, elle est plus complexe et plus coûteuse que la détection du courant côté moins. Le recours à un circuit intégré spécial pour la détection de courant pourrait être la meilleure solution.

Votre avis, s'il vous plaît ?
N'hésitez pas à poser vos questions ou envoyer vos commentaires sur cet article à l'auteur clemens.valens@elektor.com

DEUX MÉTHODES DE DÉTECTION DE COURANT CÔTÉ HAUTE TENSION

Fondamentalement, l'amplificateur de détection de courant (à gauche) divise le courant à travers R_{sense} par un facteur de R1/R_{sense} (= 1 000) et le force à travers R3. La tension sur R3 est donc proportionnelle au courant traversant R_{sense}. Un courant de 1 A à travers R_{sense} se traduit par un courant de 1 mA à travers R3 (1 kΩ), ce qui donne une tension de 1 V sur R3.

L'amplificateur différentiel (à droite) multiplie la tension sur R_{sense} par R4/R2 (= 10). Un courant de 1 A à travers R_{sense} produit une tension de 0,1 V sur R_{sense}, qui, multipliée par 10, donne 1V à la sortie de l'amplificateur.
Les deux circuits produisent la même sortie, mais ils ne sont pas équivalents.

Bande passante

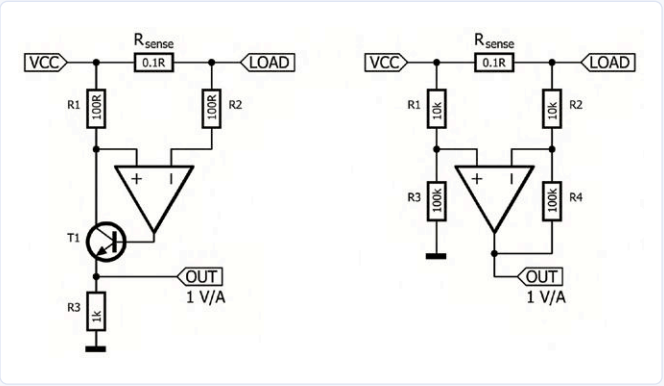
Premièrement, la largeur de bande des deux circuits n'est pas la même. Celle de l'amplificateur différentiel est généralement (beaucoup) plus étroite que celle de l'amplificateur de détection de courant et donc mieux adaptée à la mesure du courant moyen dans une charge. L'amplificateur de détection de courant, d'autre part, est rapide et peut mesurer le courant instantané de la charge à des fréquences élevées.

CMR

Un autre facteur important est le rejet en mode commun ou CMR. Comme la tension de mode commun aux entrées est élevée, une asymétrie, même petite, entre les deux entrées entraînera une erreur à la sortie de l'amplificateur. Il faut donc pour l'amplificateur différentiel des résistances de précision pour maintenir l'erreur CMR aussi faible que possible. Dans l'amplificateur de détection de courant, c'est principalement l'ampli op qui détermine le CMR, un paramètre fixé par son fabricant.

Robustesse

D'autre part, grâce à ses résistances externes, un amplificateur différentiel peut plus facilement être adapté à des tensions de mode commun très élevées que l'amplificateur de détection de courant. Ceci nous conduit au sujet de la robustesse du circuit de détection de courant. Si des transitoires avec des tensions excessives devaient apparaître sur la résistance de détection, l'amplificateur différentiel serait protégé par ses résistances de forte valeur. Dans de tels cas, l'amplificateur de détection de courant doit principalement sa survie aux protections contre les surtensions d'entrée intégrées à son amplificateur opérationnel. De même, un amplificateur différentiel résiste généralement mieux en cas d'inversion de polarité de l'alimentation.



Il existe plusieurs méthodes de détection des courants côté haute tension. L'amplificateur différentiel et le moniteur de courant en dérivation ou l'amplificateur de détection de courant présentés ici en sont deux.

Consommation d'énergie

Enfin, la question de la consommation, cruciale dans les applications à consommation (ultra) faible. S'il est connecté directement à l'alimentation, l'amplificateur différentiel avec ses résistances consommera toujours un peu de courant même lorsque son ampli op n'est pas alimenté. L'amplificateur de détection de courant présente une impédance d'entrée beaucoup plus élevée, ce qui ménage davantage les piles.

Circuits intégrés spéciaux

Pour balayer toutes ces subtilités, les fabricants de semi-conducteurs proposent des circuits intégrés spéciaux de détection de courant pour différentes topologies, facilitant ainsi la vie des concepteurs souhaitant ajouter la détection de courant à leurs applications.

elles sont petites, mais font de grandes choses

Les pépites d'Elektor

Compilation : **Eric Bogers**

Vous n'êtes pas d'humeur à entreprendre de grands travaux ? Ça tombe bien, voici encore quelques petits circuits comme on les aime bien. Rien qu'à les regarder, on comprend déjà comment ils marchent et on se sent bien, même sans les avoir réalisés.
Un remède idéal contre le «blues de janvier»…

Idée: Peter Neufeld (Allemagne)

Tranquili-T

Généralement ce ne sont ni les grandes idées ni les grands projets d'un passionné d'électronique que remarquent et apprécient ses amis et sa famille. Ni l'oscillateur ultra-stable ni l'asservissement linéaire ne sont un plaisir pour les foules. Essayez peut-être ce circuit, le Tranquili-T qui a reçu beaucoup d'éloges de la part des proches de son auteur. À tel point qu'il a dû leur en construire plusieurs.
Le circuit (**fig. 1**) est construit autour d'un lecteur MP3 bon marché, dans lequel on a chargé de la musique (agréable). L'effet apaisant de la musique est renforcé par un éclairage tamisé fourni par une LED arc-en-ciel. Une vidéo de l'auteur montre son Tranquili-T en action [1].
Pour l'alimentation, fera l'affaire toute source de 4,5 à 5,0 V, trois piles de 1,5 V par exemple, mais n'oubliez pas que la LED consomme environ 20 mA et que le lecteur, à volume modéré, demande ses 50 mA. Une batterie USB rechargeable serait donc une meilleure option.
La solution la plus simple est l'utilisation d'un lecteur MP3 avec ampli de classe D intégré. L'auteur a été satisfait du JQ6500 et du DFplayer Mini trouvés à bas prix chez ses fournisseurs en Extrême-Orient. La figure 1 présente deux approches de la commande de volume (cf. encadrés). La variante avec S2 et S3 utilise deux entrées numériques ; la variante avec S4 et S5 combine une seule entrée analogique et deux résistances différentes. Les valeurs dans le schéma s'appliquent au JQ6500. Il faudra éventuellement les adapter à d'autres modèles de lecteurs MP3 en tâtonnant. Pensez aussi à faire des recherches sur l'internet.
Comme HP, choisissez un modèle aussi petit et plat que possible avec une puissance d'au moins 0,5 W et une impédance de 8 à 32 Ω.
L'auteur a constaté que le volume était trop élevé avec un HP de 4 Ω. La **figure 2** montre une version construite autour du DFPlayer Mini, la **figure 3** montre un Tranquili-T complet.

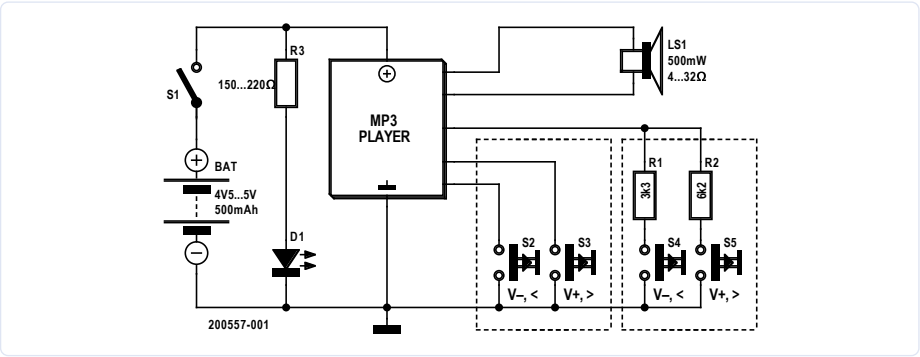


Figure 1. Le cœur du circuit est un lecteur MP3 (module).



Figure 2. Montage compact sur un morceau de circuit imprimé à pastilles.



Figure 3. Le Tranquili-T en pleine gloire.

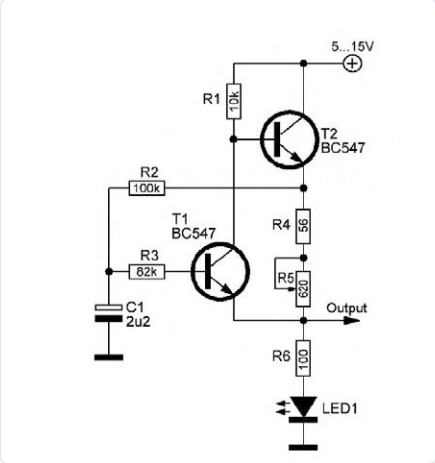


Figure 4. Schéma de principe d'un générateur d'impulsions basé sur un trigger de Schmitt modifié.

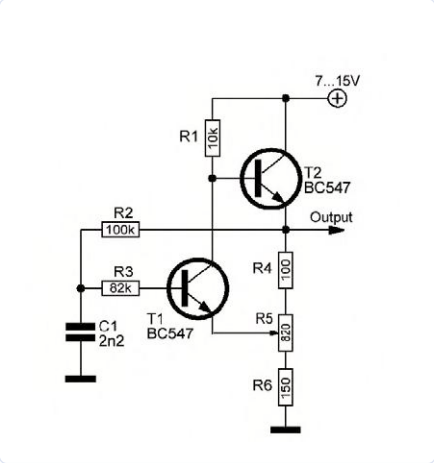


Figure 5. Variante du générateur d'impulsions basé sur un trigger de Schmitt modifié.

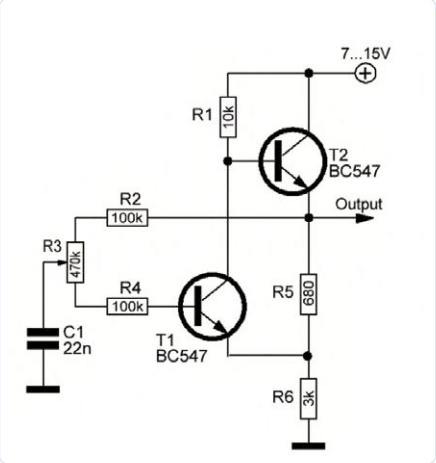


Figure 6. Une autre variante ; ici, la durée de l'impulsion est réglable.

Idée: Andrey M. Shustov (Russie) & Michael A. Shustov (Allemagne)

Trois variantes de l'oscillateur à trigger de Schmitt

Les *triggers* ou déclencheurs (*asymétriques* à émetteurs couplés) de Schmitt sont utiles pour obtenir de simples oscillateurs à impulsions à large gamme de fréquence. Pour transformer en oscillateur un déclencheur de Schmitt, sa sortie doit être reliée à son entrée par une résistance, tandis qu'un condensateur sera connecté entre cette entrée et le rail d'alimentation.

Les figures 4, 5 et 6 montrent des circuits pratiques d'oscillateurs à impulsions basés sur le déclencheur de Schmitt modifié. Un circuit en pont se compose d'un diviseur résistif, de l'émetteur du transistor d'entrée connecté au point central du diviseur résistif, et d'un circuit RC de réglage de la constante de temps par l'intercession de laquelle le déclencheur devient oscillateur à impulsions répétitives. La jonction p-n du transistor est connectée entre une paire de jonctions diagonalement opposées d'un pont. Lorsque C1 est déchargé, T1 se bloque et T2 conduit. Lorsqu'au fil de la charge du condensateur, la tension à ses bornes dépasse de plusieurs volts la tension au point central du diviseur résistif, le transistor d'entrée T1 conduit et T2 se bloque. Par conséquent, le diviseur résistif est mis hors tension, et C1, le condensateur de la constante de temps, est déchargé. De ce fait, le T1 est à nouveau bloqué, ce qui rend T2 conducteur. Le processus se répète.

La fréquence d'oscillation de l'oscillateur à impulsions (**fig. 4**) peut être modifiée proportionnellement en ajustant la capacité de C1. Le potentiomètre R5 permet de couvrir une gamme de fréquences supérieure à 1:10. La LED1 permet un contrôle visuel de l'accord de la fréquence : l'intensité de l'éclairage atteint sa valeur maximale au début de la plage et sa valeur minimale à la fin de la plage. L'oscillateur fonctionne entre 3 à 30 Hz. Avec les valeurs nominales indiquées, le courant à travers la LED1 varie entre 20 et 2 mA pour une tension d'alimentation de 9 V. Sur une plage de tension d'alimentation de 5 à 15 V, la fréquence d'oscillation ne varie pas de plus de 10 %.

Les oscillateurs à impulsions (**fig. 5 et 6)** fonctionnent à une tension d'alimentation de 9 V nominalement dans la gamme de fréquences de 0,8 à 10 kHz et de 0,35 à 2,8 kHz, respectivement. L'oscillateur de la figure 5 est contrôlé en modifiant le rapport des diviseurs de tension résistifs (R4, R5, R6, la moitié droite du circuit en pont). Le mode de fonctionnement de l'oscillateur de la figure 6 est commandé par la chaîne R2, R3 et R4 qui régulent les processus de décharge et de charge dans la moitié gauche du circuit en pont.

Idée : Elektor Labs

Avant l'entrée en scène du traitement numérique du signal audio, les musiciens et leurs techniciens étaient d'étonnants bricoleurs. Pour obtenir certains effets sonores, ils se faisaient menuisiers et mécaniciens. Prenons l'exemple de l'orgue Hammond, précurseur des instruments électroniques modernes. Pour donner de l'ampleur au timbre de cet instrument monophonique, Donald Leslie a imaginé un traitement acoustique ***mécanique*** du son issu des haut-parleurs. Au lieu de le projeter directement dans la pièce, le son est projeté sur un déflecteur en mouvement rotatif placé sur la trajectoire des ondes acoustiques.

De cette centrifugation résulte un effet sonore qui s'apparente à celui des sources sonores en mouvement rapide, les sirènes de police, le fameux effet Doppler. Si le signal direct est superposé au signal ainsi trituré, l'image sonore résultante paraît beaucoup plus ample. Aujourd'hui, on ne s'aventure plus dans la fabrication d'un tel dispositif mécanique. Le modèle de l'illustration (**fig. 1**) est une vue d'artiste d'un tel meuble : un cylindre tournant, dans lequel est percée une fenêtre, est placé au-dessus d'un grand haut-parleur.

Ce circuit date d'une quarantaine d'années. C'est un équivalent électronique analogique, qui imite l'effet obtenu avec une cabine Leslie à moindres frais. Il est instructif si vous aimez les bricolages sonores (**fig. 2**).
À gauche, un multivibrateur astable discret,

dont la fréquence très lente peut être réglée avec P1 entre environ 1 et 8 Hz. Attention au câblage du potentiomètre double, dit stéréo, dont la caractéristique est linéaire (et non logarithmique). Les ampoules La1 et La2 servent de résistances de collecteur pour les transistors et font, en s'allumant et en s'éteignant tour à tour, varier la résistance des LDR R3 et R4 devant lesquelles elles sont placées.

Un signal sonore mono arrive à C3. Il est divisé en deux par P2. Ce potentiomètre est en quelque sorte un réglage de balance statique entre les deux sorties qui vont aux entrées L et R d'un ampli stéréo. Les deux photorésistances atténuent à tour de rôle une partie du signal mono en le conduisant plus ou moins directement vers la masse selon la valeur de la résistance de la LDR dans cette branche. Or celle-ci dépend à son tour de la lumière reçue. C'est le rôle de la lumière émise alternativement par les deux lampes commandées par le multivibrateur de faire varier le volume du signal sur les deux canaux.

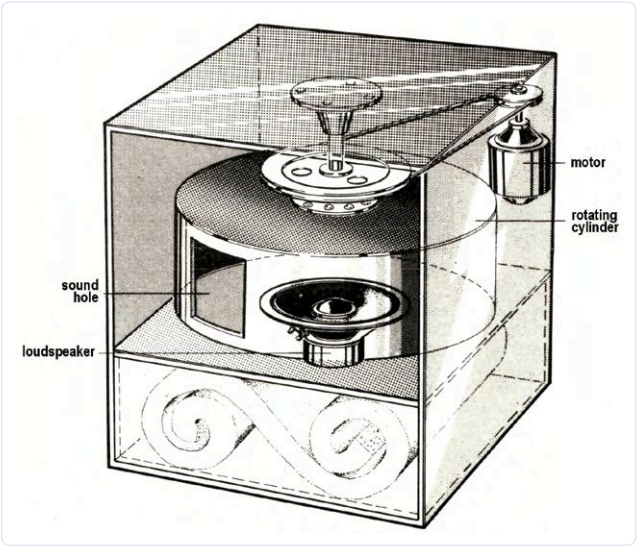


Figure 7. Vue d'artiste de la cabine Leslie : un HP monté horizontalement avec un cylindre rotatif ouvert sur le côté.

Ce circuit fait appel à des ampoules à filament parce qu'il est ancien, mais aussi parce qu'elles ont un une certaine inertie qui arrondit les angles lors de l'allumage et de l'extinction. Pour obtenir une courbe aussi progressive avec des LED, il faudrait rajouter quelques composants pour créer une oscillation en dents de scie.

Il est important de monter chaque lampe à proximité immédiate de la LDR correspondante. Il faut ensuite les isoler chacune séparément de toute lumière incidente, et pour qu'elles ne puissent pas s'influencer mutuellement. Pour que ce genre d'effets sonores soient bien perceptibles, il importe aussi de mélanger une certaine proportion du signal original au signal transformé.

200557-02

Votre point de vue, s'il vous plaît ?
Adressez vos questions ou vos commentaires par courriel à redaction@elektor.fr

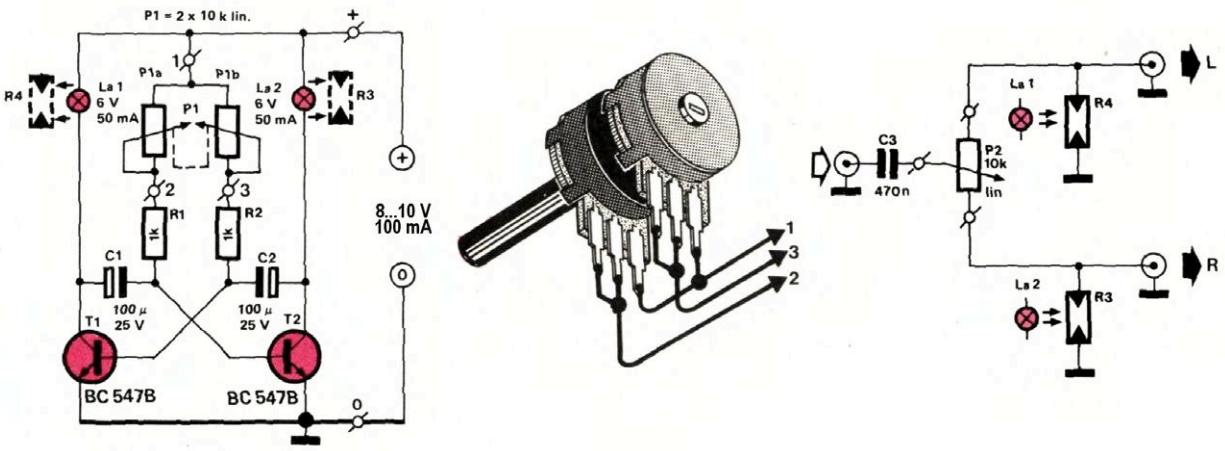


Figure 8. Le circuit (à gauche) est relativement simple. Au milieu, on voit comment câbler le potentiomètre stéréo, et à droite, comment le tout est relié à un amplificateur.

LIENS

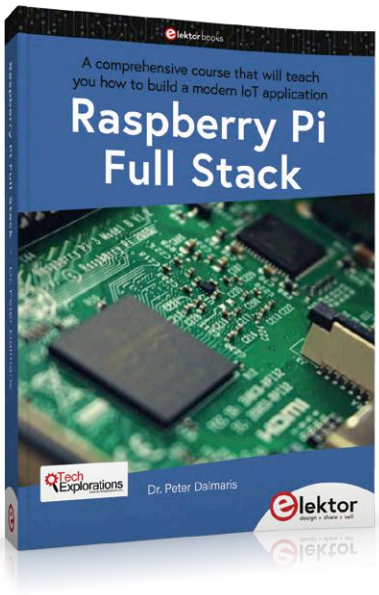
[1] **Tranquili-T en action** : https://vimeo.com/272785420
[2] **Donald Leslie** : https://en.wikipedia.org/wiki/Donald_Leslie

PRODUITS

- **Livre (en anglais) "Electronic circuits for all"**
  www.elektor.fr/electronic-circuits-for-all

Raspberry Pi Full Stack

RPi et RF24 au cœur d'un réseau de capteurs



Peter Dalmaris (Australie)

Cet extrait d'un livre d'Elektor propose en condensé deux chapitres du livre de Peter Dalmaris *Raspberry Pi Full Stack*, publié récemment par Elektor. Destiné aux utilisateurs de RPi curieux et désireux d'associer matériel et logiciel dans leur voyage initiatique, le potentiel éducatif de ce livre est riche. Voici comment y est décrite en détail la combinaison d'un RPi avec un module RF24 pour lire des capteurs distants intégrés dans un petit réseau.

Nœud de base du réseau RF24, le Raspberry Pi traite les valeurs des capteurs qu'il reçoit du nœud Arduino (ou des nœuds, s'il y en a plusieurs). Le nœud Arduino est décrit dans les chapitres précédents du livre, non repris ici. Ce chapitre explique la connexion du module RF24 au Raspberry Pi. Vous pouvez la réaliser sur une carte d'essai ou utiliser mon module HAT spécialement conçu à cet effet.

J'ai réalisé mon module RF24 Raspberry Pi en utilisant mon module HAT (**fig. 1**). Le HAT est équipé d'un module radio nRF24, d'un DHT22, d'un poussoir et de deux LED (témoins d'alimentation et d'activité). Les fichiers Gerber du HAT sont téléchargeables sur la page du projet [1]. Vous pouvez aussi commander ce circuit imprimé directement auprès de PCBWay [2].

La liste des connexions (**fig. 2**), et le schéma du circuit imprimé (**fig. 3**) vous permettront de réaliser un prototype de ce circuit sur plaque d'essai. Vous pouvez obtenir une version à haute résolution du schéma du HAT sur la page du projet [3].

La liste suivante contient tous les composants nécessaires pour réaliser le circuit de la fig. 2. La connexion de certains composants, dont le poussoir et le capteur DHT22, a déjà été décrite plus tôt dans le projet. Les seuls nouveaux composants sont l'émetteur-récepteur nRF24 et son condensateur de découplage.

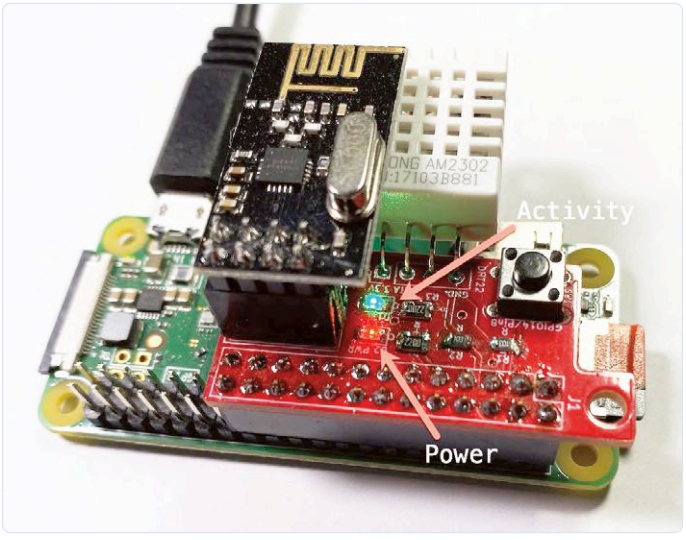


Figure 1. Le HAT installé sur un Raspberry Pi Zero W.

Raspberry Pi - RF24 - DHT22

Raspberry Pi	nRF24	DHT22	Comments
3.3V	Vcc	Pin 1 - Power	220uF elec. cap pin "+".
GND	GND	Pin 4- GND	220uF elec. cap pin "-".
GPIO 11	SCK		
GPIO 9	MISO		
GPIO 10	MOSI		
GPIO 8	CSN		
GPIO 7	CE		
GPIO4 (Pin 7)		Pin 2 - Data	Add a 10KOhm pull up resistor between Data and GND.

Raspberry Pi Full Stack — Tech Explorations

Figure 2. Les détails de connexion pour les modules nRF24 et DHT22.

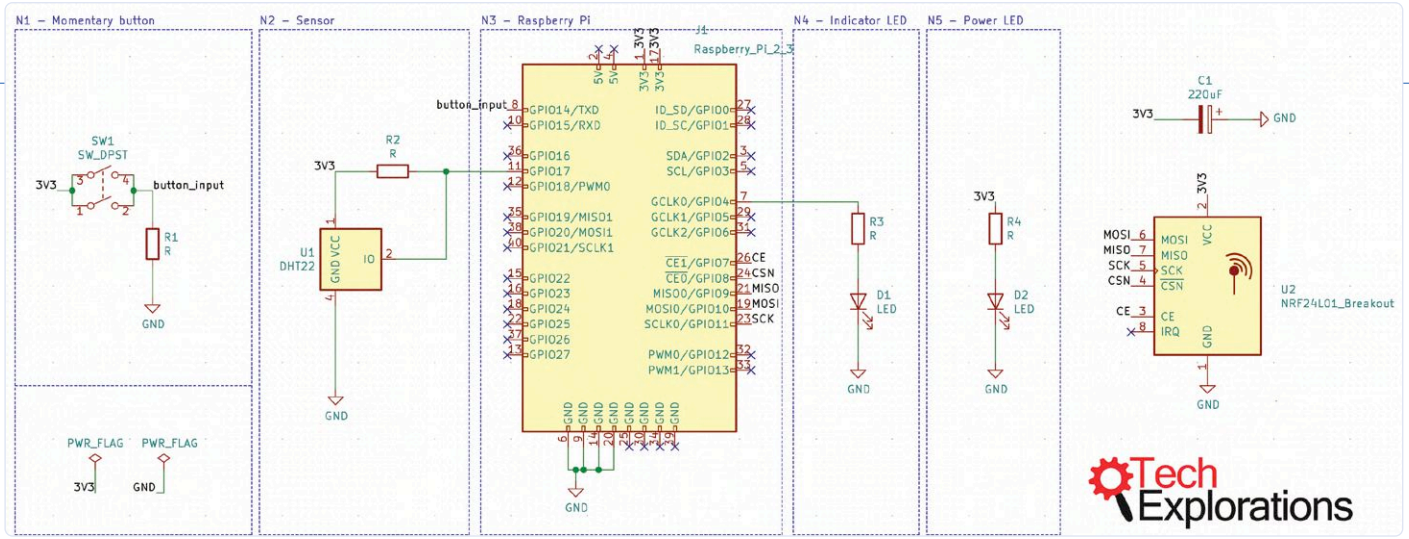


Figure 3. Le schéma du HAT RPi pour la carte d'accès au nRF24 et au DHT22.

Composants nécessaires :

- un capteur DHT22
- un module d'accès nRF24
- deux résistances de 10 kΩ
- deux résistances de 330 Ω
- une LED rouge (témoin d'alimentation)
- une LED bleue (témoin d'activité)
- un condensateur électrolytique de 220 µF ou équivalent

Le script récepteur du RPi nRF24

Ce chapitre présente la fonction du script Python récepteur qui tourne sur le RPi et prend en charge les communications nRF24. Ce script n'est pas prêt à l'emploi. Il dépend des pilotes en langage C de RF24 et RF24Network et des encapsulations Python dont la configuration est traitée dans le prochain chapitre. Pour l'instant, occupons-nous du script Python du récepteur. Le code source complet de ce script est disponible sur la page du projet [4]. J'ai écrit ce script pour qu'il puisse tourner en arrière-plan contrôlé par `systemd` (comme a été configuré le script d'application web plus tôt dans ce projet). Mais ce sera pour plus tard, car il faut d'abord s'assurer que le script s'exécute correctement en avant-plan.

La fonction `log_values()` est une copie presque identique de la même fonction dans le script `env_log.py`, déjà configuré pour suivre un programme horaire. Recevant les valeurs des capteurs comme arguments, elle les écrit dans la base de données locale et dans le tableau Google Sheet dans le nuage.

Ci-dessous, je passe en revue certains éléments du script, en particulier ceux qui se rapportent aux communications RPI-nRF24.

- Juste après la définition de la fonction `log_values()`, le script initialise le module nRF24 en commençant par la variable `radio` où il utilise le constructeur RF24. Ce dernier fait partie de la bibliothèque Python d'encapsulation de RF24 qui permet d'utiliser le pilote C de RF24 à partir du script Python. Vous pouvez vous familiariser avec cette fonction en étudiant le code source du pilote [5].
- Une fois l'objet radio créé et initialisé, le script crée l'objet `network` à l'aide du constructeur RF24Network. C'est cet objet qui permet au Raspberry Pi de recevoir un envoi du nœud Arduino. Le seul paramètre nécessaire pour créer l'objet réseau est l'objet `radio` créé sur la ligne précédente. Pour en savoir plus sur le constructeur du RF24Network, examinez le code source de RF24Network.h [6] à la ligne 373.
- À la ligne suivante, avec `octlit = lambda n : int(n, 8),` nous créons une fonction lambda de conversion décimal en octal, nécessaire car le pilote du réseau RF24 code les adresses de nœuds en octal. En Python, une fonction `lambda` [7] ressemble à une fonction normale (définie avec le mot-clé `def`) mais n'a pas de nom. On l'utilise pour l'évaluation d'une seule expression, comme c'est le cas ici : `octlit` reçoit un nombre décimal et renvoie l'équivalent octal en utilisant la fonction Python `int()` [8].
- Dans `this_node = octlit(«00»)`, nous utilisons `octlit` pour obtenir l'équivalent octal de `00`, et nous stockons cette valeur dans la variable `this_node`. Il s'agit de l'adresse réseau RF24 du Raspberry Pi.
- Dans les six prochaines lignes, jusqu'au bloc `while`, le script va :
  - démarrer la radio RF24 ;
  - attendre 0,1 seconde que la radio soit prête ;
  - démarrer le réseau sur le canal 90 ;
  - afficher la configuration de la radio et du réseau sur la console ;
  - remettre à zéro le compteur `packets_sent ;`
  - et remettre à zéro le compteur `last_sent.`
- Maintenant que la radio et le réseau sont prêts, le script entre dans une boucle infinie d'attente d'une transition d'un nœud Arduino.
- Au début de chaque boucle, il appelle la méthode `update()` [9] de l'objet réseau pour guetter vos messages.
- S'il n'y a pas de nouveau message, le script s'endort pendant 1 seconde, puis relance la boucle.
- S'il y a un nouveau message, le script utilise la fonction `read()` [10] pour lire les 12 octets (la charge utile) du message, et les passer à la variable `payload`. La fonction read récupère aussi l'en-tête du message et le transmet à la variable `header`. Rappelez-vous que dans le sketch Arduino, nous avons créé une charge utile de 12 octets de la forme : `51.23,23.09`.
- Sur les deux lignes suivantes, j'utilise `print` pour envoyer la charge utile vers la console. Cela m'a servi dans mon travail de décodage de la charge utile en chiffres dans le bloc `try` qui suit.
- La variable `payload` contient une chaîne de caractères de la forme : `51.23,23.09`. Le script utilise `decode()` [11] pour la coder en UTF-8, et la fonction `split()` [12] pour séparer les

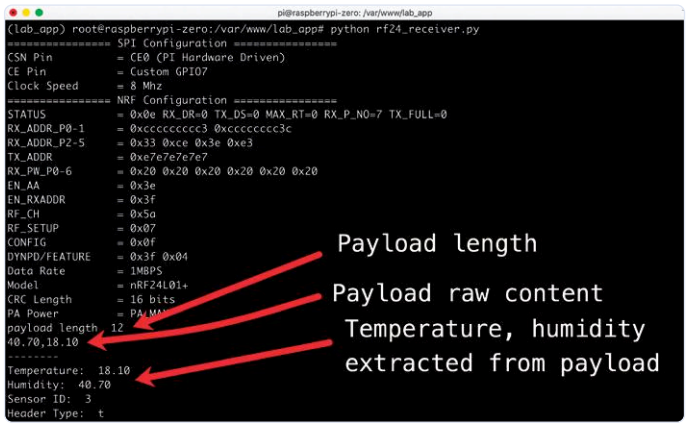


Figure 4. Exemple de sortie du script du récepteur RF24.

valeurs de température et d'humidité au niveau de la virgule et les stocker dans la variable `values` (un tableau de chaînes de caractères).

- Dans le bloc `try`, le script utilise la fonction `float()` [13] pour extraire les nombres stockés dans la charge utile et les convertir en nombre à virgule flottante. La fonction `float()` reçoit une chaîne de caractères, et si elle peut la convertir en un nombre, elle le renvoie.
- Avec `float(values[1][0:5])`, le script prend les 5 premiers caractères de la chaîne stockée dans l'index 1 du tableau `values`, et utilise la fonction `float()` pour les convertir en un nombre et le ranger dans la variable `temperature`.
- L'humidité subit le même traitement en utilisant `float(values[0][0:5])`
- Si l'une de ces deux opérations échoue, le bloc `try` se termine et un message d'erreur est envoyé à la console. Une conversion peut échouer si la charge utile n'est pas formatée correctement par l'Arduino, ou si elle est altérée pendant l'émission ou la réception. Des interférences peuvent provoquer une telle altération.

Prenez tout le temps nécessaire pour vous familiariser avec ce code. Lorsque vous êtes prêt, copiez-le dans votre répertoire applicatif. Utilisez Vim pour créer un nouveau fichier nommé «rf24_receiver.py». Copiez le code dans le tampon de `Vim` depuis le dépôt du projet [14].

Avant de pouvoir tester les communications RF24, vous devez compiler les pilotes en langage C et les encapsulations Python de RF24 et RF24Network. Vous le ferez dans le prochain chapitre. En attendant, voici à quoi ressemble l'exécution du script étudié (**fig. 4**). J'ai annoté cette capture d'écran pour que vous puissiez voir les impressions intégrées dans le script. Passons au chapitre suivant avec la configuration des pilotes RF24 et RF24Network.
Une fois que vous avez assemblé le circuit, passez au chapitre suivant pour mettre en place le script Python de gestion des communications nRF24.

Comment installer les modules Python nRF24 sur le RPi

Dans ce chapitre, nous allons voir comment compiler et installer les pilotes en langage C et les encapsulations Python nécessaires au module nRF24. Deux modules sont concernés : le module principal RF24 et le sous-module RF24Network. Il a plusieurs variantes du projet RF24 ; celle que vous allez utiliser est gérée par TMRh20 [15]. Elle est optimisée pour le module RPi.
Vous pouvez obtenir le code source des deux modules à partir de leurs dépôts Github respectifs :

- Dépôt RF24 : https://github.com/nRF24/RF24 ;
- Dépôt RF24Network : https://github.com/nRF24/RF24Network ;

Pour chaque module, le processus d'installation comprend ces étapes :

- télécharger le code source du module à partir de Github ;
- compiler et installer le pilote C ;
- compiler et installer l'encapsulation Python.

Avant de commencer, assurez-vous que l'interface SPI de votre RPi est activée. Le module nRF24 communique avec le RPI par SPI. Pour le vérifier (et activer le SPI si ce n'est pas déjà fait), connectez-vous à votre RPi et saisissez la commande suivante :

```
sudo raspi-config
```

Utilisez les flèches du clavier et la touche Entrée pour accéder aux options d'interface, puis à SPI. Activez SPI si nécessaire, et quittez raspi-config. Continuez en mettant à jour Raspbian :

```
sudo apt-get update
sudo apt-get upgrade
```

Vous devez télécharger le code source des modules dans votre répertoire de travail. Vous devez également compiler les encapsulations Python avec l'environnement virtuel Python de votre application activé afin qu'elles soient disponibles pour les scripts Python de votre application.
Préparez votre travail avec ces commandes :

```
$ sudo su
# cd /var/www/lab_app/
# . bin/activate
# mkdir rf24libs
# cd rf24libs
```

La compilation et l'installation des deux modules peuvent commencer. Il va y avoir beaucoup de texte affiché sur la console. Mes commandes sont en gras.

Compilation et installation de RF24

Créez un clone du code source de RF24, à partir du git à https://github.com/tmrh20/RF24.

```
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs# git clone
https://github.com/tmrh20/RF24.git RF24
Cloning into 'RF24'...
remote: Enumerating objects: 46, done.
remote: Counting objects: 100% (46/46), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 3545 (delta 16), reused 15 (delta 4),
   pack-reused 3499
Receiving objects: 100% (3545/3545), 1.54 MiB |
   679.00 KiB/s, done.
Resolving deltas: 100% (2119/2119), done.
```

Dans votre répertoire de code source, il y a maintenant un nouveau répertoire : «RF24».

```
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs# ls -al
total 12
drwxr-xr-x 3 root root 4096 May 13 01:38 .
drwxr-xr-x 14 root root 4096 May 13 01:39 ..
drwxr-xr-x 9 root root 4096 May 13 01:38 RF24
```

Passez sous le répertoire RF24 et installez le pilote :

```
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs# cd RF24/
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24# make install
[Running configure]
[SECTION] Detecting arm compilation environment.
[OK] arm-linux-gnueabihf-gcc detected.
[OK] arm-linux-gnueabihf-g++ detected.
… (omitted output)…
[Installing Libs to /usr/local/lib]
[Installing Headers to /usr/local/include/RF24]
```

Le pilote C est installé. Passons à l'encapsulation Python. Cette opération peut prendre plusieurs minutes sur le RPi Zero W, plus lent.

```
(lab_app) root@rpi4:/var/www/lab_app/rf24libs/RF24/
   pyRF24# cd pyRF24/
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24/pyRF24#
python setup.py install
running install
running bdist_egg
running egg_info
creating RF24.egg-info
… (omitted output)…
Installed /var/www/lab_app/lib/python3.8/
   site-packages/RF24-1.3.4-py3.8-
linux-armv6l.egg
Processing dependencies for RF24==1.3.4
Finished processing dependencies for RF24==1.3.4
```

Le module RF24 est installé, et l'encapsulation Python est prête à être utilisée. Poursuivons avec RF24Network.

```
RF24Network
```

Retournez à la racine du répertoire rf23libs et créez un clone du code source de RF24Network, depuis https://github.com/nRF24/RF24Network.git.

```
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24/pyRF24# cd
../..
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs# git clone
https://github.com/nRF24/RF24Network.git
Cloning into 'RF24Network'...
remote: Enumerating objects: 2249, done.
remote: Total 2249 (delta 0), reused 0 (delta 0),
   pack-reused 2249
Receiving objects: 100% (2249/2249), 1.60 MiB |
   733.00 KiB/s, done.
Resolving deltas: 100% (1342/1342), done.
```

L'opération de clonage a créé un nouveau répertoire, «RF24Network». Passez sous le nouveau répertoire et compilez le code source :

```
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs# cd RF24Network/
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24Network# make
install
g++ -Wall -fPIC -c RF24Network.cpp
g++ -shared -Wl,-soname,librf24network.so.1 -o
   librf24network.so.1.0
RF24Network.o -lrf24-bcm
[Install]
[Installing Headers]
```

Le pilote est installé. Continuez avec l'encapsulation Python pour le réseau RF24.

