ISSN 0181-7450 F : 15,50 €

(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

elektor 492

créer > partager > vendre

NOVEMBRE & DÉCEMBRE 2021
ELEKTORMAGAZINE.FR

60 years young

TRAITEMENT D'IMAGES FACILITÉ !





premiers pas avec le kit Jetson Nano de Nvidia et Edge Impulse p. 6

dans ce numéro :

- Arduino Nano RP2040 Connect en détail
- kit de mesure du CO_2 pour salle de classe
- extension à mémoire analogique, protégée par ampoule de flash
- étalonneur de haute précision
- balises Bluetooth : la pratique
- création d'interfaces graphiques en Python avec guizero
- Elektor 60 ans : que la lumière soit !
- voyage dans les réseaux neuronaux : les neurones logiques
- visite à domicile : Junior Computer

et bien d'avantage !



traceur GPS à code source ouvert p. 16
pas besoin de serveurs nuage tiers



synthétiseur de bruit p. 24
matériel minimal, logiciel intelligent !



test de préconformité CEM avec RSIL p. 60
comment utiliser le matériel

L 19624 - 492 - F: 15,50 € - RD

NOTRE GAMME

PAR DES TECHNICIENS POUR LES TECHNICIENS

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

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

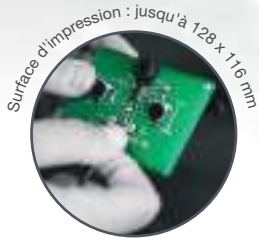
Nos responsables produits sont employés par Reichelt depuis de nombreuses années et connaissent les exigences de nos clients. Ils rassemblent une large gamme de produits de qualité, à la fois parfaits pour les besoins dans les domaines de la recherche et du développement, la maintenance, l´infrastructure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

Accélérez votre développement de hardware

Imprimante de circuits imprimés Voltera V-One

Réalisez des prototypes sur votre propre bureau avec l'imprimante Voltera V-One. Elle permet de créer rapidement et en toute simplicité des circuits imprimés personnalisés, des pièces uniques aux petites séries.

- Création de circuits imprimés à deux couches
- Impression de pistes conductrices, perçage et brasage (refusion)
- Efficacité accrue de la recherche et du développement







NOUVEAU

N° de commande :
VOLTERA V-ONE

5.041,01 (4200,84)

Notre dernière sélection d'imprimantes 3D, de scanners 3D et de matériaux de production est toujours disponibles en ligne !

En savoir plus ▶ www.reichelt.com/3d-printing



Types de paiement : VISA PayPal

PRIX DU JOUR! Prix à la date du: 11. 10. 2021

- Excellent rapport qualité prix
- Plus de 120 000 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° 492 – novembre-décembre 2021

ISSN 0181-7450
Dépôt légal : novembre 2021
CPPAP 1125 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 :
Raoul Morreau
Tél. : +31 (0)6 4403 9907
Courriel : raoul.morreau@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.



rédacteur en chef d'Elektor Magazine

L'apprentissage automatique facilité

À l'heure actuelle, il n'y a guère de sujet plus fascinant dans le domaine de l'informatique et de l'électronique que l'apprentissage machine (*Machine Learning*) par le biais des réseaux neuronaux. Des applications telles que la commande vocale et la reconnaissance d'images fonctionnent même sur des cartes compactes, peu énergivores et bon marché. Cela s'explique par le fait que le réseau a déjà été entraîné sur des plateformes plus puissantes, par exemple dans le nuage. L'enregistrement et le traitement des données nécessaires, l'entraînement et le test du réseau, et enfin le transfert vers le microcontrôleur d'exécution ne sont pas des tâches faciles – surtout pour la cohorte d'ingénieurs en électronique qui veulent s'aventurer dans cette voie passionnante, que ce soit par curiosité personnelle ou à titre professionnel.

C'est là que des *frameworks* de ML comme la plateforme web Edge Impulse entrent en jeu. Pour les premiers tests, on peut se passer de carte à microcontrôleur. Un smartphone suffit pour enregistrer des images ou des sons et pour entraîner un premier réseau de manière à reconnaître des modèles. Mon collègue Mathias Claussen a essayé le tout avec le kit Jetson Nano de Nvidia, un simple appareil photo et une pile de cartes à jouer (p. 6). Son article est un bon guide pas-à-pas pour commencer.

Nous souhaitons également vous inviter cordialement au salon *productronica*, qui se déroulera du 16 au 19 novembre 2021 à Munich (Allemagne). En collaboration avec *Messe München*, nous organisons cette année encore le célèbre concours de start-ups « Fast Forward » (www.elektormagazine.com/pffwd21). Mais ce n'est pas tout, car nous voulons rendre hommage à six décennies d'Elektor lors du salon *productronica*. Cette année, nous animerons le *World Ethical Electronics Forum*, en abrégé WEEF (www.elektormagazine.com/weef), en collaboration avec *productronica* et le magazine *Elektronikpraxis*. Des conférences intéressantes vous attendent sur le thème de l'éthique et de la durabilité en électronique, animées par mon collègue Stuart Cording. Passez nous voir !

ELEKTOR A 60 ANS : LA FÊTE CONTINUE !



Elektor célèbre ses 60 ans de publications et d'innovations dans le monde de l'électronique avec quelques projets spéciaux passionnants. Nous préparons actuellement le « World Ethical Electronics Forum 2021 » qui aura lieu le 18 novembre 2021. Vous avez des idées pour des projets spéciaux ? Envoyez-moi vos idées : shenja.panik@elektor.com

Shenja Panik, coordinatrice du comité « Elektor 60 »

notre équipe



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

FIPP Connecting Global Media

Elektor est membre de la FIPP, une organisation qui « se développe depuis presque 100 ans pour réunir des propriétaires de médias et des créateurs de contenu du monde entier ».

DEUTSCHE FACHPRESSE

Elektor est membre de VDZ (association d'éditeurs de magazines allemands) qui « représente les intérêts communs de 500 éditeurs allemands grand public et B2B. »

DANS CE NUMÉRO

Elektor n° 492
novembre - décembre 2021

traitement d'images avec le kit Jetson Nano de Nvidia

2e partie : reconnaissance d'images

6





Rubriques

3 **Édito**

14 **Elektor Jumpstarter : quoi de neuf ?**
Campagnes à venir

31 **Démarrer en électronique… (10)**
Passons aux bobines (ou inductances)

80 **Visite à domicile**
Junior Computer en forme après 40 ans de sommeil

92 **Questions d'éthique**
Le corps physique de l'intelligence artificielle

94 **Projet 2.0**
Corrections, mises à jour et courriers des lecteurs

114 **Hexadoku**
Casse-tête pour elektroniciens

Articles de fond

22 **Testeur multifonction LCR-T7 de Joy-IT**
Test de semi-conducteurs passifs, discrets et de télécommandes IR

34 **Voyage dans les réseaux neuronaux**
2e partie : les neurones logiques

50 **Balises Bluetooth : la pratique**
Géolocalisation intra muros

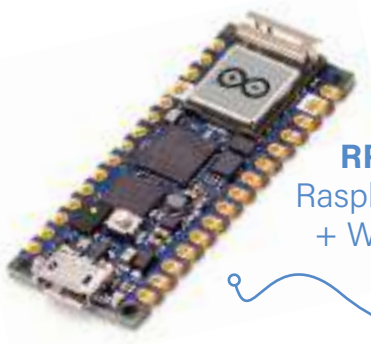
57 ***C Programming on Raspberry Pi***
Extrait de livre : communiquer par Wi-Fi

69 **Propeller 2 de Parallax (5)**
La fonction de « broche intelligente »

89 **Arduino Nano RP2040 Connect en détail**
Raspberry Pi RP2040 + Wi-Fi + Bluetooth

96 **Création d'interfaces graphiques en Python**
avec guizero

108 **Elektor 60 ans**
Que la lumière soit !



Arduino Nano RP2040 Connect
Raspberry Pi RP2040 + Wi-Fi + Bluetooth
89



Kit de mesure du CO_2 pour salle de classe
Montage à base d' ESP8266
101

combattez le feu par le feu !





sécurité protégée par une ampoule de flash

40

étalonneur de haute précision

–10 V à +10 V, 0 à 40 mA, 0,001%



84

Réalisations

6 **Traitement d'images avec le kit Jetson Nano de Nvidia**
2e partie : reconnaissance d'images

16 **Traceur GPS à code source ouvert**
Traccar cartographie les déplacements de véhicules, sans recours à un serveur tiers du nuage

24 **Synthétiseur de bruit**
Du bruit à la musique avec le PRBSynth1

40 **Problèmes de sécurité ? Combattez le feu par le feu !**
Extension à mémoire analogique, protégée par ampoule de flash, pour la boîte à témoin d'effraction

48 **Kit du LCR-mètre 2 MHz d'Elektor**
Le projet Jumpstarter en un coup d'œil !

60 **Test de préconformité CEM pour un projet alimenté en courant continu**
Partie 2 : le matériel et son utilisation

74 **Modbus sans fil (partie 1)**
Matériel et programmation

84 **Construire son propre étalonneur de haute précision**
–10 V à +10 V, 0 à 40 mA, 0,001 %

101 **Kit de mesure du CO_2 pour salle de classe**
Montage à base d'ESP8266, conçu par l'Université des Sciences Appliquées d'Aix-la-Chapelle

106 **MK484, radiorétro PO/GO**
…Toujours le plaisir de construire !

Bientôt dans ces pages

Le numéro de janvier-février 2022 d'Elektor
Vous retrouverez dans le prochain magazine Elektor l'habituel mélange stimulant de réalisations originales, de circuits soigneusement étudiés, d'articles de fond, de sujets nouveaux, de trucs et d'astuces pour les électroniciens actifs.

Quelques-uns des points forts :

- Pi-KVM : gestion à distance d'autres postes de travail
- Kit de robotique et commande de moteur bon marché
- Mesures audio avec interface audio USB
- Testeur de servo polyvalent
- Réparation de batterie au lithium
- Mises à jour « over the air » pour Arduino et ESP
- Moteurs électriques de taille moyenne
- Pratique de NB-IoT

et bien d'avantage !

Ce numéro paraîtra le 6 janvier 2022.

elektor
créer > partager > vendre

TRAITEMENT D'IMAGES avec le kit Jetson Nano de Nvidia



Figure 1. Kit de développement Jetson Nano. (Source : developer.nvidia.com)

2e partie : reconnaissance d'images

Mathias Claußen (Elektor)

Si vous recherchez un moyen simple pour vous lancer dans le monde des réseaux neuronaux, vous devriez jeter un coup d'œil à Edge Impulse. Vous aurez ainsi accès à un large éventail de cartes et dispositifs de développement, par ex. pour capturer et construire des données d'images utilisables, ce qui vous permettra de créer des réseaux neuronaux dans le nuage. Vous pourrez ensuite les utiliser dans vos propres applications. Nous avons ainsi testé Edge Impulse, une carte de développement Jetson Nano et un certain nombre de cartes à jouer.

Après la présentation de Jetson Nano (**fig. 1**) et du kit JetBot AI de SparkFun (**fig. 2**) dans la première partie de cette série [1], il est maintenant temps de faire nos premiers pas dans la reconnaissance d'images. Pour reconnaître une image à l'aide du Jetson Nano, il faut entraîner un logiciel et un réseau neuronal. Comme nous l'avons déjà décrit dans l'article en ligne « Faire du café grâce au MAX78000 et à une pincée d'IA » [2], l'entraînement du réseau n'est pas une tâche facile. D'abord vous devrez vous familiariser avec l'application de type console qui s'appuie sur le *framework* PyTorch ou la bibliothèque TensorFlow. La création d'un réseau neuronal sur votre propre ordinateur prendra également un certain temps, en fonction de la carte graphique et du processeur de la machine.

Les défis ne s'arrêtent pas là : il faut avant tout entraîner le réseau neuronal à l'aide d'un ensemble de données. Des applications supplémentaires sont également nécessaires pour construire l'ensemble de données et le préparer pour l'entraînement. Un certain temps est donc nécessaire pour pouvoir entraîner notre premier réseau neuronal.

Edge Impulse

Si vous cherchez une introduction relativement simple au monde des réseaux neuronaux, n'hésitez pas à découvrir les capacités d'Edge Impulse [3]. Il s'agit d'une plateforme web destinée à traiter vos ensembles de données pour construire des réseaux neuronaux et les porter sur une grande variété de dispositifs. Ce portage peut aussi bien concerner un Arduino qu'un ordinateur portable doté d'un processeur x86/AMD64. Ici, Edge Impulse sert à créer un réseau neuronal, ensuite utilisé avec le kit JetBot pour reconnaître quelques objets. Les données peuvent être capturées à l'aide de différents appareils, par ex. un PC avec un navigateur web ou un smartphone. L'ensemble de données est envoyé directement aux serveurs Edge Impulse et y est stocké. Si vous disposez déjà d'un tel ensemble de données, vous pouvez simplement le télécharger via l'interface web.
Le tri et l'étiquetage du jeu de données s'effectuent ensuite via l'interface web. Les paramètres de création du réseau neuronal y sont également définis et configurés.
La création d'un réseau neuronal est très exigeante en puissance de calcul. Les données sont donc téléchargées sur le serveur Edge Impulse et y sont converties en réseau neuronal. L'entraînement a donc lieu dans le nuage et n'impose pas de charge à votre propre ordinateur. Le réseau neuronal créé peut ensuite être converti en bibliothèque de programmes, téléchargé puis utilisé dans d'autres applications.

Nous détaillons toutes ces étapes en construisant la modeste application de reconnaissance d'images proposée ci-dessous. Nous avons publié un article sur l'utilisation d'Edge Impulse pour la détection du bruit [4] dans le magazine Elektor Industry, autre publication d'Elektor.



Figure 2. Kit JetBot AI de SparkFun. (Source : SparkFun)



Figure 3. Cartes UNO.

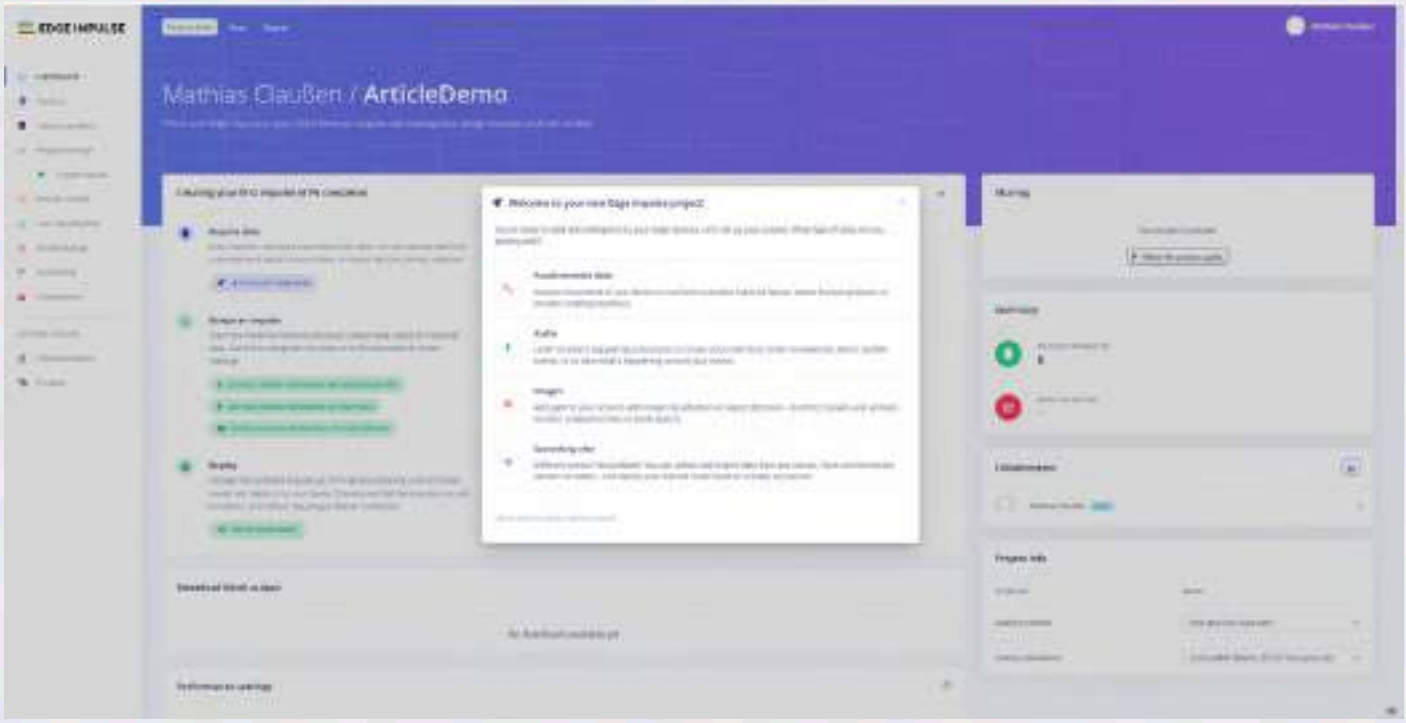


Figure 4. Assistant Edge Impulse.

Distribuer les cartes

Pour commencer, il est bon d'utiliser des objets visuellement très distincts comme éléments de test. Nous avons choisi différentes cartes à jouer du jeu UNO de Mattel (**fig. 3**), que nous avions à portée de main, mais vous pourriez utiliser d'autres objets, comme des panneaux de signalisation. Un Jetson Nano et un kit JetBot AI de SparkFun (fig. 2), qui contient une caméra à 8 mégapixels, forment la plateforme matérielle.

Notre premier objectif est de pouvoir distinguer l'image de la carte UNO n°1 de la carte n°2 et du verso d'une carte. Pour ce faire, comme nous l'avons déjà mentionné, nous devrons entraîner le réseau neuronal en utilisant des images des objets à reconnaître. Pour cela, découvrons l'acquisition de données et l'interface d'Edge Impulse.

Un projet Edge Impulse

Il faut d'abord s'inscrire pour pouvoir utiliser Edge Impulse. Nous utilisons ici la version gratuite de la plateforme, qui présente certaines limitations d'utilisation (nous y reviendrons). Après l'enregistrement, un nouveau projet peut être créé ; nous sommes alors accueillis par un assistant (**fig. 4**).

Lorsqu'on nous demande quel type de données nous souhaitons traiter, nous répondons par l'option *Images*, puisque ce sont les images des cartes UNO qui nous intéressent pour ce test. Dans ce cas, les cartes UNO doivent être classées (*Classify a single object*) car, dans un premier temps, nous ne voulons traiter qu'une seule carte et déterminer de laquelle il s'agit, comme le montre la **figure 5**. Le projet est maintenant configuré avec les paramètres de reconnaissance. La prochaine étape consiste à construire le jeu de données avec lequel nous pouvons commencer l'entraînement.

Acquisition de données

Dans ce projet, nous produirons nous-mêmes les données pour l'entraînement du réseau neuronal. Pour la reconnaissance d'images, il existe des jeux de données contenant des milliers d'images déjà triées dans certains groupes ; selon votre application, il peut être judicieux d'utiliser ce type de données. Nous n'avons pas trouvé de jeu de données de cartes UNO, nous devons donc en créer un. C'est un bon exercice pour passer en revue les étapes nécessaires pour créer soi-même un jeu de données d'images et montrer comment éviter certains pièges.

Pour capturer les images, l'idéal est d'utiliser la même caméra que celle qui sera utilisée ultérieurement pour la reconnaissance d'images. Ainsi le réseau neuronal apprend en même temps les particularités de la caméra. Afin de collecter les données à l'aide du Nvidia Jetson, nous pouvons installer un client pour Edge Impulse. Celui-ci peut alors envoyer les images (et l'audio si nécessaire) à Edge Impulse.

Pour installer le client, suivez les instructions [5] fournies par Edge Impulse. Pour le Nvidia Jetson, il est nécessaire de se connecter à la carte à l'aide du protocole SSH, puis d'entrer les commandes via une console. Autre solution : vous pouvez connecter une souris, un clavier et un moniteur et entrer les lignes de commande directement dans le système Linux qui fonctionne sur le Jetson Nano.

Lorsqu'une liaison SSH a été établie ou qu'une console est ouverte, il suffit de saisir les commandes figurant dans les instructions. Une connexion internet est nécessaire pour l'installation. Si le Jetson Nano n'est pas encore connecté à l'internet, commencez par cette étape.

Ensuite, nous entrons

```
wget -q -O - https://cdn.edgeimpulse.com/build-system/jetson.sh | bash
```

dans la console pour démarrer l'installation de *edge-impulse-linux*. L'installation prend un certain temps. Une fois qu'elle est terminée, entrez `edge-impulse-linux -clean` dans la console. L'outil requiert nos informations d'accès et le périphérique audio et vidéo que nous utiliserons pour l'acquisition des données. Une fois que tout est configuré, le Jetson Nano nouvellement configuré apparaît sur la page d'Edge Impulse sous la rubrique *Devices* (**fig. 6**). Grâce à cela, nous pouvons maintenant obtenir des images à partir de l'appareil photo.

Sur la page web d'Edge Impulse, nous pouvons les capturer à l'aide de l'option de menu *Data aquisition* (**fig. 7**). Comme une source audio a également été configurée ici, il faut d'abord commuter le capteur sur *Camera*

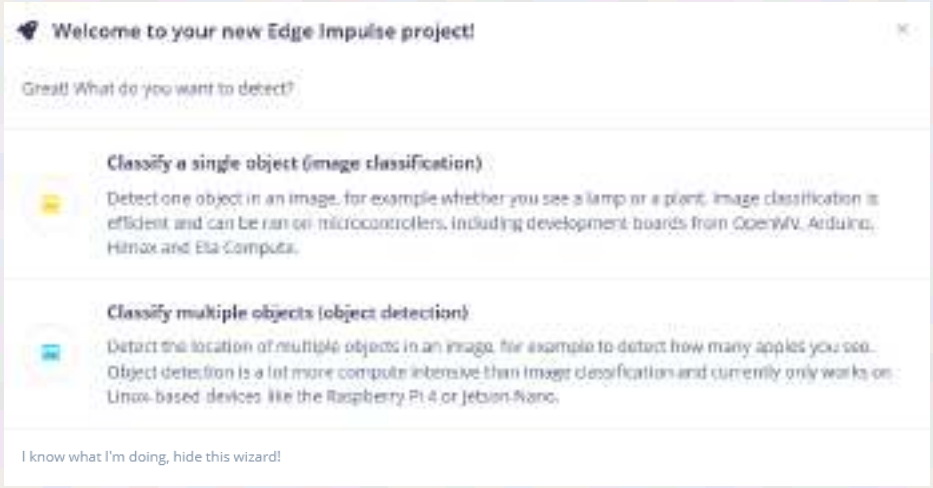


Figure 5. Sélectionner la classification des objets.



Figure 6. Définir JetBot comme dispositif destiné à acquérir les données.

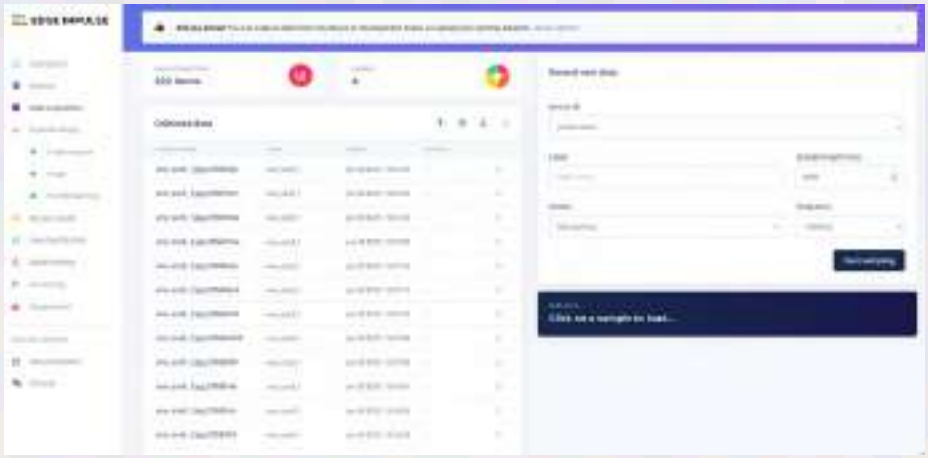


Figure 7. Acquisition des données avec Edge Impulse.

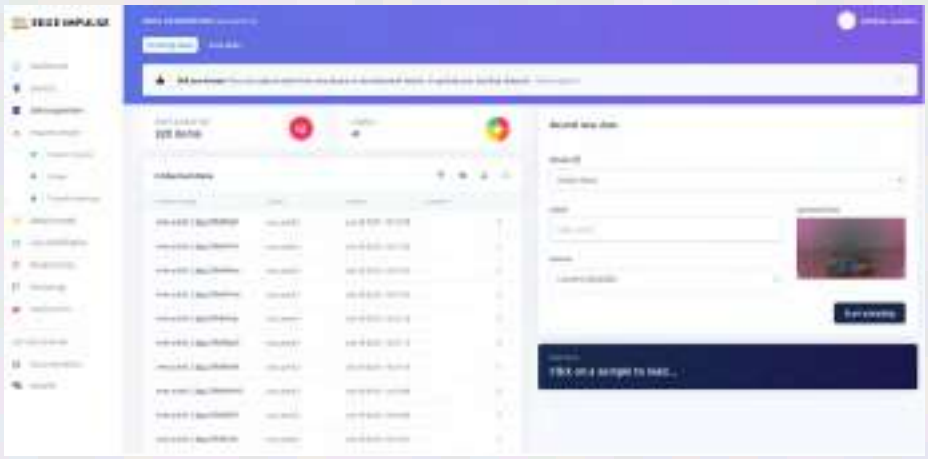


Figure 8. Configuration de la caméra.

(640x480). Le flux de la caméra (*Camera feed*) apparaît alors, montrant les images en direct (**fig. 8**). À l'aide de la mention *Label* (libellé), nous pouvons maintenant indiquer le type de chaque image (« class ») (c.-à-d. aucune carte, carte n°1, carte n°2 ou le verso de la carte).
Passons maintenant à la partie la plus ennuyeuse – l'acquisition des données de chaque carte selon différentes orientations. Au début, environ 80 à 100 images doivent être prises pour chaque carte que nous reconnaîtrons plus tard. Le libellé associé aux images enregistrées doit également être attribué en même temps.

Entraînement initial

Une fois toutes les images prises, il est temps de procéder à la première session d'entraînement. Avec *Impulse design* (**fig. 9**), nous pouvons maintenant définir l'impulsion (c'est-à-dire le mode opérationnel du réseau neuronal). Pour la taille de l'image, nous utilisons 160 × 160 pixels et l'option *Fit shortest axis* (adapter l'axe le plus court) dans le champ déroulant *Resize mode* (mode de redimensionnement). Comme aucune couleur ne sera évaluée, nous pouvons sélectionner *Grayscale* (niveaux de gris) (**fig. 10**). Après avoir cliqué sur *Save parameters* (enregistrer les paramètres), les classes de reconnaissance peuvent être créées sur la page suivante.
Le nuage de points affiché dans l'explorateur de caractéristiques montre la distribution de nos images sur la base d'un premier entraînement approximatif. Il permet d'identifier les données incorrectement étiquetées. La proximité des points individuels (*cluster*) donne une indication de la difficulté qu'a le réseau neuronal à distinguer les cartes les unes des autres (**fig. 11**).
Les paramètres d'entraînement peuvent maintenant être définis dans *Transfer learning* (**fig. 12**). Les limites de la version gratuite ont été mentionnées au début et nous les rencontrons maintenant. Les serveurs d'Edge Impulse allouent 20 min de temps de calcul pour le processus d'entraînement. Pour nos quatre classes, ce temps est suffisant pour obtenir des résultats utiles. Si le processus d'entraînement nécessite plus de 20 min, le processus est simplement annulé.
Le nombre de cycles d'entraînement affecte sa qualité, ainsi que le temps nécessaire. Dans le cas présent, la valeur est fixée à 20 pour l'apprentissage relatif aux données ; l'augmentation du nombre de cycles de 20 à 50 améliore le résultat. Comme



Figure 9. Créer une nouvelle impulsion.

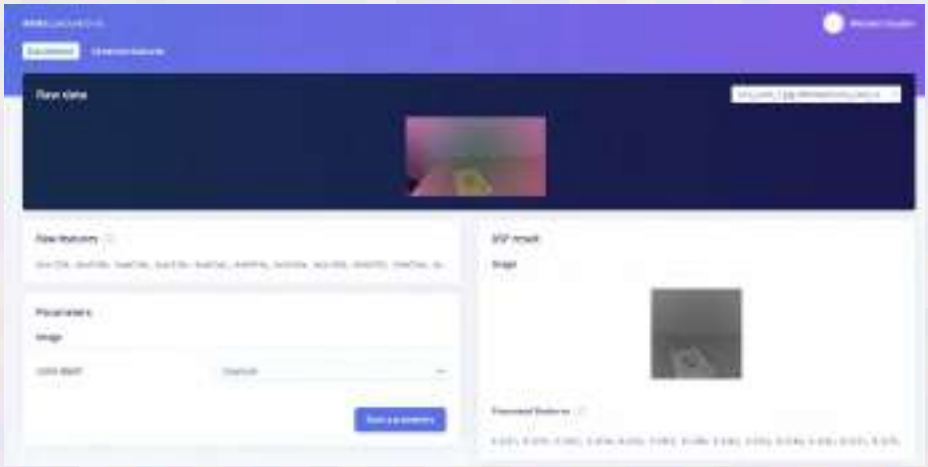


Figure 10. Travailler en niveaux de gris.

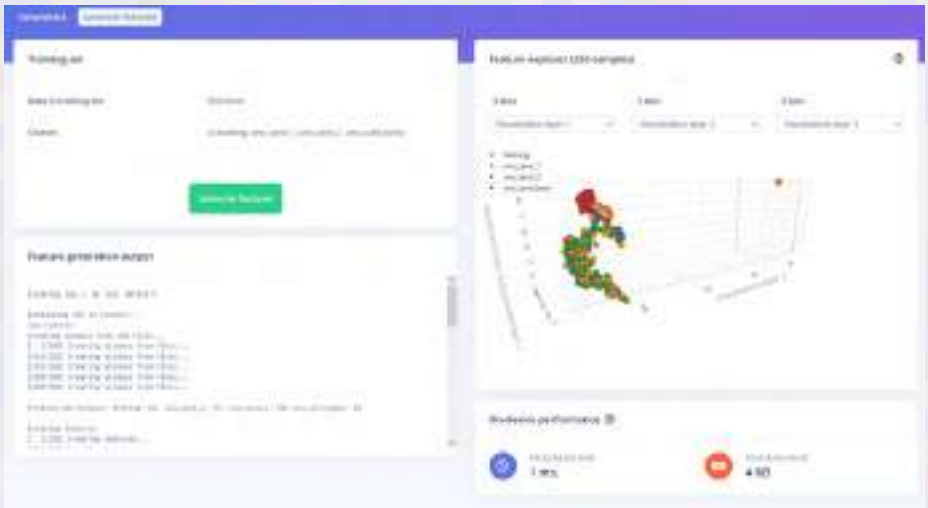


Figure 11. Génération des caractéristiques du réseau neuronal.

l'ensemble de données utilisé ici est très facile à gérer, l'augmentation des données peut servir à améliorer la reconnaissance. Cette augmentation des données déforme, étire et fait pivoter les images afin d'élargir artificiellement l'ensemble de données. Cela peut améliorer la confiance dans la reconnaissance.

Une fois l'entraînement terminé, des statistiques sont calculées (**fig. 13**) qui montrent le bon fonctionnement de notre réseau neuronal. Le réseau qui vient d'être créé permet d'effectuer un test sur le Jetson.

Premier test à l'aide du Jetson

Une fois le réseau neuronal entraîné, nous devrons ouvrir à nouveau une console sur le Jetson Nano. Le moyen le plus simple est de se connecter au Nano à l'aide de SSH. La commande `edge-impulse-linux-runner --clean` permet de lancer un assistant qui demande nos données d'accès. Si plusieurs projets ont été créés, il faut sélectionner celui qui doit être testé.

Une fois que tout a été configuré, la console produit une liste de classification pendant que la caméra fonctionne (**fig. 14**). La valeur numérique indiquée après chaque classe individuelle indique le niveau de confiance avec lequel la classe apparaît dans l'image qui vient d'être enregistrée, avec 1,0 comme maximum (c'est-à-dire 100 % de certitude) et 0,00 à titre de minimum.

Si nous plaçons maintenant la carte n°1 face à la caméra, nous pouvons voir que le niveau de confiance change et qu'un 1 est reconnu avec un niveau de confiance de 0,9420 (c.-à-d. un très haut degré de certitude).

Se fier entièrement au résultat donné par la console n'est pas très pratique. Il se peut qu'une carte ne soit pas reconnue correctement et que vous vouliez savoir ce que le Jetbot voit réellement. Pour cela, l'outil *edge-impulse-linux-runner* ne se contente pas d'exécuter le réseau neuronal, il fournit également un serveur web accessible via le port 4912 (**fig. 15**). Sur la page web, vous pouvez suivre en direct quelle carte est actuellement reconnue ou, comme le montre la figure 15, si elle n'est pas reconnue correctement. Il est également possible de produire de nouvelles images, avec lesquelles le réseau neuronal peut être réentraîné. Pour ce faire, utilisez CTRL+C pour arrêter *impulse-linux-runner*. Avec `edge-impulse-linux`, Jetbot apparaît à nouveau comme une source de données dans le portail web Edge

Impulse et de nouvelles images peuvent être enregistrées.

Deuxième cycle d'entraînement

Avec de nouvelles données valides supplémentaires, vous pouvez lancer une nouvelle procédure d'entraînement. Plus il y a de données ajoutées à la catégorie avec laquelle le réseau neuronal actuel a des difficultés, plus le résultat sera satisfaisant. Avec `edge-impulse-linux`, le JetBot redeviendra le collecteur de données ; nous pouvons prendre des images des cartes à jouer et les étiqueter de manière appropriée. Lorsque l'acquisition de nouvelles données est terminée, le processus d'entraînement peut être de nouveau lancé avec *Retrain Model* pour réutiliser les paramètres précédents avec les nouvelles données (**fig. 16**). Une fois cela fait, nous pouvons tester la nouvelle version du réseau sur le Jetson. En utilisant `edge-impulse-linux-runner`, le nouveau réseau neuronal sera téléchargé et exécuté sur JetBot. Nous pouvons maintenant utiliser un navigateur web pour voir en temps réel comment les images des cartes sont classées avec une plus grande confiance (**fig. 17**).

De l'entraînement à l'application

Une fois les données enregistrées et le réseau neuronal entraîné, la question se pose de savoir comment l'utiliser dans une application spécifique. Edge Impulse propose des kits de développement logiciel pour prendre en charge différents langages de programmation. Si on veut coder en langage C / C++, le JetBot rencontre alors un problème concernant la commande de la caméra et la bibliothèque OpenMV utilisée. Aucune image ne peut être capturée à l'aide de la caméra avec cette bibliothèque. (La capture d'images via Edge Impulse décrite ci-dessus n'est pas affectée ; elle ne semble pas utiliser OpenMV.)

Le processeur graphique du Jetson Nano n'est pas non plus utilisé pour le traitement des images, tous les calculs sont donc effectués par l'unité centrale. Il n'y a peut-être pas de perte majeure pour les petites cartes comme le Raspberry Pi, mais avec le Jetson Nano, une grande partie de la capacité de calcul de l'IA est ainsi perdue.

Le travail d'entraînement effectué pour le réseau n'a pas été vain. Il est possible d'exporter le réseau neuronal entraîné sous la forme d'une bibliothèque indépendante de la plateforme, ce qui lui permet de fonctionner sur d'autres systèmes (voir



Figure 12. Définir les paramètres pour l'entraînement.



Figure 13. Résultats de l'entraînement.

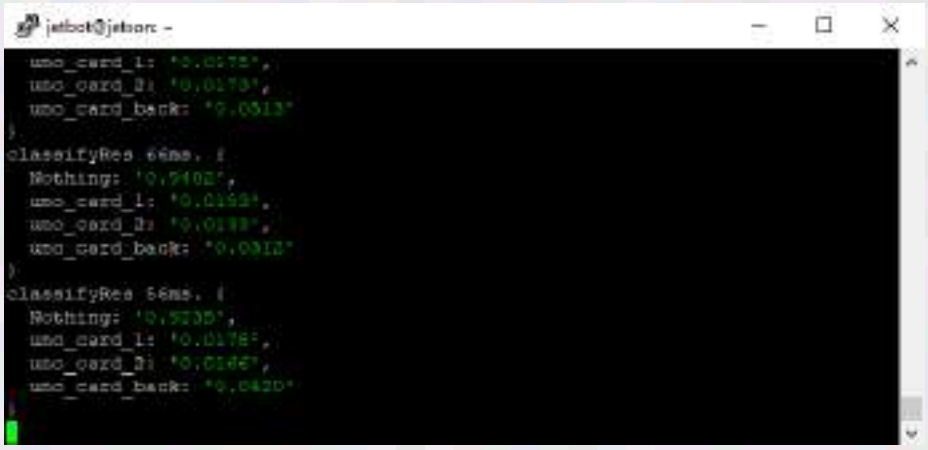


Figure 14. Affichage de la reconnaissance d'images sur la console.



Figure 15. Visualisation en temps réel de la reconnaissance d'images à l'aide d'un navigateur.

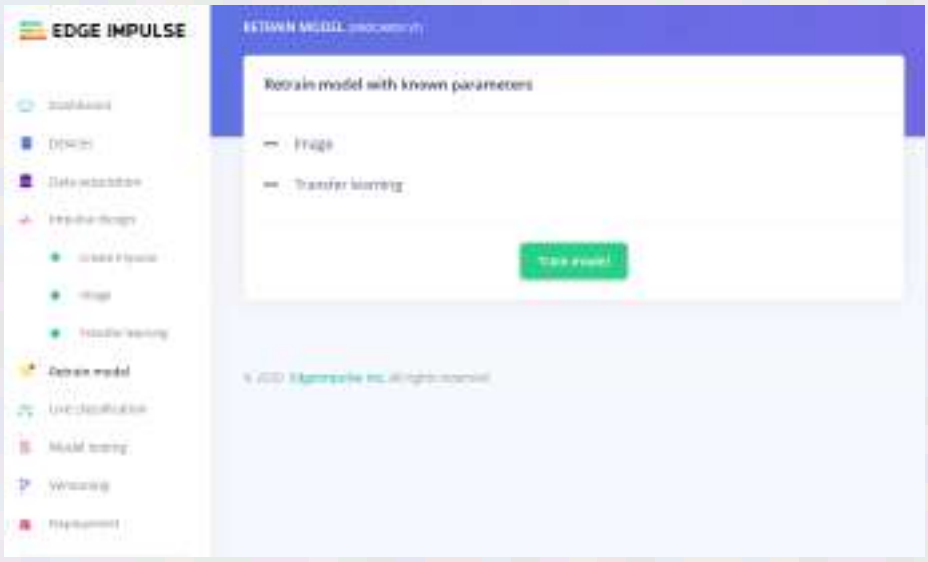


Figure 16. Nouvel entraînement avec davantage de données.

téléchargement [6]). Il n'est pas nécessaire qu'il s'agisse d'un Raspberry Pi ; un ESP32 avec une caméra est tout aussi approprié pour utiliser le réseau que nous avons entraîné.

Malheureusement, les données brutes enregistrées avec Edge Impulse ne sont disponibles que là. Les images que nous avons étiquetées ne peuvent pas être exportées vers d'autres plateformes ou vers un système local pour être traitées. C'est un compromis que nous devrons accepter pour utiliser la plateforme Edge Impulse.

Ceci n'est pas un filigrane, c'est un cheval…

L'évolution nous a dotés d'une capacité sophistiquée à reconnaître des formes. Nous pouvons apprendre par ex. à quoi ressemble un cheval. Si l'on nous montre un certain nombre de photos de ces animaux, nous apprenons que s'il a quatre pattes, des sabots, une certaine taille et des proportions spécifiques, il s'agit probablement d'un cheval. Au cours de notre vie, notre propre définition du cheval s'affine (par ex. un âne n'est pas un cheval, même s'il en partage de nombreux attributs).

Avec un traitement d'images classique, on apprend à un algorithme les caractéristiques à rechercher dans les images pour reconnaître les chevaux (par ex. sabots, yeux, crinière, couleur, proportions). Toutefois, cette méthode a ses limites lorsqu'il s'agit de généraliser la détection des chevaux. C'est ici qu'entrent en jeu les réseaux neuronaux. Il suffit de leur donner un grand nombre d'images (plusieurs milliers) et de laisser aux algorithmes qui forment le réseau neuronal le soin d'identifier un cheval. Cette approche semble plausible. Qu'est-ce qui peut bien se passer ?

Comme l'a mentionné Stuart Cording lors du webinaire « Get to know the NVIDIA Jetson Nano Developer Kit » [7], alors qu'il travaillait sur un projet d'IA, il a téléchargé une série d'images trouvées via un moteur de recherche afin de les utiliser comme données pour identifier un cheval sur une photo. Malheureusement, les images contenaient un filigrane. Cela signifie que le réseau neuronal identifiait l'image comme étant un cheval s'il détectait un filigrane dans l'image traitée.

Quelque chose de similaire s'est produit lorsque nous avons tenté pour la première fois de construire un ensemble de données avec les cartes UNO. Une partie d'une main a



Figure 17. Image en temps réel indiquant une plus grande confiance dans la reconnaissance.

été capturée dans certaines des images. Ainsi, la main est devenue l'une des caractéristiques les plus déterminantes pour distinguer les cartes numérotées 1 et 2. Nous avons dû jeter l'ensemble de données original et en créer un nouveau en prenant soin de supprimer toute information non pertinente. Il est important de faire attention lors de l'enregistrement des images à l'ensemble de données. En effet, un filigrane ou toute autre caractéristique non pertinente capturée par inadvertance dans les clichés faussera le processus de reconnaissance.

Résumé et suites possibles

L'entraînement de notre propre réseau neuronal à l'aide d'Edge Impulse et du Jetson Nano est une procédure assez simple. L'étape suivante consiste à intégrer ce réseau dans une application pratique. Le Jetson Nano n'est pas la plateforme matérielle la mieux adaptée pour cette application ; certaines de ses ressources sont inutilisées, et il y a quelques problèmes logiciels qui empêchent la caméra de fonctionner directement avec le SDK Edge Impulse dans certaines applications.

Si vous voulez explorer l'IA et la reconnaissance d'images de manière simple, vous devriez tout de même essayer Edge Impulse. Il n'est pas nécessaire qu'il tourne sur un Jetson Nano. Un Raspberry Pi 4 fera tout aussi bien l'affaire.

Et ensuite ? Le Jetson Nano dispose d'une puissance suffisante pour reprendre ce qu'Edge Impulse a effectué ici. L'étape suivante pourrait consister à créer un réseau neuronal capable de reconnaître les cartes UNO et pouvant fonctionner de manière indépendante sur le Jetson Nano.

210318-B-01

PRODUITS

- Module Jetson Nano de Nvidia
  www.elektor.fr/nvidia-jetson-nano-developer-kit
- Kit AI JetBot v2.1 de SparkFun (sans kit de développement NVIDIA Jetson Nano inclus)
  www.elektor.fr/sparkfun-jetbot-ai-kit-v2-1-without-nvidia-jetson-nano-developer-kit
- Carte de développement ESP32-Cam-CH340
  www.elektor.fr/19333

Des questions ? Des commentaires ?

Envoyez un courriel à l'auteur (mathias.claussen@elektor.com) ou contactez Elektor (redaction@elektor.fr).

Contributeurs

Conception et texte : **Mathias Claußen**
Rédaction : **Jens Nickel**
Mise en page : **Harmen Heida**
Traduction : **Pascal Godart**

LIENS

[1] M. Claussen, « Traitement d'images avec le kit Jetson Nano de Nvidia (1ère partie) », Elektor, 09-10/2021 : www.elektormagazine.fr/210318-04
[2] M. Claussen, « Faire du café grâce au MAX78000 et à une pincée d'IA », site elektormagazine.fr, 04/2021 : www.elektormagazine.fr/articles/faire-du-cafe-grace-au-max78000-et-a-une-pincee-ia-2e-partie
[3] Edge Impulse : www.edgeimpulse.com
[4] A. Garrapa, « MCU Machine Learning With Edge Impulse », Elektor Industry, 02/2021 : www.elektormagazine.com/magazine/elektor-238/59631
[5] Edge Impulse – Nvidia Jetson Nano : https://docs.edgeimpulse.com/docs/nvidia-jetson-nano
[6] Téléchargement du réseau entraîné : www.elektormagazine.fr/210318-B-04
[7] Webinaire d'Elektor, « Meet the Engineers (Part 2): Get to Know the NVIDIA Jetson Nano Developper Kit », 05/2021 : https://youtu.be/KGazuosH0xw

JUMPSTARTER

Campagnes à venir

QUOI DE NEUF ?

Clemens Valens (Elektor Labs)

Elektor Jumpstarter est destiné à aider les innovateurs à transformer leurs projets électroniques en produits. Avec Elektor Jumpstarter et le soutien de notre communauté, vous pouvez réaliser votre rêve de mettre un produit électronique sur le marché !



Figure 1. Prototype du Bumblebee Automator.

Bumblebee Automator – carte de domotique

Ce projet a commencé par un thermostat avec Wi-Fi pour garder un œil sur la température d'une cave à vin. Il a été publié dans l'édition de septembre 2021 d'Elektor sous le titre « thermostat connecté à ESP32 » [1][2]. En préparant le projet pour la publication, il nous est rapidement apparu que ce montage était bien plus polyvalent qu'un simple thermostat. L'auteur l'a déjà utilisé comme contrôleur d'humidité et un autre garde l'eau de sa piscine propre. Chez Elektor, nous l'avons utilisé pour des mesures précises d'intensité lumineuse, et nous y avons connecté un lidar pour des mesures de distance et la détection d'intrus. Comme il s'agit plus d'une sorte d'ordinateur de domotique que d'un thermostat, il a été rebaptisé « Automator ».

L'Automator est basé sur un module ESP32 librement programmable, il dispose donc du Wi-Fi et du Bluetooth. Il est possible d'ajouter un petit écran OLED ; il dispose d'un relais et de quelques ports d'extension configurables. De nos jours, de nombreux capteurs et actionneurs se présentent sous la forme de petits modules dotés d'un bus série, SPI ou I2C, et peuvent être facilement connectés, ce qui permet toutes sortes d'applications. Bien sûr, il y a aussi quelques LED et un buzzer.

L'appareil est doté d'une alimentation à courant alternatif intégrée qui accepte tout type de tension de 100 à 240 V AC et de 50 à 60 Hz. Comme il est logé dans un boîtier étanche, il peut être installé à l'extérieur et dans d'autres environnements humides.

Côté logiciel, l'Automator est compatible avec ESPHome et donc avec Home Assistant, notre logiciel de domotique préféré, mais comme cette carte est basée sur un ESP32, elle s'intègre dans une multitude de systèmes de domotique. Bien entendu, elle peut aussi fonctionner de manière autonome.

La carte de base de l'Automator ne contient que des composants traversants, elle peut donc être facilement modifiée si nécessaire.

Détails de la campagne Jumpstarter

Prix unitaire : 59,95 €
Objectif : 100
Date de fin : 31 décembre 2021
Lien :
www.elektormagazine.fr/labs/automator



Figure 2. Prototype du CO_2 Guard.

CO_2 Guard

Le CO_2 Guard [3] sert à surveiller et évaluer la qualité de l'air intérieur (QAI) en fonction de la concentration de dioxyde de carbone (CO_2). Une concentration inférieure à 1.000 ppm (c'est-à-dire 0,1%) indique un renouvellement de l'air adéquat du point de vue hygiénique, dans des conditions normales, et peut contribuer à minimiser le risque d'infection par la COVID-19.

Le CO_2 Guard a une plage de mesure de 400 à 2.000 ppm et présente les résultats sous la forme d'un feu de circulation. Le vert est excellent, le jaune est sensible et le rouge est inacceptable. Les valeurs intermédiaires sont représentées par l'allumage simultané de deux LED : vert + jaune signifie acceptable et jaune + rouge signifie mauvais. Un buzzer émet une alarme sonore lorsque la concentration devient trop élevée et que des mesures doivent être prises (comme l'ouverture d'une fenêtre ou d'une porte).

Le CO_2 Guard utilise un capteur MH-Z19C du spécialiste des capteurs Winsen. Ce capteur utilise la détection infrarouge non dispersive (NDIR), une technique couramment utilisée qui mesure la quantité de lumière infrarouge absorbée par les molécules de CO_2 présentes dans l'air (ou dans un autre gaz).

Le capteur est lu par un module à microcontrôleur, basé sur l'ESP8266, avec Wi-Fi. Ce module pilote également les LED et le buzzer et lit le bouton-poussoir.

L'appareil peut se connecter à l'internet, et plus précisément à la plate-forme ThingSpeak, où les valeurs mesurées peuvent être enregistrées et affichées sous forme graphique. Cela permet de suivre les concentrations de CO_2 dans le temps et de les comparer aux données capturées par d'autres dispositifs. Le logiciel est écrit sous forme d'un croquis Arduino, ce qui le rend facile à modifier. Le CO_2 Guard est également compatible avec ESPHome, Home Assistant et d'autres plateformes de domotique.

Détails de la campagne Jumpstarter

Prix unitaire : 69,95 €
Objectif : 100
Date de fin : 31 décembre 2021
Lien : www.elektormagazine.fr/labs/co2-guard

210435-04

LIENS

[1] Thermostat connecté à ESP32, Elektor Labs : https://www.elektormagazine.com/labs/esp32-thermostat
[2] « Thermostat connecté à ESP32 », Elektor, 09-10/2021 : https://www.elektormagazine.fr/200497-04
[3] CO2 Guard, Elektor Labs : https://www.elektormagazine.fr/labs/co2-guard
[4] Informations complémentaires : https://www.elektormagazine.fr/news/elektor-jumpstarter-projet

traceur GPS à code **source ouvert**

Traccar cartographie les déplacements de véhicules, sans recours à un serveur tiers du nuage



Figure 1. Un traceur GSM-GPS à petit budget.

Mathias Claußen (Elektor)

Il existe de nombreux modules matériels pour tracer par GPS des véhicules et autres objets mobiles. Cependant, pour analyser, sauvegarder et afficher les données, vous devez généralement les envoyer à un serveur externe dans le nuage. Le logiciel à code source ouvert Traccar est une solution qui permet d'héberger le serveur localement, et ainsi de décider qui a accès à vos données. Traccar peut être exécuté sur un Raspberry Pi et fonctionne avec l'environnement d'automatisation Node-RED.

Si vous souhaitez suivre les déplacements d'un véhicule ou de tout autre objet mobile, vous avez le choix parmi une large gamme de dispositifs de suivi prêts à l'emploi. La plupart des traceurs actifs bon marché utilisent le réseau de téléphonie mobile 2G ou 3G pour transférer les informations de position sous forme de messages. Nous avons déjà passé en revue des traceurs tels que le Fortebit Polaris 3G Kit+ [1]. Pour cet article, nous avons commandé deux modules du commerce pour évaluer leur fonctionnement avec le logiciel Traccar. Le premier a été acheté sur l'une des plus grandes plateformes de vente par correspondance en ligne pour moins de 20 € (**fig. 1**). Le dispositif ressemble à un relais automobile de 12 V et peut être installé de manière assez discrète dans un compartiment moteur. Certains traceurs transfèrent les données de suivi en utilisant une autre technologie de réseau à longue portée, telle que LoRaWAN, comme nous l'avons vu avec le traceur LoRa d'Elektor [2].

La plupart des traceurs commerciaux offrent à l'utilisateur une solution basée sur le nuage pour stocker et afficher les données de position. C'est un moyen pratique pour afficher les données de véhicules et gérer l'historique des mouvements et événements pour de nombreux utilisateurs. Toutefois, avec ce type de service, vous devez tenir compte de l'endroit où sont conservées vos données. Même si le prestataire de services vous assure que les données sont traitées conformément au règlement général sur la protection des données (RGPD), ce n'est pas forcément le cas. Il serait préférable que vous soyez sûr que le prestataire respecte les termes de la loi et ne mette pas vos données personnelles à la disposition d'un tiers non autorisé. Traccar vous offre la possibilité de contourner les services du nuage en hébergeant le service sur votre propre serveur local.

Installation de Traccar

Le logiciel Traccar [3] s'exécute sur des ordinateurs x86 fonctionnant sous Windows et Linux, et sur des systèmes Linux fonctionnant sur ARM, comme le Raspberry Pi. Il s'agit d'une solution à code source ouvert conçue pour tracer des objets mobiles. La **figure 2** montre la représentation cartographique fournie par Traccar indiquant

l'emplacement du traceur. Le logiciel affiche le parcours de tout objet mobile équipé d'un traceur.

L'installation de Traccar sur un Raspberry Pi est simple. Vous aurez besoin d'une carte fonctionnant sous Raspberry Pi OS. Le modèle 3B(+) fonctionnera pour les premiers essais, mais un RPi 4 est recommandé pour une utilisation plus sérieuse. Si vous préférez ne pas utiliser un Raspberry Pi, Traccar fonctionne également sur d'autres systèmes.

On peut télécharger une variante ARM spéciale du serveur pour le Raspberry Pi à partir de la page d'accueil de Traccar en utilisant le navigateur web du RPi. Les fichiers peuvent être téléchargés à l'aide de `wget` en ligne de commande. La page d'accueil de Traccar comporte un lien vers les dernières versions du logiciel. Le fichier téléchargé devra être décompressé avec l'outil graphique de bureau ou des instructions en ligne de commande de Raspberry Pi OS. Si le téléchargement est enregistré sous forme de fichier zippé, on peut utiliser la commande `unzip` depuis un terminal.

Les fichiers décompressés sont présentés à la **figure 3**. Il faut maintenant un terminal car le programme d'installation a besoin des privilèges de l'utilisateur *root* pour s'exécuter. Il suffit de se rendre à l'endroit où se trouvent les fichiers décompressés et d'exécuter le programme d'installation à l'aide de la commande `sudo ./traccar.run`.

Une fois l'installation terminée, on peut configurer Traccar en tant que service lancé automatiquement au démarrage du système. Pour cela, entrez `sudo systemctl enable traccar.service` dans le terminal. Pour éviter de redémarrer, on peut lancer Traccar immédiatement en utilisant `service traccar start`. Une fois le service lancé, l'interface web est disponible à l'adresse *http://localhost:8082/*.

Si le navigateur du RPi est trop lent, vous pouvez accéder à l'interface web depuis un autre ordinateur via le réseau. Par défaut, le nom d'utilisateur est *admin* et le mot de passe *admin*. N'oubliez pas de changer le

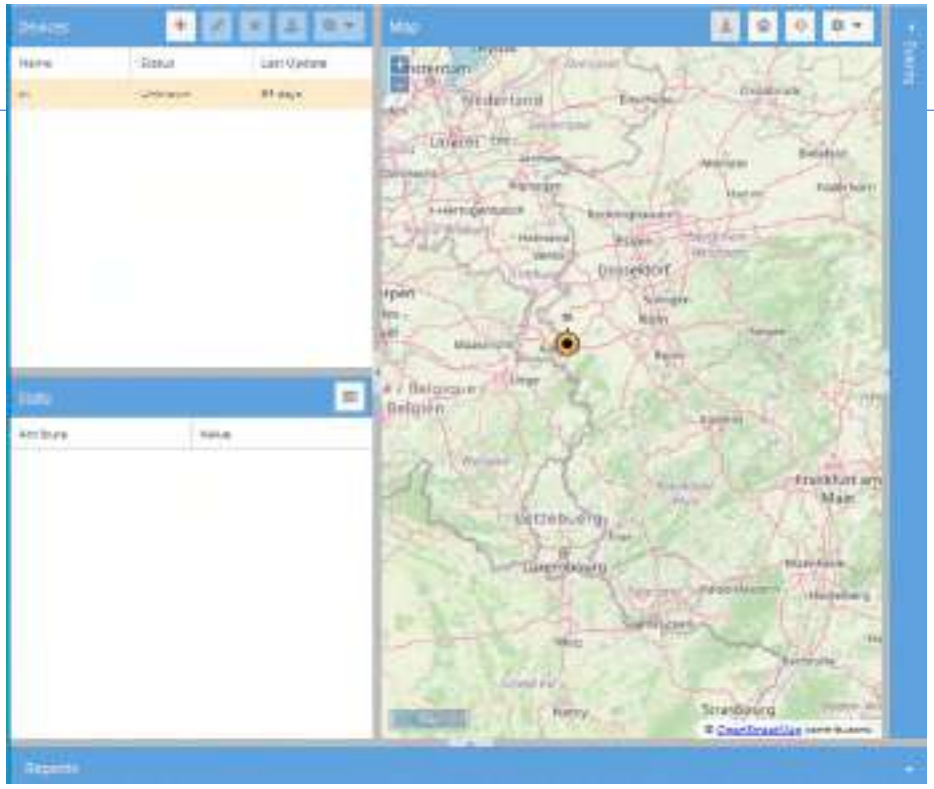


Figure 2. L'écran de Traccar.

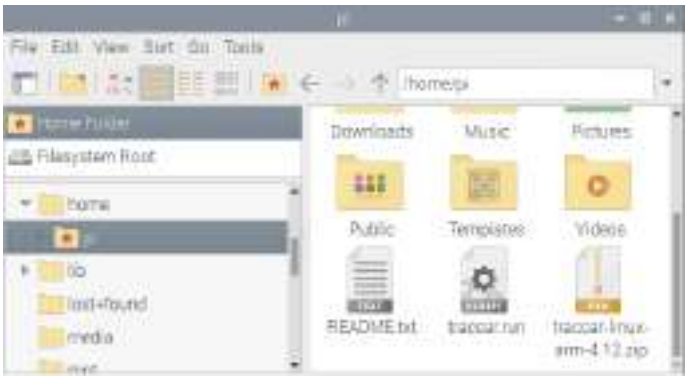


Figure 3. Fichiers extraits pour l'installation de Traccar.

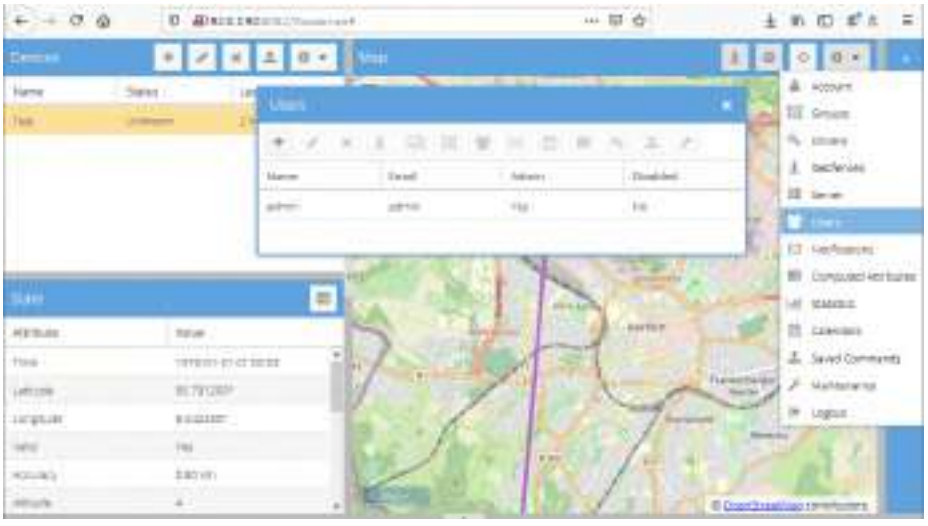


Figure 4. Configuration de l'utilisateur dans le menu Traccar.



