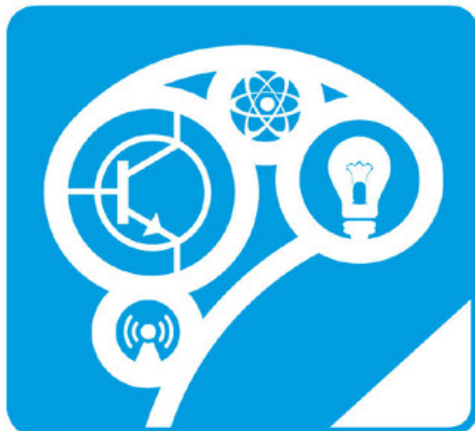




HACKABLE

MAGAZINE

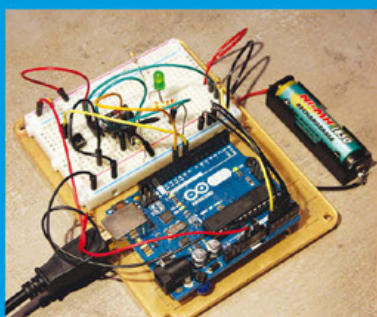
DÉMONTEZ | COMPRENEZ | ADAPTEZ | PARTAGEZ

France METRO : 7,90 € – CH : 13 CHF – BEL/PORT.CONT : 8,90 € – DOM TOM : 8,50 € – CAN : 14 \$ cad – TUNISIE : 18 TND – MAR : 100 MAD

~ ARDUINO ~

Testez la fiabilité
et vérifiez
l'état de santé
de vos piles
rechargeables

p. 60



~ ÉQUIPEMENT ~

Alimentation
stabilisée de
laboratoire :
pourquoi, comment
et combien ?

p. 14



~ DÉBUTER ~

Mauvaise
question : Arduino
ou Raspberry Pi ?

p. 04

~ UTILITAIRES ~

Rendez accessible
votre machine
Windows par la
Raspberry Pi
grâce à SSH

p. 90

(re)découvrez les nouvelles cartes RASPBERRY PI A+ et B+ !

p. 28

- L'indispensable pour bien débuter avec la Raspberry Pi
- Découvrez toutes les nouveautés des modèles A+ et B+
- Préparez et installez votre Pi avec Windows, Linux ou Mac OS X



~ ORDINOSAURE ~

Ordinosaure :
Ressuscitons
une machine
mythique :
NeXTstation Turbo

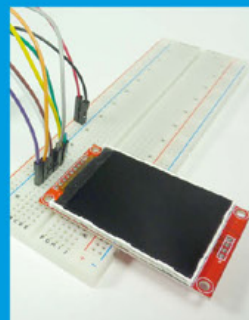
p. 78



~ AFFICHAGE ~

Les Arduino
aussi ont droit à
leurs afficheurs
LCD couleurs à 5
euros !

p. 22



L 19338-4-F-7,90 € - RD



7^{ème} Forum International de la Cybersécurité

20 ET 21 JANVIER 2015

LILLE GRAND PALAIS

PLUS D'INFORMATIONS SUR
www.forum-fic.com



**L'événement européen de référence
sur la cybersécurité**



**Cybersécurité
et Transformation Numérique**

contact@forum-fic.com

Un événement co-financé par



RÉGION
Nord-Pas de Calais

et organisé par



ÉDITO



Pas très envie...

Nous bouclons ce numéro en retard, et ce quelques jours après l'attaque ayant coûté la vie à nombre de nos confrères de Charlie Hebdo. Autre publication, autres sujets, autre domaine, autre approche, mais pourtant

le même travail, la même (dés)organisation et surtout la même motivation de vouloir (in)former, partager et faire réfléchir (et tantôt le même humour parfois déplacé).

Après le choc viennent la réflexion, l'introspection et les questions... surtout les questions. Pourquoi ? Qui ? Comment ? Que va-t-il se passer maintenant ? Quelles conséquences découleront de ce qui s'est passé ce 7 janvier 2015 ? Peut-on rire de tout ? Peut-on tout dire ? Où en sont vraiment nos libertés et notre liberté d'expression en tant que journalistes, mais aussi en tant qu'individus ?

Bien sûr, *Hackable* n'est pas un magazine politique ou satirique. Nous n'adressons pas les mêmes problèmes de société et pas avec la même irrévérence. Je ne me sens pas visé, mais je me sens vraiment concerné. Par ce qui se passe et ce qui va se passer, en particulier l'inéluctable réduction de nos libertés, sous couvert de la crainte d'un côté, mais aussi de la « nécessité » d'imposer plus de surveillance de l'autre...

Difficile donc de vous gratifier d'un éditto guilleret, vindicatif ou édifié à propos de telle ou telle dérive ou technologie. Mais ma motivation, notre motivation, profonde elle ne faiblira pas.

Alors les gars, si vous nous voyez de là-haut (ou de là en bas), continuez à bien « foutre le bordel », on ne vous oubliera pas et surtout, on n'oubliera pas que la Liberté est notre bien le plus précieux. La Liberté d'agir, d'apprendre, de partager et d'exprimer nos idées sans retenue aucune !

Denis Bodor

Hackable Magazine

est édité par Les Éditions Diamond



B.P. 20142 - 67603 Sélestat Cedex
Tél. : 03 67 10 00 20 - Fax : 03 67 10 00 21
E-mail : lecteurs@hackable.fr

Service commercial : cial@ed-diamond.com

Sites : boutique.ed-diamond.com

Directeur de publication : Arnaud Metzler

Rédacteur en chef : Denis Bodor

Réalisation graphique : Kathrin Scali

Responsable publicité : Valérie Fréchar, d

Tél. : 03 67 10 00 27 v.frechard@ed-diamond.com

Service abonnement : Tél. : 03 67 10 00 20

Impression : pva, Landau, Allemagne

Distribution France : (uniquement pour les dépositaires de presse)

MLP Réassort : Plate-forme de Saint-Barthélemy-d'Anjou. Tél. : 02 41 27 53 12

Plate-forme de Saint-Quentin-Fallavier.

Tél. : 04 74 82 63 04

IMPRIMÉ en Allemagne - PRINTED in Germany

Dépôt légal : À parution,

N° ISSN : en cours

Commission paritaire : en cours

Périodicité : bimestriel

Prix de vente : 7,90 €

La rédaction n'est pas responsable des textes, illustrations et photos qui lui sont communiqués par leurs auteurs. La reproduction totale ou partielle des articles

publiés dans Hackable Magazine est interdite sans accord écrit de la société Les Éditions Diamond. Sauf accord particulier, les manuscrits, photos et dessins adressés à Hackable Magazine, publiés ou non, ne sont ni rendus, ni renvoyés. Les indications de prix et d'adresses figurant dans les pages rédactionnelles sont données à titre d'information, sans aucun but publicitaire. Toutes les marques citées dans ce numéro sont déposées par leur propriétaire respectif. Tous les logos représentés dans le magazine sont la propriété de leur ayant droit respectif.