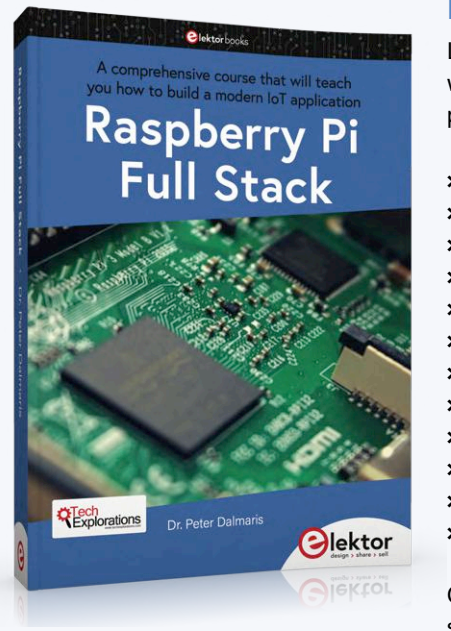
```
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24Network# cd
../RF24/pyRF24/pyRF24Network/
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24/pyRF24/
pyRF24Network# python setup.py install
running install
running build
running build_ext
building 'RF24Network' extension
… (omitted output)…
running install_egg_info
Removing /var/www/lab_app/lib/python3.8/
   site-packages/
RF24Network-1.0-py3.8.egg-info
Writing /var/www/lab_app/lib/python3.8/site-packages/
RF24Network-1.0-py3.8.egg-info
```

Le réseau RF24 en Python est installé. Testons-le.

Test

La façon la plus simple de vérifier que les modules RF24 et RF24Network fonctionnent correctement est d'exécuter les programmes d'exemple Python livrés avec le code source. Vous trouverez ces exemples dans le répertoire `examples` dans `pyRF24Network`. Exécutez l'exemple 'rx' comme ceci :

```
(lab_app)root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24/pyRF24/
pyRF24Network# cd examples/
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24/pyRF24/
pyRF24Network/examples# ls
helloworld_rx.py helloworld_tx.py
(lab_app) root@raspberrypi-zero:/var/www/lab_app/
   rf24libs/RF24/pyRF24/
pyRF24Network/examples# python helloworld_rx.py
================ SPI Configuration ================
CSN Pin = CE0 (PI Hardware Driven)
CE Pin = Custom GPIO22
Clock Speed = 8 Mhz
================ NRF Configuration ================
STATUS = 0x0e RX_DR=0 TX_DS=0 MAX_RT=0 RX_P_NO=7
```



LIVRE RASPBERRY PI FULL STACK

Le livre *Raspberry Pi Full Stack* est un tour d'horizon du développement d'applications web sur le Raspberry Pi. Il montre comment réaliser une application de A à Z, et vous permet d'acquérir expérience et savoir-faire techniques sur des sujets tels que :

- le système d'exploitation Linux et son langage de commande
- le langage de programmation Python
- les broches d'entrée-sortie à usage général (GPIO) du RPi
- le serveur web Nginx
- le micro framework d'applications web Flask Python
- JQuery et CSS pour la création d'interfaces utilisateurs
- la gestion des fuseaux horaires
- créer des tableaux avec Plotly et Google Charts
- enregistrement des données avec Google Sheet
- développer des applets avec IFTTT
- sécuriser votre application avec SSL
- recevoir des notifications par SMS sur votre ordiphone en utilisant Twilio

Ce livre vous apprendra également comment installer un nœud de capteurs Arduino sans fil et en recueillir les données. Votre application web RPi sera capable de traiter les données de nœuds Arduino de la même manière qu'elle traite les données de son capteur embarqué.

Avec le livre *Raspberry Pi Full Stack* vous acquerrez de nombreuses compétences essentielles à la réalisation d'applications Web et Internet des objets. L'application que vous construirez dans le cadre de ce projet est une plateforme que vous pouvez étendre. Ce n'est qu'un début de ce que vous pouvez faire avec un RPi et les composants logiciels et matériels utilisés. L'auteur offre aux lecteurs de son livre un espace de discussion en ligne.

- Livre imprimé : www.elektor.fr/raspberry-pi-full-stack
- E-book : www.elektor.fr/raspberry-pi-full-stack-e-book

LIENS

[1] **Fichiers Gerber pour le HAT RF24** : https://techexplorations.com/parts/rpifs-parts/

[2] **HAT PCB (sans les composants) de PCBWay** : www.pcbway.com/project/shareproject/Raspberry_Pi_Full_Stack_RF24_and_DHT22_HAT.html

[3] **Version haute résolution du schéma** : https://techexplorations.com/parts/rpifs-parts/

[4] **Script rf24_receiver.py** : https://github.com/futureshocked/RaspberryPiFullStack_Raspbian/blob/master/rf24_receiver.py

[5] **Voir la ligne 137 de RF24.h** : https://github.com/nRF24/RF24/blob/master/RF24.h#L137

[6] **Voir la ligne 373 de RF24Network.h** : https://github.com/nRF24/RF24Network/blob/master/RF24Network.h#L373

[7] **Pour en savoir plus sur les « expressions lambda »** : https://docs.python.org/3/tutorial/controlflow.html#lambda-expressions

[8] **Pour en savoir plus sur 'int()'** : https://docs.python.org/3/library/functions.html#int

[9] **Informations complémentaires** : https://github.com/nRF24/RF24Network/blob/master/RF24Network.h#L413

[10] **Informations complémentaires** : https://github.com/nRF24/RF24Network/blob/master/RF24Network.h#L467

[11] **Pour en savoir plus sur 'decode()'** : https://docs.python.org/3/library/stdtypes.html#bytes.decode

[12] **Pour en savoir plus sur 'split()'** : https://docs.python.org/3/library/stdtypes.html#str.split

[13] **Pour en savoir plus sur 'float()'** : https://docs.python.org/3/library/functions.html#float

[14] **Mise à jour de rf24_receiver.py** : https://github.com/futureshocked/RaspberryPiFullStack_Raspbian/blob/master/rf24_receiver.py

[15] **La page du projet** : https://tmrh20.github.io/RF24/

```
   TX_FULL=0
RX_ADDR_P0-1 = 0xcccccccccc3 0xcccccccc3c
RX_ADDR_P2-5 = 0x33 0xce 0x3e 0xe3
TX_ADDR = 0xe7e7e7e7e7
RX_PW_P0-6 = 0x20 0x200x200x200x200x20
EN_AA = 0x3e
EN_RXADDR = 0x3f
RF_CH = 0x5a
RF_SETUP = 0x07
CONFIG = 0x0f
DYNPD/FEATURE = 0x3f 0x04
Data Rate = 1MBPS
Model = nRF24L01+
CRC Length = 16 bits
PA Power = PA_MAX
```

Notez que la sortie contient des statistiques du module RF24 au moment de son démarrage. Elle contient des valeurs valides pour le RX, une adresse (c'est-à-dire des valeurs non nulles) ainsi que le reste de la configuration nRF. Si les modules RF24 et RF24Network n'étaient pas installés correctement, ou si le RPi ne pouvait pas communiquer avec le module nRF24 via SPI, vous obtiendriez soit un message d'erreur du programme exemple (il ne démarrerait pas du tout), soit des valeurs de configuration vides. Maintenant que les modules nRF24 sont installés, le script Python que vous avez installé va pouvoir fonctionner.

200517-03

L'auteur



Peter Dalmaris est enseignant, ingénieur et passionné d'électronique. Il a donné plusieurs cours en ligne sur l'électronique et écrit plusieurs livres techniques. En tant que *Chief Tech Explorer* depuis 2013 au sein de *Tech Explorations*, la société qu'il a fondée à Sydney, en Australie, Peter s'est donné pour mission d'explorer la technologie et de contribuer à la diffusion du savoir dans le monde.

Ont contribué à cet article

Auteur : **Peter Dalmaris**
Illustrations : **Peter Dalmaris**
Rédaction : **Jan Buiting**
Maquette : **Giel Dols**
Traduction : **Helmut Müller**

Votre avis, s'il vous plaît ?

Vous pouvez envoyer un courriel à l'auteur peter@txplore.com avec vos questions ou vos commentaires sur le livre.

Publicité

nouveau livre elektor



Initiation au langage CircuitPython et à la puce nRF52840

- **Python** est apprécié par les pédagogues… et par les informaticiens expérimentés.
- **CircuitPython** est une version spéciale de Python pour microcontrôleurs à 32 bits.
- Entrez directement sur le terrain avec des projets qui mettent en œuvre les cartes **Feather Sense** et **Clue Express** *d'Adafruit* et différents périphériques.
- Plus d'une quarantaine d'exemples et de montages pour découvrir la richesse de CircuitPython.

- Préface de ***ladyada***, fondatrice et PDG d'Adafruit
- Plus d'une quarantaine d'exemples et de montages pour découvrir la richesse de CircuitPython et de cartes avec puce nRF52840
- Véritable boîte à outils pour concrétiser vos propres projets (professeurs, lycéens, étudiants, *makers*
- Moins de 100 lignes de code par projet

Michaël Bottin recherche l'équilibre entre l'électronique avec fer à souder (!) et la programmation en C et en Python. Il est enseignant à l'IUT de Rennes où il transmet sa passion à ses étudiants en privilégiant un apprentissage par projet.



Sommaire et extraits : www.elektor.fr/19523

ISBN 978-2-86661-211-5, 552 pages, couleur

multitâche en pratique avec l'ESP32 (6)

Groupes d'événements

Warren Gay (Canada)

La synchronisation entre plusieurs tâches est souvent nécessaire lors de la conception d'applications avec *FreeRTOS*. Dans les précédents épisodes de cette série, nous avons examiné les notifications de sémaphores et de tâches. Cependant, celles-ci ne sont capables d'envoyer un événement que d'une tâche ou d'une routine d'interruption de service (ISR) à une seule tâche de réception. Lorsque vous avez besoin de diffuser un événement à plusieurs tâches, la capacité de groupe d'événements est la solution dont vous avez besoin.

Commençons par examiner un cas d'utilisation potentiel. Une boîte à rythmes MIDI doit produire quatre sons simultanés en réponse à une commande reçue par son canal série. Idéalement, toutes les tâches d'émission d'un son devraient dans ce cas commencer exactement en même temps pour éviter que l'oreille ne distingue plusieurs attaques décalées. La coordination pourrait être assurée par l'utilisation de quatre sémaphores distincts et en se fiant à la vitesse de l'unité centrale pour une livraison en temps voulu. Ce serait maladroit, surtout si le nombre de tâches devait augmenter. Les groupes d'événements sont la méthode d'implémentation préférée dans *FreeRTOS* pour coordonner des tâches multiples à partir d'une source unique.

Drapeaux des événements

FreeRTOS définit 24 drapeaux d'événement pour chaque objet de groupe d'événements créé (**fig. 1**). Ces drapeaux sont représentés dans le type de données C `EventBits_t`, un type de données de 32 bits dont 24 bits sont disponibles. Le bit 0 est le bit de poids le plus faible (LSB) des drapeaux disponibles. Les 8 bits de poids le plus fort sont *réservés* à l'usage interne de *FreeRTOS*.

La manière dont ces 24 bits sont attribués et utilisés par l'application est laissée à l'appréciation du programmeur. Dans cette démo, la tâche `loop()` produit un événement toutes les secondes qui se traduit par le démarrage synchrone de deux tâches avec des clignotements différents via les LED associées. Le bit 0 du groupe d'événements (valeur `0b0001`/valeur décimale 1) notifie une tâche nommée `blink2()` qui fait clignoter deux fois la LED1. Le bit 1 du groupe d'événements (valeur `0b0010`/valeur décimale 2) est affecté à la notification d'une tâche nommée `blink3()` qui fait clignoter trois fois la LED2.

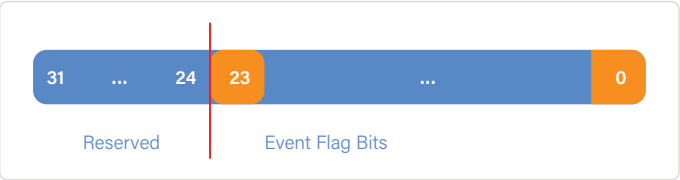


Figure 1. Drapeaux d'événement dans le type de données C EventBits_t.

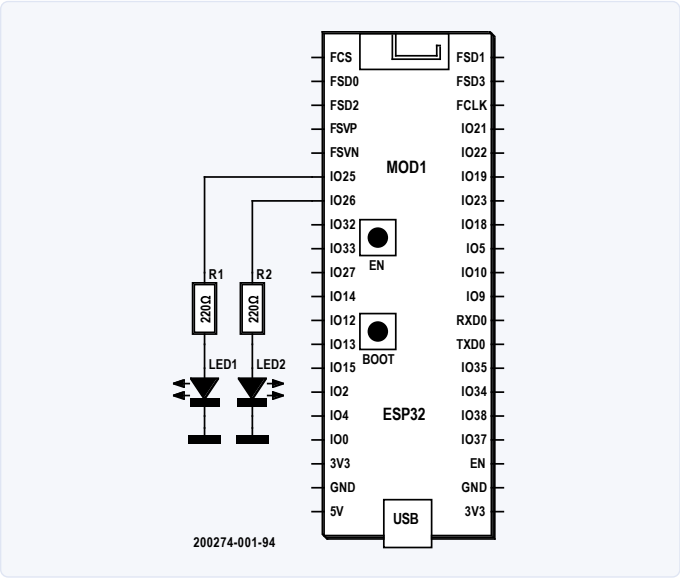


Figure 2. Le schéma du programme de démonstration evtgrp.ino.

Programme de démonstration

Avant de nous plonger dans le code, passons en revue ce que le programme de démonstration espère accomplir. Deux LED sont pilotées par les broches 25 et 26 du GPIO (lignes 5 et 6) dans la configuration haute active (**fig. 2**). Ces GPIO sont configurés dans la fonction `setup()` en tant que sorties (lignes 62 à 65). La fonction task `blink2()` commande la LED1 en la faisant clignoter deux fois avant d'attendre le prochain événement (lignes 19 à 25). Task `blink3()` commande la LED2 en la faisant clignoter trois fois avant d'attendre le prochain événement (lignes 41 à 47). Les deux tâches sont identiques, sauf pour le nombre de fois que la boucle de clignotement est exécutée (lignes 27 et 49) et les retards utilisés.

Liste 1. Le programme Arduino evtgrp.ino utilisant des groupes d'événements. [1]

```
0001: // evtgrp.ino
0002: // MIT License (see file LICENSE)
0003:
0004: // LED is active high
0005: #define GPIO_LED1     25
0006: #define GPIO_LED2     26
0007:
0008: #define EVBLK2        0b0001
0009: #define EVBLK3        0b0010
0010: #define EVALL         0b0011
0011:
0012: static EventGroupHandle_t hevt;
0013:
0014: void blink2(void *arg) {
0015:
0016:   for (;;) {
0017:     // Call blocks until EVBLK2 bit set,
0018:     // and same bit is cleared upon return
0019:     xEventGroupWaitBits(
0020:       hevt,
0021:       EVBLK2,
0022:       pdTRUE,
0023:       pdFALSE,
0024:       portMAX_DELAY
0025:     );
0026:     // Blink 2 times
0027:     for ( int x=0; x < 2; ++x ) {
0028:       digitalWrite(GPIO_LED1,HIGH);
0029:       delay(120);
0030:       digitalWrite(GPIO_LED1,LOW);
0031:       delay(120);
0032:     }
0033:   }
0034: }
0035:
0036: void blink3(void *arg) {
0037:
0038:   for (;;) {
0039:     // Call blocks until EVBLK3 bit set,
0040:     // and same bit is cleared upon return
0041:     xEventGroupWaitBits(
0042:       hevt,
0043:       EVBLK3,
0044:       pdTRUE,
0045:       pdFALSE,
0046:       portMAX_DELAY
0047:     );
0048:     // Blink 3 times
0049:     for ( int x=0; x < 3; ++x ) {
0050:       digitalWrite(GPIO_LED2,HIGH);
0051:       delay(75);
0052:       digitalWrite(GPIO_LED2,LOW);
0053:       delay(75);
0054:     }
0055:   }
0056: }
0057:
0058: void setup() {
0059:   int app_cpu = xPortGetCoreID();
0060:   BaseType_t rc;
0061:
0062:   pinMode(GPIO_LED1,OUTPUT);
0063:   pinMode(GPIO_LED2,OUTPUT);
0064:   digitalWrite(GPIO_LED1,LOW);
0065:   digitalWrite(GPIO_LED2,LOW);
0066:
0067:   delay(2000);
0068:
0069:   hevt = xEventGroupCreate();
0070:   assert(hevt);
0071:
0072:   rc = xTaskCreatePinnedToCore(
0073:     blink2,   // func
0074:     "blink2", // name
0075:     1024,     // stack bytes
0076:     nullptr,  // arg ptr
0077:     1,        // priority
0078:     nullptr,  // ptr to task handle
0079:     app_cpu   // cpu#
0080:   );
0081:   assert(rc == pdPASS);
0082:
0083:   rc = xTaskCreatePinnedToCore(
0084:     blink3,   // func
0085:     "blink3", // name
0086:     1024,     // stack bytes
0087:     nullptr,  // arg ptr
0088:     1,        // priority
0089:     nullptr,  // ptr to task handle
0090:     app_cpu   // cpu#
0091:   );
0092:   assert(rc == pdPASS);
0093: }
0094:
0095: void loop() {
0096:
0097:   delay(1000);
0098:   xEventGroupSetBits(
0099:     hevt,
0100:     EVBLK2|EVBLK3
0101:   );
0102: }
0103:
0104: // End evtgrp.ino
```

Synchronisation

La synchronisation est réalisée en faisant bloquer les fonctions de tâche dans un appel à `xEventGroupWaitBits()`. Jusqu'à ce que l'événement soit déclenché, l'exécution est bloquée dans cet appel de fonction. Une fois que le groupe d'événements est notifié, l'exécution des tâches bloquées reprend en revenant de la fonction appelée. L'événement est déclenché toutes les secondes par un appel correspondant à `xEventGroupSetBits()` dans la tâche `loop()` (voir les lignes 98 à 101 de la **liste 1** [1]).

Création de groupes d'événements

Le groupe d'événements est créé en appelant `xEventGroupCreate()` (ligne 69). Il n'y a pas d'arguments à fournir et la fonction renvoie une poignée (ou *handle*) à l'objet de groupe d'événements alloué. Le type de données renvoyé est `EventGroupHandle_t`. Si la valeur retournée est égale à zéro, votre pile est à court de mémoire (d'où le contrôle d'assertion à la ligne 70).

```
0069:   hevt = xEventGroupCreate();
0070:   assert(hevt);
```

Notification

Il existe différents choix pour manipuler les groupes d'événements. Dans cet article, nous utilisons une des fonctions les plus simples appelée `xEventGroupSetBits()`.

```
EventBits_t xEventGroupSetBits(
  EventGroupHandle_t xEventGroup,    // Handle
  const EventBits_t uxBitsToSet      // Bits to set
);
```

La poignée `xEventGroup` spécifie le groupe d'événements à exploiter, tandis que l'argument `uxBitsToSet` spécifie les bits d'événements que vous souhaitez définir de manière atomique**. La valeur retournée sera les bits d'événement qui étaient en vigueur *après* l'appel à `xEventGroupSetBits()`. Cependant, sachez que la valeur retournée peut ne pas toujours inclure les bits que vous venez de définir car les tâches de réception peuvent les avoir déjà effacés à la réception. La tâche `loop()` appelle la fonction `xEventGroupSetBits()` en fixant les bits 0 et 1 à l'aide de la macro expression `EVBLK2|EVBLK3`.

```
0008: #define EVBLK2        0b0001
0009: #define EVBLK3        0b0010

0098:   xEventGroupSetBits(
0099:     hevt,
0100:     EVBLK2|EVBLK3
0101:   );
```

De cet appel, on peut voir que les bits 0 et 1 sont réglés atomiquement** par cet appel.

La tâche loop()

La tâche `loop()` retarde d'une seconde (ligne 97) et envoie ensuite une notification d'événement (lignes 98 à 101). Ensuite, suivant la tradition d'Arduino, la fonction `loop()` quitte et se répète. Les événements sont alors déclenchés à environ une seconde d'intervalle.

Réception des tâches

Les tâches `blink2()` et `blink3()` sont notifiées par le groupe d'événements à l'aide de l'identifiant enregistré (lignes 12 et 69). Par exemple, `blink2()` voit son exécution suspendue aux lignes 19 à 25 jusqu'à ce qu'un événement soit envoyé. De même, l'exécution de `blink3()` `est suspendue aux` lignes 41 à 47 jusqu'à ce qu'un événement soit envoyé. Comme les événements sont déclenchés atomiquement** par les lignes 98 à 101, les deux tâches reprendront en même temps. Si les deux tâches sont assignées à la même unité centrale, elles seront alors programmées pour s'exécuter l'une après l'autre. Si les deux tâches sont assignées à des CPU différentes, il est alors possible qu'elles reprennent et s'exécutent simultanément.
Une fois que l'exécution reprend dans chaque tâche, ils peuvent alors faire clignoter leur LED de manière unique. Une fois ce clignotement terminé, l'exécution reprend au sommet de la boucle de tâches en attendant l'arrivée de l'événement suivant.

Compensation des événements

L'appel de la fonction `xEventGroupWaitBits()` est délicat et la flexibilité qu'il offre ajoute de la complexité. Décomposons le tout :

```
EventBits_t xEventGroupWaitBits(
  const EventGroupHandle_t xEventGroup, // Event group handle
  const EventBits_t uxBitsToWaitFor,
  const BaseType_t xClearOnExit,
  const BaseType_t xWaitForAllBits,
  TickType_t xTicksToWait
);
```

Le premier argument est la poignée du groupe d'événements référencés. C'est assez simple. Le deuxième argument spécifie les bits d'événement que la fonction veut attendre. Dans le programme de démonstration, nous avons spécifié la macro `EVBLK2` (ligne 21) pour la tâche `blink2()` et la macro `EVBLK3` pour la tâche `blink3()`. La raison de l'utilisation de bits séparés apparaîtra bientôt.
Le troisième argument `xClearOnExit` est une version en langage C d'une valeur booléenne (`pdTRUE` a été fourni aux lignes 22 et 44). Cela informe la fonction d'effacer les bits d'événement reçus avant de revenir. Ainsi, la tâche `blink2()` attend que le bit 0 se mette en place (macro `EVBLK2`), et le troisième argument `pdTRUE` l'informe d'effacer ce bit à la réception et au retour. Cette opération d'attente et d'effacement est effectuée de manière atomique**.
Dans notre cas particulier, l'argument quatre est académique mais examinons-le. L'argument `xWaitForAllBits` est également booléen et précise quand la fonction doit revenir :

- lorsque l'*un* des bits attendus est activé (`pdFALSE`), *ou*
- seulement une fois que *tous les* bits attendus sont réglés (`pdTRUE`).

Notre exemple n'attend qu'un seul bit (le bit 0 pour `blink2()` et le bit 1 pour `blink3()`), ce qui signifie que l'indication d'un ou de tous n'a pas d'effet matériel. Cependant, si vous souhaitez attendre une combinaison de bits, cette possibilité peut être très utile.

Contrôle des événements

La raison pour laquelle il était nécessaire de disposer de bits d'événements séparés peut maintenant être évidente. Chaque tâche de réception attend et efface ensuite son propre bit d'événement par un

*** Ces opérations dites atomiques assurent une synchronisation non bloquante pour résoudre les problèmes de performance causés par des opérations asynchrones qui agissent sur la mémoire partagée.*

LIENS

[1] **Code pour evtgrp.ino** : https://bit.ly/35YrjCN
[2] **Livre : *FreeRTOS for ESP32-Arduino*, W. Gay, Elektor 2020** : https://bit.ly/2U2Yhg1
[3] **Documentation FreeRTOS** : https://bit.ly/386HZL6

appel à `xEventGroupWaitBits()`. Il doit donc y avoir un bit séparé pour chaque tâche. Dans la fonction `loop()`, les deux tâches sont notifiées simultanément en réglant leurs bits d'événement spécifiques en même temps (ligne 100).
Comme on peut le voir, les bits du groupe d'événements peuvent être utilisés pour créer un code d'application très complexe et créatif. Pour plus de détails sur les fonctions supplémentaires, voir le livre *FreeRTOS for ESP32-Arduino* [2] et la documentation de *FreeRTOS* [3].

Lancer la démo

La démo n'utilise pas le moniteur sériel et peut être exécutée sur n'importe quel ESP32 qui rend disponible les GPIO 25 et 26 (ou changez les lignes 5 et 6 de la **liste 1** pour utiliser d'autres sorties). Faites clignoter le programme *evtgrp.ino* dans l'ESP32 avec deux LED et des résistances de 220 Ω comme sur la **fig. 2**. Une fois que l'ESP32 commence l'exécution, les LED devraient clignoter - une fois par seconde, la LED1 devrait clignoter deux fois et ensuite attendre tandis que la LED2 devrait clignoter trois fois (rapidement) puis attendre. Le processus se répétera toutes les secondes.

Synchronisation totale

Cette démo illustre comment deux tâches complètement indépendantes, `blink2()` et `blink3()`, peuvent être synchronisées en utilisant un seul objet de groupe d'événements. La tâche `loop()` diffuse son événement des lignes 98 à 101. Cette approche peut être étendue jusqu'à 24 tâches si nécessaire (la limite est définie par le nombre de drapeaux d'événement disponibles dans un seul groupe d'événements). Cet article a montré une façon d'utiliser les groupes d'événements, mais il existe bien d'autres façons plus subtiles de coordonner des événements en utilisant d'autres fonctions de *FreeRTOS*.

200528-03

Votre avis, s'il vous plaît
Vos questions et vos commentaires sont bienvenus chez l'auteur ve3wwg@gmail.com en anglais, ou à la rédaction redaction@elektor.fr en français.

Ont contribué à cet article

Auteur : **Warren Gay**
Schéma : **Patrick Wielders**
Illustration : **Jack Jamar**
Rédaction : **Stuart Cording**
Maquette : **Giel Dols**
Traduction : **Léo Helba**

PRODUITS

> **Livre : *FreeRTOS for ESP32-Arduino*, W. Gay, Elektor 2020**
https://bit.ly/2U2Yhg1

Publicité



analyseur de puissance multicanal

Avec affichage graphique et alphanumérique de la puissance sur 3 canaux

Wil Dijkman (Pays-Bas)

C'est une chose que de mesurer la tension, le courant et la puissance d'appareils alimentés en courant continu, c'en est une autre, plus ardue, en courant alternatif, en raison de la forme d'onde et du déphasage entre tension et courant. Cet instrument ne se contente pas de mesurer et de calculer des quantités, mais affiche également les formes d'onde et le spectre des signaux alternatifs sur un LCD graphique.

Ce projet a été inspiré par le wattmètre CA/CC publié dans Elektor en septembre 2015 [1]. À première vue, il m'a plu, et j'ai souhaité le construire moi-même, mais je lui ai trouvé des inconvénients qui m'ont conduit à essayer de l'améliorer. Voici comment :

- circuit d'entrée : je souhaite mesurer le courant et la tension indépendamment l'un de l'autre : le courant à travers les lignes d'alimentation ne doit pas influencer la lecture de la tension.
- échantillonnage : au lieu de me contenter des huit premières harmoniques d'un signal de 50 Hz (avec un filtre parfait), j'augmente la fréquence d'échantillonnage pour mesurer les 40 premières harmoniques.
- changement de calibre et correction de décalage (*offset*) automatiques

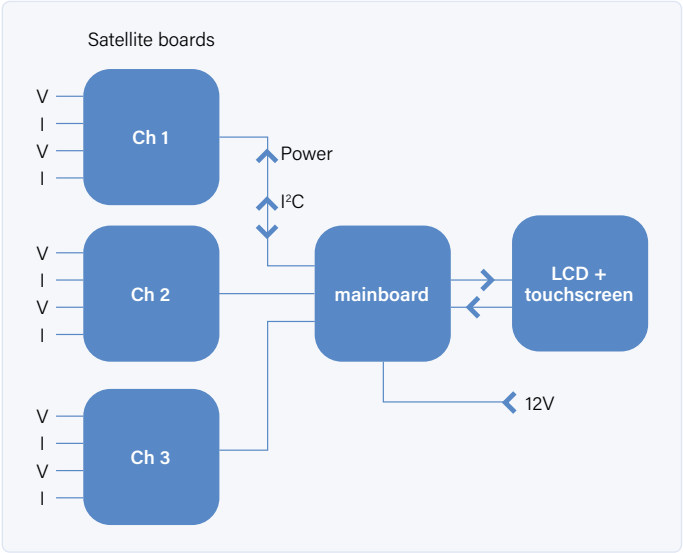


Figure 1. Schéma de l'analyseur de puissance.

Cela implique une révision totale du circuit d'entrée et d'amplification. Côté chaud, il faut un microcontrôleur pour la correction automatique du décalage et pour l'autoréglage. La **figure 1** montre la carte principale et (jusqu'à) trois cartes satellites.

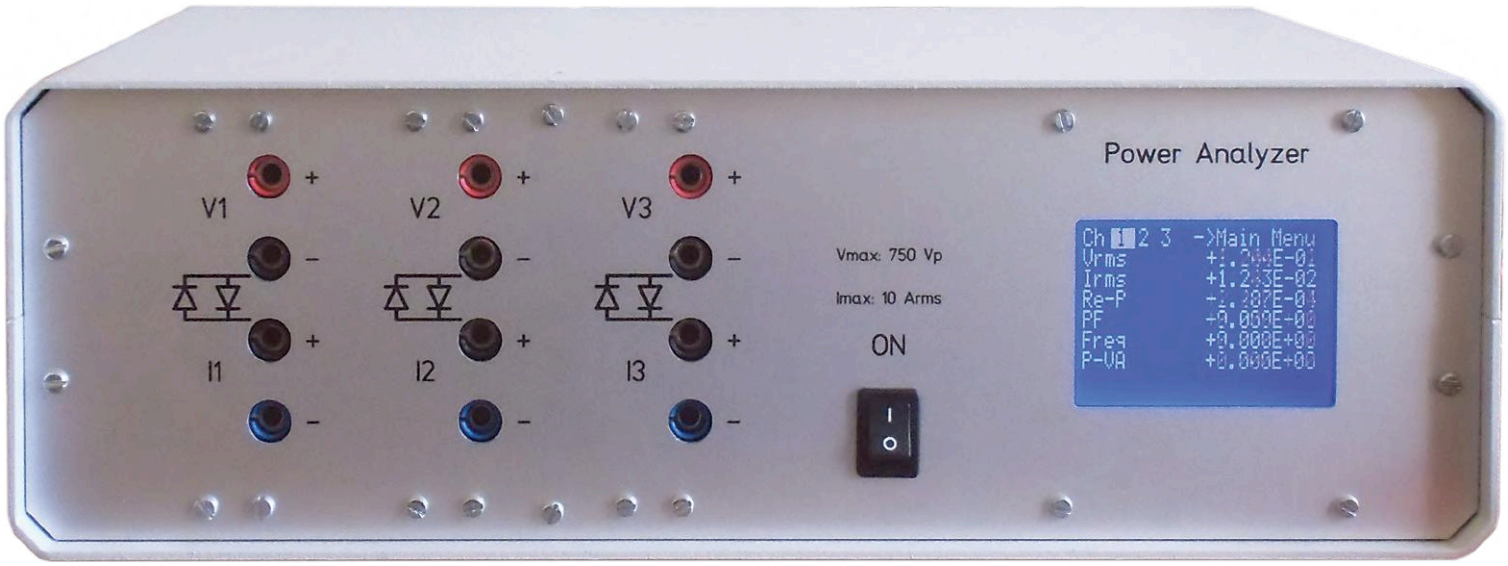
Carte principale

Le schéma (**fig. 2**) est en grande partie une copie de la carte EasyPIC V7, avec comme interface utilisateur un afficheur LCD graphique tactile. Le générateur d'ondes carrées (autour d'IC1) alimente les cartes satellites avec un signal de 12 $V_{càc}$ d'environ 150 kHz. La communication des données avec les cartes satellites passe par une interface I²C, où le µC de la carte principale est le maître et les cartes satellites sont les esclaves.

Selon que le point de soudure SJ1 est présent ou pas, la carte principale accepte une alimentation de 12 V stabilisés ou de 15 à 18 V non stabilisés. Les deux autres points de soudure, MAX1 et MAX2, permettent d'indiquer le nombre de canaux pour le maître : MAX1 court-circuité = un seul canal, MAX2 court-circuité = deux canaux, MAX1 et MAX2 court-circuités = trois canaux (cartes satellites). Grâce aux cavaliers marqués Jx sur les schémas et sur le circuit imprimé principal, on peut se contenter d'un circuit imprimé à une seule couche.

Carte satellite

Sur les cartes satellites (**fig. 3**), je souhaite mesurer indépendamment la tension et le courant. Ce n'est réalisable qu'en partie : l'entrée V basse et l'entrée I peuvent flotter à +/–1 V l'une par rapport à l'autre, grâce aux diodes D3, D4, D7 et D8. En l'absence de précautions, ce circuit pourrait être endommagé par une mauvaise connexion : un potentiel élevé à l'entrée basse tension et aussi l'entrée de courant du côté basse tension. Pour prévenir ce risque, j'ai rajouté le circuit T10 à T14 et le relais Reed K1. Une première limitation de courant est assurée par R11 : sous une tension d'entrée de 750 V, l'intensité maximale du courant est de 0,5 A. Quand le courant à travers R11 dépasse 10 à 20 mA, il circule à travers R34 un faible courant est détecté par T11 et T13. Cela déclenche le circuit du thyristor avec T10 et T12. Une fois



amorcé, ce circuit à thyristor reste dans cet état et coupe T14 et K1. Le signal OC indique au µC que la protection est déclenchée. En l'absence de courant, K1 est également désactivé (V-low est déconnecté), de sorte que le circuit est en sécurité. Pour le remettre en fonction, il faut déconnecter l'alimentation du circuit puis la reconnecter.

Le diviseur de tension donne 100 mV quand la tension d'entrée atteint le maximum de 750 V (CC), la tension alternative maximale étant d'environ 500 V. La chute de tension sur R62, la résistance de détection de courant de 6,5 mΩ, est d'environ 100 mV à 15 A CC (environ 10 A CA). Les amplificateurs de tension et de courant sont identiques, je ne décrirai donc que l'amplificateur de tension. T1, T2, T4 et T6 servent d'interrupteurs pour relier l'amplificateur différentiel soit au signal du circuit d'entrée, soit pour court-circuiter l'entrée afin de rendre possible la mesure du décalage (***offset***) dont la valeur pourra être mémorisée. Ce décalage pourra être soustrait ultérieurement pour affiner la mesure. T3 et T5 permettent de choisir entre 1x ou 25x pour le gain du premier étage de l'amplificateur d'instrumentation ordinaire. IC5 rend asymétrique le signal différentiel. Le gain du dernier étage avec IC4 sera de 1x ou 5x. De sorte qu'il y a quatre gains possibles : 1x, 5x, 25x ou 125x. C'est le microcontrôleur IC13 qui fixe le gain. L'amplificateur délivre un signal positif ou négatif par rapport à COMV. Ce signal est 0,5 fois VREF, délivré par le circuit autour d'IC3. VREF est de 4,5 V, donc COMV, COMI et CINV sont de 2,25 V. VREF et CINV sont connectés au CA/N du contrôleur PIC18F26K80. Ce convertisseur peut opérer sur 13 bits s'il est utilisé de la manière indiquée (12 bits + 1 bit de signe). Pour 750 V, cela donne une résolution d'environ 200 mV par bit. Pour 6 V, cela donne environ 1,6 mV/bit. Cela me suffit.

Pour fixer l'adresse I²C d'une carte satellite, il faut fermer ou ouvrir SJ1 et SJ2 : SJ1 fermé = canal 1, SJ2 fermé = canal 2, SJ1 et SJ2 fermés = canal 3.

Isolation galvanique

Le signal carré provenant de la carte mère est envoyé au transformateur de droite (rapport 1:1). La tension sur les redresseurs est alors un peu inférieure à 6 V. Les régulateurs en parallèle la mettent +/–5 V.

ATTENTION ! Les tensions élevées sont létales. Le circuit décrit ici ne convient donc pas aux débutants. Ne le construisez et ne l'utilisez que si vous connaissez toutes les règles à appliquer en présence de tensions dangereuses.

Le seuil d'isolation des enroulements du transformateur est de 900 V, donc l'isolation totale sera de 1800 $V_{crête}$, à condition que le reste de la construction soit conforme. Le signal I²C passera par le circuit IC14 dont la valeur d'isolation est de 4 $kV_{crête}$. Les cartes satellites seront donc complètement flottantes à la fois par rapport à la carte principale, et les unes par rapport aux autres.

Mise à jour de la carte satellite

Des tests sur le prototype ont montré que des corrections étaient nécessaires. Il a fallu réduire la diaphonie entre signal de commutation et signal de mesure dans les commutateurs de décalage. C'est la fonction de C41-R102 et C40-R103. Le condensateur C33 réduit les déclenchements intempestifs du circuit de protection, par exemple lors du branchement d'un cordon de mesure. D14 et R101 limitent également la sensibilité pour les signaux positifs.

Avec le gain le plus élevé des amplificateurs (environ 2750x), il faut un contrôle de décalage. Cette commande est désactivée lorsque le gain est plus faible. Selon le signe du décalage, une sélection doit être faite avec le cavalier à souder.

J'avais oublié quelques condensateurs de découplage dans le circuit d'alimentation, il a fallu les ajouter aussi. La liste des composants de ce projet est téléchargeable [2] avec les références de Farnell, ou n'importe quel autre fournisseur.