Figure 5. L'intérieur du traceur GSM-GPS bon marché.



Figure 6. Transport des données dans le réseau de téléphonie mobile.



Figure 7. Prises d'antenne du traceur GSM-GPS TK103.



Figure 8. Connecteur véhicule du traceur GSM-GPS TK103.

mot de passe par défaut par quelque chose de plus sécurisé. La **figure 4** montre les options de menu pour la configuration de l'utilisateur.

Maintenant que Traccar fonctionne, des traceurs peuvent être ajoutés au système. Examinons de plus près les traceurs basés sur le GSM et leurs méthodes de transfert des données.

Traceur GSM-GPS

La majorité des traceurs GPS bon marché sont équipés d'un modem de réseau cellulaire 2G. Certains de ces traceurs coûtent moins de 20 € (**fig. 5**) et sont suffisamment petits pour être facilement dissimulés dans un véhicule. Les traceurs plus chers utilisent parfois un modem 3G, mais, dans l'idéal, il est préférable d'investir dans un modem 4G ou 5G pour garantir une bonne pérennité.

En Suisse, les autorités compétentes ont mis hors service le réseau 2G à la fin de 2020, libérant ainsi la bande de fréquences pour d'autres usages. La fin de la 2G est proche pour le reste de l'Europe également, les rumeurs indiquent un arrêt d'ici la fin de 2025. Par conséquent, les traceurs GSM-GPS les moins chers cesseront de fonctionner.

La fonction de ces traceurs est simple. Le traceur peut être configuré pour envoyer un message d'état à un intervalle de temps défini ou après un événement tel que la mise en marche ou l'arrêt du moteur. L'état peut être envoyé sous la forme d'un SMS ou d'un paquet de données. La **figure 6** montre la méthode de transport des données.

Nous nous concentrons ici sur les solutions envoyant des données par paquets de données plutôt que par SMS. La plupart des traceurs utilisent des paquets UDP. Contrairement aux paquets TCP, les paquets UDP ne nécessitent pas d'accusé de réception de la part du destinataire, ce qui réduit au minimum la quantité de données et l'énergie nécessaire pour transmettre les informations d'état.

La configuration du traceur dépend du fabricant et on ne peut pas qualifier ce processus de convivial selon les standards actuels. La plupart des appareils bon marché utilisent la messagerie SMS pour la configuration, comme le détecteur de panne de courant Elektor [4]. Le forum Traccar [5] fournit une multitude d'informations et de conseils sur le fonctionnement et la configuration de toute une série de modèles de traceurs différents.

Le serveur RPi se trouve derrière le pare-feu du routeur, il faut donc ouvrir un port pour permettre aux appareils externes (le traceur) d'accéder au serveur. La page Traccar [6] indique le numéro de port nécessaire. La procédure pour ouvrir le port vers le serveur dépend du routeur utilisé. De plus, l'adresse IP externe du routeur doit pouvoir être résolue par un service DNS. Si vous n'avez pas d'adresse IP fixe, vous pouvez utiliser un fournisseur de service DNS dynamique tel que dynv6, No-IP, ou similaire.

Configuration de test du traceur GSM-GPS

Nous avons d'abord choisi le traceur GPS LK720 (figures 1 et 5) pour tester le système Traccar. Ce modèle fonctionne avec une tension

d'alimentation comprise entre 6 V et 38 V CC et consomme environ 150 mA sous 12 V. La configuration a été tentée avec l'aide du fil de discussion [7] du forum Traccar.

Malheureusement, il s'est avéré qu'il ne répondait pas du tout aux commandes SMS. Il a donc été impossible de configurer et de mettre en service l'appareil. Il a été difficile de déterminer si l'appareil provenait d'un mauvais lot ou si le problème venait de la carte SIM. Sur la base de cette expérience, ce modèle est à éviter si vous souhaitez un traceur GPS fiable.

Comme alternative, nous avons commandé un traceur GPS GSM de type TK103 (**figures 7** et **8**) qui coûte environ 50 € (y compris les frais de port dans l'UE). Bien qu'il ne soit pas aussi compact que le traceur LK720, il est directement pris en charge par Traccar, et par conséquent son protocole est traité. En utilisant ce dispositif, on doit ouvrir le port 5001 dans le routeur pour permettre aux données UDP d'être transmises à l'ordinateur sur lequel le serveur Traccar fonctionne.

Le traceur est configuré par des commandes SMS. Les commandes à rechercher dans le manuel sont UP, DNS, APN et GPRS. Pour le GPRS, il existe deux variantes de commandes. La première active le mode GPRS, la seconde configure le protocole (UDP ou TCP). La meilleure approche ici est de configurer le traceur en mode UDP, car c'est ce qui a permis la transmission de données pendant nos tests. Un intervalle de temps fixe de 11 s a été configuré pour envoyer les informations de position (que le véhicule soit en mouvement ou non).

Si tout est correctement configuré, les données devraient maintenant arriver sur le serveur Traccar, mais elles ne seront pas encore traitées ou affichées. Tout d'abord, le traceur doit être ajouté à l'interface web de Traccar. Pour cela, il faut cliquer sur le bouton *Add* (**fig. 9**) et saisir un nom pour notre dispositif (**fig. 10**). Après avoir choisi un nom significatif, il faut entrer le numéro IMEI du modem TK103. Ce numéro peut être obtenu par messagerie SMS comme décrit dans le manuel du traceur. Une fois ces données saisies, le nom du traceur, sur fond vert, apparaît dans la liste des appareils (**fig. 11**). À partir de là, la position actuelle du traceur devrait s'afficher sur la carte.

Traccar enregistre maintenant les mouvements et les événements qui affectent le traceur. La **figure 12** montre le traceur dans une boîte transportable, prêt pour un petit parcours de test. On voit le résultat sur la **figure 13**. En plus des informations de position, tous les événements qui se produisent sont également signalés par le traceur et traités par Traccar (**fig. 14**).

C'est tout ce dont vous avez besoin pour un traceur rudimentaire par GSM. Toutefois, si vous souhaitez afficher davantage d'informations, comme les données OBD du véhicule en temps réel, vous devrez réaliser votre propre traceur GPS et envoyer les informations à afficher via Traccar.

Autres types de traceurs et Node-RED

Traccar est capable de prendre en charge plus que les seuls traceurs GSM. Certains dispositifs communiquent en utilisant Sigfox ou LoRaWAN pour transférer leurs informations de position, au lieu

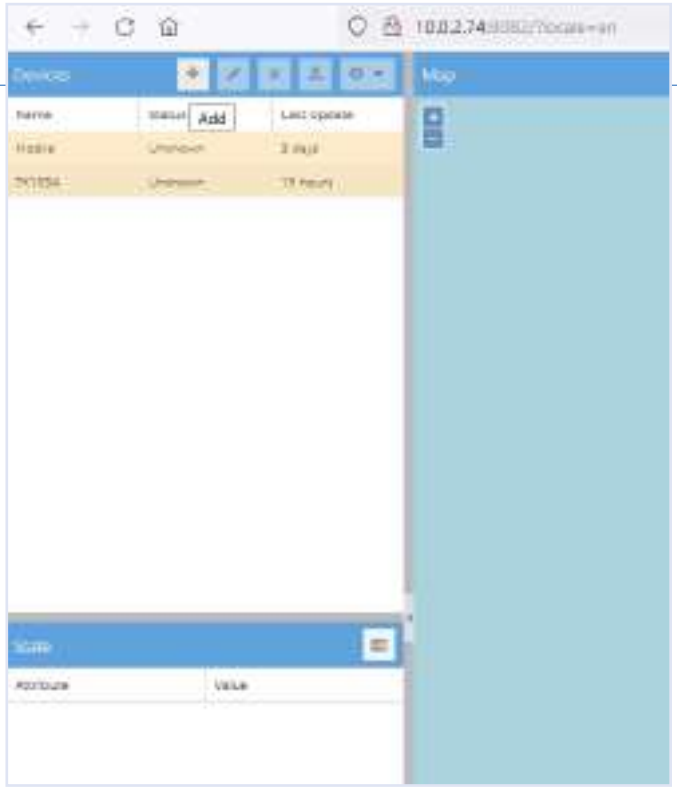


Figure 9. Ajout d'un appareil dans Traccar.



Figure 10. Saisie du nom et de l'identifiant du traceur.

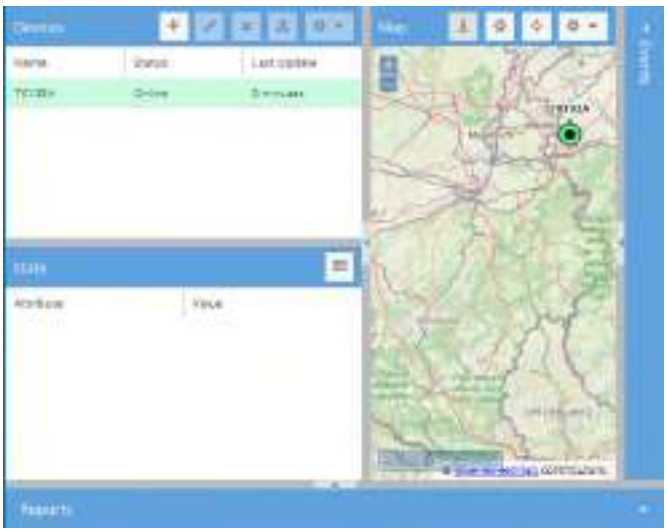


Figure 11. Le traceur est en ligne.

du réseau cellulaire. Notez toutefois qu'une passerelle est nécessaire pour relayer les informations lorsque le traceur ne peut pas envoyer ses données directement au serveur (comme avec Sigfox ou LoRaWAN). Les passerelles transforment les données dans un format que Traccar peut traiter, avant de les lui transmettre. OsmAnd, une application de cartographie et de navigation hors ligne, utilise un protocole que Traccar comprend et qui est facile à mettre en œuvre. Le protocole OsmAnd formate les données du traceur en http POST. Cette conversion s'effectue facilement à l'aide du puissant environnement d'automatisation Node-RED.

Pour installer Node-RED, consultez les instructions d'installation [8] pour le Raspberry Pi. Après l'installation, Node-RED peut être lancé depuis un terminal en utilisant `node-red-start`. Si Traccar et Node-RED fonctionnent maintenant en tandem, les données LoRaWAN peuvent être modifiées et traitées pour Traccar. La séquence est illustrée à la **figure 15**.

Un bon exemple de dispositif qui n'est pas directement pris en charge est le traceur GPS LoRa d'Elektor [9]. Il existe déjà un flux Node-RED pour le traceur LoRa qui peut afficher les informations de position dans OpenStreetMap. Cependant, cette variante présente certaines limitations pour les dispositifs de suivi et n'affiche pas les itinéraires. Nous pouvons maintenant utiliser Traccar avec Node-RED pour analyser et afficher les données des traceurs LoRaWAN. Ce n'est pas tout, avec Node-RED le système aura une connexion de données universelle. Cela vous permet de suivre l'emplacement de ballons météo et de nombreux autres appareils.

L'article à télécharger à l'adresse [10] contient un exemple de flux pour Node-RED qui envoie des données à Traccar. Si vous souhaitez en savoir plus sur Node-RED et ses capacités sur la plateforme Raspberry Pi, nous vous recommandons vivement de consulter le livre *Programming with Node-RED* [11] dans l'e-choppe d'Elektor.

Vos données, votre serveur

Avoir le contrôle de ses propres données, plutôt que de les confier à d'autres organisations, est le choix de beaucoup



Figure 12. Traceur et batterie installés et prêts au déploiement sur le terrain.



Figure 13. L'itinéraire de test tel que cartographié par Traccar.

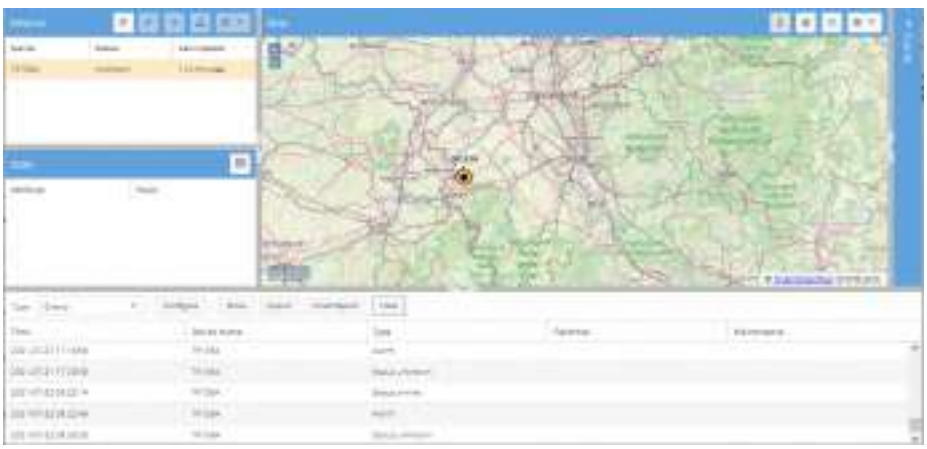


Figure 14. Détails de l'itinéraire de test dans le journal de Traccar.



Figure 15. Exécuter Traccar avec Node-RED permet de recevoir les données des traceurs GPS LoRaWAN.

aujourd'hui. Trop d'opérateurs de sites web stockant des données personnelles dans le nuage ont récemment découvert des violations de données [12] et appris que les données pouvaient disparaître en fumée [13].

Le fait d'avoir accès à toutes vos données vous offre une certaine souplesse pour les traiter ultérieurement, pour exploiter les capacités de suivi et pour améliorer les fonctions du système. Les données peuvent également être importées et traitées sans grand effort dans le projet *ResQ Search and Rescue Tools* sur le site du labo d'Elektor [14]. Cela permettrait de créer des cartes de localisation, affichant la position d'une personne disparue dans un endroit éloigné sans couverture cellulaire, si les signaux des balises Wi-Fi ou Bluetooth du téléphone peuvent encore être détectés.

Les faibles besoins énergétiques du RPi le rendent très adapté à une utilisation mobile, tandis que les cartes SD à auto-chiffrement [15] peuvent être utilisées pour protéger les données collectées contre tout accès non autorisé si le RPi risque d'être volé.

200532-04

Contributeurs

Texte : **Mathias Claußen**
Rédaction : **Jens Nickel, Stuart Cording**
Mise en page : **Harmen Heida**
Traduction : **Denis Lafourcade**

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (mathias.claussen@elektor.com) ou contactez Elektor (redaction@elektor.fr).

PRODUITS

- **Raspberry Pi 4 B (4 GB RAM) (SKU 18964)**
  www.elektor.fr/18964
- **Boîtier acrylique JOY-iT à deux ventilateurs pour le Raspberry Pi 4 (SKU 19053)**
  www.elektor.fr/19053

LIENS

[1] « Banc d'essai - traceur GPS Polaris 3g kit+ de Fortebit », Elektor, 09/2020 : www.elektormagazine.fr/news/fr-review-fortebit-polaris-3g-kit
[2] « LoRa GPS Tracker », Elektor Labs, 02/2020 : www.elektormagazine.fr/labs/lora-gps-tracker
[3] Page d'accueil de Traccar : www.traccar.org
[4] L. Lemmens et M. Claussen, « détection de coupure de secteur avec alarme SMS », Elektor, 09-10/2018 : www.elektormagazine.fr/180344-02
[5] Forum Traccar : http://www.traccar.org/forums
[6] Ports pour divers traceurs GPS : http://www.traccar.org/devices
[7] Configuration du CJ720, forum Traccar : https://bit.ly/3isroGz
[8] Installation de Node-RED sur Raspberry Pi : https://bit.ly/2WR8oJd
[9] M. Claussen, « balise GPS LoRa », Elektor, 11-12/2020 : www.elektormagazine.fr/200096-04
[10] Téléchargement du flux Node-RED : https://bit.ly/3yHYPKW
[11] D. Ibrahim, « Programming with Node-RED », Elektor : www.elektor.fr/19224
[12] « Facebook faces mass legal action over data leak », BBC News, 04/2021 : https://bbc.in/3jAs6Rs
[13] M. Rosemain, R. Satter, « Millions of websites offline after fire at French cloud services firm », Reuters, 03/2021 : https://reut.rs/3fD7p5T
[14] MKME Lab, « ResQ Search and Rescue Tools », Elektor Labs, 01/2021 : www.elektormagazine.fr/labs/resq-search-and-rescue-tools
[15] M. Claussen, « Solution de démarrage sécurisé pour Raspberry Pi », Elektor, 02/2021 : www.elektormagazine.fr/news/translation-pascal-secure-boot-solution-for-raspberry-pi

testeur multifonction LCR-T7 de Joy-IT



Figure 1. L'afficheur montrant les résultats du test.

Test de semi-conducteurs passifs, discrets et de télécommandes IR

Luc Lemmens (Elektor)

Vous êtes à la recherche d'un testeur multifonctions simple et utile ? Le Joy-IT LCR-T7 pourrait bien être la solution dont vous avez besoin.

Les testeurs multifonctions sont toujours utiles pour les composants discrets, qu'ils soient passifs ou actifs. Parfois, les valeurs ou les numéros de type sont difficiles à lire et, dans le cas de semi-conducteurs discrets avec des numéros exotiques, vous pouvez vouloir vérifier de quel type de composant il s'agit. Et même si vous pouvez identifier visuellement le composant, vous voudrez peut-être savoir s'il fonctionne toujours et s'il est, dans une certaine mesure, conforme à ses spécifications d'origine. C'est là qu'un testeur comme le Joy-IT LCR-T7 s'avère utile : il permet de vérifier rapidement à quel composant nous avons affaire, ou du moins de voir s'il s'agit du composant auquel nous nous attendions. En d'autres termes, avec cet instrument, il est facile de tester des composants discrets, d'identifier le type dont il s'agit et dans le cas des semi-conducteurs de déterminer le brochage correct. Comme son nom l'indique, il s'agit d'un testeur, ne vous attendez donc pas à obtenir des mesures très précises avec. Vous ne devriez pas exiger cela d'un appareil de mesure qui coûte moins de trente euros.
L'un des avantages du LCR-T7 est qu'il est très facile à utiliser : il suffit de connecter le composant à tester au connecteur ZIF sur le panneau avant, avec ou sans les fils et les clips fournis avec le testeur et d'appuyer sur le bouton *Start*. Le tour est joué. L'appareil reconnaît automatiquement le type de composant connecté et affiche ses principaux paramètres et (le cas échéant) son brochage sur l'écran LCD.

Composants pris en charge

La liste des composants qui peuvent être testés avec le Joy-IT LCR-T7 (voir encadré) est assez impressionnante pour un appareil aussi petit et abordable. Les valeurs qui sont mesurées et affichées dépendent bien sûr du type de composant testé. La **figure 1** montre quelques résultats pour différents types de composants, le manuel [1] fournit un aperçu complet de tous les composants qui peuvent être testés.
Bien que les transistors Darlington ne soient pas répertoriés, il semble que le LCR-T7 donne néanmoins des informations sur ces composants. Ils peuvent être identifiés par une tension V_{be} élevée (> 0,7 V), et l'identification du brochage est correcte pour les Darlington que j'ai testés. Cependant, ne vous fiez pas aux autres mesures qui sont affichées pour ce type de transistor.

Pour commencer

Après le déballage, la première chose à faire est de charger la batterie au lithium à l'aide du câble micro-USB fourni avec le LCR-T7. La LED bicolore située à côté du connecteur USB s'allume en rouge pendant la charge, et en vert lorsqu'elle est complètement chargée. Le testeur peut également être utilisé s'il est alimenté par le port USB.
Effectuez l'étalonnage automatique du LCR-T7 en court-circuitant les bornes (voir **fig. 2**) et en appuyant sur le bouton *Start*.
Lorsque l'étalonnage est terminé à environ 22%, l'instrument vous invite à retirer les pontages ; lorsque ceux-ci sont déconnectés, l'étalonnage se poursuit et se termine automatiquement. Le manuel ne mentionne

pas quand ou bien à quelle fréquence le testeur doit être recalibré, mais je pense qu'il n'y a pas de mal à le faire de temps en temps, surtout si vous n'êtes pas sûr des résultats d'un test.
Pour les premiers tests, un condensateur électrolytique et une LED sont fournis. À mon avis, il aurait été plus judicieux d'ajouter quelques semi-conducteurs discrets à trois pattes pour démontrer l'identification automatique du type et des broches.

Connexion des composants

Le connecteur ZIF pour la connexion des objets sous test (**fig. 3**) est une excellente solution, car une simple prise normale, comme on en trouve sur de nombreux testeurs similaires, s'use en un rien de temps. Trois cordons de test avec pinces sont fournis pour connecter des composants avec des broches plus grosses afin d'éviter d'endommager les contacts du support ZIF. Toute la rangée supérieure et les quatre connexions les plus à droite de la rangée inférieure sont utilisées pour la plupart des tests de composants, avec « 1 », « 2 » et « 3 » marquant l'emplacement des broches de l'objet sous test. Les trois broches restantes à gauche de la rangée inférieure (marquées « K » et 2 × « A ») servent à tester les diodes Zener avec des tensions allant jusqu'à 30 V.

Détecteur IR

Le LCR-T7 dispose également d'un détecteur IR intégré qui peut décoder et afficher les codes de périphérique et de données des télécommandes utilisant le protocole NEC. Pour les appareils utilisant d'autres protocoles IR, le testeur affiche un point rouge dans le coin de l'écran lorsqu'un signal IR est détecté, de sorte que vous pouvez au moins vérifier si votre télécommande fonctionne ou non.

Simple, mais utile

Dans toute sa simplicité, le LCR-T7 est certainement utile. Je préfère un multimètre normal pour mesurer des grandeurs comme la tension d'une batterie ou une résistance, surtout parce que cet instrument est toujours sur mon établi. Mais pour d'autres composants, notamment pour les semi-conducteurs, c'est vraiment génial d'identifier rapidement le type et le brochage, et d'obtenir les principaux paramètres sur l'écran en connectant le composant inconnu et en appuyant simplement sur un bouton. Et ce que nous apprécions tous : le prix est également intéressant !

210365-04 – VF : Maxime Clemens

PRODUIT

- **Testeur multifonction LCR-T7 de Joy-IT**
  www.elektor.fr/19709

Caractéristiques et spécifications

CARACTÉRISTIQUES PRINCIPALES	
Composants mesurables	Résistance, condensateur, inductance, thyristor, TRIAC, (double) diode, diode Zener, transistor à effet de champ, transistor bipolaire, télécommande infrarouge.
Protocole IR pris en charge	NEC (utilisé par de nombreux fabricants)
Type d'affichage	1.8" TFT LCD (160 × 128 pixels)
Caractéristiques spéciales	Calibrage automatique, opération à une touche
Batterie intégrée	Lithium-ion rechargeable, 3,7 V, 350 mAh

GAMMES DE MESURE	
Capacité	25 pF - 100 mF
Résistance	0,01 Ω - 50 MΩ
Inductance	0,01 mH - 20 H
Batterie	0,1 V – 4,5 V
Tension de la diode Zener	0,01 V - 30 V
Diode	UF < 4,5 V

LIEN

[1] SIMAC Electronics (Joy-It), « Multi-Functional Tester », 2021 : https://bit.ly/3cY0Ppc



Figure 2. Calibrage avec une prise de test court-circuitée.

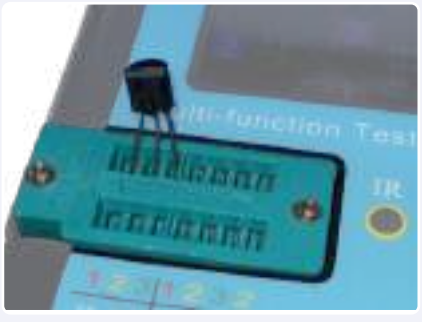


Figure 3. Test des composants dans le connecteur ZIF ou avec les clips de test.



elektor TV
vidéo du projet Elektor sur YouTube

synthétiseur de bruit

Du bruit à la musique avec le PRBSynth1

Raymond Schouten (Pays-Bas)



Le bruit est souvent utilisé en musique électronique et dans les effets sonores. Le synthétiseur décrit ici invite le lecteur à explorer les sons à base de bruit. Le matériel est assez simple, un Arduino Pro Mini étendu avec un CN/A, un contrôle de volume et un circuit de panoramique stéréo, ainsi qu'une entrée et une sortie MIDI. Le logiciel réalise la plupart des fonctions de génération du son sur quatre types de filtres et un puissant *LFO* (oscillateur BF) qui peut aussi contrôler le son par des motifs rythmiques. L'espace d'expérimentation du matériel et du logiciel est vaste – amusez-vous !



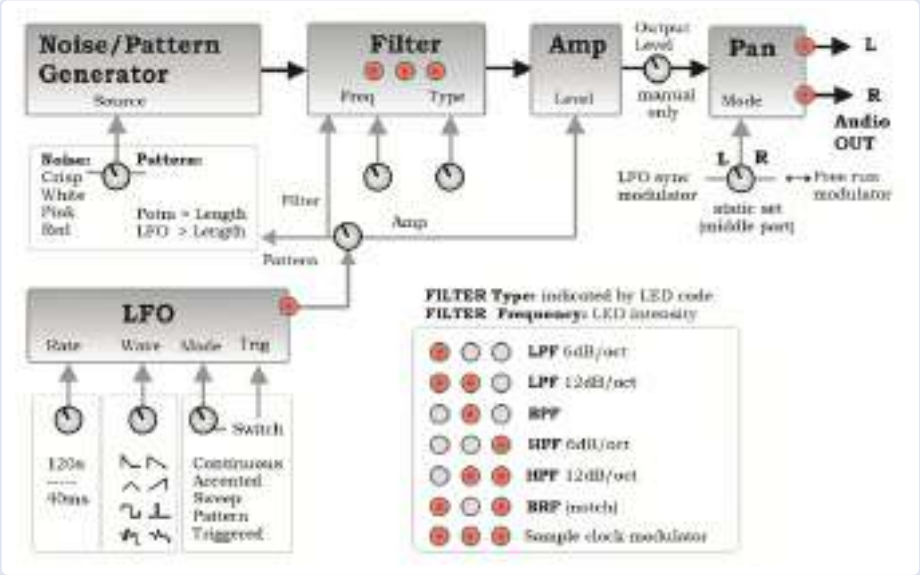


Figure 1. Schéma-bloc du PRBSynth1. Chaque bloc possède une ou plusieurs commandes sous forme de potentiomètres ou de paramètres MIDI CC. Les points orange représentent des LED.

Le PRBSynth1 est un synthétiseur musical destiné à produire, façonner et moduler des sons à base de bruit. Si de nombreux sons de synthétiseurs classiques peuvent être créés, l'unité permet aussi d'explorer d'autres espaces sonores.

« Cherchez votre propre voie »

Si cet article décrit le synthétiseur, ce n'est pas son unique propos : il vous invite à l'expérimentation, soit en l'adaptant, soit en utilisant certaines des idées présentées ici (voir les encadrés). La réalisation peut débuter par une version réduite sur une plaque d'essai sans le MIDI et le circuit de panoramique avec son potentiomètre et ses LED (exemple disponible). Un Arduino Uno peut être utilisé bien que ses six entrées analogiques limitent à six le nombre de potentiomètres utilisables. Une autre option est de réaliser une version MIDI seulement, sans potentiomètres (il faut désactiver la partie du programme qui les lit).

Le moteur sonore

L'architecture du PRBSynth1 est assez classique (voir **figure 1**). Un son est réalisé en jouant sur huit fonctions puissantes. Elles sont commandées soit par huit potentiomètres, soit par MIDI. Il n'y a pas d'affichage. Le synthétiseur répond aussi aux messages MIDI *Note-On* et *Note-Off* avec des données de vélocité. Un 9e potentiomètre règle le niveau de sortie et son interrupteur intégré connecte la batterie interne. Quelques LED renseignent sur le mode de fonctionnement du synthétiseur.

Générateur de bruit

Dans un synthétiseur classique, un oscillateur produit le son sous forme d'ondes sinusoïdales, en dents scie ou carrées. Ici, l'oscillateur est remplacé par un générateur de bruit qui en produit quatre types : blanc, rose, rouge et popcorn.

Le spectre du bruit blanc est horizontal, c.-à-d. que comme pour la lumière blanche, chaque fréquence a la même intensité. C'est pourquoi on l'appelle bruit blanc. Il ressemble au son d'une radio entre deux stations. En accentuant certaines parties du spectre, on obtient d'autres *couleurs* de bruit.

Le bruit rose en 1/f a un spectre qui décroit de 10 dB quand la fréquence croît d'un facteur 10, pour le bruit rouge ou brownien, c'est 20 dB. Le bruit de pop-corn est un crépitement dû à l'écrêtage de la sortie du générateur de bruit. Il en existe deux sortes.

Le bruit est produit numériquement par logiciel à l'aide d'un registre à décalage et d'une rétroaction. (Voir l'encadré **Pour faire du bruit, lancez des pièces en l'air !**) En plaçant judicieusement les rétroactions, le flux binaire de sortie devient quasi aléatoire. Un tel flux de bits est une séquence binaire pseudoaléatoire, ou *PRBS* (d'où le nom du synthé : PRBSynth). Elle est pseudoaléatoire, car le flux de bits de sortie se répète périodiquement. Lorsque le registre à décalage est assez long, la période du flux est très longue et il peut être considéré comme du bruit. Donc, techniquement parlant, un tel générateur de bruit est un oscillateur à forme d'onde très complexe et à très longue période.

Le PRBSynth1 permet aussi de contrôler la longueur de la séquence aléatoire pour passer progressivement du bruit à des motifs répétitifs au son aigu. C'est le mode *pattern* (motif). La hauteur de ces motifs est calée sur des notes de musique et peut être jouée sur un clavier MIDI.

Filtres

La sortie du générateur de bruit est connectée à un filtre. C'est un filtre multimode créé par logiciel (voir l'encadré **Un logiciel de filtres presque trivial**) avec modes passe-bas, passe-bande, passe-haut et réjection de bande (*notch*). Les filtres passe-bas/haut sont déclinés en deux pentes : 6 dB/octave ou 1er ordre et 12 dB/octave ou 2e ordre. On ne peut utiliser qu'un seul filtre à la fois.

Un potentiomètre règle la fréquence de coupure ; la résonance (ou facteur de qualité, Q) est fixe, mais divers préréglages de Q existent : trois pour le filtre passe-bas à 12 dB, cinq pour le filtre passe-bande ; le filtre passe-haut à 12 dB et le filtre coupe-bande en ont chacun deux.

Cela donne un total de 14 préréglages. Il en existe un 15e : c'est un filtre passe-bas à fréquence fixe, avec une fréquence d'échantillonnage modulée.

Trois LED codent la variante de filtre activée. L'intensité des LED reflète la fréquence de coupure.

Amplificateur

La sortie du filtre est suivie d'un amplificateur. Le lecteur attentif aura remarqué l'absence de générateurs d'enveloppe (*ADSR*) typiques des synthés (analogiques). Le PRBSynth1 n'en possède pas, mais, réglé en mode « déclenché », le LFO se transforme en générateur d'enveloppe (voir ci-après). En appuyant sur le clavier ou le commutateur de déclenchement, le LFO exécute un seul cycle. Les potentiomètres de forme d'onde et de vitesse définissent respectivement la forme de l'enveloppe et la durée. L'enveloppe peut contrôler la fréquence de coupure du filtre ou l'amplitude du signal. En outre, l'amplificateur transmet le signal si une touche (MIDI) est enfoncée, de sorte que le son est coupé en relâchant la touche.

Le volume du signal numérique est contrôlable par MIDI et bien que cela agisse sur le volume global, ce n'est pas recommandé. Pour une

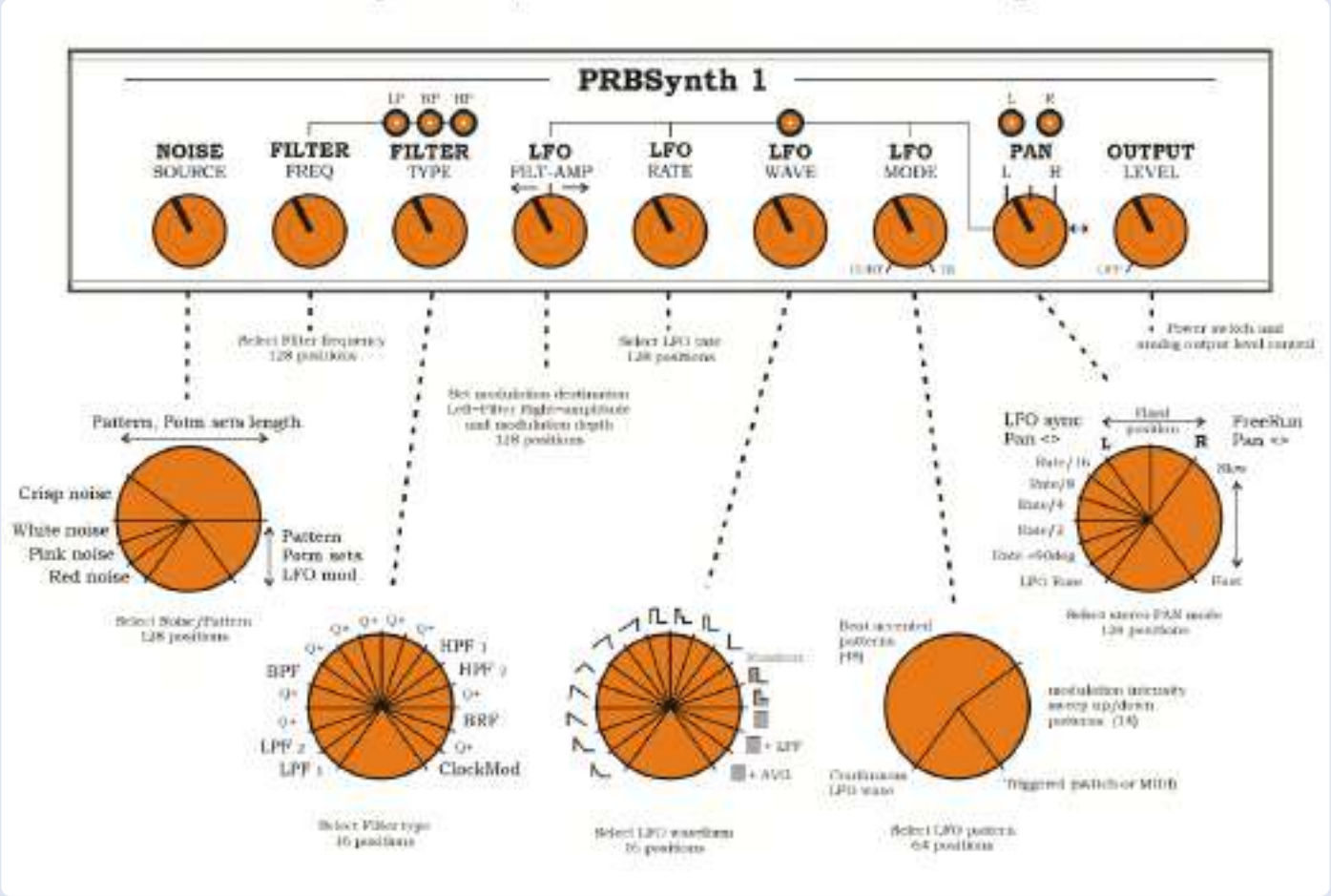


Figure 2. Synthèse des fonctions des potentiomètres du synthétiseur. Certains des boutons agissent en partie comme commutateur multiposition et en partie comme potentiomètre. Les échelles sont les mêmes que pour les commandes MIDI CC équivalentes. Un dessin plus grand est disponible sur la page web du labo [1].

qualité optimale du signal, il faut s'efforcer d'avoir un signal numérique de volume maximal. Un potentiomètre placé après le convertisseur numérique-analogique (CN/A, voir ci-après) règle le niveau de sortie analogique du synthétiseur.

Panoramique stéréo

Bien que le signal de sortie du synthétiseur soit monophonique, le panoramique le place où l'on veut dans l'image stéréo. La sortie du synthétiseur est donc stéréo.

Un seul potentiomètre permet de positionner le signal manuellement de gauche à droite et de donner le contrôle au LFO ou à un modulateur indépendant ayant sa propre horloge. Deux LED montrent ce qui se passe.

Oscillateur basse fréquence

Un oscillateur BF (*LFO*) ajoute une certaine variation aux sons synthétiques statiques. Par ex. si le LFO balaie lentement la fréquence de coupure du filtre, on crée des sons évoquant le vent, la mer ou la pluie. Le LFO peut également moduler l'amplitude du signal, la longueur du registre à décalage qui génère le bruit et la position stéréo du signal de sortie. La période du LFO est réglable entre 0,04 s (25 Hz) et 120 s. Outre le contrôle de période, le LFO dispose de 16 formes d'onde, incluant des dents de scie variées, un triangle, quelques impulsions et des formes aléatoires. La vitesse du LFO et la forme d'onde sont indiquées par une seule LED.

Un potentiomètre pilote le LFO selon quatre modes de fonctionnement. Avec le potentiomètre à fond à gauche, le LFO est toujours actif, comme un LFO classique. C'est le mode 1. Le 2e ajoute des accents à partir d'une table de motifs. Le potentiomètre de mode LFO permet de choisir parmi 48 motifs rythmiques, la plupart d'une longueur de huit pas et 12 d'une longueur de six.

Le 3e mode offre 16 motifs de balayage de la profondeur du LFO dont trois d'une longueur de 12 mesures. Tous les autres font 16 mesures. Le bouton de mode LFO sélectionne le type de balayage.

Si le bouton de mode est à fond à droite, le 4e mode est activé. Ce mode lance un cycle LFO unique quand on appuie sur le bouton de déclenchement ou sur une touche du clavier MIDI. Il applique manuellement une modulation de type enveloppe.

Interface utilisateur

Avec seulement huit potentiomètres (sans compter celui du volume de sortie), l'utilisateur accède à de nombreux paramètres sonores. La division en secteurs de ces potentiomètres permet d'en tirer le maximum. Seuls les boutons de fréquence de coupure du filtre et de vitesse du LFO agissent comme des potentiomètres. Les autres, dont certains ont un grand nombre de secteurs, tiennent plus du commutateur multiposition. C'est pourquoi une surface de contrôle MIDI est utile. En effet, l'affichage numérique des valeurs CC (contrôle continu) y est fréquent et permet de savoir où le bouton se situe sur son échelle. La **figure 2** montre la répartition des différentes fonctions sur les potentiomètres.

À titre d'exemple, prenons le potentiomètre de la source de bruit (code MIDI CC 75). Le secteur de −135° à −45° (valeurs MIDI CC 0 à 39) agit comme un interrupteur de sélection entre différentes couleurs de bruit. C'est le mode *noise* (bruit). De −45° à +90° (MIDI CC 40 à 111), ce potentiomètre détermine la longueur du registre à décalage. Le synthé est alors en mode *pattern* (motif). Le dernier secteur, de +90° à +135° (MIDI CC 112 à 127) règle la profondeur de modulation LFO de la longueur du registre à décalage.

Ce n'est pas l'interface la plus intuitive qui soit, et au début de l'utilisation du PRBSynth1, il faut fréquemment se référer à la figure 2, mais certaines LED aident à appréhender certains réglages. Un dessin plus grand peut être téléchargé sur [1]. Une autre solution serait de réaliser une face avant avec ces informations imprimées autour des boutons.

Trois LED indiquent le type de filtre sélectionné ; la fréquence de coupure du filtre détermine son intensité. La LED LFO scintille au rythme du LFO et son intensité suit la forme d'onde sélectionnée. La position stéréo du signal est matérialisée par deux LED. Enfin, il y a un interrupteur de déclenchement qui permet soit de lancer un seul cycle du LFO (mode « déclenché »), soit de modifier les accents du LFO ; ce mode est intéressant pour les sons rythmiques.

Aperçu du circuit

Le schéma du PRBSynth1 (**fig. 3**) reste simple car la plupart des fonctions sont réalisées en logiciel. On y voit un Arduino Pro Mini entouré d'un CN/A, d'une commande de volume et d'un circuit de panoramique stéréo, ainsi que d'une entrée/sortie MIDI.

Un convertisseur N/A de 7,75 bits

À première vue, le CN/A qui convertit le signal audio numérique en signal analogique semble être un simple réseau de résistances pondérées de 6 bits, mais les valeurs des résistances et leur commande cachent quelques astuces. Ce CN/A à 6 bits peut produire 216 valeurs de sortie au lieu des 64 attendues. Comment est-ce possible ? Grâce aux trois bits inférieurs (LSB) qui peuvent prendre trois valeurs ('0', 'ouvert' et '1')

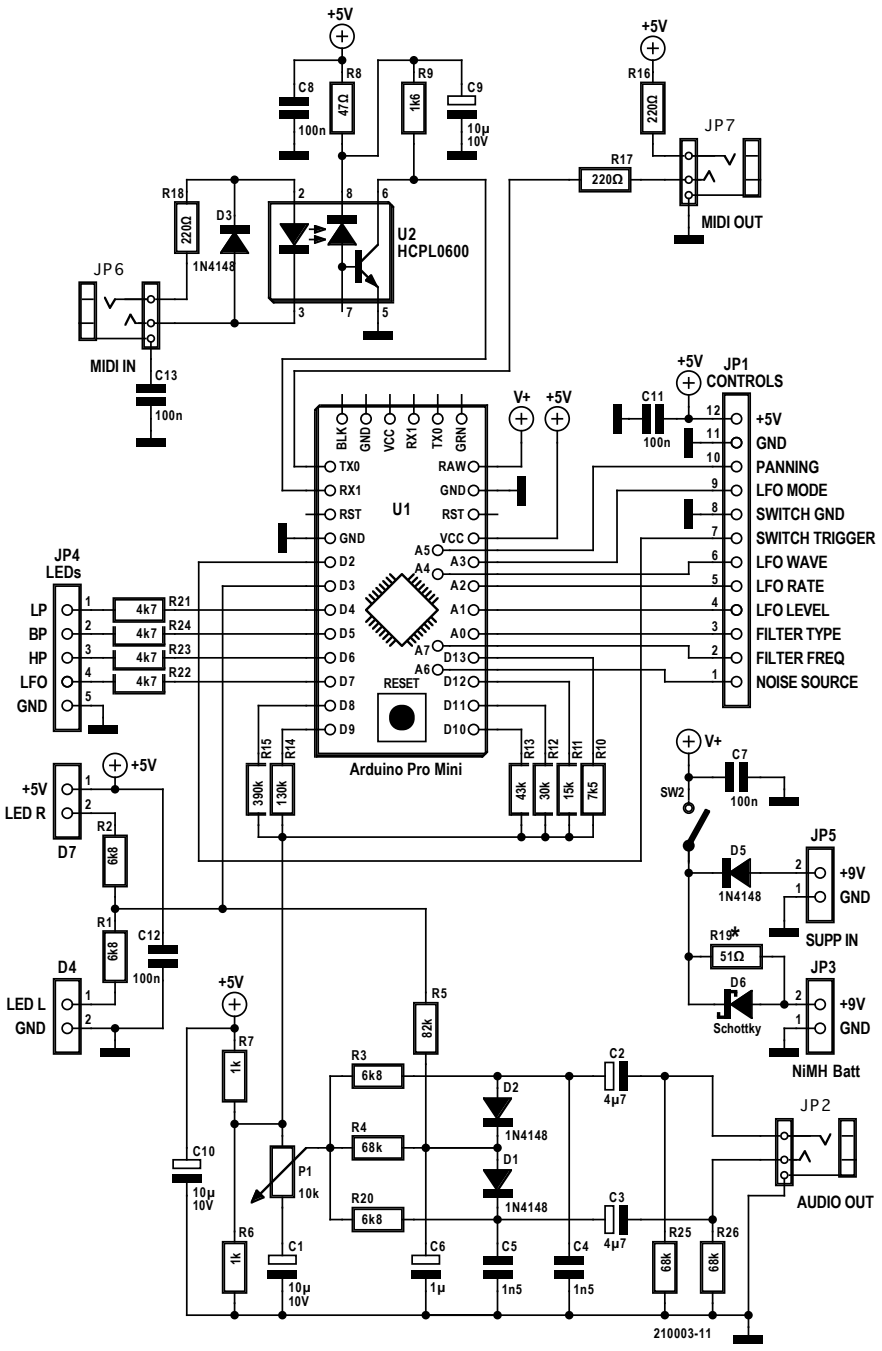


Figure 3. Un Arduino Pro Mini, un optocoupleur et une poignée de composants passifs constituent le circuit complet. Les plus remarquables sont le circuit de panoramique stéréo (D1 et D2) et le BiTriDAC à 6 bits (R10 à R15).

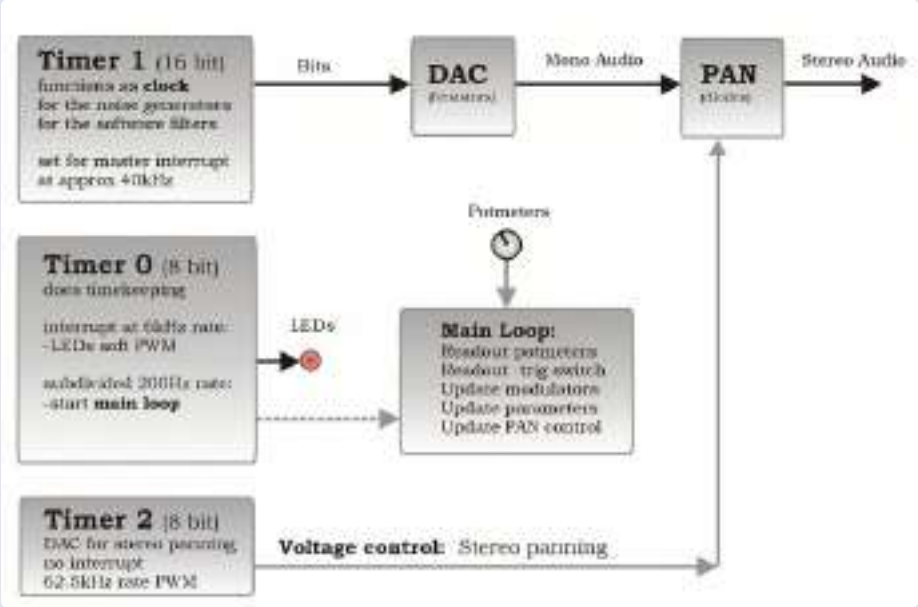


Figure 4 : Le moteur sonore est piloté par des timers. Le Timer 1 à haute fréquence produit des échantillons qui sont modulés par le Timer 0 à basse fréquence. Le Timer 2 commande le panoramique stéréo. Le traitement des données MIDI (non illustré) est effectué lorsque les timers tournent.

au lieu de deux : ce sont des bits trinaires. Les trois bits supérieurs (MSB) sont des bits ordinaires. Comme j'aime l'appeler, ce *BiTriDac* a une résolution de 7,75 bits.

La contrepartie est un logiciel un peu plus complexe, parce que les valeurs de 8 bits doivent être réparties (mappées) sur 7,75 bits et que la mise en place d'un bit trinaire est plus compliquée. Mais la fonction de mappage applique un écrêtage léger, ajoutant au son la touche analogique recherchée.

Pour une performance optimale du CN/A, il est vivement conseillé de retirer de l'Arduino Pro Mini la LED (ou mieux, sa résistance série) reliée à la broche 13 (à côté du bouton *reset*) car elle est en parallèle avec R10 du CN/A. Cela permet d'améliorer la précision du CN/A et d'économiser un peu d'énergie.

Panoramique stéréo commandé en tension

Les diodes D1 & D2 agissent comme des résistances CA variables qui atténuent le petit signal audio (moins de 200 mV) pour les voies gauche (D2) et droite (D1). Les deux diodes sont polarisées en moyenne à 2,5 V par le curseur de volume P1. L'atténuation est pilotée par les courants continus parcourant les diodes. Ils sont influencés par la tension sur le nœud R5/C6/D1/D2. Cette tension est une version filtrée du signal MLI sur la broche 3 de l'Arduino Pro Mini. La dissymétrie du courant des diodes déplace le son dans le champ stéréo. Pour que l'effet soit un peu plus doux, le panoramique est limité à environ 10 dB de déséquilibre.

Stricto sensu, ce circuit n'est pas un simple panoramique stéréo car il ajoute également une modulation d'amplitude (trémolo) tout en déplaçant le son. Appliqué à un synthétiseur, cela peut être considéré comme un plus. Il en va de même pour la distorsion ajoutée au signal d'entrée par la caractéristique de transfert non-linéaire des diodes pour les forts signaux.

Pour finir, une note sur R19. On peut monter cette résistance dans le cas où l'appareil est

Un logiciel de filtres presque trivial

Pour programmer les filtres, il suffit d'imiter un filtre analogique. Le logiciel se contente d'imiter le flux de signaux des filtres analogiques et ne nécessite que quatre lignes de code :

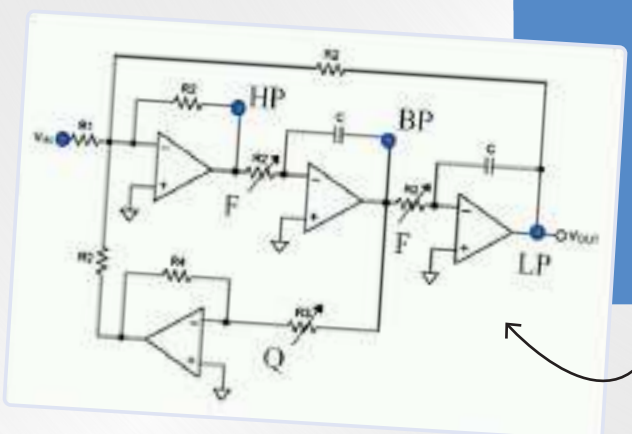
```
LP = LP + F*BP; // F est le réglage de la fréquence du filtre
HP = Vin - LP - q*BP; // note : q = 1/Q, évite une division
BP = BP + F*HP;
DAC = LP; // ou sélectionner DAC=HP, DAC=BP
```

Et les deux intégrateurs ? Le code s'exécute à une vitesse fixe. Par conséquent, reprendre la valeur précédente et y ajouter une variable équivaut à une intégration analogique (voir la 1re ligne du code et aussi la 3e ligne).

Disposer d'un processeur à 8 bits avec 2 Ko de mémoire pour calculer un filtre logiciel implique un code compact et rapide dans la partie traitement audio. L'utilisation d'octets et, dans certains cas, d'entiers de 16 bits est la meilleure solution. Il faut bannir les divisions dans la partie traitement audio.

Dans le PRBSynth1, ce filtre logiciel fonctionne à 40 kHz* max. et les variables LP, BP, HP sont des entiers. Les autres variables sont des octets. Cela donne-t-il un filtre « Hi-Fi » ? Non, mais comme le signal d'entrée est du bruit, nous pouvons faire l'impasse sans trop d'artefacts audibles.

*Astuce : en mode *noise* (bruit), la plage d'accord du filtre est étendue d'une octave environ en ralentissant lentement l'horloge d'échantillonnage pour les plus basses fréquences de coupure du filtre. Comme en entrée on a du bruit, il n'y a pas de problème de crénelage.



Potentiomètre	Code CC
Fréquence de coupure du filtre	1
Volume (toujours maximiser)	7
Panorama	10
Type de filtre & Q	73
Source de bruit	75
Profondeur de modulation LFO	76
Vitesse LFO	77
Onde LFO	78
Mode LFO	79

Tableau 1. Correspondance entre le potentiomètre et le code MIDI CC.

alimenté par une batterie rechargeable de 9 V *NiMH* (et *non* Li-Po !) connectée à JP3 car cela la recharge à travers JP5. Dans tous les autres cas, mieux vaut ne pas la monter.

Logiciel

Le programme ou moteur sonore du PRBSynth1 est assez simple. Il peut être divisé en quatre tâches fonctionnant en parallèle (**fig. 4**) :

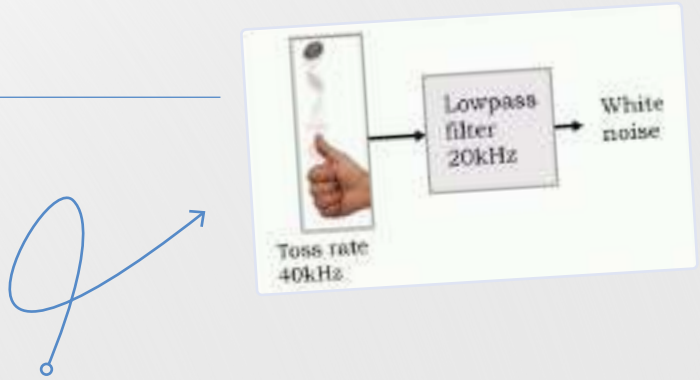
- Traitement des échantillons audio
- Modulation et interface utilisateur
- Contrôle du panorama
- Traitement MIDI

La 1re tâche, la plus importante, est commandée par le Timer 1 cadencé par la fréquence d'échantillonnage. À chaque tic, cette tâche produit un échantillon de bruit, le fait passer par le filtre et l'amplificateur, puis l'envoie au CN/A.

La 2e tâche exécute le LFO, lit les potentiomètres et le bouton de déclenchement, et met à jour les paramètres modulés. Le Timer 0 la contrôle et gère également l'intensité des LED par le biais d'une MLI logicielle.

Le Timer 2 tourne seul et sert de CN/A MLI pour produire la tension de commande du panoramique stéréo. C'est la tâche 3.

La tâche 4 est une tâche de fond qui s'exécute en l'absence de prise en charge d'interruption. Elle traite les données MIDI entrantes. C'est possible parce que le MIDI est relativement lent, 310 µs par message, et que l'UART a un tampon. L'utilisation de ce synthétiseur avec un séquenceur logiciel actif sur plusieurs



Pour faire du bruit, lancez des pièces en l'air !

Si on tire à pile ou face de façon répétitive et rapide, on obtient un flux binaire similaire à un signal MLI aléatoire. En traitant ce signal avec un filtre passe-bas, on obtient un flux de valeurs moyennes fluctuant de façon aléatoire entre 0 et 1.

On peut simuler le lancer d'une pièce de monnaie dans un logiciel en produisant une longue séquence de valeurs aléatoires, dont on n'utilise qu'un seul bit (par ex. le bit 0). Le résultat n'est plus vraiment aléatoire, mais il en est suffisamment proche pour être utilisé comme tel. Ce type de séquence s'appelle séquence de bits pseudoaléatoire (PRBS, d'où le nom du synthé) et peut être générée à l'aide d'un registre à décalage dit *tapped* (dont les bits peuvent être lus) et d'un XOR de rétroaction. Si la séquence est assez longue (quelques secondes ou plus), elle ressemblera à du bruit. On peut créer un tel registre à décalage avec rétroaction par logiciel en seulement deux lignes de code :

```
// Un registre à décalage de 16 bits et XOR sur les bits 16, 15, 13 et 4 produit une PRBS.
// init. sr comme mot sr=1, init. DAC comme octet.
sr = (sr<<1 | (((sr>>15)&1) ^((sr>>14)&1) ^ ((sr>>12)&1) ^ ((sr>>3)&1) ));
DAC = sr && 128; // prendre un bit de sr par la fonction AND avec 2^n.
```

Bruit rose ou rouge

Une particularité du signal PRBS est que la fréquence d'échantillonnage *F* est totalement absente du spectre. De ce fait, l'abaissement de cette dernière ne crée pas d'artefacts d'horloge audibles. On met à profit cette propriété en « lançant » des pièces à *F*, *F*/3, *F*/9 et *F*/27 et en additionnant les signaux. Ce faisant, on augmente le contenu en BF nécessaire pour produire du bruit rose ou rouge.

Réduction de la charge de calcul

Cette façon de produire du bruit rose ou rouge entraîne un pic de charge de travail si, à certains moments, deux, trois, voire quatre pièces doivent être lancées en même temps. Pour y remédier, une priorité à la pièce la plus lente est instaurée : seule la plus lente est lancée, les autres sautent un battement. En pratique, cela ne fait aucune différence audible.

Créer des motifs pour le mode *pattern*

C'est facile, car il suffit de raccourcir la chaîne en réinitialisant le registre à décalage à une valeur de départ mise à jour par le LFO.

PRODUITS

- **Arduino Uno**
  www.elektor.fr/15877
- **Joy-IT Nano V3**
  www.elektor.fr/18615



- **C. Valens, « Maîtrisez les microcontrôleurs à l'aide d'Arduino (3e édition) »**
  www.elektor.fr/18064



Figure 5. Le PRBSynth1 assemblé à l'intérieur d'une goulotte électrique en PVC. Pour garder le circuit imprimé petit et facile à intégrer, les LED et potentiomètres ne sont pas montés, mais câblés via des connecteurs. Une pile de 9 V alimente le circuit.

canaux MIDI n'a posé aucun problème. Notez que la *sortie* MIDI (logiciel non écrit à ce jour) est là pour faire jaillir des idées (extension future).

Synthèse

Le PRBSynth1 est un minisynthétiseur (**fig. 5**) offrant de nombreuses possibilités, nonobstant l'Arduino Pro Mini qui l'anime. Il peut être construit sur une plaque d'essai et se prête bien à l'expérimentation matérielle et logicielle. Tous les détails de sa réalisation et de sa programmation, ainsi que des exemples sonores, sont disponibles sur la page web du projet au labo [1]. Une vidéo est disponible sur YouTube [3]. Amusez-vous bien !

210003-04

Implémentation MIDI

Nous nous sommes efforcés d'améliorer la musicalité du PRBSynth1 s'il est connecté à un clavier MIDI, de préférence avec molettes de vélocité, hauteur et modulation. Les huit potentiomètres peuvent être neutralisés par un contrôleur continu MIDI CC. On reprend le contrôle en tournant le potentiomètre correspondant du synthé. Pour une version du synthétiseur uniquement MIDI, il faut donc supprimer la partie du programme qui lit les potentiomètres. Sans cette suppression, les entrées « en l'air » liront des valeurs fluctuantes qui prendront le relais des commandes MIDI. La version du programme appelée no_potm a déjà cette partie désactivée.

Avec un clavier (ou un séquenceur) MIDI, la molette de modulation règle la fréquence de coupure du filtre. Le son est coupé quand aucune touche n'est enfoncée. Ce mode s'active dès l'entrée de la première note MIDI et est réinitialisé à la mise sous tension.

La vélocité de la note (c.-à-d. la vitesse d'enfoncement de la touche) est convertie en profondeur de modulation du filtre du LFO.

En mode *noise*, le son n'a pas de hauteur, et le clavier contrôle donc la vitesse du LFO à la place. Une octave vers le haut double la vitesse. Cela permet de produire facilement des motifs rythmiques lorsque le LFO est en mode continu. Si le LFO est en mode « déclenché », on peut produire des sons enveloppés longs/courts en jouant une touche basse/haute.