Suivez-nous sur Twitter

[@hackablemag](https://twitter.com/hackablemag)

SOMMAIRE

REPÈRE & SCIENCE

04

Spécial débutant : « Dois-je choisir Arduino ou Raspberry Pi ? »

ÉQUIPEMENT

14

Alimentation de laboratoire

ARDUINO'N'CO

22

Utilisez un écran LCD à 5€ avec votre Arduino

EN COUVERTURE

28

Raspberry Pi : ce que vous devez savoir avant de commencer

40

Raspberry Pi « plus » : quoi de neuf docteur ?

48

Installation et premier démarrage de la Raspberry Pi B+

TENSION & COURANTS

60

Testeur de piles rechargeables

DÉMONTAGE, HACKS & RÉCUP

78

Les bonnes leçons du passé ou la résurrection d'une NeXTstation Turbo

EMBARQUÉ & INFORMATIQUE

90

SSH ou comment avoir une ligne de commandes via le réseau

ABONNEMENT

33

Abonnements tous supports

89

Offres spéciales professionnels



SPÉCIAL DÉBUTANT : « DOIS-JE CHOISIR ARDUINO OU RASPBERRY PI ? »

Denis Bodor



Voici une question qui revient assez fréquemment du côté des messages reçus de la part des nouveaux lecteurs. Ceci est parfaitement compréhensible, car à première vue et sans expérience aucune, Arduino et Raspberry Pi peuvent paraître très similaires : un circuit imprimé, plein de connecteurs, une utilisation très orientée « bidouille », un écosystème open source, des tutos à la pelle pour faire tout et n'importe quoi... Pourtant les deux plateformes sont radicalement différentes. Quelques explications s'imposent...

A la question « me conseillez-vous plutôt Arduino ou Raspberry Pi pour commencer ? », il n'y a pas de vraie réponse. Pas même un « ça dépend ». Les deux plateformes proviennent de mondes différents et possèdent une philosophie tout autant disparate. Arduino est une famille de cartes construites autour de microcontrôleurs alors que les Raspberry Pi, quelque soit le modèle, reposent sur ce qu'on appelle un SoC ou *System on Chip*. Ce sont là de jolis termes, mais en creusant un peu, on se rend compte que finalement, par bien des aspects, les SoC sont de gros microcontrôleurs et que cette désignation est surtout commerciale et marketing. Pour ajouter à la confusion, certains fabricants, comme Cypress appellent leurs composants des PSoC ou *Programmable System-on-Chip* alors qu'il s'agit de microcontrôleurs (8051 ou Cortex M) couplés, sur la puce, à des éléments programmables numériques ou

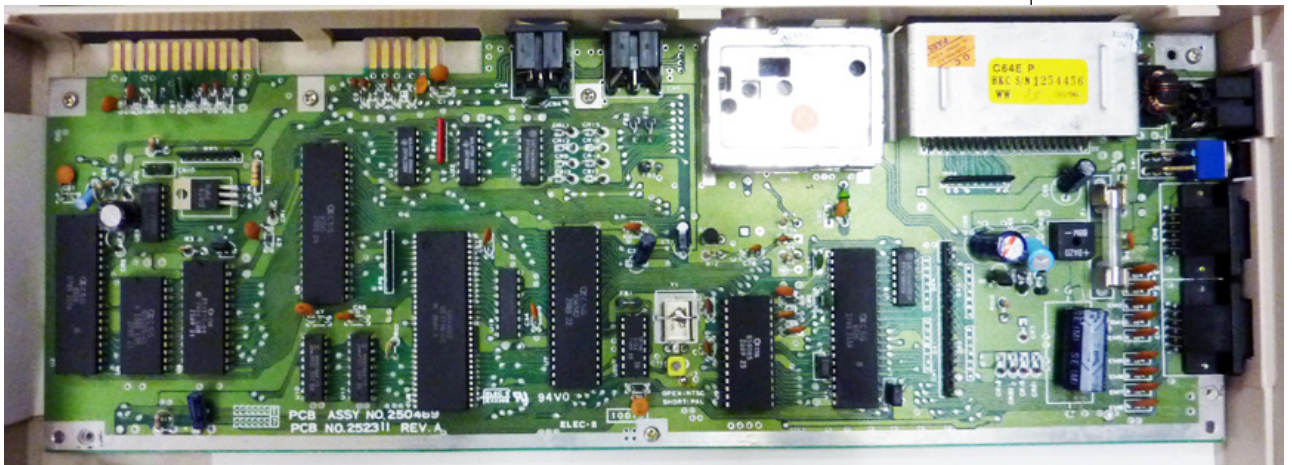
analogiques. Différencier microcontrôleurs et SoC, et ainsi Arduino et Raspberry Pi, nécessite quelques connaissances de base sur le fonctionnement des processeurs et des premiers ordinateurs.

1. AU DÉBUT, TOUT ÉTAIT BIEN SÉPARÉ !

Réglons la date de destination et activons le convecteur temporel de la DeLorean sur 1982, année de commercialisation de l'ordinateur personnel et familial sans doute le plus vendu au monde : le Commodore 64 (je pourrai prendre n'importe lequel de cette époque, mais moi, petit, j'avais un C64). Cet ordinateur repose sur le processeur ou microprocesseur MOS 6510. Mais qu'est-ce qu'un processeur exactement ? Ce que les anglophones appellent un CPU (pour *Central Processing Unit*) ou unité centrale de traitement, est l'élément d'un ordinateur qui exécute des instructions. Un ensemble d'instructions s'appelle un programme. On parle de microprocesseur (ou plutôt on « parlait ») quand cet élément prend la forme d'un seul composant.

Le processeur seul ne peut finalement pas faire grand-chose parce qu'il lui faut un endroit où lire les instructions qu'il doit exécuter et les données sur lesquelles doivent opérer ses actions. Ces données sont traitées en morceau par le processeur. Le 6510 du Commodore 64 est un processeur 8 bits. Cela signifie qu'il peut traiter les données par paquet de 8 bits (0 ou 1).