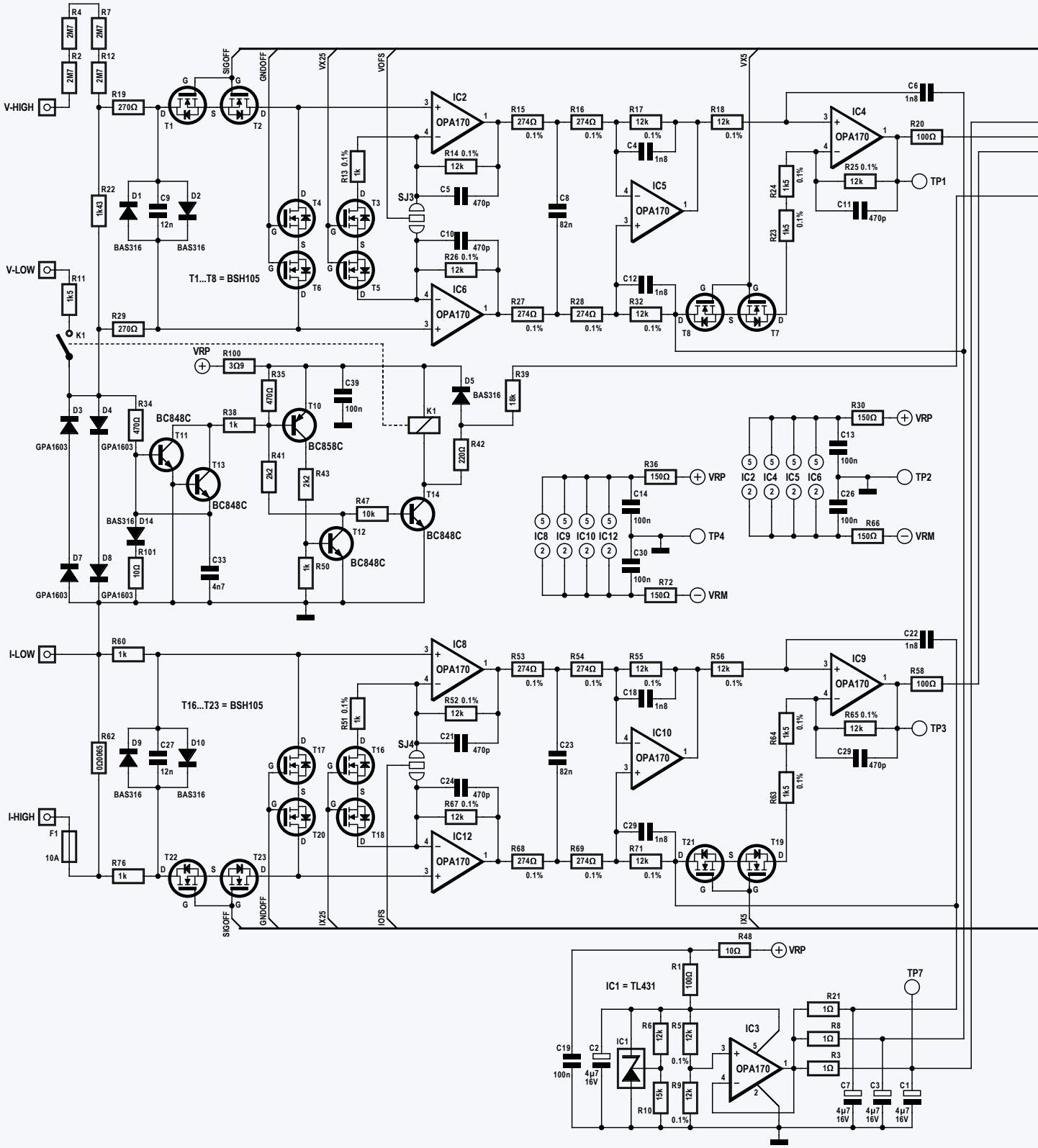
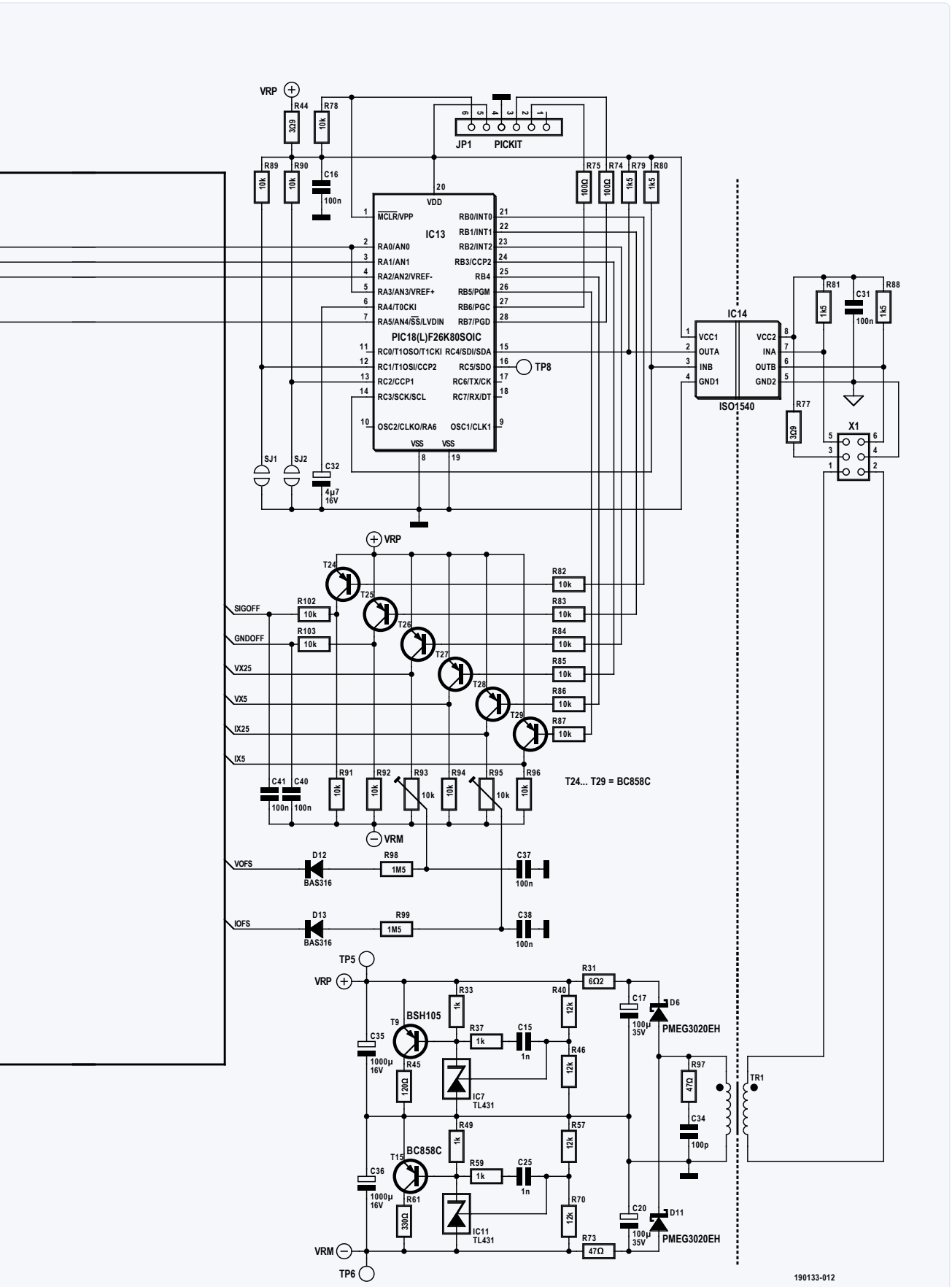


Figure 2. Schéma de la carte mère.


VRP
R44
3Ω9
R78
10k
JP1
PICKIT
R89
R90
C16
100n
R75
R74
R79
R80
100Ω
1k5
IC13
PIC18(L)F26K80SOIC
MCLR/VPP
RA0/AN0
RA1/AN1
RA2/AN2/VREF-
RA3/AN3/VREF+
RA4/T0CKI
RA5/AN4/SS/LVDIN
RC0/T1OSO/T1CKI
RC1/T1OSI/CCP2
RC2/CCP1
RC3/SCK/SCL
OSC2/CLKO/RA6
RB0/INT0
RB1/INT1
RB2/INT2
RB3/CCP2
RB4
RB5/PGM
RB6/PGC
RB7/PGD
RC4/SDI/SDA
RC5/SDO
RC6/TX/CK
RC7/RX/DT
OSC1/CLK1
VDD
VSS
TP8
SJ1
SJ2
C32
4µ7
16V
IC14
ISO1540
VCC1
OUTA
INB
GND1
VCC2
INA
OUTB
GND2
R81
R88
C31
100n
R77
X1
VRP
T24
T25
T26
T27
T28
T29
R82
R83
R84
R85
R86
R87
10k
R102
R103
SIGOFF
GNDOFF
VX25
VX5
IX25
IX5
C41
C40
100n
R91
R92
R93
R94
R95
R96
T24... T29 = BC858C
VRM
VOFS
IOFS
D12
D13
BAS316
R98
R99
1M5
C37
C38
TP5
TP6
R31
6Ω2
R33
1k
R40
12k
C17
100µ
35V
D6
PMEG3020EH
T9
BSH105
C35
1000µ
16V
R37
1k
C15
1n
R45
120Ω
R46
IC7
TL431
R97
47Ω
TR1
C34
100p
R49
R57
T15
BC858C
C36
R59
C25
R61
330Ω
R70
IC11
R73
47Ω
C20
D11
190133-012

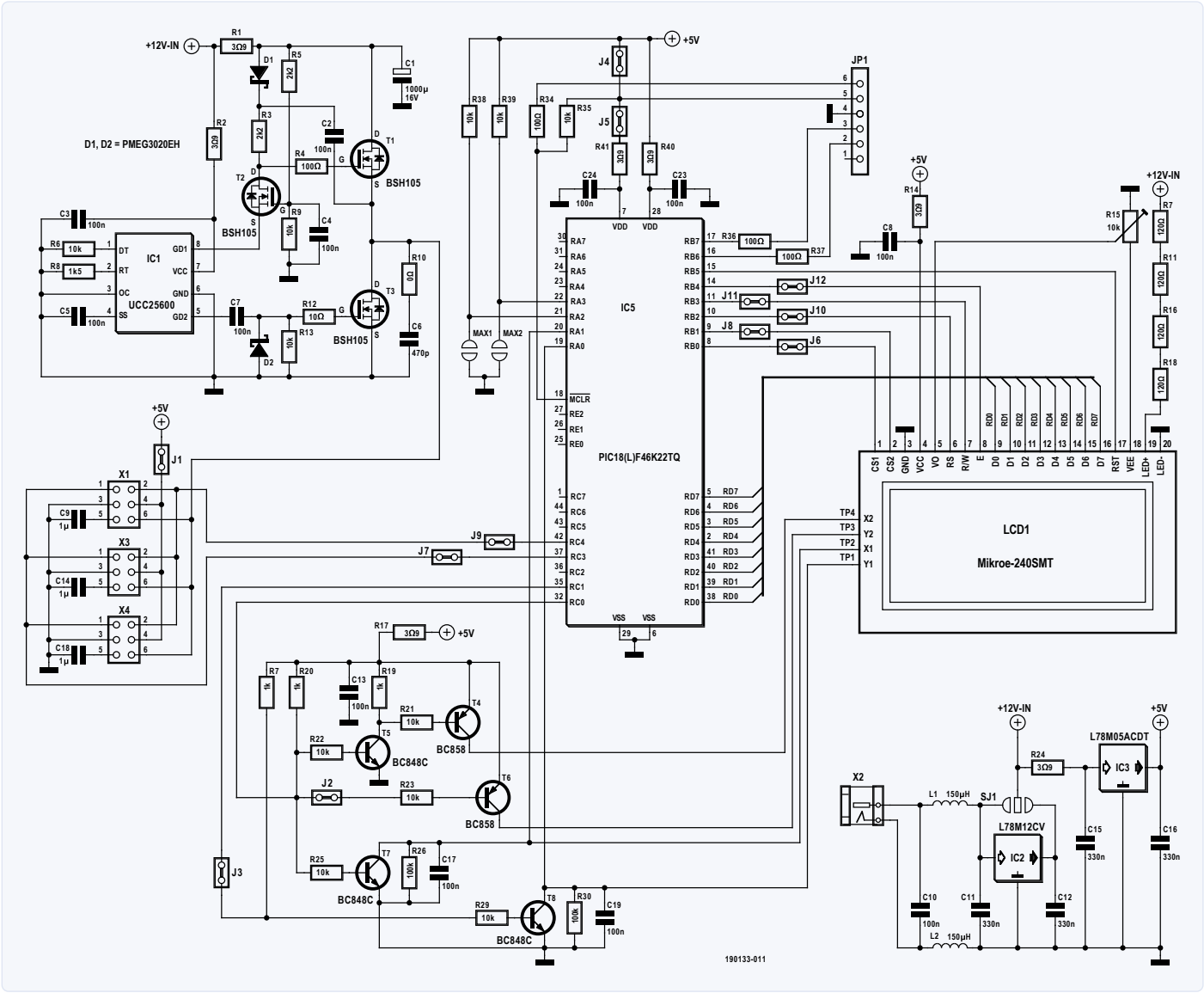


Figure 3. Schéma de la carte satellite.

Logiciel pour la carte satellite

Le programme est écrit en mikroBasic pour PIC (V5.6.1) de MikroElektronika, un langage de programmation relativement facile d'accès.

Je voulais une gamme de fréquences d'un peu moins de 50 Hz à quelques centaines de Hz. Par conséquent, le signal devait être échantillonné pendant au moins 30 ms. À 50 Hz, ça fait au moins trois passages par zéro, assez pour détecter une période. Du moins tant que ces passages à zéro sont équidistants. Avec un rapport cyclique différent de 50 % à 50 Hz, 30 ms est une durée d'échantillonnage insuffisante, et je suis donc passé à 35 ms. Avec un rapport cyclique inférieur à 25 %, il est possible qu'à 50 Hz la détermination de la fréquence soit incertaine. À des fréquences plus élevées, il n'y a pas de problème. Les signaux ayant moins de trois passages par zéro durant la fenêtre d'échantillonnage sont traités comme du CC.

La deuxième question est de savoir combien d'échantillons seront prélevés au cours de ces 35 ms. Une fréquence d'échantillonnage de 10 kHz ou plus serait idéale. J'ai opté pour 357 échantillons à 10,2 Kéch/s. Cela donne un multiple entier d'échantillons pour les signaux de 50 et 60 Hz, fréquences auxquelles sera obtenue la plus grande précision de l'analyse du signal.

Il existe dans mikroBasic une bibliothèque avec des commandes I²C, hélas réservées à un maître I²C. J'ai dû écrire mes propres procédures pour un esclave I²C. Ces procédures sont nécessaires pour établir et contrôler la communication avec la carte mère, le maître I²C. Ce dernier ne fait que demander et afficher les informations des cartes satellites, le calibre (automatique) et le décalage. Tout le traitement du signal, échantillonnage et filtrage compris, est effectué sur la ou les cartes satellites.

Les transformées de Fourier rapides sont également calculées sur les tableaux des satellites. Les calculs sont empruntés [3]. *Le* filtrage numérique est basé sur le chapitre 16 du *The Scientist and*



Figure 4a. La carte principale et l'afficheur en façade.

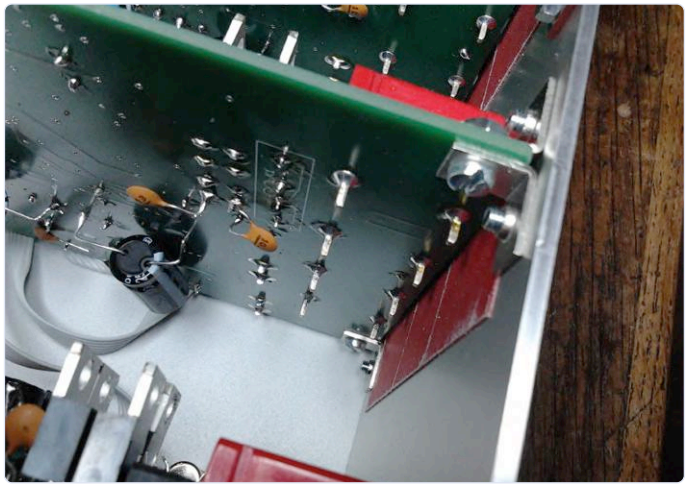


Figure 4b. Une carte satellite à l'avant. Notez l'isolation supplémentaire (ruban rouge épais) entre la carte et la façade.

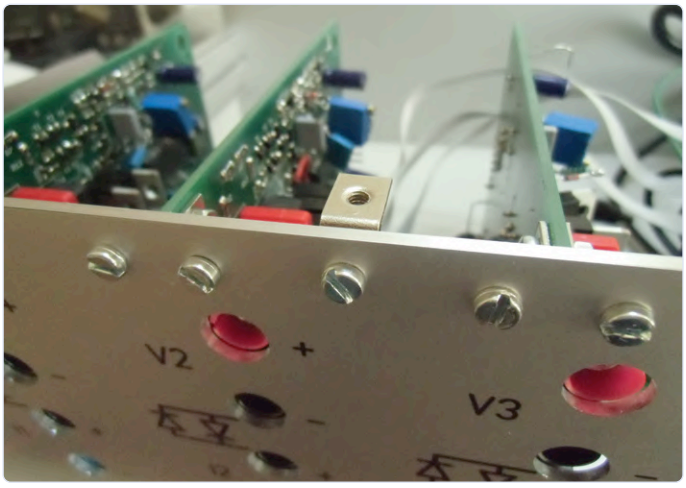


Figure 4c. Une des équerres de renfort entre façade et couvercle.

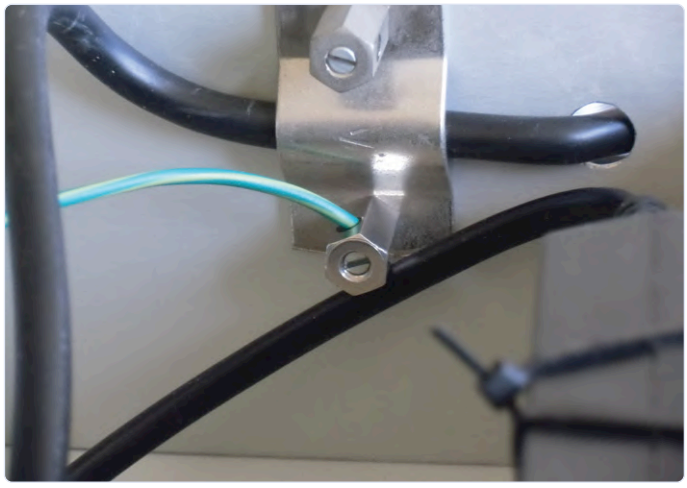


Figure 4d. Connexion PE

Engineer's Guide to Digital Signal Processing [4]. Une fenêtre de Blackman est utilisée et le filtre a 13 coefficients (M=12). Le choix de la longueur du filtre est un compromis entre temps de calcul, mémoire nécessaire et performances (c'est-à-dire la suppression des signaux supérieurs à 2,55 kHz et l'atténuation des signaux inférieurs à 2,55 kHz).

Assemblage

J'ai dessiné une façade pour trois canaux, qui peut être commandée chez *Schaeffer*, pour un boîtier métallique *Metcase* M5503110 (réf. 1510827 chez *Farnell*).

La **figure 4** montre le matériel dans ce boîtier. On y voit l'agencement de mon *Power Meter*. Notez le bloc d'alimentation standard utilisé pour alimenter l'analyseur de puissance.

La **figure 4c** montre l'une des deux équerres ajoutées sous la façade pour la rigidifier, notamment pour lui permettre de résister aux tractions mécaniques quand on branche et débranche les cordons de mesure. Il faut percer un trou dans le couvercle et le fond pour visser ces équerres. Comme le boîtier est métallique, il doit être connecté à la terre du secteur (**fig. 4d**).

Utilisation

Après la mise en marche, l'écran de mesure apparaît. Il affiche quelques unités et leurs valeurs pour le canal sélectionné. En haut, vous pouvez sélectionner d'autres canaux (s'ils existent).

L'écran affiche jusqu'à sept valeurs de mesure, sélectionnées dans l'écran de configuration.

Dans le *menu principal* (en haut à droite), choisissez *Configuration :* l'écran de configuration apparaîtra avec 16 valeurs à choisir. Les valeurs sélectionnées à afficher seront surlignées.

Certaines de ces valeurs vont de soi, d'autres nécessitent des explications. Ainsi Vdis et Idis (distorsion) sont calculés à l'aide de :

COMMENT CONNECTER L'ANALYSEUR ENTRE CHARGE ET SOURCE

Comme le montre ce croquis, on ne connecte pas les cordons de mesure de la même manière selon que l'on mesure la consommation d'une charge ou la puissance fournie par une source. Il s'agit d'éviter que la chute de tension dans les cordons ne fausse les mesures du courant. La différence de potentiel entre les connexions I et la connexion V– doit rester inférieure à 0,5 V. Sinon, le circuit de protection sera déclenché et il faudra initialiser l'alimentation.

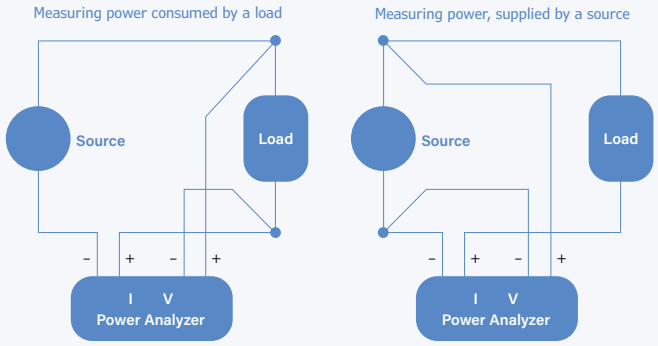


Figure 5. Connexion de la source, de la charge et de l'analyseur

$$V_{dis} = \frac{\sqrt{\sum_{n=2}^{40} V_n^2}}{V_1}$$

où VN = amplitude de l'harmonique N du signal
Re-P = puissance réelle,
P-VA = Vrms x Irms,
Im-P = puissance imaginaire
PF = facteur de puissance, c'est-à-dire Re-P / P-VA.

Si vous choisissez *Eff->*, vous entrez dans l'écran Efficacité pour y choisir la formule qui décrit la relation de puissance recherchée, puis vous retournez à l'écran de configuration. Si vous revenez à l'écran de mesure, la formule choisie avec la valeur calculée est affichée pour tous les canaux.
De retour dans l'écran de configuration, vous pouvez également choisir *Graphs->* qui permet de choisir entre *Scope* et *FFT.* Appuyez sur *Scope*. Cela vous permet de sélectionner les traces. Trace1 sera toujours affichée. Une trace sélectionnée sera surlignée. Toutes les traces peuvent être «connectées» à un canal, et à V, I ou P de ce canal. L'affichage sera «déclenché» par un passage par zéro positif de Trace1. L'affichage vous donnera une impression de la forme d'onde des signaux et de leur relation de phase. Deux périodes du signal seront affichées. L'amplitude des signaux est toujours la même : elle est maximale pour les signaux de tension. Pour les distinguer des signaux de tension, l'amplitude des signaux de courant est de 80 %, celle des signaux de puissance de 60 %. Voir p. ex. **fig. 6** les formes d'onde d'un transformateur qui entre en saturation. L'affichage de *Scope* ne donne pas de valeurs absolues, c'est une indication de la forme et de la relation de phase.
De retour dans l'écran graphique, appuyez sur *FFT*. Ceci affiche les composantes FFT d'un signal sélectionné. En haut de l'écran, vous pouvez sélectionner le canal, l'affichage LINear ou LOGarithmique et soit V soit I. L'harmonique la plus forte, généralement mais pas nécessairement la première, est affichée à 100 % ou 0 dB. L'échelle horizontale donne le rang des harmoniques, et non leur fréquence.
Lorsque vous accédez à la fonction d'enregistrement, vous devez d'abord choisir la période concernée. Les valeurs possibles sont comprises entre 0,25 h et 128 h par pas d'un facteur 2. L'étape suivante consiste à fixer le nombre de traces à enregistrer (trois au maximum). Chaque trace peut être liée à un canal et dans ce canal, vous pouvez choisir V, I ou P. En appuyant sur *Continue*, vous lancez l'enregistrement. Un écran alphanumérique s'affiche, dans lequel vous pouvez voir les valeurs moyenne, maximale et minimale ainsi que le temps écoulé. Vous pouvez également passer à l'écran graphique. En haut, vous pouvez choisir une autre trace, si elle est activée. En appuyant sur *Return*, vous revenez à l'écran principal, les données collectées sont perdues.
Vous pouvez sélectionner le facteur de crête (*V-CF* ou *I-CF*) dans l'écran de configuration. Il divise simplement la valeur de crête la plus élevée (positive ou négative) par la valeur efficace du signal (RMS). N'oubliez pas que la largeur de bande des amplificateurs de mesure est de 2,5 kHz, cette fonction n'est donc pas adaptée à l'audio. Seulement jusqu'à, disons, 250 Hz selon vos goûts.

BOBINER LE TRANSFORMATEUR

Le transformateur TR1 est fait maison. En voici la recette. Ingrédients (cf la nomenclature) : deux moitiés de noyau, deux clips, un support de bobine, pour EF20, fil avec isolation de 900 V et diamètre de 1,3 mm maximum.

- Prenez le corps de bobine et coupez les broches 2, 3, 4, 7, 8 et 9.
- Faites 9 tours de fil entre les broches 1 et 10. Ces spires doivent faire exactement une couche.
- Répétez pour faire 9 tours de fil entre les broches 5 et 6. Là encore, ces spires doivent remplir une couche exactement.
- Mettez les moitiés de noyau et les clips. Il n'y a pas de vide.
- Si tout s'est bien passé, vous aurez un transformateur avec une inductance primaire et secondaire égale d'environ 125 µH. La capacité entre primaire et secondaire sera de l'ordre de 30 pF.

Étalonner et régler

Depuis l'écran principal, vous pouvez accéder à l'écran d'étalonnage (*Calibration*). Vous pouvez choisir entre calibrage de l'écran tactile et

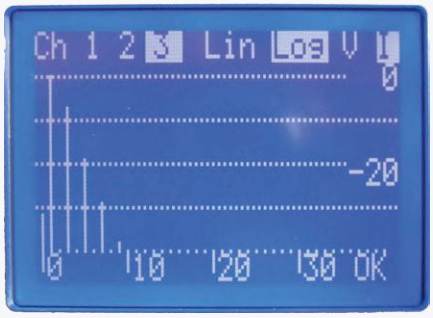


Figure 6a. Mesure d'un transformateur en saturation, FFT du courant.

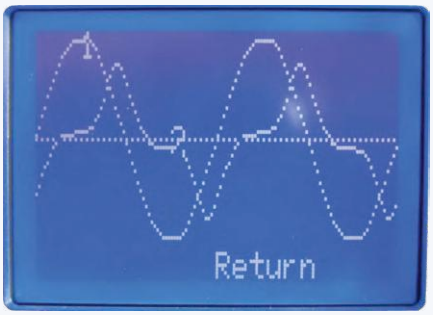


Figure 6b. Mesure d'un transformateur en saturation, tension et courant.

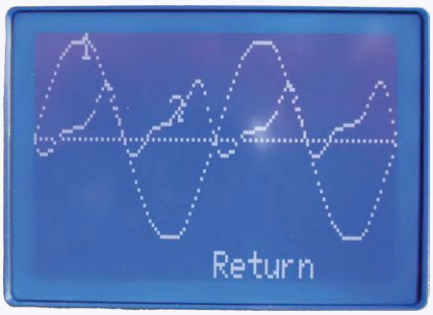


Figure 6c. Mesure d'un transformateur en saturation, tension et puissance.

calibrage des canaux. La première fois, c'est la procédure de calibrage de l'écran tactile qui apparaît automatiquement. Utilisez un crayon ou un stylet en bois pour calibrer (et utiliser) l'écran tactile.
Dans le calibrage du canal, vous serez d'abord invité à sélectionner un canal. Ensuite, le logiciel présente un écran de mesure et d'étalonnage de la tension, du courant et de la fréquence. Sur cet écran, le calibre automatique pour V et I est désactivé, ce qui permet d'étalonner facilement chaque gamme. Avec les flèches (< et >), vous pouvez changer de calibre et ajuster la valeur mesurée.
La meilleure méthode pour étalonner la fréquence est de connecter un oscilloscope numérique à TP8 (signal) et TP2 (GND) sur la carte satellite du canal sélectionné. Ajustez la largeur d'impulsion visible à 35,00 ms à l'aide des flèches sur la ligne de fréquence. La deuxième meilleure méthode pour étalonner la fréquence est d'utiliser une source de signal avec une fréquence calibrée de 50 ou 60 Hz et de régler l'affichage sur 50,00 ou 60,00 Hz. Une erreur de ±0,5 % est admissible. Une fois l'étalonnage d'un canal effectué, appuyez sur *->OK.* Les données de calibrage sont alors stockées sur la carte satellite de la chaîne sélectionnée.

Mon projet est terminé, il n'y aura plus d'ajouts ni de changements significatifs, mais je répondrai volontiers à toutes les questions.
Cet article est basé sur le contenu de la page du projet Elektor Labs [2] où l'on trouvera des informations plus détaillées sur cet analyseur de puissance. Son logiciel, ses fichiers de PCB et la nomenclature sont téléchargeables. Les fichiers CAO Eagle sur Elektor Labs sont à jour, mais les nouvelles versions des PCB n'ont pas été testées !

190133-B-02

Votre avis, s'il vous plaît ?
N'hésitez pas à poser vos questions ou envoyer vos commentaires sur cet article, en anglais à l'auteur w.j.dijkman@onsbrabantnet.nl
ou en français à redaction@elektor.fr

Ont contribué à cet article
Auteur : **Wil Dijkman**
Rédaction : **Luc Lemmens, CJ Abate**
chéma : **Patrick Wielders**
Maquette : **Giel Dols**
Traduction : **Maya Zep**

PRODUITS

› **Livre : Les microcontrôleurs PIC pour les débutants (Rémy Mallard)**
www.elektor.fr/15959

LIENS

[1] **wattmètre CA/CC** : www.elektormagazine.fr/magazine/elektor-201509/28016
[2] **Power Analyzer** : www.elektormagazine.fr/labs/power-analyzer
[3] **calcul de FFT** : www.nicholson.com/dsp.fft1.html
[4] **The Scientist and Engineer's Guide to Digital Signal Processing, by Steven W. Smith, Ph.D.** : www.dspguide.com/pdfbook.htm

conception de filtres analogiques (3)

Filtres passifs

Alfred Rosenkränzer

Le troisième et dernier volet de cette série traite de quelques subtilités des filtres passifs. Comme seuls des composants purement passifs tels que des résistances, des bobines et des condensateurs sont utilisés ici, il n'est pas question d'amplification. Il est donc impossible d'obtenir une impédance d'entrée élevée et une impédance de sortie faible. Cependant, les hautes fréquences dans la gamme des MHz, à trois chiffres et au-delà, ne posent pas de problème.

Avec des filtres actifs, l'impédance d'entrée peut être très élevée grâce à un tampon, tandis qu'un tampon de sortie peut abaisser fortement l'impédance de sortie. On n'a donc pas à se préoccuper de la terminaison correcte ni de l'adaptation adéquate des impédances à l'entrée et à la sortie. Cependant, à des fréquences très hautes, les filtres passifs sont encore courants. Ces filtres sont conçus pour

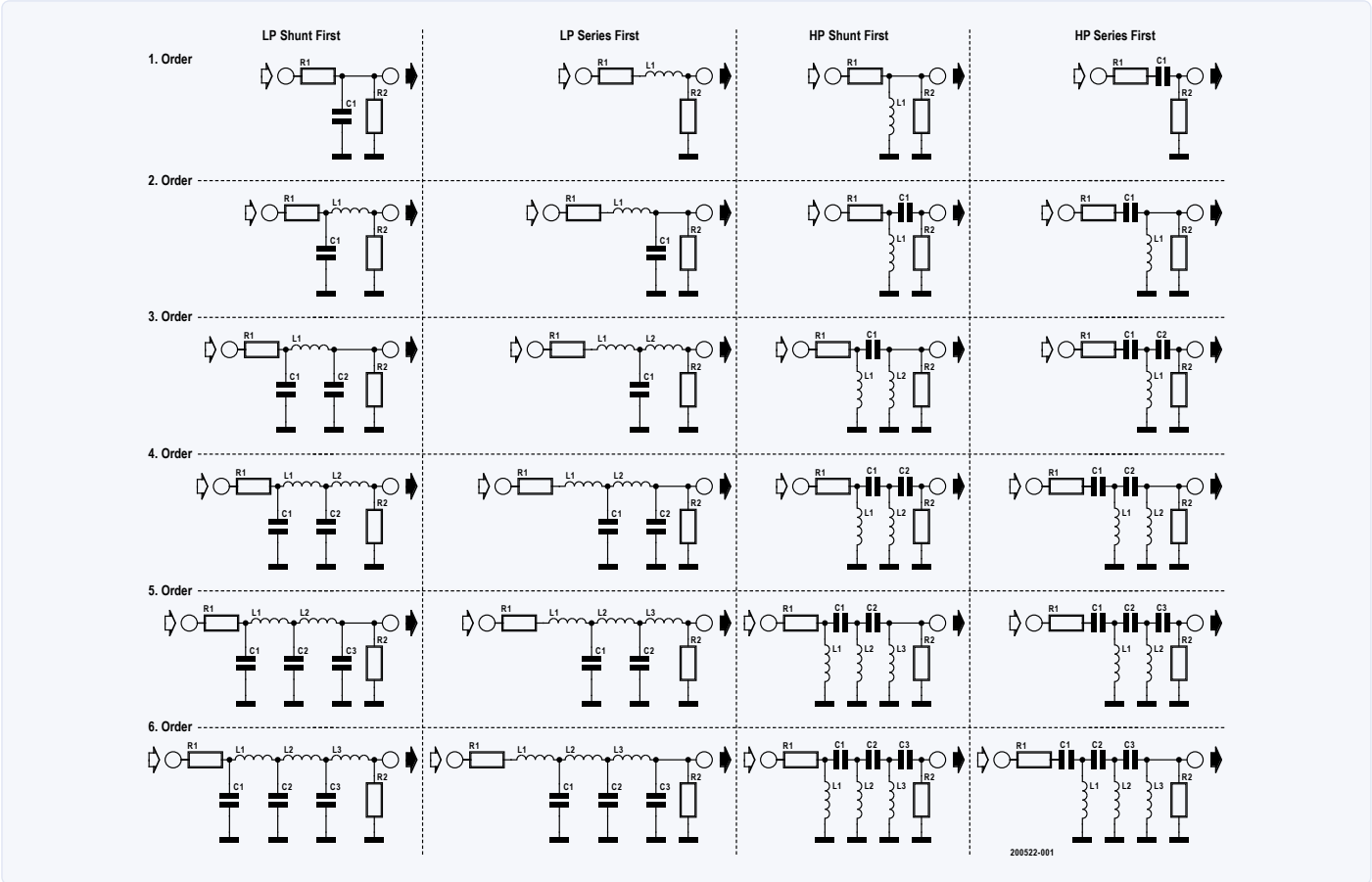


Figure 1. Circuits de base de filtres passe-bas et passe-haut passifs du 1er au 6e ordre en PI et T. La fréquence de coupure, mais aussi les caractéristiques du filtre dépendent des valeurs des composants. Ces modèles conviennent aux filtres de Bessel, Butterworth ou Tchebychev.

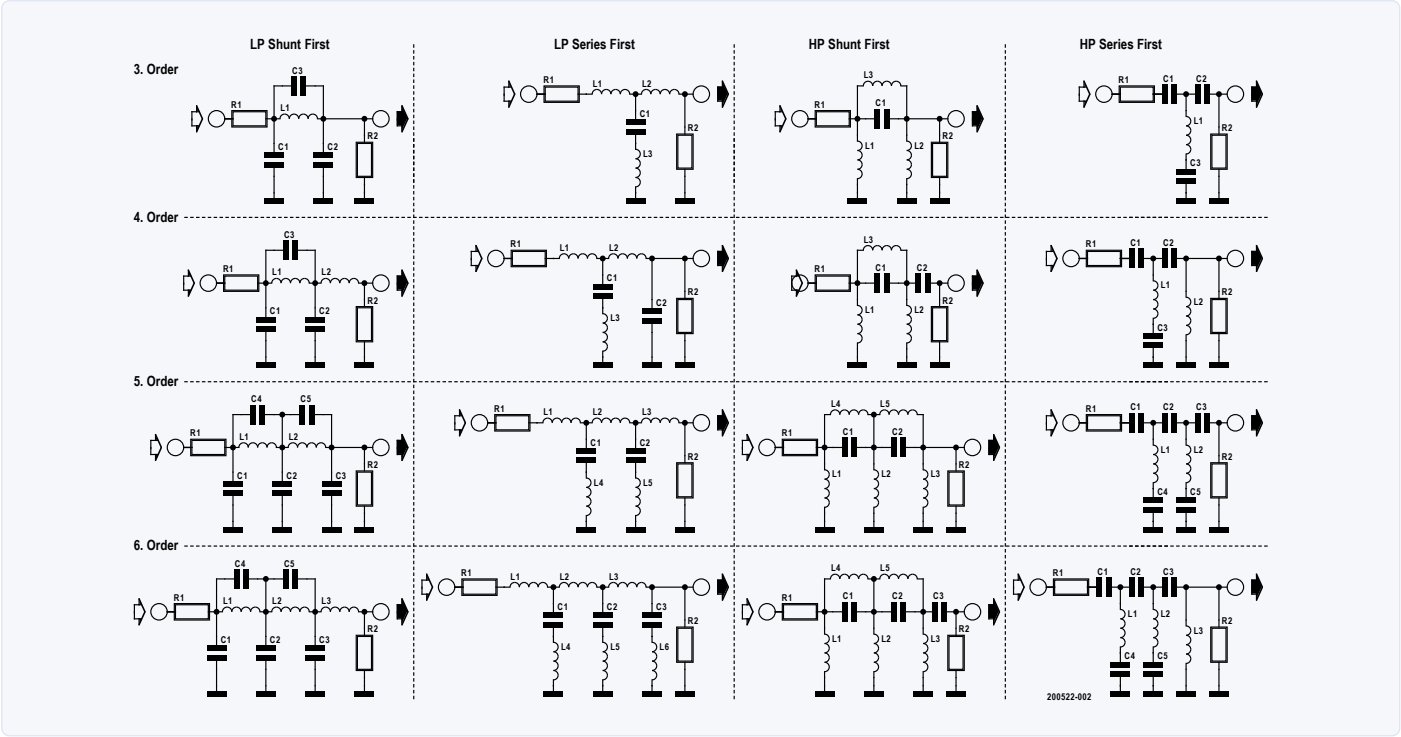


Figure 2. Circuits de base de filtres passe-bas et passe-haut du 3e au 6e ordre en PI et T avec caractéristiques de Cauer ou de Tchebychev inverse.

une impédance d'entrée et de sortie spécifique (pas nécessairement la même). Un écart par rapport à l'impédance de source et de terminaison supposée peut donc affecter de manière significative les caractéristiques du filtre. La section suivante décrit les types de filtres passifs couramment utilisés et ce qu'il faut en retenir.