En mode *pattern*, le clavier contrôle la longueur du motif. Ceci a été astucieusement conçu pour créer une échelle musicale (par ex., une octave vers le haut divise la longueur par deux) et permet de jouer des mélodies. Dans ce mode, la vitesse du LFO est commandée par le potentiomètre correspondant.

Contributeurs

Idée et conception : **Raymond Schouten**
Rédaction : **Clemens Valens**
Mise en page : **Harmen Heida**
Traduction : **Yves Georges**

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (rs.elc.projects@gmail.com) ou contactez Elektor (redaction@elektor.fr).

LIENS

[1] PRBSynth1, page Elektor Labs : www.elektormagazine.fr/labs/synthesizer-prbsynth1
[2] Site web de l'auteur : http://www.rs-elc.nl
[3] Vidéo de démonstration : https://youtu.be/agUzP0t1k7Y

démarer en électronique... (10)

...est moins difficile qu'on ne l'imagine ! Passons aux bobines (ou inductances)

Eric Bogers (Elektor)

Dans ce dernier numéro de l'année, nous allons voir qu'il y a bien des choses intéressantes à faire avec les inductances. Pas convaincu ? Lisez la suite et vous verrez !



L'inductance comme résistance en courant alternatif

Tout comme un condensateur, une bobine se comporte comme une résistance en courant alternatif, mais « en sens opposé ». Pour comprendre, examinons un circuit composé d'une bobine et d'une résistance en série.

Prenons une bobine d'inductance 10 mH et une résistance de 50 Ω, à une fréquence de 1 kHz. À cette fréquence, la réactance de la bobine s'élève à :

$$X_L = 2 \cdot \pi \cdot f \cdot L = 2 \cdot \pi \cdot 1000\ \text{Hz} \cdot 0,01\ \text{H} = 62,8\ \Omega$$

Là encore, il n'est pas correct de « simplement » additionner cette impédance à la résistance. Cela doit être fait de manière vectorielle :

$$X_{tot} = \sqrt{X_L^2 + R^2} = \sqrt{(68,2\Omega)^2 + (50\Omega)^2} = 80,3\Omega$$

Comme pour le condensateur, il existe un déphasage entre le courant et la tension. Le déphasage s'élève ici à :

$$\varphi = \arctan\frac{X_L}{R} = \arctan\frac{62,8\Omega}{50\Omega} = 51,5°$$

Nous vous faisons grâce du calcul pour le circuit d'une bobine et d'une résistance en parallèle ; en utilisant ces mêmes valeurs de composants, l'impédance totale est de 39,1 Ω avec un angle de phase de –38,5° (vérifiez par vous-même).

Filtres passe-haut et passe-bas

Les filtres passe-haut et passe-bas peuvent être réalisés non seulement à partir de condensateurs et de résistances, mais aussi à partir d'inductances et de résistances (**fig. 1**).

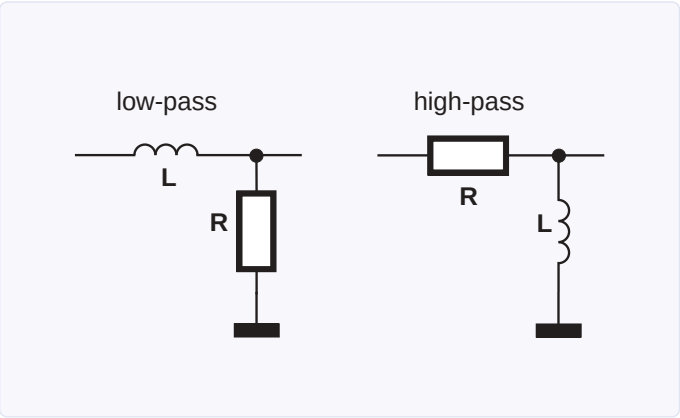


Figure 1. Filtres RL.

Pour les caractéristiques de fréquence et de phase, on procède comme pour les filtres constitués de condensateurs et de résistances. À la fréquence de coupure, l'impédance est égale à la résistance :

$$R = 2 \cdot \pi \cdot f \cdot L \;\rightarrow\; f_{transition} = \frac{R}{2 \cdot \pi \cdot L}$$

Il va de soi qu'on peut réaliser des circuits de filtrage comprenant à la fois des inductances et des condensateurs. C'est d'ailleurs précisément ce que nous ferons plus tard. Mais penchons-nous sur autre chose.

Compensation et puissance réactive

Une application remarquable est la compensation de la puissance réactive. Mais de quoi s'agit-il précisément ?
L'énergie consommée par une résistance « ohmique » (purement résistive) est dissipée sous forme de chaleur par cette résistance. En revanche, l'énergie absorbée par un condensateur ou une inductance est stockée dans ces composants sous la forme d'une charge électrique ou d'un champ magnétique puis restituée à la première occasion. En moyenne, sur une longue période, les condensateurs et les inductances ne consomment donc absolument aucune énergie, ce que traduit la formule suivante (qu'il est inutile de retenir) :

$$p = \int_0^{2\pi} U_0 \cdot \sin\varphi \cdot I_0 \cdot \cos\varphi = U_0 \cdot I_0 \cdot \sin^2\varphi \Big|_0^{2\pi} = 0$$

$$p = \int_0^{2\pi} U_0 \cdot \sin\varphi \cdot I_0 \cdot -\cos\varphi = U_0 \cdot I_0 \cdot -\sin^2\varphi \Big|_0^{2\pi} = 0$$

Cela s'explique par le fait que dans ces composants le courant et la tension ne sont pas en phase, mais sont décalés exactement de 90° l'un par rapport à l'autre, en avance pour le condensateur et en retard pour l'inductance. La puissance instantanée est tantôt positive et tantôt négative – lorsqu'elle est intégrée sur une période entière, elle est égale à zéro. À l'inverse, dans une résistance ohmique, le courant et la tension sont toujours en phase, de sorte que la puissance est toujours positive. L'énergie électrique absorbée par une résistance ohmique est convertie dans une autre forme – en chaleur par exemple – on parle alors de *puissance active*. En revanche, celle que les inductances et les condensateurs stockent temporairement est qualifiée de *puissance réactive*. Dans les formules, on la désigne par la lettre Q.

Un circuit qui, outre des éléments ohmiques, contient également des condensateurs et/ou des inductances, consomme de la puissance active et de la puissance réactive. Comme ces composantes de la puissance sont déphasées, il faut les additionner vectoriellement en appliquant la formule suivante (**fig. 2**) :

$$S = \sqrt{P^2 + Q^2}$$

dans laquelle *S* est la puissance apparente et *P* la puissance active. Afin de la distinguer de la puissance active, *S* n'est pas exprimée en watts (W) mais en voltampères (VA). Pour les puissances active et réactive, on peut maintenant écrire :

$$P = S \cdot \cos\varphi \qquad Q = S \cdot \sin\varphi$$

La valeur du cos φ est souvent indiquée sur la plaque d'identification d'un moteur électrique, ce qui permet de déduire la puissance active de la puissance apparente. Supposons que pour un moteur donné soient indiqués une tension de 230 V, un courant de 2 A et un cos φ de 0,8. Pour la puissance active, on a :

$$P = U \cdot I \cdot \cos\varphi = 230\ \text{V} \cdot 2\ \text{A} \cdot 0{,}8 = 368\ \text{W}$$

Et de la formule

$$\sin^2\varphi + \cos^2\varphi = 1 \;\rightarrow\; \sin\varphi = \sqrt{1 - \cos^2\varphi} = \sqrt{1 - (0{,}8)^2} = 0{,}6$$

on déduit la puissance réactive :

$$Q = U \cdot I \cdot \sin\varphi = 230\ \text{V} \cdot 2\ \text{A} \cdot 0{,}6 = 276\ \text{VA}$$

Vous pensez peut-être, qu'en tant qu'utilisateur de ce moteur, que la puissance réactive ne vous concerne pas – après tout, votre compteur électrique ne mesure que la puissance active. Mais

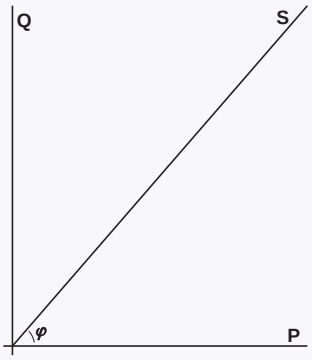


Figure 2. Puissance apparente, puissance active et puissance réactive.

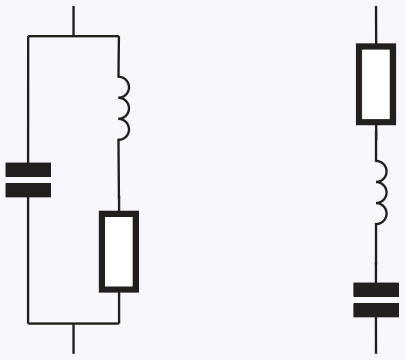


Figure 3. Compensation en parallèle (à gauche) et en série (à droite).

en raison de cette puissance réactive, un courant inutile circule dans les câbles d'alimentation, qui doivent être dimensionnés en conséquence ; en outre, ce courant « inutile » entraîne des pertes réelles dans la résistance des câbles. Une raison suffisante pour compenser la puissance réactive. Pour cela, il suffit de connecter un condensateur de valeur appropriée en parallèle ou en série avec la charge (**fig. 3**).

Commençons par la compensation en parallèle (c'est en général celle qu'on préfère). La puissance réactive globale est complètement compensée lorsque la puissance réactive du condensateur est égale à celle de la bobine. Dans ce cas, le condensateur cède sa puissance réactive à la bobine et inversement, et l'ensemble du circuit se comporte (vu de l'extérieur) comme une charge ohmique (à la fréquence d'utilisation, bien sûr). Pour le courant à travers le condensateur, on a :

$$I = \frac{Q_C}{U} = \frac{276\ \text{VA}}{230\ \text{V}} = 1,2\ \text{A}$$

Ce qui donne pour l'impédance :

$$X_C = \frac{U}{I} = \frac{230\ \text{V}}{1,2\ \text{A}} = 191,67\ \Omega$$

Et donc pour la capacité :

$$C = \frac{1}{2 \cdot \pi \cdot f \cdot X_C} = \frac{1}{2 \cdot \pi \cdot 50\ \text{Hz} \cdot 191,67\ \Omega} = 16,6\ \mu\text{F}$$

Grâce au condensateur, le courant total passe de 2 A à 1,6 A, tandis que la résistance équivalente du circuit passe de 115 Ω à 143,75 Ω. (Elle augmente ? En effet, cela est dû à l'opposition de phase entre le condensateur et l'inductance).

Nous avons mentionné que la compensation en parallèle est généralement préférée. Si nous devions utiliser la compensation en série dans l'exemple ci-dessus, le condensateur devrait avoir une valeur de 46,1 µF, soit presque le triple. La compensation en parallèle est donc plus économique à mettre en œuvre dans la pratique. De plus, avec la compensation en série, le courant total augmente au lieu de diminuer.

210374-04

La série d'articles « démarrer en électronique » est basée sur le livre « Basic Electronics Course » de Michael Ebner, publié par Elektor.

Des questions, des commentaires ?

Contactez Elektor (redaction@elektor.fr).

Contributeurs

Idée et illustrations : **Michael Ebner**
Texte et rédaction : **Eric Bogers**
Traduction : **Helmut Müller**
Mise en page : **Giel Dols**



PRODUITS

- **B. Kainka, *Initiation à l'électronique et programmation de montages pour débutants***
www.elektor.fr/19339

- **R. Mallard, *L'électronique pour les débutants***
www.elektor.fr/15662

voyage dans les réseaux neuronaux (2e partie)

Les neurones logiques

Stuart Cording (Elektor)

Dans la première partie de cette série d'articles, nous avons découvert comment les chercheurs ont lentement approché les caractéristiques fonctionnelles du neurone. La véritable percée des neurones artificiels est venue du perceptron multicouche (MLP) et de l'utilisation de la rétropropagation pour lui apprendre à classer les entrées. En réalisant un MLP à partir de zéro dans Processing, nous avons également montré comment il fonctionnait et ajustait ses poids pour apprendre. Ici, nous reprenons les expériences du passé pour apprendre à notre réseau neuronal le fonctionnement des portes logiques et vérifier si notre perceptron multicouche (MLP) est capable d'apprendre la fonction XOR (OU exclusif).

Nous disposons d'une classe `Neural` flexible pour implémenter un MLP qui pourra être incorporé dans un projet Processing. Pour autant, les exemples examinés jusqu'à présent n'ont pas donné grand-chose. Ils confirment simplement le calcul correct de la progression avant et la façon dont la rétropropagation ajuste les poids du réseau pour apprendre une tâche donnée.
Le moment est maintenant venu d'appliquer ces connaissances à une tâche réelle, celle-là même qui a été étudiée lors des premières recherches sur les unités logiques à seuil (TLU) de McCulloch-Pitts : la mise en œuvre de la logique. Comme nous l'avons déjà découvert, notre MLP devrait résoudre facilement des problèmes linéairement séparables tels que les fonctions ET et OU. De plus, il devrait également être capable de résoudre la fonction OU exclusif, ce que ne pouvaient pas faire les TLU et les premiers neurones artificiels. Au cours de ce voyage, nous examinerons également comment ces réseaux apprennent grâce à une implémentation visuelle du réseau neuronal. Nous étudierons également l'impact du taux d'apprentissage choisi sur l'erreur de sortie pendant l'apprentissage.

ET

Si un réseau neuronal peut apprendre à reproduire la fonction ET, il n'opère pas tout à fait de la même manière. Ce que nous faisons en fait, c'est appliquer des entrées à un réseau qui a appris la fonction ET et lui demander : « Dans quelle mesure es-tu sûr que cette combinaison d'entrées est le modèle auquel nous attribuons un 1 ? ».
Pour le démontrer, un exemple de projet a été préparé et est disponible dans le dossier */processing/and/and.pde* du dépôt GitHub. Il doit être ouvert à l'aide de Processing.
Notre réseau neuronal comporte deux entrées et une seule sortie pour répondre aux caractéristiques d'une porte ET à deux entrées. Entre les nœuds d'entrée et de sortie, nous implémentons quatre nœuds cachés (**fig. 1**). Nous aborderons plus tard la manière de déterminer le nombre de nœuds cachés requis. Le **listage 1** présente le code permettant de préparer le réseau.
L'objectif est d'entraîner le réseau à reconnaître le motif '11' sur les entrées. Nous voulons également nous assurer que les alternatives '00', '01' et '10' ne dépassent pas notre seuil de classification. Pendant l'apprentissage du réseau, nous appliquerons les stimuli et les résultats attendus indiqués dans le **tableau 1**.

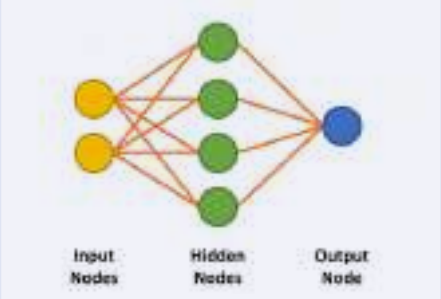


Figure 1. Configuration MLP utilisée pour apprendre la fonction ET (les biais ne sont pas indiqués).

entrée		sortie attendue
i_1	i_2	o_1
0.01	0.01	0.01
0.01	0.99	0.01
0.99	0.01	0.01
0.99	0.99	0.99

Tableau 1. Entrées et sorties attendues du MLP après apprentissage complet de la fonction ET.

Notez que, plutôt que de travailler avec des niveaux logiques, ces réseaux fonctionnent avec des valeurs décimales. Dans ce cas, un 1 est entré comme 0,99 (presque 1), tandis qu'un 0 est entré comme 0,01 (presque 0).
La sortie sera également comprise entre 0,0 et 1,0. Nous pouvons voir cela comme un niveau de confiance dans la correspondance entre les entrées et la classification apprise : nous avons par ex. 96,7 % de confiance que les deux entrées sont à '1', plutôt que d'avoir une sortie logique tranchée 0/1 d'une porte ET réelle.
Commençons par déterminer si ce réseau nouvellement créé « sait » quelque chose en lui appliquant quelques entrées et en lui demandant ce qu'il a en sortie. N'oubliez pas que les résultats produits seront différents à chaque fois, car le constructeur applique des valeurs aléatoires aux poids.
Le code ci-dessous donne la réponse du réseau pour les entrées '11' et '00'. Il est fort probable que le résultat pour '11' soit proche de 0,99, alors que le résultat pour '00' sera beaucoup plus grand que le 0,01 espéré :

```
// Vérifier la sortie de la
   fonction ET pour une entrée 00
network.setInputNode(0, 0.01);
network.setInputNode(1, 0.01);
network.calculateOutput();
println("For 00 input, output is: ",
network.getOutputNode(0));
// Vérifier la sortie de la
   fonction ET pour une entrée 11
network.setInputNode(0, 0.99);
network.setInputNode(1, 0.99);
```

Listage 1. Section de 'and.pde' qui configure le réseau neuronal pour apprendre la fonction ET.

```
// We'll use two inputs, four hidden nodes, and one output node.
network = new Neural(2,4,1);

// Set learning rate here
network.setLearningRate(0.5);

println(network.getNoOfInputNodes(), " ", network.
getNoOfHiddenNodes(), " ", network.getNoOfOutputNodes());

// Set network biasing here
network.setBiasInputToHidden(0.25);
network.setBiasHiddenToOutput(0.3);

network.displayOutputNodes();

println(network.getTotalNetworkError());

network.turnLearningOn();
```

```
network.calculateOutput();
println("For 11 input, output is: ",
network.getOutputNode(0));
```

La sortie obtenue au cours du test est la suivante :

```
For 00 input, output is: 0.7490413
For 11 input, output is: 0.80063045
```

Nous voyons ici que lorsque nous appliquons 0,99 aux deux entrées, le réseau neuronal pense que l'entrée est '1 ET 1' avec un niveau de confiance de 0,8006, ce qui correspond à 80,06 %. Ce n'est pas si mal à ce stade. Cependant, en appliquant 0,01 aux deux entrées, le réseau neuronal a un niveau de confiance de 0,7490 (74,90 %) que l'entrée est '1 ET 1'. Nous sommes encore loin de ce que nous voulons, soit une valeur proche de 0 %.

Pour apprendre au réseau comment fonctionne une fonction ET, un entraînement est nécessaire. Pour ce faire, il faut définir les entrées et la sortie souhaitée de manière appropriée pour les quatre cas (00, 01, 10 et 11, respectivement 0, 0, 0 et 1) et appeler la méthode `calculateOuput()` en activant l'apprentissage après chaque modification. Cette opération est effectuée dans une boucle comme suit :

```
while (/* learning the AND function
      */) {
// Learn 0 AND 0 = 0
network.setInputNode(0, 0.01);
network.setInputNode(1, 0.01);
network.setOutputNodeDesired(0, 0.01);
network.calculateOutput();
// Learn 0 AND 1 = 0
...
// Learn 1 AND 0 = 0
...
// Learn 1 AND 1 = 1
network.setInputNode(0, 0.99);
network.setInputNode(1, 0.99);
network.setOutputNodeDesired(0, 0.99);
network.calculateOutput();
}
network.turnLearningOff();
```

La décision d'arrêter l'entraînement du réseau peut être déterminée de plusieurs façons. Dans cet exemple, chaque cycle d'apprentissage est considéré comme une époque. Il est possible d'arrêter l'apprentissage lorsqu'un nombre spécifique d'époques, par exemple 10 000, est atteint. Il est également possible d'éliminer l'erreur dans la sortie. Une fois qu'il a atteint un niveau inférieur à 0,01 % par ex., le réseau peut être considéré comme suffisamment précis pour la tâche de classification en cours.

À noter qu'un MLP ne converge pas toujours vers le résultat souhaité. Il est possible que la combinaison des poids et des poids de biais sélectionnés soit malchanceuse. Il se peut également que la configuration du MLP ne soit pas capable d'apprendre votre tâche, probablement en raison d'un nombre trop élevé ou trop faible de nœuds cachés. Ici, il n'y a pas de règles – le nombre approprié de nœuds, les poids de départ et les biais ne peuvent être déterminés que par essai et erreur ou par expérience. Pour cet exemple, quatre nœuds cachés ont été choisis, car l'apprentissage porte sur quatre états. L'idée était que chaque nœud caché apprendrait un état.

Il convient également de noter que l'apprentissage doit se faire par lots, c'est-à-dire que l'ensemble des données d'apprentissage doivent être parcourues du début à la fin, de manière répétée. Si '0 ET 0 = 0' est appliqué de manière répétée au cours de plusieurs milliers de cycles, le réseau s'oriente vers ce résultat et il devient presque impossible d'entraîner les données restantes.

Visualisation

La mise en œuvre de base étant traitée, examinons maintenant de plus près l'exemple d'entraînement du réseau pour apprendre la fonction ET. Pour mieux montrer comment les réseaux neuronaux apprennent, le réseau est visualisé dans l'application pendant l'apprentissage et, ensuite, au cours du fonctionnement.

En cliquant sur 'Run' (Exécuter), on obtient la sortie représentée sur la **figure 2** (capture d'écran). Au départ, l'application est en mode apprentissage et enseigne au réseau la sortie attendue pour une fonction ET, pour les deux entrées. Les nœuds d'entrée se trouvent à gauche. Pendant l'apprentissage, les valeurs d'entrée passent rapidement entre les niveaux logiques 0 et 1. À droite se trouve le nœud de sortie unique. Initialement, il est à 0. La décision de produire un 1 en sortie n'est prise que si le nœud de sortie donne un niveau de confiance de plus de 90 % pour les deux entrées à 1. Sinon, un 0 est émis. Cette décision est prise entre les lignes 341 et 346 dans *and.pde*

```
// Output Node Text
if (network.getOutputNode(0) > 0.9) {
text("1", 550, 280);
} else {
text("0", 550, 280);
}
```

Au départ, la sortie reste à 0, car le niveau de confiance de 90 % n'a pas encore été atteint. Après environ 5 000 époques, la valeur de sortie devrait commencer à osciller entre 0 et 1, ce qui prouve que le réseau a commencé à classifier avec succès que l'entrée '11' doit produire un '1'. À ce stade, l'erreur totale du réseau est d'environ 0,15 %. Si cela ne se produit pas, il est probable que le réseau s'est bloqué et qu'il ne pourra pas apprendre cette fois-ci.

Au fur et à mesure que le réseau apprend, les poids entre les nœuds sont affichés sous forme de lignes d'épaisseur et de couleurs variables. Plus la ligne est épaisse, plus la valeur est grande. Les lignes noires indiquent des nombres positifs, tandis que les lignes marron indiquent des nombres négatifs.

Chaque fois que le code est exécuté, les lignes seront différentes. Vous remarquerez cependant qu'un modèle se développe. Deux nœuds cachés ont toujours une ligne marron et une ligne noire ; un nœud caché a deux lignes noires ; et un nœud caché a deux lignes marron. Les lignes entre les nœuds cachés et le nœud de sortie se conformeront également à un modèle, la seule ligne noire émanant du nœud avec deux lignes de poids noires entrantes.

Il s'agit d'un aperçu intéressant, car il montre comment le réseau a appris la fonction ET. Le '00' en entrée se convertit facilement en un 0 à la sortie, tout comme le '11' en un 1. Pour les combinaisons '01' et '10', il semble que les 0 fassent le plus gros du travail en poussant la sortie vers 0.

L'application est programmée pour arrêter l'apprentissage lorsque l'erreur totale du réseau est < 0,05 % à la ligne 57 de *and.pde*. Il est également possible de programmer l'apprentissage pour qu'il s'arrête après un certain nombre d'époques en utilisant la ligne 55. Une fois l'apprentissage terminé, l'application parcourt simplement les entrées binaires dans l'ordre, permettant ainsi au réseau neuronal de montrer ce qu'il a appris (**fig. 3**).

Dans la console de texte, les entrées sont affichées (sous la forme d'une valeur décimale comprise entre 0 et 3) avec la sortie calculée au format texte, comme suit :

```
0 : 5.2220514E-4
1 : 0.038120847
2 : 0.04245576
3 : 0.94188505
0 : 5.2220514E-4
1 : 0.038120847
2 : 0.04245576
3 : 0.94188505
```

Pour ceux qui sont intéressés, l'erreur de sortie pour les entrées appliquées et l'erreur moyenne du réseau sont écrites toutes les 50 époques dans un fichier CSV nommé *and-error.csv*. Il est possible d'importer ce fichier dans Excel pour examiner comment le réseau a convergé vers la solution (**fig. 4**). La sortie montre comment l'erreur oscille entre des valeurs élevées et faibles pour des combinaisons d'entrée/sortie spécifiques. Comme nous l'avons déjà vu, l'erreur élevée est probablement liée à la sortie pour les motifs '00', '01' et '10' qui est beaucoup trop haute pendant la première phase d'apprentissage. L'erreur faible est probablement due au fait que le réseau évalue l'entrée '11'.

Ces erreurs de motifs individuels sont moyennées sur quatre époques pour calculer l'erreur moyenne du réseau. Si votre PC n'utilise pas la langue anglaise, vous pouvez remplacer le ',' dans le fichier CSV par un ';' comme séparateur, et ensuite le '.' par ',' dans un éditeur de texte (tel que Notepad++) avant d'effectuer une importation de données dans Excel.

Le fichier CSV est également intéressant pour examiner l'impact du taux d'apprentissage sur le réseau. Le code de l'exemple utilise un taux d'apprentissage η de 0,5. Des nombres plus élevés permettent au réseau d'apprendre plus rapidement, comme on peut le voir sur la **figure 5**. Cependant,

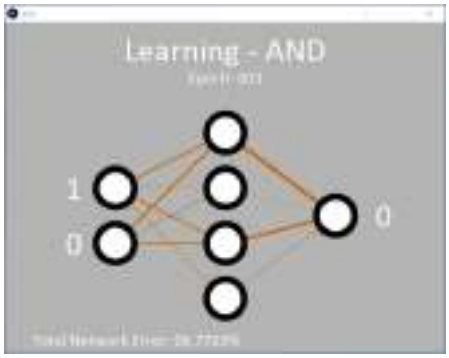
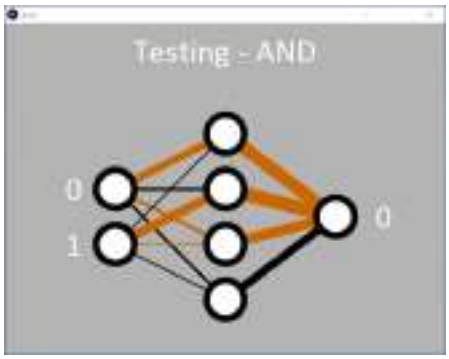


Figure 2. Capture d'écran du réseau neuronal pendant la phase d'apprentissage de la fonction ET.



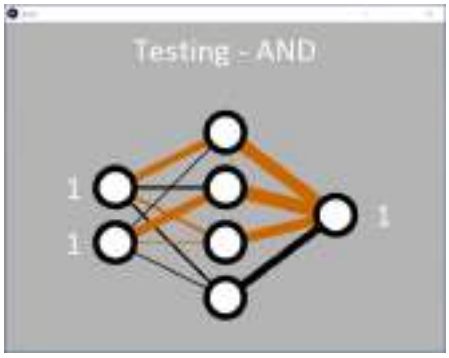


Figure 3. Une fois que le modèle ET a été appris, l'application passe en revue les quatre combinaisons d'entrées binaires et affiche la réponse de sortie du réseau.



Figure 4. Erreur toutes les 50 époques et erreur moyenne pendant l'apprentissage de la fonction ET.



Figure 5. Impact du taux d'apprentissage η sur l'erreur pendant l'apprentissage (10 000 premières époques).

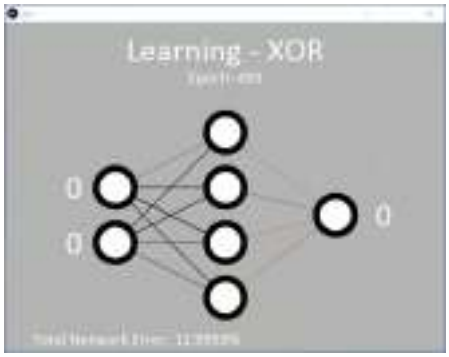


Figure 6. Pour l'apprentissage de la fonction OU exclusif, le MLP montre qu'il a des difficultés à trouver les poids appropriés.

PRODUITS

- Livre en anglais, « Artificial Intelligence » de B. van Dam
  www.elektor.fr/artificial-intelligence-e-book
- Kit AIY Vision pour Raspberry Pi de Google
  www.elektor.fr/google-aiy-vision-kit-for-raspberry-pi
- Caméra AI avec boîtier en silicone de HuskyLens
  www.elektor.fr/huskylens-ai-camera-with-silicone-case

ils peuvent également provoquer des oscillations et aboutir à un réseau qui ne converge jamais vers le résultat d'apprentissage souhaité. Tous les taux d'apprentissage testés ici ont donné lieu à un réseau fonctionnant correctement et ayant appris la fonction ET. Il convient toutefois de noter que les poids de départ ont été choisis de manière aléatoire à chaque fois.

Le MLP peut-il apprendre la fonction OU exclusif ?

Le référentiel comprend également des exemples pour une fonction OU dans *processing/or/or.pde*. Étant linéairement séparable, le MLP n'a aucun problème à apprendre cette fonction non plus. Peut-être trouverez-vous utile d'examiner la différence des poids après l'apprentissage par rapport à l'exemple de la fonction ET. Les fichiers *or.pde* et *and.pde* peuvent être facilement modifiés pour apprendre au réseau les fonctions NON ET et NON OU. Cependant, le moment de vérité arrive avec la fonction OU exclusif.

Un exemple est proposé dans *processing/xor/xor.pde* qui fonctionne comme le code précédent et utilise la même configuration de nœuds MLP 2/4/1 (entrée/caché/sortie) (**fig. 6**). Avec le taux d'apprentissage appliqué ($\eta = 0{,}5$), il faudra probablement 15 000 époques ou plus avant que la sortie ne commence à changer. Environ 35 000 époques sont nécessaires avant que l'erreur moyenne cible de 0,05 % soit atteinte.

Et il est clair que le réseau a du mal à apprendre la fonction OU exclusif. Cela se reflète dans les poids affichés qui oscillent entre positif et négatif avant de choisir une direction, et l'erreur du réseau diminue très progressivement. Cela s'explique par le fait que '00' et '11' (représentés par 0,01 et 0,01, et 0,99 et 0,99 sur les entrées) sont tous deux censés donner une sortie à 0,01. Mathématiquement, des valeurs d'entrée de 0,99 entraînent des valeurs de sortie élevées jusqu'à ce que le réseau soit capable de faire baisser le résultat vers 0,01 pendant l'apprentissage. Cela se voit dans l'erreur de sortie enregistrée dans *xor-error.csv* au cours de l'apprentissage (**fig. 7**).

Malgré les difficultés de la tâche, le réseau apprend la fonction OU exclusif comme demandé. Une fois que l'erreur est inférieure à 0,05 %, le code Processing applique consciencieusement les entrées binaires au réseau, et la sortie répond en détectant correctement les modèles '01' et '10'. Le code affiche alors un 1 sur le nœud de sortie (**fig. 8**).

Comme avec le code ET, nous pouvons voir comment le réseau a appris la fonction OU exclusif. Deux nœuds cachés possèdent une ligne noire et brune entrante, et une ligne noire épaisse sortante (deux nœuds du milieu de la figure 8). Ils semblent être responsables de la classification '01' et '10'. Le réseau a également très bien résolu le passage de '00' en entrée à '0' en sortie (nœud caché supérieur). À cette occasion, l'entrée '11' semble être traitée par le nœud caché inférieur, mais il se peut qu'elle n'ait pas été très bien résolue pendant l'apprentissage, ce qui a entraîné une erreur plus élevée que souhaitée pour l'entrée '11'. Si l'on réexécute le code, il est probable qu'un nœud caché traite manifestement le '11' et que deux lignes noires entrent dans l'un des nœuds cachés (**fig. 9**)



Figure 7. Les difficultés rencontrées par le MLP pour apprendre la fonction OU exclusif se retrouvent dans le résultat d'erreur.

Et après ?

L'une des choses les plus importantes à retenir est peut-être qu'avec les réseaux neuronaux, il n'y a pas de bonne ou de mauvaise réponse. Le réseau lui-même ne fait que classer la probabilité selon laquelle les entrées que vous avez fournies correspondent aux entrées que vous recherchez. Si vous l'avez configuré, entraîné et qu'il donne le résultat souhaité, il est probablement correct. Idéalement, vous cherchez également à obtenir ce résultat avec un nombre minimal de nœuds pour économiser de la mémoire et du temps de calcul.

Bien que la visualisation soit agréable, elle n'est pas totalement nécessaire. Si vous souhaitez en savoir plus, vous pouvez modifier le projet *processing/fsxor/fsxor.pde* qui se passe des visualisations. La suppression du code qui écrit les valeurs d'erreur dans un fichier CSV accélère aussi considérablement l'exécution. Vous pouvez ensuite écrire votre propre code en utilisant la classe `Neural` pour étudier les questions suivantes :

- Quel est l'impact du taux d'apprentissage sur le réseau lors de l'apprentissage de la fonction OU exclusif ? Vous pouvez également essayer les mêmes poids de départ à chaque fois.
- Pouvez-vous initialiser les poids avec des valeurs qui encouragent le réseau à un processus de résolution exigeant moins d'époques ? Examinez éventuellement les poids de sortie d'une session exécutée précédemment.
- De combien de nœuds cachés avez-vous besoin pour l'apprentissage de la fonction ET ? Combien en faut-il pour apprendre la fonction OU exclusif ? Est-il possible d'avoir trop de nœuds cachés ?
- Est-il judicieux d'avoir deux nœuds de sortie ? Le premier pourrait classer les modèles indésirables comme 0,99 (pour le OU exclusif, '00' et '11'), tandis que le second est utilisé pour classer les modèles souhaités comme 0,99 (pour le OU exclusif, '10' et '01').

Dans le prochain article de cette série, nous entraînerons le réseau neuronal à reconnaître les couleurs à l'aide d'une webcam fixée à notre PC. Si vous le souhaitez, pourquoi ne pas développer et tester une configuration de nœuds MLP que vous pensez être à la hauteur de la tâche ?

210160-B-04

Des questions ? Des commentaires ?

Avez-vous des questions ou des commentaires à propos de cet article ? Envoyez un courrier électronique à l'auteur à l'adresse stuart.cording@elektor.com.

Contributeurs

Idée, texte et illustrations : **Stuart Cording**
Rédaction : **Jens Nickel, C. J. Abate**
Illustrations : **Patrick Wielders**
Mise en page : **Harmen Heida**
Traduction : **Pascal Godart**

Apprentissage créatif

L'intelligence artificielle et l'apprentissage machine sont souvent utilisés pour classer intelligemment des images et des ensembles de données complexes, mais peuvent-ils être créatifs ? Le projet FlowMachines du laboratoire de recherche SONY CSL aimerait vous en convaincre. Avec « Daddy's Car », leur IA a apparemment écrit un tube digne des Beatles [4]. Cependant, en creusant un peu plus, on s'aperçoit qu'il a fallu beaucoup d'intervention humaine pour arranger et produire la chanson et écrire les paroles. Créatif ou non, le processus est novateur et passionnant, et une telle IA pourrait être d'une aide précieuse si la créativité humaine venait à se tarir.

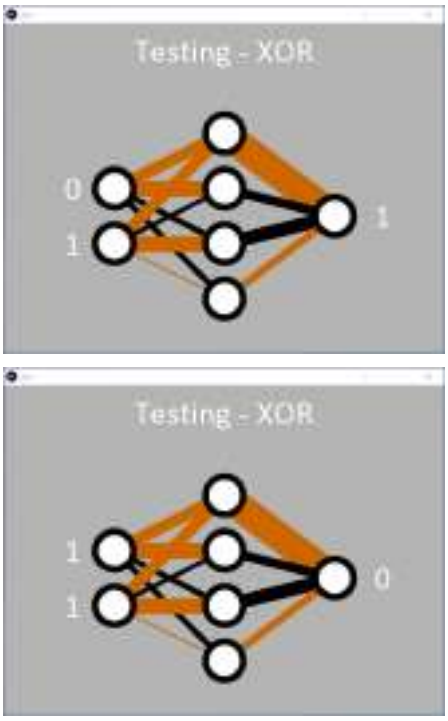


Figure 8. Le réseau montre qu'il a appris avec succès la fonction OU exclusif.

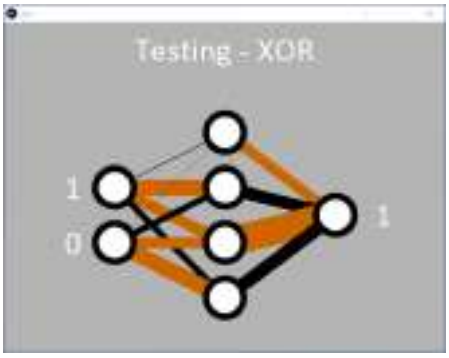


Figure 9. Après avoir réexécuté le code de la fonction OU exclusif, le nœud caché supérieur est manifestement responsable de la classification '11', tandis que le troisième nœud caché est responsable de '00'.

LIENS

[1] Dépôt GitHub, « simple-neural-network » : http://bit.ly/2ZHLv9p
[2] EDI Processing : https://processing.org/
[3] J. Brownlee, « How to Configure the Number of Layers and Nodes in a Neural Network », 07/2018 : http://bit.ly/3aMHqam
[4] T. Lee, chanson « Daddy's Car » chanson composée par intelligence artificielle, Übergizmo, 09/2016 : http://bit.ly/3shyfo6

problèmes de sécurité ?

Combattez le feu par le feu !

Extension à mémoire analogique, protégée par ampoule de flash, pour la boîte à témoin d'effraction

Luka Matić



Dans l'article précédent de notre série de projets sur les communications hautement sécurisées [1], nous avons montré comment construire une boîte à témoin d'effraction (BTE) sécurisée qui préviendra son destinataire, Bob, qu'elle a été ouverte après qu'Alice l'a mise à la poste. Dans cet article, nous la rendons plus sûre en utilisant une technologie du début du siècle dernier.

La sécurité de la BTE repose sur la mise à zéro des codes de défi et de réponse dans la SRAM du microcontrôleur en cas d'ouverture non autorisée du dispositif. Puisque Eve (l'espionne) ne peut pas lire les codes en SRAM, elle ne peut pas la restaurer dans son état d'origine, et Bob sera averti. La communication entre Alice et Bob est alors totalement sécurisée. Est-ce bien le cas ?

Certes, Eve ne sera pas en mesure de restaurer la BTE dans son état d'origine, mais que se passe-t-il si Mallory intervient entre-temps ? Tout ce qu'elle a à faire, c'est de remplacer le microcontrôleur qu'Alice a commandé chez son fournisseur de composants préféré par une variante falsifiée. À moins qu'Alice n'ait accès à des équipements de laboratoire haut de gamme, elle n'a aucun moyen de savoir si son UC est un original vierge ou une variante modifiée malveillante. Mallory peut ajouter des portes dérobées matérielles sans modifier les plans des microcircuits de la matrice de silicium de l'UC [2], « simplement » en manipulant les niveaux de dopage. Les microcircuits auront l'air identiques, même sous le super microscope d'Alice !

L'ouverture de la BTE peut être détectée de manière fiable, mais la mise à zéro de la SRAM peut ne pas être efficace à 100 %. Même si les octets de mémoire contenant les codes critiques sont régulièrement inversés bit à bit pour limiter les effets résiduels de persistance à long terme, Eve peut toujours essayer d'exploiter la rémanence de la SRAM – du moins en théorie – en utilisant une attaque dite de démarrage à froid [3] et lire son contenu. Cela signifie-t-il qu'Alice et Bob devraient renoncer à l'espionnage avec un budget d'amateurs et chercher un emploi dans une grande agence à trois lettres ?

Eh bien, s'ils ajoutent à la BTE une mémoire vraiment irrécupérable, ils pourront poursuivre leurs opérations de guérilla à petit budget. Cela fonctionnera contre les chevaux de Troie matériels d'usage général aussi bien que contre ceux spécifiques à une application. Cependant, après avoir lu [4], [5] et [3], vous vous rendrez rapidement compte que tous les types de mémoire numérique connus au monde souffrent d'une forme de rémanence indésirable des données. Donc, si le numérique ne peut pas faire le travail, alors passons à l'analogique !

Principe de fonctionnement

Les ampoules de flash au magnésium (Mg) (**fig. 1**) sont utilisées en photographie pour l'éclairage artificiel depuis presque le premier jour. Les ampoules de flash jetables à usage unique (par ex. le type standard AG-1B) utilisent la température très élevée du magnésium en combustion pour créer un éclair chaud et brillant. Elles sont encore fabriquées aujourd'hui [6] et sont utilisées en photographie analogique et numérique en raison des effets uniques qu'elles créent.

Papier, bande magnétique ou film analogique, tous perdent leurs données lorsqu'ils sont surchauffés, voire surexposés. Si on en enroule un petit bout autour d'une ampoule, il peut être détruit (brûlé) en allumant l'ampoule. Si nous plaçons l'ampoule à l'intérieur de la BTE, on peut l'utiliser pour avertir Bob.

Ainsi, Alice écrit le code de réponse spécial sur un morceau de papier fin (ou l'enregistre sur une bande magnétique), l'enroule autour de l'ampoule de flash à l'intérieur de la BTE, arme la BTE et l'envoie à Bob. Si Eve manipule la boîte pendant son voyage, l'ampoule se déclenche et le code est détruit. Si Eve ne peut pas lire le code, elle ne pourra pas remettre la BTE dans son état d'origine et, pour finir, Bob sera au courant.

En plus d'être efficace contre les chevaux de Troie matériels, ce circuit permet également d'éviter les attaques par démarrage à froid. Bien que très difficiles à réaliser contre la BTE qui est protégée contre les basses températures et qui efface la SRAM à chaque redémarrage, elles méritent d'être mentionnées. La récupération de la SRAM ne servira à rien non plus si un flash détruit les informations contenues sur un support physique.



Figure 1. Cette ampoule de flash au magnésium est la pierre angulaire de notre alarme anti-effraction.

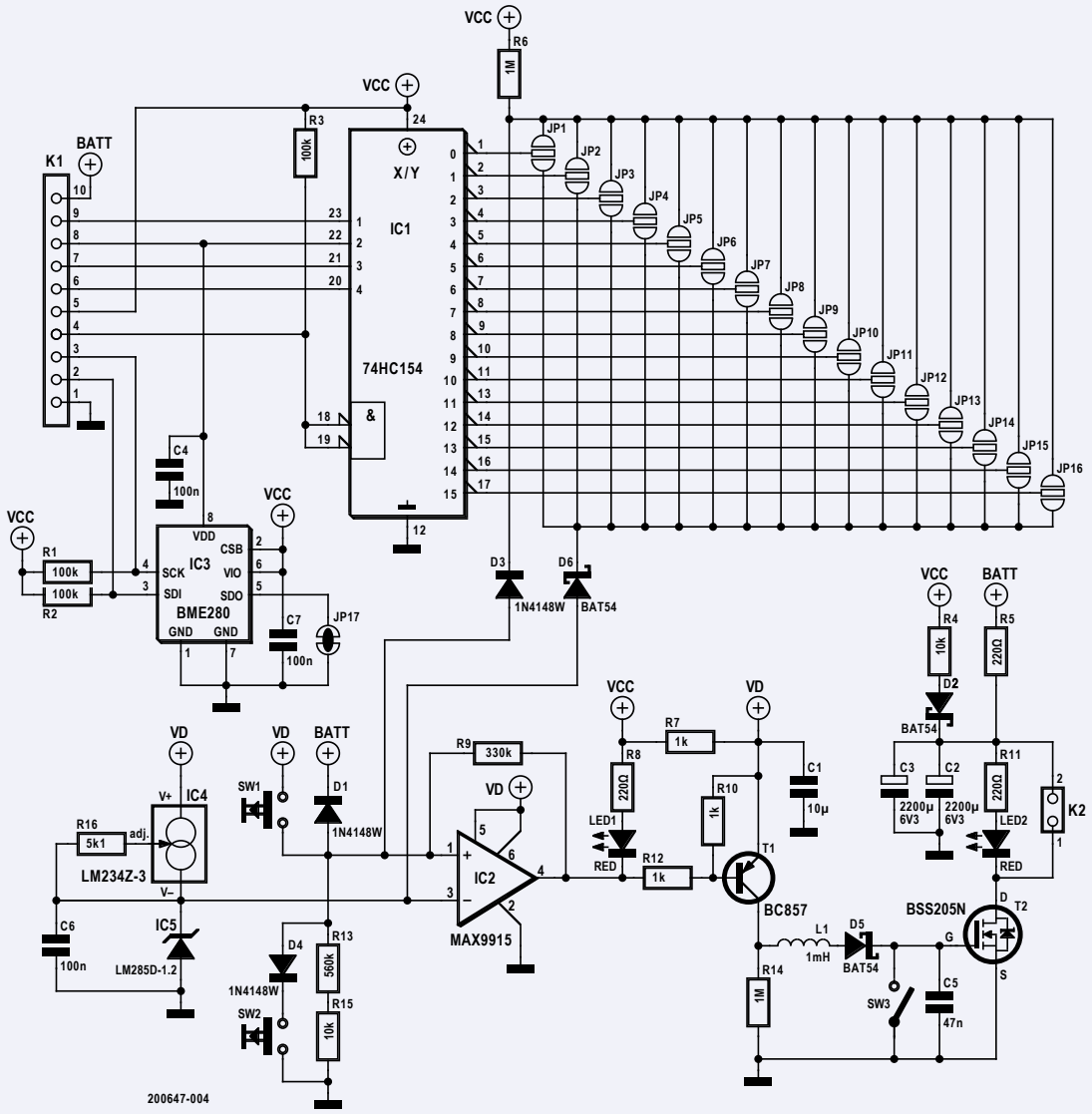


Figure 2. Le schéma de l'extension de mémoire analogique protégée par flash. La grande matrice sert de protection contre les chevaux de Troie.

Description du circuit

La **figure 2** donne le schéma de l'extension à mémoire analogique protégée par flash. Le connecteur K1 se branche sur le connecteur K3 prévu pour cela de la carte principale de la BTE [1]. À l'exception des broches d'alimentation, toutes les autres broches de K1 sont reliées au microcontrôleur de la BTE.

Lorsque l'effraction a été détectée par la BTE et que l'UC décide de mettre à zéro sa mémoire, il place une combinaison binaire sur les entrées de IC1, un décodeur binaire standard de 4 à 16, qui activera la sortie *Fire* (cathode de D3). Si, en revanche, la BTE a été déverrouillée avec succès, l'UC activera la commande *Block* (cathode de D6) à la place. Maintenant, la BTE peut être ouverte, l'interrupteur de sécurité SW3 peut être fermé et l'ampoule de flash peut être retirée de K2. Bob peut dérouler la bande de papier/cassette de l'ampoule de flash et vérifier le code de réponse spécial.

Les sorties qui activent les commandes *Fire* et *Block* sont sélectionnées par les cavaliers JP1 à JP16. (N'utilisez pas le même cavalier pour les deux commandes !) Avec IC1, ils constituent une protection contre un éventuel cheval de Troie spécifique à une application qui tenterait de désactiver la commande *Fire*. Avant chaque mission, les codes des commandes *Fire* et *Block* sont modifiés en sélectionnant différents cavaliers sur la carte (**fig. 3**) et en adaptant le microprogramme de l'UC en conséquence. Le cheval de Troie spécifique à l'application de Mallory ne peut pas fonctionner s'il ne peut pas reconnaître avec certitude le microprogramme ou les sorties. Un cheval de Troie à usage général peut copier les codes de la SRAM vers une mémoire flash secrète, mais il ne peut pas copier les codes d'un papier ou d'une bande magnétique enroulée autour de l'ampoule.

IC2 est câblé comme un comparateur avec une rétroaction positive pour réagir à une faible tension sur son entrée non-inverseuse. IC4 fournit un courant constant pour IC5, une référence de tension précise de 1,2 V à large plage de température, utilisée par IC2.

Lorsque la commande *Fire* est activée, l'entrée non-inverseuse est tirée vers le bas par la diode D3. La sortie de IC2 passe au niveau bas, permettant au transistor T1 de conduire. La charge stockée dans C1 circulera à travers L1 et D5, ce qui produit une tension sur C5 supérieure



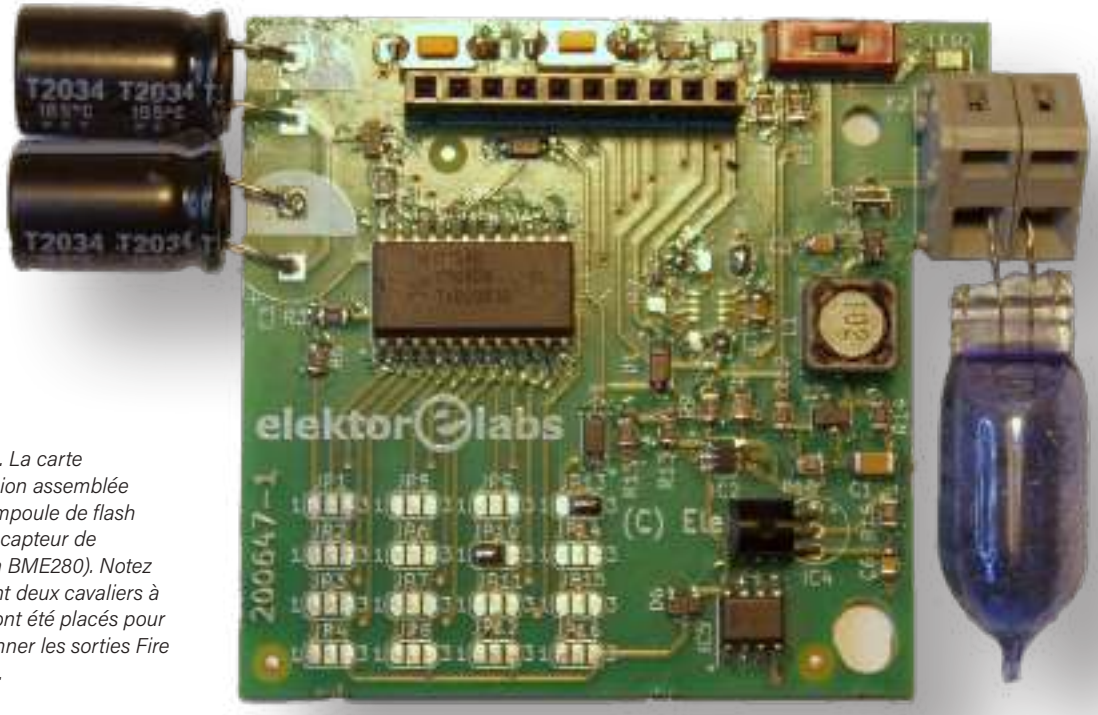


Figure 3. La carte d'extension assemblée avec l'ampoule de flash (sans le capteur de pression BME280). Notez comment deux cavaliers à souder ont été placés pour sélectionner les sorties Fire et Block.

Publicité

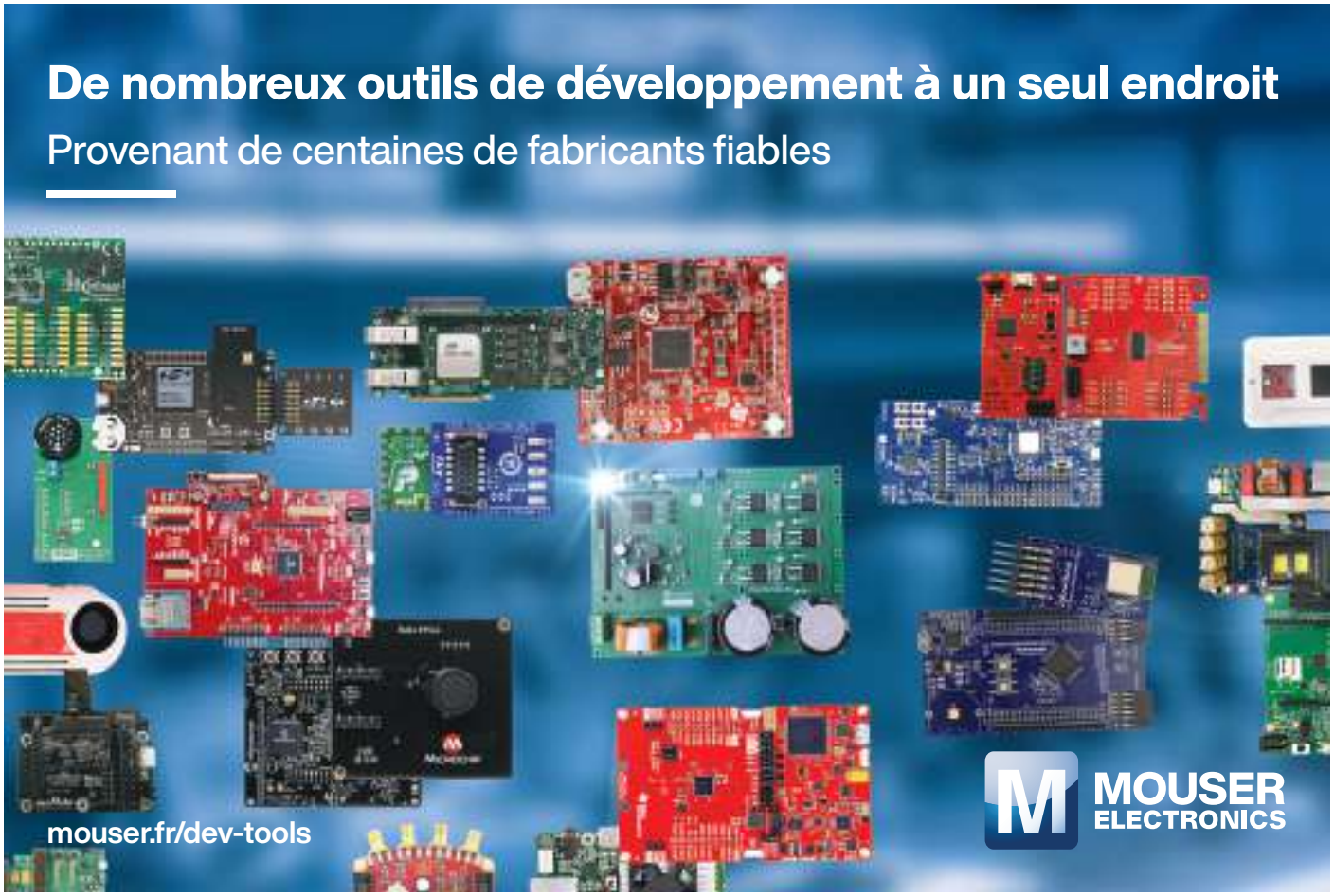




Figure 4. La carte d'extension avec ampoule de flash se branche en dessous de la carte principale de la BTE. Les deux boutons-poussoirs et son interrupteur sont accessibles sur le côté.

à V_d pour activer complètement le transistor T2. C5 maintient T2 allumé pendant au moins 10 à 20 ms, le temps nécessaire à l'allumage de l'ampoule connectée à K2. C2 et C3 stockent l'énergie nécessaire à une impulsion de courant de 1 A à 2 A pendant 10 ms pour enflammer le magnésium contenu dans l'ampoule.

Dans le cas de la commande *Block*, la diode D6 tirera l'entrée inverseuse de IC2 à 0,3 V, verrouillant effectivement la sortie de IC2 à l'état haut. Le circuit de mise à feu est maintenant désactivé.

Le bouton-poussoir SW1 est utilisé pour armer le circuit de mise à feu en chargeant C1 ; le bouton-poussoir SW2 sert à tester la mise à feu. SW3 est un interrupteur de sécurité, qui désactive la mise à feu en bloquant la grille de T2 à 0 V. Il peut être relâché une fois que tout est prêt.

LED1 et LED2 sont des témoins lumineux rouges. Si l'un d'eux ou les deux sont allumés, quelque chose n'est pas prêt, et vous ne devez pas connecter une ampoule de flash à K2 ou relâcher SW3.

Contrairement à la BTE, qui sort du mode économie d'énergie toutes les deux secondes, la carte d'extension est constamment alimentée afin de pouvoir réagir le plus rapidement possible aux courts-circuits ou aux chutes de tension provoqués par les attaques par perçage (voir ci-dessous). C'est pourquoi on utilise des circuits intégrés à faible puissance et à faible tension (jusqu'à 2,0 V). IC1, par exemple, *doit* être de type HC. Ils nécessitent un courant d'alimentation supplémentaire total de 30 µA seulement. Les trois condensateurs (C1, C2 et C3) conservent suffisamment d'énergie pour déclencher l'ampoule de flash même sans commande de l'UC en cas de court-circuit ou de faible tension sur V_{CC} ou V_{bat}.

Utilisation

Branchez la carte d'extension sur la carte principale (**fig. 4**) et mettez le système complet sous tension. Appuyez sur SW1 pour armer le circuit de mise à feu. Lorsque LED1 et LED2 s'éteignent, on peut connecter à K2 l'ampoule avec le message secret enroulé autour et relâcher SW3. On place ensuite l'ensemble dans une boîte (**fig. 5**) que l'on peut verrouiller en suivant la procédure expliquée dans [1].

Attaques par perçage

Maintenant que se passe-t-il si Eve essaie de désactiver le circuit de mise à feu de l'ampoule en perçant la BTE – par ex. en court-circuitant C1, C2 ou C3 à la masse avec une mèche de perceuse ? C'est pour cela que le circuit comporte également un capteur de pression barométrique (IC3, un BME-280 qui mesure également la température et l'humidité relative). La BTE lit la pression barométrique (absolue) de IC3 toutes les deux secondes via un bus I2C (réalisé par logiciel) sur les broches 2 et 3 de K1.

Lors de l'armement du boîtier, on doit chasser un peu d'air avec une pompe à vide. La pression de l'air mesurée par IC3 est lue par l'UC avec correction de température. Le rapport entre la pression de l'air et la température est ce qui compte pour déclencher la mise à zéro. Comme le volume de la BTE est constant, on n'autorise que des variations de pression proportionnelles à la température dans la plage de −20 °C à +60 °C (en fait 253 K à 333 K). Cela permettra de détecter une attaque par perçage dans les deux secondes (dès que le MCU se réveille du mode économie d'énergie).

Faites attention lorsque vous aspirez l'air de la BTE – il faut que Bob

puisse l'ouvrir sans avoir besoin d'un pied de biche. La force requise par Bob pour ouvrir le couvercle peut être calculée avec : $F = A \times \Delta p$, où A est la surface du couvercle et Δp la différence de pression.

Pas de place pour les logiciels libres

La mémoire protégée par une ampoule au magnésium résout efficacement le problème des chevaux de Troie matériels à usage général. Eve peut lire ce qu'elle veut dans la SRAM ou dans sa mémoire flash secrète, mais cela ne sert à rien si le code de réponse spécial est brûlé en même temps que le papier ou la bande magnétique. Cependant, un cheval de Troie spécifique à une application peut toujours empêcher l'allumage de la lampe flash, et on doit aussi se prémunir de cette menace.

Jusqu'à présent, nous avons recommandé l'utilisation de logiciels à code source ouvert chaque fois que cela était possible. Bien que les logiciels libres soient bien pour de nombreuses raisons, nous voici devant une exception.