La carte mère d'un Commodore 64 (fin des années 80). L'ensemble des composants noirs forme l'ordinateur complet : mémoire, CPU, ROM, contrôleur, périphérique, audio, vidéo. Tout ceci réuni dans une seule puce est appelé un microcontrôleur ou selon la nature et l'usage, un SoC.





Le circuit intégré de droite est l'un des deux CIA du Commodore 64. C'est une interface programmable permettant de dialoguer avec le monde extérieur. Celui-ci se trouve juste à côté du connecteur pour le clavier du micro-ordinateur et est en charge de sa gestion. Ce type de composant est une partie intégrante d'un microcontrôleur ou un SoC.

Si une donnée est d'une taille plus importante, comme une valeur sur 16 bits par exemple, elle doit donc être traitée en deux fois. Empiriquement, on peut donc dire qu'un processeur 8 bits devra faire deux fois plus de travail pour opérer sur une valeur de 16 bits qu'un processeur 16 bits. Les processeurs actuels fonctionnent en 64 bits et donc manipulent en une opération des blocs de données 8 fois plus gros que ne pouvait le faire un 6510 à son époque. Ce n'est, bien entendu, qu'un seul élément déterminant la puissance d'un processeur. Le 6510 fonctionnait à environ 1 Mhz, soit un million de cycles par seconde, chaque instruction pouvant nécessiter un ou plusieurs cycles pour être exécutée. Un core i7 Intel comme le 4790K fonctionne à 4 Ghz, soit une fréquence 4000 fois plus importante ! D'autres technologies entrent bien entendu en jeu aujourd'hui, et comparer un 6510 à un core i7 est un peu comme tenter de mettre en concurrence une roue de carrosse en bois avec un train d'atterrissage d'Airbus A380...

Pour traiter les données, les lire et stocker un résultat, ce qu'il faut au processeur, c'est de la mémoire. Le vénérable Commodore 64 utilisait 64 Ko (65536 octets) de mémoire. Celle-ci prenait la forme, dans la plupart des versions de la machine, de huit composants HM 4864 permettant chacun de stocker 8 Ko (64 kilo bits). Cette mémoire appelée mémoire vive ou RAM pour *Random Access Memory* est volatile. Ceci signifie que lorsqu'elle n'est plus alimentée, son contenu disparaît. Pourtant, lorsqu'on allume un Commodore 64, on se

retrouve avec un écran bleu et un langage de programmation directement utilisable (BASIC). D'où sort-il alors ?

Ce programme que l'utilisateur obtient dès la mise sous tension n'est bien entendu pas stocké en RAM, mais en ROM, la *Read Only Memory*. C'est un autre composant qui matérialise cette mémoire qui, comme son nom l'indique, n'est accessible qu'en lecture. La ROM est inscrite une fois, à l'usine, avec différents programmes et donnée comme la base du système, le langage BASIC ou encore une table de caractères. Sans entrer dans le détail du fonctionnement du Commodore 64, lorsque l'ordinateur est allumé, le processeur utilise ces programmes et « fusionne » une partie de la RAM avec la ROM, ne laissant à l'utilisateur que 32 Ko pour ses programmes, les 32 autres Ko étant « consommés » par le BASIC.

Mais ce n'est pas tout. Pour afficher des choses sur l'écran, qui à l'époque était une simple télévision, un autre composant entre en jeu. Le processeur ne sait pas faire cela, il ne fait d'exécuter des instructions qui se résument à des opérations arithmétiques ou logiques et des lectures/écritures en mémoire. Le processeur repose donc sur une puce graphique VIC-II (*Video Interface Chip II*) connectée au processeur. C'est donc un périphérique qui contrôlé par le processeur va créer un signal vidéo qui, traité par un ensemble de composants analogiques, va finir par créer l'image à l'écran. En fonction du modèle d'ordinateur et de la région

où il est commercialisé, le signal produit est PAL (Europe) ou NTSC (USA) et la puce graphique d'un type différent (MOS 6567, 8567, 6569 ou 8569). Ce que fait le processeur ou plus exactement l'ensemble d'instructions exécutées par ce dernier, c'est simplement dire à la puce VIC-II des choses comme « dessine un pixel là », « affiche le caractère A en vert ici » ou « remplis un bloc de tant par tant en rouge à cet endroit ».

Il en va de même pour l'audio qui prend la forme d'un autre périphérique, lui aussi matérialisé en une puce dédiée : le SID pour *Sound Interface Device*. Ce composant, MOS 6581 ou 8580, fonctionne de la même manière que le VIC-II. Le processeur donne des ordres au composant qui produit alors un son selon ses directives.

Enfin, deux autres périphériques du processeur étaient également de la partie afin de permettre aux utilisateurs de brancher un lecteur de disquettes, des joysticks ou encore une imprimante. Ces deux périphériques prenaient eux aussi la forme de composants : les CIA pour *Complex Interface Adapter* (MOS 6526).

Tout ce petit monde prenait place sur un circuit imprimé de la taille de la machine en compagnie de l'alimentation, d'un ensemble de composants passifs (résistances, condensateurs, etc.) et de connecteurs divers. Les pistes du circuit connectaient tous les composants de manière à ce qu'ils puissent communiquer, le tout orchestré par le maître des lieux, le microprocesseur.

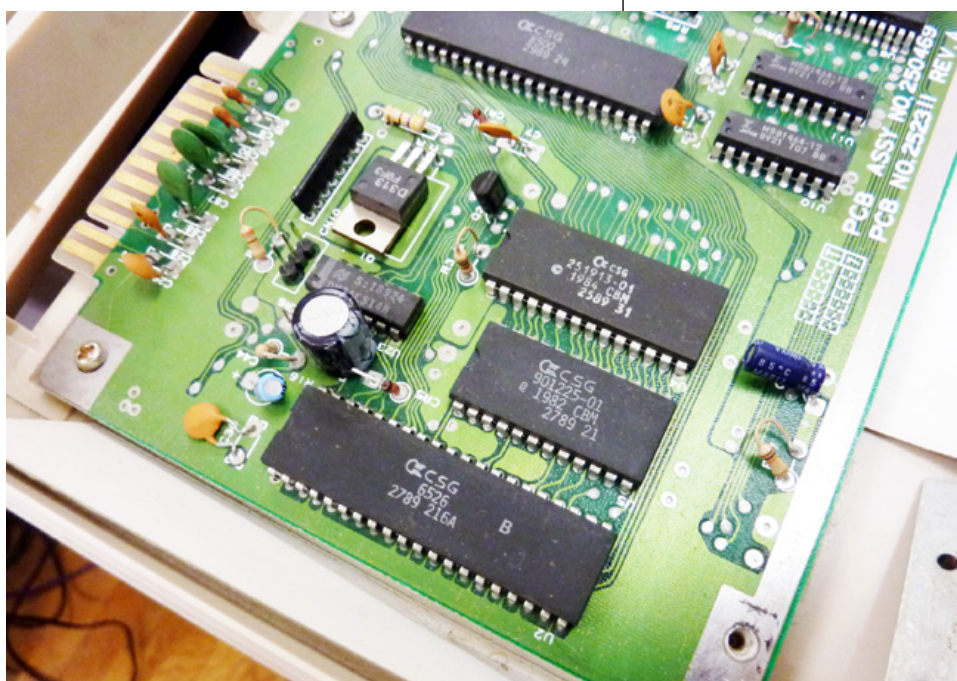
2. C'EST BIEN GENTIL LES VIEUX TRUCS, MAIS...