Structurons

La **figure 1** montre les structures de base des passe-bas et haut du 1er au 6e ordre. Si les composants sont bien dimensionnés, ces structures permettent de réaliser des filtres qui présentent les caractéristiques de Bessel, Butterworth ou Tchebychev. Vous pouvez choisir si un filtre commence par un composant en série (structure en T) ou par un composant relié à la masse (structure en PI).

La colonne de gauche de la figure 1 montre les filtres passe-bas PI, qui commencent tous par un condensateur à la masse. À droite, on trouve des filtres passe-bas en T, qui commencent donc par une bobine «en série».

Pour les passe-haut, c'est l'inverse : dans la troisième colonne se trouvent les passe-haut PI avec une bobine à la terre et dans la colonne de droite, les passe-haut T avec un condensateur en série. Pour obtenir les ordres supérieurs, on ajoute à tour de rôle les composants longitudinaux (en série) ou transversaux (reliés à la masse). Cependant, les valeurs des autres composants changent. Le choix de la structure T ou PI n'a aucune importance pour la fonction. Comme les bobines ne sont pas très populaires auprès de nombreux électroniciens, on essaie souvent de s'en sortir avec le moins de selfs possible.

La **figure 2** montre les structures de base des filtres de Cauer (elliptique) et de Tchebychev inverse (dit de type 2). Les bobines ou les condensateurs sont remplacés par des circuits résonnants parallèles ou en série. On y voit des passe-bas et passe-haut en PI et en T du 3e au 6e ordre.

La **figure 3** montre les structures de base des passe-bande et des coupe-bande avec des caractéristiques de Bessel, Butterworth ou Tchebychev (du 3e au 7e ordre), et la **figure 4** montre des structures plus complexes correspondantes avec des caractéristiques de Cauer et de Tchebychev inverse (5e et 7e ordre). Ici aussi, il existe des variantes PI et T.

Dimensionnons

Un filtre de Bessel ou de Butterworth est entièrement défini par l'approximation du filtre, la fréquence de coupure de –3 dB, la structure choisie, l'ordre et l'impédance d'entrée et de sortie. Contrairement aux filtres actifs, il n'y a sur les filtres passifs aucune liberté pour modifier les valeurs des composants. Par exemple, si vous voulez utiliser une self d'une valeur standard de la série E, vous devez légèrement modifier la fréquence de coupure du filtre à une impédance donnée. Bien entendu, cela ne peut se faire que si l'application le permet.

La première partie de cette série [1] a montré comment les propriétés d'un filtre de Butterworth changent en choisissant des ordres différents. Il est intéressant d'observer à présent comment les valeurs des composants changent si on fait varier la fréquence de coupure. La **figure 5** montre le circuit entièrement dimensionné d'un filtre passe-bas Butterworth de 5e ordre en structure PI avec une fréquence de coupure de 1 MHz. Si vous voulez doubler la fréquence de coupure à 2 MHz, il vous suffit de réduire de moitié les valeurs des selfs et des condensateurs. Ce n'est pas étonnant, car la fréquence de coupure est proportionnelle $1 / \sqrt{(L * C)}$. Par conséquent, même avant de disposer de calculatrices ou de PC,

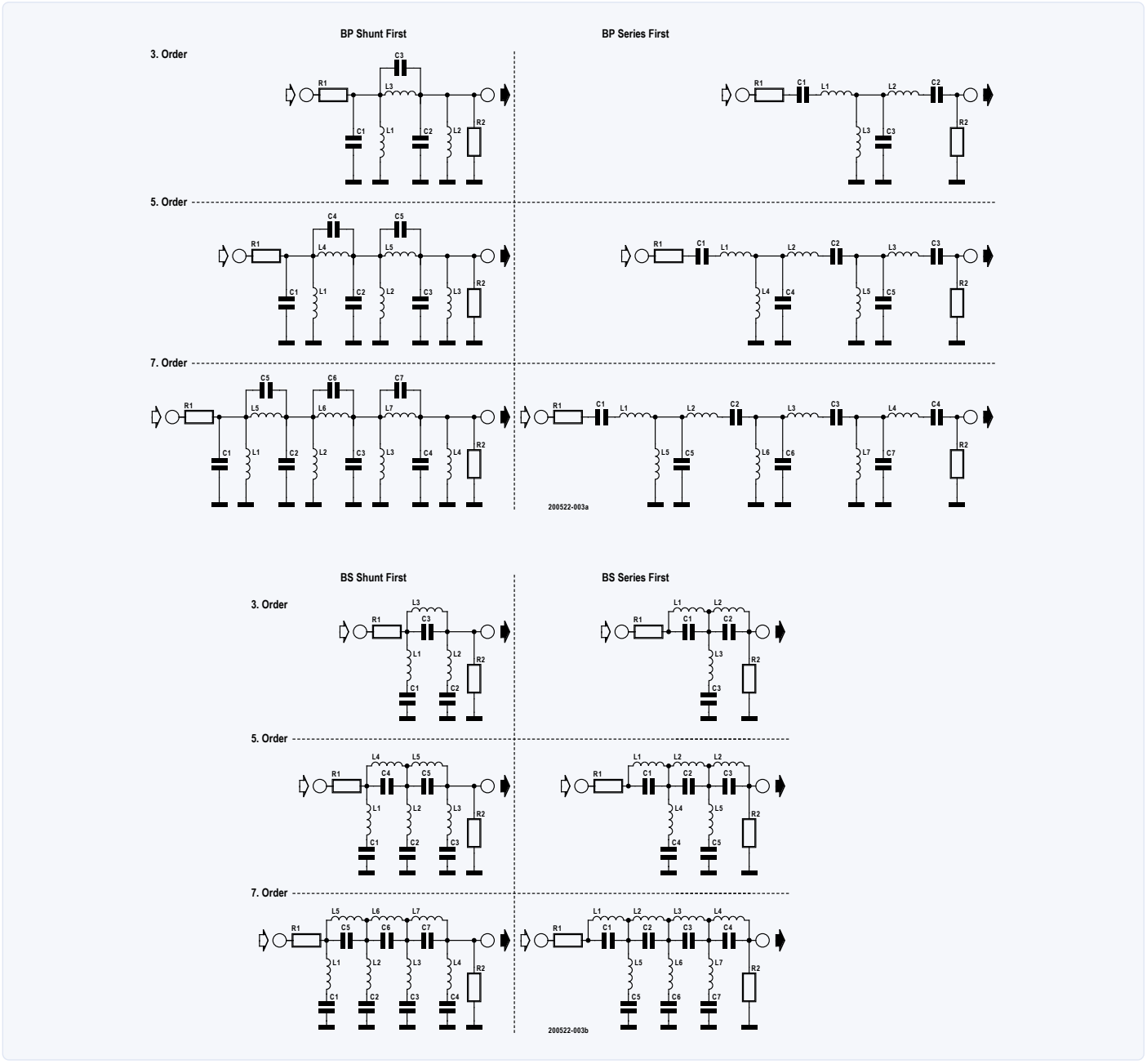


Figure 3. Circuits de base de filtres passe-bande et coupe-bande du 3e, 5e et 7e ordre en PI et T avec caractéristiques de Bessel, Butterworth ou Tchebychev

il était facile de convertir un circuit de base avec une fréquence de coupure donnée à la fréquence cible souhaitée. Bien sûr, avec l'impédance standard de 50 Ω, les valeurs de résistance restent les mêmes.

Il est également intéressant de voir comment les valeurs des composants se comportent lorsqu'un filtre est calculé pour d'autres impédances. Même sans ordinateurs : si vous voulez doubler les 50 Ω d'impédance du filtre passe-bas de 1 MHz de la fig. 5, logiquement seule la valeur des bobines double, tandis que la valeur des condensateurs diminue de moitié. Pas étonnant, puisque Z est proportionnel à √(L/C).

C'est le bon moment pour montrer ici l'effet d'une mauvaise source et une résistance terminale. Le filtre calculé pour une impédance de 100 Ω a été délibérément «mal» piloté avec une source de 50 Ω. Sa terminaison est aussi de 50 Ω. La **figure 6** montre sa réponse en fréquence dans la bande passante.

Retour au dimensionnement : dans un filtre de Tchebychev, l'ondulation résiduelle de la bande passante est un paramètre supplémentaire. L'effet sur la réponse en fréquence est illustré à la **figure 7**, qui montre la réponse en amplitude des filtres passe-bas Tchebychev du 7e ordre de 1 MHz avec une ondulation résiduelle dans la bande passante de 0,1 dB, 0,5 dB, 1 dB et 3 dB. Plus on tolère d'ondulation résiduelle, plus la pente sera forte dans la bande de coupure. C'est vous et l'application du filtre qui fixez l'ondulation résiduelle

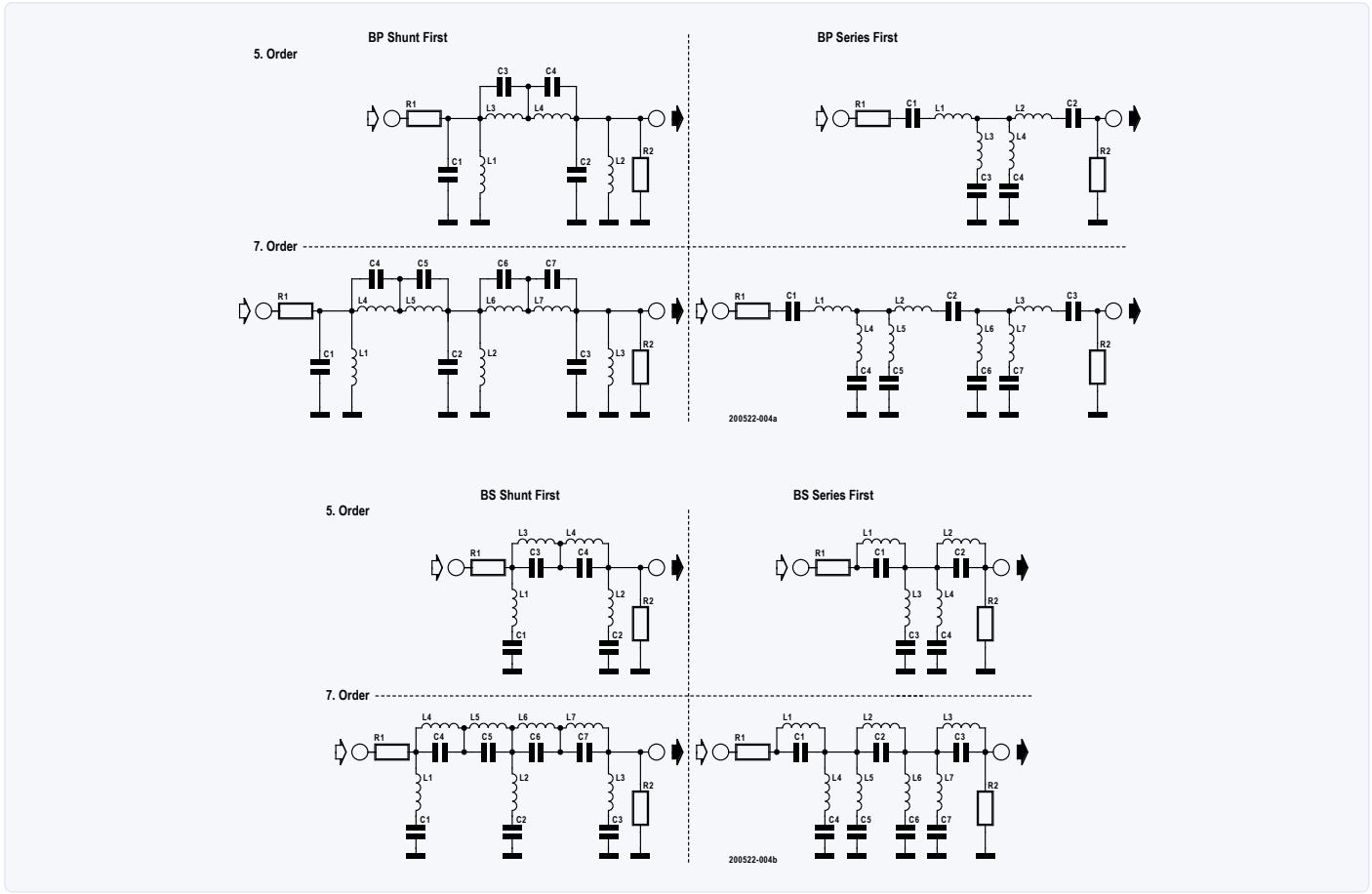


Figure 4. Circuits de base de filtres passe-bande et coupe-bande du 5e et du 7e ordre en PI et T avec caractéristiques de Cauer ou de Tchebychev inverse.

maximale acceptable.

La **figure 8** montre une version agrandie de la figure 7 dans la bande passante. Ici, vous voyez que la fréquence de coupure n'est pas le point –3 dB comme pour les autres filtres, mais le point de la courbe où l'ondulation passe sous un seuil défini. Si l'on normalise les courbes au point –3 dB pour pouvoir mieux comparer, il faut ajuster la fréquence de coupure en conséquence.

Avec le filtre de Cauer, l'atténuation minimale dans la bande de

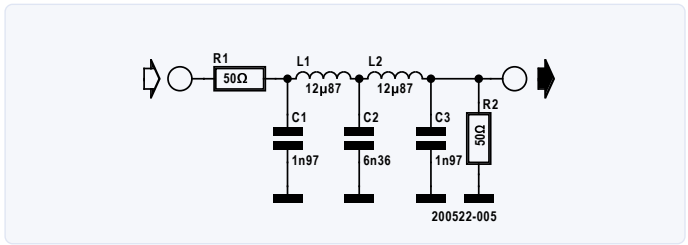


Figure 5. Passe-bas Butterworth du 5e ordre en PI avec fréquence de coupure de 1 MHz.

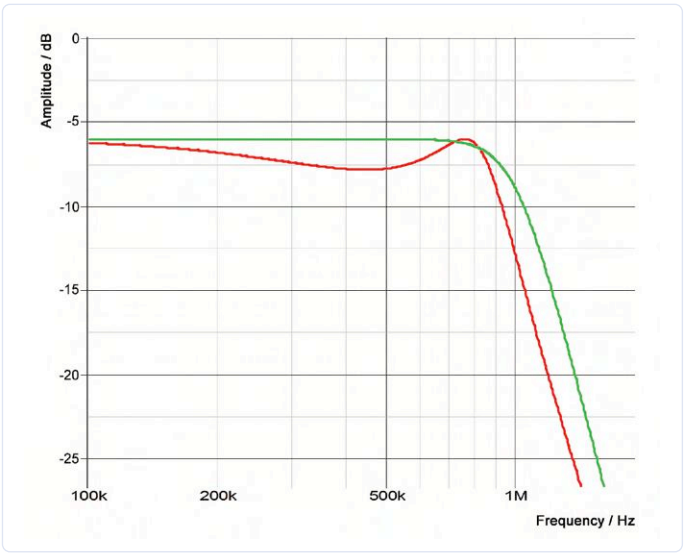


Figure 6. Réponse en fréquence d'un filtre passe-bas Butterworth du 5e ordre dans la bande passante. La courbe verte correspond à la bonne réponse et la rouge résulte de l'inadaptation d'un filtre 100 Ω et de terminaisons 50 Ω.

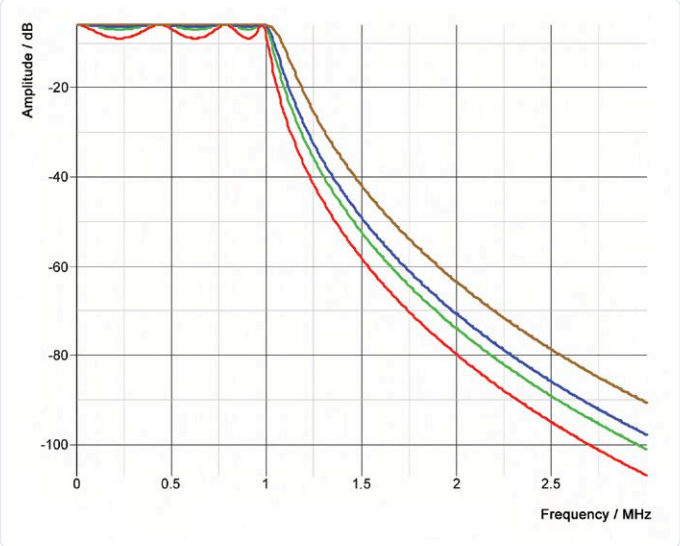


Figure 7. Réponses en fréquence de filtres passe-bas Tchebychev du 7e ordre de 1 MHz avec une ondulation dans la bande passante de 0,1 dB (brun), 0,5 dB (bleu), 1 dB (vert) et 3 dB (rouge).

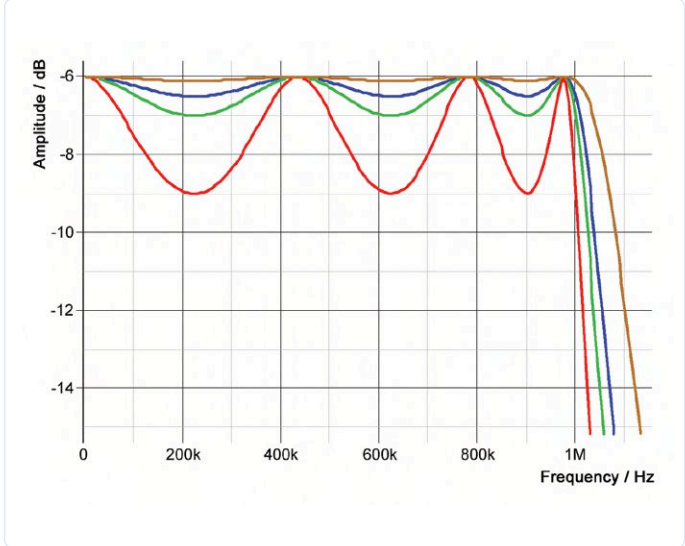


Figure 8. Agrandissement de la figure 7 dans la bande passante. La fréquence de coupure correspond au point de la courbe où l'ondulation passe sous un seuil défini : ondulation = 0,1 dB (brun), 0,5 dB (bleu), 1 dB (vert) et 3 dB (rouge).

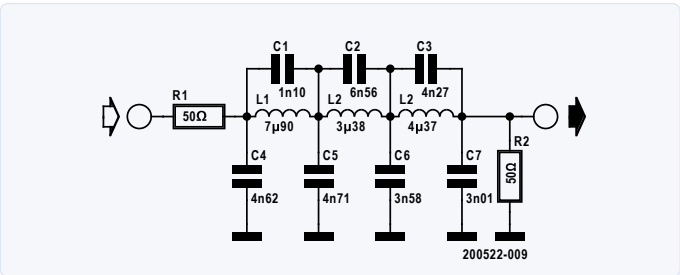


Figure 9. Filtre passe-bas Cauer du 7e ordre avec une ondulation de 0,5 dB dans la bande passante et une atténuation minimale de 40 dB. Le tableau 1 donne les valeurs des composants pour obtenir une plus forte atténuation.

coupure est un paramètre supplémentaire. Les encoches assurent une transition encore plus raide vers la bande de coupure, mais la courbe revient ensuite à une atténuation moindre. Le nombre d'encoches correspond au nombre de circuits résonnants dans le schéma (s'ils ont des fréquences différentes). La **figure 9** montre un filtre passe-bas Cauer du 7e ordre avec une atténuation minimale de 40 dB. Le **tableau 1 Valeurs des composantes de la fig. 9** contient les valeurs d'atténuation minimale de 40, 50, 60 et 70 dB. La hauteur des bosses correspond à l'atténuation minimale.

La **figure 10** montre la réponse en fréquence des quatre filtres passe-bas de Cauer. La hauteur des bosses correspond à l'atténuation minimale. Vous pouvez voir les différentes pentes des courbes

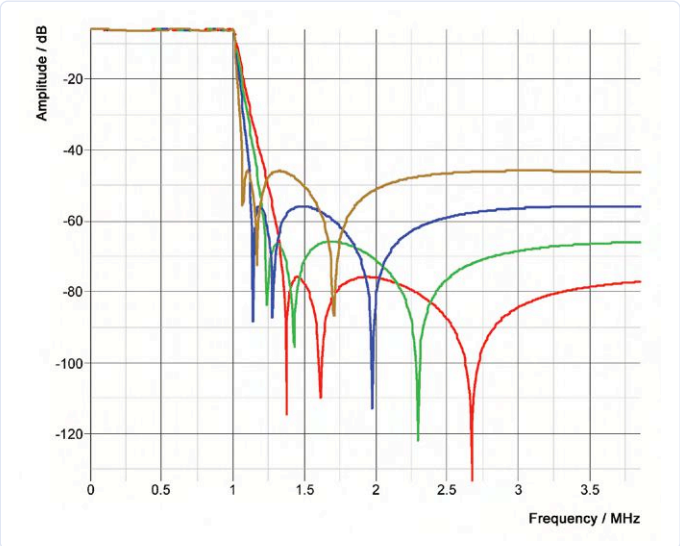


Figure 10. Réponse en fréquence des filtres passe-bas Cauer du 7e ordre de 1 MHz avec atténuation minimale de 40, 50, 60 et 70 dB.

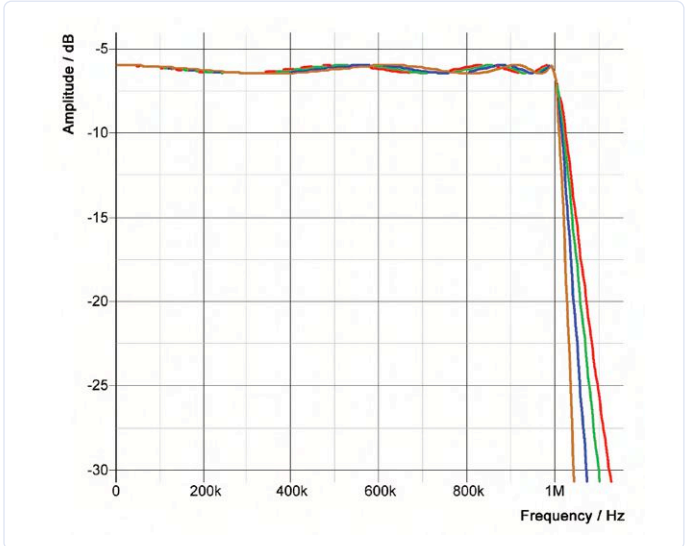


Figure 11. Agrandissement de la figure 10 dans la bande passante avec les atténuations minimales de 40 dB (marron), 50 dB (bleu), 60 dB (vert) et 70 dB (rouge).

Tableau 1. Valeurs des composants de la figure 9.

atténuation	C1	C2	C3	C4	C5	C6	C7	L1	L2	L3
40 dB	1n10	6n56	4n27	4n62	4n71	3n58	3n01	7µ90	3µ38	4µ37
50 dB	766p	4n10	2n77	4n90	5n48	4n52	3n61	8µ50	4µ78	5µ61
60 dB	541p	2n74	1n88	5n07	6n14	5n36	4n09	8µ92	6µ04	6µ65
70 dB	385p	1n90	1n31	5n20	6n68	6n07	4n45	9µ22	7µ11	7µ49

dans la zone de transition et l'atténuation minimale associée. Mais cela modifie également les fréquences des encoches. Si vous avez une fréquence d'interférence fixe à filtrer, vous pouvez déplacer une encoche en ajustant soit la fréquence de coupure, soit l'atténuation minimale. Dans ce cas, gardez un œil sur les tolérances des composants et donc sur la position réelle des encoches. La section agrandie des réponses en fréquence de la **figure 11** montre que l'ondulation est de 0,5 dB pour tous les filtres.

La dernière approximation du filtre est la caractéristique inverse de Tchebychev (IT). Elle ressemble à un Butterworth dans la bande passante et n'a donc pas d'ondulation, ce paramètre est donc omis. Dans la bande de coupure, cependant, la caractéristique IT correspond davantage à un filtre de Cauer avec des encoches et une atténuation minimale correspondante. Le circuit de base est identique à celui du filtre de Cauer (fig. 9), seul le dimensionnement est différent. La **fig. 12** montre la réponse en fréquence des filtres de Tchebychev inversés avec une atténuation minimale de 50, 60 et 70 dB. La **fig. 13** est la version agrandie de la fig. 12 dans la bande passante.

Laissons tout passer

En télévision analogique, il fallait (autrefois) des filtres raides, dont les fortes résonances en régime impulsionnel étaient gênantes. Défaut du retard de groupe que l'on compensait avec force passe-tout. Ce n'était pas du gâteau, car ces filtres devaient être accordés manuellement à l'aide d'un analyseur de réseau. Il n'existe que peu de programmes de filtrage capables de calculer le filtre dans son ensemble. Avec la numérisation croissante de la technologie vidéo, à des fréquences d'horloge plus élevées et grâce au suréchantillonnage dans les CA/N et CN/A, les exigences en matière de filtres analogiques ont diminué au point qu'aujourd'hui on peut se passer de la compensation du retard de groupe. La **fig. 14** montre le circuit d'un filtre passif passe-tout de 2e ordre et la **fig. 15** montre le retard de groupe correspondant.

Un peu de pratique maintenant : la **fig. 16** montre un filtre vidéo fait maison avec des bobines Neosid (les cubes couleur cuivre à noyau accordable). Les deux bobines de gauche font partie d'un filtre passe-bas Cauer du 5e ordre. Le bloc des six bobines restantes appartient à un passe-tout. Pour obtenir les valeurs «tordues» requises, il y a toujours plusieurs condensateurs connectés en parallèle. Les condos ronds bleus sont des KP dans la gamme des nF avec une tolérance de 2 %.

Filtres spéciaux

Les filtres considérés jusqu'à présent étaient des filtres *asymétriques* dont les signaux étaient tous référencés à la masse. Nous verrons que des composants passifs peuvent également filtrer les signaux différentiels.

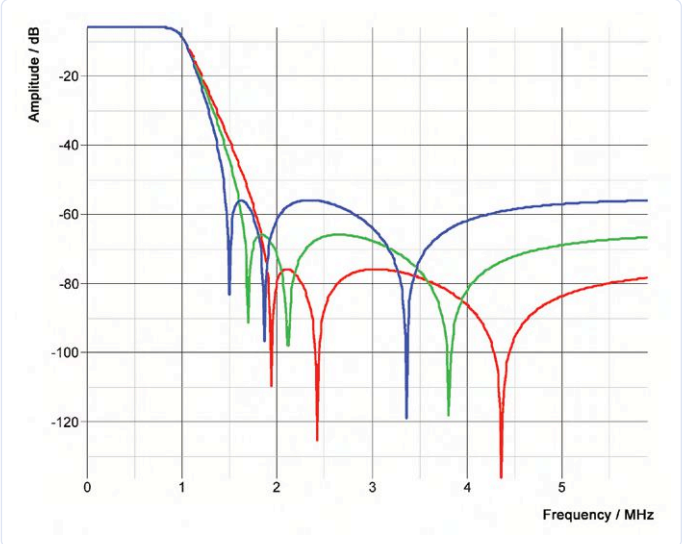


Figure 12. Réponses en fréquence des filtres de Tchebychev inversés avec 50, 60 et 70 dB.

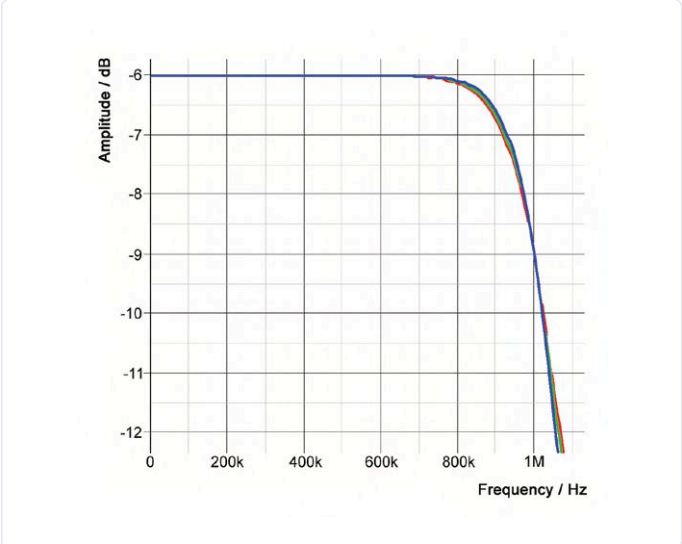


Figure 13. Agrandissement de la figure 12 dans la bande passante avec atténuation minimale de 50 dB (bleu), 60 dB (vert) et 70 dB (rouge)

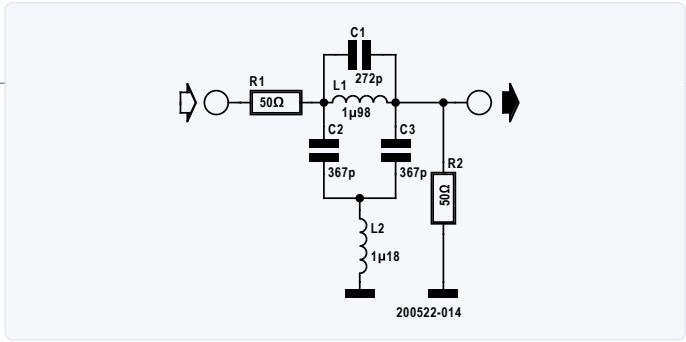


Figure 14. Circuit d'un filtre passif passe-tout du 2e ordre.

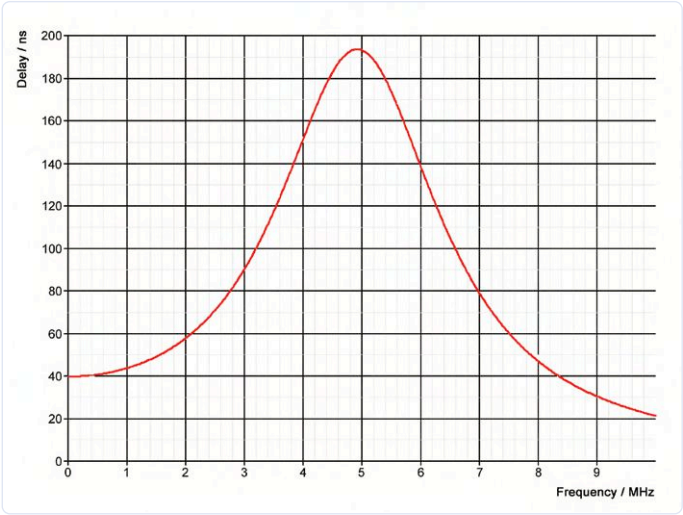


Figure 15. Parcours du retard de groupe du filtre passif passe-tout du 2e ordre de la figure 14.

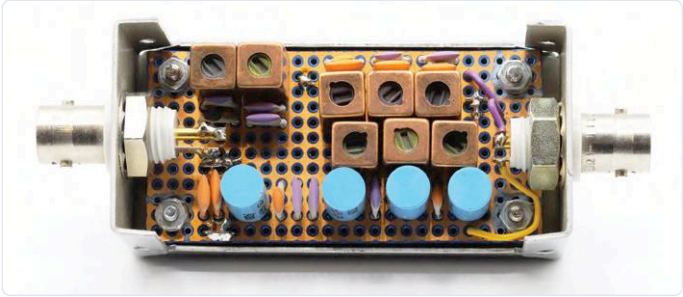


Figure 16. Filtre vidéo fait maison avec huit bobines Neosid.

Filtres différentiels

Les CA/N et les CN/A modernes pour signaux de fréquence plus élevée sont dotés d'entrées et de sorties différentielles. Il est donc logique d'avoir recours à un filtrage différentiel pour les filtres passe-bas directement à la sortie du CN/A et même pour le filtre anti-repliement avant l'entrée du CN/A. Le circuit de la **fig. 17** est simplement constitué de deux filtres passe-bas Butterworth de 5e ordre identiques pour 10 MHz dans une conception classique à une seule extrémité. Hormis la masse, ils n'ont aucun composant commun, c'est pourquoi les tolérances des composants peuvent entraîner des différences de comportement entre les deux «canaux».

La deuxième variante de la **fig. 18**, en revanche, va plutôt dans le sens d'un véritable filtre différentiel, car ici deux des condensateurs reliés à la terre de la fig. 17 ont été combinés en un seul. Ils influencent désormais les deux voies de la même manière. Seules les selfs pourraient encore causer des différences ici.

Pour les signaux différentiels ou *push-pull*, l' effet de filtre est le même pour les deux circuits. Pour les signaux en mode commun, il y a cependant un comportement de filtre différent. La **fig. 19** montre qu'avec le filtre combiné de la fig. 18, seules les bobines influencent la réponse en fréquence et donc que seul un faible effet de filtre en résulte pour les signaux en mode commun. Bien que les signaux en mode commun soient réduits par les CN/A de l'amplificateur différentiel suivant, ce n'est pas satisfaisant, car la réjection en mode commun des amplificateurs RF se détériore avec l'augmentation de la fréquence. Chaque solution a ses avantages et ses inconvénients. Afin d'obtenir des branches de filtres aussi identiques que possible, on peut par exemple utiliser des modules de filtres finis provenant du même lot de production, puisque les dispersions entre ces filtres seront moindres.

Filtre de Tchebychev double inverse du 4e ordre

J'ai remarqué ce filtre en analysant un filtre acheté, construit à partir de deux filtres composés du 4e ordre (**fig. 20**). Il est facile à réaliser avec des composants de la série E classique et demande moins de valeurs tordues. Pour modifier sa fréquence de coupure, décalez les valeurs dans la série E. Le prix de cet avantage est une petite bosse dans la bande passante. La **fig. 21** donne la réponse en fréquence de ce filtre, qui atteint une atténuation minimale impressionnante de 70 dB dans la bande de coupure. Les deux

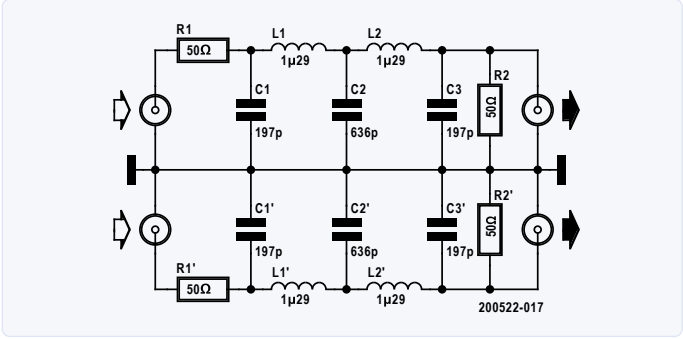


Figure 17. Deux filtres passe-bas Butterworth «normaux» pour un signal différentiel.

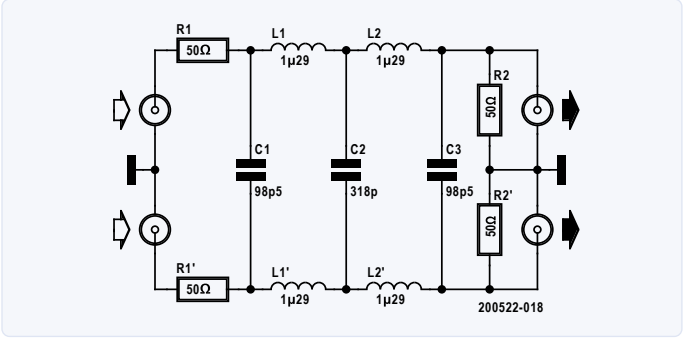


Figure 18. Cette variante combine Cx et Cx' de la figure 17, connectés chacun à la masse, en un seul condensateur dont la capacité est donc égale à la moitié de celle des deux condensateurs.

encoches ont la même fréquence et renforcent l'effet. La **fig. 22** montre la réponse en fréquence agrandie dans la bande passante ainsi que la petite bosse caractéristique à 75 MHz.

Choisissons

Nous l'avons déjà dit au début de cet article, il n'y a pas de marge de liberté pour dimensionner des filtres passifs. Pour le calcul de ces filtres assez sophistiqués, un logiciel spécialisé est indispensable. Sous [2, 3 et 4], vous trouverez une sélection de programmes de filtrage adaptés. Les valeurs des bobines et des condensateurs calculées de cette manière ne seront pas disponibles dans la plupart des cas. Avec les condensateurs, le problème est vite résolu : Il suffit de connecter deux ou trois condensateurs en parallèle pour se rapprocher le plus possible de la valeur requise. Les condensateurs sont petits et aussi moins chers, mais il ne faut jamais perdre de vue leurs tolérances.
Une simulation [5] vous permet de vérifier si avec les valeurs réelles trouvées la réponse en fréquence du filtre reste suffisamment proche de l'idéal. Lorsque vous construisez un prototype, donnez-vous la peine de mesurer les condensateurs avant de les monter. Il en va des filtres passifs comme des filtres actifs. Tout filtre est aussi bon que ses condensateurs. Renoncez aux condensateurs CMS de type X7R etc. à des fins de filtrage. Les bons condensateurs sont (malheureusement) souvent volumineux et moins bon marché. Ne commencez pas l'étude du circuit imprimé avant que le filtre soit au point et que vous savez exactement quels composants seront utilisés.

Les bobines sont moins commodes. Ce n'est pas la peine d'essayer de les connecter en parallèle. Même la connexion en série ne fonctionne que s'il n'y a aucun couplage magnétique des bobines. Pour éviter toute induction mutuelle entre bobines, vous devez les décaler de 90° les unes par rapport aux autres et les disposer en zigzag sur la carte. Admirez le décalage des bobines du filtre fini de la **fig. 23,** même si la troisième avec son angle de 45° est légèrement décalée. Dans le filtre fini de la **fig. 24,** ni les petites bobines bleues ne sont connectées en série ni les condensateurs en parallèle. On peut donc supposer que le fabricant a fait fabriquer ces composants pour qu'ils s'adaptent exactement.