Un cheval de Troie spécifique à une application est beaucoup plus compliqué qu'un cheval de Troie à usage général. Il ne fonctionne que s'il peut reconnaître le microprogramme avec certitude. Une façon pour Alice de contrer cette menace est d'écrire elle-même le micrologiciel. Elle le fera sur son ordinateur dédié qui n'est jamais connecté à l'internet et qu'elle utilisera aussi pour programmer l'UC. Si Mallory n'a aucune connaissance du microprogramme, son cheval de Troie ne saura pas comment le reconnaître.

La distribution des rôles

Dans les ouvrages de cryptographie, la communication a généralement lieu entre Alice (A) et Bob (B). Eve, l'espionne, essaie d'écouter passivement sans interférer, tandis que Mallory, le malveillant, n'hésite pas à modifier, substituer ou rejouer d'anciens messages.

Quelques précautions de sécurité

Les ampoules au magnésium fonctionnent avec des tensions très faibles – 1,0 V seulement peut suffire à les allumer. Cette mise en garde concerne le magnésium à l'intérieur de l'ampoule, qui brûle à des températures supérieures à 3.000 °C. Parfois, l'ampoule en verre se brise et vole en éclats en raison de ce stress thermique extrême. Les éclats de ce type de verre ne sont pas du tout tranchants, il n'y a donc aucun risque de coupure. C'est pourquoi, s'il est présent, on peut retirer le revêtement en plastique de l'ampoule de verre afin d'améliorer le transfert de chaleur et d'assurer la destruction du support qui l'entoure.

Des morceaux de métal fondu à ces températures extrêmes peuvent provoquer des brûlures. En revanche, l'allumage de l'ampoule à l'intérieur de la BTE est parfaitement sûr – de nombreux appareils électroniques courants libèrent beaucoup plus d'énergie lorsqu'ils tombent en panne en se consumant pendant leur fonctionnement (certes rarement à des températures aussi élevées). Le déclenchement d'une ampoule de flash au Mg à proximité de vos yeux non protégés peut être très désagréable et peut même provoquer une cécité temporaire (c'est ainsi que fonctionnent les grenades FlashBang). Par conséquent, enroulez le papier ou la bande autour de l'ampoule avant de la connecter à K2 – cela réduira la luminosité.

Astuce pour l'utilisation du papier

Pour utiliser du papier comme support d'écriture du code secret, il vaut mieux prendre du papier à rouler pour cigarettes fin. Il est aussi recommandé de le tremper dans une solution d'eau saturée de permanganate de potassium. Cet oxydant puissant abaisse la température d'ignition du papier. Laissez sécher le papier avant d'y écrire le message secret.



Figure 5. La boîte inviolable équipée de son flash de protection prête à être envoyée à Bob.

LISTE DES COMPOSANTS



Résistances (5%, 0805, 1/8 W)

R1, R2, R3 = 100 kΩ
R4, R15 = 10 kΩ
R5, R8, R11 = 220 Ω
R6, R14 = 1 MΩ
R7, R10, R12 = 1 kΩ
R9 = 330 kΩ
R13 = 560 kΩ
R16 = 5,1 kΩ (5,6 kΩ // 56 kΩ)

Condensateurs

C1 = 10 µF (1206)
C2, C3 = 2200 µF 6V3, pas de 5 mm
C4, C6, C7 = 100 nF (0805)
C5 = 47 nF (0805)

Inducteurs

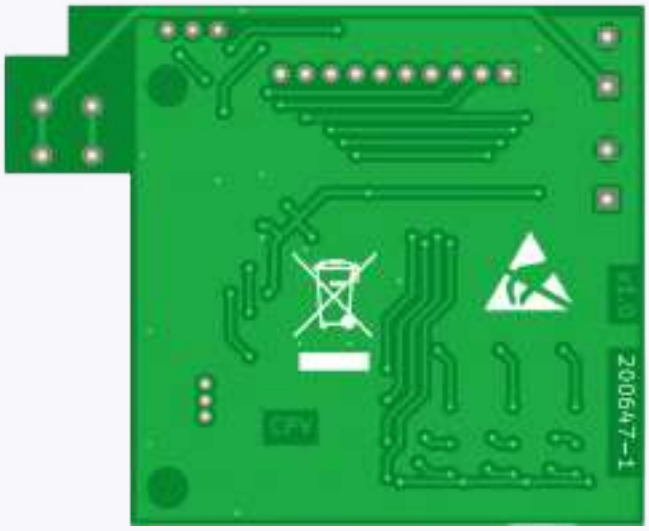
L1 = 1 mH, 10 Ω

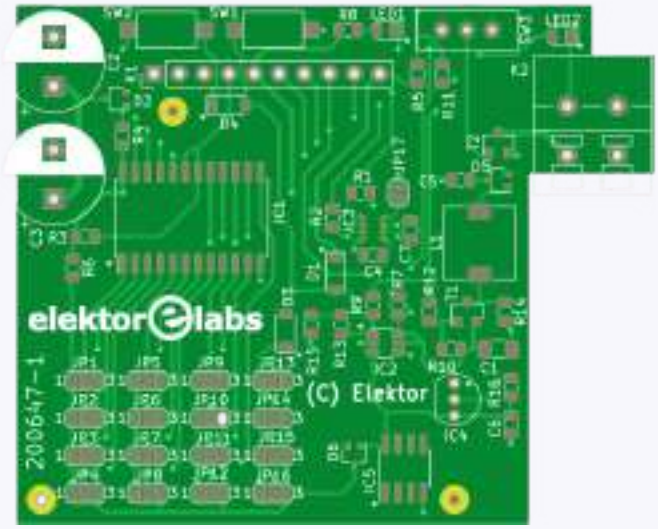
Semi-conducteurs

D1, D3, D4 = 1N4148W
D2, D5, D6 = BAT54
IC1 = 74HC154D
IC2 = MAX9915
IC3 = BME280
IC4 = LM234Z-3
IC5 = LM285D-1.2
LED1, LED2 = LED, rouge, 0805
T1 = BC857
T2 = BSS205N

Divers

K1 = barrette femelle à 10 voies, pas de 2,54 mm
K2 = connecteur à vis à 2 voies, pas de 5,08 mm, Wago 236-402
SW1, SW2 = bouton-poussoir, RS282G05A3
SW3 = interrupteur à glissière, SS12SDP2





LIENS

[1] Luka Matić, « boîte inviolable protégée par un témoin d'effraction », Elektor, 05-06/2020 : www.elektormagazine.fr/180445-02
[2] Implanter un Trojan matériel dans une puce de silicium : www.schneier.com/blog/archives/2018/03/adding_backdoor.html
[3] Rémanence des données en SRAM à basse température : https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-536.pdf
[4] Effacement des données des disques flash : https://www.schneier.com/blog/archives/2011/03/erasing_data_fr.html
[5] Rémanence des données en mémoire RAM : https://www.usenix.org/legacy/events/sec01/full_papers/gutmann/gutmann.pdf
[6] Meggaflash, fabricant d'ampoule de flash au magnésium : http://www.meggaflash.com/
[7] Ce projet sur Elektor Labs : www.elektormagazine.fr/labs/magnesium-bulb-analogue-memory-add-on-to-the-tamper-evident-box
[8] Boîte à témoin d'effraction sur Elektor Labs : www.elektormagazine.fr/labs/tamper-evident-paper-mail-box-for-secure-distribution-of-one-time-pads
[9] Entretien avec Luka Matić sur Elektor TV : www.elektormagazine.fr/news/fr-secure-communications-an-interview-with-luka-matic

Améliorations supplémentaires

Outre l'abandon des logiciels à code source ouvert, nous pouvons également faire une petite entorse à nos principes de matériel ouvert. Alice peut améliorer la sécurité en modifiant également les connexions entre K1 et IC1 (par ex. en introduisant des cavaliers de soudure comme JP1-JP16). Créer sa propre variante du circuit imprimé ne peut qu'améliorer la robustesse contre les chevaux de Troie matériels de Mallory et les attaques par perçage d'Eve.
Même si l'extension de la BTE est protégée contre les courts-circuits dus à une attaque par perçage, il faut noter qu'Eve dispose d'une fenêtre théorique de 2 s pour en effectuer une (jusqu'à ce que l'UC se réveille et détecte un changement de pression de l'air). Cela signifie qu'il faudrait renforcer mécaniquement contre le perçage les pistes vulnérables du circuit imprimé (V_d étant la plus importante) avec par ex. de fines barres d'alliages très durs insérées à l'intérieur d'un corps en métal mou (laiton ou aluminium) pour casser une mèche ou la faire dévier – comme ce qui est utilisé dans les serrures de haute sécurité.

Amélioration de la sécurité

L'extension simple et peu coûteuse décrite ici améliore considérablement la sécurité du boîtier inviolable, et le prémunit contre de nombreuses attaques à haute technologie. Tous les types de cassettes magnétiques essayés avec ce système ont été brûlés par une ampoule au magnésium AG-1. Tous les tests effectués avec du papier ont également été concluants.
Tout ceci reste abordable et réalisable par nos amis Alice et Bob avec leur budget limité. Appliquée et combinée avec soin, une technologie analogique simple et pas chère peut venir à bout de la coûteuse high-tech numérique. Cela peut sembler contre-intuitif, mais il ne faut pas toujours se fier à son intuition. Comme pour tout dispositif sécurisé, plusieurs experts indépendants devront explorer de nouveaux moyens d'attaque et de défense pour évaluer pleinement la sécurité globale.
Tous les fichiers de conception de ce projet et bien d'autres encore se trouvent sur [7], [8] et [1]. Regarder [9] peut vous aider à développer votre aptitude à la paranoïa. Bon espionnage !

200647-04

Contributeurs

Idée, conception et texte : **Luka Matić**
Conception et modification des circuits imprimés : **Clemens Valens**
Mise en page : **Harmen Heida**
Traduction : **Denis Lafourcade**

Des questions, des commentaires ?

Envoyez un courriel au rédacteur (clemens.valens@elektor.com).

Publicité



kit du LCR-mètre 2 MHz d'Elektor



Tous les composants requis pour l'assemblage et l'étalonnage de l'appareil sont inclus :

- 2 circuits imprimés assemblés (boutons, sélecteur rotatif et LCD compris)
- 4 embases BNC (pas implantées), 4 cavaliers
- pinces Kelvin avec câbles et 4 BNC
- câble de liaison mini-USB / USB A
- câble en nappe à 24 voies (15 cm)
- bouton en alu pour sélecteur rotatif
- outil de réglage pour les ajustables

Coffret alu Hammond, usiné et sérigraphié + visserie

Un soin particulier a été accordé au confort d'utilisation et à l'étalonnage

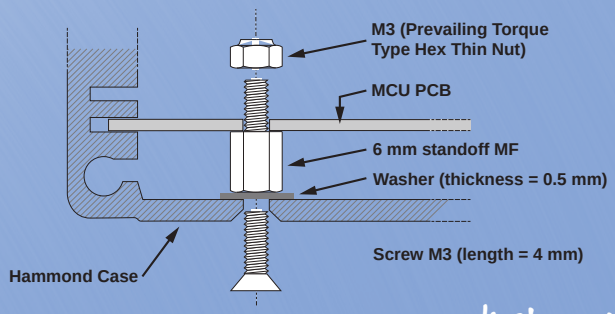
Informations complètes :

www.elektor.fr/19883

elektormagazine.com/posters

ENFIN DISPONIBLE !

Un sélecteur rotatif facilite la navigation (menus) et le choix de paramètres de mesure comme la fréquence



En coopération avec :

HAMMOND MANUFACTURING®

Instructions d'assemblage précises et illustrées

Fonctions étendues :

- pont de mesure d'impédance automatique
- mesure la résistance, la capacité et l'inductance de composants sous une impédance de 10 mΩ à 100 MΩ
- fréquence de test comprise entre 50 Hz et 2 MHz
- quatre tensions de test possibles (0.1, 0.2, 0.5 et 1 Vrms)
- polarisation CC possible jusqu'à 5 V pour les condensateurs et 50 mA pour les inductances
- deux configurations possibles :
  - mode autonome : carte principale + extension d'affichage
  - carte principale (sans affichage) connectée par USB à un ordinateur avec le programme adéquat (sous Windows, Linux ou MacOS)

Manuel d'assemblage et d'étalonnage imprimé en couleurs (32 pages)



Pour en vérifier le fonctionnement, les deux cartes de chaque kit ont été soumises à un protocole de test conçu par l'auteur Jean-Jacques Aubry

elektor JUMPSTARTER
from project to product

elektor
créer > partager > vendre

balises Bluetooth : la pratique

Géolocalisation *intra muros*

Veikko Krypczyk

Pour la géolocalisation à l'extérieur, on fait appel généralement au GPS ou au téléphone portable, mais que fait-on pour localiser des personnes et des objets à l'intérieur de bâtiments où ne pénètre aucun signal de satellite et sans connexion de téléphonie mobile ? On peut utiliser des balises *Bluetooth Low Energy*. Voici comment cela fonctionne !



Balise sphérique de la pointe de Cuxhaven érigée en 1718 (© A.Savin, WikiCommons, https://de.wikipedia.org/wiki/Kugelbake#/media/Datei:Cuxhaven_07-2016_photo23_Kugelbake.jpg)

Il y a des siècles, la balise (***beacon*** en anglais) était un repère visible, naturel ou artificiel, qui indiquait p. ex. aux marins la route à suivre dans les eaux côtières dangereuses. Avec l'avènement de la communication radio dans la navigation à la fin du XIXe siècle, ces balises fixes, généralement disposées en chaîne, ont été équipées d'émetteurs (balises maritimes). Chacune émettait à intervalles réguliers un code Morse individuel d'identification tous azimuts, de sorte qu'un navire pouvait caboter de balise en balise.

Après les marins vinrent les aviateurs : dès les premières tentatives dans les années 1920 d'utiliser la radio pour l'approche à l'atterrissage sans visibilité, l'ingénieur Ernst Ludwig Kramar a conçu un système d'atterrissage aux instruments dans le bas de la gamme VHF, appelé Faisceau Lorentz [1], installé dans des aéroports du monde entier au début des années 1930. Avec l'avènement de la navigation GPS, les radiobalises ont perdu de leur importance dans la navigation maritime (à l'exception des balises de détresse). Dans l'aviation, différentes radiobalises sont toujours en service.

Au début du 21e siècle, avec les téléphones portables, une nouvelle forme de balise est apparue, analogue aux balises marines originales : les balises multiples émettent des identifiants individuels reçus par un appareil mobile. En fonction de la puissance des signaux d'identification reçus, un logiciel approprié détermine la position du téléphone (de la tablette ou d'une autre forme d'électronique portable) par rapport aux balises reçues. Bien entendu, on n'utilise pas de transmission FM pour cela, mais une transmission à *faible consommation d'énergie connue* sous le nom *Bluetooth Low Energy (BLE)*. Cela vous dit-il quelque chose ? Mais oui, c'est le principe des applications d'alerte Corona !

Cet article présente la technique qui réunit matériel (balises) et logiciel (configuration et application). Lors de nos premières expériences, nous nous familiariserons avec le sujet et créerons ainsi la base de nos propres applications.

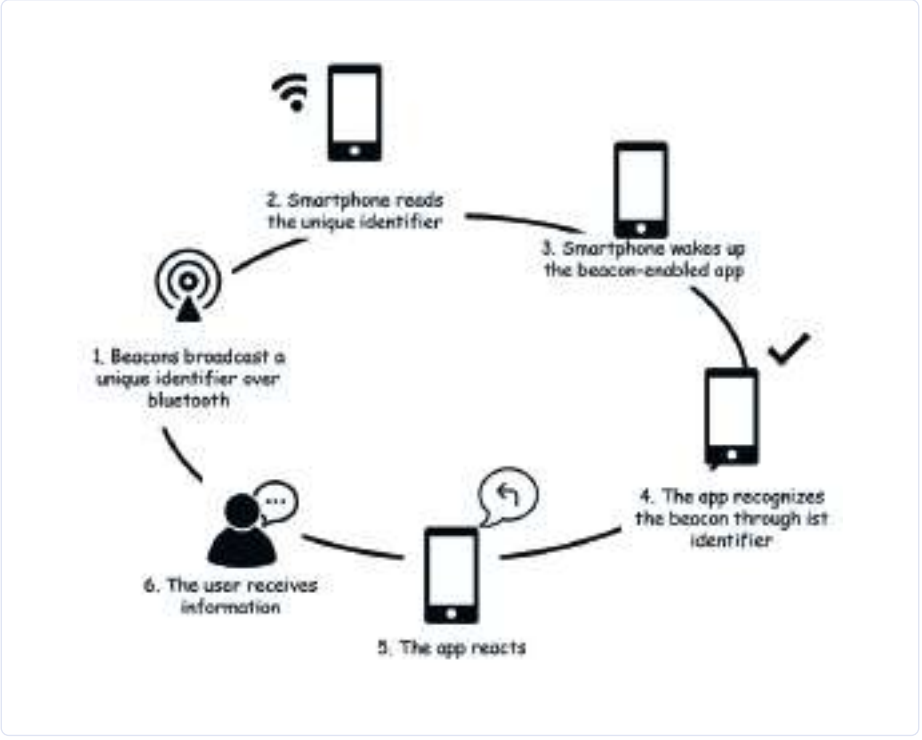


Figure 1. Cas d'utilisation de balises.

Innombrables applications

Les mesures de distance et la localisation à l'intérieur des bâtiments sont des applications typiques des balises. En combinant plusieurs balises, une grande variété d'utilisations est concevable dans la vie privée ainsi que dans les affaires : le *marketing* géolocalisé, les *achats* mobiles et diverses applications de réalité mixte et augmentée. Des informations ou des publicités spécifiques peuvent être transmises à des utilisateurs quand ils se trouvent à un endroit précis dans un bâtiment.

- *Achats mobiles* : les informations sur les produits, les offres en cours, les codes de réduction, etc. pour des articles individuels peuvent être affichés sur le téléphone tactile du client. Aux utilisateurs intéressés par des produits spécifiques d'une certaine catégorie, on peut proposer des produits complémentaires. Des clients peuvent ainsi être guidés à travers une gamme de produits. Les balises permettent aux détaillants de contacter le téléphone tactile des clients en fonction de leur position. Les analyses de marché exploitent des données de progression des clients dans le magasin pour en optimiser le placement sur des surfaces étendues. C'est ce qu'illustre le scénario de la **figure** 1. La balise envoie un signal par BLE (étape 1). Ce signal est reçu par le téléphone tactile (étape 2). L'application appropriée est alors réveillée en arrière-plan (étape 3). L'application reconnaît et enregistre la balise en fonction de son ID (étape 4), demande des informations complémentaires à un serveur en coulisses (étape 5) pour les présenter sur l'écran du téléphone tactile (étape 6).
- *Navigation* : les balises sont utilisables pour la navigation dans les bâtiments, p. ex. dans les musées, les gares, les aéroports ou les stades. Elles sont placées à différents endroits du bâtiment, les signaux sont envoyés aux appareils mobiles par Bluetooth. De cette façon, la position de chaque appareil mobile et de son utilisateur peut être déterminée en permanence et transmise au système de navigation intérieure. Pour interpréter ces signaux, il faut une application sur le téléphone tactile. Le nombre de balises dans le bâtiment dépend de la superficie à couvrir et de la précision souhaitée.
- *Améliorer l'accessibilité* : des instructions vocales et visuelles ciblées, p. ex. sur un écran, peuvent faciliter la circulation de personnes handicapées dans des lieux complexes ou très fréquentés.
- *Logistique* : en plaçant des balises le long de la chaîne d'approvisionnement ou de logistique, il est possible de combiner les balises avec d'autres techniques modernes telles que la RFID ou la gestion de flotte en temps réel, afin d'optimiser la logistique.

Un avantage particulier de ces balises est leur faible consommation d'énergie, qui permet un fonctionnement de longue durée sans alimentation électrique externe. L'utilisation des données résultantes est toutefois un sujet sensible pour des raisons de protection de la vie privée, car elles permettent d'analyser le comportement de l'utilisateur (position, distances de marche, temps de résidence). Les utilisateurs doivent consentir à l'utilisation de ces données par l'application qui assure le suivi sur l'appareil mobile. Les balises elles-mêmes n'envoient que les signaux, elles ne sont généralement pas des récepteurs (voir

BALISES ET VIE PRIVÉE

Les balises n'émettent que des **informations sans référence personnelle** et ne peuvent ni recevoir ni traiter elles-mêmes des données. À cet égard, l'envoi de signaux ne pose aucun problème de protection des données. Une référence personnelle ne serait concevable que si une balise est attribuée à une personne spécifique (qui porte une balise à des fins d'identification, par exemple), à condition que d'autres informations de l'utilisateur soient liées à cette donnée.

Ces informations sont reçues et traitées par l'**application installée** sur le téléphone. Comme cette application est ordinairement connectée à un **compte d'utilisateur**, on aura ici une référence personnelle. Le programme peut p. ex. traiter des informations telles que le lieu et la durée de présence d'un client particulier dans un rayon, ou même analyser son comportement d'achat. En raison de cette référence personnelle, ces données ne peuvent être traitées qu'avec le **consentement de la** personne concernée. Étant donné que la personne concernée doit installer activement l'application nécessaire sur son téléphone, elle doit être informée dans une **déclaration de protection des données** avant l'utilisation conformément à la réglementation sur la protection des données. Dans ce contexte, l'utilisateur doit notamment être informé de la **finalité de ce démarchage commercial** (art. 13 RGPD). La responsabilité de la protection des données incombe donc essentiellement aux fournisseurs d'applications qui traitent les informations reçues des balises. Si le fournisseur informe suffisamment ses utilisateurs et obtient leur consentement conformément aux lois sur la protection des données, rien ne devrait s'opposer à l'utilisation de balises.

(selon David Oberbeck, du bureau allemand de protection des données [19])

également l'encadré « Balises et vie privée »).

Faible énergie Bluetooth

C'est Nokia qui le premier a utilisé BLE en 2006 pour la transmission unidirectionnelle de données. Les appareils BLE consomment beaucoup moins que les appareils Bluetooth traditionnels. La batterie d'une balise dure en moyenne plusieurs années. La production de modules BLE est également beaucoup moins chère que celle des appareils Bluetooth classiques. En 2013, Apple a mis la technologie *Beacon* à la disposition du grand public. Fabricants et concepteurs désireux de l'utiliser devaient d'abord en demander une licence à Apple avant de pouvoir utiliser le *kit de développement logiciel* (SDK) et les bibliothèques de programmes. Il existe aujourd'hui de nombreuses entreprises spécialisées dans la production de dispositifs de signalisation par balises. Il existe trois types de balises avec différents formats de données, qui se distinguent par le format des données :

- **iBeacon** : défini par Apple. Voir [2] pour une introduction au sujet et la spécification complète.
- **AltBeacon** : ouvert, sans frais de licence. [3].
- **Eddystone** : ouvert, défini par Google [4].

Comment fonctionnent les balises ? Les balises compatibles BLE envoient à intervalle fixe de courts messages en mode Bluetooth à faible consommation d'énergie, qui peuvent être reçus par les appareils compatibles Bluetooth à proximité immédiate. Il s'agit d'un *processus dit de diffusion*. De même, tout téléphone, tablette ou montre tactiles équipés du BLE peut être considéré comme balise potentielle. Il existe toutefois une différence importante : un téléphone tactile ou une tablette peut envoyer des signaux BLE et en recevoir. Les balises, en revanche, se contentent d'envoyer les signaux à courts intervalles réguliers pour créer une région de signal. Un logiciel correspondant (appli) doit être installé sur le téléphone tactile afin d'utiliser les données de localisation résultantes.

Spécifications

Quelles sont les informations émises par une balise ? Il faut ici distinguer les types de balises *iBeacon*, *AltBeacon* et *Eddystone* (**fig. 2**).

Commençons par la spécification des données des *iBeacons* (fig. 2a). Ici, les valeurs individuelles signifient [5] :

- *Préfixe* : il contient les données hexadécimales : *0x 020106 1AFF 004C 02 15* où *0x 020106* définit l'information générale selon laquelle l'émetteur est incompatible avec le haut débit et se contente de la diffusion (envoi) sans connexion (bidirectionnelle). *0x1AFF* indique que les données suivantes font 26 octets et sont spécifiques au fabricant. *0x004C* est l'ID du signal Bluetooth d'Apple. *0x02* est un ID secondaire et *0x15* définit la longueur restante, soit 21 octets (16+2+2+1).
- *UUID (Universally Unique Identifier) :* cette valeur de 16 octets identifie de manière unique les données envoyées à l'*iBeacon* dans les systèmes à composants multiples.
- *Major* : la valeur *major* compte 2 octets. Elle définit la région du signal.
- *Minor* : la valeur *minor* compte également 2 octets et détermine une sous-région.
- *TX Power* : indique la puissance poten-

Tableau 1 : Les termes « majeur » et « mineur » sont utilisés pour attribuer une région et une sous-région à une balise [14].

		Taille	San Francisco	Paris	Londres
UUID		16 octets	D9B9EC1F-3925-43D0-80A9-1E39D4CEA95C		
Major		2 octets	1	2	3
Minor	**Habits**	2 octets	10	10	10
	Maison		20	20	20
	Électro		30	30	30

tielle du signal. Elle est utilisée par l'application pour déterminer à quelle distance se trouve la balise du téléphone. Veuillez noter que la valeur de *TxPower* doit être calibrée par l'utilisateur pour être précise.

Le **tableau 1** présente un exemple de cartographie des données *majeures* et *mineures* pour la navigation intérieure dans un grand magasin.

Venons-en maintenant à la spécification du format des données pour les *types AltBeacon* (fig. 2b). La spécification *AltBeacon* compte 28 octets, dont 26 personnalisables par l'utilisateur. Les deux premiers octets sont déterminés par la pile BLE. *ADV Length* et *ADV Type* précisent la longueur du paquet de données et le type. Toutes les autres valeurs peuvent être configurées.
Enfin, les balises de type *Eddystone* peuvent être définies dans différentes variantes (fig. 2c), utilisables en combinaison ou individuellement. Cela signifie :

- *UUID* : un identifiant unique de 16 octets est émis.
- *EID* : cette trame envoie un identifiant de courte durée et chiffré pour augmenter la sécurité des données.
- *URL* : une URL est émise. Les utilisateurs peuvent utiliser les URL pour accéder au contenu sur l'internet.
- *TLM* : réservé à l'émission des données d'état de la télémétrie par la balise elle-même. Ces informations comprennent la tension de la batterie, le nombre de paquets de diffusion et la température de l'appareil.

Si vous développez des applications avec des balises, tenez compte du type de balise ou prévoyez une option de sélection par le logiciel.

Localisation par trilatération

L'application sur l'appareil mobile identifie les générateurs de signaux et calcule les distances entre les émetteurs et le récepteur en fonction de la puissance de transmission entrante. La puissance d'émission a un effet direct sur la portée du signal. Plus la puissance est forte, plus la portée est grande. L'*indicateur de force du signal reçu (RSSI)* décrit la force du signal de la balise sur le téléphone. La distance est déterminée en mettant en relation la puissance du signal reçu avec la valeur RSSI attendue (selon la fiche technique) à une distance d'un mètre.

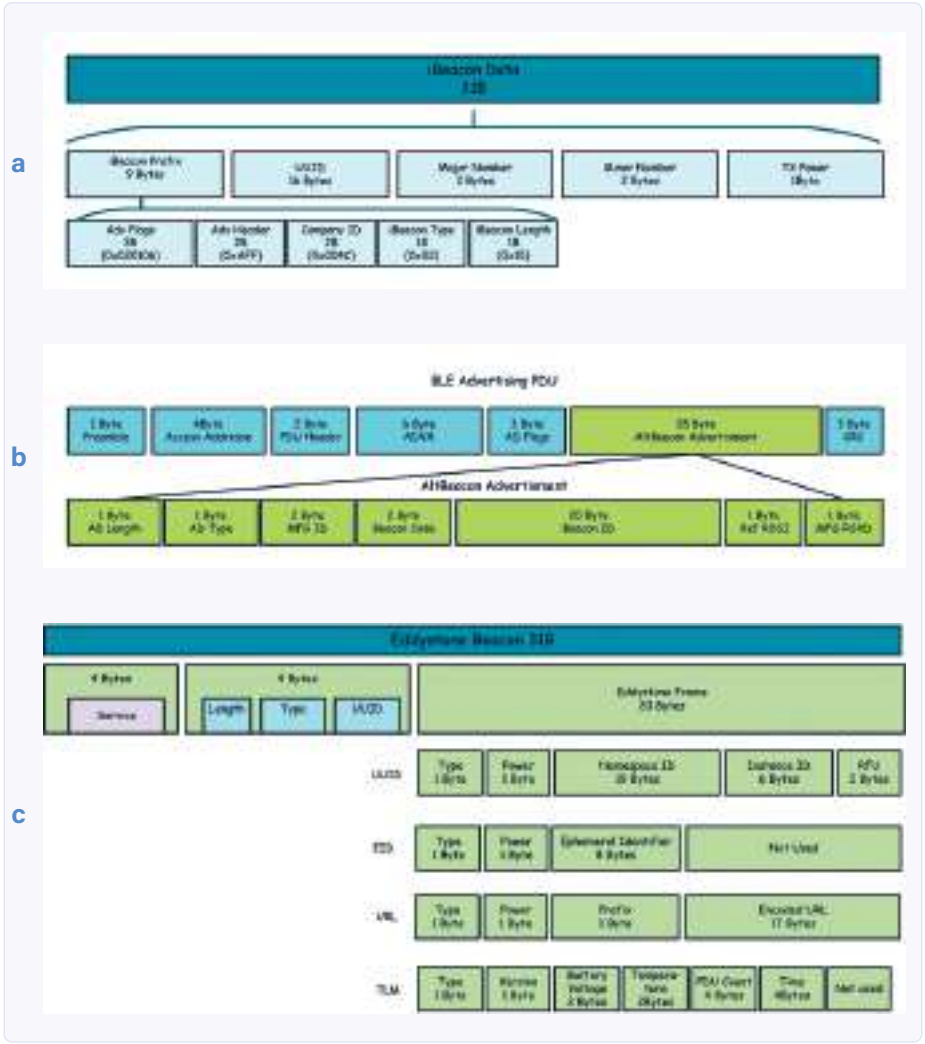


Figure 2. Spécifications des types de balises ([15], [16], [17], [18]).

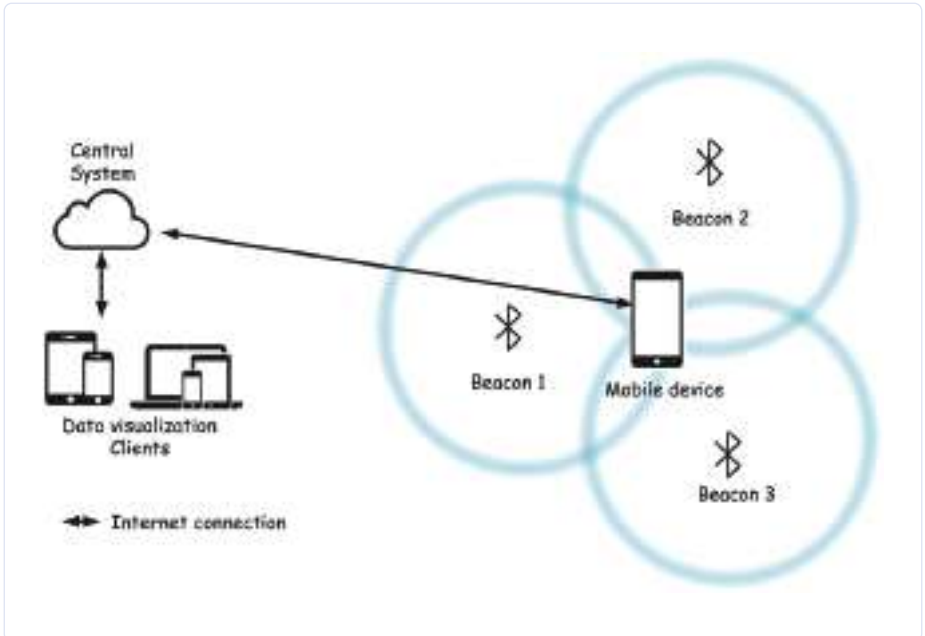


Figure 3. Le principe du suivi à l'intérieur à l'aide de balises.



Figure 4. Balise EMBC01 de EM Microelectronic.

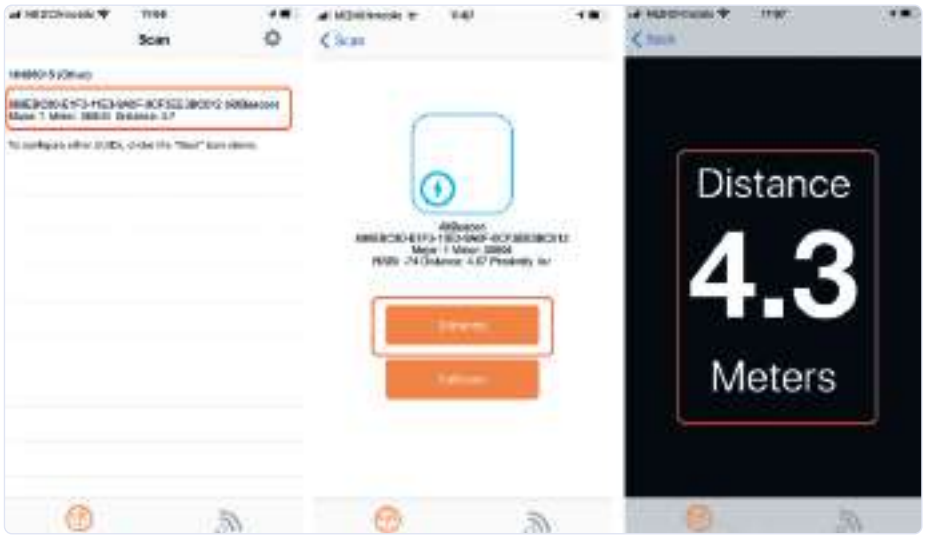


Figure 5. Affichage de la distance dans l'application EMBC Finder.

La mise à l'échelle des valeurs du RSSI est laissée aux fabricants de puces : les fiches techniques montrent comment la valeur du RSSI peut être convertie en puissance. Les valeurs sont exprimées en décibel milliwatt (dBm). La formule suivante décrit généralement la relation entre valeur du RSSI et distance [6], [7] :

$$RSSI = -(10\ n\ \log_{10} d + a)$$

où

n = valeur d'amortissement dépendant de l'environnement

a = puissance du signal reçu avec l'émetteur à un mètre de distance (puissance du signal de référence selon la fiche technique) et

d = distance entre émetteur et récepteur.

Pour un environnement intérieur, on suppose une valeur de n = 3. Il convient de noter que plus l'émetteur et le récepteur sont éloignés l'un de l'autre, plus la précision des mesures se dégrade et donc aussi la précision de la distance déterminée.

Pour pouvoir déterminer la position avec précision, au moins trois balises sont nécessaires (**fig. 3**). Comme pour le GPS, les distances entre unités sont mesurées selon le *principe*

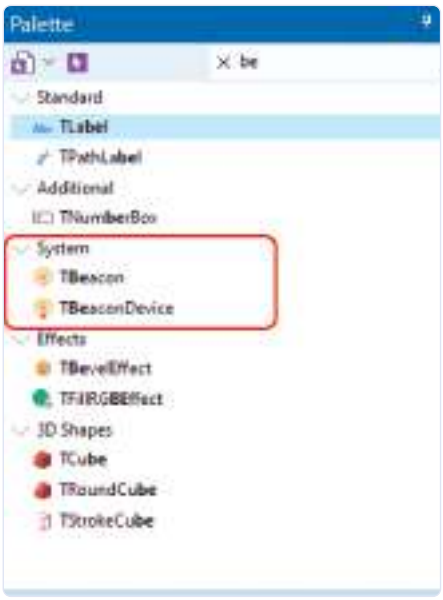


Figure 6. Grâce aux composants TBeacon et TBeaconDevice la complexité est bien encapsulée.

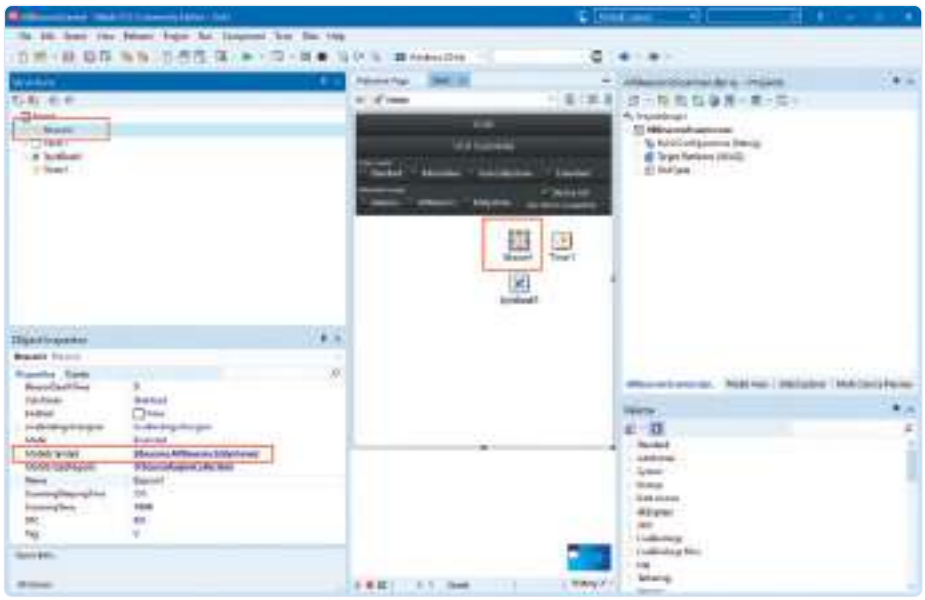


Figure 7. Configuration de la composante TBeacon.

de la trilatération. Cependant, l'émission de données par balises est conçue pour une courte portée. Une distinction est faite entre *immédiat* (moins de 50 cm), *proche (*jusqu'à 3 m), *loin (*plus de 10 m) et *inconnu.* Des balises puissantes peuvent émettre des signaux jusqu'à 450 m. La portée peut être réduite par des obstacles. Le géorepérage (flexible) ou *geofencing*, est modifiable par repositionnement de balises ; des zones sont définies par une *latitude, une longitude* et un *rayon.*

Première expérience

Après ces nécessaires considérations théoriques, voyons comment les balises fonctionnent dans le cadre d'un projet. Il nous faut une ou plusieurs balises (**fig. 4**), comme on en trouve sous de nombreuses formes [8] (à l'échelle européenne).

Le modèle *EMBC01* est une balise Bluetooth à faible consommation d'énergie, spécialement conçue pour la mesure de distance. Les données émises sont conformes à la norme Apple UUID, Major ID et Minor ID. L'EMBC01 est sous boîtier étanche avec pile bouton (remplaçable). Un bouton permet de passer d'un mode à l'autre :

- *veille* : mode de stockage avec une durée de vie typique de la batterie de plus de 7 ans.
- *ID à courte portée* : intervalle d'émission de 100 ms, portée de 15 m et autonomie typique de 1,5 mois.
- *ID à moyenne portée* : intervalle de 500 ms, portée de 30 m et durée de vie de la batterie de 7,5 mois.
- *ID longue portée* : intervalle de 1 s, portée typique de 75 m et durée de vie typique de la batterie de 12,5 mois.
- Un numéro de série et un code QR sont imprimés sur le boîtier de l'EMBC01. Ce type de balise est préprogrammé avec un *UUID*, un *Major ID* et un *Minor ID* définis de manière unique et ne peut pas être reprogrammé sans fil pour des raisons de sécurité. Pour faire la démonstration de la fonction, vous avez besoin d'une application sur votre téléphone, sous Android p. ex. l'application *EMBC-Finder* et sous iOS l'application *Locate*. Sélectionnez une balise d'envoi active dans l'application, puis vous pouvez déterminer sa distance (**fig. 5**).

Programmer soi-même une application

Pour mettre en œuvre vos propres applications de *Beacon*, vous devez également développer votre propre programme pour le dispositif BLE mobile, en bref une application. Vous pouvez utiliser l'édition communautaire gratuite de RAD Studio [9]. Grâce à l'environnement de développement intégré, des applications pour iOS et Android, mais aussi des applications pour les systèmes de bureau (Windows, macOS, Linux) peuvent être créées à partir d'une base de code source commune [10]. Grâce à des composants spéciaux pour la technique *Beacon*, le concepteur n'a pas à se soucier de l'évaluation des signaux et des protocoles BLE. Les composants sont configurés dans l'environnement de développement et sont ensuite immédiatement disponibles dans le code du programme. RAD Studio propose les deux composants suivants pour travailler avec la technologie des balises (**fig. 6**) [11] :

- *TBeacon* surveille la liste des régions de balises spécifiées et gère les informations sur les événements associés à la balise ; entre autres, les changements de distance sont détectés.
- *TBeaconDevice* peut utiliser un dispositif BLE comme balise normale. Vous pouvez configurer un *composant TBeaconDevice* dans une application fonctionnant sur un appareil BLE de sorte que celui-ci envoie des données de proximité. Les valeurs *UUID*, *Major*, *Minor* et *TxPower* peuvent être configurées.

Pour utiliser des émetteurs externes (balises), vous utilisez le composant *TBeacon*. Avec le composant *TBeaconDevice,* un téléphone tactile peut être converti en balise active. En quelques étapes, vous pouvez montrer comment cela fonctionne : sur GitHub, vous pouvez trouver des exemples de code source [12]. Clonez le dépôt complet sur votre ordinateur et passez au sous-répertoire *RADStudio10.3.3Demos/Object Pascal/Multi-Device Samples/Device Sensors and Services/Bluetooth/Beacons/ExtendedBeaconScanner/.* Ouvrez le projet dans RAD Studio.

La **fig. 7** illustre la procédure de configuration et de programmation. Le composant *TBeacon* est configuré ici pour reconnaître tous les types de balises (*iBeacon, AltBeacon et Eddystone*).

Le processus de numérisation est appelé par la procédure *StartScan.* Examinez le code source. Si des balises d'émission se trouvent à proximité, elles seront affichées sur l'interface utilisateur après le démarrage du



Figure 8. Les balises détectées sont affichées.

programme (**fig. 8**). Pour votre propre application, les données peuvent être évaluées sur le téléphone.

Scénarios de localisation complexes

Pour une tâche simple, par exemple pour déterminer si un appareil mobile se trouve à proximité d'une balise, aucune planification globale n'est nécessaire. C'est moins simple si vous voulez diviser un volume en zones (*BeaconFence*), p. ex. dans un magasin. Dans ce cas, les balises doivent être disposées pour surveiller les zones. Les utilisateurs peuvent alors être localisés grâce à cette disposition géographique, et l'entrée ou la sortie d'une zone peut être reconnue. C'est possible avec un éditeur graphique, tel que l'éditeur de cartes *BeaconFence* de *RAD Studio* [13]. Un tel éditeur de cartes peut être utilisé pour définir la disposition des pièces et l'agencement des balises, des zones et des chemins. Les actions peuvent être définies lorsqu'un appareil BLE entre dans une zone couverte par une ou plusieurs balises. Cette approche graphique facilite considérablement les analyses de rentabilité en vous évitant d'avoir à effectuer manuellement l'ensemble de la disposition et de la configuration des balises.

Conclusion et perspectives

Les balises sont basées sur la technologie BLE et permettent de déterminer la position des clients, des utilisateurs, des visiteurs ou des objets en mouvement dans une pièce. Les protocoles sont normalisés. Le recours à des éléments prêts à l'emploi pour l'évaluation des données reçues simplifie la programmation. Au début d'un tel projet, il y a une idée : *quel est le but du géorepérage intérieur ?* Puis il y a la conception : o*ù et comment les balises sont-elles disposées ?* Ensuite vient la mise en œuvre technique du logiciel étape par étape. Des bibliothèques prêtes à l'emploi facilitent la tâche du programmeur. Les scénarios intéressants ne manquent pas dans le triangle entre jeux, loisirs et applications sérieuses.

(200550-02)





Votre avis, s'il vous plaît
N'hésitez pas à poser vos questions ou envoyer vos commentaires sur cet article à l'adresse redaction@elektor.fr

Ont contribué à cet article
Auteur : **Veikko Krypczyk**
Rédaction : **Rolf Gerstendorf**
Maquette : **Giel Dols**
Traduction : **Ursel Keit**

LIENS

[1] **faisceau Lorenz** : https://fr.wikipedia.org/wiki/Faisceau_Lorenz

[2] **format des données des balises ibeacon** : https://developer.apple.com/ibeacon/

[3] **format de données ouvert AltBeacon** : https://altbeacon.org/

[4] **format Google Beacons** : https://github.com/google/eddystone/blob/master/protocol-specification.md

[5] **comparaison** : https://austinblackstoneengineering.com/ble-beacons-ibeacon-altbeacon-uribeacon-and-derivatives/

[6] **Anton Anders : Détermination de la position intérieure à l'aide de balises Bluetooth à faible énergie et géorepérage des piétons à l'estime** : https://cse.cs.ovgu.de/cse-wordpress/wp-content/uploads/2016/08/BA_Anton_Anders.pdf

[7] **puissance d'émission et distance** : https://bit.ly/3onjNZY

[8] **balises européennes** : www.beaconshop24.de/

[9] **RAD Studio** : www.embarcadero.com/products/rad-studio

[10] **Krypczyk, Veikko : Applications mobiles pour Android et iOS à partir d'un moule unique, Elektor 11-12/2020** : www.elektormagazine.fr/200265-03

[11] **utilisation des balises dans RAD Studio** : http://docwiki.embarcadero.com/RADStudio/Sydney/en/Using_Beacons

[12] **RAD Studio Demos** : https://github.com/Embarcadero/RADStudio10.3.3Demos

[13] **Map-Editor de RAD Studio** : http://docwiki.embarcadero.com/IoT/en/BeaconFence_Map_Editor

[14] **iBeacons pour les concepteurs** : https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf

[15] **Understanding the different types of BLE Beacons** : https://os.mbed.com/blog/entry/BLE-Beacons-URIBeacon-AltBeacons-iBeacon/

[16] **Spécifications de AltBeacon** : https://github.com/AltBeacon/spec

[17] **Projets IoT avec Bluetooth Low Energy** : www.oreilly.com/library/view/iot-projects-with/9781788399449/19495b89-8a0a-43f8-8dfd-955bdcc203db.xhtml

[18] **ixys L. Hernández-Rojas; Tiago M. Fernández-Caramés; Paula Fraga-Lamas & Carlos J. Escudero: Design and Practical Evaluation of a Family of Lightweight Protocols for Heterogeneous Sensing through BLE Beacons in IoT Telemetry Applications, in Sensors 201** : www.mdpi.com/1424-8220/18/1/57

[19] **Beacons et vie privée (en allemand)** : www.datenschutzkanzlei.de/fachbeitrag-beacons-gefahr-fuer-den-datenschutz/

C Programming on Raspberry Pi

Extrait : communiquer par Wi-Fi

Dogan Ibrahim (Royaume-Uni)

Le Raspberry Pi est traditionnellement programmé en Python. Bien que Python soit un langage très puissant, le langage C est probablement le plus utilisé par les programmeurs, et pas seulement parce qu'il est enseigné dans pratiquement toutes les écoles et universités techniques. Cet article donne un avant-goût du C sur Raspberry Pi, afin de vous aider à évaluer l'effort que vous devez fournir pour développer vos propres projets matériels. Vous pouvez travailler depuis votre laboratoire personnel, avec un livre et votre Raspberry Pi préféré à portée de main comme support de cours. Voyons ce qui se passe en C !



***Note de l'éditeur** : cet article est un extrait du livre de 376 pages, intitulé « C Programming on Raspberry Pi – Develop innovative hardware-based projects in C » formaté et légèrement modifié pour correspondre aux normes éditoriales et à la mise en page du magazine Elektor. Puisque cet article est extrait d'une publication plus vaste, certains termes peuvent faire référence à des passages du livre d'origine situés ailleurs. L'auteur et l'éditeur ont fait de leur mieux pour l'éviter et seront heureux de répondre aux questions – pour les contacter, voir l'encadré « Des questions, des commentaires ? ».*

UDP et TCP/IP

La communication par liaison Wi-Fi se déroule entre un client et un serveur. On utilise des *sockets* pour envoyer et recevoir des paquets de données. Le dispositif côté serveur attend généralement une connexion des clients et une fois celle-ci établie, la communication bidirectionnelle peut commencer. On utilise principalement deux protocoles pour envoyer et recevoir des paquets de données sur une liaison Wi-Fi : UDP et TCP. TCP est un protocole qui nécessite l'établissement d'une connexion, ce qui garantit la livraison des paquets. Les paquets reçoivent des numéros de séquence et la réception de tous les paquets fait l'objet d'un accusé de réception. La numérotation sert à remettre les paquets dans l'ordre s'ils ont suivi des chemins différents. À cause du système d'acquittement, TCP est généralement lent mais fiable, car il garantit la livraison des paquets. UDP, quant à lui, est un protocole sans connexion préalable. Les paquets n'ont pas de numéro de séquence et, par conséquent, il n'y a aucune certitude qu'ils arriveront à destination. UDP est moins surchargé que TCP et il est donc plus rapide. Le **tableau 1** énumère quelques différences entre les protocoles TCP et UDP.

Tableau 1. Communications par paquets TCP et UDP.

TCP	UDP
Les paquets ont des numéros de séquence et la livraison de chaque paquet fait l'objet d'un accusé de réception.	Il n'y a pas d'accusé de réception
Lent	Rapide
Pas de perte de paquets	Des paquets peuvent être perdus
Surcharge importante	Surcharge faible
Nécessite plus de ressources	Nécessite moins de ressources
Établissement d'une connexion	Sans connexion
Plus difficile à programmer	Plus facile à programmer
Exemples : HTTP, HTTPS, FTP	Exemples : DNS, DHCP, jeux vidéo

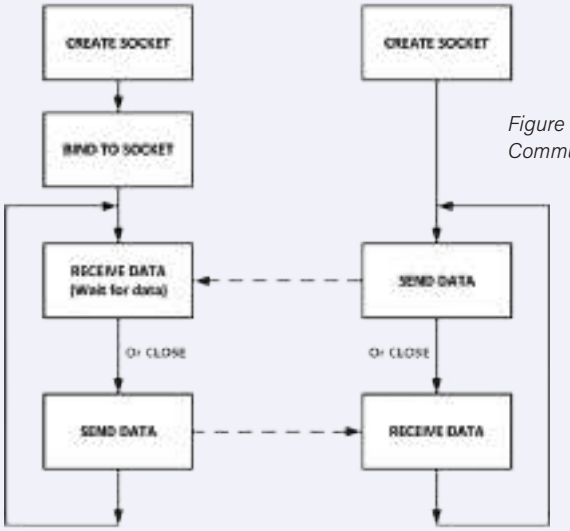


Figure 1. Communication UDP.



Figure 2. Schéma fonctionnel du projet.

Communication UDP

La **figure 1** montre la communication UDP sur une liaison Wi-Fi.

Serveur

1. Créer un *socket* UDP.
2. Lier le *socket* à l'adresse du serveur.
3. Attendre que le paquet de données arrive du client.
4. Traiter le paquet de données.
5. Envoyer une réponse au client, ou fermer le *socket*.
6. Retourner à l'étape 3 (si le *socket* n'est pas fermé).

Client

1. Créer un *socket* UDP.
2. Envoyer un message au serveur.
3. Attendre la réponse du serveur.
4. Répondre au processus.
5. Retourner à l'étape 2, ou fermer le *socket*.

Projet 1 : communiquer avec un ordiphone Android en utilisant UDP (le RPi est le serveur)

Dans ce projet, on utilise le protocole UDP pour recevoir et envoyer des données vers/depuis un ordiphone Android. Ici, le RPi est le serveur et l'ordiphone Android le client.

Objectif : ce projet vise à montrer comment une liaison UDP peut être établie entre un RPi et un ordiphone Android.

Schéma fonctionnel : la **figure 2** présente la structure générale du projet. Le RPi et l'ordiphone Android communiquent par l'intermédiaire d'une liaison routeur Wi-Fi.

Listage des programmes : le **listage 1** présente le programme *MyServer.c* qui peut être extrait du paquetage logiciel du livre. Rendez-vous sur le site (*www.elektor.fr/19703*), faites défiler la page vers le bas jusqu'à *Téléchargements*, puis cliquez sur *Software_C Programming on Raspberry Pi*. Stockez le fichier .zip localement, puis extrayez tous les fichiers ou seulement *MyServer.c*.

Les fichiers d'en-tête nécessaires au programme sont inclus au début de *MyServer.c*. Le message *Hello from Raspberry Pi* est stocké dans le tableau de caractères `msg`. Un *socket* de type UDP est ensuite créé en appelant la fonction `socket` et en enregistrant son identifiant



Figure 3. Appli de terminal UDP sur un ordiphone Android.



Figure 4. Entrez les numéros de port.



Figure 5. Exemple d'exécution du programme.

dans la variable `sock`. Les informations sur l'ordinateur serveur (RPi) sont ensuite données, l'adresse étant définie comme `INADDR_ANY`, de sorte que tout autre ordinateur sur le même réseau avec le numéro de port réglé sur 5000 pourra établir la communication avec le RPi. Les informations de l'ordinateur serveur sont ensuite liées au port spécifié en appelant la fonction `bind`.

Le reste du programme s'exécute dans une boucle. Dans celle-ci, on appelle la fonction `recvfrom` pour attendre la réception d'un paquet de données de l'ordinateur client (ordiphone Android). Remarquez qu'il s'agit d'un appel bloquant et que la fonction attendra jusqu'à la réception de données du client. Le programme se termine si le caractère `x` est reçu de l'ordinateur client. Un caractère `NULL` est ajouté aux données reçues qui sont affichées sur l'écran du RPi sous forme d'une chaîne de caractères à l'aide de la fonction `printf`. Une variable entière appelée `count` est convertie en caractère et ajoutée à la fin du tableau de caractères `msg`. Celui-ci est envoyé à l'ordinateur client. Dans la première la boucle, l'ordinateur client affichera *Bonjour de Raspberry Pi 1*. La deuxième fois, il affichera *Bonjour de Raspberry Pi 2*, et ainsi de suite. Le *socket* est fermé juste avant la fin du programme. Le programme peut être compilé et exécuté comme suit

```
gcc -o MyServer MyServer.c
sudo ./MyServer
```

Il existe de nombreuses applications UDP disponibles gratuitement sur le Play Store. Dans ce projet, on utilise le Terminal UDP de mightyIT (voir **fig. 3**). Saisissez le numéro de port et l'adresse IP de votre ordiphone Android et cliquez sur *Start Terminal*, comme indiqué à la **figure 4**. Un exemple d'exécution du programme est présenté à la **figure 5**, où l'on voit à la fois l'écran de l'ordiphone et celui du PC.
Remarque : vous pouvez trouver l'adresse IP de votre RPi en utilisant la commande `ifconfig`.

210346-04

Listage 1. MyServer.c

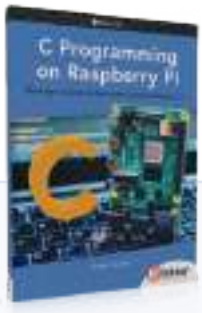
```
/*-----------------------------------------------------------------
   COMMUNICATION RASPBERRY PI - ORDIPHONE ANDROID
   ==============================================
Il s'agit d'un programme UDP. Le programme reçoit puis envoie des
messages à un ordiphone via le socket UDP. Le programme se termine
lorsque le caractère x est envoyé depuis l'ordiphone. Ceci est le
programme serveur UDP, qui communique sur le port 5000.
Auteur : Dogan Ibrahim
Fichier : MyServer.c
Date : Décembre 2020
-----------------------------------------------------------------*/
   #include <netinet/in.h>
   #include <stdio.h>
   #include <string.h>
   #include <stdlib.h>
   #include <unistd.h>
   #define Port 5000
   #define BUFFSIZE 1024
//
    // Début du programme MAIN
    //
   int main(void)
   {
     int sock, len, num, count = 1 ;
     char buffer[BUFFSIZE] ;
     struct sockaddr_in serveraddr, clientaddr ;
     char msg[26] = "Bonjour de Raspberry Pi" ;
  sock = socket(AF_INET, SOCK_DGRAM, 0) ;
     len = sizeof(clientaddr) ;
  serveraddr.sin_family = AF_INET ;
     serveraddr.sin_addr.s_addr = INADDR_ANY ;
     serveraddr.sin_port = htons(Port) ;
     bind(sock,(struct sockaddr *)&serveraddr, sizeof(serveraddr))
   ;
  while(1)
     {
       num = recvfrom(sock, buffer, BUFFSIZE,MSG_WAITALL,
             struct sockaddr *)&clientaddr, &len) ;
       if(buffer[0] == 'x')break ;
       buffer[num] = '\0' ;
       printf("%s\n",buffer) ;
           sprintf(&msg[24], "%d",count) ;
       count++ ;
       sendto(sock, &msg, strlen(msg), MSG_CONFIRM,
       struct sockaddr *)&clientaddr, len) ;
     }
     close(sock) ;
   }
```

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (d.ibrahim@btinternet.com) ou contactez Elektor (redaction@elektor.fr).

Contributeurs

Texte : Dogan Ibrahim
Rédaction : Jan Buiting
Mise en page : Harmen Heida
Traduction : Denis Lafourcade



PRODUITS

- Livre en anglais, « C Programming on Raspberry Pi » de D. Ibrahim
  Version papier : www.elektor.fr/19703
  Version numérique : www.elektor.fr/19704

test de préconformité CEM **pour un projet alimenté en courant continu**

Partie 2 : le matériel et son utilisation

Robert Schillinger (Würth Elektronik) et **Ton Giesberts** (Elektor)



Dans notre article précédent, nous avons présenté le réseau de stabilisation d'impédance de ligne (RSIL) pour courant continu qu'Elektor et Würth Elektronik proposent dans la boutique en ligne d'Elektor. Cette fois-ci, nous examinons le matériel de plus près et son utilisation dans les tests de préconformité CEM des montages électroniques.

Un RSIL est un filtre passe-bas, généralement placé entre une source de courant alternatif ou continu et le MàT (matériel à tester), pour créer une impédance connue et fournir un port de mesure du bruit radiofréquence (RF). Il isole également les signaux RF indésirables de la source d'alimentation. En outre, les RSIL peuvent être utilisés pour prédire les émissions conduites pour les tests de diagnostic et de préconformité [1].
Le double RSIL pour courant continu (DC) présenté ici est utilisé pour mesurer les émissions d'interférences conduites d'un objet de test dans la gamme de fréquences de 150 kHz à 200 MHz. L'appareil est basé sur la norme CISPR 25/ISO 7637 pour les systèmes électriques automobiles. Il mesure les interférences RF sur les deux canaux de ces systèmes en utilisant des inductances de blocage de 5 µH, valeur typique pour les RSIL de test des équipements automobiles. La conception est basée sur une note d'application d'Analog Devices (DC2130A – anciennement Linear Technology [2]). Ce RSIL offre la possibilité de mesurer le bruit en mode différentiel (DM) et en mode commun (CM).