« C'est quoi le rapport » me direz-vous. C'est tout simple, prenez le circuit imprimé du Commodore 64 et ratatinez-le (gentiment) jusqu'à le faire tenir dans un seul composant : bravo, vous venez de créer un microcontrôleur. Observez maintenant un Arduino Uno, sur la carte se trouve un gros composant marqué ATMEGA328P. Ceci est un microcontrôleur Atmel AVR contenant :

- un processeur 8 bits,
- une zone de stockage pour vos programmes de 32 Ko,
- une mémoire volatile de 2 Ko,
- et un ensemble de périphériques.

Bien entendu, dans ce composant, il n'y pas de puce graphique ou audio comme le VIC-II ou le SID, mais on trouve des choses qui se rapprochent des CIA du Commodore 64 : des périphériques série pour communiquer, des convertisseurs analogique/numérique pour lire des tensions, des GPIOs pour lire et écrire des 0 et des 1 sur les broches, des *timers* qui sont des petites horloges et même un capteur de température.

Contrairement à une Arduino, un Commodore 64 n'a pas de mémoire flash pour le stockage des programmes. Sur cette photo, nous avons de haut en bas, le processeur (version CSG du 6510), la mémoire ROM contenant le BASIC, la ROM contenant la table de caractères et un CIA pour les périphériques externes.





La version CSG du SID, le composant audio permettant de produire le son et la musique caractéristiques de l'époque 8 bits. Notez qu'il se trouve enfiché sur un emplacement et peut donc être facilement retiré et remplacé. Ce composant peut être interfacé avec une carte Arduino pour produire du son !

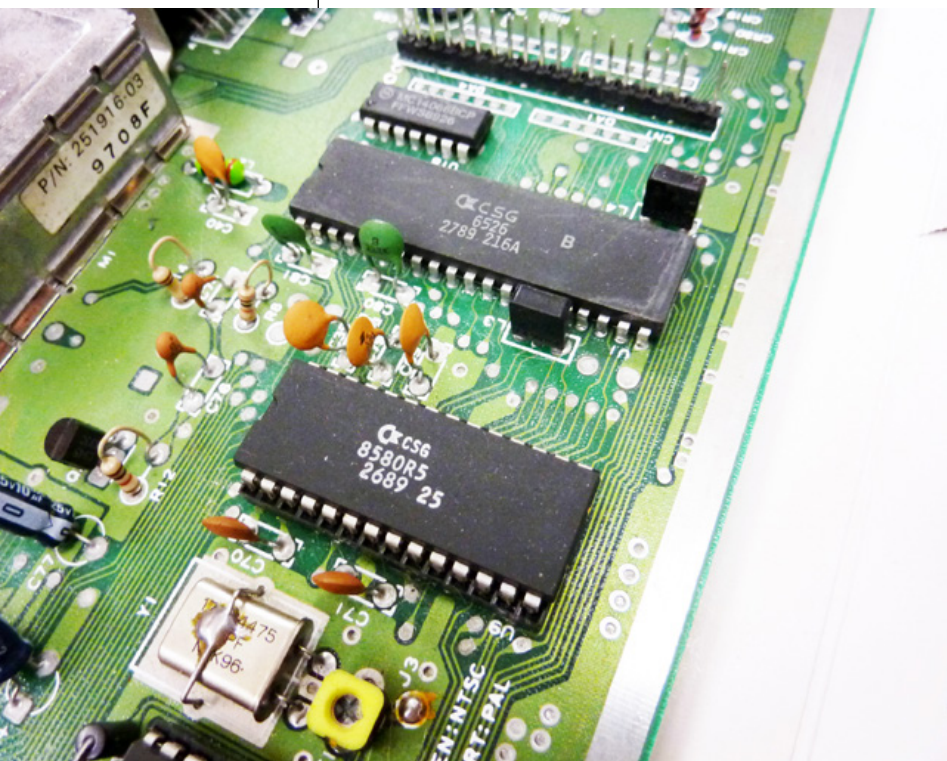
Il n'y a plus de ROM contenant le langage BASIC qui permet d'écrire des programmes. À la place, nous avons une mémoire flash réinscriptible qui vous permet d'enregistrer votre programme que le processeur va automatiquement exécuter à la mise sous tension ou après une réinitialisation. En d'autres termes, nous troquons la ROM et son contenu défini par le fabricant contre une « ROM » réinscriptible où vous mettez ce que vous voulez.