Une solution partielle du problème des bobines est l'utilisation de composants accordables dont vous modifiez l'inductance (dans certaines limites) en réglant une coque en ferrite ou un noyau. C'est le cas des bobines Neosid de la fig. 23.
Il ne faut pas négliger la résistance des bobines, mais la mesurer ou la lire sur la fiche technique du composant, afin de l'inclure dans la simulation et la prendre en compte lors du calcul des filtres. Du fait de leur résistance, on s'éloigne du comportement idéal des bobines au point d'affecter la réponse en fréquence dans la bande passante, un dommage collatéral qu'il faudra corriger ou du moins prendre en compte dans l'étage suivant.
Les bobines à force inductance font appel à un noyau de ferrite pour « amplifier » en quelque sorte l'inductance propre du composant. Selon la taille de ce noyau, le nombre de spires et l'intensité des courants qui y circulent, il se peut que le noyau entre en saturation magnétique, ce qui se traduit par des distorsions massives. Si l'on relève plus d'harmoniques à la sortie d'un filtre qu'à son entrée, il faut soit réduire le niveau du signal, soit utiliser des bobines avec un noyau de ferrite plus gros.

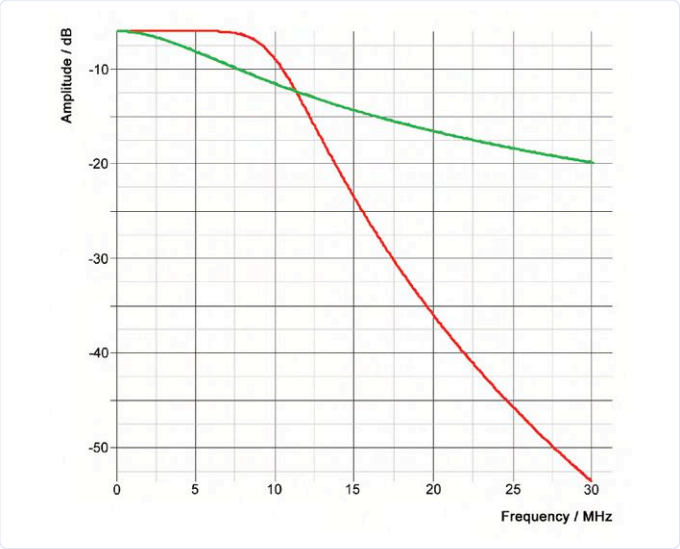


Figure 19. Réponse en fréquence en mode commun du filtre double SE de la figure 17 (rouge) et du filtre combiné de la figure 18 (vert).

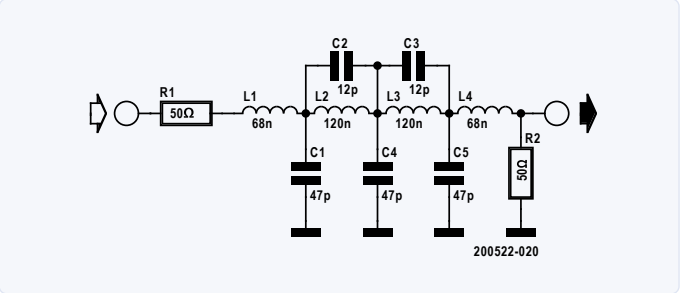


Figure 20. Filtre de Tchebychev double inverse du 4e ordre.

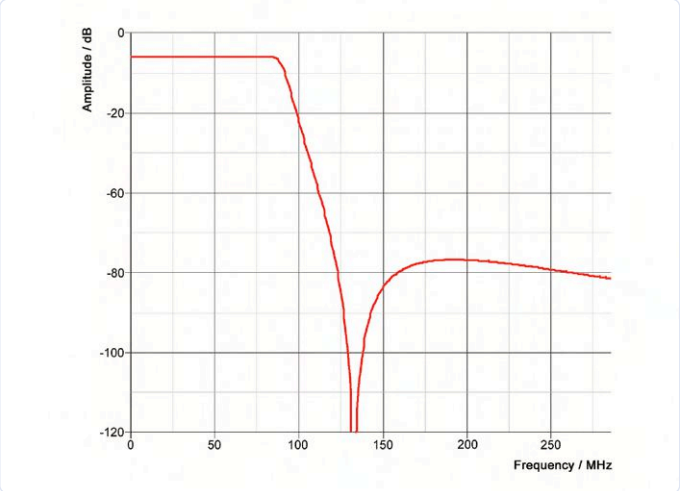


Figure 21. Réponse en fréquence d'un filtre de Tchebychev double inverse du 4e ordre.

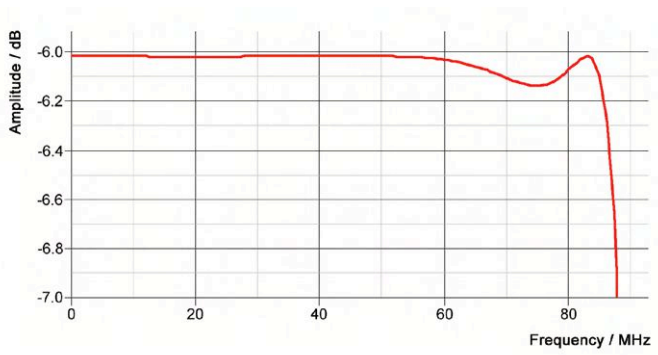


Figure 22. Réponse en fréquence agrandie du filtre de la fig. 20 dans la bande passante. La petite bosse à 75 MHz est caractéristique.

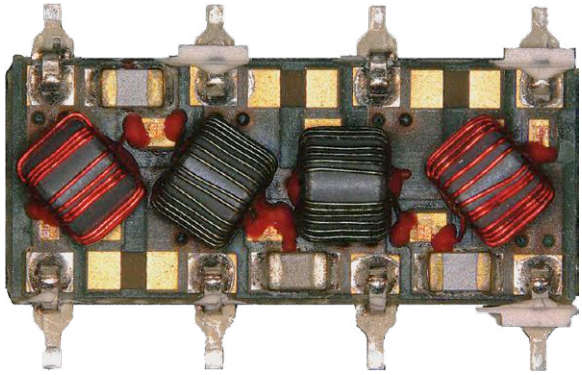


Figure 23. Le fonctionnement interne d'un filtre préfabriqué. Remarquez la disposition décalée des bobines.

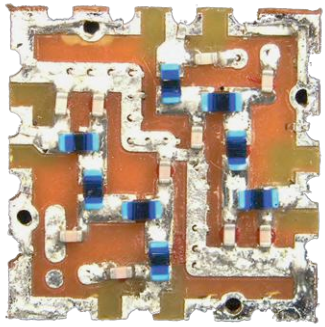


Figure 24. Un filtre fini composé de CMS avec des bobines à 90°. Remarquable : ni les bobines bleues ne sont connectées en série, ni les condensateurs en parallèle.

Disposons

Outre la disposition des bobines en zigzag déjà mentionnée, il faut une connexion courte et directe à la masse. L'emploi de cartes double face avec une couche pour les signaux et une couche sur la face opposée pour la masse facilitera la tâche. Pour un circuit à hautes fréquences, avec par conséquent des condensateurs à faible capacité, il faut se souvenir que les pistes elles-mêmes agissent comme de minuscules condensateurs reliés à la masse. Le cas échéant, cette capacité *parasite* doit être prise en compte lors du choix de la valeur du condensateur réel.

200522-02

Votre avis, s'il vous plaît ?
N'hésitez pas à poser vos questions ou envoyer vos commentaires (en allemand ou en anglais) à l'auteur alfred_rosenkraenzer@gmx.de
ou (en français) à redaction@elektor.fr

Ont contribué à cet article
Auteur : **Alfred Rosenkränzer**
Rédaction : **Thomas Scherer**
Maquette : **Giel Dols**
Traduction : **Frédéric Handle**

PRODUITS

- **OWON SDS1102 2-Kanal-Digital-Oszilloskop (100 MHz)**
www.elektor.fr/owon-sds1102-2-ch-digital-oscilloscope-100-mhz
- **Siglent SDG2042X Arbitrary Waveform Generator (40 MHz)**
www.elektor.fr/siglent-sdg2042x-arbitrary-waveform-generator-40-mhz
- **OWON XSA1015-TG Spectrum Analyser (9 kHz – 1.5 GHz)**
www.elektor.fr/owon-xsa1015-tg-spectrum-analyser-9-khz-1-5-ghz
- **livre : Les oscilloscopes anciens & modernes (Rémy Mallard)**
www.elektor.fr/19124

LIENS

[1] **conception de filtres analogiques (1)** : www.elektormagazine.fr/magazine/200318-03
[2] **LC Filter Design Tool** : https://rf-tools.com/lc-filter/
[3] **AADE** : https://getwinpcsoft.com/Filter-Design-179557/download/
[4] **Quickfil 5.1 (sous DOS)** : www.omicron-lab.com/products/vector-network-analysis/quickfil/
[5] **Simetrix** : www.simetrix.co.uk

module de mesure sans fil
JOY-iT VAX-1030

Harry Baggen

Les modules de mesure sont des espèces de multimètres que l'on connecte à un circuit ou à un appareil pour en mesurer puis en afficher la tension, le courant et d'autres informations. On en trouve de toutes sortes, sous des formes intéressantes, et à des prix… ouh la la, est-ce bien la peine d'en parler ?
Le JOY-iT VAX-1030 est l'un de ces modules, il coûte (tout de même) 40€, mais on dirait qu'il offre de nombreuses fonctions, dont une connexion sans fil.
Voyons ça de près.



À quoi s'attendre pour environ 40 €…? En fait, avant d'avoir la boîte devant moi, je n'y avais pas réfléchi. J'ai même été surpris en l'ouvrant de découvrir de si nombreuses connexions, plus un module d'affichage séparé, avec des boutons tactiles, un câble USB, un câble



Figure 1. Voici le matériel livré pour le JOY-iT VAX-1030.

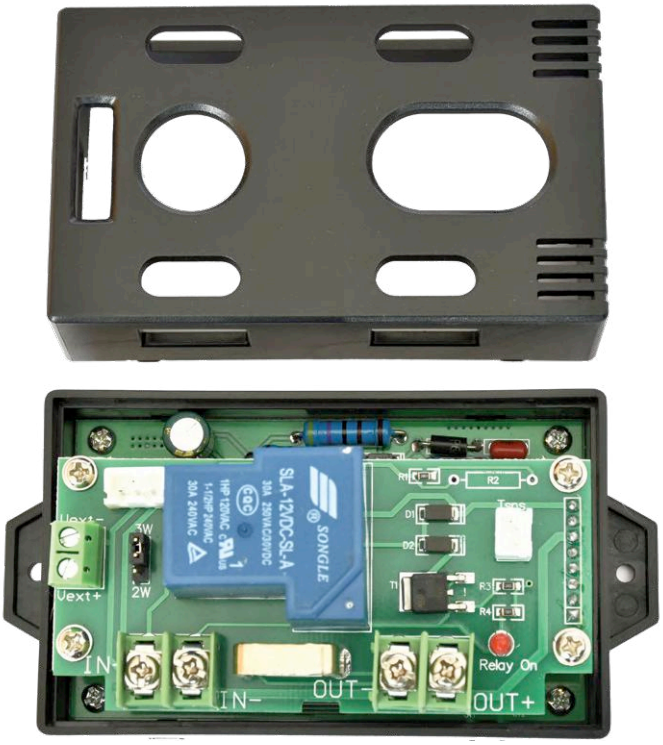


Figure 2. Le module de mesure avec son capot déposé.

avec une sonde de température et enfin d'un câble de connexion séparé pour alimenter l'afficheur.

Beaucoup de pouvoir

La version que j'ai entre les mains convient pour des tensions continues jusqu'à 100 V et des courants jusqu'à 30 A. Il existe une version qui supporte jusqu'à 100 A, mais tant que vous n'avez pas à mesurer des batteries de voiture ou des courants de cet ordre, les 30 A suffiront. Le module de mesure est monté dans un boîtier en plastique dont le dessus présente plusieurs ouvertures pour le passage des câbles et de l'alimentation au bornier à vis dans la boîte. Deux cartes sont montées l'une sur l'autre. Celle du haut réunit les borniers à vis et un grand relais, celle du dessous contient l'électronique de commande et un émetteur-récepteur de 2,4 GHz pour la communication sans fil avec le module d'affichage. Sa puce est cachée sous un point de colle, mais ça ressemble beaucoup à un module WiFi ESP8266. La moitié supérieure du boîtier est facile à détacher, ce qui vous permet d'accéder correctement aux borniers. Un tel module est destiné à être inséré en parallèle ou en série à un point d'un circuit où vous souhaitez mesurer tension et courant. Le module de mesure a besoin de sa propre tension d'alimentation de 12 V. Un capteur de température peut être connecté à la carte selon les besoins.

Afficher

Le module d'affichage peut être connecté de deux façons. Soit directement au module de mesure par le câble USB fourni, soit par la connexion sans fil intégrée. Celle-ci est établie automatiquement dès que la connexion USB est déconnectée. Il est possible d'utiliser plusieurs modules conjointement. Selon le fabricant, 26 canaux peuvent être utilisés simultanément. En cas d'utilisation sans fil, le module d'affichage doit toutefois être doté de sa propre alimentation électrique. Soit via le bus micro-USB (5 V) soit via un connecteur JST bipolaire (de 8 à 16 V), pour lequel un câble approprié est fourni. L'afficheur d'environ 3x2,5 cm est lumineux, très lisible même en pleine lumière. À côté se trouvent trois boutons de commande tactiles. Par défaut, le module affiche tension, courant et temps mesurés depuis le moment où la mesure est lancée. Sur la gauche, un symbole de pile indique la capacité restante de la pile si on en a saisi auparavant la capacité nominale. En haut, vous pouvez voir le type de connexion, l'état du relais et la température mesurée. Lorsque l'on appuie sur l'un des boutons, de plus amples informations apparaissent. La tension et le courant indiquent maintenant la puissance fournie ou absorbée. Dans une zone plus petite située en dessous, nous voyons le nombre d'ampères-heure fournis par la source ou la batterie, le nombre de watts-heure et le temps de consommation écoulé. Sur la droite se trouve une liste avec un certain nombre d'éléments de menu que vous pouvez parcourir

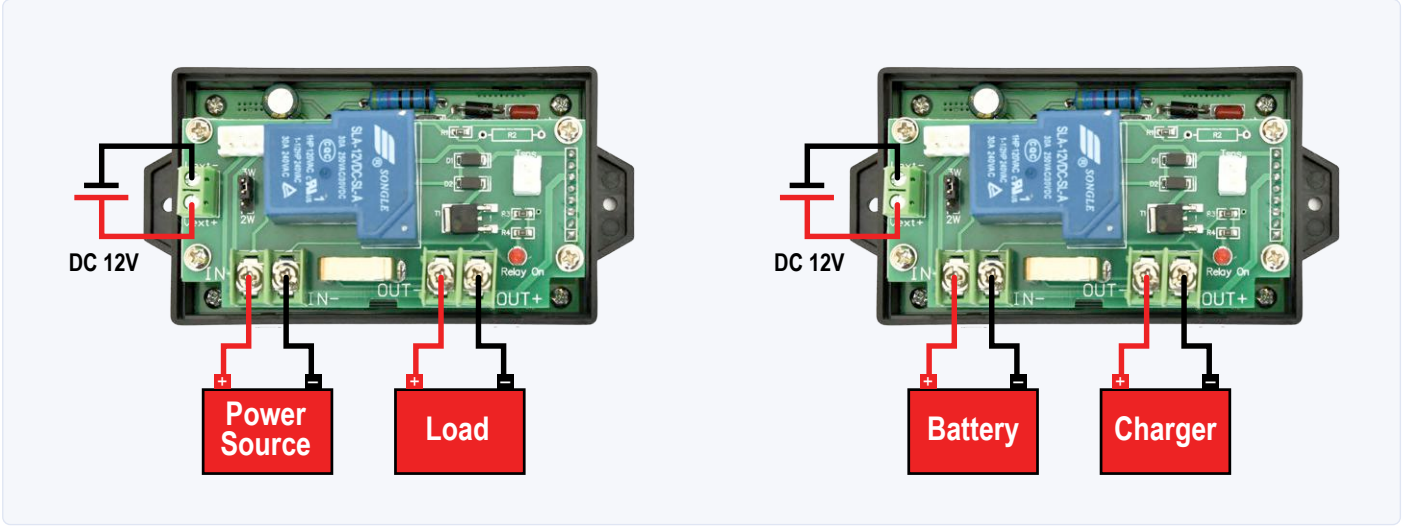


Figure 3. La charge de sortie peut être remplacée par un chargeur de batterie, et on mesure dans les deux sens.

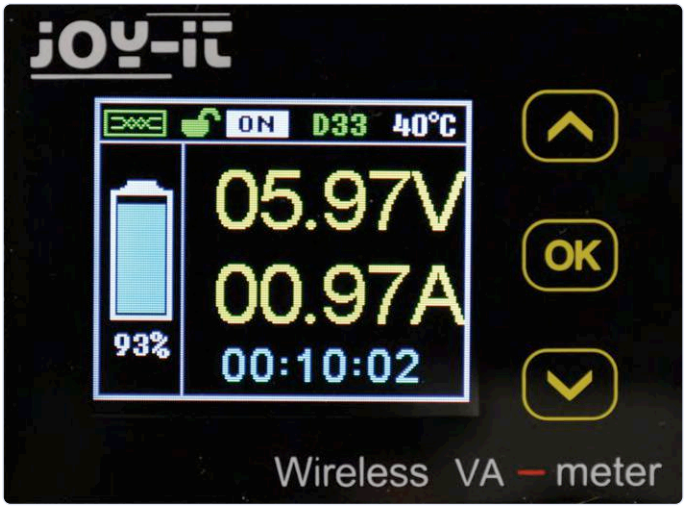


Figure 4. L'affichage par défaut.

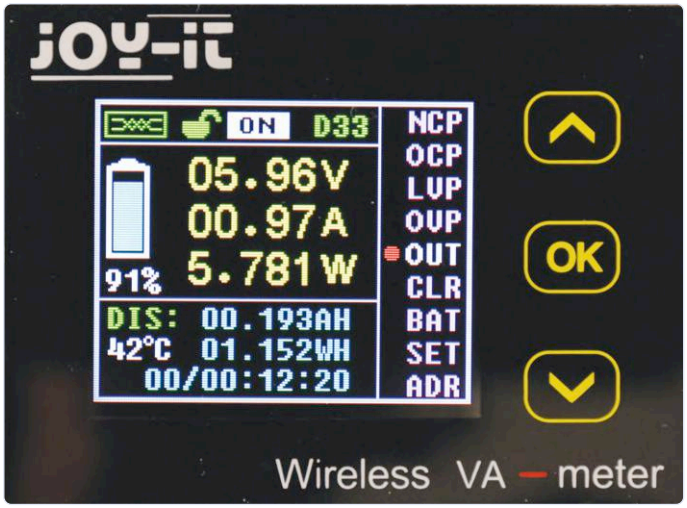


Figure 5. Touchez l'un des boutons, pour obtenir une vue étendue avec une liste de menus.

avec le bouton haut et bas. Un certain nombre de ces points seront abordés plus tard.

Possibilités d'application

Cette combinaison de module d'affichage et de mesure peut être utilisée à des fins diverses. Par exemple, vous pouvez mesurer le courant fourni à une charge par une alimentation. Si celle-ci est une batterie ou un accumulateur, vous pouvez utiliser le module pour suivre la quantité d'Ah délivrée. Mais cela vaut aussi en sens inverse, lorsque la charge est remplacée par un chargeur de batterie. Vous pouvez alors mesurer la quantité d'énergie emmagasinée. Il y a là des menus intéressants, en particulier pour l'utilisation de piles. Vous pouvez fixer un seuil inférieur et un seuil supérieur au-delà desquels le relais s'éteint. Cela permet d'éviter une décharge profonde de la batterie pendant la charge. Il est également possible de fixer une valeur maximale de courant positif (de l'alimentation à la charge) et négatif (du chargeur à la batterie) lorsque le relais s'éteint. De plus, grâce à cette combinaison, vous pouvez facilement suivre sans calculatrice la consommation d'énergie d'un circuit connecté ou la quantité d'énergie absorbée par une batterie pendant la charge. Par ailleurs, le symbole de la pile peut également être désactivé si cela n'est pas nécessaire dans telle ou telle application.

Pratique

Le manuel fourni par JOY-iT contient suffisamment d'informations pour permettre à l'utilisateur de démarrer, mais pour connaître toutes les possibilités, je recommande d'essayer vous-même les différents éléments du menu ; vous découvrirez alors rapidement la fonction ou l'opération correspondante.
La tension et le courant mesurés sont affichés par deux chiffres avant et deux après la virgule. Selon le manuel, la précision de la tension est de ±2 % et celle du courant de ±5 %. C'est raisonnable par rapport à un multimètre précis, mais il est dommage qu'en l'absence de tension d'entrée le module d'affichage affiche encore une faible valeur de tension. C'est moins de 2 % d'erreur, mais un peu étrange.
Sur le circuit imprimé de connexion du module de mesure se trouve un cavalier à deux positions : 2 W et 3 W. Ce cavalier n'est pas mentionné dans le manuel, mais après quelques recherches, il apparaît qu'il sert à la sélection de la tension d'alimentation séparée de 12 V. En position 3 W, ces 12 V sont nécessaires pour que le module de mesure fonctionne. Dans la position 2 W, le module peut dériver cette tension de la source d'alimentation connectée, à condition que la tension soit comprise entre 10 et 30 V. Chez moi, cela a déjà fonctionné à partir de 8 V.
La connexion sans fil est très pratique pour surveiller à distance. J'ai alimenté le module d'affichage à partir d'une petite banque d'énergie et j'ai pu me promener dans la maison avec le module. Il faut cependant garder un œil sur la distance, car la portée (max. 10 m) diminue rapidement en présence de murs entre le module de mesure et d'affichage. Mais c'est facile à suivre grâce à l'indicateur de puissance du signal affiché.

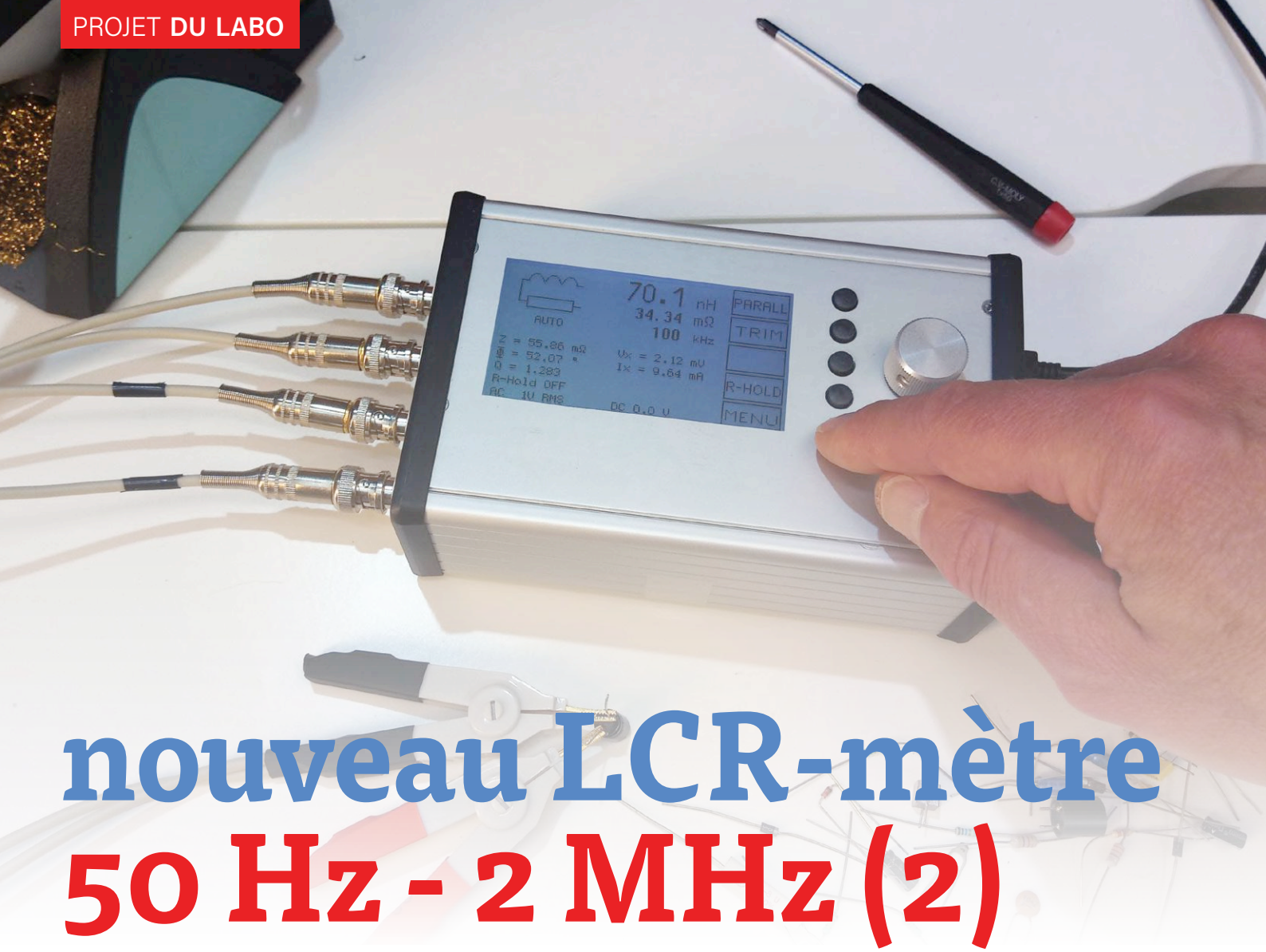
Conclusion

Pour mesurer le courant et la tension dans toutes sortes de circuits, cette combinaison de modules de mesure et d'affichage est très pratique et abordable. Grâce à sa plage de courant jusqu'à 30 A, il convient aux circuits où l'intensité des courants est forte. L'afficheur déporté et sans fil permet de lire les données à un autre endroit que celui où se trouve le circuit. C'est un avantage précieux et vraiment inattendu à ce prix. Ce duo vaut largement son prix d'environ 40 € !

200571-03

PRODUITS

› **JOY-iT VAX-1030 Wireless Multifunction Meter**
www.elektor.fr/joy-it-vax-1030-wireless-multifunction-meter



nouveau LCR-mètre 50 Hz - 2 MHz (2)

Fonctionnement, étalonnage et microprogramme

Jean-Jacques Aubry (France)

Le principe de mesure, l'étalonnage du logiciel et le matériel de ce nouveau LCR-mètre ont été abordés dans la première partie de cet article. Ce second volet couvrira l'interface utilisateur, l'étalonnage et la programmation du micrologiciel de l'AU2019.

Dans la première partie de cet article, nous avons vu comment fonctionne l'AU2019, comment il mesure les impédances, les principes d'étalonnage et de compensation utilisés et le matériel nécessaire. Il est temps d'examiner cet appareil du point de vue de l'utilisateur.

Fonctionnement du LCR-mètre

L'AU2019 peut être utilisé comme un appareil de mesure autonome, grâce à son afficheur LCD, son encodeur rotatif et ses boutons, mais il peut également être commandé par un ordinateur à travers une liaison USB.

Mode autonome

Si l'appareil est équipé de l'extension d'*affichage*, il fonctionne sans ordinateur. Utilisé avec une alimentation USB externe de 5 V, il démarre automatiquement dans ce mode. S'il est alimenté à partir d'un PC, mais que vous

Tableau 1.

							50	60				*Hz*
100	120	150	200	250	300	400	500	600	700	800	900	*Hz*
1,0	1,2	1,5	2,0	2,5	3,0	4,0	5,0	6,0	7,0	8,0	9,0	*kHz*
10	12	15	20	25	30	40	50	60	70	80	90	*kHz*
100	120	150	200	250	300	400	500	600	700	800	900	*kHz*
1,0	1,2	1,5	2,0									*MHz*

voulez toujours l'utiliser en mode autonome, vous devrez appuyer sur le bouton du haut avant de l'alimenter (le nouveau mode est mémorisé et aucune autre action n'est requise au prochain démarrage). La même action est nécessaire pour revenir au mode PC.
Lorsque l'écran de mesure est affiché, le bouton rotatif modifie la fréquence de mesure. Les 54 valeurs prédéfinies sont données par le **tableau 1** au bas de la page ci-contre.
Une pression longue sur le bouton rotatif vous permet d'entrer dans le menu de sélection d'une *fréquence utilisateur* (à partir de la fréquence actuelle). Le chiffre à modifier est sélectionné à l'aide des deux boutons supérieurs, et le bouton rotatif change la valeur. Après validation, c'est cette valeur qui est utilisée (elle s'affiche en négatif) et ne peut plus être modifiée par le bouton rotatif.
Une nouvelle pression longue sur le bouton rotatif est nécessaire pour sortir de ce mode. Si la valeur est sauvegardée, elle sera proposée lors de la demande suivante.
Les étiquettes des boutons sont affichées et modifiées en fonction des paramètres que vous choisissez.
Pour certains boutons, l'action est différente selon la durée de la pression. Si elle est longue, l'afficheur clignote brièvement pour signaler que cet état est détecté.

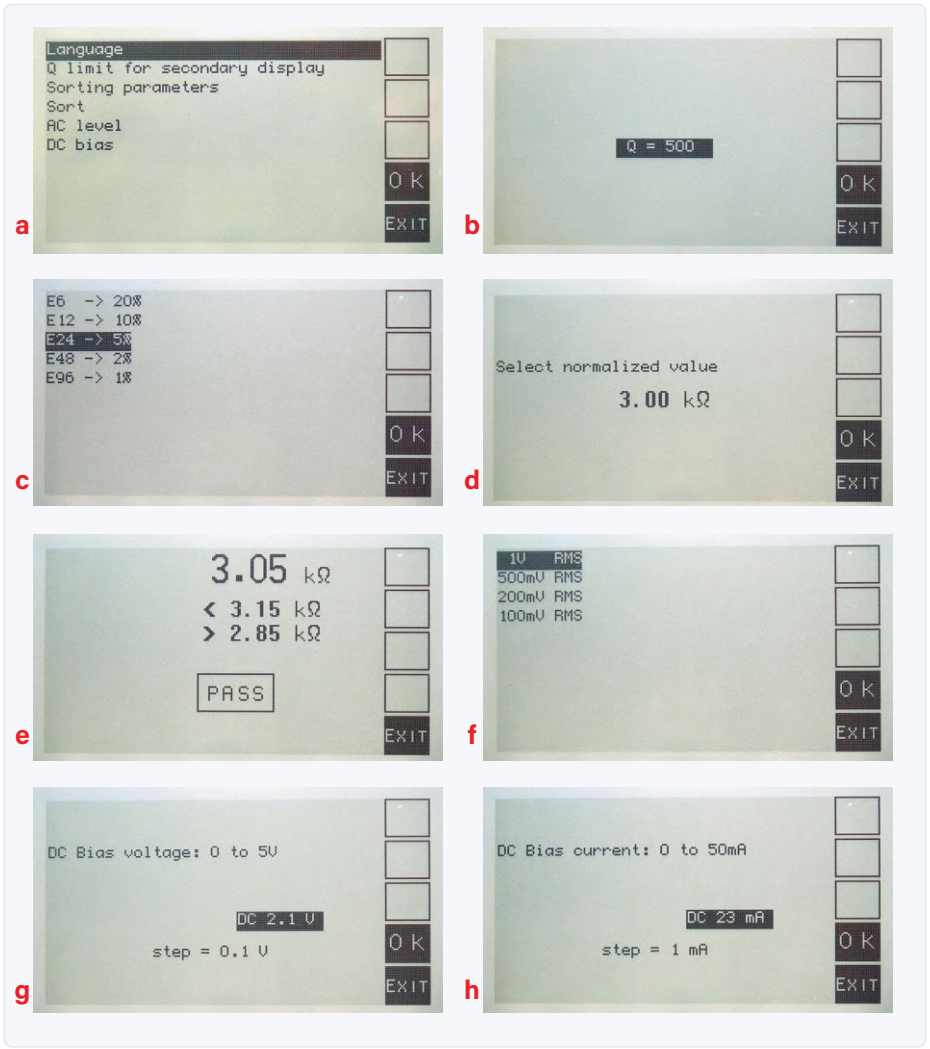


Figure 1. Le menu utilisateur.

- Le 1er bouton (en haut) permet de changer la représentation des composants : de *AUTO* (mode par défaut) à *SERIES* ou *PARALLEL*.
- Le 2e bouton *[TRIM]* lance la procédure *TRIM* **pour la fréquence actuelle** (*OPEN* ou *SHORT selon l'*impédance mesurée). Une *pression longue* sauvegarde toutes les actions TRIM effectuées depuis le dernier démarrage.
- Le 3e bouton *[AVG S]* ou *[AVG F]* modifie le nombre de mesures utilisées (moyennage) pour l'affichage des valeurs, et donc la réactivité de l'instrument. Les deux modes d'affichage alternent : *lent* (**Average SLOW**) ou *rapide* (**Average FAST**). Cette fonction n'est apparue qu'avec *la version 2.0.0 du firmware*, mais la plupart des captures d'afficheur avaient été faites avant cette mise à jour !
- Le 4e bouton *[R-HOLD]* fige la gamme actuelle. Une *pression longue* affiche les réglages du LCR-mètre : la plage (R1 à R4), le gain PGA en mesure de la tension (Ux) et de celle du courant (Ix) respectivement (avec x = 1, 3 ou 10).
- Le 5e bouton *[MENU]* : une *pression courte* ouvre le menu des paramètres de mesure de l'utilisateur, une *pression plus longue* ouvre le menu d'étalonnage.

Menu utilisateur (fig. 1)

- **Langue (1a)**

Langue des menus : anglais, français, allemand et néerlandais (à ce jour)

- **Q limite pour affichage secondaire (1b)**

Choix de la valeur du Q (= 1/D) au-dessus de laquelle le paramètre secondaire n'est pas affiché. Les valeurs possibles sont :
100 200 500 1000 2000 5000 no limit

- **Paramètres de tri (1c)**

Ce menu permet de sélectionner les paramètres (tolérance et valeur) pour le tri des composants. La valeur proposée sera celle du paramètre primaire du composant de référence connecté *avant de passer en mode menu*. Ces paramètres seront mémorisés. La première étape (**1c**) permet de sélectionner la tolérance standard, de la série E6 (20 %) à E96 (1 %). La deuxième étape permet de sélectionner la valeur standardisée dans cette série (**1d**).

- **Tri (1e)**

Ce menu lance la comparaison, avec l'affichage de la valeur mesurée (en haut), les limites supérieure et inférieure, et bien sûr le résultat : *PASS* = succès ou *FAIL* = échec.

- **Amplitude de la sinusoïde (1f)**

Ce menu permet de choisir l'amplitude du signal de test hors charge. Les valeurs possibles sont :
100 mV 200 mV 500 mV 1 V (valeur efficace)

- **Polarisation continue (1g/h)**

Ce menu permet d'ajuster la tension (pour le condensateur) ou le courant (pour l'inductance) de la polarisation continue. Les valeurs possibles sont :
– 0 à 5,0 V, par pas de 1 V ou 0,1 V
– 0 à 50 mA par pas de 10 mA ou 1 mA

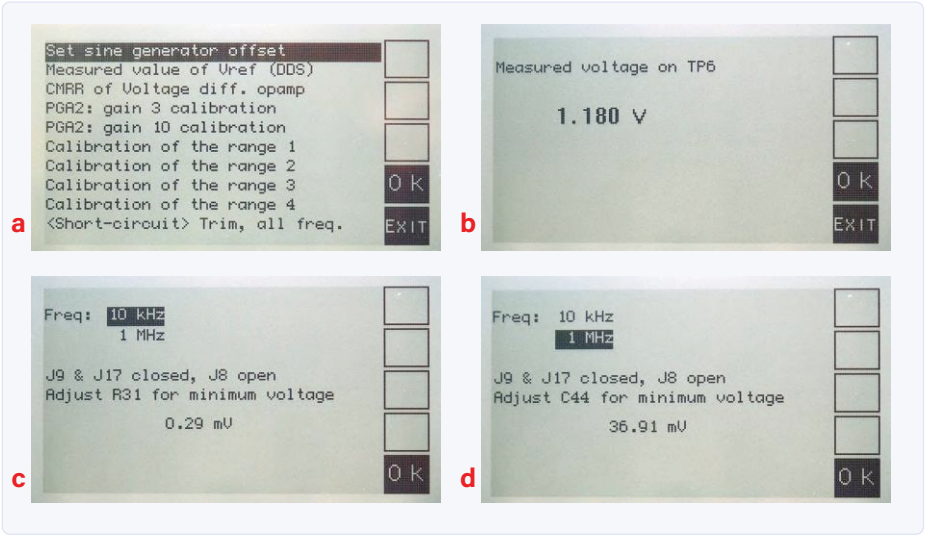


Figure 2. Le menu d'étalonnage.

Une pression sur le bouton rotatif modifie la valeur du pas.
Pour les inductances, après validation et quelques secondes pour la stabilisation, l'intensité réelle du courant est affichée sous l'étiquette <DC>, car elle dépend de la résistance du composant testé. L'étiquette du second bouton change, car le TRIM n'est plus possible avec la polarisation continue. Elle devient *BIAS 0* et le fait d'appuyer sur le bouton désactive la polarisation.

Menu d'étalonnage (fig. 2)

Toutes les étapes d'étalonnage de l'appareil sont accessibles dans ce menu. Pour un premier étalonnage, il est recommandé de les exécuter dans l'ordre du menu. Utilisez des cavaliers plaqués or pour éviter les résistances parasitaires et déconnectez tous les câbles des connecteurs BNC. La procédure complète d'étalonnage est décrite dans la documentation téléchargeable [1]. **Elle n'est pas difficile du tout** et ne requiert qu'un multimètre, des cavaliers de configuration, un petit outil de réglage… et un peu de patience, car certaines étapes durent plusieurs minutes. Nous n'en donnons ici qu'un aperçu.

- **Décalage du générateur de sinus (2a)**
- **Valeur mesurée de la V_{Ref} (DDS) (2b)**
- **CMRR de l'ampli différentiel de mesure U (2c/d)**

Les étapes suivantes durent un peu. À la fin de chaque étape, la sauvegarde des nouvelles valeurs doit être confirmée. Veuillez noter que les anciens paramètres seront effacés, même si les nouvelles valeurs ne sont PAS sauvegardées.

- **PGA2 : étalonnage du gain 3 suivi de l'étalonnage du gain 10**
- **Étalonnage des gammes 1 à 4**
- TRIM **<circuit ouvert>, toutes fréquences**
- TRIM **<court-circuit>, toutes fréquences**

À la fin de l'étalonnage, retirez tous les cavaliers et connectez les câbles Kelvin ou les appareils d'essai aux connecteurs BNC. Répétez la procédure de TRIM (en circuit ouvert et en court-circuit) pour tenir compte des nouveaux composants parasites.
Avant un test, à une fréquence **donnée**, il est possible et même recommandé d'effectuer les deux corrections TRIM en appuyant sur le bouton correspondant (le second). Les nouveaux paramètres (ainsi que ceux effectués à d'autres fréquences) peuvent être sauvegardés par une pression longue sur ce même bouton.