Le circuit RSIL

La partie supérieure de la **figure 1** montre le schéma de base de tous les types de RSIL : une inductance de blocage du bruit RF, un condensateur pour supprimer la composante continue de la sortie RF, et une impédance de 50 Ω pour le port de mesure. En pratique, cette dernière impédance est beaucoup plus complexe qu'une simple résistance. La partie inférieure de ce

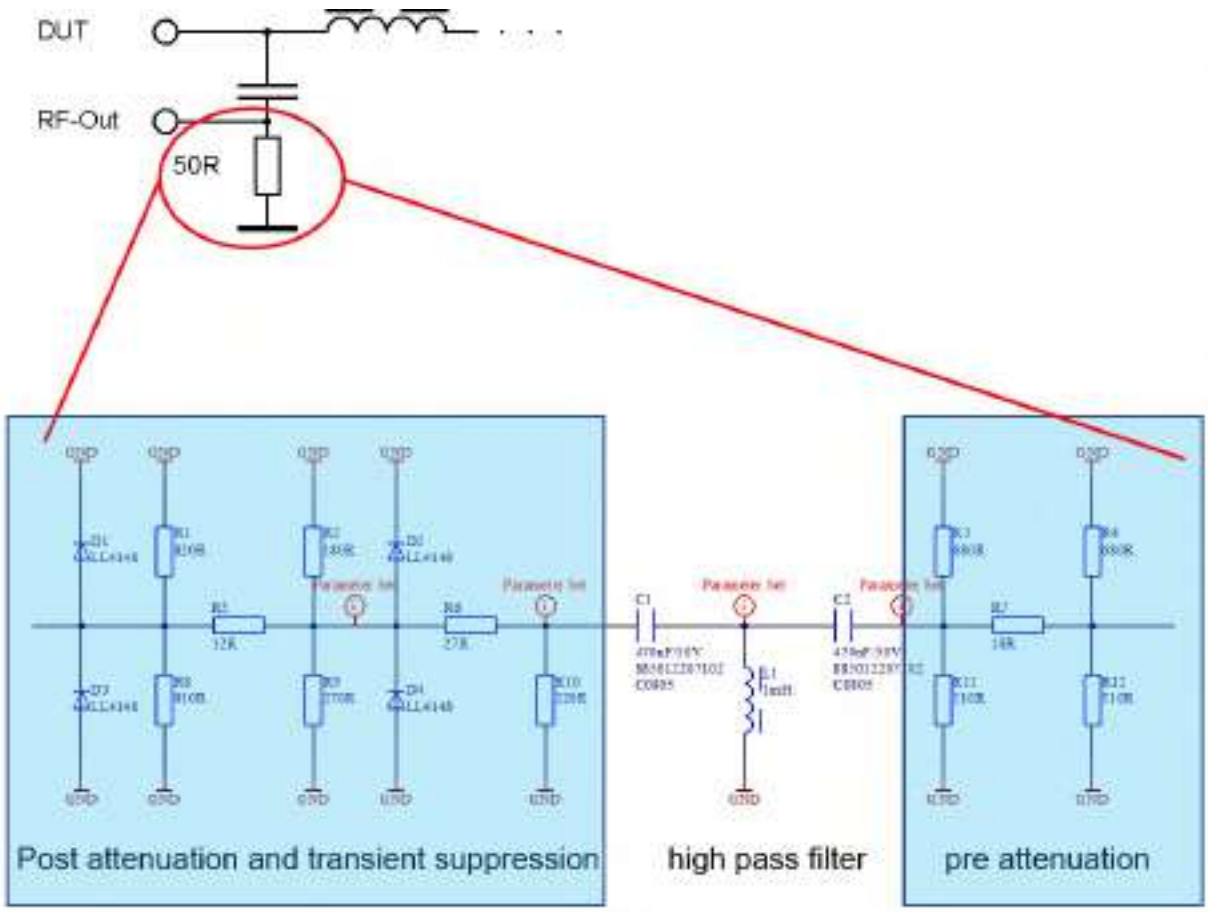


Figure 1. Les étages de sortie RF en détail.

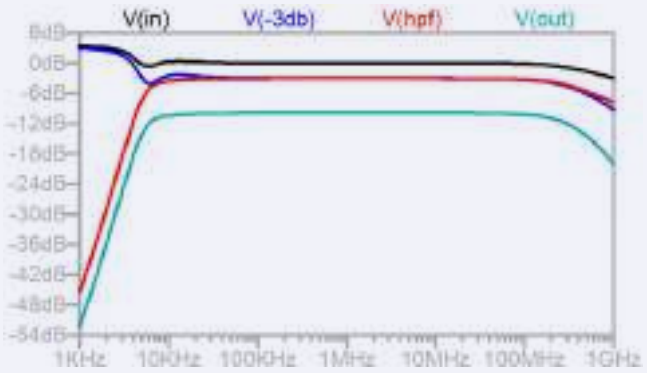


Figure 2. Caractéristiques du RSIL.

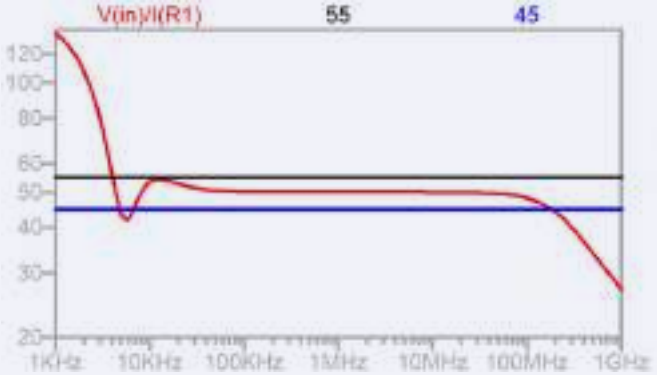


Figure 3. Courbe d'impédance.

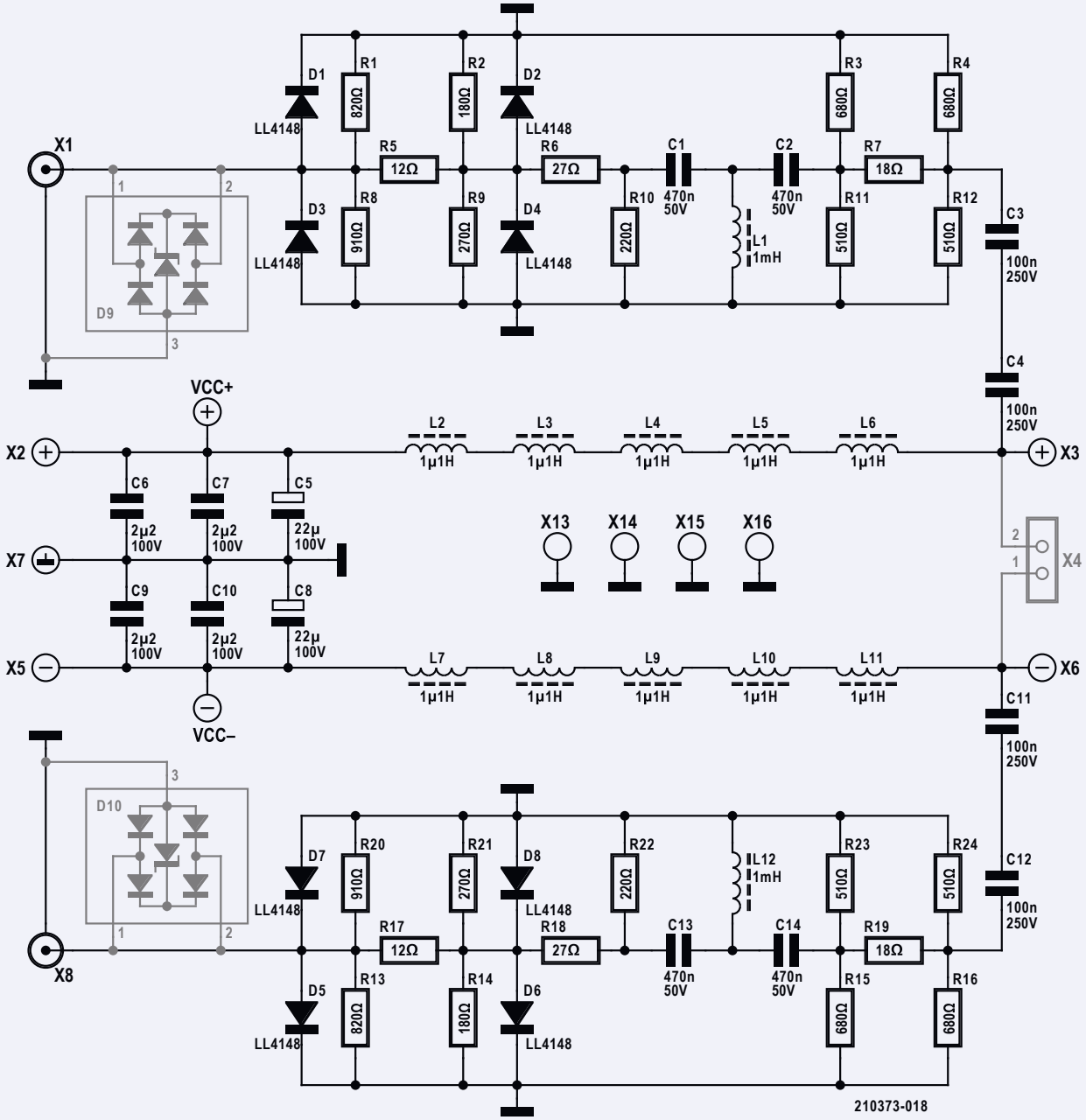


Figure 4. Le schéma complet de notre RSIL.

schéma montre qu'elle peut être subdivisée en trois étages : pré-atténuation, filtre passe-haut et post-atténuation. Nous allons les examiner tous les trois plus en détail. Les courbes présentées sont le résultat de simulations informatiques des circuits.
Ce qui suit décrit le réseau d'atténuation, à l'exclusion des condensateurs de couplage C3/C4 et C11/C12. Ces condensateurs limitent la fréquence inférieure de la plage de mesure à la norme de 150 kHz (la fréquence de coupure est d'environ 64 kHz ; à 125 kHz, l'atténuation est d'environ –1 dB).

Pré-atténuation :
Au premier étage, un filtre en π [V(–3db)] de –3 dB à 50 Ω réduit le niveau de bruit. La courbe bleue de la **figure 2** représente l'atténuation en fonction de la fréquence.

Filtre passe-haut :
Le deuxième étage est un filtre passe-haut du troisième ordre [V(HPF)] avec une fréquence de coupure de 5,5 kHz (–3 dB) pour limiter la gamme de fréquences de 9 kHz à environ 200 MHz, comme le montre la figure 2 (courbe rouge). Cet étage ne filtre que les fréquences inférieures à 6 kHz et n'est pas absolument nécessaire pour atteindre l'atténuation de 10 dB pour le signal. Il serait possible de le supprimer de la conception.

Post-atténuation :
Le dernier étage porte l'atténuation à –10 dB [V(out)] (courbe bleue de la figure 2). Il s'agit d'une valeur typique pour un RSIL qui doit être compensée par l'appareil de mesure (par exemple, par un gain interne de 10 dB) ou par le post-traitement des données de mesure dans une feuille Excel.
Les inductances de 5 µH bloquent le chemin de retour vers l'alimentation du bruit produit par le MàT sur toute la gamme de fréquences. 5 µH est une valeur typique utilisée dans la norme CISPR25. Lorsqu'elle est utilisée pour une configuration CISPR22, il faut observer ce qui se passe aux fréquences inférieures à 400 kHz. En raison de la faible inductance (diminution de la réjection AC à des fréquences plus basses), vous mesurerez les niveaux les plus bas.
La diminution de l'impédance au-delà de 100 MHz (**fig. 3**) est due à la capacité parasite des diodes d'écrêtage. Ces diodes ont pour fonction de supprimer les pointes de tension élevées sur les sorties RF. Pour les tests de préconformité, si l'on dispose d'un oscilloscope avec des entrées moins sensibles et protégées en interne, on peut les omettre.

Webinaire sur la CEM : comment déboguer une alimentation à découpage à transfert indirect (*flyback*) !
Notre prochain webinaire présentera les bases du débogage d'une alimentation *flyback* à l'aide d'un oscilloscope et évaluera les instruments complémentaires nécessaires. Nous vous donnerons un aperçu approfondi des alimentations à découpage, de leurs sources de bruit et du type de filtrage nécessaire pour passer avec succès les tests de conformité CEM. À la fin du webinaire, vous devriez être en mesure de faire la distinction entre le bruit en mode différentiel (DM) et celui en mode commun (CM) ainsi que de comprendre comment déterminer le mode dominant.Ce webinaire aura lieu le 2 décembre 2021, à 16h00 CET. Davantage d'informations bientôt sur www.elektormagazine.com/webinars.

La **figure 4** montre le schéma complet de notre double RSIL DC. Les circuits de sortie RF décrits sont facilement identifiables, X1 et X8 étant les sorties RF pour les mesures d'interférences conduites. L'alimentation en courant continu (entrée, mono-tension et connexion supplémentaire à l'alimentation et au plan de masse) est connectée à X2, X5 et X7. Les bornes d'alimentation du MàT (sortie) sont connectées à X3 et X6 (ou X4, qui n'est pas inclus dans notre kit RSIL).

Les inductances de 5 µH normalement employées pour ce RSIL sont réalisées ici par la mise en série de cinq bobines standard de 1,1 µH du catalogue de Würth Elektronik. Une des raisons du choix de ce composant est le courant maximal pouvant atteindre 10 A. Le modèle utilisé est petit, avec une fréquence de résonance élevée, et donc moins de capacité parasite. Il a un bon rapport inductance/courant (variation d'inductance limitée sur toute la gamme de courant jusqu'à 10 A). La plupart des RSIL du commerce utilisent des bobines à air, mais qui nécessitent plus d'espace et ne sont pas disponibles en tant que composants indépendants.

Acheter ou faire soi-même ?

Un kit complet pour ce RSIL – sponsorisé par Würth Elektronik – est disponible dans la boutique d'Elektor [3]. Comme tous les composants CMS sont préassemblés, vous n'aurez qu'à souder les connecteurs et monter le circuit imprimé à l'intérieur du boîtier déjà percé et fraisé pour réaliser le RSIL.

Pour les plus courageux d'entre vous, les fichiers Gerber nécessaires pour commander votre propre exemplaire du circuit imprimé à quatre couches sont disponibles sur la page Elektor Lab de ce projet [4] (voir **fig. 5**). La liste détaillée du matériel vous guidera pour dénicher les

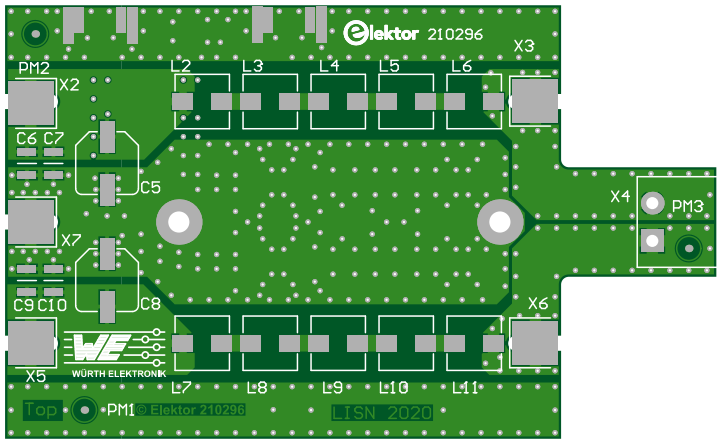


Figure 5. Implantation sur le circuit imprimé.

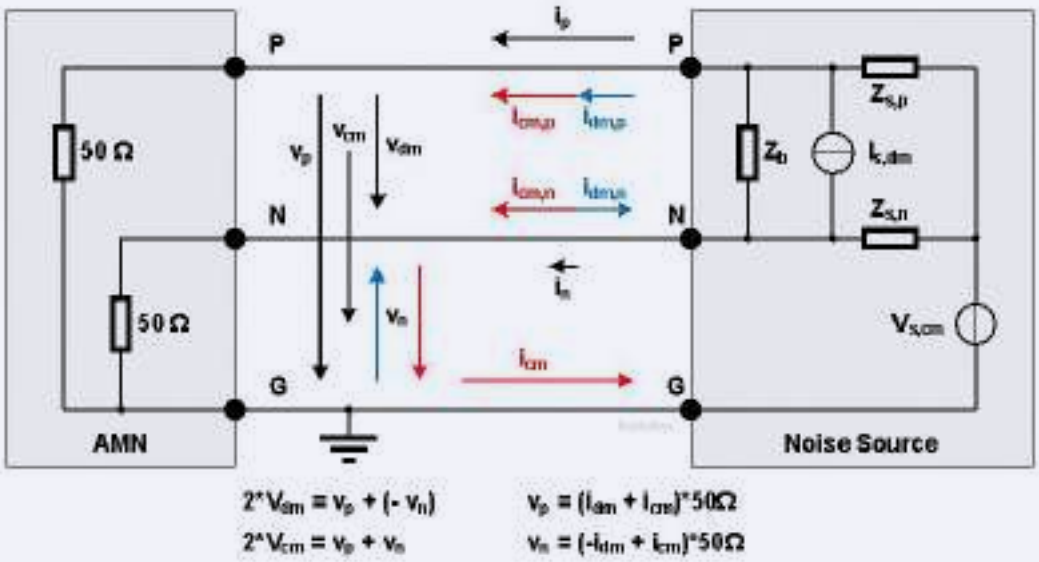


Figure 6. Tensions et courants en modes commun et différentiel.

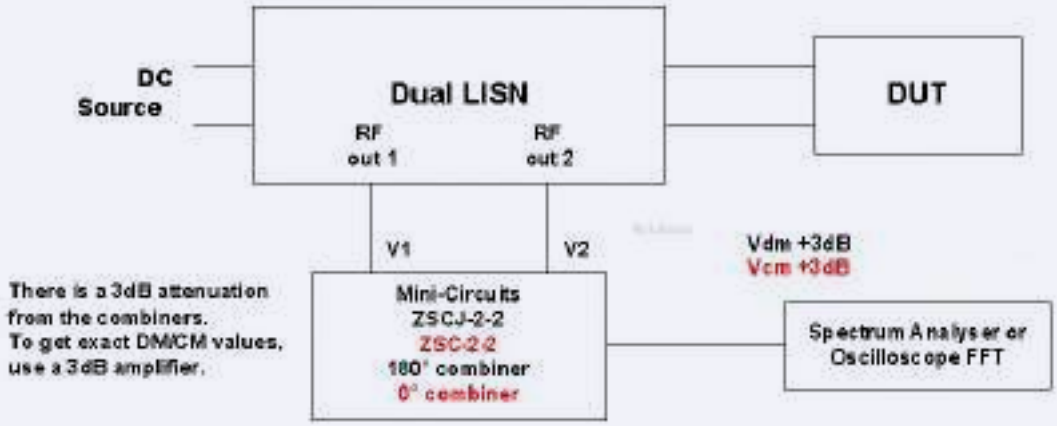


Figure 7. Configuration de test utilisant des combinateurs Mini-Circuit (ou pas).

composants nécessaires à la réalisation de votre RSIL.

Les composants CMS sur la face supérieure de la carte, C5-C10 et L2-L11, peuvent être soudés dans un four à refusion. Le soudage des inductances à l'aide d'une station de retravail à air chaud ou d'un fer à souder à pointe fine sera difficile – voire impossible – car les pastilles situées entre les inductances sont presque inaccessibles. De plus, le circuit imprimé est de type 4 couches et une grande puissance de soudage est nécessaire pour chauffer une pastille, surtout si elle est reliée à un plan d'alimentation.

Les composants sur la face inférieure du circuit imprimé, comme les petites résistances 0603, les condensateurs et les deux petites inductances, peuvent être soudés à l'air chaud. Souder une 0603 avec un petit fer à souder est possible, mais beaucoup de chaleur est perdue dans les quatre couches. Avec l'air chaud, réglez son débit au minimum, sinon ces petits composants risquent de partir à la dérive.

Les connecteurs SMA femelles (X1, X8) peuvent être soudés à l'air chaud. Tout d'abord, appliquez de la pâte à souder sur toutes les pattes en haut et en bas du premier connecteur, positionnez-le sur le circuit imprimé, et augmentez le flux d'air pour le souder des deux côtés. Assurez-vous que la broche centrale est bien positionnée au milieu de sa pastille et soudez-la en premier. Puis soudez les pattes de masse des deux côtés du circuit imprimé.

Les contacts à ressort (doigts de contact) peuvent également être soudés à l'air chaud avec de la pâte à souder. Mettez-en un peu sur les pastilles (X13-X16) et placez le contact en position verticale. Arrêtez lorsque toute la pâte à souder coule et que le contact a bien pris sa place.

Les diodes en boîtiers MiniMELF peuvent être soudées à l'aide d'un fer à petite panne plate et un fil de soudure très fin, par exemple de 0,35 mm. Commencez par mettre un peu de soudure sur une pastille, de préférence celle de masse, car faire couler la soudure sur ces pastilles prendra un peu de temps. Quand c'est prêt, placez la diode. La pastille de signal est plus facile à souder. À l'aide de brucelles, alignez soigneusement la diode bien au centre de son repère sur le circuit imprimé et soudez une extrémité. Pour chaque paire de diodes, la cathode de l'une (anneau noir) et l'anode de l'autre sont soudées à la masse. Soudez maintenant l'autre côté

et reprenez la première en ajoutant un peu de soudure.

Une fois que tous les composants sont soudés sur les deux faces, fixez les douilles bananes au boîtier avec les contacts en position verticale. Faites glisser le circuit imprimé dans le boîtier avec les prises SMA dans les trous correspondants. Veillez à ce que les contacts à ressort sur le bas du circuit imprimé n'accrochent pas le bord du boîtier.

Appuyez doucement sur le circuit imprimé et placez la rondelle frein et l'écrou sur les connecteurs SMA. Soudez ensuite les contacts des fiches bananes aux pastilles (X2, X3, X5, X6, X7) tout en maintenant le circuit imprimé en place. Fixez le couvercle sur le boîtier et le RSIL est prêt à l'emploi.

Utilisation du RSIL

La **figure 6** illustre les tensions et les courants impliqués dans la mesure des deux types de bruit conduit.

Pour le bruit en mode différentiel, le courant circule de la source vers la charge et revient sur l'autre ligne. La mise à la terre n'est pas importante dans ce cas. La tension en mode différentiel est calculée avec $V_{dm}=0,5*(V_p-V_n)$.

Pour le mode commun, le courant circule sur les deux lignes dans le même sens et retourne à la source via la terre. Ce courant de retour peut également être conduit par des effets de couplage parasite. La tension de mode commun est calculée avec $V_{cm}=0,5*(V_p+V_n)$.

Il y a deux façons de mesurer le bruit en mode différentiel et en mode commun avec notre RSIL (voir **fig. 7**). La première utilise un combinateur, disponible chez Mini-Circuits [5], pour additionner les deux sorties RF (avec ou sans déphasage de 180°), puis effectuer les mesures et les calculs de mode différentiel/mode commun.

La deuxième approche utilise un oscilloscope numérique à deux canaux et ses fonctions mathématiques (addition et FFT). Dans ce cas, les combinateurs de Mini-Circuits sont inutiles, car V1 et V2 sont directement connectés aux canaux d'entrée de l'oscilloscope.

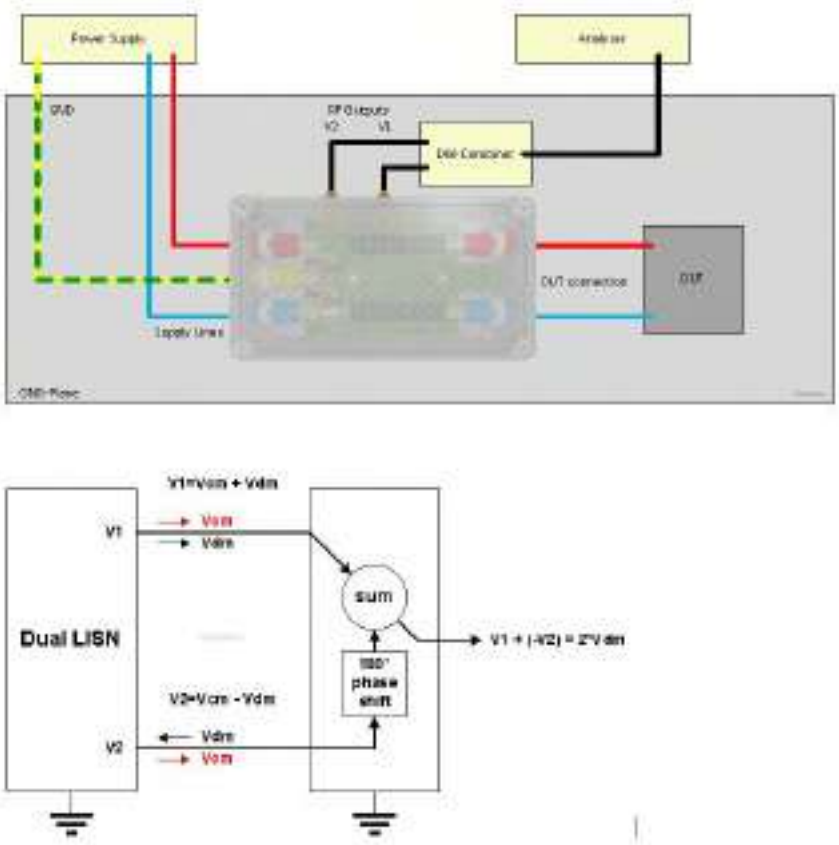


Figure 8. Configuration de la mesure du bruit en mode différentiel.

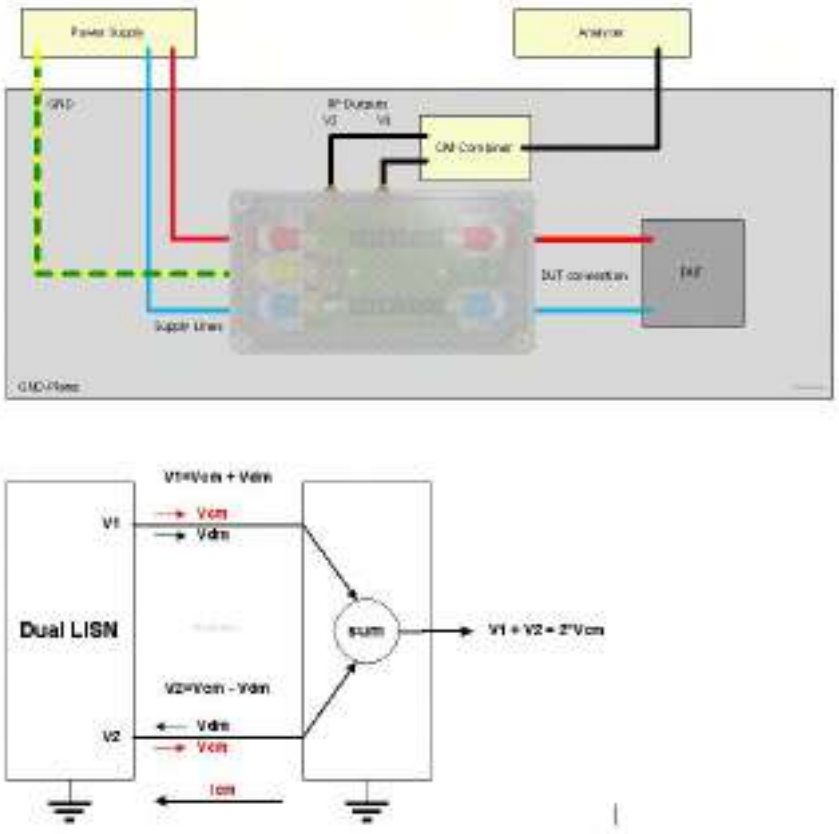


Figure 9. Configuration de la mesure du bruit en mode commun.

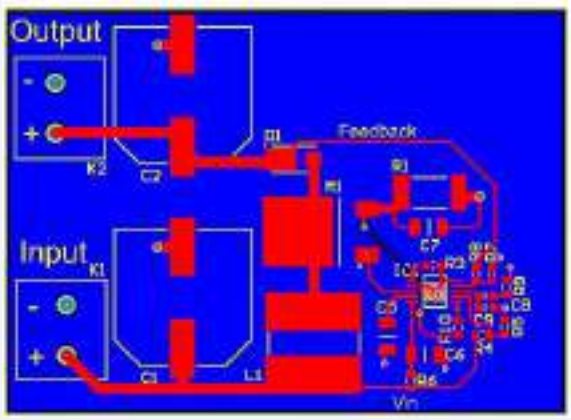


Figure 10a. Première version, non conforme.

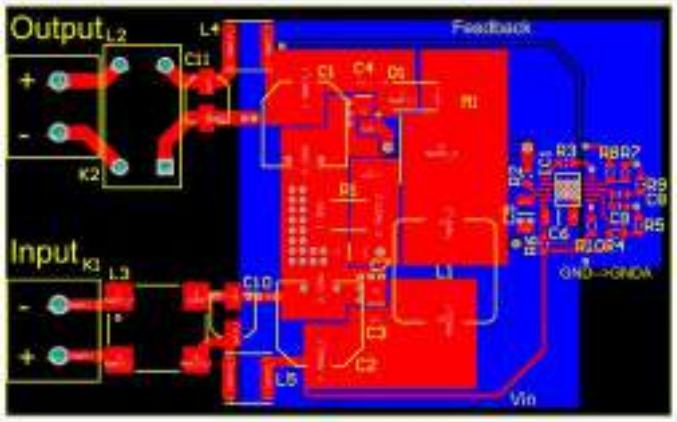


Figure 10b. Version avec implantation et filtrage améliorés, conforme à la norme CISPR22B.

Figure 10. Deux versions du circuit imprimé d'une alimentation à découpage :

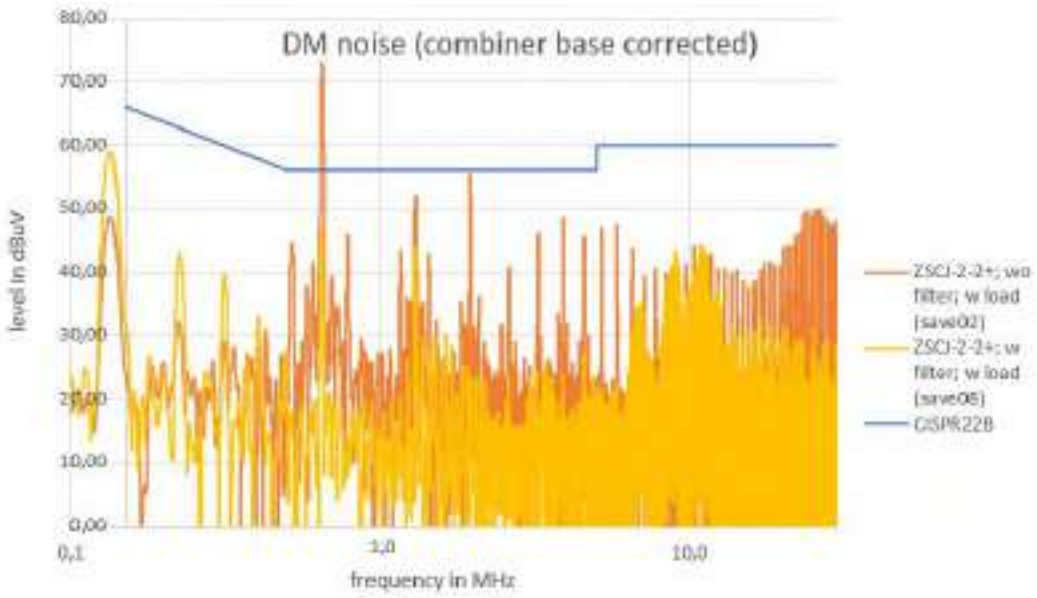


Figure 11a.
Mode différentiel.

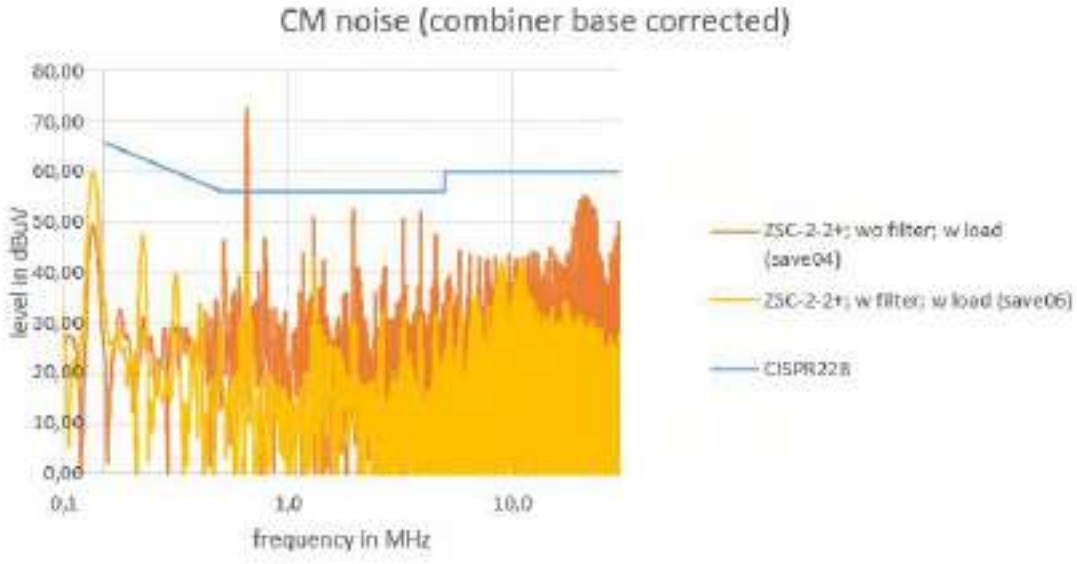


Figure 11b.
Mode commun.

Figure 11. Bruit en fonction de la fréquence pour les deux versions :

Par précaution, ne connectez l'analyseur que lorsque la source d'alimentation connectée au RSIL est sous tension afin de protéger l'instrument de mesure. Déconnectez l'analyseur du RSIL avant d'éteindre ou de débrancher la source d'alimentation du RSIL. Tout port de sortie non utilisé doit être terminé avec 50 Ω.

La **figure 8** montre comment effectuer les mesures de bruit en mode différentiel. Les deux tensions de sortie du RSIL sont additionnées, l'une étant déphasée de 180° avant l'addition. Dans le cas idéal, les tensions de mode commun s'annulent et celles de mode différentiel sont doublées. Pour obtenir le niveau correct pour cette mesure, il faut soustraire 6 dB à la valeur mesurée. Il faut aussi tenir compte de l'atténuation de –3 dB du combinateur ce qui, avec les –10 dB du RSIL, conduit aux corrections suivantes :
–6 dB + 3 dB + 10 dB = +7 dB (avec le combinateur)
–6 dB + 10 dB = +4 dB (corrections uniquement mathématiques)

La **figure 9** illustre la mesure du bruit en mode commun. Idéalement, la somme de V1 et V2 donne deux fois la composante de tension en mode commun, tandis que celle en mode différentiel s'annule. Si les tensions sont égales en amplitude et en phase sur les deux lignes, on obtient le double (+6 dB) de la tension sur chaque ligne.

Exemple : test d'une alimentation à découpage

Pour montrer comment les tests de préconformité sont réalisés avec ce RSIL et comment les résultats doivent être traités, nous avons effectué des mesures sur deux exemples de circuits. L'un (**fig. 10a**) est un premier jus grossier d'une alimentation à découpage, l'autre (**fig. 10b**) est la version améliorée du premier après des tests de préconformité réalisés dans un labo CEM professionnel. Ces mesures nous ont permis de corriger les niveaux de CEM obtenus avec notre propre RSIL et l'instrument de mesure de notre labo. Voilà un bon usage de la préconformité ou des mesures croisées pour déboguer un circuit maison.

Caractéristiques techniques

Chemin RF

Canaux :	2 (avec diodes d'écrêtage)
Bande passante :	200 MHz
Inductance :	5 µH || 50 Ω
Atténuation interne :	10 dB

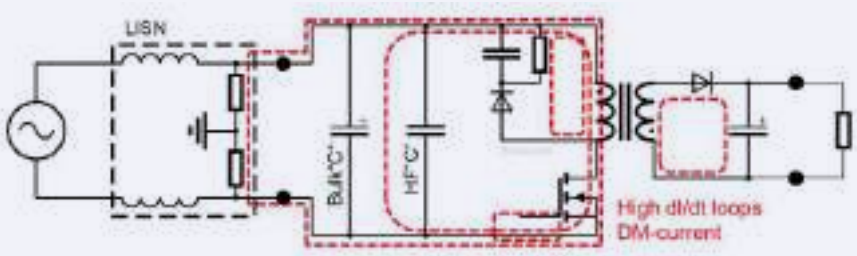
Chemin DC

Courant max. :	< 10 A continu
Tension max. :	< 60 V continu
Résistance en continu :	< 2 × 70 mΩ
Taille du circuit imprimé :	94,2 × 57,4 mm (sans connecteurs)

Bruit conduit

Il existe deux types de bruit conduit, et tous deux peuvent être mesurés avec ce RSIL.

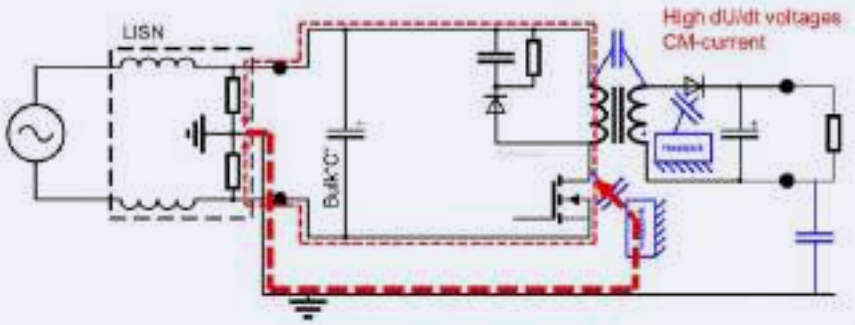
Mode différentiel



Il est causé par un dI/dt élevé ; les circuits se comportent comme des antennes-cadres ou des dipôles magnétiques.

- Chemin du courant comme sur le schéma ; relativement facile à identifier
- Chemin de retour du courant très proche
- Courants relativement élevés
- dI/dt est le facteur principal
- Filtrage possible par filtres LC en π et en T

Mode commun



Il est causé par un dV/dt élevé ; les circuits se comportent comme des antennes dipôles et monopôles électriques.

- Chemins inattendus des courants parasites
- Long chemin de retour du courant
- Courants relativement faibles (gamme des µA)
- dV/dt est le facteur principal
- Filtrage possible avec condensateurs céramiques multicouches (CMC) et condensateurs Y

La **figure 11** donne les résultats des mesures pour le bruit conduit en mode commun et pour celui en mode différentiel. Pour les deux modes, la deuxième version, avec un dessin amélioré du circuit imprimé et un filtrage supplémentaire, est conforme aux limites de la norme CISPR22B.

Les deux types de bruit sont mesurés de 150 kHz à 30 MHz, la gamme de fréquences correspondant aux exigences de la norme CISP22. À des fréquences plus élevées, les émissions rayonnées sont mesurées à l'aide d'antennes. Les courbes en orange représentent la carte sans filtre. À 650 kHz, on distingue la fréquence de base de l'alimentation à découpage (SMPS) avec 72 dBµV. On voit également quelques harmoniques à 1,3 MHz, 1,95 MHz, etc. La ligne bleue indique la limite de la classe B de la norme CISPR22 ; le bruit conduit doit se situer à environ 10 dB en dessous de cette ligne. Le pic à la fréquence de découpage doit donc être réduit de 22 dB !

En améliorant l'implantation des composants et en ajoutant des filtres dans la deuxième version, nous avons réussi à réduire ce pic. Les courbes en jaune montrent le nouveau résultat de 45 dBµV à 650 kHz.

Quand on compare les résultats des tests CEM d'un labo certifié avec ceux de notre double RSIL DC, il en ressort qu'il faudra peut-être ajuster la configuration de mesure pour garantir que vos propres tests donneront une bonne évaluation de la conformité CEM de vos produits électroniques..

210373-01

LIENS

[1] Wikipedia sur les réseaux RSIL : https://en.wikipedia.org/wiki/Line_Impedance_Stabilization_Network
[2] Note d'application DC2130A d'Analog Devices : https://bit.ly/2SST8dc
[3] Kit du double RSIL pour courant continu : http://www.elektor.fr/19869
[4] Page de ce projet sur Elektor Labs : https://bit.ly/2V9zBpJ
[5] Mini-circuits pour combinateurs : https://bit.ly/3CrVzWg

PRODUITS

- Kit RSIL DC double www.elektor.fr/19869
- STEMlab 125-14 (kit de démarrage) www.elektor.fr/17939

Contributeurs

Idée et conception : **Robert Schillinger** (Würth Elektronik)
Texte : **Robert Schillinger, Ton Giesberts**
Illustrations : **Robert Schillinger, Patrick Wielders**
Rédaction : **Luc Lemmens, Stuart Cording**
Traduction : **Helmut Müller**
Mise en page : **Harmen Heida**

Des questions, des commentaires ?

Envoyez un courriel au rédacteur (luc.lemmens@elektor.com) ou contactez Elektor (redaction@elektor.fr).

LISTE DES COMPOSANTS

Résistances (toutes CMS 0603)

R1,R13 = 820 Ω 1%
R2,R14 = 180 1%
R3,R4,R15,R16 = 680 Ω 1%
R5,R17 = 12 Ω 1%
R6,R18 = 27 Ω 1%
R7,R19 = 18 Ω 1%
R8,R20 = 910 Ω 1%
R9,R21 = 270 Ω 1%
R10,R22 = 220 Ω 1%
R11,R12,R23,R24 = 510 Ω 1%

Condensateurs

C1,C2,C13,C14 = 470 nF, 50 V, CMS 0805 X7R (Würth Elektronik 885012207102)
C3,C4,C11,C12 = 100 nF, 250 V, CMS 1206 X7R (Würth Elektronik 885342208004)
C5,C8 = 22 µF, 100 V, CMS WCAP-ASLL (8x10.5) (Würth Elektronik 865060853003 ou 865080853006)
C6,C7,C9,C10 = 2,2 µF, 100 V, CMS 1210 X7R (Würth Elektronik 885012209071)

Inductances

L1,L12 = 1 mH, 200 mA, CMS WE-LQS 5040 (Würth Elektronik 74404054102)
L2,L3,L4,L5,L6,L7,L8,L9,L10,L11 = 1,1 µH, 15000 mA, CMS WE-HCI 7050 (Würth Elektronik 744314110)

Semi-conducteurs

D1,D2,D3,D4,D5,D6,D7,D8 = LL4148 100 V 100 mA, CMS MiniMelf (SOD-80)
D9,D10 = diode TVS, non utilisée/montée

Divers

X1,X8 = SMA femelle 9,52, 5 broches, 180° pour circuit imprimé 1,6 mm (Würth Elektronik 60312242114510)
X2,X3 = prise de sécurité, 4 mm, jack, montage sur panneau, 24 A, 1 kV, contacts dorés, rouge (Staubli 23.3000-22)
X5,X6 = prise de sécurité, 4 mm, jack, montage panneau, 24 A, 1 kV, contacts dorés, bleu (Staubli 23.3000-23)
X7 = prise de sécurité, 4 mm, jack, montage panneau, 24 A, 1 kV, contacts dorés, vert/jaune (Staubli 23.3000-20)
X13,X14,X15,X16 = doigt de contact WE-SECF SMD EMI, 7×2,5×13 mm (Würth Elektronik 331161702513)
X4 = non monté
Boîtier = 1590N1 aluminium moulé, 121,2 × 65,5 × 39,8 mm, Hammond MFG

Circuit imprimé 210296-1 v1.0

Optionnel (pour connecter le RSIL au STEMlab 250-14)
4 adaptateurs SMA mâle – BNC en cas d'utilisation de 2 câbles BNC standard ou de 2 câbles SMA (mâle-mâle)

Propeller 2 de Parallax (5)

La fonction de « broche intelligente »

Mathias Claussen (Elektor)

Pour conclure cette série sur le microcontrôleur Propeller 2 de Parallax, nous présentons la fonction « smart pin » (broche intelligente), qui permet des configurations universelles et flexibles pour les broches d'E/S. Nous nous intéresserons également aux résistances de rappel vers le haut et le bas des broches d'E/S. Et ce n'est pas tout…

Comme les broches du Propeller 2 possèdent davantage de fonctions que celles que l'on trouve habituellement dans d'autres microcontrôleurs, un examen plus approfondi pourrait révéler des informations intéressantes. Pour établir une sorte de référence, nous jetterons d'abord un coup d'œil à la structure GPIO d'un ATmega328P de Microchip Technology. Ensuite, nous passerons au Propeller 2 pour voir s'il y a des différences et comment elles pourraient être utiles ultérieurement. Enfin, la lecture de l'état de quelques boutons devrait être assez facile.

Fonctionnement des broches d'E/S sur les autres microcontrôleurs

Les lecteurs familiers avec l'ATmega328P savent que les broches d'E/S ont pour l'essentiel quatre états : entrée, entrée avec rappel au niveau haut, sortie à niveau bas et sortie à niveau haut. Les fonctions analogiques occupent spécifiquement quelques broches sur l'ATmega328P et ne seront pas abordées ici. D'après la fiche technique, la **figure 1** montre comment est construite une broche d'E/S.

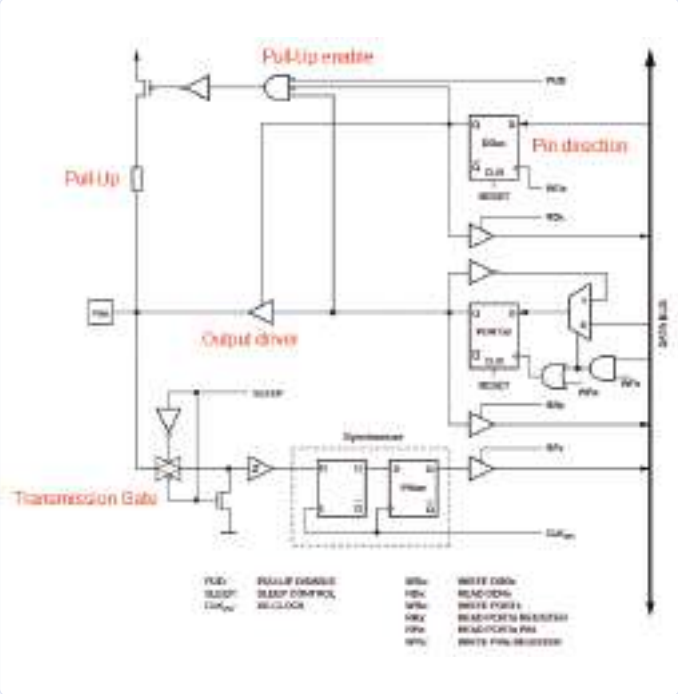


Figure 1. Partie de la fiche technique consacrée au bloc d'E/S ATmega328.

Nous avons essentiellement un mécanisme de commande pour une résistance de rappel au niveau haut, d'une valeur comprise entre 20 kΩ et 50 kΩ, composée d'un FET et de la résistance elle-même, et d'une logique de commande pour activer le FET (comme le montre la **figure 2**).
Dans la partie inférieure de la figure 1, vous pouvez voir une porte de transmission (**figure 3**) qui joue le rôle de commutateur à commande numérique permettant le passage de tensions analogiques.

Ce qui est intéressant, c'est le signal *SLEEP*, car il désactive la porte de transmission et ramène en même temps sa sortie à la masse avec un FET qui lui est propre. Ceci est dû au trigger de Schmitt, situé à la sortie de la porte de transmission. Il sert à transformer un niveau de tension analogique en valeur binaire zéro ou un. Au centre de la **figure 4**, se trouve l'étage de sortie avec activation.
Si le signal *enable* n'est pas actif, l'étage de sortie sera déconnecté ou il produira un niveau bas ou haut, selon l'entrée appliquée. Nous donnons beaucoup d'explications pour une simple activation ou désactivation,

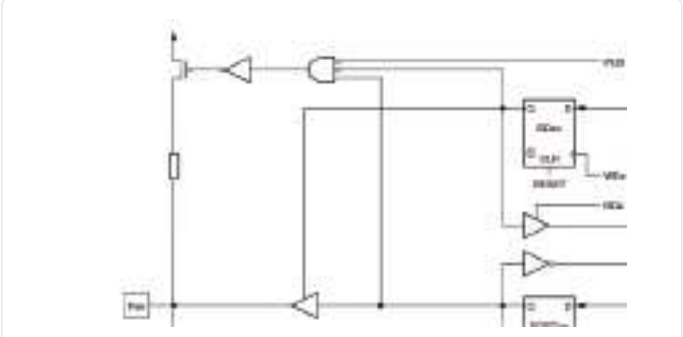


Figure 2. Transistor FET de l'ATmega328 pour le rappel au niveau haut.

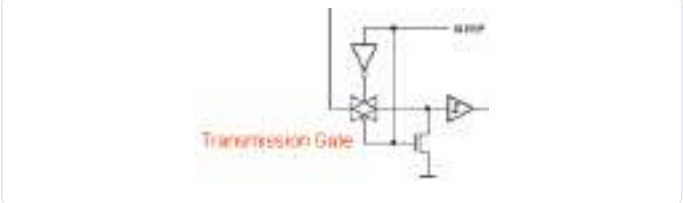


Figure 3. Porte de transmission.

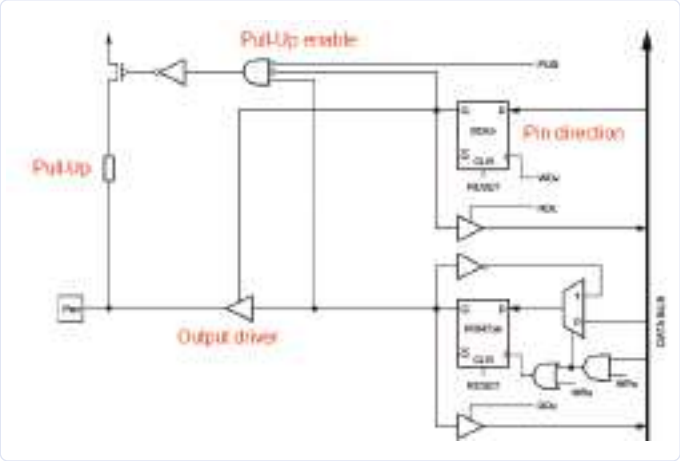


Figure 4. FET de l'ATmega328 pour le rappel au niveau haut.

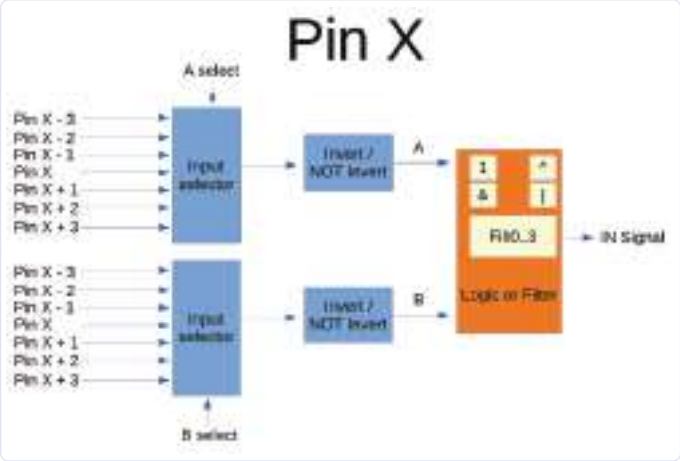


Figure 5. Voies d'entrée pour les broches d'E/S du Propeller 2.

même s'il s'agit surtout de décrire l'essentiel. La fonction de rappel au niveau haut va être très pratique pour une bonne raison. Nous ne pouvons mettre toutes les broches en sommeil que simultanément (comme d'autres parties de l'ATmega328). Si nous voulons qu'une broche soit inutilisée et ne fasse rien, son entrée sera flottante. Dans ce cas, elle capte du bruit aléatoire que le trigger de Schmitt convertit en zéro ou un binaire. Comme l'entrée est aléatoire, le trigger de Schmitt et la logique interne vont basculer rapidement entre zéro et un. Ainsi, à chaque transition, nous avons besoin d'un peu d'énergie. La broche étant inutilisée, ce n'est pas souhaitable et cela gaspille de l'énergie. Cette approche est à éviter, surtout si le montage fonctionne sur batterie. L'utilisation du rappel au niveau haut va donc fixer la tension à l'entrée à VCC et la commutation aléatoire de l'entrée ne se produira pas. Cette introduction d'une fonction simple était un peu longue, mais nous allons maintenant nous concentrer sur la mise en œuvre du microcontrôleur Propeller 2.

Broches intelligentes du Propeller 2

Comme je l'ai déjà mentionné, il n'y a pas de broches d'E/S simples sur le Propeller 2. En ce qui concerne les fonctions d'E/S de base, un coup d'œil à la fiche technique préliminaire du microcontrôleur révèle que nous avons plus de quatre choix, même pour une configuration simple d'entrée et de sortie. Toutes les broches peuvent fonctionner en mode entrée ou sortie numérique ou analogique. Commençons par le mode numérique. Pour l'entrée, si vous vous souvenez de l'ATmega328, tout était simple avec une porte de transmission et un trigger de Schmitt pour lire le signal. Dans le cas présent, nous avons un peu plus d'intelligence à l'intérieur de la broche. La **figure 5** montre la voie d'entrée numérique, ou les autres voies possibles, pour une seule broche.

La première chose étrange est que nous avons deux sélecteurs d'entrée parmi lesquels choisir. Vous pouvez ainsi sélectionner la broche actuelle et ± trois broches proches pour chacune de ces entrées. Une fois cette sélection effectuée, nous pouvons utiliser le signal inversé ou non inversé, ce qui donne un terme A ou B. Ces termes A et B sont ensuite introduits dans une deuxième partie où il est possible d'effectuer des opérations logiques ou des filtrages. Le résultat final sélectionné est ensuite présenté comme signal d'entrée (IN) du système. Ainsi, les broches standard sont un peu plus compliquées, mais aussi un peu plus polyvalentes que celles d'un ATmega328P. Pour la sortie numérique, les choses sont simples : nous avons une sortie au niveau bas et au niveau haut. Rien d'extraordinaire à première vue.

Vous vous souvenez de la résistance de rappel au niveau haut dont nous avons parlé au début ? Le Propeller 2 en a plusieurs et peut aussi appliquer un rappel vers le haut et vers le bas. Les combinaisons possibles sont assez simples :

- 1,5 kΩ
- 15 kΩ
- 150 kΩ
- 19 Ω
- 1 mA
- 100 µA
- 10 µA
- État flottant

Comme nous pouvons choisir d'avoir des résistances de rappel vers le haut ou le bas ou bien des sources de courant, les broches sont très flexibles même pour les différents bus dont nous pouvons avoir besoin ou avec lesquels nous voulons interagir.

Les rappels vers le haut ou le bas ne sont pas réalisés comme dans l'ATmega328 avec un FET spécial qui les active ou les désactive. Ces résistances agissent plutôt comme une *force de commande*, comme si elles se situaient entre l'étage de commande et la sortie de la broche. Pour utiliser les rappels vers le haut ou vers le bas, vous basculez la broche en mode sortie, avec la résistance donnée, et vous appliquez le niveau haut ou bas pour un rappel correspondant. Mais l'histoire ne s'arrête pas là, comme le montre la **figure 6**, qui donne un aperçu de la structure détaillée d'une broche.
Nous avons également un CN/A (convertisseur numérique-analogique) et un CA/N (convertisseur analogique-numérique) pour chaque broche qui peut également être utilisée en mode analogique. Comme je l'ai écrit dans un article précédent, la documentation n'est pas encore complète. Il manque notamment la description de certains domaines pour quelques configurations d'E/S, surtout lorsqu'il s'agit de configurations détaillées. Pour une première expérience pratique avec un composant électronique de pré-production, c'est assez courant. La **figure 7** propose un aperçu du registre de configuration des broches intelligentes. Et nous en avons fini avec la théorie pour le moment.

Mise en pratique

Pour notre premier test dans le monde réel, nous utiliserons nos quatre boutons d'entrée. Nous pouvons par ex. allumer une LED en appuyant sur un bouton. Pour cela, nous étendons notre code Spin2 existant et utilisons toutes les capacités que nous avons, comme la commande d'une LED à l'aide d'une broche d'E/S et la fonction `prints()` pour écrire des chaînes de caractères dans un UART. Notre objectif est

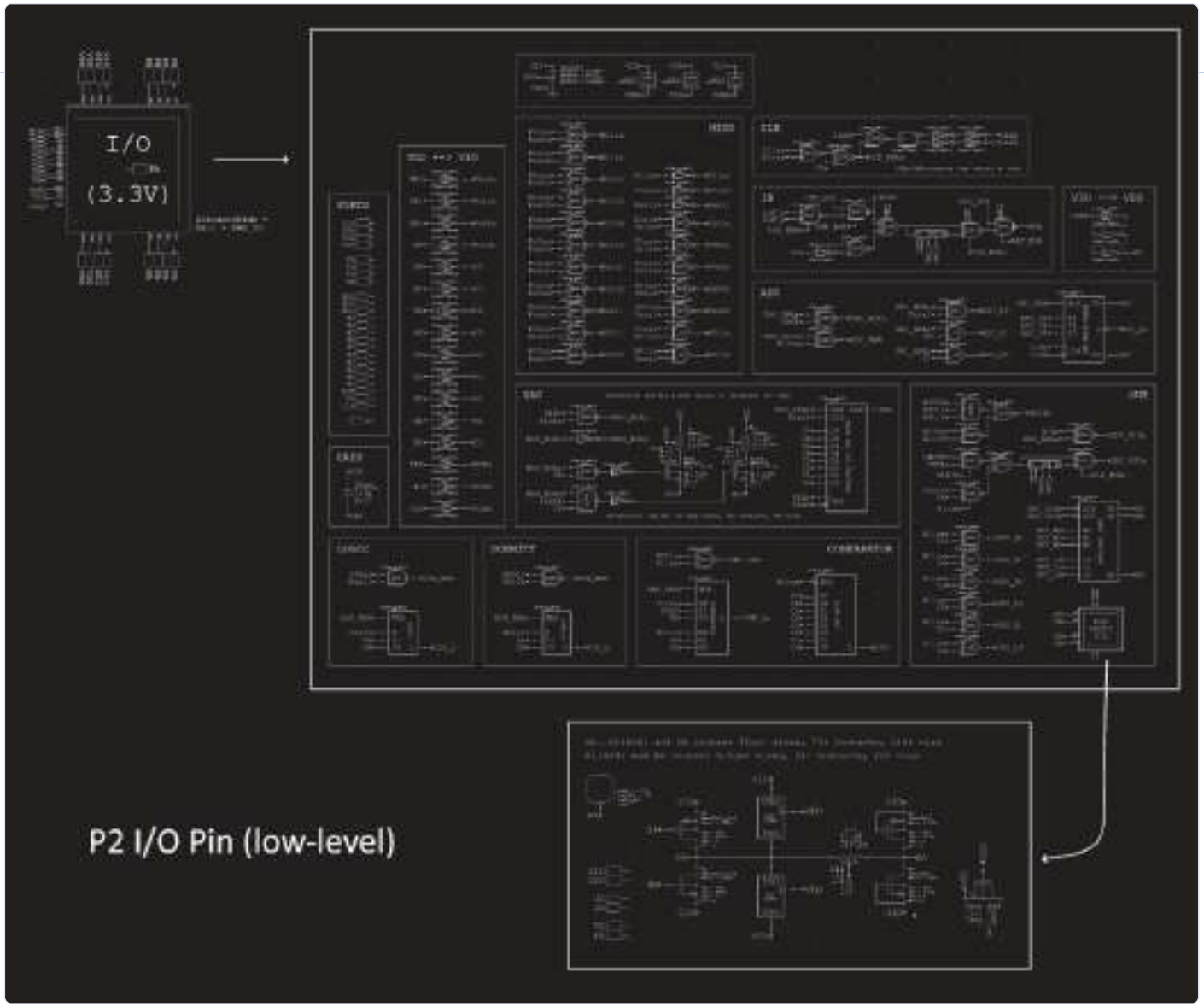


Figure 6. Description détaillée d'une broche du Propeller 2.