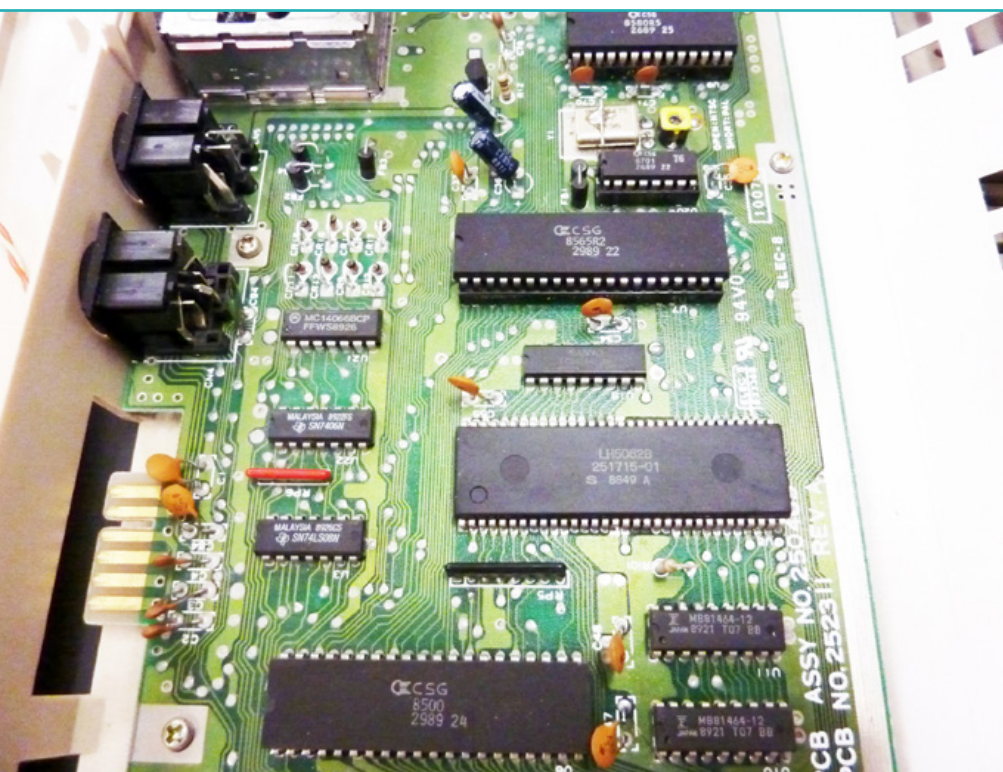
Techniquement parlant, en termes de programmation, rien n'a vraiment changé entre le bon vieux Commodore 64 et un AVR d'Atmel. Les instructions qui composent le programme sont lues depuis la mémoire de stockage et exécutées par le processeur. Ces instructions manipulent des données et permettent de donner des ordres aux périphériques. Ce programme est seul maître à bord. Il fait ce qui lui chante sans avoir de comptes à rendre à qui que ce soit. S'il fonctionne correctement, ou en d'autres termes que vous avez écrit un bon programme sans bogue, tout va bien. Dans le cas contraire, le processeur exécutera tout de même les instructions que vous lui avez données, même si celles-ci doivent conduire à une erreur ou, tout simplement, à l'arrêt du système.

Ceci n'est pas anodin. Prenons un exemple concret en imaginant un programme qui va mettre une broche du microcontrôleur à +5V, attendre une seconde et mettre cette même broche à la masse. Vous créez le programme, le chargez dans la mémoire flash et redémarrez le microcontrôleur. Immédiatement, la broche est à +5V et une seconde après à la masse et... et puis c'est tout ! Diantre ! Vous avez oublié de prévoir dans votre programme une petite instruction qui dit « recommence ». Le processeur dans le microcontrôleur a fait EXACTEMENT ce que vous lui avez dit de faire et maintenant, il n'a pas d'autre instruction et est donc tout bonnement à l'arrêt. Vous ne pouvez littéralement rien faire d'autre que de réinitialiser le microcontrôleur ou, plus intelligemment, corriger votre programme et l'écrire à nouveau dans la mémoire flash.

Être seul maître à bord c'est bien. On peut tout faire. Mais... il faut tout faire ! Ce n'est pas comme sur un PC ou, si le programme s'arrête, on le relance. Avec un microcontrôleur, votre programme commande directement les actions. C'est une partition pour un seul instrument. Vous êtes le compositeur et le musicien c'est le microprocesseur dans le microcontrôleur. Ceci signifie également que du point de vue du processeur, le programme exécuté fait absolument tout et ne repose sur aucun autre élément logiciel.

Bien entendu, lorsque vous programmez votre carte Arduino vous pouvez utiliser des bibliothèques permettant par exemple d'accéder aux fonctionnalités offertes par les





De haut en bas, le SID, le contrôleur vidéo VIC-II, le contrôleur de mémoire et le processeur. Notez la présence des deux petites puces en bas à droite, ceux-ci forment les 64 Ko de RAM du Commodore 64.

3. SYSTÈME D'EXPLOITATION ET COMPLEXITÉ

shields. Cependant, en utilisant le bouton **Vérifier** de l'interface de développement (IDE) Arduino, vous allez compiler le programme ou croquis afin de le rendre intelligible par le processeur. Les bibliothèques seront alors fusionnées avec votre croquis et l'ensemble sera transformé, en plusieurs étapes, en une suite d'instructions en langage machine que vous pourrez ensuite inscrire dans la mémoire flash (ou « Téléverser » comme spécifié dans l'IDE Arduino). Il n'y aura donc bel et bien qu'un seul programme qui fonctionne dans le microcontrôleur et celui-ci utilise directement le matériel.

Ceci est précisément ce qui différencie l'utilisation d'une carte Arduino d'une plateforme comme la Raspberry Pi.

Exactement comme on peut le voir aujourd'hui avec les smartphones, avec le gain en puissance de calcul et de traitement arrive le désir pressant de pouvoir faire fonctionner plusieurs programmes en même temps sur une seule et unique machine. Ceux qui se souviennent de l'époque glorieuse de MS/DOS se remémorent également son caractère monotâche ou, autrement dit, le fait de ne pouvoir utiliser qu'une seule application à la fois (sauf si on « trichait » en utilisant un programme dit « résident » ou TSR, mais c'est une autre histoire).

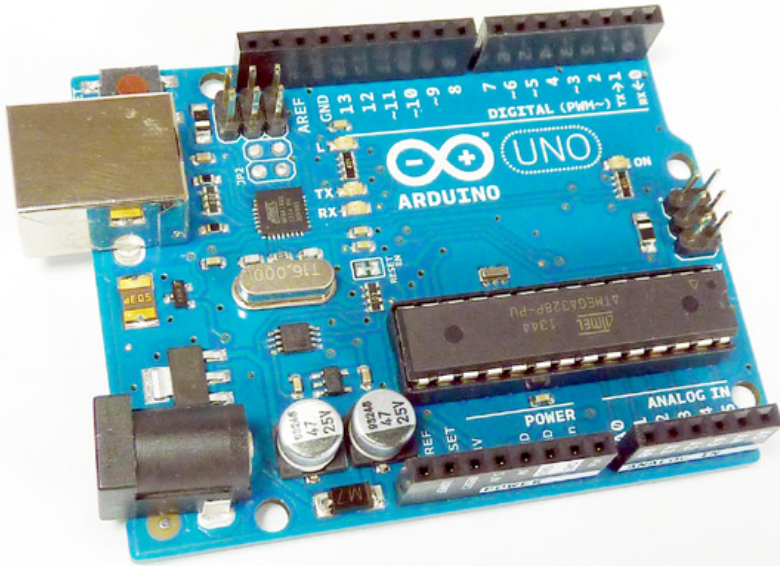
MS/DOS n'avait pas pour objectif de permettre de faire fonctionner plusieurs programmes ou applications. Sa tâche était de servir de « couche intermédiaire » entre les programmes des utilisateurs et le matériel. Revenons rapidement à

notre carte Arduino et à la notion de bibliothèques. Lorsque vous souhaitez activer une broche en sortie de la carte, vous utilisez dans vos croquis la fonction ou directive **digitalWrite()**. Peu importe que vous l'utilisiez sur une carte Arduino Uno, Leonardo ou Mega, cela fonctionne de la même manière. Vraiment ?