Messages d'erreur

Au cours de la séquence de démarrage du microprogramme, certaines vérifications ont lieu. Tout d'abord, le chargeur d'amorçage vérifie l'intégrité du programme en

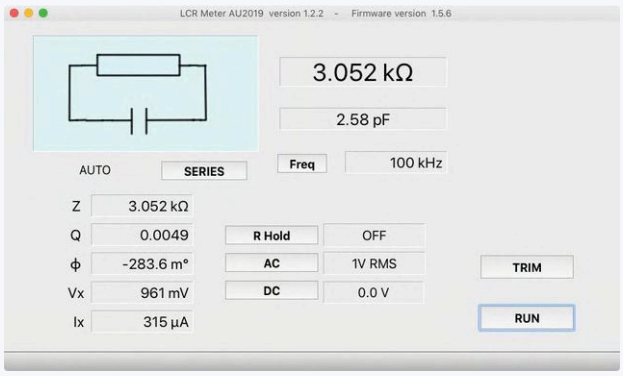


Figure 3. Le LCR-mètre commandé par l'application AU2019

Tableau 2.

Bit	Ux	Le bit est à 0 si la tension mesurée est
1	-5V	–5,3V < Ux < –4,7V
2	+5V	4,7V < Ux < 5,3V
3	+3V	2,85V < Ux < 3,15V
4	+6,5V	6,29V < Ux < 6,68V
5	+7,5V	7,30V < Ux < 7,60V

mémoire. Ensuite, celui-ci vérifie la présence en mémoire vive de données linguistiques valides et vérifie les tensions d'alimentation (**tableau 2**). Si la vérification de l'alimentation échoue, le message *Power supply Test Error, code A* est affiché et le programme tourne en boucle tandis que la LED D12 clignote, allumée pendant ½ s et éteinte pendant 1 s. Chaque bit du nombre équivalent binaire de A indique l'état d'une des tensions d'alimentation vérifiées : 0 si la tension est correcte, sinon 1.

Mode PC

Démarrez l'appareil en mode PC, il indique *Waiting for the GUI…* et quelques secondes plus tard, lancez le programme AU2019 sur le PC (**fig. 3**). Dans le menu *Port*, sélectionnez le bon port série, puis cliquez sur le bouton *Open port* (= *ouvrir le port).* Une fois la communication établie, la version du micrologiciel sera affichée à droite de la version de l'application AU2019 du LCR-mètre. Après un court laps de temps, apparaissent tous les paramètres du DUT (*device under test* = le composant testé). Les boutons agissent comme pour le mode autonome.

Certains menus sont spécifiques au mode PC, d'autres sont les mêmes que ceux du mode autonome, tout cela est expliqué dans la documentation. Il y a également des instructions pour la mise à jour du micrologiciel et du texte des menus. Une description complète se trouve dans le manuel de l'utilisateur [1].

Comment mesurer ?

Lors des mesures, les cordons entre le composant testé et l'appareil exercent une influence prédominante en fin de plage de mesure et/ou à des fréquences élevées. Une attention particulière sera accordée aux procédures *TRIM.* Pour que la mesure soit précise lorsque le DUT est une inductance de petite valeur ou une résistance, il est important que la compensation de *TRIM en court-circuit* soit effectuée avec soin juste avant la mesure. Il en va de même pour mesurer une faible capacité, avec dans ce cas la compensation de *TRIM en circuit ouvert.*

COMPARAISON AVEC DES APPAREILS DE MESURE DE FABRICATION INDUSTRIELLE

Pour comparer notre LCR-mètre AU2019 avec des appareils du commerce, il faut disposer de moyens que ni le concepteur ni le labo d'Elektor n'avaient, entre autres parce qu'un LCR-mètre capable de mesurer à des fréquences jusqu'à 2 MHz, ça ne se trouve pas facilement. Grâce au concours aimable de notre auteur Alfred Rosenkränzer dont nos lecteurs apprécient les articles fouillés, le LCR-mètre AU2019 a pu être comparé à un Hameg HM8118, à un HP/Agilent 4263B et à un Keysight U1732A portable. C'est le Hameg qui, avec ses 200 kHz, a la plus haute fréquence de mesure. Les conclusions générales sont les suivantes :

L'AU2019 se présente comme un bon appareil de mesure, avec des difficultés à certains endroits, p.ex. avec de petites selfs à la fréquence de mesure de 100 Hz, qui n'est certainement pas la fréquence idéale pour mesurer de faibles inductances, quand bien même le Hameg et le HP y ont produit des valeurs assez sensées.

La précision globale de l'AU2019 est de 1 % et elle atteint 0,1 % dans le milieu de la plage de mesure, avec des conditions de mesure correctes en fonction du type et de la valeur du composant.

La bonne connexion entre le composant et le LCR-mètre est la clé d'une mesure stable. Il ne faut donc surtout pas se priver des câbles Kelvin ni du terminal de test TH26001A 4 que l'on trouve heureusement à bas prix sur eBay, mais il est difficile de dire à quel point la qualité est bonne. L'appareil de test 16047E (même type d'appareil que le TH26001A) est très utile.

Résistance (fig. 4)

Dans la plupart des cas, les résistances peuvent être mesurées avec un bon multimètre, mais quand il faut connaître l'induc-

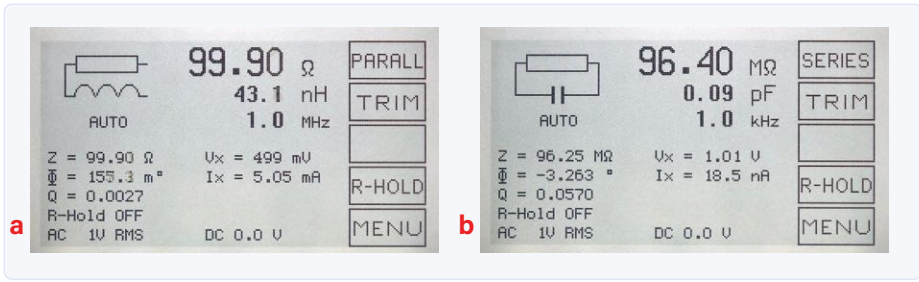


Figure 4a. Mesure d'une faible résistance à haute fréquence.

Figure 4b. Mesure d'une forte résistance à basse fréquence.

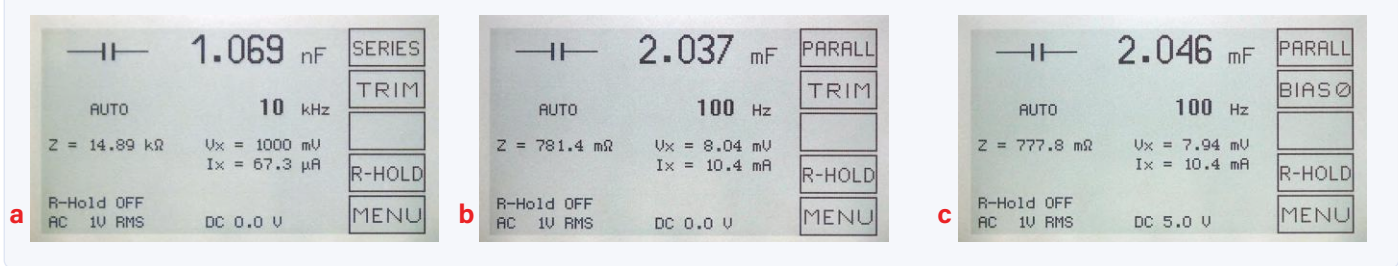


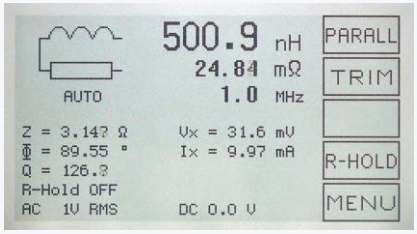
Figure 5a. Mesure d'un condensateur de faible capacité à fréquence élevée.

Figure 5b. Mesure d'un condensateur de forte capacité à basse fréquence.

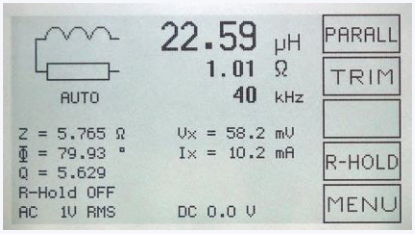
Figure 5c. Mesure d'un condensateur de forte capacité à basse fréquence, avec polarisation continue de 5 V.



a



b



c

Figure 6a. Mesure d'un inducteur avec un courant de polarisation.

Figure 6b. Mesure d'un inducteur à une fréquence plus élevée.

Figure 6c. Mesure d'un inducteur sans courant de polarisation.

PREMIER DÉMARRAGE DE L'APPAREIL AVEC MICROCONTRÔLEUR VIERGE

Lors de la première mise sous tension, il vous faut un PC avec *Windows*, et l'adaptateur de débogage USB de Silicon Labs (p.ex. RS référence 757-0297) pour charger le chargeur d'amorçage dans le microcontrôleur via l'interface JTAG (J15). Vous devez avoir téléchargé le fichier *MergedLCR6.hex*, qui comprend à la fois le chargeur d'amorçage (*boot_LCR6.hex*), et le microprogramme *LCR6.hex*. Les procédures et instructions complètes figurent dans le manuel de l'utilisateur [1].

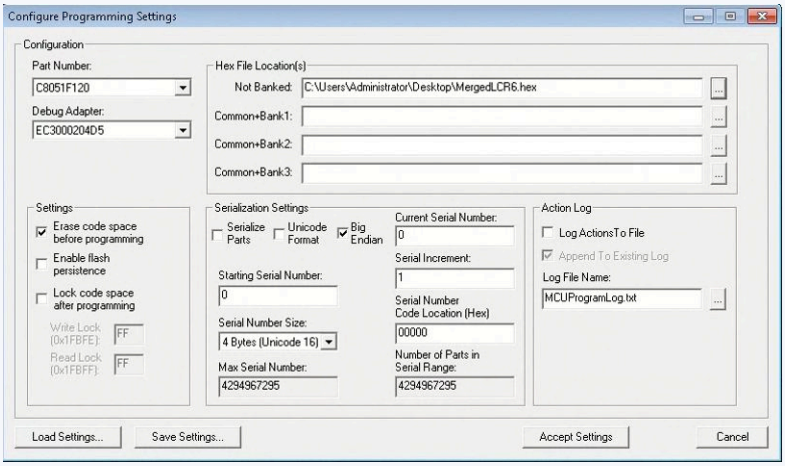


Figure 7a. Paramètres de programmation dans le MCU Production Programmer de Silicon Labs.

PREMIER CHARGEMENT DU MICROPROGRAMME

Connectez le câble de l'adaptateur de débogage USB à J15 sur la carte et le PC puis mettez SW1 sur ON. Lancez le programme *MCUProductionProgrammer* [3] :

- Dans *Program Menu / Configure Programming Information...* renseignez les différents champs (**fig. 7a/b**), puis *Accept Settings* (vous pouvez sauvegarder ces réglages par *Save Settings...*)
- Cliquez sur *Program Device* pour lancer l'opération
- Une fois la programmation terminée, apparaît le message *Device Programmed and Verified* dans la fenêtre *Status Log*
- Ensuite, il faut charger les textes pour l'affichage (fichier de langue) en suivant les instructions du manuel de l'utilisateur §4.2.5

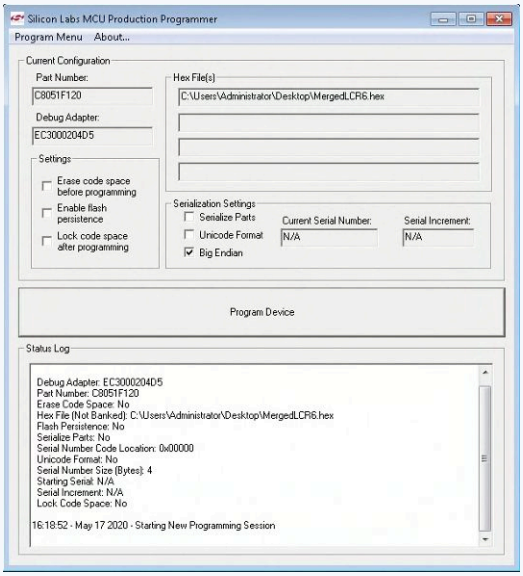


Figure 7b. Écran de programmation du MCU Production Programmer de Silicon Labs.

LIENS

[1] **1ère partie de l'article – documentation et téléchargement de logiciels** : www.elektormagazine.fr/190311-01

[2] **La page du projet du LCR-mètre sur Elektor Labs** : www.elektormagazine.com/labs/remake-lcr-meter

[3] **Téléchargement du MCU Production Programmer de Silicon Labs** : www.silabs.com/documents/login/software/MCUProductionProgrammer.zip

tance série parasite à haute fréquence d'une résistance de faible valeur, ou la capacité parasite d'une résistance de grande valeur, alors on sort son LCR-mètre.

Condensateur (fig. 5)
Pour un condensateur, il est possible de superposer au signal de test une tension continue entre 0,0 V et 5,0 V, voir le paragraphe sur la polarisation continue. Le pôle positif de cette tension de polarisation est sur J4/J5. La capacité doit être la partie dominante de l'impédance, sinon le LCR-mètre va mettre automatiquement la tension de polarisation à zéro ! Pour mesurer un condensateur électrolytique de forte capacité à l'aluminium, il est important de choisir une fréquence de test appropriée au composant et qui soit inférieure à sa fréquence de résonance.

Inductance (fig. 6)
Pour une inductance, il est possible de superposer au signal d'essai un courant continu entre 0 mA et 50 mA (cf. § 3.1.5) Il faut veiller à ce que la tension continue qui règne aux bornes de la résistance interne en raison de ce courant de polarisation reste inférieure à 0,3 V, sinon le circuit de compensation automatique d'offset de l'amplificateur différentiel U7 ne fonctionne plus correctement.

REMARQUE : l'application d'une polarisation de 50 mA entraîne une augmentation d'environ 210 mA du courant total consommé par l'appareil. Assurez-vous que votre source d'alimentation peut le fournir.

Pour les petites inductances RF (dont la fréquence de résonance est très élevée), assurez-vous que la mesure est effectuée à une fréquence suffisamment élevée, de sorte que la partie dominante du DUT soit vue comme une inductance ! ⏮

190311-B-03

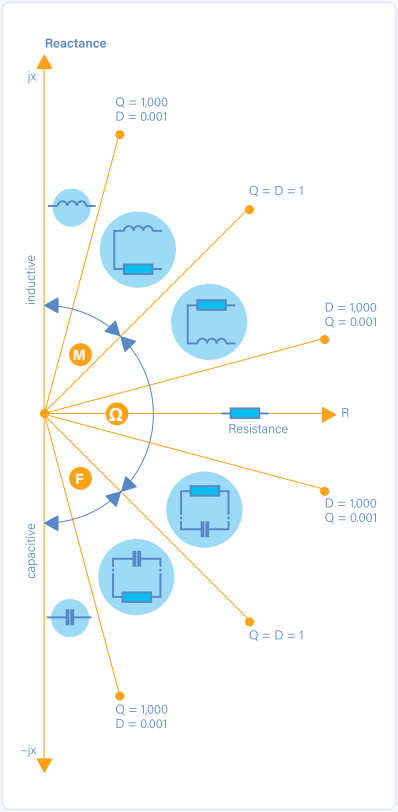


Figure 8. Circuit équivalent basé sur le facteur de qualité et la composante réactive. (source : Fluke)

DÉTERMINER LE TYPE DE COMPOSANT ET LE CIRCUIT ÉQUIVALENT

Cet appareil est conçu pour mesurer trois types de composants : résistances, condensateurs, inductances. Pour chacun des types, les conditions de mesure sont à choisir en fonction de sa valeur et de son utilisation. Outre la valeur du paramètre principal, qui caractérise le type de composant, celui-ci possède également un paramètre secondaire qui dépend principalement de la fréquence de test. L'appareil évalue la nature du composant sur la base de son équation équivalente :

$Z = Rs + jXs$

à partir de laquelle il calcule le facteur de qualité :

$Q = |Xs|/Rs$

En mode AUTO, le critère de décision pour la définition du paramètre principal et du circuit équivalent est la valeur de Q (=1/D) et le signe de la composante réactive Xs. Voir la **fig. 8** pour la relation entre Q, Xs et le circuit équivalent.

Suivez ce projet sur la page Elektor Labs [2], où l'auteur répond aux questions et commentaires.

Votre avis, s'il vous plaît ?
Vos questions et vos commentaires sur cet article sont bienvenus par courriel adressé à l'auteur jjacques.aubry@free.fr ou à redaction@elektor.fr



PRODUITS

› **Elektor "Kickstarter Project":**
Kit comprenant une carte mère + une carte d'extension (affichage) + tous les composants
www.elektor.com/lcr

Ont contribué à cet article
Auteur : **Jean-Jacques Aubry**
Rédaction : **Luc Lemmens, Denis Meyer**
Illustrations : **Kurt Diedrich, Patrick Wielders**
Mesures comparatives : **Alfred Rosenkränzer**
Maquette : **Giel Dols**

erreurs fécondes

Conseils sur les régulateurs de tension et sur la conception des circuits imprimés, etc.



Compilé par **C. J. Abate** (Elektor)

Les seules vraies erreurs sont celles dont personne ne tire aucune leçon. Les électroniciens de tout poil savent qu'ils ont chacun beaucoup à apprendre de ses propres erreurs, mais aussi de celles des autres. Nous sommes tous demandeurs de partage et de retour d'expérience de conception, de programmation, de test, etc.

Un sujet bateau : la régulation de tension

Vous passez des heures sur un circuit, tout va comme sur des roulettes, puis, pour une bourde idiote, tout part en fumée… Vous connaissez ça, tout comme S. Coppersmith, ingénieur à l'université de Notre Dame à South Bend, Indiana, États-Unis. Il nous fait part de son expérience, avec un bon conseil.

«J'avais imaginé un bel éclairage pour mon bateau fait de LED que j'ai enfilées patiemment dans des tubes translucides. Tout était calculé pour 13 V_{CC} mesurés pendant que le moteur du bateau fonctionnait. J'ai ensuite fait l'erreur de démarrer le moteur sans que la batterie soit connectée et toutes les LED ont grillé à l'unisson quand la sortie de l'alternateur est passée à 21 V_{CC}. Des heures de travail parties en fumée. La sortie d'une alim auto ou bateau peut varier énormément, de 8 à 18 V ou plus, selon l'état de la batterie. Il faut donc des circuits de régulation de tension spécifiques pour ces applications». - *Scott Coppersmith*

EN SAVOIR PLUS SUR LA RÉGULATION DE TENSION, LES ALIMENTATIONS ET LES LED

Vous cherchez des informations sur les alimentations, les LED et les régulateurs de tension ? Voici des ressources utiles sur Elektor :

- Régulateurs de tension (www.elektormagazine.com/tags/voltage-regulators)
- Alimentations (www.elektormagazine.fr/categories/alimentation)
- C. Valens, « Comment calculer la résistance talon d'une LED [1] », ElektorMagazine.fr, août 2019 (https://bit.ly/resistance-LED)
- M. Heine, « Article inédit génial : convertisseur élévateur à LED pour µC », ElektorMagazine.fr, 2020 (https://bit.ly/booster-LED)
- C. Valens, « Commande linéaire de LED : kézako ? », ElektorMagazine.fr, février 2020 (https://bit.ly/LED-lineaire)

Tous les composants sont suspects

Vous avez besoin de conseils pour la conception de vos PCB ? Chris Clapham est ingénieur en matériel informatique à Auckland, en Nouvelle-Zélande. Après plus de 30 ans d'expérience dans le domaine des PCB, son conseil, excellent, est de toujours vérifier tous vos composants. Un défaut peut en cacher un autre.

«En cas de panne, n'oubliez jamais de mettre aussi les composants neufs sur votre liste de suspects ! J'ai travaillé au sein d'une entreprise de reconditionnement de circuits imprimés pour la télévision. Nous réparions des PCB défectueux : soudures froides, remplacement des électrolytiques et des composants qui présentaient des signes de stress thermique, etc.). Souvent nous tombions sur des pannes qui persistaient une fois les composants remplacés. Il s'agissait p. ex. de transistors mal polarisés ou mal marqués – un PNP dans un boîtier indiqué comme celui d'un NPN ou inversement, des diodes de puissance apparemment en bon état, mais qui perturbaient l'alimentation, laquelle se mettait à chauffer. Je me souviens d'une diode dont l'anode et la cathode étaient permutées. Le composant utilisé par le fabricant de TV (sans doute d'un lot obtenu à bas prix) avait sa cathode connectée à la partie métallique du boîtier (TO-220, de mémoire) en contact direct avec le radiateur lui-même à la masse (pas d'isolant). Le composant de remplacement (correctement fabriqué) avait son anode connectée au radiateur qu'elle court-circuitait donc. Les dépanneurs remplaçaient l'original par un composant neuf qui causait un court-cir-

> En cas de panne, les composants neufs ont aussi leur place sur la liste des suspects !

cuit, sans comprendre pourquoi la platine restait défectueuse. Sans la nouvelle diode, ça fonctionnait bien. Il a suffi de rajouter une plaquette de mica. Autre incident avec une diode DO-201, court-circuitée sur le PCB et remplacée : la carte ne fonctionnait toujours pas. J'ai remarqué que, par rapport à un exemplaire du PCB en état de marche, les fils du multimètre devaient être intervertis pour que la diode conduise. La diode neuve était bien orientée sur le PCB, c'est la polarité de la jonction dans le corps de la diode qui était inversée. Il ne suffit pas de vérifier la polarité de la diode, il faut encore qu'elle soit mise dans le bon sens sur le PCB. Ne pas se fier aveuglément au marquage ni du boîtier ni du PCB». - *Chris Clapham*

200562-04

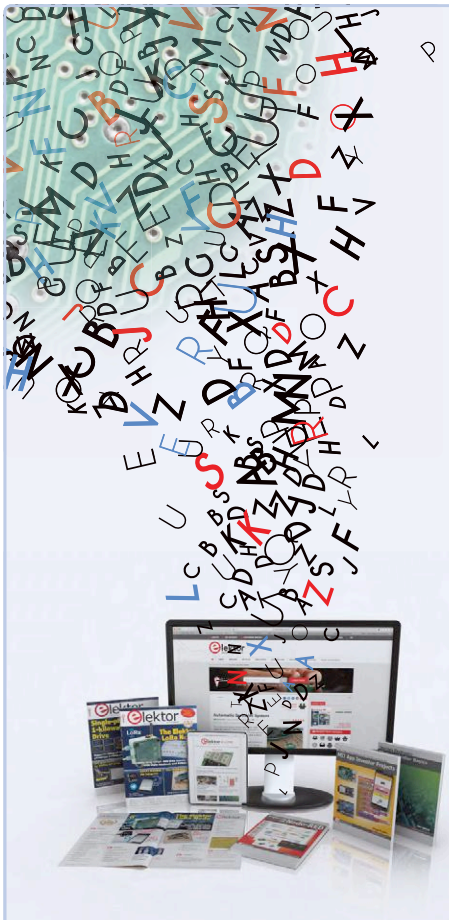
Partagez votre expérience d'analyse d'erreur
Vous souhaitez partager vos erreurs en matière d'électronique et les enseignements que vous en avez tirés ? Que vous soyez électronicien chevronné, amateur motivé ou étudiant, nous vous invitons à nous communiquer votre expérience en cliquant sur le lien ci-dessous :
www.elektormagazine.com/news/error-analysis-submission

POUR EN SAVOIR PLUS SUR LES CIRCUITS IMPRIMÉS, LEUR CONCEPTION ET LE PROTOTYPAGE

Si vous souhaitez des conseils et des informations utiles sur la conception et le prototypage de circuits imprimés, consultez ces ressources d'Elektor :

- ElektorTV, « PCB Layout with Pcbnew », 6/5/2020 (https://youtu.be/-_nZZLuwYd0)
- Clemens préfère KiCad EDA pour ses PCB, avec Eeschema pour ses schémas(https://bit.ly/KiCad-EDA)
- « ElektorPCB4Makers : nouveau service écoresponsable de production de PCB » 1/23/2020 (https://bit.ly/ElektorPCB4Makers-fr)
- Plugins et add-ons KiCad (http://bit.ly/kicad-plugins)
- Capture de schémas avec Eeschema (http://bit.ly/kicad-Eeschema)

Publicité



Elektor cherche des auteurs

Le coronavirus bouleverse nos vies, avec parfois des conséquences positives.

Le temps libéré, vous pouvez l'utiliser pour **partager** vos connaissances en **électronique** avec d'autres. Selon vos talents, le plus simple consiste à donner des cours **vidéo** ou à écrire un **article** ou un **livre**. Vous avez une bonne idée ? Action ! **Elektor vous assistera**. Outre la satisfaction de cette expérience, il y aura des recettes pécuniaires.

Faites-nous part de votre idée, nous vous répondrons.

elektor.fr/cherche-des-auteurs

elektor design > share > sell

corrections, mises à jour et courrier des lecteurs

Rédaction : **Ralf Schmiedel** & **Jens Nickel**

interface USB-S/PDIF

Elektor 09-10/2020, p. 6 (180027)

Le schéma (fig. 5) et la liste des composants donnent tous deux un type erroné pour IC1. C'est un PIC32MX270F256B-50I/SO et non un PIC32MX274F256B-I/SO qu'il faut. Pas de problème sur le circuit imprimé.

Clignotant éternel…ektor

Elektor 07-08/2020, p. 65 (200200)

Dans le schéma, T1 à T3 sont des BC548C (NPN), mais T3 est représenté comme PNP, ce qui est correct ! T3 doit être un transistor PNP, p. ex. un BC558.

Avez-vous quelque chose pour cette rubrique ?
Si vous avez aussi quelque chose à signaler pour cette rubrique, faites-le par courriel envoyé à l'adresse redaction@elektor.fr

comment (bien) photographier l'électronique

Elektor 07-08/2020, p. 44 (200186)

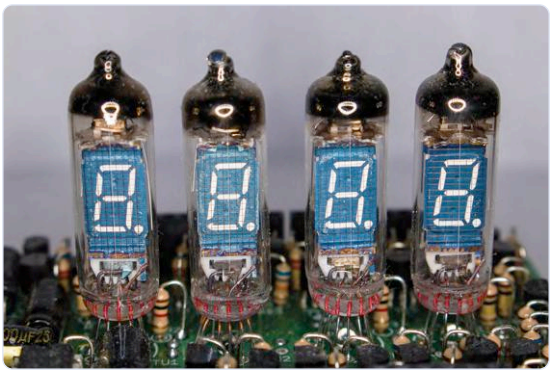
J'ai lu avec grand intérêt l'article sur les photos de circuits électroniques. On peut se simplifier la vie et laisser l'objectif dans la lampe sans que la qualité des photos en souffre. Il existe des flashs annulaires très bon marché pour tous les appareils photo reflex courants, et leur qualité reste honorable. Pour mes photos, j'utilise le flash annulaire RF-550D de Fositan. Il est disponible via Azamon pour environ 34 €. Il est fourni avec un grand nombre d'adaptateurs pour différents diamètres d'objectif. Si vous ne possédez pas d'objectif macro, vous devriez faire les frais d'un objectif pour les gros plans. Vous pourrez ainsi vous rapprocher davantage du sujet et atteindre des taux de reproduction élevés sans dégrader la qualité de l'image.

Pour augmenter la profondeur de champ, vous fermerez le diaphragme. Avec le flash annulaire, l'appareil devra fonctionner en mode manuel, l'ouverture et le temps de pose doivent être réglés manuellement. Avec la luminosité standard du flash (réglage 0) et une sensibilité de 100 ISO, j'ai eu de bons résultats avec un diaphragme de 11 au 1/60e. Si les photos manquent de luminosité, passez à 200 ISO. N'ouvrez pas trop le diaphragme, vous y perdriez en netteté.

Vous pourriez donc travailler sans trépied. Le flash annulaire mentionné fonctionne très bien avec mon Canon 80D et un zoom 18-55 mm, sans aucun réglage spécial sur l'appareil. La bague d'adaptation appropriée est vissée sur l'objectif, l'unité de commande placée sur le rail du flash. Puis montez le flash sur l'adaptateur.

Volker Schmidt





190381-E-02

Robert Lacoste : grands et petits secrets de l'électronique

hors-circuits avec Robert Lacoste



Le monde de l'électronique est à la fois vaste et tout petit ! Tout touche à tout, le plus petit détail peut avoir les plus grandes conséquences.
L'objectif de R. Lacoste, l'auteur de ***Percer les mystères de l'électronique***, est de vous donner des pistes pour comprendre et vous permettre ensuite de progresser seul. Repoussez vos propres limites et apprenez à détecter celles du matériel et du logiciel utilisés. Repassez par les notions de base et distinguez les véritables progrès techniques des laïus commerciaux.
Ce livre vous aide à y voir plus clair. Sans formule mathématique qui ne soit pas à la portée d'un lycéen, il balaye tout le champ de l'électronique.
L'auteur explique comment ça marche, pourquoi parfois ça ne marche pas, et différentes techniques pour que vos projets marchent toujours.

elektor



Auteur de plus d'une centaine d'articles dans les revues techniques spécialisées Elektor et Circuit Cellar, ***Robert Lacoste*** *a plus de trente ans d'expérience dans le domaine des signaux mixtes : acquisition et traitement du signal, radiofréquences et antennes, hyperfréquences, électronique ultra-rapide.*

Auteur : Robert Lacoste
Pages : 247 (en couleur)
Format : 17 x 23,5 cm (broché)
ISBN : 978-2-86661-207-8

Pour commander ce livre :
www.elektor.fr/19080

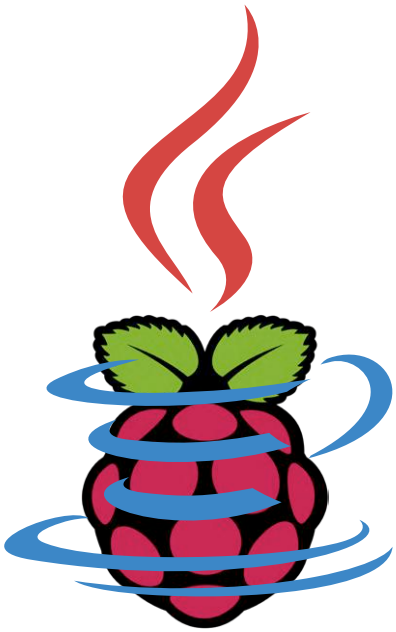
Java sur Raspberry Pi

Entretien avec Frank Delporte



C. J. Abate (Elektor)

Vous pouvez faire tourner Java sur Raspberry Pi. Dans cet entretien, l'auteur Frank Delporte parle des avantages de Java, ses projets basés sur Raspberry Pi et raconte comment il en est venu à marier la carpe et le lapin.



Faire tourner Java sur Raspberry Pi, ça vous dirait ? C'est ce que vous propose Frank Delporte. Son nouveau livre, *Getting Started with Java on the Raspberry Pi* [1], est une excellente ressource pour les programmeurs professionnels et les mordus avides d'expérimentation, désireux d'apprendre à leur propre rythme. Dans cet entretien, Delporte parle des avantages de la combinaison de Java et de Raspberry Pi, ainsi que de ses expériences en tant que programmeur.

Programmation, conception et rédaction

Abate : *Félicitations pour la publication du livre* Getting Started with Java on the Raspberry Pi *(Elektor 2020). Nous reparlerons du livre dans un instant, mais quel est ton sentiment maintenant que tous les travaux d'écriture et d'édition sont derrière toi ? As-tu aimé ce genre de travail ?*

Delporte : J'aime écrire, il n'y a qu'à voir mon blog [2], mais écrire un livre entier, c'est du travail. Il faut l'écrire, bien sûr, mais avant il faut rassembler toutes les infos, faire des recherches, expérimenter, dessiner les schémas, mener des entretiens, et après il faut relire, etc. La satisfaction de tenir enfin entre mes mains mon premier livre imprimé (**fig. 1**) me récompense largement pour tout ce travail.

Abate : *Nous avons présenté ton espace de travail à domicile [3] sur le site d'*Elektor *en mai 2020. Travailles-tu toujours à la maison en raison de la COVID-19 ?*

Delporte : Ici la situation est redevenue presque normale. Je travaille toujours à la maison, mais pas pour cause de Covid. Pour me concentrer sur une tâche précise, rien de tel que de travailler chez moi.

Abate : *Ton parcours est intéressant, à la fois concepteur de logiciels, responsable technique, auteur et monteur vidéo. Comment t'y es-tu pris ?*

Delporte : J'ai toujours été intéressé par la technique et le fonctionnement des choses. J'étais le genre de gamin qui démonte chaque machine à café, radio ou tout autre appareil cassé. Je n'arrivais pas à les réparer toutes, mais chaque fois j'apprenais quelque chose de nouveau ! Adolescent, j'ai participé comme DJ à une émission de radio locale, ce qui m'a donné la possibilité d'expérimenter davantage l'électronique.

C'est ainsi que j'ai décidé d'étudier dans une école (technique) de cinéma, pour apprendre les techniques du cinéma, de la radio et de la télévision, l'étalonnage les caméras et les connexions de tous ces appareils et les techniques d'enregistrement. Après mon diplôme, le montage informatique a bouleversé la production



Figure 1. Le livre de Frank Delporte : Getting Started with Java on the Raspberry Pi.

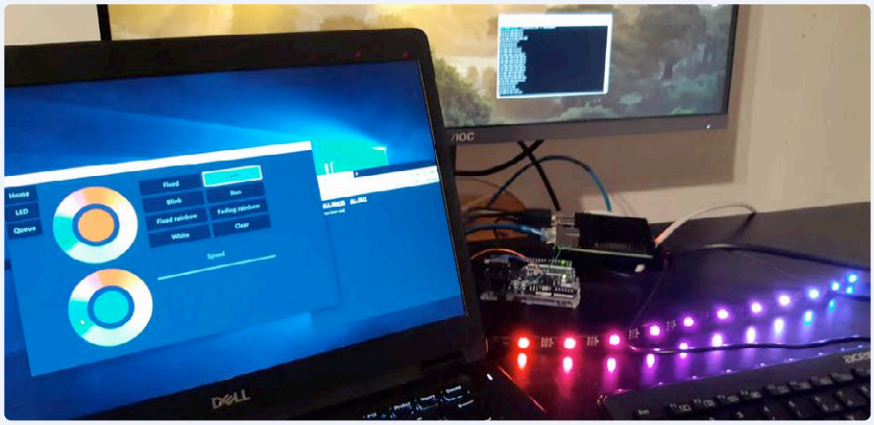


Figure 2. La configuration de la bande de LED.

vidéo, et c'est ainsi que je suis revenu à la programmation parce que des clients voulaient leur vidéo d'entreprise sur CD-ROM puis sur l'internet.

Abate : *Dans ta biographie figure le Commodore 64. Te souviens-tu de tes premières expériences ?*
Delporte : Je n'avais qu'un seul jeu sur mon C64, parce que ce qui m'intéresse, c'est la programmation. Grâce à Elektor, j'ai trouvé un livre (vers 1987) avec une carte électronique à huit relais que l'on pouvait commander en Basic sur le C64. Je l'ai utilisée pour commander mon train Lego ; des interrupteurs magnétiques reliés aux ports du joystick détectaient la position du train. Ce fut ma première combinaison réussie de logiciel et de matériel. Aujourd'hui, un tel projet serait beaucoup plus facile (et moins cher) avec un Arduino ou un Raspberry Pi et les nombreuses cartes d'extension géniales.

Abate : *Quels étaient tes objectifs de carrière en 1994 à ta sortie du NARAFI* (Nationaal Radio en Filmtechnisch Instituut) ?
Delporte : Mon premier travail a été le montage vidéo pour une TV locale, et après quelques années, puis j'ai fait ça comme pigiste. Sans objectif de carrière précis, je suis passé d'un emploi à l'autre en apprenant beaucoup en cours de route. C'est ainsi que je suis passé de monteur vidéo à développeur multimédia en créant des présentations d'entreprise sur CD-ROM puis DVD. Avec l'essor de l'internet, je suis passé au développement web car mes clients voulaient diffuser les mêmes informations sur des sites avec un système de gestion de contenu. Grâce à ces connaissances, je suis devenu développeur Java et responsable technique du développement de produits utilisant cette technologie.

Enseigner le codage

Abate : *Quand as-tu commencé à organiser des sessions de CoderDojo ? Et quel genre de cours donnes-tu ?*
Delporte : Dans chaque entreprise où j'ai travaillé, le défi a été de trouver les bons collègues sur le plan technique. Les techniques et certainement l'informatique sont encore trop masculines. Pour moi, l'ingénierie est magique. Quelques lignes de code ou quelques composants électroniques suffisent pour construire des "trucs". Pourquoi tant d'enfants, qui pourtant aiment construire et expérimenter, s'arrêtent-ils soudain de le faire et ne choisissent pas une orientation où ils peuvent continuer à inventer ?
CoderDojo [4] est un club gratuit où des bénévoles aident les jeunes de 7 à 18 ans à expérimenter des *trucs numériques*. Nous programmons en Scratch (par blocs), construisons des mondes dans Minecraft avec JavaScript, contrôlons l'électronique avec Arduino, construisons des robots avec Lego, et bien davantage. En 2013, j'ai lancé un tel club à Ypres et à Roulers, en Belgique, et je dirige toujours celui d'Ypres. La crise sanitaire nous a contraints à suspendre provisoirement nos rencontres informelles à plusieurs autour d'un PC. Grâce au CoderDojo et à d'autres initiatives STEM (Science, Technologie, Ingénierie et Mathématiques), nous constatons une progression du nombre d'étudiants en technique, garçons et filles !
Abate : *Quand t'es-tu dit : "Hé, je suis un bon formateur, et je peux aider d'autres personnes intéressées par Java" ? Ou est-ce un ami ou un collègue qui t'a indiqué la direction à suivre ?*
Delporte : J'aime expliquer et je crois fermement au principe d'*apprendre en apprenant*. C'est ce que je fais au CoderDojo avec les jeunes, mais aussi sur mon blog et au travail. Pour bien comprendre un sujet, il faut être capable de l'expliquer et vice versa. Les articles que j'écris pour mon blog naissent toujours de quelque chose que je veux essayer sans (encore) vraiment savoir comment le faire. Au fil du processus de compréhension, j'écris les étapes suivies et ce qui a fonctionné et ce qui a cafouillé. C'est ainsi que j'apprends et que j'acquiers des connaissances à partager avec d'autres.