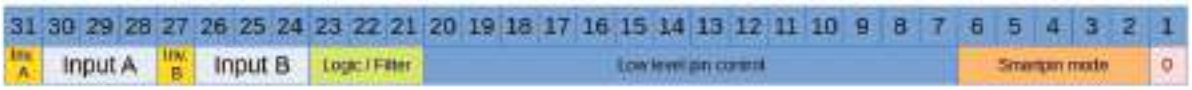


Figure 7. Registre de configuration des broches intelligentes.

très simple. Nous lisons l'état de la broche, et nous l'appliquons à la LED, quatre fois seulement, une fois pour chaque LED et bouton présent. De plus, si un bouton est actionné ou relâché, nous émettons une chaîne correspondante. Commençons par la configuration de l'entrée. Comme les boutons matériels ne possèdent pas leurs propres résistances de rappel au niveau haut ou bas, nous devons utiliser les résistances internes de la puce. À partir de la **figure 8** et de la figure 7, nous calculons la valeur du registre de configuration.

En gardant la figure 5 à l'esprit, nous n'aurons à la fin que le niveau logique présenté par notre broche. Nous sélectionnons donc les entrées A et B pour qu'elles soient identiques, non inversées et transférées vers la sortie A, sans aucune logique ou filtrage appliqués. D'après la fiche technique du Propeller 2, cela donne 0000 0000 000 000 010 010 00 00000 en binaire ou 0x900 en hexadécimal si nous voulons une résistance de rappel au niveau haut de 15 kΩ.
Pour notre code, cela signifie que nous ajoutons maintenant quatre entrées et utilisons quatre sorties en connectant notre petite carte d'extension (**fig. 9**) à la carte d'évaluation Propeller 2.

LIEN

[1] **Dépôt GitHub de cette série d'articles** : https://github.com/ElektorLabs/200479-propeller2-hands-on



Mode Groups	%PPPPPPPPPPPP	Input	Output	Notes	CIOHHHLLL Bits Explained
Non-ADC, Non-DAC Modes	0000_CIOHHHLLL	PinA Logic	OUT	Output enabled by DIR = 1 (otherwise floating)	Clocked mode I/O available with C:
	0001_CIOHHHLLL	PinA Logic	IN		
	0010_CIOHHHLLL	PinB Logic	IN		
	0011_CIOHHHLLL	PinA Schmitt	OUT		
	0100_CIOHHHLLL	PinA Schmitt	IN		
	0101_CIOHHHLLL	PinB Schmitt	IN		
	0110_CIOHHHLLL	PinA ^ PinB	OUT		
	0111_CIOHHHLLL	PinA ^ PinB	IN		
ADC Modes without DAC	100000_OHHHLLL	ADC, GIO 1x	OUT	Output enabled by DIR = 1 (otherwise floating)	
	100001_OHHHLLL	ADC, VIO 1x	OUT		
	100010_OHHHLLL	ADC, PinB 1x	OUT		
	100011_OHHHLLL	ADC, PinA 1x	OUT		
	100100_OHHHLLL	ADC, PinA 3.16x	OUT		
	100101_OHHHLLL	ADC, PinA 10x	OUT		
	100110_OHHHLLL	ADC, PinA 31.6x	OUT		
	100111_OHHHLLL	ADC, PinA 100x	OUT		
ADC + DAC Modes	10100_DDDCDDDD	ADC, PINA 1x	DAC 990 Ω, 3.3 V	ADC activated when OUT = 1	
	10101_DDDCDDDD	ADC, PINA 1x	DAC 600 Ω, 2.0 V	DDDDCDD = DAC Level	
	10110_DDDCDDDD	ADC, PINA 1x	DAC ±23.75 Ω, 3.3 V		
	10111_DDDCDDDD	ADC, PINA 1x	DAC 75 Ω, 2.0 V		
DAC Comparison Modes (Non-ADC)	1100_CDDDCDDDD	PinA = D	OUT, 1.5 kΩ	HHH=001, LLL=001 DDDDCDDD = DAC Level	
	1101_CDDDCDDDD	PinA = D	!IN, 1.5 kΩ		
	1110_CDDDCDDDD	PinB = D	IN, 1.5 kΩ	Output enabled by DIR = 1 (otherwise floating)	
	1111_CDDDCDDDD	PinB = D	!IN, 1.5 kΩ		

Clocked mode I/O available with C:

C	IN/OUT
0	Live
1	Clocked

Can invert polarity of I/O with I, O:

I	IN
0	Non-Inverted
1	Inverted

O	OUT
0	Non-Inverted
1	Inverted

Pull up and/or down with HHH/LLL:

HHH/LLL	Drive
000	Fast
001	1.5 kΩ
010	15 kΩ
011	150 kΩ
100	1000 µA
101	100 µA
110	10 µA
111	Float

Figure 8. Vue d'ensemble des modes de fonctionnement des broches du Propeller 2.

Le **listage 1** contient le code Spin2 modifié pour configurer nos quatre broches d'entrée avec des rappels au niveau haut ainsi que la configuration de sortie pour nos broches de sortie LED.
Nous utilisons quatre variables globales pour stocker le dernier niveau de broche lu et envoyer un message série, uniquement si une broche a changé. Nous étendons le code avec deux fonctions, la première pour initialiser les LED et la seconde pour initialiser nos entrées. Le **listage 2** montre les quatre octets déclarés pour stocker l'état des quatre interrupteurs connectés à la carte.
Vous pouvez également voir que le code a été modifié avec les deux fonctions d'initialisation (**listage 3**). La magie du fonctionnement des entrées avec la capacité de rappel au niveau haut est enfouie dans les bits décrits dans le **listage 4**.
Nous configurons l'entrée et la sortie pour qu'elles soient pilotées à l'aide d'une résistance de 15 kΩ. Avec la dernière ligne, nous configurons nos quatre broches d'entrée pour qu'elles soient fixées au niveau bas, car nous les utilisons effectivement comme sorties.

Une fois l'initialisation effectuée, la fonction `read_switch()` est appelée à plusieurs reprises afin que nous puissions examiner de plus près le code qui se déroule dans cette fonction. Comme on le voit dans le **listage 5**, le code utilise un certain nombre de structures IF et IF-ELSE après avoir lu les états des broches dans les variables `a` à `d`. La première chose à faire est de détecter si une broche a changé en comparant l'ancien état mémorisé avec le nouvel état lu. Si ces valeurs diffèrent, nous pouvons supposer que l'état de l'entrée de la broche a changé et déterminer si c'est parce que l'interrupteur correspondant a été actionné ou relâché. En fonction de l'état actuel de la broche, nous utilisons la fonction `prints()` pour afficher la mention « Switch pressed » (bouton pressé) ou « Switch released » (bouton relâché). Le **listage 5** montre comment notre fonction `read_switch()` se présente et fonctionne, avec de simples chemins IF-THEN-ELSE pour chaque

Listage 1. Configuration des quatre broches d'entrée.

```
WRPIN(52, %0000_0000_000_000_010_010_00_00000)
WRPIN(53, %0000_0000_000_000_010_010_00_00000)
WRPIN(54, %0000_0000_000_000_010_010_00_00000)
WRPIN(55, %0000_0000_000_000_010_010_00_00000)
```

Listage 2. Déclaration des octets de stockage des états des boutons.

```
VAR BYTE PinA, PinB, PinC, PinD
```

Listage 3. Fonctions d'initialisation ajoutées.

```
PUB main()

  LED_init()
   Input_init()

  pinwrite(56, 1 )
  serial_start()
  repeat
     read_switch()
```

Listage 4. Initialisation des entrées.

```
PUB Input_init( )
' A and B as input using A with 15k drive
' strength for pull-up and pull-down as output no
' smartpin function
   WRPIN(52, %0000_0000_000_000_010_010_00_00000)
   WRPIN(53, %0000_0000_000_000_010_010_00_00000)
   WRPIN(54, %0000_0000_000_000_010_010_00_00000)
   WRPIN(55, %0000_0000_000_000_010_010_00_00000)
   PINWRITE(55..52,0)
   'Drive pin low with 15k resistance
```

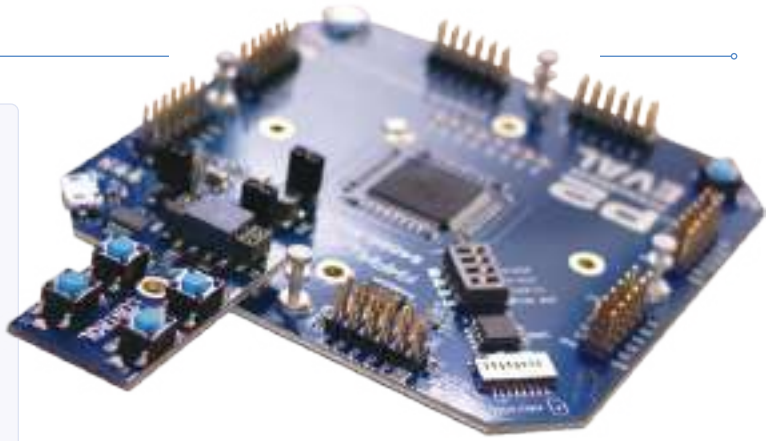


Figure 9. Propeller 2 avec la platine des boutons raccordée.

Listage 5. Fonction read_switch( ).

```
PUB read_switch( ) | a,b,c,d
 'We will read the pin, reflect the LED state
 'We send a change in state with the UART
  a := pinread(52)
  b := pinread(53)
  c := pinread(54)
  d := pinread(55)
  IF(PinA <> a )
     PinA := a
     prints(string("A:"))
     IF( a <> 0 )
       pinwrite(49,1)
       prints(@PinPressed)
     ELSE
       pinwrite(49,0)
       prints(@PinReleased)
  IF(PinB <> b )
     PinB := b
     prints(string("B:"))
     IF( b <> 0 )
       pinwrite(48,1)
       prints(@PinPressed)
     ELSE
       pinwrite(48,0)
       prints(@PinReleased)
  IF(PinC<> C )
     PinC := c
     prints(string("C:"))
     IF( c <> 0 )
       pinwrite(50,1)
      prints(@PinPressed)
     ELSE
       pinwrite(50,0)
       prints(@PinReleased)
  IF(PinD <> d )
     PinD := d
     prints(string("D:"))
     IF( d <> 0 )
       pinwrite(51,1)
       prints(@PinPressed)
     ELSE
       pinwrite(51,0)
       prints(@PinReleased)
```

bouton utilisé. Vous pouvez également y lire que l'état des LED est modifié en fonction du nouvel état détecté.
Pour l'instant, ce sera tout avec les broches, mais nous espérons que cela vous a donné un petit aperçu de leur fonctionnement. Cette série se termine avec cet article. Toutefois le Propeller 2 reste sur ma paillasse, j'écrirai encore à son sujet ou je ferai une courte vidéo. Comme le logiciel et l'EDI sont appelés à évoluer, nous laissons à ce microcontrôleur le temps de grandir. En attendant, vous pouvez accéder au code écrit pour cette série dans notre dépôt GitHub [1].

Première impression finale

Après avoir travaillé avec le Propeller 2, je peux dire qu'il s'agit d'un puissant microcontrôleur qui nécessite sans aucun doute d'accorder plus d'attention à ses caractéristiques qu'un Raspberry Pi Pico. L'utilisation du langage Spin2 et de l'assembleur est particulière, et rend impossible la réutilisation du code C existant. Personnellement, je pense que ce ne sera pas mon microcontrôleur de prédilection pour les projets courants, mais cette puce pourra certainement être utile pour un certain nombre d'applications de niche.
Qu'en est-il des interfaces SPI, I²C et HDMI mentionnées au début de cette série d'articles ? Au fur et à mesure des évolutions apportées aux chaînes d'outils, il serait intéressant de les finaliser et de passer à des outils améliorés. Impatient d'en savoir plus sur le Propeller 2 ? Envoyez-nous un courriel ou utilisez l'un de vos canaux de réseaux sociaux préférés pour nous envoyer un message.

200479-E-04

Contributeurs

Conception et texte : **Mathias Claußen**
Rédaction : **Jens Nickel, C. J. Abate**
Mise en page : **Giel Dols**
Traduction : **Pascal Godart**

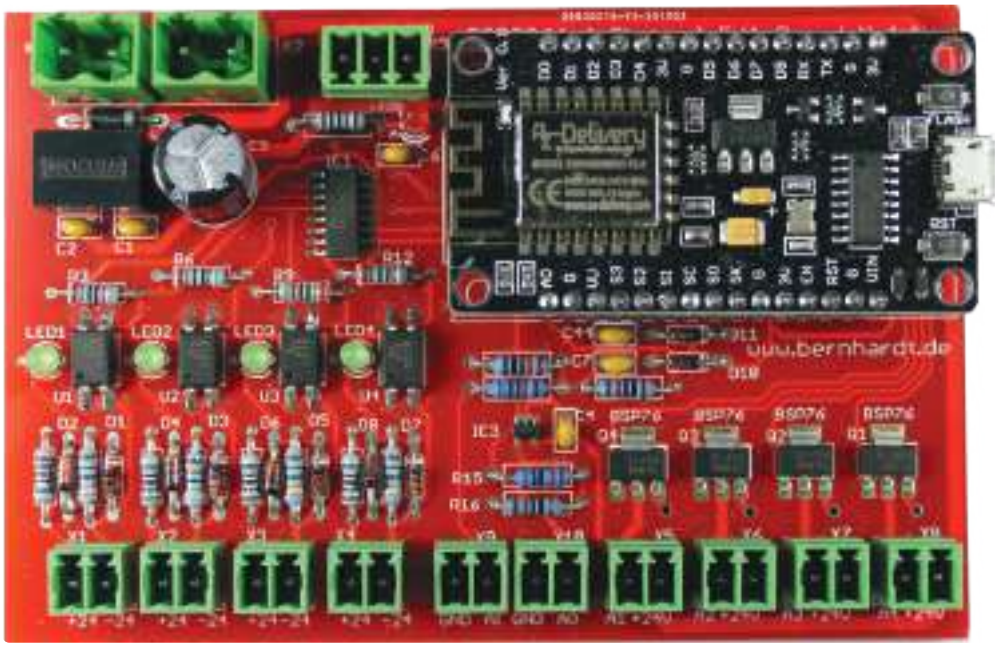
Des questions, des commentaires ?

Envoyez un courriel à l'auteur (mathias.claussen@elektor.com) ou contactez Elektor (redaction@elektor.fr).

Modbus sans fil (partie 1)

Matériel et programmation

Josef Bernhardt (Allemagne) et
Martin Mohr (Allemagne)



Carte Modbus avec module NodeMCU.

Le protocole Modbus est très utilisé dans l'industrie pour la communication entre systèmes et contrôleurs. En pratique, il met généralement en œuvre des interfaces filaires RS485, éprouvées et fiables. Nous présentons ici un module qui permet d'utiliser le protocole Modbus sur un réseau local sans fil. Le module est construit autour d'une carte Espressif NodeMCU à microcontrôleur ESP8266. Une carte de base Modbus complémentaire permet de travailler avec des signaux de 24 V très courants en environnement industriel. Pour illustrer le fonctionnement, les auteurs ont réalisé la commande d'un jouet, une porte d'ascenseur.

Pour la plupart des lecteurs d'*Elektor*, le module *NodeMCU* d'*Espressif* et l'*EDI Arduino* sont déjà familiers. Si c'est votre cas, vous pouvez sauter l'introduction et passer à la description de la carte *Modbus TCP*. Sinon, voici ce que vous devez savoir en quelques mots.

Ce projet est conçu autour d'un module *NodeMCU* (disponible dans la boutique Elektor). Ce module est équipé d'un microcontrôleur *Espressif ESP8266*, de la taille d'un timbre-poste, il est doté d'une interface réseau local sans fil (*WLAN* ou *Wireless LAN* en anglais). Malgré sa petitesse, sa puissance de traitement est grande. Le **tableau 1** résume les principales caractéristiques du microcontrôleur ESP8266. La carte NodeMCU produit la tension d'alimentation de l'ESP8266 et gère l'interface de programmation du microcontrôleur. La **figure 1** donne le brochage de la carte NodeMCU utilisée dans notre circuit Modbus.

L'EDI Arduino est très bien adapté à la programmation de la carte NodeMCU. Le site web d'Arduino [2] permet de télécharger gratuitement la version de l'EDI Arduino

adaptée au système d'exploitation de votre ordinateur et de l'installer en quelques instructions. Si vous exécutez l'EDI pour la 1re fois, vous voyez une fenêtre comme celle de la **figure 2**. Le volet du code du programme contient deux fonctions prédéfinies : la fonction `setup()` qui est exécutée une seule fois au démarrage du programme. Elle gère – entre autres – l'initialisation des interfaces du microcontrôleur. Puis vient la fonction `loop()` qui accueille normalement le code source du programme.

La fonction `loop()` s'exécute à la suite de la fonction `setup()`. Si le programme atteint la fin de la fonction `loop()`, il recommence depuis le début. L'ESP8266 gère l'interface WLAN **entre la fin et le redémarrage** de la fonction `loop()`. Cela signifie qu'il faut éviter de créer des boucles infinies dans la fonction `loop()`, sinon l'ESP8266 produira une erreur. Par conséquent, le code de la fonction `loop()` doit être conçu pour s'exécuter de manière cyclique.

Un grand nombre d'erreurs mystérieuses de l'ESP8266 s'expliquent par le fait que le processeur n'obtient pas assez de temps CPU pour gérer l'interface WLAN. Si l'exécution du programme peut prendre un temps excessif, par ex. en raison de grandes boucles, mieux vaut utiliser la fonction `yield()` ou la fonction `delay()` pour accorder à l'ESP8266 le temps requis pour gérer le WLAN.

Pour brancher le module NodeMCU sur la carte Modbus, il faut le connecter à un port USB de l'ordinateur, mais au préalable, quelques opérations préparatoires sont indispensables. L'EDI Arduino standard ne prend pas l'ESP8266 en charge, il faut donc d'abord le mettre à jour. Pour ce faire, sélectionner *Fichier* -> *Préférences* et dans la zone URL du gestionnaire de cartes supplémentaires, saisissez :

http://arduino.esp8266.com/stable/package_esp8266com_index.json

Cliquez sur *OK*, puis sélectionnez *Outils* -> *Type de carte …* -> pour ouvrir la fenêtre *Gestionnaire de cartes*, et recherchez *ESP8266* puis installez les cartes de la *Communauté ESP8266*.

Après cette installation, la carte *NodeMCU 1.0 (Module ESP-E12)* figure sous *Outils* -> *Type de carte* > et sous *Outils* -> *Port* > figure le port auquel la carte NodeMCU est connectée (*/dev/ttyUSBx* sous Linux ou *COMx* sous Windows).

Dès lors, le premier programme simple d'essai du module NodeMCU peut être exécuté. À cet effet, ouvrez le programme d'exemple *Blink* sous *Fichier* -> *Exemples* -> ESP8266 -> *Blink*. Le programme du **listage 1** fait clignoter la LED installée sur la carte ESP8266. La fonction `loop()` accueille les instructions d'allumage/extinction de la LED. La fonction `setup()` configure la broche GPIO de la LED en sortie.

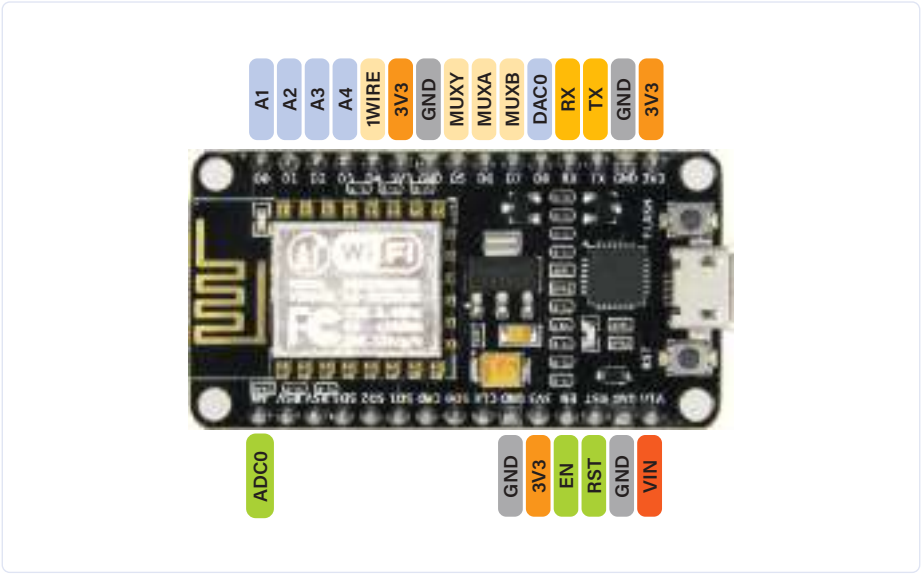


Figure 1. Brochage du module NodeMCU pour le projet Modbus.

Tableau 1. Caractéristiques techniques de l'ESP8266.

CPU RISC 32 bits	Tensilica Xtensa LX106 cadencé à 80 MHz
Mémoire de programme	64 Ko
Mémoire de données	96 Ko
Mémoire flash externe (en option)	512 Ko à 4 Mo
GPIO	10 broches ; interfaces prises en charge : SPI, I²C, I²S, UART. DMA et IRQ sont aussi prises en charge.
Wi-Fi	- IEEE 802.11b/g/n - Authentification WEP ou WPA/WPA2 - Wi-Fi direct (P2P), Soft AP - Pile de protocole TCP/IP intégrée
CA/N	1× convertisseur 10 bits
Antenne	intégrée
Puissance de sortie	+19,5 dBm en mode 802.11b
Taille	17,2 × 12,3 mm²
Courant de veille	< 1,0 mA

Listage 1. Programme de LED clignotante pour l'ESP8266.

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(2000);
}
```



Pour charger le programme dans la carte NodeMCU, cliquez sur l'icône *Téléverser* (flèche vers la droite). Au bout de quelques secondes, le programme est transféré puis la LED se met à clignoter. Ce test valide l'installation et la configuration de l'EDI.

Figure 2. Fenêtre initiale de l'EDI Arduino.

Carte Modbus TCP

Toutes les entrées/sorties sont réalisées avec des borniers à vis enfichables ce qui confère un aspect industriel (v. photo en tête d'article) à la carte Modbus TCP qui accueille le module NodeMCU. Cela simplifie l'assemblage et permet de remplacer facilement les bornes à vis en cas de détérioration. La principale tâche de la carte Modbus TCP est de convertir le niveau de signal de 3,3 V des entrées/sorties du microcontrôleur en 24 V, compatible avec

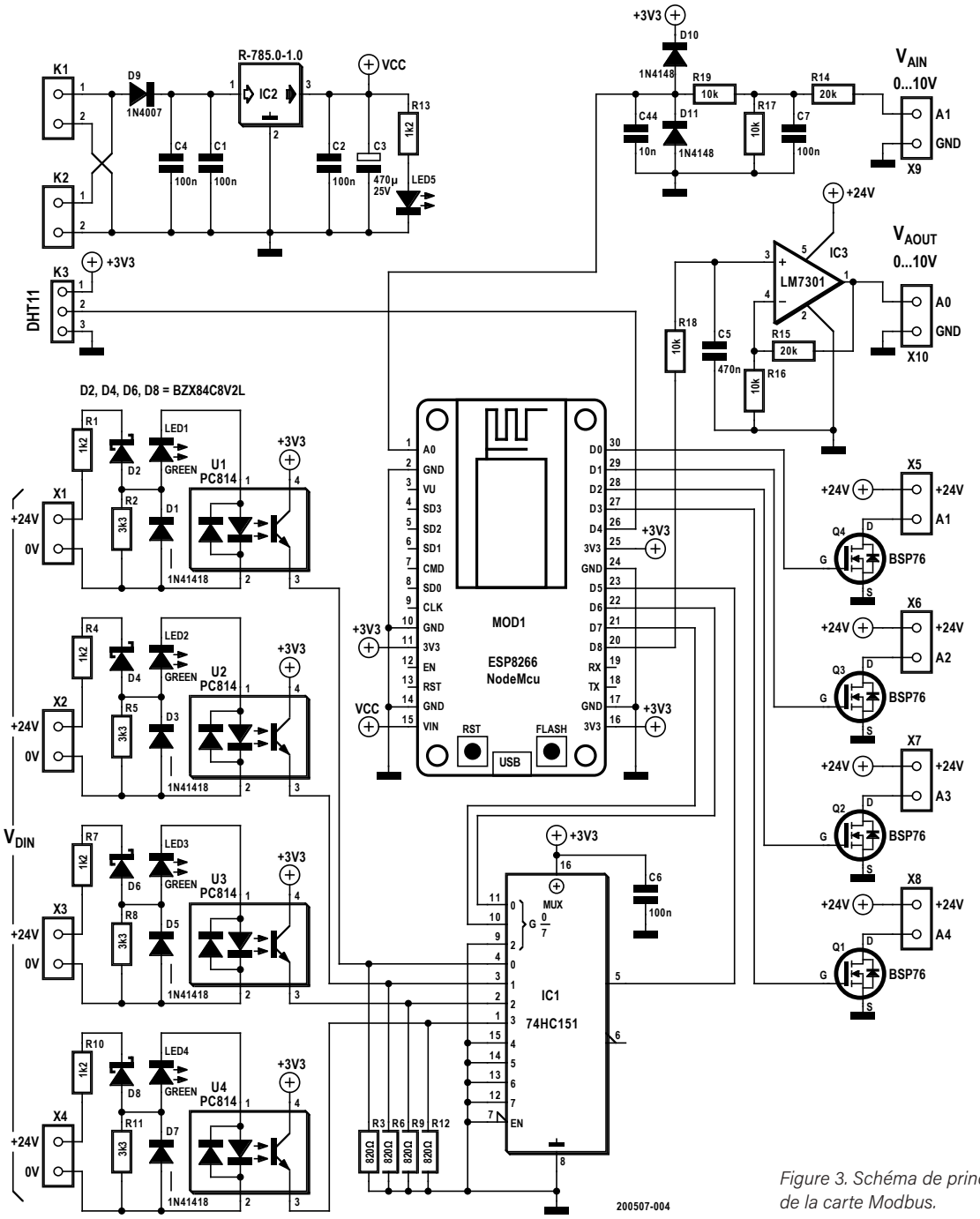


Figure 3. Schéma de principe de la carte Modbus.

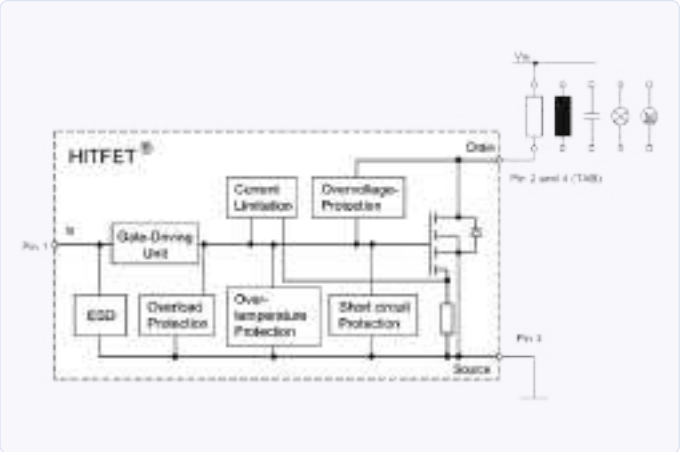


Figure 4. Structure interne du BSP76 (source : fiche technique Infineon).

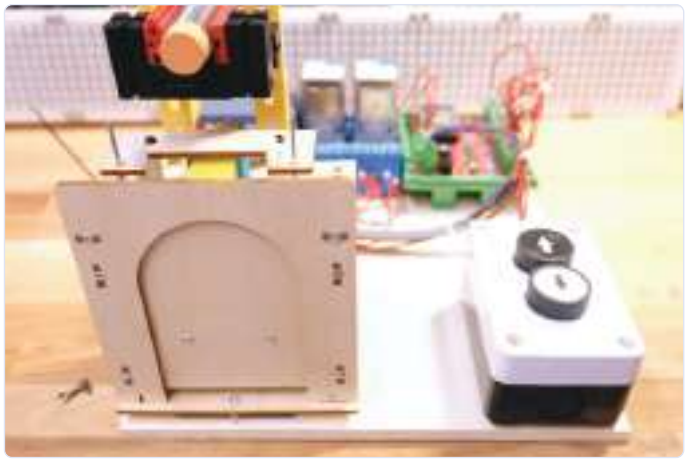


Figure 5. Porte d'ascenseur de maquette, en bois.

l'environnement industriel. Le schéma de la **figure 3** montre que les quatre entrées sont isolées galvaniquement du microcontrôleur par des opto-isolateurs PC814. En outre, ces entrées ne sont pas reliées directement à la carte ESP8266, mais à travers un multiplexeur 74HC151. Cette astuce permet d'économiser une broche GPIO et d'équiper le circuit d'un capteur DHT11 pour mesurer la température et l'humidité. La carte Modbus est donc capable d'exécuter des tâches simples de contrôle CVAC (Chauffage Ventilation Air Conditionné).

Les étages de sortie font appel à des *BSP76* de la série *HITFET* d'*Infineon* (**fig. 4**). Ces composants de commutation électronique ne sont pas de simples MOSFET de puissance, mais des circuits intégrés dotés de protections contre les surtensions et la surchauffe et qui limitent le courant. Ils sont donc capables de commuter indifféremment des charges résistives, inductives ou capacitives.

En plus des entrées et sorties numériques, la carte Modbus TCP a une entrée et une sortie analogiques. Ces deux ports fonctionnent sur la gamme de tension standard de l'industrie, soit 0 à 10 V. Le diviseur de tension R14/R17 réduit la tension à un maximum de 3,3 V sur l'entrée analogique. Cette entrée n'est pas isolée galvaniquement, mais C7, D10 et D11 assurent une protection notable contre les crêtes et les surtensions. Les résistances série R14 et R19 limitent le courant d'entrée pour que les diodes ne partent pas en fumée en cas de problème sur l'entrée.

Sur la sortie analogique, l'*AOP IC3* à usage général (*LM7301* avec capacité rail à rail en entrée comme en sortie) augmente la tension de sortie du *CA/N* du NodeMCU. Les résistances R15 et R16 confèrent un gain de 3 à cet AOP par ailleurs alimenté par le rail 24 V.

Le convertisseur DC/DC de *Recom* mérite une mention spéciale. Il abaisse la tension d'alimentation de 24 V aux 5 V nécessaires à l'alimentation du module NodeMCU. Le rendement de ce convertisseur DC/DC est largement supérieur à 90 %, et son utilisation est très polyvalente. Il peut par ex. être utilisé pour produire une tension d'alimentation négative pour les AOP. Cela vaut la peine de jeter un coup d'œil à la fiche technique du convertisseur DC/DC [4]. La diode *1N4007* montée en série sur l'entrée 24 V protège le circuit contre l'inversion de polarité de connexion.

Les fichiers *Gerber* de la carte imprimée peuvent être téléchargés depuis la page du projet [8]. La carte est disponible auprès de l'auteur [7] soit nue, soit entièrement assemblée et prête à l'emploi.

Démonstration à l'aide d'une maquette en bois

Pour montrer la capacité du module Modbus à gérer une tâche industrielle type, les auteurs ont utilisé une porte d'ascenseur de maquette provenant du magasin en ligne chinois bien connu [5] (**fig. 5**). Un bouton-poussoir ouvre la porte, l'autre la ferme. Deux capteurs de proximité capacitifs détectent les positions extrêmes de la porte. Le moteur qui ouvre et ferme la porte est alimenté par un pont en H basé sur deux relais de puissance.

Le **listage 2** donne le programme de commande. Celui-ci définit d'abord des mnémoniques pour les entrées et sorties du

LIENS

[1] Carte NodeMCU : www.elektor.fr/17952
[2] Téléchargement de l'EDI Arduino : www.arduino.cc/en/software
[3] Fiche technique BSP76 : https://bit.ly/3r3GP99
[4] Fiche technique du convertisseur DC/DC : https://recom-power.com/pdf/Innoline/R-78-0.5.pdf
[5] Porte d'ascenseur de maquette, en bois : www.aliexpress.com/item/33008243573.html
[6] Porte d'ascenseur de maquette, en bois sur YouTube : https://youtu.be/VHIBQswdA0E
[7] Josef Bernhardt : www.bernhardt.de
[8] Page du projet : www.elektormagazine.fr/200507-04

Listage 2. Commande de porte d'ascenseur.

```
#define A1 16
#define A2 5
#define A3 19
#define A4 0
#define MUXA 12
#define MUXB 13
#define MUXY 14
#define E1 0
#define E2 1
#define E3 2
#define E4 3


void setup() {
  pinMode(A1, OUTPUT);
  pinMode(A2, OUTPUT);
  pinMode(A3, OUTPUT);
  pinMode(A4, OUTPUT);
  pinMode(MUXA, OUTPUT);
  pinMode(MUXB, OUTPUT);
  pinMode(MUXY, INPUT);
}


bool input(int i){
  digitalWrite(MUXA,(i&1));
  digitalWrite(MUXB,(i&2)>>1);
  delay(1);
  return digitalRead(MUXY);
}