En réalité, pas tout à fait. Chacune de ces cartes utilise un microcontrôleur d'un modèle sensiblement différent, respectivement ATmega328, ATmega2560 et ATmega32u4. Le code compilé, celui compris par le processeur, à destination d'un microcontrôleur ne fonctionnera pas « tel que » sur un modèle différent. Vous pourrez effectivement utiliser le même croquis, mais devrez le recompiler pour chaque modèle de carte Arduino.

Ceci vous fournit un modèle d'abstraction. Les programmes que vous allez écrire n'ont pas besoin d'être adaptés d'un modèle de carte à l'autre. C'est un avantage non négligeable, car vos croquis peuvent ainsi, sans modification, fonctionner chez tous les possesseurs de carte Arduino et même, sous certaines conditions, avec des cartes complètement différentes comme les Launchpad de Texas Instruments.

À l'époque de MS/DOS, les ordinateurs étaient construits autour de processeurs Intel et AMD (8086, 80286, 80386, 80486, Pentium, etc.). Ils étaient donc matériellement différents, mais pourtant, des applications comme le tableur Multiplan, le traitement de texte Word ou encore la base de données dBase étaient livrés



Une carte Arduino Uno. La grosse puce rectangulaire est le microcontrôleur Atmel AVR ATmega328P. Celui-ci regroupe le processeur, la mémoire, la mémoire flash, et les périphériques. Les autres composants de la carte servent à l'alimentation en courant et à la connexion du microcontrôleur avec le monde extérieur.

sur disquettes et fonctionnaient exactement de la même manière, quels que soient le modèle et la marque de votre ordinateur de type PC. Mieux encore, que vous ayez de simples lecteurs de disquettes souples 5"1/4, 3"1/2 ou la chance de disposer d'un disque dur (type MFM de 12 Mo) les applications étaient capables d'enregistrer des fichiers de la même manière.

Pourquoi ? Simplement parce que MS/DOS fournissait une interface pour cela. Exactement de la même manière que vous n'avez pas à vous soucier de savoir sur quelle carte Arduino vos croquis vont être utilisés, les applications n'ont pas à se soucier du modèle de processeur ou du type de support de stockage utilisé. Les applications demandent simplement « je veux un fichier à cet endroit, voici les données à mettre dedans » et le système se charge d'organiser tout cela et de procéder à l'enregistrement. Cette couche intermédiaire entre le matériel et les applications est un système d'exploitation.

Le système d'exploitation permet ainsi de garantir qu'une application fonctionnera de la même manière avec une machine et une autre, pour peu que le système d'exploitation soit le même. Attention cependant, les programmes ou applications restent dépendants du processeur, car composés d'instructions qui lui sont destinées. Un certain niveau de compatibilité est généralement possible. Ainsi une application pour MS/DOS fonctionnera de la même manière avec un 80286 et un 80386 parce que le 80386 dispose d'une compatibilité ascendante, c'est-à-dire qu'il supporte les instructions du 80286, mais en ajoute de nouvelles. La compatibilité descendante ou

rétrocompatibilité est bien plus rare. Les « demandes » ou les appels système, eux, sont propres au système d'exploitation et couvrent toute une gamme de services allant de l'ouverture d'un fichier à la lecture du clavier.

Fournir cette couche d'abstraction n'est qu'une des tâches d'un système d'exploitation. Avec l'augmentation de la puissance et des volumes de mémoire, est arrivée la notion de multitâche et tout un florilège de problèmes associés. Lorsque deux programmes fonctionnent en même temps, il est impératif de s'assurer qu'ils ne se marchent pas dessus ou plutôt que l'un n'utilise pas les ressources de l'autre. Un arbitrage est donc mis en place par le système d'exploitation qui doit gérer qui utilise quoi, qui a droit à quoi et qui passe devant qui. Hé oui, même si vous avez l'impression que tous les programmes fonctionnent en même temps, les ressources sont limitées et il n'y a qu'un processeur (de nos jours les processeurs multicœurs sont devenus le standard, mais nous érudons délibérément cet état de fait afin de garder les explications les plus intelligibles possible). Le système d'exploitation doit donc également passer le relais à chaque programme qui souhaite utiliser le processeur pour traiter ses données. Il le fait si rapidement que vous avez effectivement l'impression que tout fonctionne en même temps, mais ce n'est pas le cas.

Comme vous pouvez le constater, si vous écrivez un programme dans un tel contexte, il ne s'agit plus d'écrire une partition pour un seul musicien, mais plutôt de vous

intégrer dans un orchestre. Le chef d'orchestre est monsieur le système d'exploitation qui décidera quand votre partition sera jouée et comment, de manière à ne pas transformer ce bel ensemble en cacophonie innommable ou pire encore, le faire s'arrêter de jouer ! La métaphore de l'orchestre est bien adaptée, car si vous avez l'intention d'intégrer un vuvuzela dans l'orchestre et de lui faire hurler une seule horrible note qui couvre les autres instruments, le chef d'orchestre fera son travail : prendre le fautif par le col et gentiment le passer par la sortie de secours la plus proche (tout en continuant à diriger l'orchestre bien sûr). Dans les faits, ceci est presque tout aussi brutal avec un arrêt pur et simple du programme fautif et un message explicite affiché à l'utilisateur. Avec un système de type UNIX (comme GNU/Linux), on dit même que le processus a été tué, si par exemple, il a provoqué une erreur de segmentation (*segmentation fault* en anglais ou *segfault*) en essayant d'accéder à la mémoire qui n'était pas la sienne.

4. LA RASPBERRY PI

Une carte comme la Raspberry Pi ou n'importe quelle autre plateforme de ce type (PCDuino, BeagleBone Black, ACME Arietta, etc.) est un ordinateur, tout comme votre PC ou votre Mac... ou presque. À l'instar de notre exemple de départ où nous avons ratatiné le contenu d'un Commodore 64 dans une

puce pour créer un microcontrôleur, il existe une différence notable entre votre PC et une Raspberry Pi : l'intégration des composants.