Zoom sur Java

Abate : *Raconte-nous ton histoire avec Java. L'as-tu appris par curiosité ? Était-ce pour un cours ? Ou pour le travail ?*
Delporte : Pour produire des applications multimédias, j'ai dû apprendre *ActionScript* (et même *Lingo* avant). Plus tard, je suis passé au C# et au SQL pour les applications web. On aura compris que j'apprends surtout en expérimentant, mais aussi en lisant des livres et en suivant de courts cours (en ligne). Lorsque j'ai commencé à *Televic Rail* en 2010, j'ai rejoint une équipe qui utilisait déjà Java. Le passage de C# à Java a été très facile. Après toutes ces années de programmation, je réalise que c'est avec mes collègues que j'apprends le plus ! Partager son travail avec d'autres lors de présentations, améliorer le code avec des demandes d'extraction, accepter des commentaires comme moyen d'amélioration sont autant d'excellents moyens d'apprendre des autres.
Depuis que j'ai commencé à expérimenter en Java sur Raspberry Pi, je me suis impliqué dans des projets et des discussions sur les logiciels libres, et c'est un monde complètement nouveau pour moi, où je rencontre beaucoup de gens brillants, également prêts à partager leurs connaissances et leur expérience. Il ne passe pas un jour sans que je sois étonné par tout ce

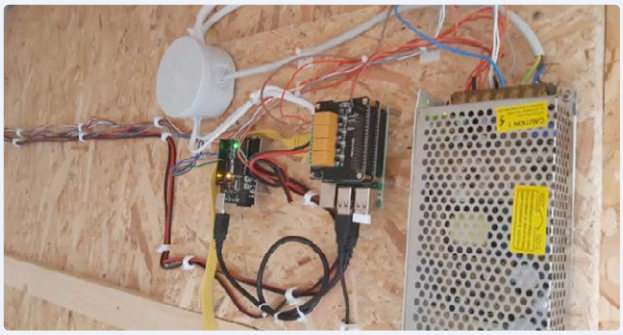


Figure 3. Les composants installés de la console de commande de la cabine de batterie.



Figure 4. Effet d'arc-en-ciel réalisé avec des bandes de LED.

> « Java n'est pas seulement un langage de programmation, c'est aussi une machine virtuelle qui exécute le code Java. »

que ces projets et ces personnes peuvent nous apprendre. Vous n'êtes pas obligé de contribuer au code, mais vous pouvez aussi rejoindre un tel projet en examinant les demandes de retrait, en aidant à tester ou à documenter le code.

Abate : *Es-tu anti-python ou anti-C ? Je suppose que non, mais je ne peux pas ne pas te poser la question.*
Delporte : Certainement pas, je me garde de ce qui est « anti ». Il n'y a pas de mauvais langages de programmation ! On ne devrait jamais perdre de vue que le meilleur outil pour faire un travail est celui que l'on connaît le mieux. Dans mon cas, c'est Java et JavaFX si je veux faire une application avec une belle interface utilisateur. Dans mon livre, j'ai aussi utilisé Python pour commander un affichage de nombres par LED et un Arduino avec des bandes de LED. Pour chaque projet (**fig. 2**), il faut décider quel est le meilleur outil, le meilleur langage de programmation ou la meilleure plateforme. Et une fois ta décision prise, vas-y ! Plus tard, tu verras que ce n'était peut-être pas le meilleur choix, mais tu auras tant appris que tu ne regretteras rien.

Travailler avec Java sur le Pi

Abate : *Tu bloques depuis 2007 sur les sujets techniques. Ton premier post sur Raspberry Pi semble être "Pong on a Raspberry Pi" [5] en décembre 2017. Parle-nous de tes premières expériences avec RPi ? Quand as-tu commencé ?*
Delporte : Quand j'ai commencé avec CoderDojo, certains moniteurs avaient déjà une expérience avec Arduino et Raspberry Pi. Ils sont venus avec leurs kits au club. La puissance de ces cartes bon marché m'a stupéfié quand j'ai compris tout ce qu'on peut en tirer en les combinant avec quelques composants électroniques.
Je bloguais déjà depuis un certain temps, mais mon premier projet Raspberry Pi "public" était en effet ce jeu de Pong que nous avons utilisé pour certaines activités de l'école de mon fils. J'ai utilisé Python pour l'interface utilisateur ; mais pour tout dire, sans grand plaisir. Pour ce genre d'application, je préfère JavaFX pour lequel il existe même un très beau cadre de jeu : FXGL [6].

Abate : *As-tu chez toi ou dans ton espace de travail des applications basées sur le RPi qui fonctionnent ?*
Delporte : J'ai commencé avec Java sur le Raspberry Pi pour construire un contrôleur de batterie [7] pour mon fils. Il s'agit d'une interface utilisateur à écran tactile permettant de commander plusieurs lumières à l'aide d'une carte à relais et des bandes de LED commandées par Arduino (**fig. 3**).
J'ai ainsi appris à utiliser la communication série entre les deux cartes et I²C pour commander les relais. Dans mon livre, j'ai étendu cette méthode et j'ai utilisé une file d'attente Mosquitto pour échanger des messages entre d'autres cartes et PC.

Abate : *Sur quoi d'autre travailles-tu ces jours-ci ? De nouveaux projets, programmes ou livres ?*
Delporte : Je poursuis mes expériences avec Java sur le Raspberry Pi, bien sûr. J'ai écrit d'autres articles de blog sur ce sujet et j'ai également expérimenté d'autres technologies Java (Quarkus [8], Spring, GraalVM) et des systèmes d'exploitation à 64 bits sur le forum.
J'ai également rejoint l'équipe Pi4J. Pi4J est un cadre et une bibliothèque permettant de combiner des applications Java avec toute la puissance des GPIO du Raspberry Pi. Ce projet a été lancé par Robert Savage qui cherchait des membres pour l'équipe afin de le porter à une nouvelle génération qui supporte entièrement Java 11+ et le Raspberry Pi 4 avec des modules Java et une architecture facilement extensible. Je suis enthousiaste à propos de la deuxième version de ce cadre, que nous espérons publier bientôt.

Abate : *Revenons au livre, Getting Started with Java on the Raspberry Pi. Pourquoi l'as-tu écrit ?*
Delporte : Pour le projet de la cabine de batterie, j'ai dû découvrir comment utiliser Java sur le Pi, comment installer la bonne version de JavaFX, comment contrôler le GPIO et un Arduino, etc. C'est alors que j'ai écrit mon premier article [9] publié dans MagPi (juillet 2019 [10]).
Elektor m'a demandé si cela pouvait faire l'objet d'un livre. Comme je ne trouvais pas de livre récent sur ce sujet et que Java a connu de grands changements ces dernières années, cette question m'a interpellé et dès le lendemain, j'ai commencé à écrire. Cela m'a pris plus de six mois et

beaucoup de soirées et de nuits, mais je me suis bien amusé en écrivant et en expérimentant. Pourvu que ce soit aussi amusant de lire le livre et d'essayer mes projets !

Abate : *As-tu des conseils à donner aux ingénieurs ou aux mordus qui envisagent d'utiliser Java pour leurs projets sur Raspberry Pi ?*
Delporte : Essayez ! Java est toujours l'un des meilleurs langages de programmation au monde. Que vous soyez un développeur Java expérimenté ou que vous partiez de zéro, il y a beaucoup à apprendre et à expérimenter lorsque vous combinez Java avec un Raspberry Pi et des composants électroniques.
Les exemples de mon livre utilisent des composants très bon marché. Ils ne sont pas difficiles à trouver. Tous les exemples du livre peuvent être utilisés pour donner vie aux projets de vos propres rêves. Le circuit de commande conçu pour la cabine de batterie de mon fils est une combinaison de plusieurs de ces exemples (**fig. 4**).

Abate : *Quels ont été les retours d'information jusqu'à présent ?*
Delporte : Python est évidemment le premier langage sur le Raspberry Pi (mais oui, c'est de là que vient le *Pi*) au point que certains le considèrent comme le seul bon choix, mais j'ai reçu beaucoup de commentaires positifs et nombre de questions sur ce sujet. J'ai même eu la chance d'écrire un billet pour l'*Oracle Java Magazine* [12], très lu et beaucoup partagé ! Il existe un intérêt évident pour ce sujet, et la future nouvelle génération de Pi4J facilitera encore la création d'applications puissantes.

Abate : *Y a-t-il un langage de programmation que tu ne connais pas, mais que tu comptes apprendre ? Y a-t-il un matériel que tu envisages d'essayer ?*
Delporte : Java n'est pas seulement un langage de programmation, mais aussi une machine virtuelle qui exécute le code Java. Sur cette même VM, vous pouvez également exécuter Scala, Kotlin et bien d'autres langages. Il y a donc encore beaucoup de choses à explorer dans ce monde. Pour le projet Pi4J, je veux étendre le site web du code d'exemple et de la documentation, je devrai donc mettre en place plusieurs petits exemples de matériel et apprendre beaucoup de nouvelles choses moi-même.

Succès de la programmation

Abate : *Concluons sur ton plus grand succès en matière de conception ou de programmation. Y a-t-il un projet spécifique (logiciel ou matériel) qui se démarque ? Qu'y avait-il de difficile dans ce projet ? Qu'as-tu appris ?*
Delporte : Dans mon travail à *Televic*, nous utilisons une combinaison de Java et de programmation intégrée pour connecter plusieurs serveurs et sources de données afin d'afficher en temps réel les informations pour les passagers sur les écrans de trains entiers. Un défi technique ! Parcourir un train en marche avec 100 écrans qui affichent en temps réel les trains en partance dans la prochaine gare avec les retards et les correspondances, c'est le pied ! Le flux nécessaire pour acheminer toutes ces données par des connexions sans fil peu fiables (les signaux GSM ne sont pas vraiment conçus pour les trains à grande vitesse) est un véritable chef-d'œuvre dont je suis très fier et que nous avons pu réaliser avec une petite équipe. Et je suis tout aussi impressionné par les enfants du CoderDojo qui ont réussi à réaliser leur premier jeu *Flappy Bird* en Scratch ou à faire clignoter une LED avec Arduino !

200503-03

PRODUITS

› **Livre :**
Getting Started with Java on the Raspberry Pi par **Frank Delporte**
www.elektor.fr/19292



LIENS

[1] **Livre** : Getting Started with Java on the Raspberry Pi : www.elektor.fr/getting-started-with-java-on-the-raspberry-pi
[2] **WebTechie** : blog de l'auteur: http://webtechie.be/
[3] **"A Software Developer's Space for DIY Projects and Writing"** : www.elektormagazine.com/news/electronics-workspace-software-developers-space
[4] **CoderDojo** : http://coderdojo.com/
[5] **"Pong on a Raspberry Pi"** : http://webtechie.be/post/2017-12-20-pong-on-a-raspberry-pi/
[6] **"Getting Started with FXGL Game Development"** : http://webtechie.be/post/2020-05-07-getting-started-with-fxgl/
[7] **"Drumbooth Controller with Raspberry Pi and JavaFX"** : http://webtechie.be/post/2020-03-30-drumbooth-controller-with-java-javafx-raspberrypi-arduino/
[8] **Quarkus** : http://webtechie.be/post/2020-07-28-spring-versus-quarkus-rest-h2-db-on-raspberry-pi/
[9] **WebTechie Articles** : http://webtechie.be/articles/
[11] **MagPi (en français !)** : www.magpi.fr/
[12] **"Getting Started with JavaFX on Raspberry Pi"** : http://blogs.oracle.com/javamagazine/getting-started-with-javafx-on-raspberry-pi



analyse de données et intelligence artificielle en Python

Interpréter les données réelles avec NumPy, pandas et le scikit-learn

Angelo Cardellicchio (Italie)

L'analyse et l'interprétation des données provenant de notre environnement est un sujet d'intérêt croissant. Ces données font désormais partie intégrante de notre vie quotidienne, qu'il s'agisse des données climatiques ou des données acquises au cours de processus de fabrication intelligents. La quantité de données nous permet, en théorie, de caractériser n'importe quel phénomène. Cependant il faut pour cela de nombreuses compétences, théoriques et pratiques. Nous examinons ici certaines de ces techniques et utilisons Python pour l'analyse de données du monde réel.

Des termes tels que *big data* (= *données massives)* et *intelligence artificielle* appartiennent au langage quotidien. Cela est principalement dû à deux facteurs. Le premier est la diffusion croissante et omniprésente des systèmes d'acquisition de données qui a permis la création de *référentiels de connaissances* pratiquement illimités. Le second est la croissance continue des capacités de calcul, grâce à l'utilisation généralisée des GPGPU (unités de traitement graphique polyvalentes) [1], qui a permis de relever des défis de calcul considérée autrefois comme impossible.

Commençons par la description d'un scénario d'application qui nous accompagnera au long de cet article. Imaginons qu'il faille surveiller une chaîne de production (peu importe le produit). Nous pouvons acquérir des données provenant d'un large éventail de sources. Par exemple, nous pouvons placer des capteurs sur toute la chaîne de production, ou utiliser des *informations contextuelles* qui indiquent l'âge et le type de chaque machine. Cet ensemble de données, ou jeu de *données*, peut être utilisé à différentes fins. Cela peut être la maintenance prédictive, pour évaluer et prévoir l'apparition de situations anormales, planifier les commandes de pièces de rechange ou entreprendre des réparations avant que les pannes ne se produisent, ce qui permet de réaliser des économies et d'accroître la productivité. En outre, la connaissance de l'historique des données nous permet de corréler les données mesurées par chaque capteur, pour mettre en évidence les éventuelles relations de cause à effet. Par exemple, si une augmentation soudaine de température et d'humidité dans la pièce a été suivie d'une diminution du nombre de pièces fabriquées, une modification pourrait consister à maintenir des conditions climatiques constantes à l'aide d'une climatisation.

La mise en place d'un tel système n'est certainement pas à la portée de tous. Cependant, elle est simplifiée par les outils mis à disposition par la communauté *open source*. Il suffit d'avoir un PC (ou, à défaut, un Raspberry Pi si la quantité de données à traiter n'est pas énorme), de connaitre Python (que vous pouvez approfondir en suivant un tutoriel comme celui-ci [2]) et, bien sûr les «outils» que je vous propose de découvrir ensemble !

Outils

Il faut savoir créer des programmes en Python. Pour cela, nous devrons installer l'interpréteur que vous trouverez sur le site officiel de Python [3]. À partir d'ici, nous supposerons que Python est installé et ajouté aux variables d'environnement de votre système.

L'environnement virtuel

Une fois l'installation de Python terminée, mettons en place un environnement virtuel, qui est en quelque sorte un «conteneur» séparé du reste de notre système et dans lequel sont installées les bibliothèques utilisées. La raison de l'utilisation d'un environnement virtuel pour l'installation globale des bibliothèques est liée à l'évolution rapide du monde Python. Très souvent, des différences substantielles surviennent même entre versions mineures de l'interpréteur, ce qui entraîne l'incompatibilité de bibliothèques



et, par conséquent, des programmes, car elles ont été écrites pour différentes versions de Python. En aménageant un environnement déterministe où la version de chaque bibliothèque installée est connue, nous aurons une sorte de garantie que nos programmes fonctionneront. En fait, il suffira de reproduire précisément la configuration de l'environnement virtuel et nous pouvons être sûrs que tout fonctionnera.
Pour gérer nos environnements virtuels, nous utilisons un logiciel appelé `virtualenvwrapper`. Celui-ci peut être installé à partir du shell en utilisant `pip` :

```
$ pip install virtualenvwrapper
```

Une fois l'installation terminée, nous créons un nouvel environnement virtuel comme suit :

```
$ mkvirtualenv ml-python
```

Notez que `ml-python` est le nom de l'environnement virtuel choisi pour notre exemple de scénario. Évidemment, ces noms peuvent varier et n'importe quel nom approprié pourra être choisi par le concepteur. Nous procédons ensuite à l'activation de l'environnement virtuel :

```
$ workon ml-python
```

Nous sommes prêts maintenant à installer les éléments nécessaires pour la suite de cet article.

Bibliothèques

Les bibliothèques présentées et utilisées ici sont les cinq plus utilisées pour l'analyse des données en Python.
La première, et peut-être la plus célèbre, est *NumPy*, qui est une sorte de port de MATLAB pour Python. NumPy est une bibliothèque pour calculs algébriques et matriciels. Par conséquent, les utilisateurs familiers de MATLAB trouveront de nombreuses similitudes, en termes de syntaxe et d'optimisation. L'utilisation du calcul algébrique dans NumPy est, en fait, plus efficace que les cycles imbriqués dans MATLAB (pour en savoir plus [4]). Comme on pouvait s'y attendre, le type de données au cœur de la fonction de NumPy est le *tableau* ou *array* en anglais. Il ne doit pas être confondu avec le *vecteur* informatique correspondant, mais doit plutôt être compris au sens algébrique et géométrique du terme comme une *matrice*. Comme l'analyse des données est basée sur des opérations algébriques et matricielles, NumPy est également à la base de deux des cadres les plus utilisés : scikit-learn (dont il sera question plus loin) et TensorFlow.
Un complément naturel à NumPy est la bibliothèque *pandas*, qui gère et lit des données provenant de sources hétérogènes, y compris des tableaux Excel, des fichiers CSV, ou même des bases de données JSON et SQL. *pandas* est extrêmement flexible et puissant, vous permettant d'organiser les données en structures appelées *dataframes* manipulables selon les besoins et exportables facilement directement dans des tableaux NumPy.
La troisième bibliothèque que nous utiliserons est *scikit-learn*. Née d'un projet universitaire, scikit-learn est un cadre qui met en œuvre la plupart des algorithmes d'apprentissage automatique utilisés de nos jours en fournissant une interface commune. Ce dernier concept est précisément celui de la programmation orientée objet : il est en effet possible d'utiliser pratiquement tous les algorithmes proposés par scikit-learn grâce à la méthode `fit_transform` en passant au moins deux paramètres, tels que les données analysées et les étiquettes associées.
Les deux dernières bibliothèques que nous utiliserons sont *Matplotlib* et *Jupyter*. La première, avec son complément *Seaborn*, est nécessaire pour visualiser les résultats de nos expériences sous forme de graphiques. La seconde nous offre l'utilisation de *notebooks* ou *carnets*, environnements interactifs d'utilisation simple et immédiate qui permettent à l'analyste de données d'écrire et d'exécuter des parties de code indépendamment des autres.
Avant d'aller plus loin, cependant, voici quelques concepts théoriques nécessaires à la construction d'une «base commune» pour le discours.

Les concepts

Le premier concept requis est celui des *ensembles de données* (ou *datasets*), ce qui est souvent considéré comme allant de soi. Il s'agit d'ensembles d'échantillons, chacun d'entre eux étant caractérisé par



Figure 1. L'écran d'accueil pour la gestion des carnets (notebook) dans Jupyter.

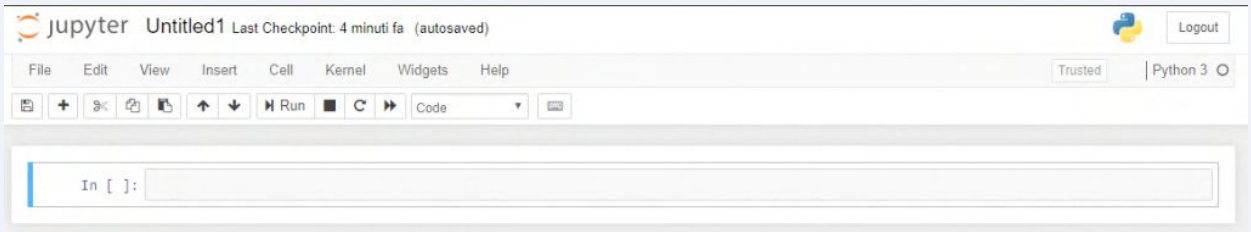


Figure 2. Un carnet (notebook) vide.

un certain nombre de variables ou de caractéristiques, qui décrivent le phénomène observé. Par souci de simplicité, nous pouvons considérer un ensemble de données comme un tableau Excel. Les lignes fournissent les *échantillons*, c'est-à-dire les *observations individuelles du* phénomène, tandis que les colonnes fournissent les *caractéristiques, c'est-à-dire les* valeurs qui caractérisent chacun des aspects du processus. Pour revenir à l'exemple de la fabrication intelligente, chaque ligne représentera les conditions de la chaîne de production à un moment donné tandis que chaque colonne comportera la valeur lue sur un capteur donné.

À propos de scikit-learn, il y a lieu de mentionner le concept de *label* ou de *classe*. La présence ou l'absence d'étiquettes permet de distinguer les algorithmes supervisés des algorithmes non supervisés. La différence est, du moins en principe, assez simple : les *algorithmes supervisés* requièrent une connaissance a priori de la classe de chaque échantillon de l'ensemble de données de l'exemple, pas les *algorithmes non supervisés*. En pratique, pour utiliser un algorithme supervisé, il est nécessaire qu'un expert du domaine établisse la *classe d'appartenance de* chaque échantillon. Dans le cas d'un processus de fabrication intelligent, un «expert» pourrait déterminer si un ensemble de lectures, à un moment précis, représente une situation anormale ou non. Ainsi, l'échantillon unique peut être associé à l'une de ces deux classes possibles (anormal/normal). Cela n'est pas nécessaire pour les algorithmes non supervisés.

Il faut aussi distinguer les processus avec des données *indépendantes et identiquement distribuées* (IID) des données dans un *ordre chronologique*. La différence est liée à la nature du phénomène observé. Les échantillons d'un processus IID sont indépendants les uns des autres, alors que dans une série temporelle, chaque échantillon dépend d'une combinaison linéaire ou non linéaire des valeurs que le processus a produites à des moments précédents.

Commençons !

Maintenant que les termes théoriques et pratiques requis sont connus, nous passons à l'utilisation d'un ensemble de données approprié pour notre exemple. L'ensemble de données utilisé est *SECOM*, un acronyme qui signifie *SEmiCOnductor Manufacturing*, qui contient les valeurs lues par un ensemble de capteurs lors de la surveillance d'un processus de fabrication de semi-conducteurs. Dans l'ensemble de données, téléchargeable à partir de différentes sources (comme Kaggle [5]), il y a 590 variables, représentatives chacune de la lecture d'un seul capteur à un moment donné. L'ensemble de données contient également des étiquettes qui permettent de distinguer les défaillances et les anomalies du bon fonctionnement du système.

Une fois l'ensemble de données téléchargé, nous installons les bibliothèques mentionnées ci-dessus. Depuis la ligne de commande, saisissez :

```
$ pip install numpy pandas scikit-learn matplotlib
   seaborn jupyter numpy pandas install
```

Une fois les bibliothèques installées, nous pouvons mettre en place un simple pipeline pour l'analyse des données.

Le premier carnet

La première étape consiste à créer un nouveau *carnet*. À partir de la ligne de commande, nous lançons Jupyter par les instructions suivantes :

```
$ jupyter-notebook
```

Un écran similaire à celui de la **figure 1** s'ouvre. Nous créons un *carnet* en sélectionnant *Nouveau* > *Python 3*. Un nouvel onglet s'ouvrira dans notre navigateur avec le carnet créé. Prenons le temps de nous familiariser avec l'interface (**fig. 2**) qui ressemble (très vaguement) à une ligne de commande interactive, un menu et plusieurs options.

Ce qui saute aux yeux est la *cellule*. L'exécution de cellules individuelles est lancée par le bouton *Run*. Elle est indépendante de celle des autres cellules (gardons à l'esprit la validité du concept de *portée des variables*).

Les trois boutons immédiatement à droite du bouton *Run* permettent d'arrêter, de redémarrer et d'initialiser le *noyau*, c'est-à-dire l'instance associée à notre carnet par Jupyter. Le redémarrage de l'instance peut être nécessaire pour réinitialiser les variables locales et globales associées au script, ce qui est particulièrement utile lorsque vous expérimentez avec de nouvelles méthodes et bibliothèques.

	a21	a87	a88	a89	a114	a115	a116	a117	a118	a120	...	a528
count	1567.000000	1567.000000	1567.000000	1567.000000	1567.000000	1567.000000	1567.000000	1567.000000	1567.000000	1567.000000	...	1567.000000
mean	1.405054	2.401872	0.982420	1807.815021	0.945424	0.000123	747.383792	0.987130	58.625908	0.970777	...	6.395717
std	0.016737	0.037332	0.012848	53.537262	0.012133	0.001668	48.949250	0.009497	6.485174	0.008949	...	1.888698
min	1.179700	2.242500	0.774900	1627.471400	0.853400	0.000000	544.025400	0.890000	52.806800	0.841100	...	2.170000
25%	1.396500	2.376850	0.975800	1777.470300	0.938600	0.000000	721.023000	0.989500	57.978300	0.964800	...	4.895450
50%	1.406000	2.403900	0.987400	1809.249200	0.946400	0.000000	750.861400	0.990500	58.549100	0.969400	...	6.410800
75%	1.415000	2.428600	0.989700	1841.873000	0.952300	0.000000	776.781850	0.990900	59.133900	0.978300	...	7.594250
max	1.453400	2.555500	0.993500	2105.182300	0.976300	0.041400	924.531800	0.992400	311.734400	0.982700	...	14.447900

Figure 3. Les cinq premières lignes de l'ensemble de données SECOM.

	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	...	a582	a583	a584	a585	a586	a587	a588	a58
0	3030.93	2564	2187.7333	1411.1265	1.3602	100	97.6133	0.1242	1.5005	0.0162	...	?	0.5005	0.0118	0.0035	2.363	?	?	
1	3095.78	2465.14	2230.4222	1463.6606	0.8294	100	102.3433	0.1247	1.4966	-0.0005	...	208.2045	0.5019	0.0223	0.0055	4.4447	0.0096	0.0201	0.00
2	2932.61	2559.94	2186.4111	1698.0172	1.5102	100	95.4878	0.1241	1.4436	0.0041	...	82.8602	0.4958	0.0157	0.0039	3.1745	0.0584	0.0484	0.014
3	2988.72	2479.9	2199.0333	909.7926	1.3204	100	104.2367	0.1217	1.4882	-0.0124	...	73.8432	0.499	0.0103	0.0025	2.0544	0.0202	0.0149	0.004
4	3032.24	2502.87	2233.3667	1326.52	1.5334	100	100.3967	0.1235	1.5031	-0.0031	...	?	0.48	0.4766	0.1045	99.3032	0.0202	0.0149	0.004

5 rows × 591 columns

Figure 4. Brève description statistique de l'ensemble de données SECOM.

Une autre option utile est le choix possible du type de cellule, entre *Code* (c'est-à-dire code Python), *Markdown* (utile pour insérer des commentaires et des descriptions dans le format utilisé, par exemple, par GitHub READMEs), *Raw NBContent* (texte *brut)* et *Heading* (offrant un raccourci pour insérer des titres).

Importation et affichage des données

Une fois familiarisés avec l'interface, nous pouvons passer à la mise en œuvre de notre script. Ici, nous importons les bibliothèques et les modules que nous utiliserons :

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from ipywidgets import interact
from sklearn.ensemble import RandomForestClassifier
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix,
   accuracy_score
from sklearn.utils import resample
```

Signalons l'instruction `%matplotlib inline` qui nous permet d'afficher correctement les graphiques produits par Matplotlib. Ensuite, les données du fichier contenant l'ensemble de données SECOM sont importées à l'aide de la fonction `read_csv` de *pandas*. Notez que, dans cet exemple, pour simplifier, le chemin relatif du fichier est codé en dur. Cependant, il est conseillé d'utiliser le paquet *os* Python pour permettre à notre programme de déterminer lui-même ce chemin lorsque c'est nécessaire.

```
data = pd.read_csv('data/secom.csv')
```

L'instruction précédente lit les données contenues dans le fichier *secom.csv, en les* organisant dans un cadre de données nommé *data*. Nous pouvons afficher les cinq premières lignes de la trame de données grâce à l'instruction `head()` (**fig. 3**).

```
data.head()
```

La visualisation des premières lignes du cadre de données peut être utile pour obtenir un premier aperçu des données à analyser. Dans ce cas, on remarque immédiatement la présence de certaines valeurs égales à « ?» qui représentent probablement des valeurs nulles. En outre, il est évident que la plage des valeurs varie considérablement, un facteur que nous devrons garder à l'esprit plus tard. Nous pouvons également utiliser la fonction `describe()` pour obtenir un aperçu rapide des caractéristiques statistiques de chaque variable (**fig. 4)**.

```
data.describe()
```

L'analyse statistique peut, en général, mettre en évidence des conditions présentant un manque de normalité (c'est-à-dire des données dont la distribution n'est pas paramétrique), ou la présence d'anomalies. À titre d'exemple, nous constatons que l'écart-type (std) associé aux variables *a116* et *a118* est assez élevé proportionnellement, de sorte que nous nous attendons à une forte signification de ces variables lors de leur analyse. D'autre part, des variables telles que *a114* ont un faible std, on s'attend donc à ce qu'elles soient écartées faute de pertinence dans le processus analysé.
Une fois le chargement et l'affichage de la trame de données terminés, nous pouvons passer à une partie fondamentale du pipeline : le *prétraitement*.

Prétraitement des données

Dans un premier temps, nous affichons le nombre d'échantillons associés à chaque classe. Pour ce faire, nous utiliserons la fonction `value_counts()` sur la colonne `classvalue` car elle contient les étiquettes associées à chaque échantillon.

```
data[«classvalue»]. value_counts()
```

Nous voyons qu'il y a 1463 échantillons prélevés en situation normale de fonctionnement (*classe -1*) et 104 en situation de défaillance (*classe 1*). L'ensemble de données est donc fortement déséquilibré et il serait approprié de prendre des mesures pour uniformiser la répartition des échantillons entre différentes classes. Cela renvoie à la fonction intrinsèque des algorithmes d'apprentissage machine qui apprennent sur la base des données dont ils



Figure 5. Nombre d'échantillons par classe dans l'ensemble de données SECOM.

disposent. Dans ce cas précis, l'algorithme apprendra à caractériser avec succès une situation de comportement standard, mais sa caractérisation des situations anormales sera affectée par de «incertitudes». Le déséquilibre est encore plus évident lorsque l'on observe le nuage de points (**fig. 5**) :

```
sns.scatterplot(data.index, data['classvalue'],
   alpha=0.2)
plt.show()
```

En gardant à l'esprit ce déséquilibre (auquel nous reviendrons), nous séparons les étiquettes des données :

```
labels = data['classvalue']
data.drop('classvalue', axis=1, inplace=True)
```

Notez l'utilisation du paramètre `axis` dans la fonction `drop` qui nous permet de spécifier que la fonction doit opérer sur les colonnes du cadre de donnés (*dataframe*). Par défaut, les fonctions *pandas* opèrent sur les lignes.

Un autre aspect qui peut être extrapolé de l'analyse des ensembles de données est la présence, dans cette version spécifique des données SECOM, de nombreuses colonnes contenant des données de différents types, c'est-à-dire à la fois des chaînes de caractères et des chiffres. Par conséquent, les *pandas,* incapables de déterminer avec certitude le type de données avec lequel chaque caractéristique est représentée, en reportent la définition sur l'utilisateur. Pour mettre toutes les données sous forme numérique, il faut donc utiliser trois fonctions offertes par les *pandas*.
La première fonction que nous utiliserons est `replace()`, avec laquelle nous pouvons remplacer tous les points d'interrogation par la valeur constante *numpy.nan,* le caractère de remplacement utilisé pour traiter les valeurs nulles dans les *tableaux* NumPy.

```
data = data.replace('?', np.nan, regex=False)
```

Le premier paramètre de la fonction est la valeur à remplacer, le deuxième est la valeur à utiliser pour le remplacement, et le troisième est un drapeau indiquant si le premier paramètre représente ou non une expression régulière. Nous pourrions également utiliser une syntaxe alternative en utilisant le paramètre `inplace` réglé sur `True`, comme suit :

```
data.replace('?', np.nan, regex=False, inplace=True)
```

Les deuxième et troisième fonctions que nous pouvons utiliser pour résoudre les problèmes signalés ci-dessus sont les fonctions `apply()` et `to_numeric()` respectivement. La première permet d'appliquer une certaine fonction à toutes les colonnes (ou lignes) d'une trame de données, tandis que la deuxième convertit une seule colonne en valeurs numériques. En les combinant, nous produisons des données uniques et nous supprimons également les valeurs que NumPy et scikit-learn ne peuvent traiter :

```
data.apply(pd.to_numeric)
```

Nous devons maintenant évaluer quelles caractéristiques contenues dans l'ensemble de données sont réellement utiles. Nous utilisons généralement des techniques (plus ou moins complexes) de *sélection des caractéristiques* pour réduire les redondances et la taille du problème à traiter, ce qui présente des avantages évidents en termes de temps de traitement et de performance de l'algorithme. Dans notre cas, nous nous appuyons sur une technique moins complexe qui implique l'élimination des caractéristiques de variance faible (et donc, comme mentionné, de faible importance). Nous créons donc un widget interactif qui nous permet de visualiser, sous la forme d'un histogramme, la distribution des données pour chaque caractéristique :

```
@interact(col=(0, len(df.columns) - 1)
def show_hist(col=1):
    data['a' + str(col)]. value_counts().
   hist(grid=False, figsize=(8, 6))
```

L'*interactivité* est assurée par le *décorateur* (= *decorator*) `@interact`, dont la valeur de référence (c'est-à-dire `col`) varie entre 0 et le nombre de caractéristiques présentes dans l'ensemble de données. En explorant les données affichées par le widget, nous déterminerons combien de caractéristiques prennent une valeur unique, ce qui signifie qu'elles peuvent être simplement ignorées dans l'analyse. Nous pouvons alors décider de les éliminer comme suit :

```
single_val_cols = data.columns [len(data)/data.
   nunique() < 2]
secom = data.drop(single_val_cols, axis=1)
```

Bien sûr, il existe des techniques de sélection de caractéristiques plus pertinentes et plus raffinées qui utilisent par exemple des paramètres statistiques. La documentation scikit-learn [6] en donne un aperçu complet.

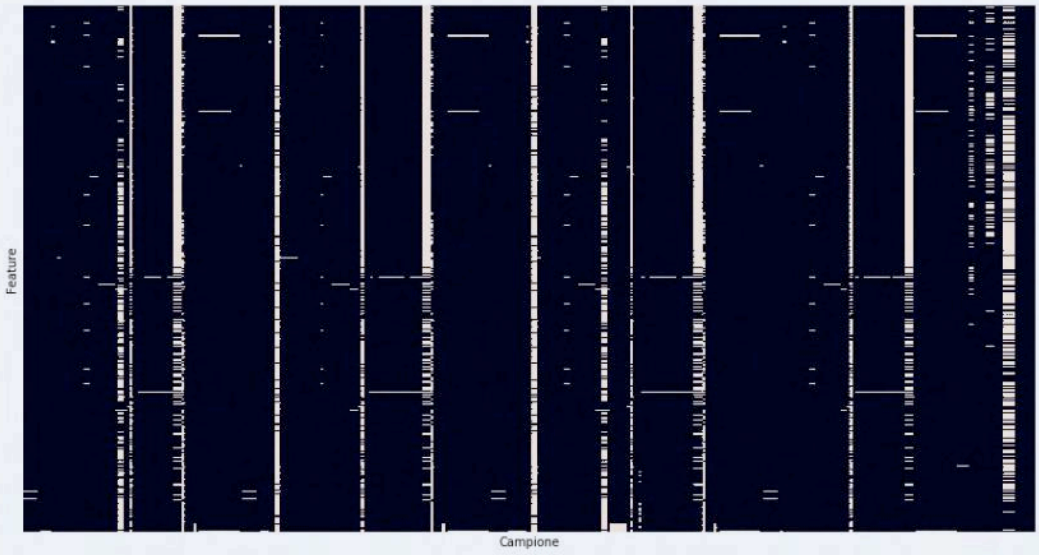


Figure 6. Carte des valeurs nulles dans l'ensemble de données SECOM.

La dernière étape consiste à traiter les valeurs nulles (que nous avons remplacées précédemment par *np.nan*). Nous inspectons l'ensemble de données pour voir combien il y en a ; pour ce faire, nous utilisons une carte thermique (**fig. 6**) où les points blancs représentent les valeurs nulles.

```
sns.heatmap(secom.isnull(), cbar=False)
```

Il est évident que de nombreux échantillons présentent un pourcentage élevé de valeurs nulles qui ne doivent pas être prises en compte afin d'éliminer l'effet de *biais* sur les données.

```
na_cols = [col for col in secom.columns if
   secom[col]. isnull().sum() / len(secom) > 0.4]
secom_clean = secom.drop(na_cols, axis=1)
secom_clean.head()
```

Grâce aux commandes précédentes, une *liste de compréhension (= comprehension list)* a maintenant isolé toutes les caractéristiques ayant plus de 40 % de valeurs nulles, ce qui permet de les supprimer de l'ensemble de données.
Les caractéristiques ayant moins de 40 % de valeurs nulles doivent encore être traitées. Nous pouvons utiliser ici notre premier objet scikit-learn, le `SimpleImputer`, qui attribue des valeurs à tous les NaNs en fonction d'une stratégie définie par l'utilisateur. Dans ce cas, nous utiliserons une stratégie `moyenne`, associant la valeur moyenne supposée par l'objet à chaque NaN.