void output(int i, int v){
  digitalWrite(i,v);
}


void loop() {
if(input(E4)& !input(E1)){output(A1,HIGH);}
if(input(E3)& !input(E2)){output(A2,HIGH);}
if(input(E1)){output(A1,LOW);}
if(input(E2)){output(A2,LOW);}
}
```

circuit. Les différentes broches d'E/S sont initialisées comme entrées et sorties par la fonction `setup()`. Viennent ensuite les fonctions `input()` et `output()` qui facilitent un peu l'accès aux broches d'entrée/sortie. La fonction `loop()` constitue le programme principal où les sorties de commande du moteur sont activées par les boutons et désactivées par les capteurs « fin de course ». La fonction `loop()` peut être exécutée de manière répétitive. Pour voir la porte d'ascenseur en action, il suffit de regarder cette vidéo *YouTube* [6].

Perspective

Malgré sa valeur démonstrative, la commande de porte d'ascenseur réalisée ne rend pas compte du potentiel de la carte Modbus et du module WLAN. Nous y reviendrons dans la 2e partie de cet article, où nous nous familiariserons avec le protocole Modbus et parlerons des logiciels utilisables pour communiquer sur le bus. Nous montrerons également comment configurer le module Modbus.

200507-04

Contributeurs
Conception et article :
Josef Bernhardt et **Martin Mohr**

Rédaction : **Rolf Gerstendorf**
Traduction : **Yves Georges**
Mise en page : **Giel Dols**

Des questions, des commentaires ?
Envoyez un courriel à l'auteur (josef@bernhardt.de) ou contactez Elektor (redaction@elektor.fr).

À propos des auteurs

Josef Bernhardt s'est intéressé à l'électronique dès son plus jeune âge. Il a construit son premier poste à galène à l'âge de douze ans, puis a continué avec d'autres circuits. Dans les années 1980, il fit ses premières expériences en programmation sur le Commodore VC20, et il connaît également bien les techniques de l'assembleur sur 8088. Il peut se prévaloir de plus de 30 ans d'expérience en électronique à l'Université de Ratisbonne (Allemagne), où il a œuvré dans le développement électronique et logiciel. Grâce à sa propre installation de fabrication CMS, il réalise également des projets électroniques pour des clients.

Martin Mohr vit le jour à l'époque des mémoires à tores magnétiques et des interrupteurs Strowger, ce qui lui permit de vivre personnellement toute l'épopée des techniques informatiques modernes. Sa fascination pour tout ce qui clignote remonte à sa prime jeunesse et a été renforcée par sa formation en électronique. Après des études en informatique, il a surtout travaillé au développement d'applications Java, mais le Raspberry Pi a ravivé sa passion pour l'électronique.



PRODUITS

› **NodeMCU-Modul**
www.elektor.fr/nodemcu-microcontroller-board-with-esp8266-and-lua

embeddedworld2022

Exhibition&Conference

... it´s a smarter world

20 years

2003 - 2022

INTELLIGENT.
CONNECTED.EMBEDDED.

JOIN THE GLOBAL PLATFORM
OF THE EMBEDDED COMMUNITY
15 – 17.3.2022

You simply have to be there!
embedded-world.com

Media partners

Markt&Technik
Elektronik
SmarterWorld
Computer & AUTOMATION
DESIGN& ELEKTRONIK
Elektronik automotive
medical-design
elektroniknet.de

NÜRNBERG MESSE

Junior Computer

en forme après 40 ans de sommeil

Laurent Françoise (France) et **Eric Bogers** (Elektor)

En 1980, l'informatique restait *terra incognita* aux yeux de bon nombre d'électroniciens amateurs. Les ordinateurs, ces drôles d'appareils, leur semblaient aussi coûteux qu'effrayants. Et d'ailleurs, à quoi pouvaient-ils bien servir ? Elektor transforma ce territoire inexploré en terre promise en dévoilant dans le numéro de mai le *Junior Computer* [1], un « ordinateur adulte pour débutants » que le lecteur pouvait (devait) assembler lui-même – pour environ mille francs de l'époque.



Figure 1. L'acte de naissance du Junior Computer.

La description complète du Junior Computer *(JC pour les intimes) n'aurait pu tenir dans les pages du magazine (**fig. 1**), aussi fut-elle répartie à travers plusieurs ouvrages publiés par Elektor. Cette série en quatre tomes n'est plus disponible depuis longtemps, mais quelques passionnés (que l'on salue !) en ont numérisé certaines parties et les ont publiées sur l'internet. En 2005, l'inégalable rubrique **Rétronique** nous a remis en mémoire ce bon vieux « JC » [2].*
En France, Laurent Françoise, dont le passe-temps favori est la restauration de vieux ordinateurs, a décidé de redonner vie au Junior Computer. *Laissons-lui la parole :*

« J'ai grandi dans les années 80, et c'est de mon père (qui était radioamateur) que vient mon amour pour l'électronique et les ordinateurs. Même si je ne comprenais pas tout à l'époque, je feuilletais régulièrement ses magazines Elektor, et leur lecture aura joué un grand rôle dans ma vie. La collection de mon père partait du premier numéro, et je possède encore bon nombre des projets qu'il a construits à l'époque (Elekterminal, micro-ordinateur BASIC, ElektorScope).
C'est en retombant sur le premier des quatre tomes décrivant le *Junior Computer* que j'ai eu envie de l'assembler. Pouvoir en commander les circuits imprimés d'époque (avec la fameuse couleur bleue d'Elektor) aurait été formidable, mais ils sont aujourd'hui introuvables. »

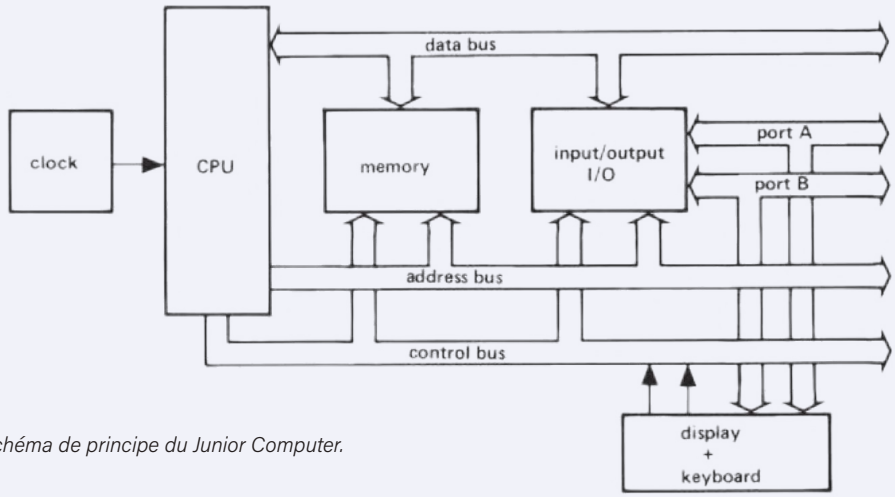


Figure 2. Schéma de principe du Junior Computer.

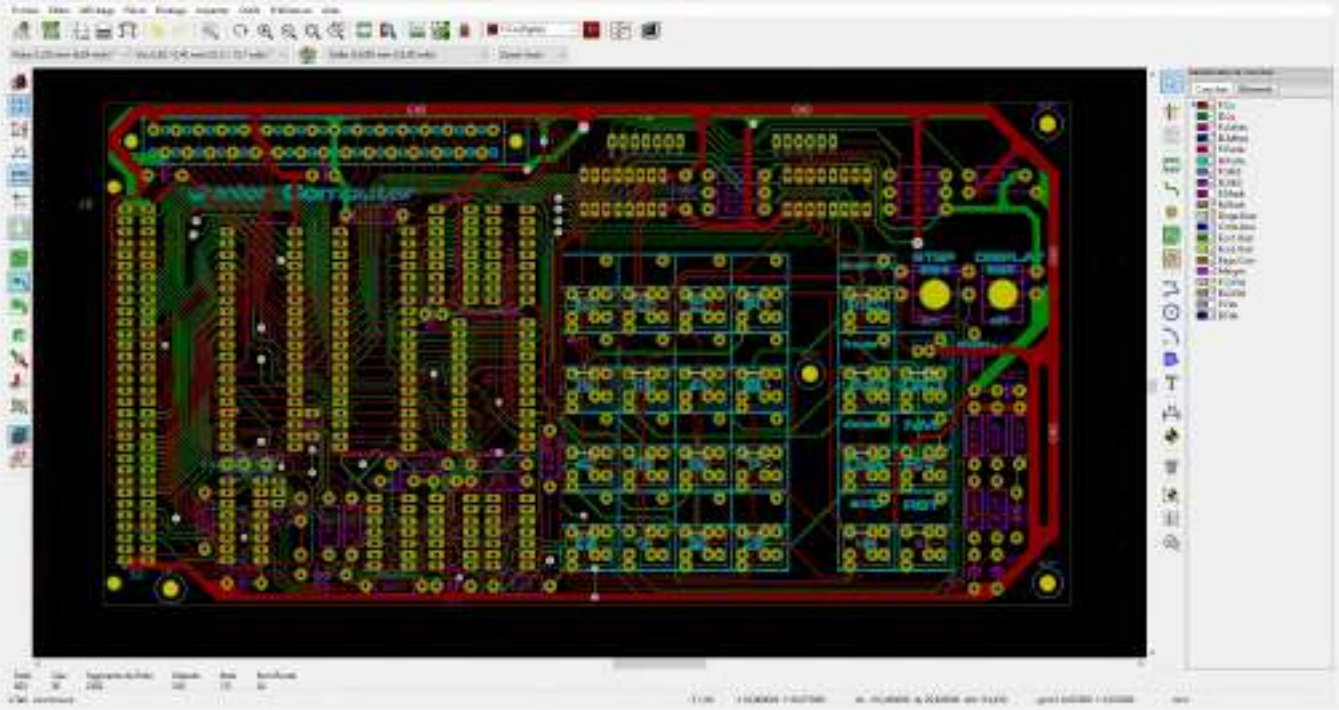


Figure 3. Reconstruction du circuit sous KiCAD.

Interrompons brièvement Laurent pour rappeler ce qu'était le Junior Computer *: un ordinateur monocarte doté d'un microprocesseur 6502, d'une ROM de 1 Ko, d'une RAM de 1 Ko (+ 128 octets de RAM dans la puce d'E/S 6532), d'un petit clavier (comportant les célèbres boutons-poussoirs Digitast), d'un afficheur à 7 segments et 6 chiffres, et d'un connecteur latéral relié au bus système – pratique pour les extensions ultérieures (et il y en eut beaucoup). Son schéma de principe est reproduit sur la* ***figure 2*** *(désolé pour la piètre qualité du scan).*

« Comment donc réaliser ces anciens circuits imprimés ? À l'époque la procédure consistait à copier sur un film transparent tel ou tel schéma en noir et blanc du magazine, à placer ce typon sur une plaque couverte de résine photosensible, puis à procéder à l'insolation, au développement, et enfin à la gravure du circuit. Même la fabrication maison de circuits imprimés à double face n'était pas insurmontable, mais la présence de trous métallisés rendait celle du Junior impossible. J'aurais pu en théorie réaliser ces trous métallisés à l'aide de fils de cuivre fins, mais à mes yeux c'était chercher les ennuis.
J'ai donc redessiné les schémas des CI et confié leur réalisation à un fabricant professionnel. Comme je souhaitais reproduire l'original aussi fidèlement que possible, j'ai d'abord réalisé les fameuses

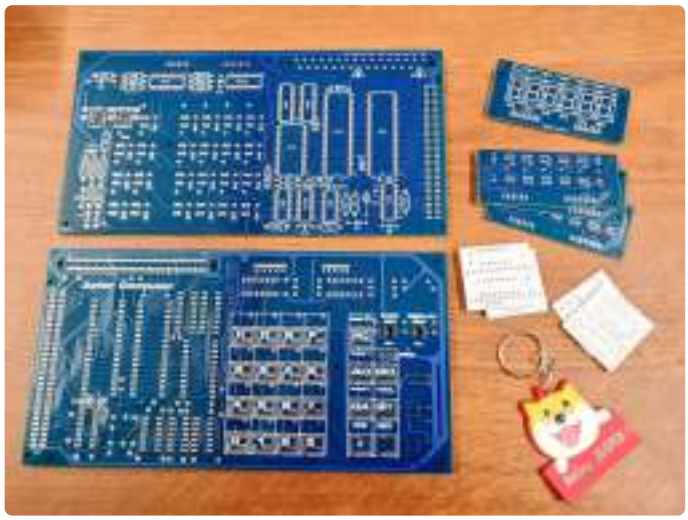


Figure 4. Les CI reconstitués du Junior Computer – la différence avec les originaux est à peine perceptible.



Figure 6. Le Junior Computer assemblé par Laurent. Tout fonctionne !

pastilles octogonales d'Elektor dans KiCAD, ainsi que les empreintes nécessaires, elles aussi caractéristiques d'Elektor. J'ai ensuite importé un scan du schéma original en tant que couche pour m'assurer que la position des composants correspondait au mieux à celle d'origine (**fig. 3**). Tout ce travail s'est avéré étonnamment facile sous KiCAD. J'ai effectué manuellement le routage des pistes. La **figure 4** montre le résultat de tous ces efforts – notez le « bleu Elektor ». Une question m'a longtemps hanté durant le projet : voulais-je la réplique exacte du *Junior Computer* original, ou quelques adaptations étaient-elles acceptables ? Après mûre réflexion, j'ai finalement décidé d'apporter à l'ordinateur certaines des améliorations et corrections publiées ultérieurement dans Elektor. J'ai en outre remplacé l'EPROM 2708 originale par une 2716 : d'abord parce qu'elle est plus facile à trouver sur eBay que la 2708, ensuite parce qu'elle ne requiert qu'une tension d'alimentation de +5 V, alors que la 2708 attend aussi –5 V et +12 V.

Ce choix d'une réplique modifiée m'a bien sûr obligé à redessiner le schéma originel dans KiCAD, mais ce ne fut là qu'une formalité. La **figure 5** montre une partie de ce nouveau schéma.
J'ai gardé le plus difficile pour la fin : le lettrage des boutons-poussoirs « Digitast ». Chance inouïe, j'ai pu disposer des lettres-transferts originales de Mecanorma, ce qui m'a permis de recréer un clavier identique à l'original. Je vous laisse admirer le résultat final sur la **figure 6**.
L'assemblage a été assez simple. Le microprocesseur 6532 s'est révélé défectueux au démarrage, mais comme j'en avais commandé deux, j'ai pu le remplacer sur-le-champ. Ensuite tout a fonctionné. J'ai testé le *Junior* en profondeur, notamment avec un petit programme affichant « Junior » sur l'afficheur à 7 segments. Quel plaisir d'avoir pu mener à bien ce projet ! Et comme de nombreuses extensions pour le *Junior Computer* ont été publiées (p. ex. des cartes vidéos et un contrôleur de disquette), il est probable que je continue à m'amuser avec pendant un certain temps encore ! »

210328-04

Des questions, des commentaires ?

Contactez Elektor (redaction@elektor.fr).

Contributeurs

Texte et illustrations : **Laurent Francoise**
Rédaction : **Eric Bogers**
Traduction : **Hervé Moreau**
Maquette : **Harmen Heida**

LIENS

[1] Junior Computer, Elektor, 05/1980 : https://www.elektormagazine.fr/magazine/elektor-198004/51229
[2] Junior Computer d'Elektor (Rétronique), Elektor, 01/2005 : https://www.elektormagazine.fr/magazine/elektor-200501/10053

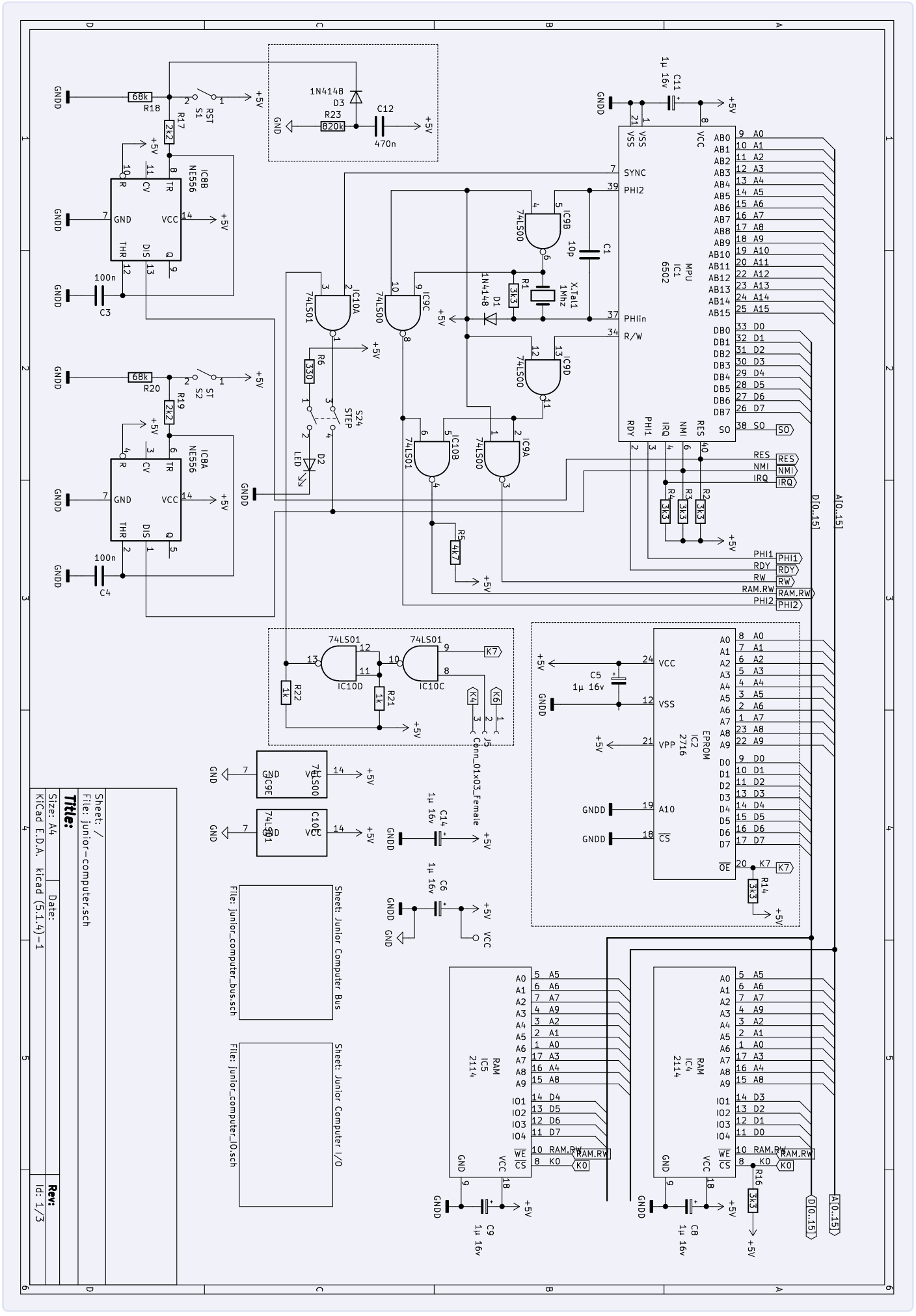


Figure 5. Une des parties du schéma légèrement modifié par Laurent.

construire son propre étalonneur **de haute précision**

–10 V à +10 V, 0 à 40 mA, 0,001 %

Vincent Gautier (France)

Lorsqu'on répare ou règle des circuits électroniques, on peut avoir besoin d'une référence de tension ou de courant de haute précision. Pour tester vos montages ou servir de source d'étalonnage pour des appareils tels que multimètres et oscilloscopes, l'étalonneur présenté ici délivre des tensions précises de –10 V à +10 V par pas de 20 µV et des courants précis de 0 à 40 mA par pas de 100 nA.

Caractéristiques et spécifications

- Tension de sortie 0 à ±10 V avec une résolution de 20 µV, max. 5 mA
- Courant 0 à 40 mA, résolution 100 nA, tension en circuit ouvert 12 V
- Précision de 10 ppm (10 parties par million) à 21 °C
- Facile à étalonner par logiciel
- Alimentation par USB (batterie en option)
- Écran tactile couleur de 3,5 pouces



Cet article est basé sur le projet que Vincent Gautier présente sur le site du labo d'Elektor [1], où l'on peut trouver ou télécharger des informations plus détaillées sur son étalonneur, y compris tous les logiciels, les circuits imprimés et les listes de composants, ainsi que les fichiers d'impression 3D. Une vidéo présentant ce projet est visible à l'adresse [2].

J'ai acheté un oscilloscope en panne sur eBay. Pour en profiter pleinement après réparation, il fallait l'étalonner avec des tensions et des courants précis. Normalement, les valeurs d'étalonnage nécessaires sont de l'ordre de 1,000 V ou 10,000 mA. Mais pour cet oscilloscope, d'après les procédures d'étalonnage fournies par le fabricant, des références de tension et de courant spécifiques étaient nécessaires : +1,7694 V et –1,7694 V, et 20,253 mA.
Il y a de nombreuses solutions d'étalonnage sur le marché, mais ces appareils dépassent largement mon budget ou n'ont pas la résolution et la précision requises. La solution a donc été pour moi de construire mon propre étalonneur.

Le matériel

Le montage se décompose en six blocs principaux :

- Alimentation électrique
- Référence de tension
- CN/A (Convertisseur Numérique/Analogique) autour de l'AD5791
- Générateur de courant
- Interface Humain-Machine (IHM), avec son microcontrôleur
- Réchauffage pour stabiliser la référence de tension

Le schéma de principe des cinq premiers blocs combinés est présenté à la **figure 1**. Le réchauffage sera discuté plus tard.
Le module IHM fournit l'interface utilisateur et commande l'étalonneur en réglant la tension de sortie du CN/A. Cette tension est également convertie en un courant de sortie par le générateur de courant ; l'IHM commande le relais K1 via le FET Q1, qui détermine si l'étalonneur délivre une tension ou un courant très précis aux bornes de sortie J6 et J7. La précision est largement déterminée par la référence de tension, qui est une exigence majeure de ce montage.

L'alimentation électrique

L'étalonneur est alimenté par un adaptateur secteur USB fournissant au moins 500 mA. Le 5 V fourni est réparti en trois branches distinctes. L'une

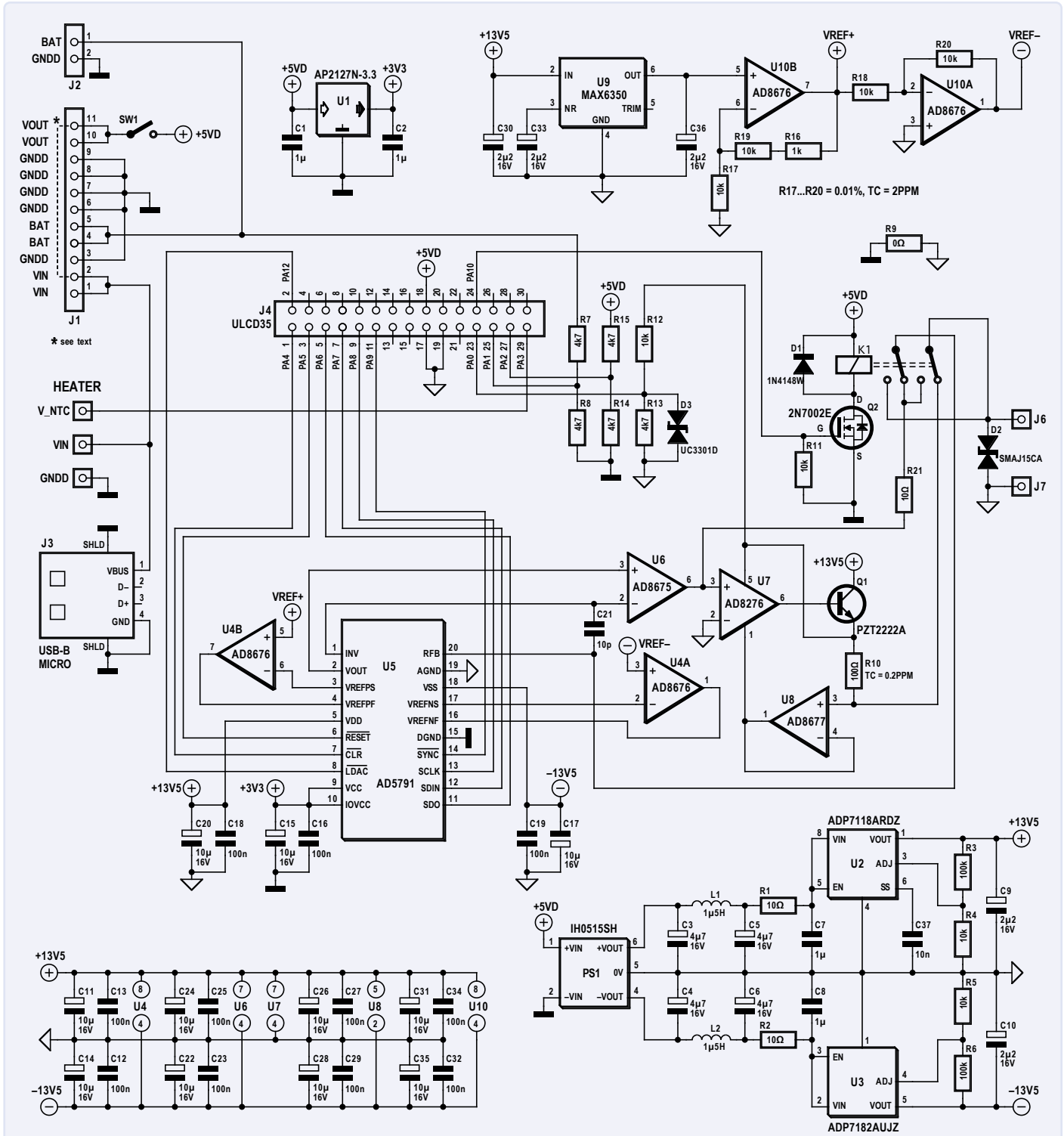


Figure 1. Schéma de l'étalonneur, circuit principal.

alimente directement l'IHM avec le microcontrôleur. La deuxième alimente U1, un régulateur à faible chute (LDO) de 3,3 V pour l'alimentation logique du CN/A (U5 et autour). La troisième branche alimente PS1 : un convertisseur CC/CC symétrique, isolé galvaniquement, avec des sorties de +15 V et –15 V. Ces sorties sont filtrées et alimentent deux régulateurs linéaires (U2 et U3). Les deux régulateurs sont de type à faible taux de chute et à faible bruit. Leurs sorties +13,5 V et –13,5 V sont utilisées pour les étages de sortie du circuit CN/A.

La référence de tension

C'est l'élément le plus important et en même temps le plus critique de ce projet. Si nous avons besoin d'une tension de référence de 10,00000 V avec une précision de ±10 µV, nous voudrions qu'elle se maintienne indéfiniment à cette valeur et à cette précision ; mais dans le monde réel, ce n'est qu'un vœu pieux. La conception de références de tension précises et stables est une science en soi. Sur le web, il existe des dizaines d'articles et de spécialistes qui discutent de ce sujet afin de repousser les limites de caractéristiques telles que : le bruit, la stabilité en ppm/°C à court et à long termes, l'hystérésis, la FEM thermoélectrique, piézoélectrique etc. [4][5]. Dans votre moteur de recherche préféré, cherchez la Rolls des références de tension : le LTZ1000 avec une dérive de seulement 0,05 ppm/°C [6]. Certains auteurs ont produit des références de tension avec de surprenants et magnifiques circuits imprimés, avec



Figure 2. Copie d'écran de l'EDI Workshop 4.

des résistances à très faible coefficient de température dans un boîtier métallique rempli d'huile.
Après avoir testé trois références de tension plus abordables (LTC6655, ADR445, MAX6350ESA), j'ai choisi le MAX6350ESA de Maxim, que j'ai trouvé le plus stable dans le temps avec une sortie de 5 V. Avec les deux ampli-op U10A&B à l'intérieur de l'AD8676 (double ampli-op à faible bruit et faible dérive de tension de décalage), on obtient des références de +10,48 V et −10,48 V (VREF+ et VREF- respectivement). Les résistances R16, R17, R18, R19 et R20 autour de l'AD8676 sont des résistances à faible dérive en température (meilleure que 2 ppm/°C).

Le CN/A autour de l'AD5791

Le CN/A que j'ai sélectionné est un AD5791, choisi pour sa résolution de 20 bits et sa linéarité de 1 ppm. Cette linéarité simplifie l'étalonnage : seuls les points 0 V et 10 V doivent être ajustés. Le schéma autour de l'AD5791 utilise des composants identiques à la carte d'évaluation d'Analog Devices pour ce CN/A [7]. Dans mon projet, le CN/A a des tensions de référence de +10,48 V et −10,48 V, ce qui produit des pas d'environ 20 µV. La tension de sortie peut être calculée en utilisant :

$$V_{out} = \frac{\left(+V_{ref} - \left(-V_{ref}\right)\right)}{2^{20} - 1} \times code - V_{ref}$$

Ou approximativement :

$$V_{out} = 20\mu V \times code - V_{ref}$$

La tension de sortie de l'étalonneur est limitée par logiciel entre −10 V et +10 V. Cela donne une grande aptitude à compenser l'erreur de décalage à 0 V.

La source de courant de précision

Les quatre principaux composants de la source de courant de précision sont U7 (amplificateur de mesure AD8276), Q1 (transistor NPN PZT1222), U8 (amplificateur de précision AD8677) et R10 (100 Ω). Q1 fournit le courant à la charge. Ce courant traverse R10 (résistance shunt), le relais puis se dirige vers la charge testée pour finalement retourner à la masse. La tension aux bornes de R10 mesurée par U8 et U7 est comparée à la tension de référence fournie par le CN/A AD5791. Le courant de la source de sortie est égal à $V_{CN/A}$ / R10, ou $V_{CN/A}$ / 100. Le courant est limité à 40 mA par le logiciel pour éviter la dérive de la mesure de tension dans R10, due à l'échauffement thermique à l'intérieur de R10 et Q1. Le coefficient de température de la résistance R10 est critique et doit être meilleur que 0,2 ppm/°C. Pour cette raison, Q1 est situé aussi loin que possible de R10 et bien sûr des tensions de référence sur le circuit imprimé.

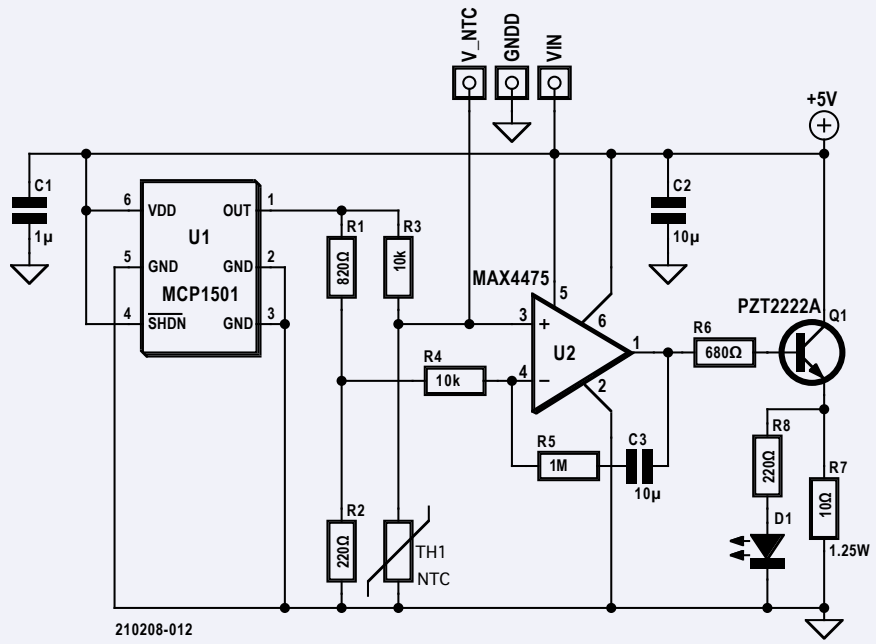


Figure 3. Schéma du réchauffeur.

L'IHM et son microcontrôleur

Cette partie est un module gen4-uLCD35-DT de 4D Systems, qui combine – entre autres – un écran tactile résistif couleur de 3,5 pouces, un microcontrôleur et un support de carte micro-SD. L'EDI 4D Workshop 4 [3] est gratuit et contient les bibliothèques de fonctions des boutons tactiles pour ce projet (voir **fig. 2**). Le code lui-même est une forme simplifiée du C. L'interface graphique utilisateur (GUI) est réalisée ainsi : les boutons tactiles et les chiffres sont des images stockées sur la carte micro-SD. Chaque image a des coordonnées fixes sur l'écran. Lorsqu'on touche l'écran, on provoque une interruption qui détermine quelle image a été touchée et déclenche les opérations associées à ce « bouton ».

Réchauffage de la référence de tension

La dérive de la tension de sortie en fonction de la température est mesurée en ppm/°C. Par exemple, avec un signal de sortie de 10 V, 1 ppm correspond à une dérive de tension de 10 µV. La température ambiante de mon laboratoire varie approximativement de 15 °C à 25 °C, donc si la sortie a un coefficient de température de 2,5 ppm/°C, la tension de sortie peut varier jusqu'à 250 µV dans cette plage de température. J'ai utilisé une chambre thermique bricolée avec un contrôleur de température pour déterminer ce coefficient.

Je me suis alors demandé si je pouvais améliorer le coefficient de température en réchauffant la référence de tension. J'ai choisi un point de consigne à 55 °C, bien plus élevé que la température ambiante normale. J'ai construit un réchauffeur commandé par la température sur un petit circuit imprimé et je l'ai collé sur le côté inférieur du circuit de référence de l'étalonneur. J'ai de nouveau mis l'étalonneur dans ma chambre thermique, fait un cycle de température de cinq heures de 15 °C à 40 °C, et obtenu une dérive de la tension de sortie de 26 µV, ce qui donne un coefficient de température de moins de 0,2 ppm/°C, ce qui s'est avéré être cinq fois mieux que sans réchauffage !

Une enceinte « four » imprimée en 3D a été conçue pour couvrir le circuit de réchauffage et stabiliser sa température et celle de la référence de tension. Les fichiers de conception 3D peuvent également être trouvés sur la page Elektor Labs [1].

La **figure 3** donne le schéma de la commande de réchauffage. La CTN (TH1) est placée sur le côté inférieur du circuit imprimé de la **figure 4** et doit être alignée avec et collée sous la référence de tension U9 sur le circuit imprimé principal. La **figure 6** montre le prototype du circuit imprimé du réchauffage.

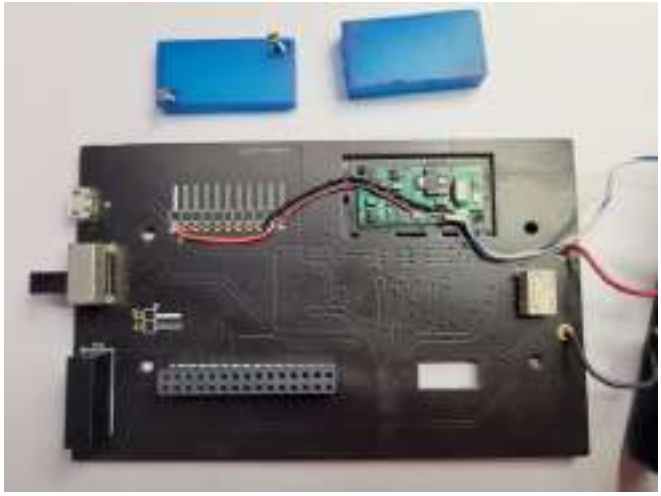


Figure 4. Le réchauffeur monté sur le circuit principal, sous la référence de tension.

Programmation du module uLCD35

Cet étalonneur ne fonctionne pas sans logiciel, tous les fichiers nécessaires sont disponibles en téléchargement sur la page Elektor Labs [1]. Pour le module IHM lui-même, achetez le kit de démarrage SK-35DT-AR de 4D Systems, qui contient l'écran tactile, une carte micro-SD de 4 Go et une interface pour programmer le module IHM. Vous trouverez ce kit chez Digikey (1613-1050-ND) ou RS-Components (841-7790).

En outre, vous devez télécharger et installer l'EDI de 4D Systems, disponible gratuitement à l'adresse [3].

Pour programmer le module IHM, assurez-vous d'abord que la carte micro-SD est formatée en FAT, et non en FAT32 ! Téléchargez *Software.ZIP* et *USD_FILES.ZIP* depuis [1] et extrayez leur contenu dans des dossiers de votre ordinateur. Copiez tous les fichiers du dossier USD_FILES sur la carte micro-SD et insérez-la dans le module IHM. Connectez l'interface de programmation sur l'embase à 10 broches du module et à un port USB de votre ordinateur. Ouvrez l'application *Program Loader* (**fig. 6**) à partir de l'EDI de 4D Systems, sélectionnez le port COM de l'interface de programmation, sélectionnez Flash comme destination, chargez le fichier *calibrator.4XE* et cliquez sur OK. Lorsque le programme est chargé, le module affiche tous les boutons et les roues.

Note : le code source se trouve dans le fichier **calibrator.4Dg** ; tous les commentaires sont en français. L'utilisateur peut l'éditer et le modifier avec l'EDI.

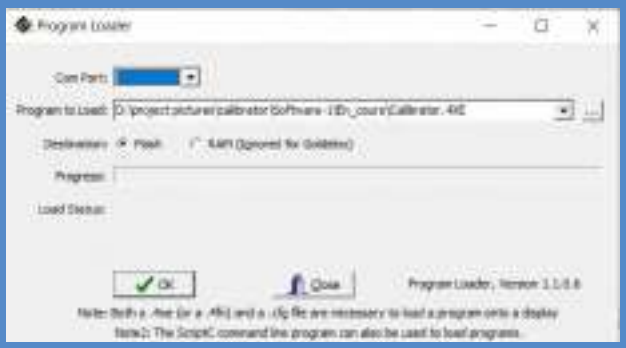


Figure 5. Workshop 4, le chargeur de programmes.

Construction de l'étalonneur

Les fichiers de conception KiCad, les fichiers gerber et de perçage pour commander le circuit imprimé principal et celui de réchauffage, et les listes de composants pour les deux cartes sont disponibles au téléchargement sur la page web Elektor Labs associée à ce projet [1]. Le logiciel de programmation du module IHM est également disponible au téléchargement sur cette page.

Le projet est conçu pour tenir dans le boîtier Hammond spécifié dans la liste de composants. Même si le soudage des composants CMS n'est pas facile pour les amateurs inexpérimentés, il ne sera pas trop difficile d'assembler les circuits imprimés à l'aide d'un petit fer à souder.

Option batterie

Le circuit imprimé de l'étalonneur a été conçu pour être utilisé avec un module (dé-)chargeur de batterie chinois (réf. DDO4CVSA) du fabricant Eletechsup et une batterie au lithium de 4,1 V, mais après quelques essais, je l'ai écarté de mon montage final car – à mon avis – il chauffait trop. N'hésitez pas à l'essayer, à vos risques et périls bien sûr. **Notez que V_{in} et V_{out} sur J1 doivent être reliés avec un fil de liaison si ce module n'est pas installé !**

Étalonnage

Comme pour toute source de tension de référence, le MAX6350 doit être vieilli pour améliorer sa stabilité à long terme. J'ai choisi 1000 h à température ambiante. Après 41 jours (environ 1000 h), la référence est stable à 4,99994 V ; les deux références de 10,50050 V et −10,49867 V, respectivement, sont stables à environ 10 µV.

Il faut ensuite un multimètre à 6,5 chiffres minimum pour l'étalonnage, et j'ai utilisé mon Agilent 34410A pour cela. Si vous maintenez le bouton *Sign de* l'IHM enfoncé pendant 8 s, tous les paramètres d'étalonnage précédemment enregistrés sont effacés.

Étalonnage de la tension de sortie

Tout d'abord, sélectionnez le mode de sortie « tension » en appuyant sur *Volt/mA*. Maintenez le bouton *000* enfoncé pendant 5 s pour lancer l'étalonnage, un bip retentit et le message *calibration mode* s'affiche. Utilisez les boutons « + » et « – » des roues codeuses pour obtenir une lecture de 0 V ±15 µV sur le multimètre, attendez environ 10 s pour stabiliser. En appuyant sur *000*, l'écran affiche *cal stored* pendant 2 s, puis les roues codeuses affichent *10.00000*. Puis, toujours à l'aide des roues codeuses, sélectionnez la valeur nécessaire pour obtenir 10,00000 V ±15 µV sur le multimètre.

Étalonnage du courant de sortie

Sélectionnez d'abord le mode de sortie « courant » en appuyant sur V*olt/mA*. Appuyez sur le bouton *000* et maintenez-le enfoncé pendant 5 s pour lancer l'étalonnage, un bip retentit et le message *calibration mode* s'affiche. Les roues codeuses affichent *00,0100 mA*. À l'aide des boutons « + » et « – » des roues codeuses, sélectionnez la valeur correcte pour afficher 10 µA ±100 nA sur le multimètre. En appuyant sur *000*, *cal stored* s'affiche sur le LCD pendant 2 s, puis les roues codeuses affichent *30,0000*. À l'aide des roues codeuses, sélectionnez la valeur nécessaire pour obtenir 30 mA ±10 nA.

Fonctionnement

L'écran de l'interface graphique se compose de roues codeuses tactiles et de trois boutons. Les boutons +/– des roues codeuses sélectionnent la valeur de sortie. Le bouton *sign* inverse la polarité de la tension de sortie, il est inactif en mode courant. Le bouton V*olt/mA* sélectionne le mode de sortie (tension ou courant), et le bouton *000* réinitialise la sortie à 0 V ou 0 mA, respectivement.



Figure 6. Le circuit du réchauffeur, CMS uniquement…

Un atout précieux

Le coût total de la construction de cet étalonneur de tension et de courant de haute précision sera d'environ 300 €, ce qui un très bon prix pour cet appareil. Certes, vous aurez également besoin d'un multimètre de haute précision et bien étalonné pour garantir un réglage précis ; cela peut poser un problème, mais la plupart d'entre nous auront d'une manière ou d'un autre accès à ce type d'appareil. Cet étalonneur sera certainement un atout précieux dans votre laboratoire pour tester et ajuster toutes sortes de circuits électroniques, d'appareils et d'instruments de mesure.

210208-04

Contributeurs

Idée, conception, texte et illustrations : **Vincent Gautier**
Schémas : **Patrick Wielders**
Rédaction : **Luc Lemmens**
Mise en page : **Harmen Heida**
Traduction : **Denis Lafourcade**

Des questions, des commentaires ?

Envoyez un courriel au rédacteur (luc.lemmens@elektor.com).

LIENS

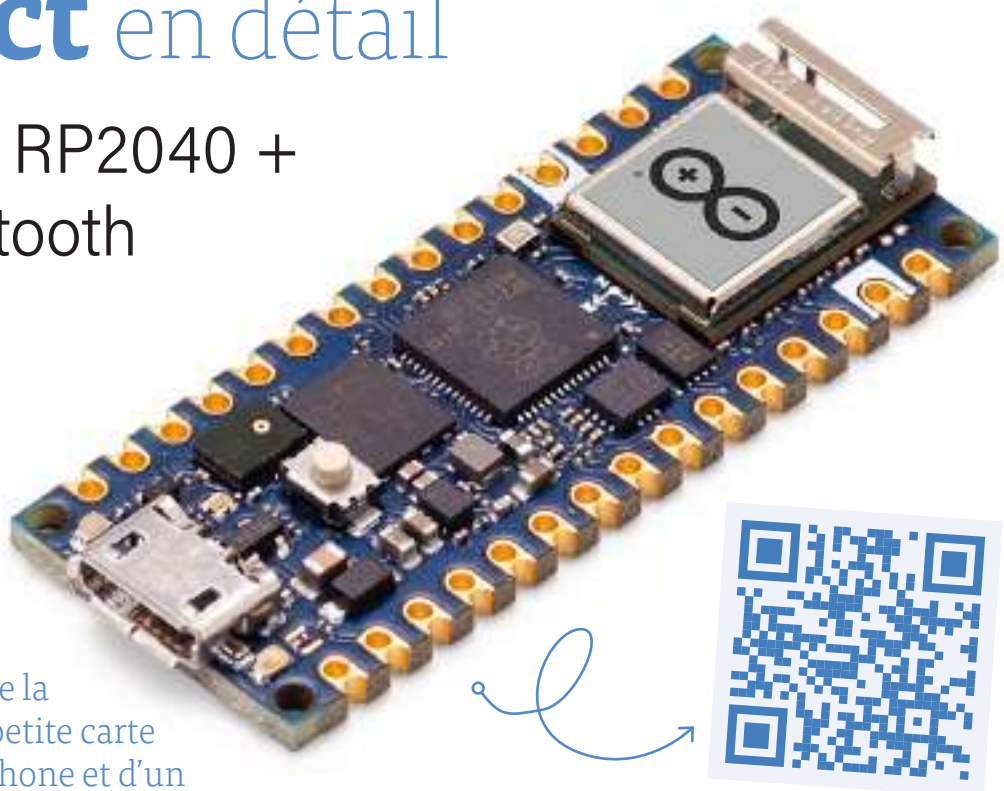
[1] Ce projet sur Elektor Labs : www.elektormagazine.fr/labs/voltage-current-calibrator-0-to-10v-and-0-to-40ma-0001
[2] Vidéo présentant ce projet : https://youtu.be/i8ZxOXWNTnU
[3] Téléchargement de l'EDI Workshop 4 : https://4dsystems.com.au/workshop4#downloads
[4] H. Halloin, P. Prat et J. Brossard, « Long Term Characterization of Voltage References », 2013 : https://arxiv.org/pdf/1312.5101.pdf
[5] L. T. Harrison, « Current Sources & Voltage References » : https://bit.ly/36eN4Pb.
[6] LTZ1000 : https://xdevs.com/article/kx-ref/
[7] Carte d'évaluation AD5791 : https://bit.ly/3hdrFMW

PRODUIT

› Multimètre numérique 3442, à valeur efficace vraie, avec Bluetooth (60000 points) de PeakTech
www.elektor.fr/18773

Arduino Nano RP2040 Connect en détail

Raspberry Pi RP2040 + Wi-Fi + Bluetooth



Clemens Valens (Elektor)

La carte Arduino Nano RP2040 Connect combine un microcontrôleur RP2040 avec un module Wi-Fi et Bluetooth. Outre la connectivité sans fil, la petite carte est équipée d'un microphone et d'un capteur de mouvement intelligent à six axes avec des capacités d'intelligence artificielle.

Ce banc d'essai en vidéo

En janvier 2021, la scène des fabricants a été prise par surprise lorsque Raspberry Pi a lancé sa carte Pico et son tout nouveau microcontrôleur RP2040. Quelques mois plus tard, Arduino a également annoncé une carte basée sur le RP2040 : l'Arduino Nano RP2040 Connect.

L'Arduino Nano RP2040 Connect reprend là où Raspberry Pi s'est arrêté en décidant de ne pas intégrer la connectivité sans fil sur la carte Pico. C'est compréhensible, car cela aurait fait grimper le prix, mais c'est ressenti comme une occasion manquée par beaucoup. Par conséquent, la « Connect » est équipée d'un module radio Wi-Fi et Bluetooth u-blox NINA-W102. Il s'agit d'un module du même fabricant que celui qui équipe l'Arduino Nano 33 BLE.

Outre la connectivité sans fil, la carte Connect dispose d'un microphone et d'un capteur de mouvement intelligent à six axes avec des capacités d'intelligence artificielle. Une LED RVB est également disponible. Les 22 ports GPIO, dont 20 avec support MLI et six entrées analogiques (huit si l'on compte le bus I2C), permettent à l'utilisateur de commander par exemple des relais, des moteurs et des LED et de lire des interrupteurs et autres capteurs.

La mémoire de programme est abondante avec 16 Mo de mémoire flash, plus que suffisante pour stocker de nombreuses pages web ou d'autres données.

L'Arduino Nano RP2040 Connect est « Raspberry Pi Compatible », ce qui signifie qu'elle prend en charge non seulement l'ensemble de l'écosystème logiciel RP2040, mais aussi MicroPython. En même temps, la carte supporte le langage de programmation Arduino, les EDI v1 et v2 et toutes les bibliothèques qui vont avec. Enfin, la Connect est entièrement compatible avec Arduino Cloud et l'application smartphone Arduino IoT Remote.

Wi-Fi et Bluetooth

La connectivité sans fil est assurée par un module NINA-W102 de u-blox (**fig. 1**). Selon sa fiche technique, c'est un module MCU multi-radio autonome qui intègre un microcontrôleur puissant et une radio pour la communication sans fil. Il possède ce qu'ils appellent chez u-blox une architecture CPU ouverte. Cela signifie que c'est à l'utilisateur d'écrire



Figure 1. Le module MCU multi-radio autonome u-blox NINA-W102 a un ESP32 d'Espressif sous le capot (source : u-blox).

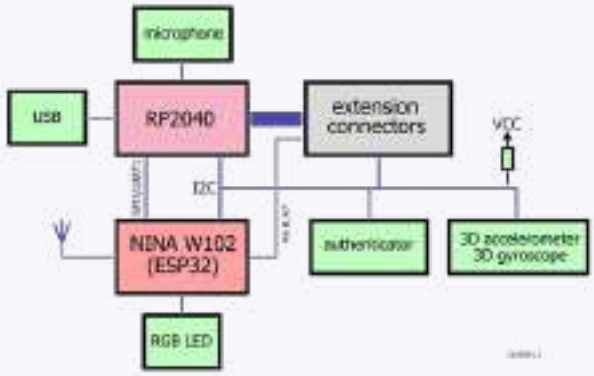


Figure 2. Synoptique de l'Arduino Nano RP2040 Connect.

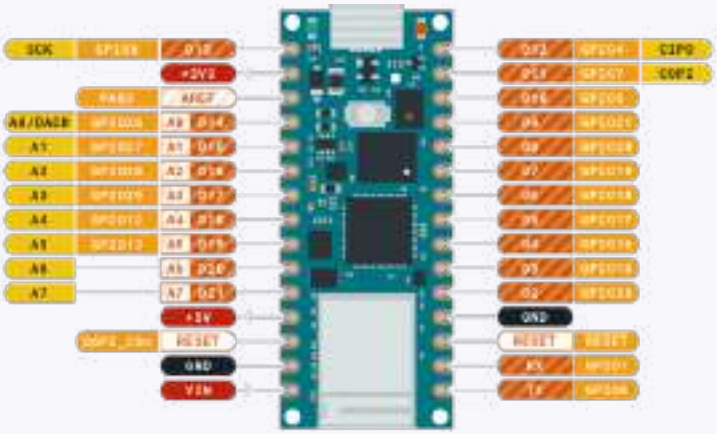


Figure 3. Légende des connecteurs d'extension de l'Arduino Nano RP2040 Connect (source : arduino.cc).

tous les logiciels pour lui. En l'état, il ne fait rien. En fait, le NINA-W102 est un module ESP32 dans une boîte de conserve avec une antenne. Le micrologiciel du module NINA a été créé par l'équipe Arduino et le code source peut être trouvé sur GitHub [1].

Maintenant que nous savons cela, nous réalisons soudainement que l'Arduino Nano RP2040 Connect est beaucoup plus puissant que nous le pensions initialement, car il ne dispose pas seulement d'un RP2040 avec deux cœurs Cortex-M0+, mais aussi d'un ESP32 à double cœur et de fonctions Wi-Fi et Bluetooth. Tout ce que l'on utilise normalement sur un module ESP32 fonctionne également sur le module NINA et le RP2040 peut donc être utilisé pour d'autres tâches.

Aperçu de la carte

En examinant le schéma fonctionnel de la carte Connect (**fig. 2**), nous constatons que les deux MCU partagent un bus I2C auquel sont connectés l'unité de mesure inertielle et le dispositif d'authentification, de sorte que les deux peuvent les utiliser. Ils sont également interconnectés au moyen d'un bus SPI/UART partagé.

La LED RVB de la carte est connectée à l'ESP32, euh, je veux dire au module NINA, tandis que le microphone est connecté au RP2040. Les connecteurs d'extension (**fig. 3**) sont également connectés à celui-ci, à l'exception des broches A6 et A7 qui sont connectées à l'ESP32. Notez que les broches analogiques A4 et A5 sont réservées au bus I2C, elles ont des résistances de tirage vers le haut de 4,7 kΩ, et elles sont câblées aux deux MCU. C'est également le cas pour la broche QSPI_CSn qui est étiquetée REC sur la carte, mais Reset sur le schéma de brochage.

La famille Arduino Nano

En fait, la Connect semble être une sorte de version améliorée de la carte Arduino Nano 33 IoT qui combine également un processeur ARM Cortex-M0+ avec un module sans fil u-blox NINA W102. Toutefois, son processeur, un SAMD21, est un dispositif à un seul cœur au lieu d'un double cœur et la carte ne dispose pas de la LED RVB et du microphone. Elle dispose également de moins de mémoire, et elle est plus lente.

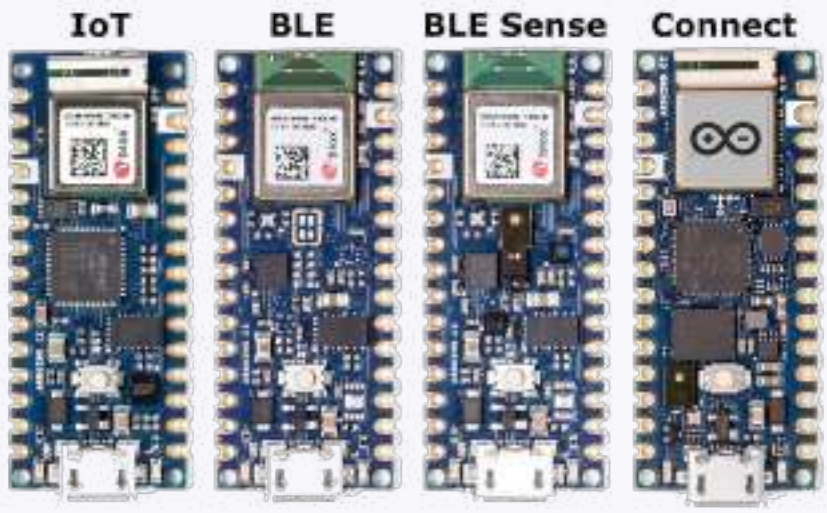


Figure 4. La famille Arduino Nano comprend actuellement quatre cartes. Elles ne mesurent que 45 × 18 mm (source : arduino.cc).

Du côté logiciel, en revanche, la Connect est regroupée avec l'Arduino Nano 33 BLE. Elle possède également un module u-blox NINA, mais celui-ci est basé sur un MCU nRF52840 de Nordic Semiconductor et il n'y a pas de second MCU. De même, elle ne dispose pas de Wi-Fi, mais uniquement de Bluetooth.
Les cartes Nano 33 « BLE », « BLE Sense » et « Connect » forment la famille Mbed OS Nano (**fig. 4**). Je ne comprends pas bien pourquoi le Nano 33 IoT ne fait pas partie de cette famille (pas assez de mémoire ?). En fait, je pense que l'Arduino Nano RP2040 Connect aurait dû s'appeler le Nano 33 Connect, mais le remplacement du « 33 » par « RP2040 » peut aider à stimuler les ventes.

Support logiciel

Étant une carte Arduino, la Connect s'intègre parfaitement dans l'EDI Arduino. Cependant, au moment de la rédaction de cet article, il ne semble pas y avoir de support pour l'utilisation des deux cœurs du RP2040 et il n'y a rien d'autre de spécifique au RP2040, comme le support de ses blocs PIO ou de l'interpolateur. Les exemples sont tous des exemples génériques Arduino qui fonctionnent sur de nombreuses cartes. Il y a quelques tutoriels qui expliquent comment utiliser le microphone ou l'IMU, mais rien qui ne fonctionne que sur un RP2040, ce que je trouve un peu décevant.
Je veux dire : « Pourquoi mettre un RP2040 sur la carte si vous ne fournissez aucun support pour lui ? » Sans cela, la carte serait simplement une Nano Connect avec des spécifications et des capacités similaires. Le RP2040 semble être principalement une extension d'E/S pour l'ESP32 afin de respecter le facteur de forme Nano. Il permet également à la carte d'être compatible avec le système d'exploitation Mbed, ce qui n'est pas le cas de l'ESP32, qui n'est pas un processeur ARM (seules les cartes basées sur ARM peuvent être compatibles avec le système d'exploitation Mbed). Mais ces objectifs auraient également pu être atteints en utilisant, disons, un MCU Cortex-M7 au lieu d'un RP2040. Et puis l'appeler quelque chose comme Arduino Portenta.

Les quatre grands

La carte Arduino Nano RP2040 Connect combine quatre grands noms de la scène des makers – Arduino, Raspberry Pi, ESP32 et Mbed OS – en un seul module, ce qui est assez intéressant. Elle témoigne d'une volonté de combiner le meilleur de l'*open source* au lieu d'essayer de tout garder propriétaire comme les grands fabricants de semi-conducteurs ont tendance à le faire.

Résumons

Donc, en résumé, la Connect est un petit module à microcontrôleur très puissant avec beaucoup de possibilités pour le prix d'une Arduino Uno (officielle). Si vous recherchez une carte pas trop chère avec Wi-Fi et/ou Bluetooth et capable d'exécuter Mbed OS pour une application IoT, alors la Connect est une option valable. Si, d'autre part, vous cherchez un moyen de jouer avec le RP2040 dans l'EDI Arduino, alors vous êtes probablement mieux loti avec la carte Raspberry Pi Pico, beaucoup moins chère mais légèrement plus grande, qui, d'ailleurs, peut également exécuter Mbed OS. Pour le double cœur et toutes les autres choses spécifiques au RP2040, installez le paquet Arduino IDE Boards de Earle Philhower III, il supporte aussi la Connect.

210400-04

Contributeurs

Texte et illustrations : **Clemens Valens**
Rédaction : **Jens Nickel, C. J. Abate**
Mise en page : **Giel Dols**
Traduction : **Maxime Valens**

Des questions, des commentaires ?

Envoyez un courriel à l'auteur (clemens.valens@elektor.com) ou contactez Elektor (redaction@elektor.fr).



PRODUITS

- **Arduino Nano RP2040 Connect**
  www.elektor.fr/19754
- **Carte Raspberry Pi Pico**
  www.elektor.fr/19562
- **Arduino Uno**
  www.elektor.fr/15877

LIENS

[1] Micrologiciel NINA : https://github.com/arduino/nina-fw
[2] « Pico/RP2040 Boards Package for Arduino » : https://github.com/earlephilhower/arduino-pico

le **corps physique** de l'intelligence artificielle

Tessel Renzenbrink (Pays-Bas)

Parce qu'ils sont d'abord perçus comme des abstractions, les termes *nuage* et *intelligence artificielle* entraînent la confusion dans bien des esprits. Le nuage est en réalité « l'ordinateur de quelqu'un d'autre », et l'IA dépend de ressources physiques comme les terres rares. De fait, ces technologies soulèvent bon nombre de questions d'ordre éthique.

Parcourant un forum en ligne, je suis tombée sur le message d'une mère en quête d'une solution pressante : sa fille utilisait Windows 365 depuis plusieurs jours pour rédiger une dissertation, mais aujourd'hui, à quelques heures seulement de l'échéance fixée par l'enseignant, le document affichait la version d'il y a trois jours, à laquelle manquait donc tout ce qui avait été écrit depuis. Des membres du forum suggérèrent plusieurs méthodes pour restaurer la version la plus récente du document, mais sans succès. La mère se tourna alors vers Microsoft, mais chacun des trois techniciens intervenus conclut que la restauration du fichier était impossible. Dans son dernier message, la mère écrivait que sa fille tentait maintenant désespérément de réécrire sa dissertation avant minuit.

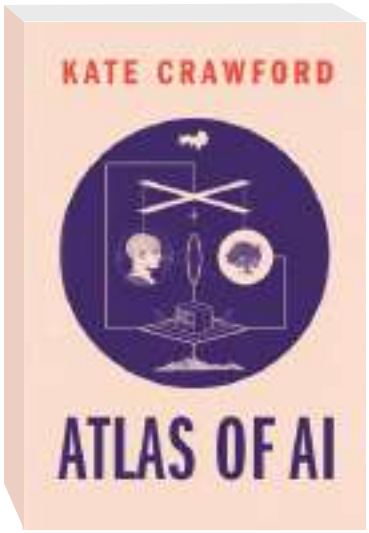
Le nuage est juste l'ordinateur de quelqu'un d'autre

Ce qui m'a le plus surprise dans cette histoire, c'est que des personnes stockent leurs fichiers importants uniquement dans le nuage, autrement dit dans l'ordinateur de quelqu'un d'autre. Cela revient à abandonner son autonomie et à prendre plusieurs risques, comme voir ses données analysées, devoir payer des frais d'accès supplémentaires, subir des modifications des conditions générales de vente ou, comme ici, une synchronisation, disons-le, foireuse. Sachant que même les ordinateurs portables bas de gamme offrent 128 Go de stockage alors qu'en moyenne la taille d'un fichier texte s'exprime en kilo-octets, rien ne semble justifier la sauvegarde de tels fichiers ailleurs que sur un disque dur local. Du moins du point de vue de l'utilisateur. Car le prestataire de service, lui, voit les choses tout autrement : remplacer la vente d'un seul produit par un « logiciel en tant que service » favorise les recettes, freine le départ des clients en rendant ce départ plus difficile, et engrange des téraoctets de données dans les serveurs – données dont l'analyse permet d'en apprendre davantage sur le monde, et qui peuvent ainsi être exploitées à profit.

« Les récits qui entourent l'IA renvoient l'image d'une activité abstraite.

Dès lors se pose la question : pourquoi les gens adhèrent-ils à ce stratagème ? Une partie de la réponse vient de ce que malgré trois décennies d'existence, nous n'avons toujours pas réellement compris le fonctionnement de l'internet, ni les rapports de force qui le sous-tendent. Et ce n'est pas un hasard. Le discours dominant sur les technologies numériques obscurcit la réalité physique et politique de l'internet, comme l'illustre l'emploi du terme nuage : de consonance sympathique et plutôt vague, il évoque un élément familier de notre écosystème, quelque chose d'à la fois distant et intangible.

L'expression « le nuage n'est que l'ordinateur de quelqu'un d'autre » a beau ne pas être nouvelle, la conscience collective continue à percevoir l'informatique distribuée comme un phénomène éphémère. Pendant ce temps, le déploiement de technologies numériques toujours plus nombreuses et puissantes se poursuit, et l'on continue à fabriquer des histoires autour d'elles pour les entourer de mystères.



Le livre 'Atlas of AI.' (Source : Yale University Press)

La forme spectrale de l'IA

Dans son ouvrage intitulé *Atlas of AI*, Kate Crawford s'emploie à montrer que le terme « intelligence artificielle » est lui aussi nimbé d'équivoque [1]. Selon elle, les récits qui entourent l'IA renvoient l'image d'une activité abstraite et immatérielle. Cette perception épurée conduit dès lors à l'idée fausse que l'IA n'a que peu de liens avec notre monde physique, qu'elle ne l'affecte pas, et que lui non plus n'est pas affecté par elle. Kate Crawford écrit : « L'IA a l'apparence d'une force spectrale – un calcul désincarné – mais ses systèmes sont tout sauf abstraits. Ce sont des infrastructures physiques qui à la fois remodèlent la Terre et modifient la façon dont le monde est vu et compris. »

Les petites mains de l'IA

Kate Crawford appuie ses dires en montrant que l'IA dépend de ressources concrètes telles que l'énergie, les terres rares, les interactions humaines stockées sous forme de données, ainsi que le travail humain non rémunéré et sous-payé. Lorsqu'un site nous impose un test CAPTCHA, nous créons gratuitement des données étiquetées servant à entraîner des systèmes d'IA. Les résultats nous reviennent sous la forme de récits fabuleux décrivant des systèmes artificiels capables d'apprendre eux-mêmes le fonctionnement du monde, sans oublier la promesse d'un futur proche dominé par les loisirs, une armée de machines se chargeant pour nous de toutes les tâches ennuyeuses. Pendant ce temps, des cohortes de petites mains sous-payées étiquettent des images sur des plateformes comme Mechanical Turk d'Amazon, tâche abrutissante s'il en est.

Consommation énergétique

Une autre composante physique de l'IA, rarement abordée, est sa gloutonnerie énergétique. Dans son livre, K. Crawford cite plusieurs articles de recherche dans lesquels les auteurs s'efforcent de chiffrer la quantité d'énergie nécessaire au fonctionnement du matériel. L'article de Strubell, Ganesh et McCullum, est à cet égard édifiant. Ces auteurs ont analysé les réseaux de neurones utilisés pour le traitement automatique du langage naturel, p. ex. la traduction et la création d'agents conversationnels [2]. L'entraînement de ces modèles repose sur des processeurs graphiques (GPU) et des unités de traitement de tenseur (TPU). Ce matériel spécialisé « doit tourner pendant des semaines ou des mois », écrivent les auteurs, qui citent l'exemple d'un modèle entraîné sans interruption sur une soixantaine de GPU durant six mois. Montant de la facture d'électricité : 9870 $ (avec un tarif de 0,12 $/kWh).

Kate Crawford s'appuie sur d'autres exemples pour montrer que l'IA est tout sauf une entreprise abstraite. En replaçant l'IA dans un cadre physique, elle nous fait comprendre que le terme intelligence artificielle recouvre une industrie complexe et multiforme. C'est de cette vision plus large dont nous avons besoin pour distinguer les structures du pouvoir sous-jacentes et comprendre comment elles affectent notre monde. K. Crawford écrit : « Nous avons besoin d'une théorie de l'IA qui tienne compte : des États et des entreprises qui la dirigent et la dominent ; de l'extraction minière qui laisse une empreinte sur la planète ; de la capture massive de données ; de l'exploitation extrêmement inégalitaire de la main-d'œuvre à son service. Ce sont là les forces souterraines qui gouvernent l'évolution de l'IA. »

210415-04 - VF : Hervé Moreau



Centrale à charbon illustrant l'impact qu'a l'IA sur l'environnement au travers de sa consommation électrique croissante. (Source : pxfuel.com/en/free-photo-ogrok/download)

LIENS

[1] K. Crawford, « Atlas of AI », Yale University Press, 2021 : https://yalebooks.yale.edu/book/9780300209570/atlas-ai
[2] E. Strubell, A. Ganesh, and A. McCallum, « Energy and Policy Considerations for Deep Learning in NLP », 57th Annual Meeting of the Association for Computational Linguistics (ACL), 2019 : http://www.arxiv.org/abs/1906.02243

projet 2.0

corrections, mises à jour et courriers des lecteurs

Ralf Schmiedel et **Jens Nickel** (Elektor)

station de soudage 2021

Elektor 05-06/2021, p. 30 (190409)





Dans ce projet, la tension de commande du HEXFET T3 à canal P (IRF9Z34NPBF) est autorisée à dépasser la tension maximale entre la grille et la source (V_{GS}). Selon sa fiche technique, ce FET peut supporter une tension maximale drain-source (V_{DS}) de 55 V, mais la tension maximale autorisée entre la grille et la source est de 20 V. Ce n'est pas un problème lorsque le circuit est alimenté par 17 V, mais avec une alimentation de 24 V, la valeur maximale de V_{GS} sera dépassée. Peu après la publication de cet article, nos lecteurs attentifs ont fait remarquer que la durée de vie de l'HEXFET serait écourtée sans autre mesure de protection. Pour éviter cela, une diode Zener de 15 V (BZX85B15) est montée entre la source et la grille, et la résistance R18 est échangée contre une 470 Ω/2 W. Les photos montrent les modifications nécessaires (diode Zener sur la face inférieure de la carte et résistance remplacée).
Ces modifications protègent le FET T3 en limitant la valeur maximale de V_{GS} à 15 V. La puissance supplémentaire dissipée dans la diode Zener et la résistance réchauffe les composants, il est donc conseillé d'installer un dissipateur thermique sur T3.

Un autre problème qui ne peut malheureusement pas être facilement résolu par l'ajout de quelques composants concerne la méthode de détection de la température utilisée par les pannes de fer à souder du HAKKO FX8801 et du JBC T-245. Le JBC T-245 ne fournit pas une tension positive pour indiquer la température de la panne comme prévu actuellement, mais une tension négative que la station ne peut pas gérer. Le HAKKO FX8801 utilise une résistance variable en fonction de la température (thermistance) au lieu d'un thermocouple pour détecter la température. Ces deux types de panne seront bientôt pris en compte par une carte supplémentaire en cours de développement. Pour cette carte, l'accent est mis sur l'utilisation de composants traversants (THT), qui sont également en cours d'achat.

Les fichiers Gerber, les schémas et les listes de composants mis à jour se trouvent sur la page web de l'article original : *www.elektormagazine.fr/190409-04*.

Mathias Claußen (Elektor)

ESP32 comme serveur de temps

Elektor 07-08/2019, p. 94 (180662)

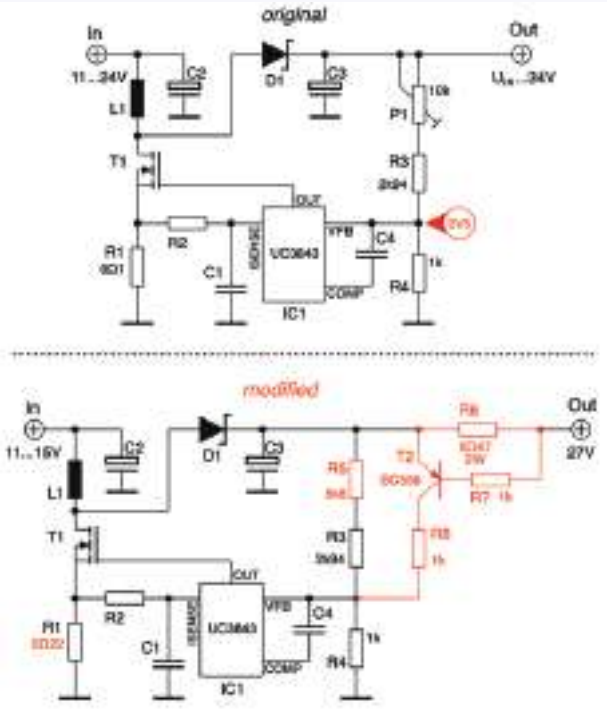
Un pointeur défectueux pouvait faire planter le serveur NTP dans la version 1.5 du micrologiciel. Grâce au conseil de « Schnarz » (https://www.elektormagazine.fr/labs/mini-ntp-server-with-gps) et au *dump* d'exception approprié, le problème a pu être résolu rapidement. La version 1.6 permet maintenant l'utilisation de la synchronisation améliorée. Avec la version 1.6, le projet est également converti en PlatformIO, de sorte que les bibliothèques peuvent maintenant être automatiquement téléchargées en conséquence. La version 1.6 est disponible sur GitHub (https://github.com/ElektorLabs/180662-mini-NTP-ESP32). Nous sommes heureux que nos lecteurs ne se contentent pas de reproduire nos projets, ils les améliorent également !

Mathias Claußen (Elektor)

énergie solaire pour les robots de tonte

Elektor 07-08/2021, p. 105 (200553)

Une erreur s'est glissée dans le schéma du convertisseur de tension illustré à la figure 9. Sur ce schéma, l'anode de D1 doit être connectée à la jonction de L1 et au drain de T1. Le condensateur C2 tamponne la tension d'entrée (voir schéma). Cela n'a aucune influence sur le texte, car il décrit uniquement le circuit supplémentaire dessiné en rouge qui fournit la sortie fixe de 27 V et la limite de courant.



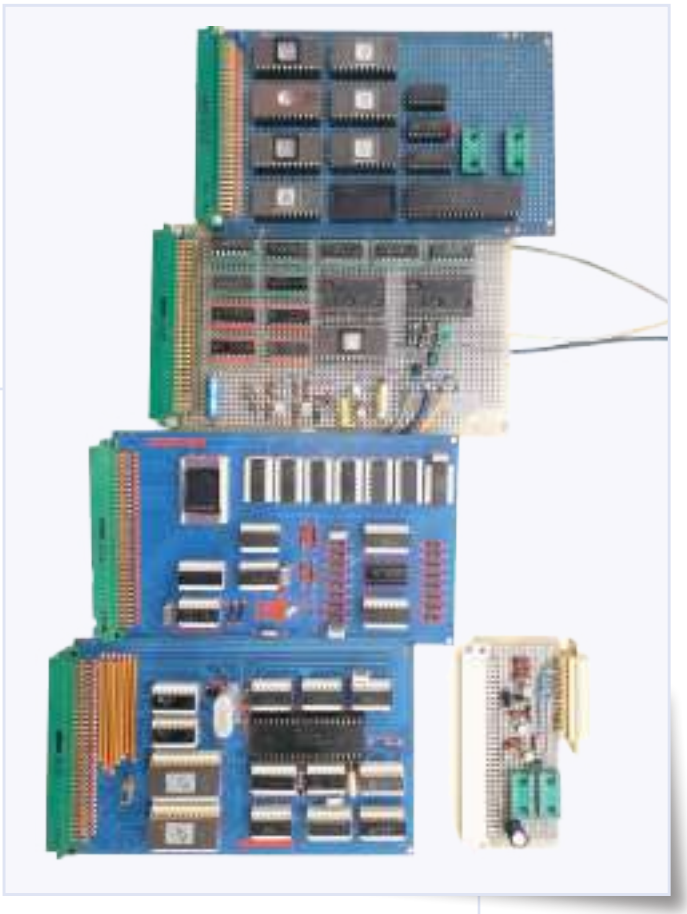
LoRa avec le Rasperry Pi Pico

Elektor 07-08/2021, p. 06 (210047)

Nous avons reçu quelques suggestions et demandes de renseignements sur la disponibilité de la carte pour ce projet. Nous n'avons pas encore eu l'occasion de commander ou d'installer la carte, elle n'a donc malheureusement pas encore été testée. Cependant, les demandes et les suggestions peuvent être partagées avec nous sur une nouvelle page Elektor Labs pour ce projet :
www.elektormagazine.fr/labs/raspberry-pi-pico-with-lora-and-recharable-lithium-battery

Mathias Claußen (Elektor)

Des questions, des commentaires ?
Contactez Elektor (redaction@elektor.fr).



Apprenons à utiliser le SC/MP

Elektor 05/1978, p. 50 (9846)

Pour moi, et sûrement pour beaucoup d'autres lecteurs aussi, l'article « Apprenons à utiliser le SC/MP » a été le plus marquant jamais publié dans Elektor. À l'époque, les microprocesseurs étaient tout nouveaux, tout le monde en parlait. Mais honnêtement, très peu d'entre nous savaient vraiment comment fonctionnaient ces nouveaux composants et ce que nous pouvions faire avec eux. Cet article a brusquement changé la donne et a suscité la curiosité et, pour beaucoup, la passion.
À cette époque, je venais de commencer mes études d'ingénieur en communication, mais je voulais vraiment savoir ce qu'étaient ces microprocesseurs. Avec deux autres étudiants, nous avons commencé à expérimenter avec le SC/MP. Notre enthousiasme et nos attentes ont atteint leur paroxysme lorsque nous avons mis la carte sous tension et que nous avons lu pour la première fois « ELBUG ... » sur l'écran. Nous avons rebaptisé notre domaine d'étude en « Technologie de l'information » – cette passion a façonné toute notre vie professionnelle. Récemment, en fouillant dans mon grenier, je suis tombé sur les cartes que nous avions construites il y a tant d'années. Par curiosité, je les ai allumées... Elles fonctionnent toujours !

Roland Stiglmayr

210434-04

1ère partie

création d'interfaces graphiques en Python **avec guizero**

Installez la bibliothèque Python guizero et créez vos propres interfaces graphiques.



MAKER

Laura Sach

Laura dirige l'équipe *A Level* de la Fondation RPi chargée des ressources pédagogiques en informatique à destination des étudiants.

@ CodeBoom



MAKER

Martin O'Hanlon

Martin crée des cours, des projets et des ressources en ligne au sein de l'équipe *Learning* de la Fondation RPi.

@ martinohanlon

Une interface utilisateur (GUI en anglais, pour *Graphical User Interface*, prononcez goo-i) facilite l'utilisation d'un programme en le rendant plus maniable. Une interface graphique est composée d'éléments graphiques (« *widgets* ») servant d'interfaces entre l'utilisateur et le programme. Vous les connaissez tous : liste déroulante d'un menu, bouton déclenchant une certaine action, etc. Nous utiliserons ici la bibliothèque *guizero*, écrite pour faciliter la création d'interfaces graphiques et donc idéale pour les débutants. La bibliothèque standard pour la création d'interfaces graphiques en Python est *tkinter*, et est installée par défaut avec Python sur la plupart des plateformes. La bibliothèque *guizero* est une adaptation des classes de *tkinter*, dont elle fournit un accès simplifié.

01 Installation de guizero

La première étape consiste bien sûr à installer la bibliothèque Python *guizero* (**lawsie.github.io/guizero**). L'installation proprement dite dépend de votre système d'exploitation et des droits dont vous disposez sur votre ordinateur (autrement dit si vous en êtes l'administrateur ou non).
Si vous avez accès à la ligne de commande (à un terminal), entrez la commande suivante :



Vous pouvez aussi installer guizero en téléchargeant son archive depuis GitHub.

```
pip3 install guizero
```

Si le terminal renvoie un message d'erreur, référez-vous aux instructions d'installation de **lawsie.github.io/guizero**, tous les systèmes d'exploitation et tous les cas sont couverts. Vous y trouverez notamment un installeur pour Windows.

02 Hello World

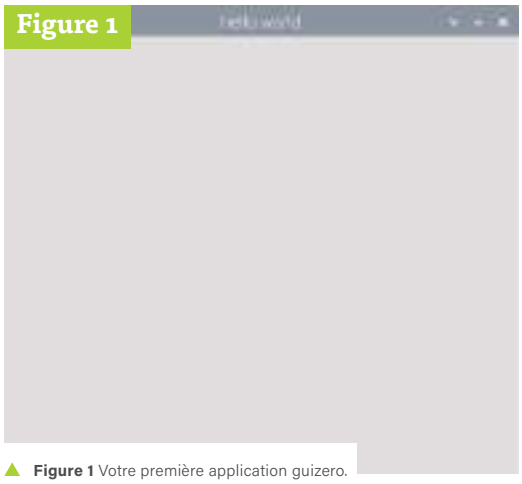
Assurons-nous de la bonne installation de *guizero* en écrivant une petite application appelée « *Hello World* », respectant en cela la tradition consistant à afficher le message Hello World lorsqu'on écrit son premier programme avec un nouveau langage.
Ouvrez votre éditeur de texte ou un EDI Python. Tout programme guizero commence par l'importation des widgets qui composeront l'interface graphique. Un widget ne s'importe qu'une seule fois et peut ensuite être utilisé autant de fois que souhaité dans le programme. Pour importer le widget `App`, ajoutez à la première ligne de votre fichier :

```
from guizero import App
```

Le widget `App` est le conteneur graphique de votre programme, autrement dit sa fenêtre principale. Un programme guizero se termine toujours par l'instruction lançant l'affichage du widget `App`. Ajoutez les deux lignes suivantes sous la première ligne d'importation :

```
app = App(title="Hello world")
app.display()
```

Enregistrez et exécutez le code : une fenêtre appelée *Hello World* devrait s'ouvrir (**fig. 1**). Bravo, vous venez de créer votre première application guizero !



▲ **Figure 1** Votre première application guizero.

01-helloworld.py

> Langage : **Python 3**

TÉLÉCHARGEZ LE CODE COMPLET :
magpi.cc/guizerocode

```
001.  <from guizero import App, Text
002.  app = App(title="Hello world")
003.  message = Text(app, text="Welcome to the app")
004.  app.display()
```



◀ Figure 2 Ajout d'un widget Text.

03 Ajout de widgets

Un widget est un élément de l'interface tel qu'un bouton ou une barre de défilement, mais peut aussi être du texte brut.
Tous les widgets se placent entre la ligne créant la fenêtre `App` et l'instruction `app.display()`. Voici comment ajouter un widget Text à notre fenêtre précédente :

```
from guizero import App, Text
app = App(title="Hello world")
message = Text(app, text="Welcome to the app")
app.display()
```

La première ligne demande l'importation de la classe de widget `Text` en plus du widget `App`, tandis que la troisième ajoute le widget `Text` à l'interface. La **figure 2** montre notre nouvelle fenêtre.

Détaillons la syntaxe de cette troisième ligne ::

```
message = Text(app, text="Welcome to the app")
```

Un widget est une variable, et donc doit avoir un nom. Nous avons choisi `message` ici. Nous définissons ensuite `message` en tant que widget de type `Text`, et indiquons entre parenthèses ses propriétés. Le premier paramètre spécifie le conteneur dans lequel doit être placé `message`, soit, ici, `app`. Tout widget doit être contenu dans un widget conteneur. Il s'agira la plupart du temps de la fenêtre principale (le conteneur `App`), mais vous découvrirez plus tard qu'un widget peut être placé dans d'autres types de conteneurs. Le second paramètre demande au widget de contenir le texte *Welcome to the app*.

Affichette *Wanted*

01 Embellissement

Notre fenêtre est pour le moins dépouillée. Une façon de l'égayer serait de lui ajouter un fond coloré, des images et du texte utilisant différentes polices de tailles et couleurs variées. Voyons comment procéder en reproduisant l'affichette « Wanted » de la **figure 9**.
Commençons par créer une fenêtre appelée *Wanted* avec le code suivant :

```
from guizero import App

app = App("Wanted!")

app.display()
```

Enregistrez et lancez le code. La **figure 3** montre ce que vous devriez obtenir, à savoir une fenêtre grise nommée *Wanted*.

02 Couleur d'arrière-plan

Pour rendre notre affichette plus réaliste, nous allons substituer au fond gris un jaune pâle, la couleur parcheminée traditionnelle des avis de recherche placardés dans les westerns.
La fenêtre (le widget) `app` possède une propriété `bg` qui définit sa couleur de fond et celle de ses widgets. `bg` (abréviation de *background*, arrière-plan) a pour valeur par défaut *None*, d'où le gris. Nous lui affectons la valeur `yellow` juste après la création de l'objet `App`, comme ceci :

Ingrédients

- Ordinateur (RPi ou PC sous Linux, Windows ou macOS)
- Connexion internet
- Python 3 (**python.org**)
- Éditeur de texte ou EDI, p. ex. IDLE (installé avec Python 3), Thonny (**thonny.org**), Mu (**codewith.mu**), PyCharm (**jetbrains.com/pycharms**).
- Bibliothèque Python guizero (**lawsie.github.io/guizero**)



Figure 3
La fenêtre de base.