Avec un PC nous avons la même architecture que celle d'un vénérable Commodore C64, avec une petite différence notable de quelque 30 ans de progrès et d'évolutions technologiques. Mais le principe est le même, nous avons :

- le processeur,
- la mémoire vive, sous la forme de barrettes enfichées dans des emplacements,
- un contrôleur audio, généralement directement soudé sur la carte même, mais qui peut également prendre la forme d'une carte (ce qui était la norme il y a quelques années, mais se limite maintenant aux cartes audio professionnelles),
- un contrôleur Sata pour les disques et les lecteurs de supports optiques,
- un contrôleur graphique soit intégré à la carte mère soit, plus souvent, prenant place dans un emplacement PCI Express,
- des ports USB qui permettent à l'utilisateur de connecter des périphériques externes (clavier, souris, imprimante, etc.).

Tous ces composants dialoguent par l'intermédiaire de pistes (des bus) sur la carte mère et de puces dédiées à la circulation de l'information,

Une Raspberry Pi modèle B+. Le composant central est le SoC sur lequel est soudée la mémoire. Ce type de carte se rapproche davantage d'un ordinateur standard en permettant l'utilisation d'un système d'exploitation.





Comparaison d'une carte Arduino Leonardo et son microcontrôleur Atmel ATmega32u avec une carte ACME Arietta G25 et son SoC Atmel AT91SAM9G25. Visuellement, les concepts peuvent sembler proches, mais l'une ne fait qu'exécuter un programme unique alors que la seconde est destinée à l'utilisation du système d'exploitation GNU/Linux.

le tout bien entendu, sous le contrôle du processeur. La technologie a évolué, la vitesse a fantastiquement augmenté, les bus véhiculent davantage de données... mais le principe lui, reste le même.

Reproduisons donc notre opération de miniaturisation et compressons tout cela dans une puce unique : et voici notre SoC. Pourquoi SoC et non-microcontrôleur ? Tout simplement parce que l'usage est bien différent. Nous avons là un ensemble plus générique, destiné à faire fonctionner un système d'exploitation et non un programme unique.

Observez maintenant une Raspberry Pi et en particulier la puce centrale carrée : c'est le SoC Broadcom BCM2835 (plus exactement c'est la mémoire soudée sur le SoC qui lui est dessous et donc invisible). Dans ce SoC vous avez :

- un processeur ARM 11 (ARM1176JZF5),
- un contrôleur graphique VideoCore fournissant la sortie HDMI,
- un contrôleur audio PCM/I2S,
- différents ports et périphériques série (SPI, i2c) et USB.

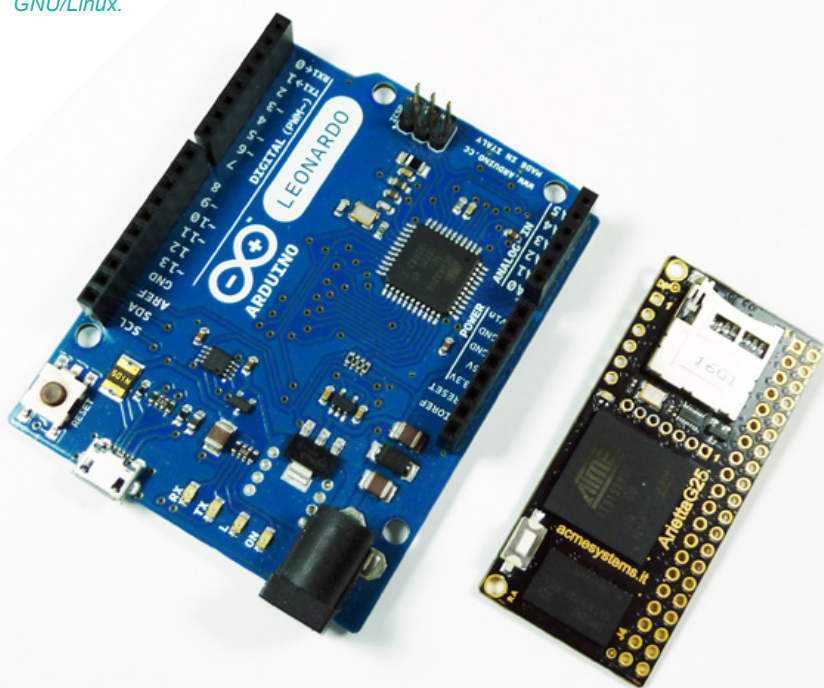
La mémoire n'est pas dans la puce Broadcom, mais soudée par dessus. C'est donc un composant à part tout comme le support pour stocker les données et les programmes qui prend la forme d'une carte SD (ou microSD avec les nouveaux modèles de Raspberry Pi).

5. OUI, MAIS MA QUESTION ALORS ?

Vous l'avez compris, la différence entre microcontrôleur et SoC est assez vague finalement, mais c'est d'un point de vue de l'utilisation que les choses diffèrent. La Raspberry Pi est un nano-ordinateur destiné à faire fonctionner un système d'exploitation (GNU/Linux en l'occurrence) alors qu'une carte Arduino a pour objectif d'être une sorte d'automate exécutant un unique programme.

L'usage que vous allez faire de ces deux plateformes est donc différent et la manière d'aborder un problème ou une réalisation personnelle tout autant. Si votre objectif est de contrôler un moteur ou une led par exemple, la présence d'un système d'exploitation n'est de loin pas indispensable. Pire encore, elle peut être gênante. Vous ne pouvez pas programmer avec une Raspberry Pi comme si vous écriviez le seul programme présent. Vous devez vous plier aux exigences du système et prendre en compte le fait que d'autres programmes utilisent également des ressources.

Inversement, si vous comptez afficher des vidéos ou de simples images sur un écran LCD, mieux



faut reposer sur les facilités mises à disposition par un système d'exploitation comme une solution unifiée pour le graphisme. Dans le précédent numéro par exemple, nous avons vu comment ajouter un petit écran LCD à la Raspberry Pi. Comme celui-ci était pris en charge par le système d'exploitation, un programme souhaitant afficher sur ce type d'écran fera exactement la même chose que s'il souhaitait afficher sur l'écran branché en HDMI. Pour l'application ou le programme que vous créez, cela ne fait aucune différence si ce n'est la taille de l'image qu'il est possible d'afficher.

Il en va de même pour le stockage des données. Avec une Raspberry Pi, votre programme dira « je veux un fichier ici » et le système, après vérification, va le créer que vous ayez désigné un chemin vers la carte SD ou vers un support USB. La manière de créer le fichier et d'écrire les données n'est pas votre problème, mais celui du système d'exploitation. Avec un croquis Arduino en revanche la notion de répertoire ou de fichier n'existe pas. Vous pouvez l'ajouter en utilisant une bibliothèque, mais du point de vue du programme final lui-même, c'est bel et bien lui qui devra faire tout le travail.