```
imputer = SimpleImputer(strategy='mean')
secom_imputed = pd.DataFrame(imputer.
   fit_transform(secom_clean))
secom_imputed.columns = secom_clean.columns
```

À titre d'exercice, nous vérifions que nous n'avons aucun zéro dans l'ensemble de données avec une autre carte thermique qui, comme on peut s'y attendre, sera uniformément foncée. Ensuite, nous pouvons passer au traitement proprement dit.

Traitement des données

Divisons notre ensemble de données en deux sous-ensembles : un ensemble de *formation* et un ensemble de *test*. Cette subdivision est nécessaire pour atténuer le phénomène de surapprentissage (= *overequipment*), par lequel l'algorithme adhère trop aux données [7], et garantit l'applicabilité du modèle à des cas différents de ceux pour lesquels il a été formé. Pour ce faire, nous utilisons la fonction `train_test_split` :

```
X_train, X_test, y_train, y_test = train_test_
   split(secom_imputed, labels, test_size=0.3)
```

Le paramètre `test_size` permet de spécifier le pourcentage de données réservées au test ; les valeurs standard de ce paramètre se situent généralement entre 0,2 et 0,3.
Il est également important de *normaliser* les données. Nous avions remarqué que les valeurs de certaines caractéristiques présentent des variations beaucoup plus fortes que d'autres, ce qui leur donne plus de poids. Le fait de les normaliser les ramène dans une seule plage de valeurs afin d'éviter les déséquilibres dus aux décalages initiaux. Pour ce faire, nous utilisons `StandardScaler` :

```
scaler = StandardScaler()
X_train = pd.DataFrame(scaler.fit_transform(X_train),
   index=X_train.index, columns=X_train.columns)
X_test = pd.DataFrame(scaler.fit_transform(X_test),
   index=X_test.index, columns=X_test.columns)
```

Il est intéressant de noter l'utilité de l'interface commune offerte par scikit-learn. Tant le `scaler` que l'`imputer` utilisent la méthode `fit_transform` pour traiter les données qui, dans des pipelines complexes, simplifient grandement l'écriture du code et la compréhension de la bibliothèque.

Nous voici enfin prêts à classer les données. En particulier, nous utiliserons une *forêt aléatoire (random forest)* [8], en obtenant, après formation, un modèle capable de distinguer les situations normales et anormales. Nous allons vérifier les performances du modèle identifié de deux manières. La première est le *score de précision (accuracy score)*. Il s'agit du pourcentage d'échantillons appartenant à la série de tests correctement classés par l'algorithme. La seconde est la *matrice de confusion* [9] qui met en évidence le nombre de faux positifs et de faux négatifs.
Créons d'abord le classificateur :

```
clf = RandomForestClassifier(n_estimators=500,
   max_depth=4)
```

Cela crée une forêt aléatoire avec 500 estimateurs dont la profondeur maximale est de 4 niveaux. Nous pouvons maintenant entraîner notre modèle sur des données d'entraînement :

```
clf.fit(X_train, y_train)
```

Une fois la formation terminée, le modèle formé est utilisé pour classer les échantillons d'essai :

```
y_pred = clf.predict(X_test)
```

Il en résulte deux étiquettes qui se rapportent à chacun des deux tests. Le premier, qui appartient à `y_test`, représente la «vérité», tandis que le second, qui appartient à `y_pred`, est la valeur prédite par l'algorithme. En les comparant, nous déterminons à la fois `accuracy` et `confusion_matrix`.

```
accuracy = accuracy_score(y_test, y_pred))
 cf = confusion_matrix(y_test, y_pred))
```

L'exemple donne les résultats suivants :

```
Model accuracy on test set is: 0.9341825902335457
The confusion matrix of the model is:
[[440    0]
 [ 31    0]]
```

La précision, d'environ 93 %, est excellente, donc le modèle semble bon. Cependant, nous constatons que le modèle est très précis pour la classification des échantillons de la classe prédominante, mais très imprécis pour la classification des échantillons de la classe minoritaire. Donc il y a manifestement un biais. Il nous faut une stratégie pour améliorer cette situation. C'est possible en suréchantillonnant les données appartenant à la classe minoritaire de manière à équilibrer, au moins partiellement, l'ensemble des données. Pour ce faire, nous utiliserons la fonction `resample` (`rééchantillonnage`) de *pandas*.

```
normals = data[data['classvalue'] == -1]
anomalies = data[data['classvalue'] == 1]
anomalies_upsampled = resample(anomalies,
   replace=True, n_samples=len(normals))
```

Cela permet d'augmenter la taille de l'ensemble de données afin de rapprocher le nombre d'échantillons normaux du nombre d'échantillons anormaux. En conséquence, nous devons redéfinir `X` et `Y` comme suit :

```
upsampled = pd.concat([normals, anomalies_upsampled])
X_upsampled = upsampled.drop('classvalue', axis=1)
y_upsampled = upsampled['classvalue']
```

En effectuant à nouveau la formation (y compris en répétant les procédures de fractionnement et de normalisation), nous obtenons les résultats suivants pour notre exemple :

```
Model accuracy on test set is: 0.8631921824104235
The confusion matrix of the model is:
[[276 41]
 [ 43 254]]
```

Visiblement la précision du modèle a diminué, probablement à cause de la plus grande hétérogénéité induite dans l'ensemble des données. Cependant, en regardant la matrice de confusion, nous remarquons immédiatement que le modèle a en réalité amélioré ses capacités de généralisation, réussissant également à classer correctement les échantillons appartenant à des situations anormales.

Conclusions et références

Dans cet article, nous avons présenté un pipeline pour l'analyse de données provenant de processus réels en Python. Manifestement chacun des sujets abordés est extrêmement vaste. Une expérience théorique et pratique est essentielle si vous voulez vous engager sérieusement dans l'analyse de données. Nous avons appris également qu'il ne faut pas s'arrêter au premier résultat obtenu, même dans des situations aussi complexes que celle dont il est question. Il convient d'interpréter de différents points de vue les résultats obtenus afin de distinguer un modèle opérationnel d'un modèle plus ou moins évidemment faussé.

Le message à retenir est donc le suivant : l'analyse des données ne peut être une discipline mécanique, elle exige au contraire une analyse critique, approfondie et variée du phénomène observé, guidée par des notions théoriques et des compétences pratiques. Je recommande vivement les références ci-dessous grâce auxquelles vous pourrez approfondir vos connaissances sur certains des aspects abordés dans l'article, ainsi que le lien vers le dépôt du GitLab où vous pourrez consulter le code écrit pour cet article.

200505-03

Article publié en italien par Elettronica Open Source (https://it.emcelettronica.com) et traduit par Elektor avec sa permission.

LIENS

[1] **Support informatique GPU MATLAB** : https://uk.mathworks.com/solutions/gpu-computing.html

[2] **Guides pour les débutants en programmation Python** : https://wiki.python.org/moin/BeginnersGuide/Programmers

[3] **Python** : www.python.org/

[4] **Bonnes pratiques d'optimisation dans MATLAB** : ttps://uk.mathworks.com/videos/best-practices-for-optimisation-in-matlab-96756.html

[5] **Ensemble de données UCI SECOM** : https://www.kaggle.com/paresh2047/uci-semcom/kernels

[6] **Guide de l'utilisateur Scikit-Learn** : https://scikit-learn.org/stable/user_guide.html

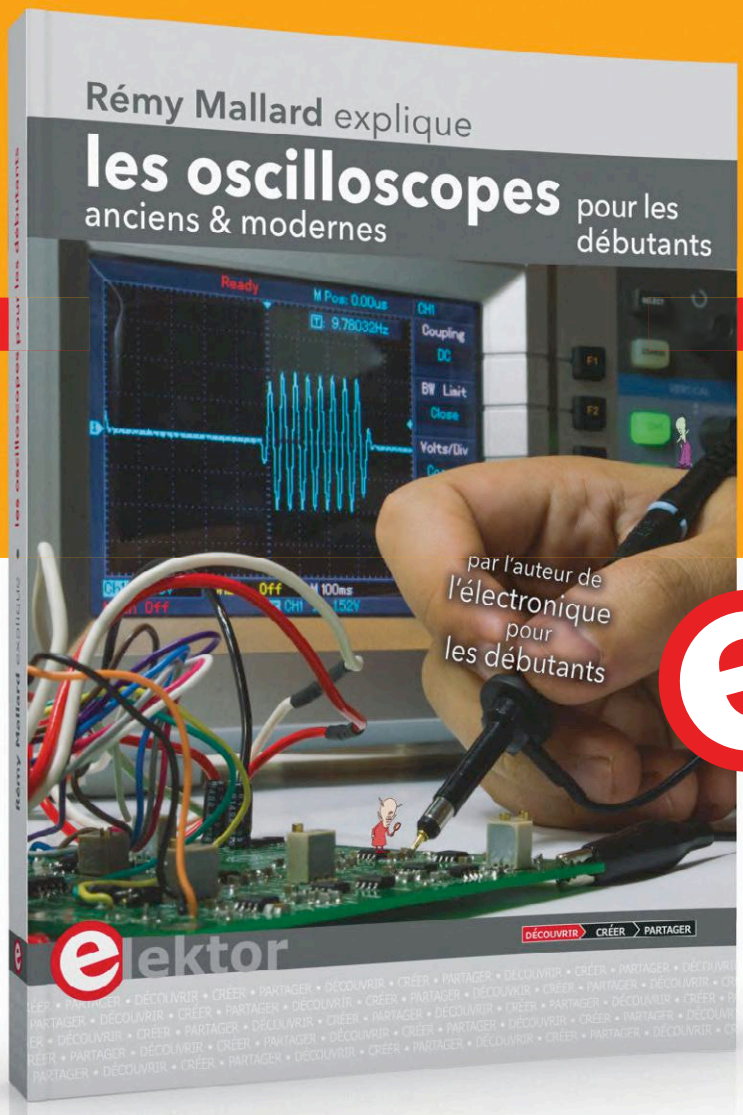
[7] **Sur- ou sous-apprentissage – un exemple complet** : https://towardsdatascience.com/overfitting-vs-underfitting-a-complete-example-d05dd7e19765

[8] **Comprendre la forêt aléatoire** : https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[9] **Comprendre la matrice de confusion** : https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

[10] **Dépôt GitLab pour cet article** : https://gitlab.com/eos-acard/machine-learning-in-python

Rémy Mallard explique
les oscilloscopes
anciens & modernes
pour les débutants



Points forts

- Livre orienté vers la pratique, avec « juste ce qu'il faut » de théorie
- Traite aussi le sujet des oscilloscopes numériques modernes
- Ouvrage utile avant l'achat pour guider le choix d'un oscilloscope
- Manuel d'utilisation à consulter régulièrement après l'achat
- Comme dans ses précédents livres, l'auteur partage sa bonne humeur communicative et sa grande expérience

elektor

En électronique, si l'on veut progresser dans le plaisir et dans la compréhension, il faut un oscillo. Comment le choisir ? À peine cette question-là aura-t-elle trouvé sa réponse, il en viendra d'autres qui se résument ainsi : comment se servir de l'oscilloscope de telle sorte que ce qu'il affiche corresponde à la réalité des signaux ?
Dans ce livre, Rémy Mallard, répond à ces questions.

Auteur : Rémy Mallard
Pages : 375 (en couleur)
Format : 17 x 23,5 cm (broché)
ISBN : 978-2-86661-208-5

Pour commander ce livre :
www.elektor.fr/19124

Propeller 2 de Parallax (1)

Une découverte

Mathias Claussen (Elektor)

Après avoir gravé dans le silicium sa nouvelle puce Propeller 2, le fabricant Parallax nous a soumis une carte d'évaluation pour que nous en palpions les jolis biceps: huit noyaux, un demi-méga de RAM et des entrés-sorties rapides, jusqu'à 300 MHz.

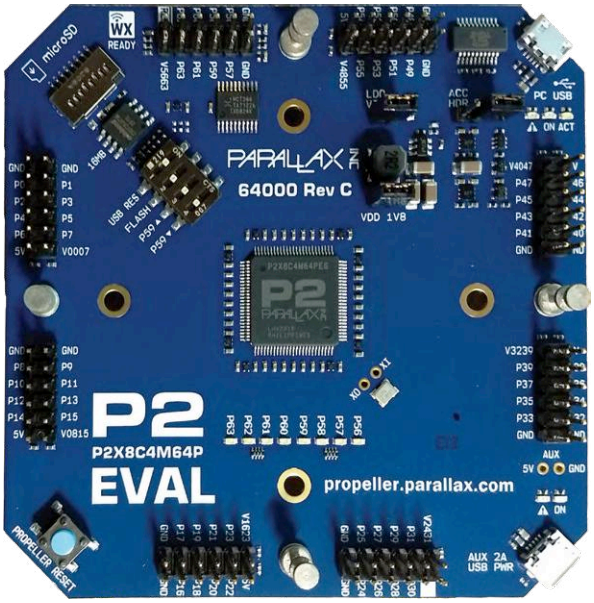


Figure 1. Évaluation de la carte Propeller 2 Rev.C de Parallax.

Lorsque Parallax a présenté sa première puce Propeller en 2006, ça ne ressemblait à rien de connu. Ces derniers temps, tandis que Parallax travaillait sur Propeller 2, ils ont demandé leur avis aux utilisateurs, non pas sur la puce sous forme de silicium qui n'existait pas encore, mais sous forme de flux binaire pour carte FPGA Altera DE10-Nano. Parallax a recueilli les commentaires avant de fondre Propeller 2 dans le silicium pour de bon. Le processus de fabrication de la puce a été décrit sur le forum Propeller pour que la communauté puisse le suivre. Au moment où j'écris ces lignes, le silicium en phase de révision C est approuvé pour la production. Jetons un coup d'œil à Propeller 2 et à ses caractéristiques.

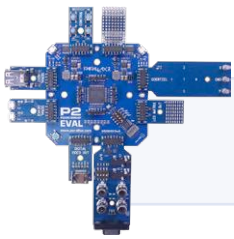
Propeller 2

La fréquence d'horloge officielle de la puce est de 180 MHz, ce qui donne 90 MIPS par noyau, et chaque instruction prend au moins deux cycles d'horloge. Le surcadencement de la Propeller 2 est possible, pour atteindre des fréquences supérieures à 300 MHz, soit 150 MIPS par noyau. Sur les microcontrôleurs, les noyaux d'unité centrale sont généralement célibataires, parfois en couple. Le Propeller 2 en a huit. Oh ! Ces coeurs indépendants, appelés *cogs* (= rouages ou engrenages), offrent une grande puissance de calcul.

Tous les rouages partagent 512 Ko de RAM sur ce modèle, avec en plus, par noyau, des registres (512 * 32 bits) et de la mémoire vive (512 * 32 bits). Outre les nombreux noyaux et la mémoire vive, la puce comprend une batterie de périphériques engageants :

- solveur CORDIC avec correction du facteur d'échelle
- sémaphore à 16 bits avec lecture-modification-écriture atomique
- compteur à 64 bits à fonctionnement libre
- interface hôte et esclave USB 2.0 FS
- broches d'E/S intelligentes (cf. encadré).

Parallax nous a confié un kit d'évaluation (**fig. 1**) avec un échantillon en version révisée C, ce qui nous permet de nous faire la main et de commencer à jouer avec. Je l'utiliserai pour cette série d'articles sur la puce, ses périphériques, son code, et bien davantage.
Au moment de boucler cet article (déc. 2020) la puce n'est pas encore disponible officiellement. Il est probable qu'en matière de logiciel nous tomberons sur du logiciel en chantier. Il y a de la matière pour expérimenter, notamment une pile de modules complémentaires à tester (**fig. 2**).



Propeller 2 possède 8 coeurs indépendants, appelés *cogs*, qui représentent une grosse puissance de calcul.

FONCTIONS DE BROCHES INTELLIGENTES

- CN/A de 8 bits, 120 Ω (3 ns) et 1 kΩ avec suréchantillonnage 16 bits, bruit, et modes numériques haut/bas
- CA/N delta-sigma à cinq gammes, deux sources et étalonnage VIO/GIO
- Plusieurs modes d'échantillonnage CA/NC : 2n SINC2 automatique, SINC2/SINC3 réglable, oscilloscope
- Modes d'entrée : logique, Schmitt, comparateur broche à broche et comparateur à 8 bits
- Filtrage d'entrée unanime sur 2/3/5/8 bits avec taux d'échantillonnage sélectionnable
- Incorporation des entrées des broches relatives, -3 à +3
- Réactions locales négatives ou positives, avec ou sans pointage
- Modes d'entraînement séparés pour la sortie haute et basse : logique/1,5 k/15 k/150 k/1 mA/100 µA/10 µA/flottant
- Sortie d'horloge à 32 bits programmable, sortie de transition, sortie NCO/duty
- Triangle/dents de scie/SMPS Sortie MLI, cadre de 16 bits avec prédiviseur à 16 bits
- Décodage en quadrature avec compteur à 32 bits, modes position et vitesse
- 16 mesures différentes de 32 bits impliquant un ou deux signaux
- USB à haut débit et à bas débit (via des paires de broches paires/impaires)
- Émission et réception série synchrones, 1 à 32 bits, clock/2 bauds
- Émission et réception série asynchrones, 1 à 32 bits, clock/3 bauds

Où sont passés les périphériques ?

Il ne vous aura pas échappé qu'il n'a pas encore été question de périphériques. Où sont donc les modules SPI, I2C et UART ? Eh bien, ils peuvent être formés au moyen des broches intelligentes et un peu de code dans les cogs. Ils ne sont pas mentionnés explicitement dans les caractéristiques, mais ils sont disponibles. Cette approche nous permettra de produire un signal HDMI directement sur un moniteur et d'afficher le contenu d'une puce flash. Ce sera pour la fin de cette série. Pour éviter une énumération barbante des périphériques et de leurs caractéristiques, nous les découvrirons *à la demande*, au fil de notre progression avec Propeller 2, Notre prochaine étape sera un aperçu de l'environnement de développement et notre émoustillante première touche avec une broche d'entrée/sortie pour piloter une LED.

200479-05

Qu'en pensez-vous ?
Vos questions et vos remarques sur cet article peuvent être adressées en anglais à mathias.claussen@elektor.com ou en français à redaction@elektor.fr

Ont contribué à cet article
Auteur : **Mathias Claußen**
Rédaction : **Jens Nickel & C. J. Abate**
Maquette : **Giel Dols**
Traduction : **Richard Kerr**

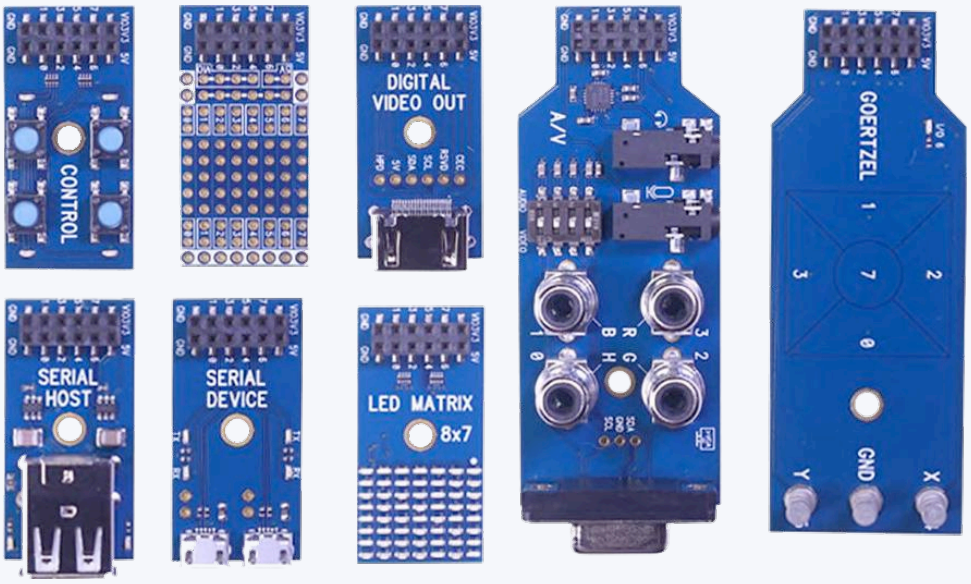


Figure 2. Cartes auxiliaires pour la carte d'évaluation de Propeller 2.

observatoire **du matériel libre**

Tessel Renzenbrink (Pays-Bas)

L'*Open Hardware Observatory* ou observatoire du matériel libre vise à rendre plus accessible le matériel à source ouverte. Il fournit un moteur de recherche pour vous aider à trouver sur la toile des projets de matériel libre. Il développe une plate-forme communautaire pour l'examen de la documentation sur les matériels libres basée sur la norme DIN SPEC 3105. L'observatoire est animé par un groupe de personnes d'*Open Source Ecology Germany e.V.* et TU Berlin. Lukas Schattenhofer travaille avec Consuelo Alberca Susano, Mehrdad Mansouri, et Nils Weiher sur la communauté autour de l'Observatoire. Schattenhofer explique son fonctionnement et invite les lecteurs d'Elektor à rejoindre la communauté.

La conception du matériel libre ou *open-source hardware* (OSH) est rendue publique afin que chacun puisse étudier, modifier, distribuer, fabriquer et vendre, soit cette conception elle-même soit le matériel basé sur cette conception [1]. Pour comprendre les avantages du matériel libre, il faut changer de point de vue sur la technologie, déclare Lukas Schattenhofer. « Pour la plupart des gens, la technologie est plus quelque chose que l'on consomme, et moins quelque chose que l'on construit et que l'on entretient. Quand un composant de leur téléphone tombe en panne, pas question de le réparer, ils achètent un nouveau téléphone. La plupart des technologies sont ***propriétaires***. Les informations requises pour des réparations ne sont pas accessibles. Si toute la documentation était disponible, vous n'auriez pas besoin d'acheter un nouveau téléphone tous les deux ans. Vous pourriez le réparer et même le construire vous-même. Cela changerait la façon dont les produits sont conçus. Ils deviendraient plus modulaires, de sorte que vous pourriez échanger des pièces et combiner les produits entre eux».

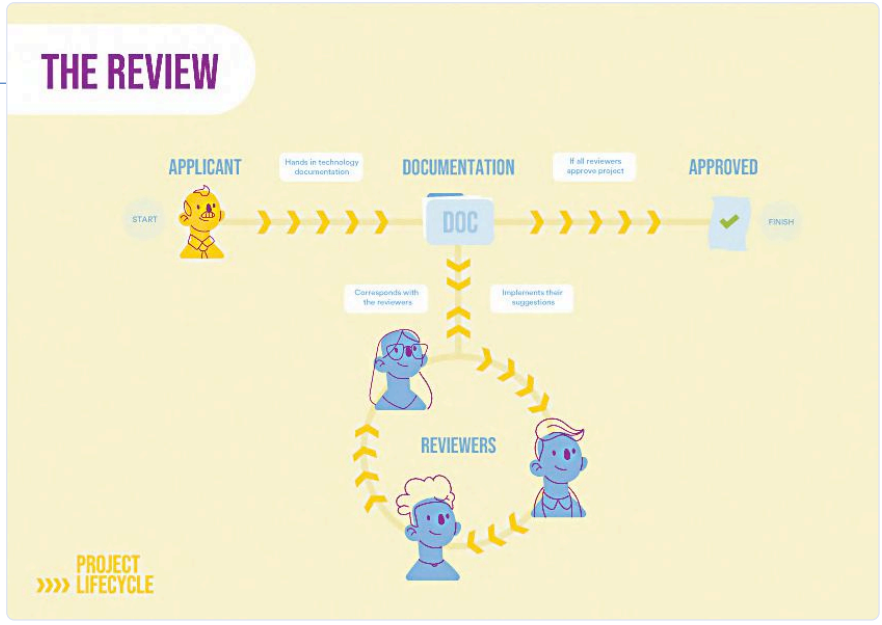
> « L'*Open Hardware Observatory* (OHO) promeut le matériel libre en fournissant un répertoire en ligne des projets ouverts. »

«La documentation est la source»

Même s'il existe une communauté florissante de matériel libre, l'OSH est encore beaucoup moins connu que les logiciels à code source ouvert. L'*Open Hardware Observatory* (OHO) vise à rendre l'OSH plus visible en fournissant un répertoire en ligne des projets ouverts [2]. Pour ce faire, l'OHO a créé un moteur de recherche qui explore la toile à la recherche de projets. Ceux-ci sont répertoriés avec des photos et une description brève. Tous les projets de cette base de données ne sont pas vraiment ouverts. C'est là qu'intervient la deuxième branche de l'OHO : la plate-forme d'**évaluation**, qui aide les électroniciens à présenter un projet conforme aux normes de l'OSH.

Il existe deux critères principaux pour qu'un projet soit libre. Tout d'abord, les licences. Le code ou le microprogramme est automatiquement protégé par des droits d'auteur. Si un fabricant inclut un logiciel dans son projet, il doit lui attribuer activement une licence ouverte comme la *Gnu General Public License* ou la licence du MIT [3]. La documentation du produit, en particulier lorsqu'elle est volumineuse, tombe souvent sous le coup de la loi sur les droits d'auteur. Tous les fabricants ne savent pas qu'ils doivent explicitement attribuer une licence ouverte pour permettre aux utilisateurs en aval de réutiliser légalement le code ou de distribuer la documentation.

Le deuxième critère concerne la documentation. Schattenhofer : «La doc est importante. L'OSH consiste à permettre à quiconque d'étudier, de modifier et de fabriquer le matériel. Et pour cela, il faut les fichiers de conception. Vous ne pouvez pas les télécharger comme vous le faites pour les logiciels. Avec le matériel, la documentation ***est*** la source». Par conséquent, la documentation doit se conformer à certaines exigences. Un électronicien enthousiaste qui a publié sur YT une vidéo de son système d'irrigation pour bricoleurs veut probablement la partager, mais si la vidéo ne contient pas les informations nécessaires pour reproduire le projet, ce n'est pas vraiment une « source libre ».



Aperçu du processus d'examen communautaire de la documentation sur le matériel à source ouverte. (Image : avec l'aimable autorisation de l'Open Hardware Observatory)

Évaluation de la documentation

Schattenhofer et son équipe travaillent sur la plate-forme d'évaluation dont le principe est le suivant. Un candidat dépose un projet. Trois examinateurs évaluent la doc. Ils peuvent commenter ou signaler les parties absentes ou à améliorer. Lorsque sa doc répond à toutes les exigences, le projet reçoit une attestation.

Les exigences relatives à la doc sont définies dans la norme DIN SPEC 3105. Il s'agit d'une spécification accessible au public, soumise à la norme DIN allemande. Elle donne des critères concrets et mesurables pour la documentation conforme à l'OSH. La spécification est elle-même le résultat d'un processus de collaboration de la communauté de l'OSH. Conformément aux principes du matériel libre, tout le monde peut contribuer au développement du DIN SPEC 3105 sur la page Gitlab [4].

Il existe déjà un processus de certification en matière d'OSH facilité par l'*Open Source Hardware Association* (OSHWA) [5]. Il résulte lui aussi du besoin ressenti par la communauté de l'OSH de disposer de lignes directrices claires pour déterminer si un projet est vraiment libre. Le processus de certification de la OSHWA se concentre principalement sur la licence de l'OSH. Ce qui manquait encore, c'était des exigences pour la doc de l'OSH, ou en d'autres termes une définition claire de ce qui constitue une source. La plate-forme d'évaluation de l'OHO, basée sur les exigences énoncées dans la norme DIN SPEC 3105, vise à combler cette lacune.

Rejoindre la communauté

La plate-forme d'évaluation est déjà opérationnelle, mais encore en développement. Les développeurs ont reçu des commentaires de la communauté qui demande un flux de travail plus intuitif. Ils travaillent donc à l'amélioration de l'expérience utilisateur. Schattenhofer et l'équipe invitent les lecteurs d'Elektor à rejoindre la communauté OHO. Vous pouvez vous joindre à eux en téléchargeant votre projet pour qu'il soit examiné. Ou vous pouvez appliquer vos connaissances techniques en tant qu'examinateur. Mais, avertit M. Schattenhofer, il faut être prêt aussi à s'occuper d'un site qui n'est pas encore tout à fait prêt. Vous pouvez donc assumer un troisième rôle dans le projet OHO. Vous pouvez participer en tant que testeur de la plate-forme et aider à créer une meilleure interface utilisateur et un meilleur flux de travail.

Schattenhofer : «Pour l'instant, peu de gens connaissent le DIN SPEC 3105 et la plate-forme d'évaluation communautaire. Mais nous voulons construire une communauté active autour de l'*Open Hardware Observatory* pour promouvoir l'adoption généralisée du matériel à source ouverte. Rejoignez la communauté de l'OHO maintenant, vous participerez à son développement, en apportant vos propres idées. Pour rejoindre l'OHO, vous pouvez nous contacter à l'adresse suivante : info@oho.wiki.»

Note : OHO a été fondé comme coopération entre l'association à but non lucratif Open Source Ecology Germany e.V. (OSEG) et le projet franco-allemand de recherche OPEN ! Méthodes et outils pour le développement de produits communautaires [6].

200560-02

Votre avis, s'il vous plaît…
Vous pouvez poser des questions sur cet article ou le commenter en vous adressant à la rédaction d'Elektor redaction@elektor.fr

On contribué à cet article :

Texte : **Tessel Renzenbrink**
Rédaction : **Stuart Cording**
Traduction : **Denis Meyer**
Maquette : **Giel Dols**

LIENS

[1] Cette définition beaucoup utilisée a été établie par l'Open Source Hardware Association (sans rapport avec OHO)
[2] **Open Hardware Observatory** : https://en.oho.wiki/wiki/Open_Hardware
[3] **Liste de licences libres** : https://opensource.org/licenses
[4] **DIN SPEC** : https://gitlab.com/OSEGermany/OHS/
[5] **Open Source Hardware Association Certification** : https://certification.oshwa.org/
[6] **Partenaires fondateurs** : https://en.oho.wiki/wiki/Founding_partners#TU_Berlin_.2F_Research_project_OPEN.21

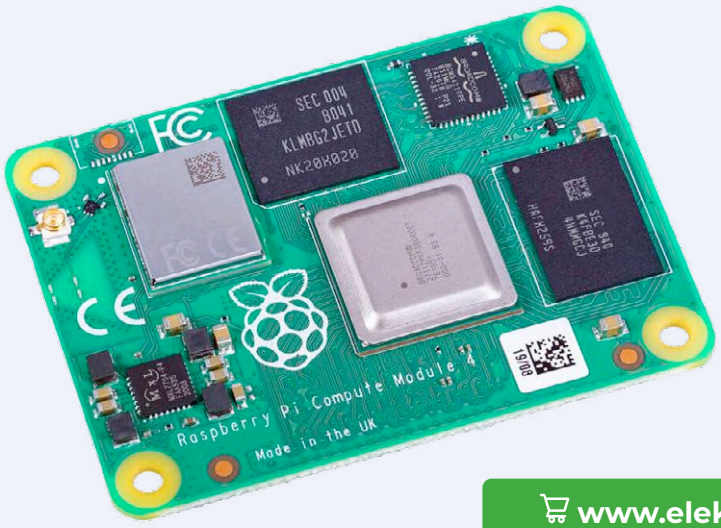
elektorstore

www.elektor.fr

e-choppe Elektor

des produits et des prix surprenants

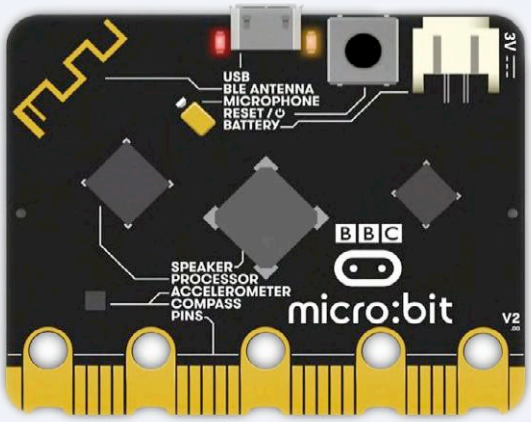
L'e-choppe Elektor a évolué. D'abord simple commerce de proximité pour nos productions (livres, magazines, kits et modules), c'est désormais une boutique en ligne bien rodée qui propose des produits surprenants à des prix très étudiés. Ce sont les produits que nous aimons et testons nous-mêmes. Si vous avez une suggestion, n'hésitez pas : sale@elektor.com. Seule exigence :
jamais cher, toujours surprenant !



Raspberry Pi Compute Module 4 (CM4)

Prix : à partir de 29,95 €

www.elektor.fr/19447



BBC micro:bit v2

Prix : 19,95 €
Prix (membres) : 17,96 €

www.elektor.fr/19488



Mini-pince brucelles électronique intelligente SeeedStudio DT71

Prix : 79,95 €

Prix (membres) : 71,96 €

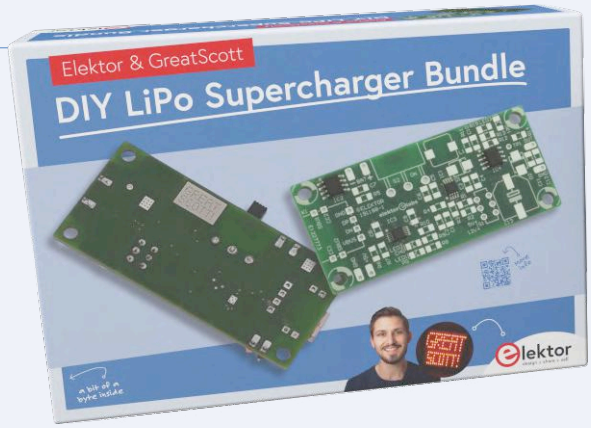
www.elektor.fr/19422



Raspberry Pi 400 – PC basé sur Raspberry Pi 4

Prix : 74,95 €

www.elektor.fr/19484

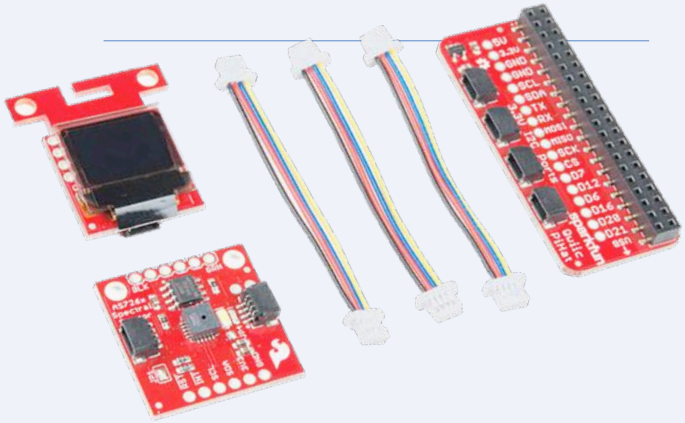


Kit de superchargeur et booster de LiPo à monter soi-même (par *GreatScott!*)

Prix : 24,95 €

Prix (membres) : 22,46 €

www.elektor.fr/19525



Kit de prise en main Qwiic de SparkFun pour Raspberry Pi

Prix : 69,95 €

Prix (membres) : 62,96 €

www.elektor.fr/19521

hexadoku casse-tête pour elektroniciens

La dernière page de votre magazine propose toujours une grille de sudoku particulière. Éteignez le fer à souder, oubliez les écrans , 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 €.

Où envoyer votre réponse ?

Envoyez votre réponse (les chiffres sur fond grisé) avec vos coordonnées par courriel, avant le **10 février 2021** à l'adresse **hexadoku@elektor.fr**

Les gagnants

La solution de la grille du numéro de novembre/décembre 2020 est **F1235.**

Les cinq bons Elektor d'une valeur de **50 €** vont à Huub Liebregts (Pays-Bas) – G.D. (Joe) Young (Canada) – Jean-Claude Carré (France) – Ralf Boldt Velbert (Allemagne) – R. Torfs (Belgique).

Bravo à tous les participants et félicitations aux gagnants !

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

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

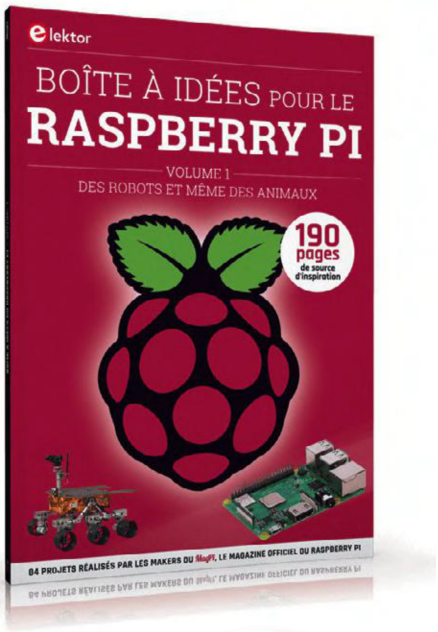
Tout recours est exclu, de même que le sont, de ce jeu, les personnels d'Elektor International Media et leur famille. Un seul gagnant par foyer.

BOÎTE À IDÉES POUR LE RASPBERRY PI

VOLUME 1

DES ROBOTS ET MÊME DES ANIMAUX

Depuis son lancement en mars 2012, plus de 30 millions d'exemplaires du **nano-ordinateur Raspberry Pi** ont été vendus. Cet ordinateur a contribué à révolutionner l'enseignement de l'informatique et a formidablement stimulé la créativité dans le monde de l'électronique. Le succès de la « tarte à la framboise » (traduction littérale de *Raspberry Pie*) est tel que même l'industrie de la microélectronique la trouve désormais à son goût et l'incorpore dans ses produits.



Eben Upton, son créateur, souligne qu'à l'origine, les acheteurs étaient presque tous des *makers*, généralement adultes. Toutefois la plupart d'entre eux étaient aussi des ingénieurs professionnels. Rapidement, ces personnes, convaincues dans leurs activités personnelles par la puissance du Raspberry Pi comme plateforme informatique stable, l'ont adoptée aussi dans leur vie professionnelle.

La Fondation Raspberry Pi publie un magazine écrit pour et par la communauté du Raspberry Pi : le *MagPi* (en anglais). Ce magazine s'adresse aux passionnés qui ne veulent rien manquer de l'actualité du nano-ordinateur et du monde numérique. Il contient des bancs d'essai de nouveaux produits, de nombreux tutoriels et un grand nombre de projets. Les membres actifs de la communauté du Raspberry Pi y partagent leur expérience.

La maison d'édition Elektor s'est associée à la Fondation Raspberry Pi pour publier en français le magazine *MagPi*. Cet ouvrage qui rassemble ici 84 projets publiés dans le *MagPi* permet de découvrir des utilisations étonnantes du Raspberry Pi et de s'en inspirer.
Grâce à la petite taille et aux possibilités infinies du Raspberry Pi, les *makers* du monde entier peuvent donner vie à leurs idées les plus folles.

34,95 €

Partez en voyage dans un monde où tout est faisable !

- Présentation de 84 créations réalisées par les makers du MagPi, le magazine officiel du Raspberry Pi
- 190 pages de source d'inspiration (en français !)
- Découverte d'utilisations inouïes, d'accessoires et diverses technologies
- Porte d'entrée dans l'incroyable et très active communauté mondiale du Raspberry Pi

www.elektor.fr/boite-a-idees-pour-le-raspberry-pi

ISBN 978-2-86661-213-9

Rejoignez les électroniciens de la communauté Elektor

Devenez membre elektor GOLD membership maintenant !



- accès à l'archive numérique depuis 1978 !
- 6x magazine imprimé Elektor
- 9x magazine numérique (PDF) dont Elektor Industry (EN)
- 10 % de remise dans l'e-choppe et des offres exclusives pourles membres
- le DVD annuel d'Elektor
- accès à plus de 1000 fichiers Gerber, collaboration avec les milliers d'électroniciens d'Elektor LAB, et une ligne directe avec nos experts !
- possibilité de voir votre projet publié ou vendu par notre boutique en ligne

Également disponible

abonnement elektor GREEN membership sans papier !

- accès à l'archive numérique d'Elektor
- 10 % de remise dans l'e-choppe
- 6x magazine Elektor (PDF)
- offres exclusives
- accès à plus de 1000 fichiers Gerber

www.elektormagazine.fr/membres

elektor
design › share › sell