```
from guizero import App

app = App("Wanted!")
app.bg = "yellow"

app.display()
```

Pour modifier une propriété, nous spécifions d'abord le widget auquel nous nous référons (`app`), le faisons suivre de la propriété à modifier (`bg`), puis affectons la valeur souhaitée (`"yellow"`).
Même si notre affichette concerne un chat, vous trouvez peut-être ce jaune un peu trop canari (**fig. 4**). La couleur pouvant être exprimée en valeur RVB ou hexadécimale, nous pouvons facilement l'adoucir en nous rendant sur l'un des nombreux sites de sélection de couleurs, p. ex. **htmlcolorcodes.com** (**fig. 5**).
Au jaune pâle que nous avons sélectionné correspondent les valeurs RGB (251, 251, 208) et hexadécimale #FBFBD0. Nous pouvons utiliser l'une ou l'autre de ces représentations dans le code, *guizero* les interprétera correctement. Autrement dit nous avons le choix entre les deux écritures suivantes :

```
app.bg = "#FBFBD0"
app.bg = (251, 251, 208)
```

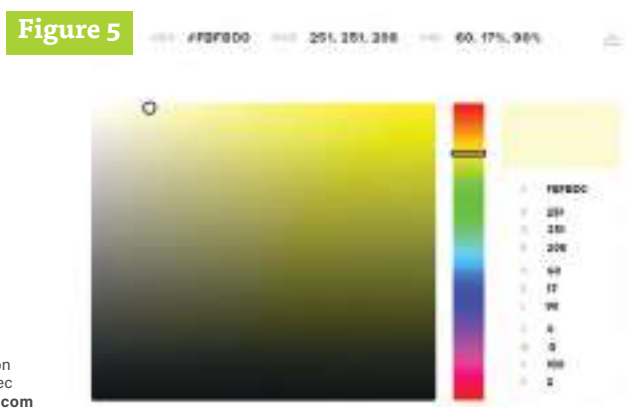


Figure 5 Sélection d'une couleur avec **htmlcolorcodes.com**

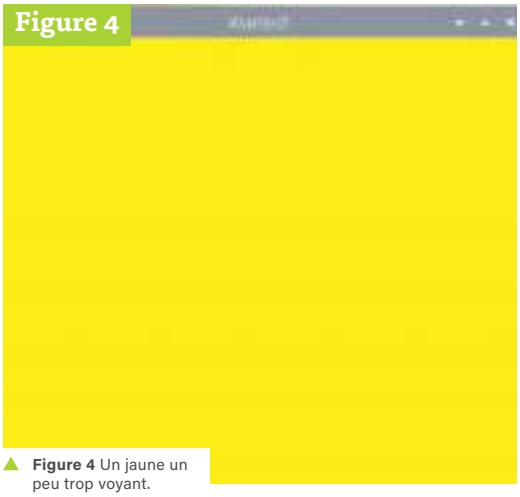


Figure 4 Un jaune un peu trop voyant.

03 Ajout de texte

Votre fenêtre devrait ressembler à celle de la **figure 6**. Ajoutons-lui le classique *Wanted* des avis de recherche, celui que même le plus analphabète des chasseurs de prime a toujours su traduire mentalement en dollars.
Nous voulons ajouter du texte, donc vous savez que c'est sur la première ligne d'importation, à savoir :

```
from guizero import App
```

que nous devons ajouter la classe **Text** :

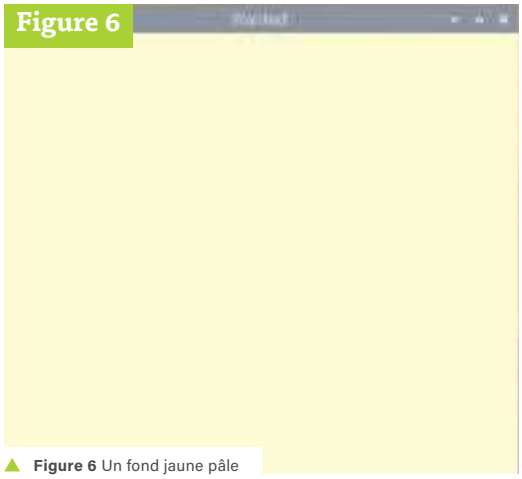
```
from guizero import App, Text
```

Vous l'avez compris, il est inutile d'encombrer le code avec une ligne d'importation par type de widget utilisé,

« La fenêtre **App** possède une propriété bg qui définit sa couleur de fond. »

Lisez la doc

Peut-être vous demandez-vous comment connaître les méthodes (fonctions), propriétés et paramètres de tel ou tel widget. Plus qu'une évidence, la réponse devrait être un réflexe : consulter la documentation. Qu'il s'agisse de *guizero* ou d'une autre bibliothèque, c'est elle qui sera votre meilleure guide. Celle de guizero se trouve sur **lawsie.github.io/guizero**. Le menu déroulant Widgets liste tous les widgets de la bibliothèque. Si par exemple vous sélectionnez le widget `Text`, vous découvrirez qu'il offre bien plus de paramètres que ceux que nous avons exploités ici. Dans la plupart des cas, la documentation illustre également l'utilisation d'une propriété ou d'une méthode particulière avec un extrait de code. Ne craignez pas de la parcourir - on n'est jamais à l'abri d'apprendre quelque chose d'utile !



▲ **Figure 6** Un fond jaune pâle



◀ **Figure 7** Le texte est trop petit.

nous les regroupons tous sur la même ligne en les séparant par une virgule.
Ajoutons maintenant un widget `Text`. Vous vous en souvenez, tous les widgets de l'interface doivent être ajoutés entre la ligne de création de la fenêtre `App` et la méthode `display()` l'instanciant. Ce qui donne

```
from guizero import App, Text

app = App("Wanted!")
app.bg = "#FBFBD0"

wanted_text = Text(app, "WANTED")

app.display()
```

Sa syntaxe ne vous est plus inconnue, mais détaillons à nouveau la ligne :

```
wanted_text = Text(app, "WANTED")
```

`wanted_text` est le nom de la variable identifiant le widget `Text`. Nous aurions tout aussi bien pu l'appeler *Grosminet*, l'ordinateur s'en bat les puces. Le point ici est qu'affecter un nom à un objet permet de s'y référer dans le code.
Ici encore nous retrouvons les deux paramètres utilisés précédemment. Le second est le texte que nous souhaitons voir afficher à l'emplacement du widget `Text`. Vous souvenez-vous du premier ? Oui, il s'agit du conteneur qui commandera le widget `wanted_text`, et ici aussi il s'agit de la fenêtre principale `app`. Nous ne l'avions pas précisé en début de tutoriel, mais ce conteneur est appelé le conteneur maître (*master*). Nous en verrons d'autres ultérieurement..

04 Taille et couleur du texte

Problème, le texte est trop petit (fig. 7) pour aider la mère Michel (qui a perdu son chat). Pour l'agrandir, nous donnons la valeur `50` à la propriété définissant la taille du texte. Rappelez-vous, nous devons spécifier trois choses :

1. Le nom du widget
2. La propriété à modifier
3. La valeur à affecter

Dans notre cas, le nom du widget est `wanted_text`, la propriété s'appelle `text_size`, et la valeur souhaitée est `50`. Nous complétons donc le code définissant notre widget avec la deuxième des lignes ci-dessous :

```
wanted_text = Text(app, "WANTED")
wanted_text.text_size = 50
```

Le texte se distingue maintenant beaucoup mieux (**fig. 8**). À titre d'entraînement, essayez d'en changer la police. Tous les systèmes d'exploitation n'offrent pas le même jeu de polices, mais voici quelques suggestions :

- **Times New Roman**
- **Verdana**
- **Courier**
- **Impact**

Un avis de recherche ne serait pas ce qu'il est sans une photo, donc ajoutons une image à notre interface. J'ai utilisé une photo de mon chat, car il aime à disparaître de ma vue pour aller gratter des trucs qui n'aiment pas être grattés.
Enregistrez votre image dans le dossier contenant votre programme. Vous pouvez utiliser les images d'un autre dossier, mais dans ce cas il vous faudra spécifier leur chemin d'accès. Un débutant trouve donc souvent plus simple de mettre les images dans le dossier du programme.



▶ **Figure 8** Un texte plus lisible.



Creation d'interfaces graphiques avec Python

Pour aller plus loin dans la création d'interfaces graphiques avec guizero, lisez notre livre (en anglais) *Create Graphical User Interfaces with Python*. Ses 156 pages abordent aussi la programmation évènementielle et comprennent dix projets stimulants. **magpi.cc/pythongui**



Figure 9 Aidons la mère Michel.

À ce stade l'ajout de widgets n'a normalement plus de secret pour vous. Rappelons tout de même qu'ils doivent toujours être importés en début de code et être définis entre la ligne créant la fenêtre principale et la ligne finale `app.display()`. Pensez aussi à leur affecter un nom parlant.

Le widget pour les images s'appelle Picture. Pour l'importer, nous écrivons donc :

```
from guizero import App, Text, Picture
```

Nous allons créer notre widget avec deux paramètres. Le premier spécifie son conteneur (`app`), le second le nom du fichier image. Dans mon cas (fichier image **tabitha.jpg**), l'instruction s'écrit :.

```
cat = Picture(app, image="tabitha.png")
```

Lancez votre programme (qui devrait ressembler à **02-wanted.py**) pour voir votre image s'afficher sous le texte (**fig. 9**).

À vous maintenant de compléter ou modifier ce code pour découvrir les nombreuses possibilités qu'offre *guizero* !

210419-04 (VF : Hervé Moreau)

Note de l'éditeur : cet article est paru initialement dans le magazine MagPi (le magazine officiel du Raspberry Pi). La maison d'édition Elektor publie ce magazine en néerlandais, allemand et français (www.magpi.fr).

Formats d'images

Notre objectif en écrivant la bibliothèque *guizero* était de la rendre facile à installer et à utiliser. Elle n'offre donc pas de fonctions complexes de manipulation d'image, ce qui aurait nécessité l'ajout de la bibliothèque *pillow*. Vous pouvez toutefois utiliser les formats GIF non-animés et PNG sur toutes les plateformes, à l'exception de PNG sur macOS. Tenez-vous-en à ces deux formats si vous n'êtes pas sûr d'avoir *pillow* sur votre système.

02-wanted.py

TÉLÉCHARGEZ LE CODE COMPLET :
magpi.cc/guizerocode

> Langage : **Python 3**

```
001. from guizero import App, Text, Picture
002.
003. app = App("Wanted!")
004. app.bg = "#FBFBD0"
005.
006. wanted_text = Text(app, "WANTED")
007. wanted_text.text_size = 50
008. wanted_text.font = "Times New Roman"

009. cat = Picture(app, image="tabitha.png")
010.
011. app.display()
```


ABONNEZ-VOUS ET RECEVEZ

Raspberry Pi précâblé GRATUIT

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





SEULEMENT 54,95 € PAR AN (6 NUMÉROS)

Vos avantages :

- Carte Raspberry Pi Zero avec broches GPIO soudées offerte
- Prix au numéro réduit
- Chaque numéro directement dans votre boîte aux lettres
- Tous les numéros disponibles sous forme numérique (PDF)
- Découverte de chaque nouveau numéro avant sa sortie en kiosque

ABONNEZ-VOUS : WWW.MAGPI.FR

Kit de mesure du CO_2 pour salle de classe

montage à base d'ESP8266, conçu par l'Université des Sciences Appliquées d'Aix-la-Chapelle

Thomas Dey, Ingo Elsen, Alexander Ferrein, Tobias Frauenrath, Michael Reke et **Stefan Schiffer** (Université des Sciences Appliquées d'Aix-la-Chapelle, Allemagne)

De nombreux enfants ne peuvent pas être vaccinés, ils seront donc le prochain groupe le plus vulnérable de la pandémie. Par conséquent, une ventilation régulière des salles de classe est cruciale, et la concentration de CO2 est un bon indicateur pour cela. Les enseignants de l'Université des Sciences Appliquées d'Aix-la-Chapelle, ville d'accueil d'Elektor, ont donc lancé un projet de compteur de CO2. Les étudiants peuvent construire l'appareil avec des modules électroniques prêts à l'emploi.



Figure 1. Dispositif final avec feu tricolore et écran dans un boîtier en carton.

L'Agence fédérale allemande pour l'environnement (BMU) indique sur son site web qu'une ventilation régulière permet de réduire l'exposition aux aérosols [1]. En outre, il est également recommandé d'utiliser un indicateur de CO_2 à feux tricolores pour avertir lorsque l'air d'une pièce est « épuisé » et qu'une ventilation est conseillée. C'est pourquoi de nombreuses écoles sont en train d'acheter des appareils de mesure. Mais le problème est que de nombreux appareils du commerce ne sont plus disponibles ou ne le seront que plus tard en raison de l'augmentation de la demande. Dans les communautés de makers sur l'internet, des instructions de fabrication circulent et montrent comment réaliser un feu tricolore pour CO_2 avec des composants électroniques disponibles sur le marché.

Indépendamment de cela, un groupe de projet de l'Université des Sciences Appliquées d'Aix-la-Chapelle s'est formé pour développer un tel kit. Dans ce cadre, différents capteurs dans différentes catégories de prix ont été étudiés et évalués.

Grâce à ces résultats, une équipe de recherche du département de génie électrique et de technologie de l'information a mis au point un appareil de mesure du CO_2 de grande qualité, peu coûteux et recourant à du matériel en stock [2]. Cet appareil de mesure est destiné à être utilisé dans les salles de classe, mais il convient bien sûr aussi à d'autres locaux fermés. Le compteur de CO_2 se présente sous la forme d'un kit de construction pratique. Un tutoriel en ligne fournit les instructions pour les enseignants, afin que l'appareil de mesure

FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

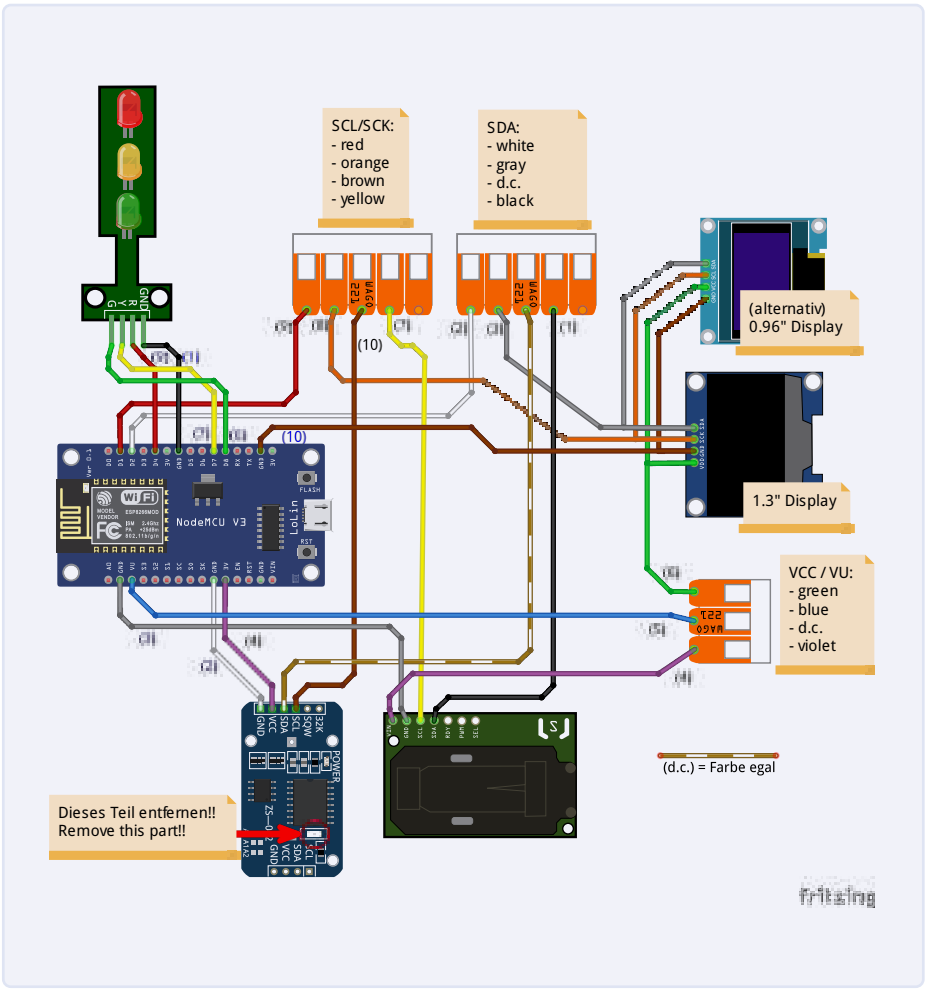


Figure 2. Schéma Fritzing des modules, avec l'ESP8266 comme cerveau du projet.

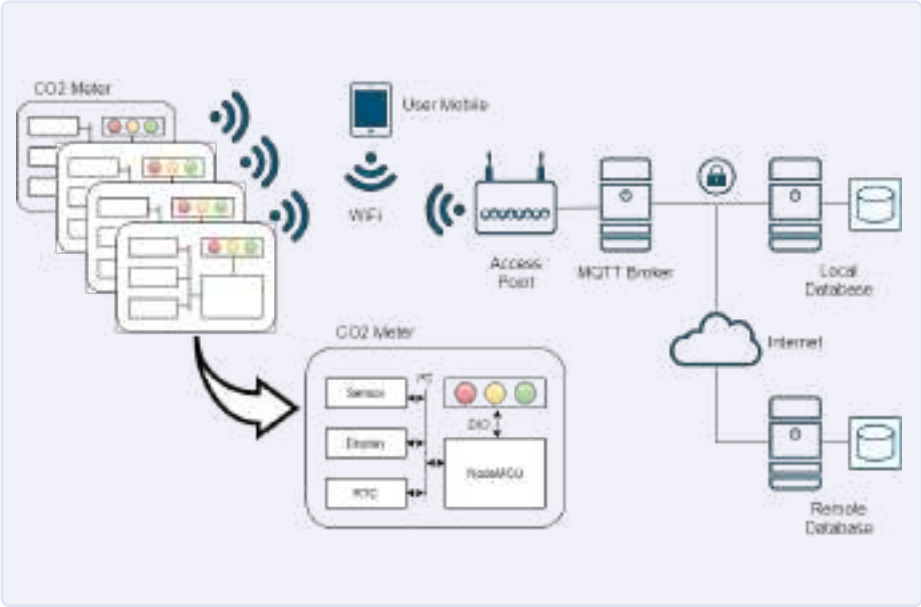


Figure 3. Nos compteurs de CO_2 peuvent envoyer leurs données au réseau via le protocole MQTT [8].

puisse être assemblé et abordé avec les élèves. Le tout est mis à la disposition des écoles au prix coûtant d'environ 65 €.

Le projet est développé et supervisé dans le nouveau « Makerspace » du département d'ingénierie électrique et de technologie de l'information [3]. Cet espace est destiné à permettre aux scientifiques et aux étudiants de mettre en œuvre leurs propres idées. On y trouve des postes de travail, du matériel et des appareils comme une imprimante 3D.

Le kit

Les principaux composants du kit de mesure du CO_2 sont les suivants :

- Capteur de CO_2 Sensirion SCD30
- Carte à microcontrôleur NodeMCU ESP8266-12F
- Écran OLED de 3,3 cm, 128×64 pixels avec contrôleur SH1106
- Module de feu tricolore à LED
- Horloge en temps réel DS3231

Le kit contient également des borniers Wago et des fils de liaison, ainsi qu'une alimentation murale USB et un câble micro-USB. On peut réaliser un boîtier simple avec du carton prédécoupé également fourni dans le kit.

La **figure 1** montre le dispositif assemblé avec le feu tricolore à LED. Nous avons décidé de proposer un kit qui ne nécessite aucune soudure afin de maintenir le niveau requis aussi bas que possible. Ainsi, aucun outil autre, à part un cutter, n'est nécessaire pour monter le kit. Les instructions sont disponibles sur [2] et [4].

Le cerveau du projet

La **figure 2** montre le câblage des modules mentionnés. Le populaire ESP8266 est le cerveau du projet ; ses capacités Wi-Fi ont été évoquées à plusieurs reprises dans les précédents numéros d'*Elektor*. L'interface sans fil peut être utilisée pour envoyer les données de concentration de CO_2 à des plateformes ouvertes du nuage sur l'internet via le protocole MQTT, ce qui les rend accessibles dans le monde entier. Cela permet également de gérer un groupe de capteurs de CO_2, par exemple dans des écoles (**fig. 3**).

Par ailleurs, le microcontrôleur ESP8266 exécute un serveur web et propose des pages



Figure 4. Une des pages web de configuration. Les utilisateurs peuvent saisir les seuils pour le feu tricolore. Une version anglaise des pages web est prévue.

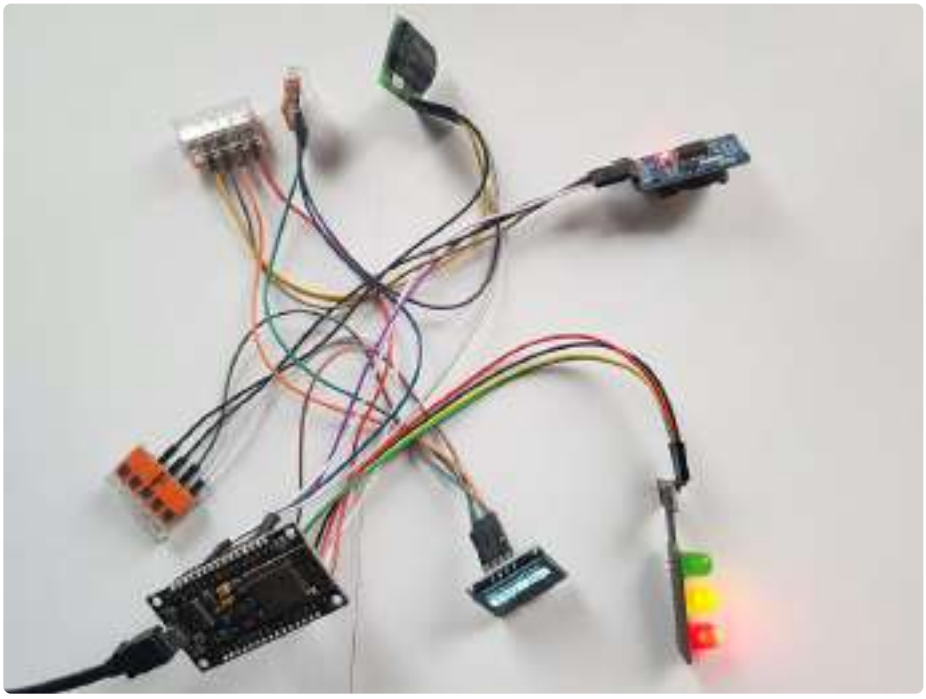


Figure 5. On utilise des borniers pour distribuer les signaux SDA et SCL de l'I2C et l'alimentation en 5 V des modules.

web à afficher dans le navigateur de son choix. L'utilisateur peut configurer l'appareil grâce à ces pages web. De nombreux lecteurs d'*Elektor* savent que l'ESP8266 peut établir son propre réseau Wi-Fi ou se connecter à des réseaux existants. Ce dispositif peut fonctionner dans les deux modes, à configurer par l'utilisateur via un navigateur web. En outre, les valeurs de seuil de CO_2 pour les trois LED du feu tricolore peuvent être configurées, en plus de nombreux autres paramètres (**fig. 4**).

Pour que la conception reste modulaire et simple, nous utilisons une carte NodeMCU comme carte d'extension pour le contrôleur ESP8266. Son connecteur USB facilite la (re) programmation. La carte NodeMCU peut être alimentée par USB, elle fournit les 3,3 V pour l'ESP8266. Les tensions de 5 V et 3,3 V sont également accessibles sur les connecteurs latéraux de la carte pour alimenter les autres modules.

Modules périphériques

L'ESP8266 communique par I2C avec le capteur de CO_2 (plus de détails sur le capteur dans le paragraphe suivant). Sur le même bus I2C, nous avons également connecté une horloge en temps réel, alimentée par une pile bouton CR2032, pour horodater les données de notre capteur. De plus, pour simplifier le câblage, nous avons décidé d'utiliser un écran avec une interface I2C. Comme le montre le schéma Fritzing du circuit sur la figure 2, nous utilisons des borniers pour distribuer les signaux SDA et SCL du bus I2C ainsi que la tension 5 V qui alimente les modules depuis la carte NodeMCU (voir **fig. 5**). Une exception pour la RTC qui doit être alimentée en 3,3 V. Les modules RTC que nous avons choisis sont livrés avec des batteries non rechargeables. Une simple diode 1N4148 en série avec 200 Ω fournit un courant de charge continu (et nécessite 3,3 V), ce qui ne vaut que pour les batteries rechargeables. Les batteries non rechargeables peuvent exploser ! Comme la durée de vie de la batterie dépasse la durée de vie prévue des capteurs, nous avons décidé de supprimer la diode sur la RTC pour désactiver le dangereux circuit de charge.

Comme nous l'avons dit, on utilise trois LED en forme de feu tricolore pour obtenir une indication rapide sur le niveau de CO_2. Ces LED sont connectées à trois broches GPIO de l'ESP8266 configurées en sortie. Nous avons choisi un module de feu tricolore intégrant les résistances nécessaires aux LED.

Capteur de CO_2

Le composant le plus important est le capteur lui-même : pour mesurer la qualité de l'air, de nombreux capteurs sont disponibles sur le marché. Les capteurs disponibles suivent tous un des deux principes de mesure : (a) les COV et (b) l'IRND.

Les COV (Composés Organiques Volatiles) indiquent la qualité de l'air, car des gaz tels que le monoxyde de carbone, le dioxyde de carbone ou le benzène modifient la conductivité du capteur. L'autre principe est celui de l'infrarouge non dispersif (IRND). En gros, la concentration de dioxyde de carbone dans un tube de mesure est estimée par la quantité de lumière infrarouge qui est réfléchie par le mélange de gaz dans le tube.

Nous avons testé plusieurs capteurs des deux classes, en remplissant une boîte en plastique avec une quantité prédéfinie de CO_2 (voir **fig. 6**). Dès le début de nos expériences, il est devenu évident que les capteurs de COV ne sont pas bien adaptés à un déploiement dans une salle de classe. Ils donnent une indication approximative de la qualité de l'air, mais nous voulons obtenir des concentrations aussi exactes que possible pour préconiser la ventilation d'une salle de classe. De plus, nous voulons comparer les résultats de différents locaux afin de tirer des conclusions.

Notre appareil de mesure du CO_2 utilise donc un capteur infrarouge non dispersif (IRND). Nous avons aussi comparé de nombreux capteurs IRND différents disponibles sur le marché. En particulier, nous avons fait



Figure 6. Test de différents capteurs avec des concentrations de CO_2 prédéfinies.

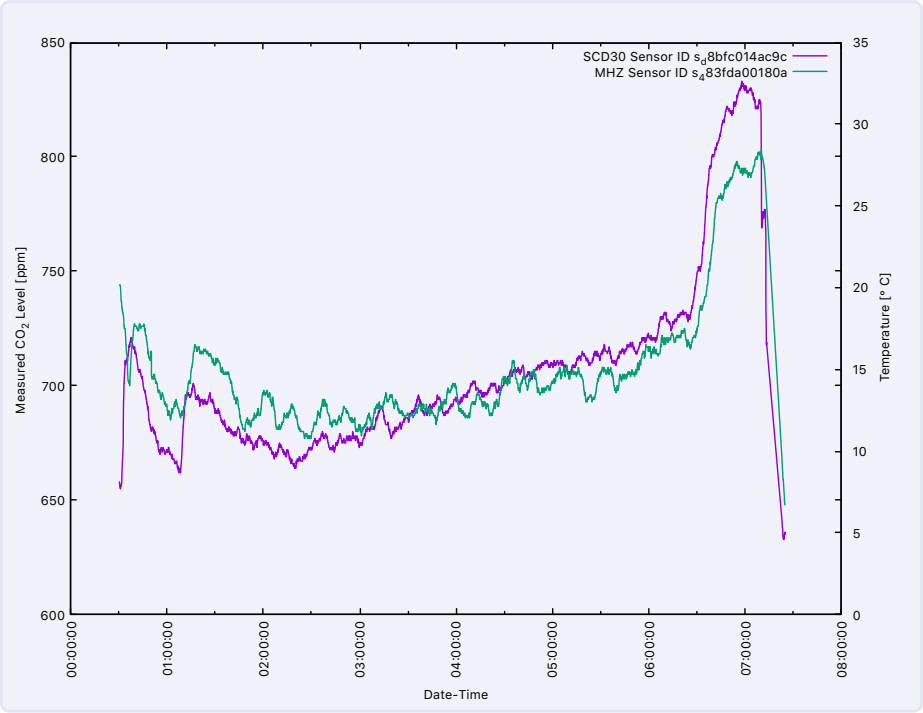


Figure 7. Comparaison entre un capteur de CO_2 MH-Z19 et un SCD-30 [8].

des expériences avec le Winsen MH-Z19B, le Sensair S8 et le Sensirion SCD-30. Nous voulions nous faire une idée de la précision et de la stabilité des mesures des différents capteurs IRND et de l'importance des écarts entre les différents lots d'un même capteur.
La **figure 7** montre une comparaison entre un MH-Z19 et un SCD-30 dans un scénario du monde réel. Elle montre que, qualitativement, les deux capteurs mesurent la même concentration de CO_2 dans ce scénario. De même, les écarts entre les capteurs de ce type se situent dans une fourchette acceptable. Cela dit, nous n'avons pris que des échantillons et nous sommes loin d'apporter des démonstrations issues de tests systématiques et à plus grande échelle.
Finalement, en tenant compte de la durée de vie du capteur, nous avons décidé de déployer le Sensirion SCD-30 dans nos appareils. Une fiche technique est disponible sur le site web de Sensirion [5].

Partenariat avec Elektor

Bien que le projet soit déjà terminé et bien fonctionnel, des choses pourraient être encore améliorées. Un étudiant de l'Université des Sciences Appliquées d'Aix-la-Chapelle et Mathias Claussen du laboratoire Elektor travaillent actuellement sur le logiciel, afin de le rendre encore plus modulaire et portable. Cela facilitera le changement des composants du projet – par exemple, le remplacement de l'ESP8266 par l'ESP32, plus puissant. Nous pourrions aussi utiliser différents modèles de capteur de CO_2.
En ce qui concerne la pandémie, ce n'est certainement pas une coïncidence si le laboratoire d'Elektor travaille également sur une autre plateforme de compteur de CO_2, basée dans ce cas sur un ESP32, et avec des circuits imprimés pour câbler les composants. Lorsque le logiciel sera entièrement portable, on pourra l'utiliser pour des projets de compteurs de CO_2, mais aussi pour d'autres projets avec des capteurs environnementaux, qui ont toujours des exigences et des fonctions très similaires à couvrir. Surveillez les articles dans les prochaines éditions de ce magazine !

Logiciel

Le cycle de vie du logiciel de l'appareil prévoit une phase de démarrage, où l'utilisateur peut configurer l'appareil, le démarrage du réseau, où les paramètres du réseau sont appliqués si nécessaire, et une phase de mesure, où les concentrations de CO_2 sont captées et affichées. (Voir le diagramme d'état de la **figure 8**).
Dans la phase de démarrage, après initialisation de l'appareil, celui-ci ouvre un point d'accès qui permet aux utilisateurs de se connecter à l'appareil et de le configurer. L'utilisateur peut alors accéder à l'appareil via un navigateur web et modifier différents paramètres de configuration, tels que le mode de l'appareil, les paramètres Wi-Fi et MQTT, régler l'horloge interne ou lancer l'étalonnage du capteur. Les paramètres sont stockés de manière persistante sur le disque flash de l'appareil.
Si aucun utilisateur ne se connecte au dispositif dans cette phase, après un délai d'environ 30 s, le dispositif passe à la phase de mesure. En fonction du mode de l'appareil, celui-ci se connecte à un point d'accès Wi-Fi externe ou au serveur MQTT spécifié.
L'intervalle de mesure de l'appareil est de 5 s : chaque nouveau point de données

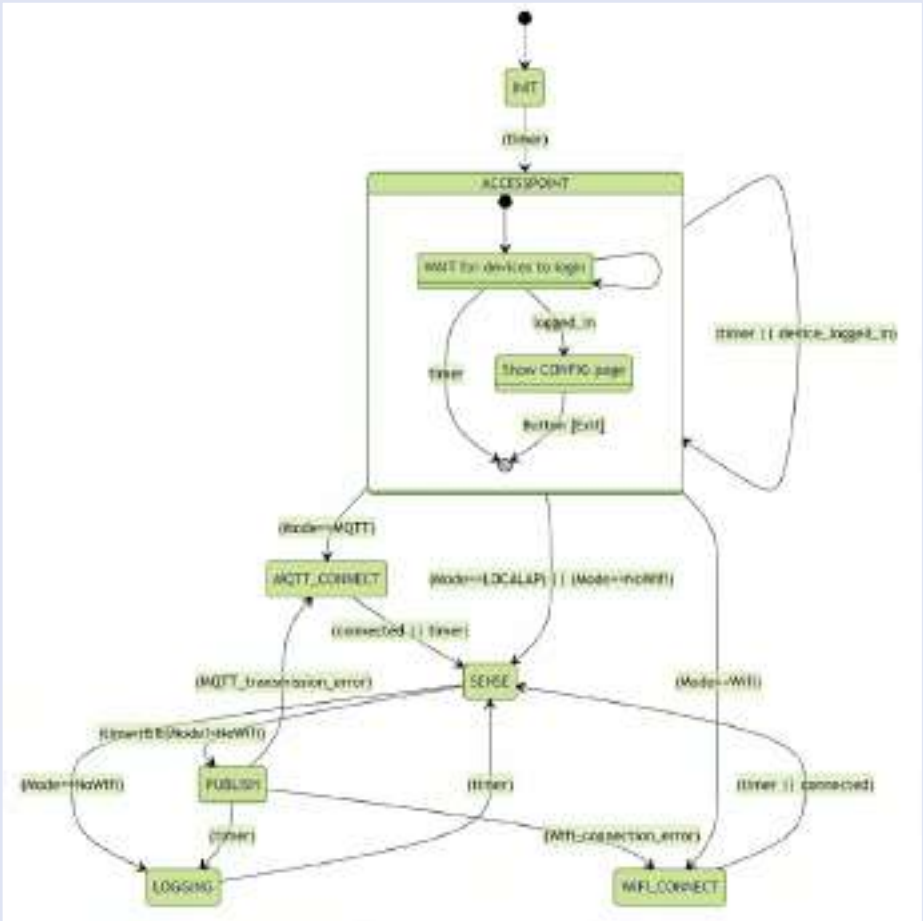


Figure 8. Automate fini implémenté dans le logiciel.

s'affiche sur l'écran OLED de l'appareil, le voyant correspondant du feu tricolore est allumé et la page web des données de l'appareil est mise à jour. Grâce à l'interface MQTT, il est particulièrement facile de traiter ultérieurement les données, par exemple en les stockant dans une base de données, en les affichant avec des outils tels que Node-RED, ou même en utilisant pour cela des services du nuage comme Grafana.
Le logiciel est disponible sous forme de fichier brut à l'adresse [4]. Les derniers fichiers sources seront disponibles sur GitHub [6].

Situation actuelle et perspectives

Les écoles, mais aussi les particuliers qui souhaitent obtenir un kit, peuvent prendre contact à l'adresse [7]. Plusieurs centaines de compteurs de CO_2 sont déjà en service dans des écoles et des jardins d'enfants allemands. Certains de ces compteurs participent à une étude de mesure de la qualité de l'air. Nous souhaitons y enregistrer et évaluer scientifiquement différents scénarios d'enseignement particulièrement problématiques en termes de qualité de l'air.

Concentration de CO_2

La concentration naturelle de CO_2 dans l'atmosphère est d'environ 400 ppm. Lorsque les gens expirent, ils émettent environ 4 % de dioxyde de carbone (environ 40000 ppm) qui se dispersent dans l'air de la pièce. Lorsque des personnes sont réunies dans une pièce sans ventilation, cela augmente inévitablement la concentration de dioxyde de carbone dans la pièce. Selon l'agence fédérale allemande pour l'environnement, des niveaux de CO_2 allant jusqu'à 1000 ppm sont encore estimés comme bons, tandis que des niveaux supérieurs à 2000 ppm sont considérés comme mauvais.

En outre, les auteurs et les autres membres de l'équipe travaillent actuellement sur une version du matériel alimentée par batterie. Celle-ci sera construite sur un petit circuit imprimé. L'objectif est d'intégrer ensuite la technologie des capteurs dans une infrastructure de bâtiment intelligent.

210388-04

Des questions, des commentaires ?

Envoyez un courriel aux auteurs (co2metergbr@gmail.com) ou contactez Elektor (redaction@elektor.fr).

Contributeurs

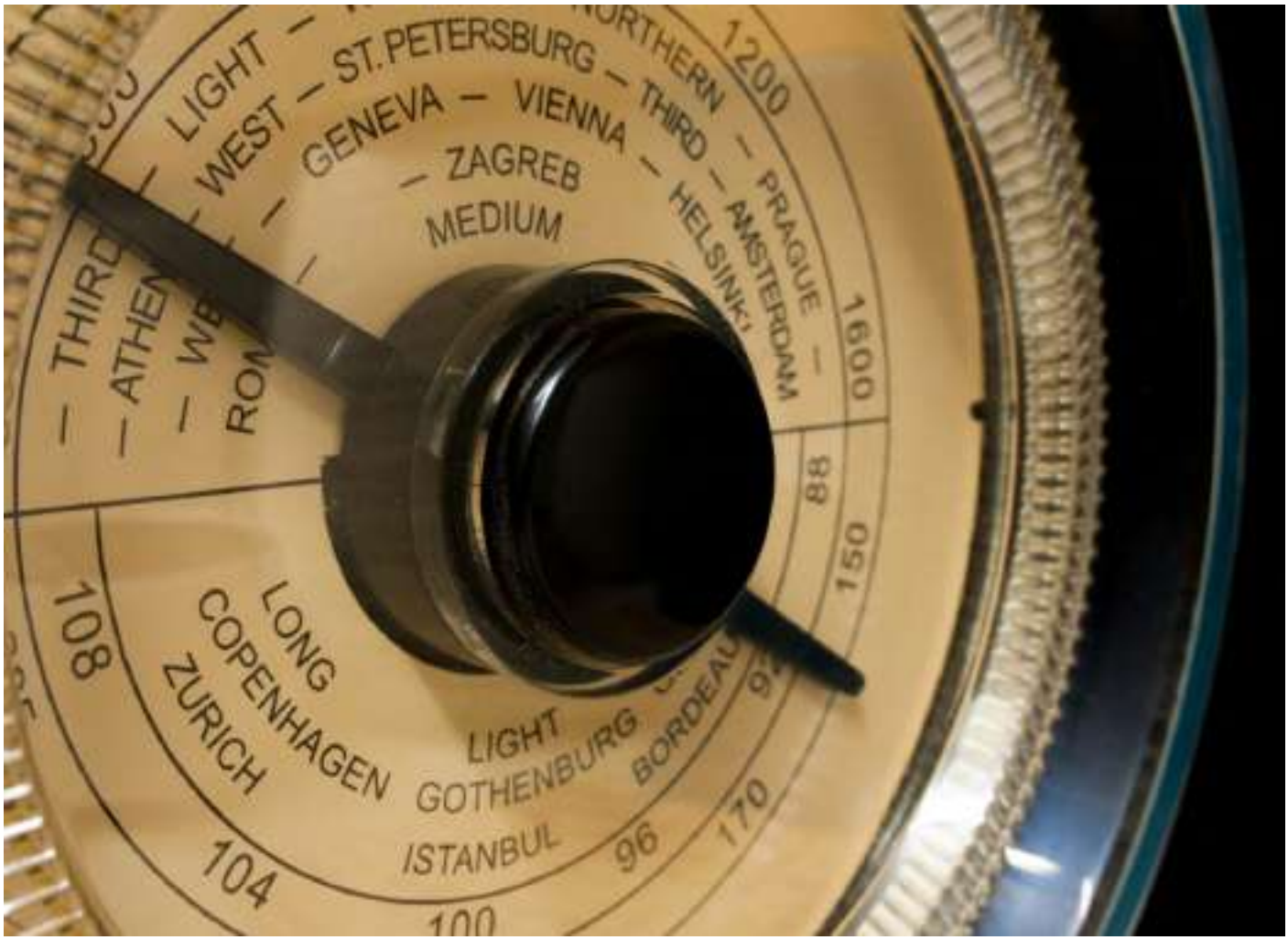
Texte et illustrations : **Thomas Dey, Ingo Elsen, Alexander Ferrein, Tobias Frauenrath, Michael Reke et Stefan Schiffer.**
Rédaction : **Jens Nickel**
Mise en page : **Harmen Heida**
Traduction : **Denis Lafourcade**

LIENS

[1] « Information About Aerosoles in Rooms » : https://bit.ly/2W1KFFX
[2] Plus d'informations sur le projet : https://maskor.fh-aachen.de/activities/CO2Meter/
[3] Makerspace du département de génie électrique et de technologie de l'information de l'Université des Sciences Appliquées d'Aix-la-Chapelle : http://makerspace.fh-aachen.de/
[4] Fichiers à code source ouvert du logiciel et du matériel du projet : https://makerspace-ac.de/seite/
[5] Sensirion, fiche technique du module capteur SCD30 Ver1, 05/2020 : https://bit.ly/3eXJa1V
[6] Dernière version du logiciel sur GitHub : https://github.com/co2mtr/co2meter.git
[7] Contact pour obtenir un kit : https://makerspace-ac.de/#contact-section
[8] T. Dey, I. Elsen, A. Ferrein, T. Frauenrath, M. Reke and S. Schiffer, « CO2 Meter: A do-it-yourself carbon dioxide measuring device for the classroom », PETRA 2021

MK484, radiorétro PO/GO

...Toujours le plaisir de construire !



Gert Baars (Pays-Bas)

La construction d'un récepteur radio a toujours fait partie des projets les plus gratifiants. Rien de nouveau ici : ce récepteur utilise un circuit intégré MK484 pour la détection AM et un vénérable LM386 comme ampli audio. Bien que très encombrée de bruits et désertée par les stations de radiodiffusion, la bande des petites ondes (PO) est parfaite pour expérimenter la réception radio.

Fonctionnement

La **figure 1** montre le circuit comportant trois étages RF pour recevoir et détecter les signaux radio AM dans la bande PO (MW en anglais, en principe de 500 à 1600 kHz).

Le transistor T1, un FET à jonction (JFET) J310, fournit une rétroaction positive au circuit d'accord d'antenne. Cela limite l'amortissement de la bobine, augmente la sensibilité et réduit la largeur de bande globale de la radio. L'antenne est constituée d'un barreau de ferrite de 120 à 200 mm de long, 10 mm de Ø avec une bobine à 55 spires jointives de fil de cuivre émaillé (ECW) de 0,2 mm de diamètre (~ #24 AWG). Le condensateur variable CV1 (500 pF, isolation mica ou air) accorde l'antenne sur la bande PO. Celle-ci peut être étendue jusqu'à environ 150 kHz en fermant le commutateur de bande S1.

Le transistor T2 (BF494) constitue le second étage. Il augmente la sensibilité du récepteur en amplifiant le signal PO transmis au troisième étage, IC1 (MK484). Ce circuit intégré à 3 broches remplace celui lancé en 1972 (!) par Ferranti sous le nom de ZN414. Il contient un amplificateur RF complet, un détecteur et un circuit de CAG (contrôle automatique de gain). Il a connu un vif succès pour la construction de récepteurs AM simples. Bien qu'obsolètes depuis longtemps, les circuits intégrés de la série ZN41x ont des équivalents modernes disponibles : le MK484 remplace le ZN414 à 3 broches.

Enfin, nous avons IC2 (le bon vieux LM386) qui fait office d'amplificateur audio et pilote le haut-parleur, avec le réseau Boucherot prescrit, formé de C11 et R10. Le LM386 évoquera bien des souvenirs chez de nombreux lecteurs.

Pour optimiser la réception, il suffit d'agir à l'oreille sur P1 qui contrôle la bande passante. On peut étendre la gamme d'accord aux GO en mettant le condensateur fixe C1 en circuit via S1. Mais cela introduit un nouvel amortissement qui peut trop réduire la rétroaction positive aux basses fréquences. Cet effet peut se compenser en réduisant la valeur de C3. Obtenir le compromis PO/GO idéal peut nécessiter de la patience et des expérimentations. Prendre une valeur pour C1 plus grande peut aussi aider à étendre la plage d'accord. Au-delà de la bande PO officielle (fréquences > 1,6 MHz), il y a peu de chances de trouver des stations AM (légales).

En mettant deux barreaux de ferrite côte à côte, on améliore la directivité et la sélectivité de cette antenne simple et bon marché.

Caractéristiques

Le récepteur fonctionne sur une alimentation de 6 à 12 VCC. Une pile de 9 V est idéale. Typiquement, il ne consomme que quelques mA. À pleine puissance audio, il faudra quelques centaines de mA. Le soir, dans de bonnes conditions de propagation, la réception en PO est optimale, et des stations distantes de 1000 km peuvent être reçues rien qu'avec l'antenne ferrite !

200372-04

Des questions, des commentaires ?

Envoyez un courriel au rédacteur (luc.lemmens@elektor.com) ou contactez Elektor (redaction@elektor.fr).

Contributeurs

Idée et conception : **Gert Baars**
Texte : **Gert Baars, Luc Lemmens**
Illustrations : **Patrick Wielders**
Rédaction : **Jan Buiting, Luc Lemmens**
Mise en page : **Giel Dols**
Traduction : **Yves Georges**

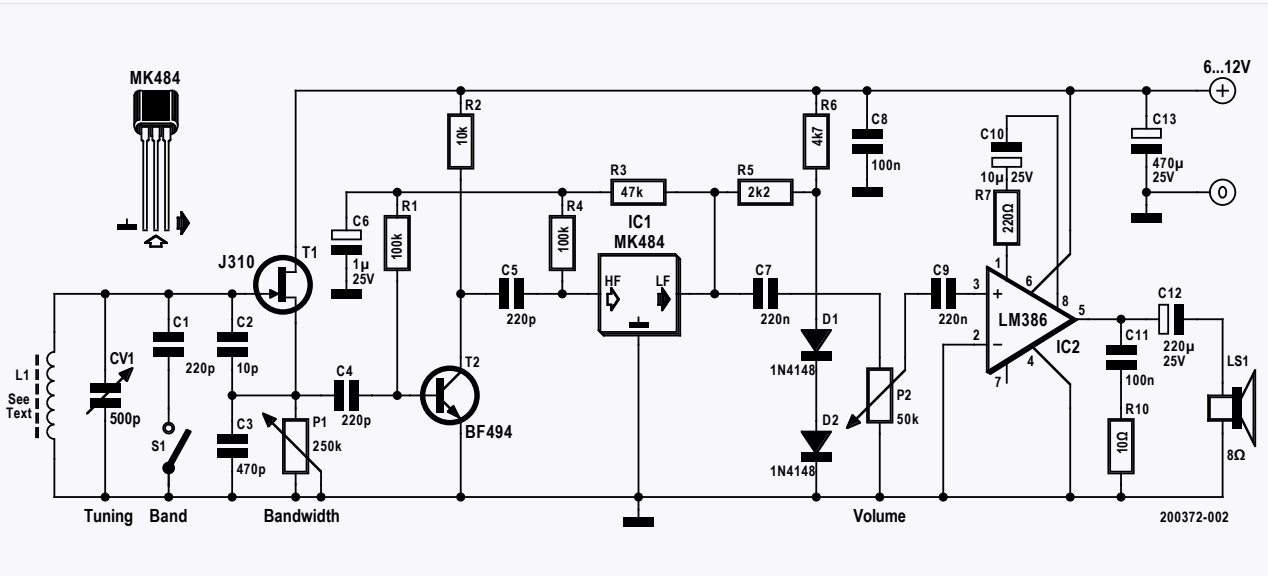


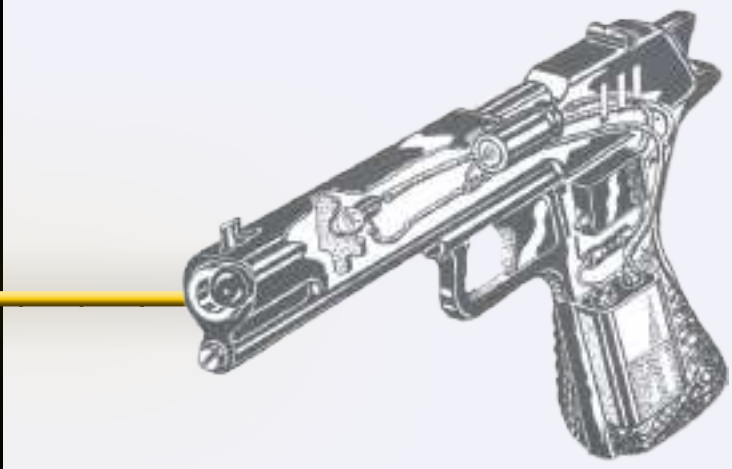
Figure 1. Le récepteur comprend trois étages RF et un amplificateur audio.

Que la lumière soit !

ELEKTOR 60 ans

Au fur et à mesure que l'hiver approche et que les nuits s'allongent, notre dépense en électricité pour l'éclairage augmente. Ces six dernières décennies, Elektor a proposé à ce sujet des idées intéressantes et novatrices. Voici une douzaine d'articles et de projets sur ce thème ! Ils mettent en évidence les progrès des LED, proposent d'astucieuses techniques de commande et des éclairages d'ambiance – pour les personnes et les plantes.

60 elektor
sixty › years › young



LEP – Light Emitting Pistol (1973, édition allemande seulement)

À l'époque, avant l'apparition des jeux vidéo de tir, ce circuit portatif composé d'une simple lampe à filament, d'une pile et d'un condensateur envoyait une impulsion lumineuse vers une cible. L'auteur, A. Schuylz, expliquait : « Un pistolet à rayon lumineux et affichage électronique du temps et du nombre de tirs rend l'exercice possible en salle de jeu. Outre l'affichage du nombre de coups, le jouet décrit ici mesure la vitesse de réaction. Le temps de latence entre la commande optique et le tir est ensuite affiché. » L'électronique, c'est amusant !
www.elektormagazine.de/magazine/elektor-197312/55393



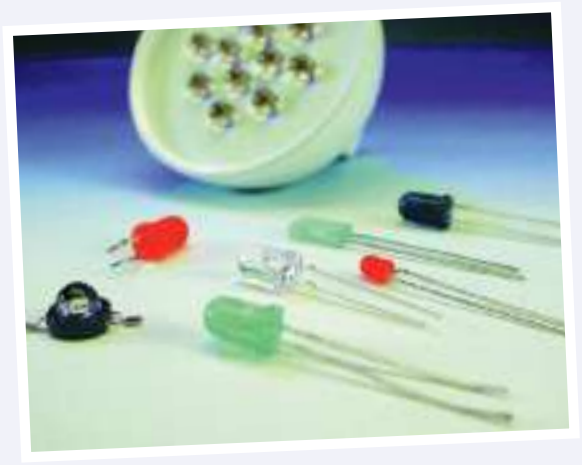
Fanal de secours à éclats portatif (1984)

Un signal de détresse portatif est pratique dans de nombreuses activités, de l'alpinisme aux longs trajets routiers. C'est l'objet de cette « fusée de détresse portative pour l'automobiliste en panne de moteur, le plaisancier en difficulté ou l'alpiniste en détresse ». La réalisation tient dans une petite mallette en plastique. Elle utilise un tube à éclats au xénon, alimenté par une batterie de voiture ou quatre piles de 1,5 V.
www.elektormagazine.fr/magazine/elektor-198406/52253



Twinkling Star (1988, édition allemande seulement)

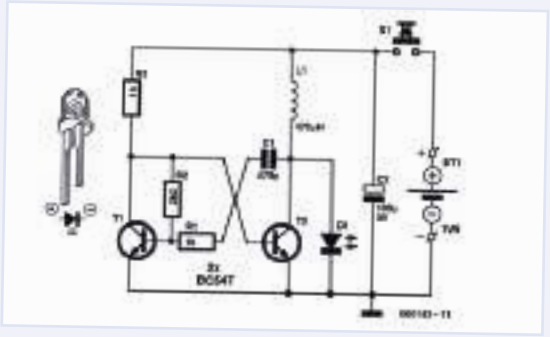
Vous le savez, les projets Elektor sont parfois futiles ! Cette étoile utilise un compteur binaire à 14 étages qui pilote une matrice de LED et ajoute une note de gaieté scintillante aux décors de fête. C'est l'un des nombreux projets Elektor sur le thème des fêtes. Tradition oblige !
www.elektormagazine.de/magazine/elektor-198812/48757



Plus de lumière – Les LED haute puissance dans la pratique (2003)

Ces dernières années, les LED ont vu leurs performances s'accroître considérablement. Elles sont de plus en plus souvent utilisées en lieu et place des lampes à filament. Elles présentent de nombreux avantages et leur fiabilité est souvent un élément important. En outre, la chute du prix des LED haute puissance les rend accessibles à la plupart des bricoleurs. Ce projet aborde le développement de ces produits et propose deux circuits de contrôle pour LED de 1 W.
www.elektormagazine.fr/030142

Téléchargement gratuit !



Appareil bio de « luminothérapie » à LED (2006)

En 2006, Jörg Trautmann expliquait : « Il est largement admis que la lumière peut avoir un effet positif sur la peau et l'âme humaine. À certaines longueurs d'onde, la lumière peut également lutter contre dépressions et allergies. » Cela peut paraître incongru, mais ce circuit thérapeutique simple trouve place dans un boîtier de tondeuse à poils de nez. Impossible de juger de son efficacité, mais il existe des produits similaires (plus chers) qui font la même chose.

www.elektormagazine.fr/magazine/elektor-200607/10528



Surround Light – L'approche analogique (2008)

Vous êtes fan de cinéma ou accro au jeu : ce projet est pour vous. Il mesure la sortie de couleur moyenne d'un écran (télévision ou moniteur) et des LED émettent cette même couleur en la projetant derrière l'écran. Cela produit une sensation plus intense, crée une référence visuelle et produit un éclairage d'ambiance raffiné. Les signaux de la sortie péritel sont utilisés comme source.

www.elektormagazine.fr/070491

Téléchargement gratuit !

Gradateur à touches à effleurement (2010)

Les circuits de gradateurs de lumière abondent. Dans ce projet du numéro d'été, un circuit intégré de LSI Computer Systems fournit une commande tactile avec mémorisation du réglage. Outre le contrôle tactile, le CI sauvegarde le réglage, ce qui permet par ex. d'allumer l'éclairage au niveau utilisé précédemment.

www.elektormagazine.fr/magazine/elektor-201007/11627



Réveil lumineux (2011)

Dans la nature, nos horloges biologiques sont synchronisées par la lumière du jour. Ce projet utilise l'heure précise du DCF77, un ATmega168, un afficheur LCD de 2×16 caractères et quatre boutons pour piloter une source lumineuse qui vous réveillera en douceur. En bref, le circuit combine les fonctions d'un gradateur contrôlé et d'une horloge de déclenchement. Ces deux fonctions sont implémentées dans le programme du microcontrôleur.

www.elektormagazine.fr/080850



Lumina : la lampe connectée par Bluetooth Low Energy (2016)

L'éclairage intelligent fait fureur. Ce contrôleur utilise des composants récents, un logiciel intelligent et la communication BLE pour détecter votre proximité et piloter un éclairage RGBW. Comme toujours, vous êtes invité à reproduire le circuit, mais aussi à en modifier et améliorer le design.

www.elektormagazine.fr/130226

Téléchargement gratuit !



Bio-lampe (2018) et Floranium (2018)

Les plantes produisent des signaux électriques, c'est prouvé. Ces deux projets de 2018 expliquent comment vous pouvez révéler ces signaux via des LED RVB. La Bio-Light est « un dispositif qui visualise les signaux électriques (biosignaux) produits (nous le pensons) par les plantes ». Plus qu'une décoration, le Floranium est un système extensible de mesure et d'expérimentation sur la physiologie des plantes.

www.elektormagazine.fr/160325
www.elektormagazine.fr/160670



Horticulture Box – éclairage de plantes (2019)

L'utilisation de lumière artificielle pour stimuler la croissance des plantes date de la 2e moitié du XIXe siècle. Aujourd'hui, l'usage horticole des LED est entré en pratique. Nous présentons ici une source à LED très sophistiquée et adaptée aux plantes. À l'ère de l'IoT, un ESP32 lui ajoute une interface web.

www.elektormagazine.fr/180583

210421-04

Contributeurs
Rédaction : **Jens Nickel**
Traduction : **Yves Georges**
Mise en page : **Harmen Heida**

PRODUIT

› **Archives Elektor 1974-2020 (clé USB)**
www.elektor.fr/19285

Quelle est votre couverture préférée d'Elektor?







Nous voulons utiliser tous les moyens existants pour rester en contact avec nos lecteurs. Nous avons donc récemment lancé une enquête sur les réseaux sociaux. Vos réponses via LinkedIn et Twitter nous ont vraiment surpris. Nous vous en remercions !

https://twitter.com/Elektor
www.linkedin.com/company/elektor-international-media

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 !

LCR-mètre 2 MHz d'Elektor



Prix : 799,00 €
Prix (membres) : 719,10 €

www.elektor.fr/19883

Anycubic Photon Mono X – imprimante 3D SLA à résine UV



Prix : 589,00 €
Prix (membres) : 530,10 €

www.elektor.fr/19831

Joy-Pi Note – solution 3 en 1 : bloc-notes, plate-forme d'apprentissage et centre d'expérimentation



Prix : 369,00 €

Prix (membres) : 332,10 €

www.elektor.fr/19877

Pokit Meter – multimètre portable, oscilloscope et enregistreur



Prix : 104,95 €

Prix (membres) : 94,46 €

www.elektor.fr/19854



Raspberry Pi 400 Buffer Board

Prix : 21,95 €

Prix (membres) : 19,76 €

www.elektor.fr/19884

DFRobot Bluno – carte compatible Arduino avec Bluetooth 4.0



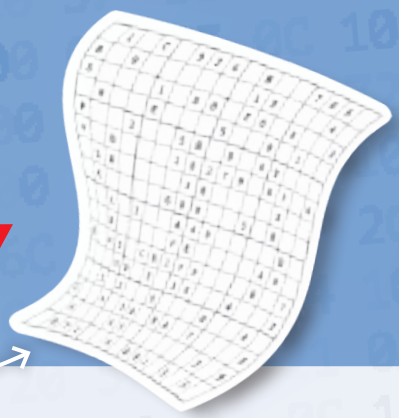
Prix : 32,95 €

Prix (membres) : 29,66 €

www.elektor.fr/19878

hexadoku

casse-tête pour elektorniciens



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 **17 décembre 2021** à l'adresse **hexadoku@elektor.fr**

LES GAGNANTS

La solution de la grille du numéro de septembre/octobre 2021 est **0718D.**

La liste des gagnants est publiée ici : www.elektormagazine.fr/hexadoku

Bravo à tous les participants et félicitations aux gagnants !

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

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

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.

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



Paul Eisler

Paul Eisler a inventé le circuit imprimé et a ainsi fait considérablement progresser la miniaturisation des circuits électroniques. En février 1943, Eisler a déposé le brevet 639.178 “Fabrication de circuits électriques et de composants de circuits” à Londres, où il était régulièrement signalé au Bureau américain des normes.
Lorsqu’une fusée de proximité pour les projectiles anti-aériens est développée aux Etats-Unis au début des années 1940, l’invention d’Eisler a été reprise et préparée pour la production en série. Ce fut le début du triomphe mondial du circuit imprimé.

Paul Eisler est un pionnier. **Soyez un pionnier** et réalisez votre idée dès aujourd’hui. **AISLER** fabrique le prototype.

Rapide, bon marché, écologique, directement de l’Europe.

Circuits imprimés

Cartes à 2 et 4 couches fabriquées industriellement dans un délai de deux jours ouvrables, par exemple 50mm x 50mm 2 couches pour 11,20 € y inclus le TVA et les frais de livraison.

Composants

Le plus grand stock de composants au monde. Peu coûteux et directement emballés pour votre projet, sans frais de livraison et sans montant minimale de commande.

Pochoirs SMD

Pochoirs SMD fabriqués au laser à partir de 10,00 € y inclus le TVA et les frais de livraison. Que ce soit à une ou à double faces et indépendamment du nombre de pads.

Assemblage

Vous n’avez pas assez de temps pour le montage de votre projet ? Pas de problème, nous pouvons aussi vous livrer votre projet complètement assemblé.

Visitez **aisler.net** maintenant et obtenez 10 € de réduction sur votre commande avec le code de réduction **TPRYFGB.**

AISLER