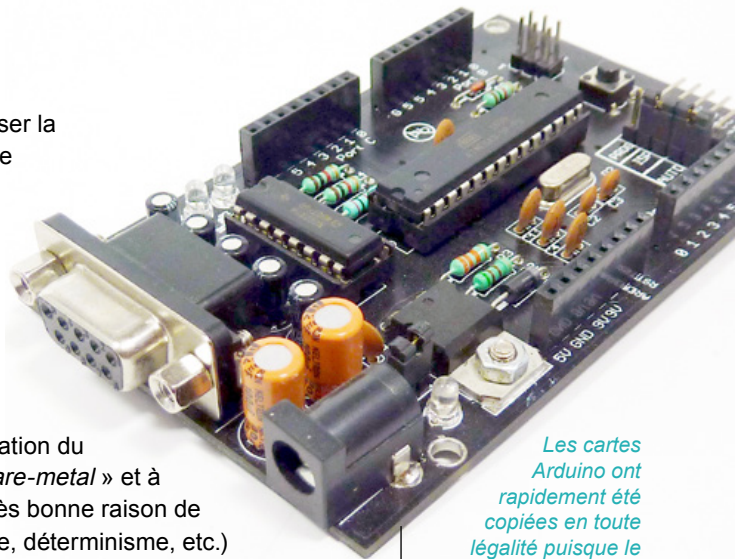
Enfin, lorsqu'un programme se termine sur une Raspberry Pi, l'ensemble de la carte ne cesse pas de fonctionner. Le programme quitte la mémoire et les ressources qu'il utilisait sont libérées. Le système d'exploitation fait le ménage et vous donne l'opportunité de le lancer à nouveau sans redémarrer tout le système.

Pouvez-vous utiliser la Raspberry Pi comme une carte Arduino et tout faire vous-même ? Oui, c'est possible. Est-ce bien raisonnable ? Non, généralement pas. On appelle ce genre de programmation du « développement *bare-metal* » et à moins d'avoir une très bonne raison de le faire (performance, déterminisme, etc.) on préférera reposer sur quelque chose de plus orthodoxe et donc un système d'exploitation.

Peut-on avoir un système d'exploitation et du multitâche sur un microcontrôleur comme ceux des cartes Arduino ? Là encore la réponse est, dans l'absolu oui, mais là encore, il faut une très bonne raison de vouloir le faire, car les ressources (la mémoire en particulier) sont très limitées et le processeur ne dispose pas de certaines fonctionnalités permettant de faciliter ce fonctionnement. Les processeurs récents (depuis le 80286) possèdent une MMU pour *Memory Management Unit*, permettant l'utilisation d'un mode protégé découpant la mémoire et gérant les permissions des différents programmes. Sans cette fonctionnalité, le système d'exploitation doit faire cela logiciellement, ce qui consomme des ressources. Dans le cas contraire, les programmes peuvent « se marcher dessus », système d'exploitation ou pas.

Au final, les cartes Arduino sont des « automates programmables » et les Raspberry Pi des nano-ordinateurs. La façon de s'en servir et de les programmer est très différente et vous choisirez l'un ou l'autre en fonction du projet que vous comptez débiter.

Si votre démarche est plus générique et que vous cherchez avant tout à explorer ce domaine en suivant une approche didactique, la réponse est encore plus simple : les deux mon bon monsieur ! L'arrivée récente de la Raspberry Pi A+ vous permet d'avoir accès, pour un budget total de quelques 50€, aux deux plateformes et donc de découvrir simultanément les deux facettes de cette petite révolution qui pourrait bien changer la manière dont on vit la technologie moderne. **DB**



Les cartes Arduino ont rapidement été copiées en toute légalité puisque le matériel est « ouvert ». Ici, une carte d'origine chinoise comprenant le même microcontrôleur que la Uno originale, mais entourée de composants plus économiques, plus faciles à trouver et assemblés avec des méthodes moins coûteuses (à l'époque).

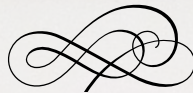


A V
0.26 0.25

ALIMENTATION DE LABORATOIRE

Denis Bodor

CURRENT



VOLTAGE

Alimenter un montage, une carte ou un nano-ordinateur passe généralement par l'utilisation d'un bloc d'alimentation tel qu'il en existe pour les smartphones ou d'autres équipements électroniques destinés aux utilisateurs finaux. Cependant, lorsqu'on décide d'expérimenter, il devient rapidement nécessaire d'avoir accès à une source d'alimentation plus contrôlable aussi bien en tension qu'en courant.

OUTPUT

ON/OFF

PAR.

SLAVE

GN

TRACK

POWER

ON

OFF

Le terme « alimentation » couvre un large panel d'équipements et de composants. Il définit généralement un « machin » qui alimente quelque chose en courant (on entend même malheureusement le terme « chargeur » maladroitement utilisé à la place). Votre PC, par exemple, contient une alimentation sous la forme d'un bloc distinct connecté aux 230V (secteur) et transformant le courant et les tensions de manière à fournir toute une gamme de sorties alimentant la carte mère et les composants de la machine (disques durs, lecteurs, etc.). Un peu plus évolué qu'un simple bloc d'alimentation de smartphone fournissant généralement un courant continu de quelques centaines d'ampères en 5 volts, une telle alimentation de PC propose +5V, -5V, +12V, -12V et +3,3V avec diverses limites de courant (souvent entre 30 A et 500 mA).

L'alimentation de PC est « un cran au-dessus » du simple bloc d'alimentation tant en termes de tension, de courant, de puissance et de technicité (refroidissement, efficacité, etc.), mais ne vous permet pas d'ajuster à votre convenance à la fois le courant délivré et la tension. Tout est fixé à l'usine et conçu pour être adapté à une génération précise de matériel PC (carte mère et périphérique) tant en termes de connectique que de caractéristiques.

1. COMPOSITION ET INTERFACE

Ce qu'on nomme « alimentation stabilisée de laboratoire » est un équipement permettant à l'utilisateur de choisir les



caractéristiques du courant délivré par le périphérique. Un aspect « contrôle » est donc associé ainsi qu'une interface spécifique généralement la plus ergonomique et logique possible.

Un tel produit peut prendre plusieurs formes, en particulier en ce qui concerne le contrôle du courant et de la tension en sortie. Les premières générations d'alimentations étaient purement analogiques. Un ensemble de circuits et de techniques permettait d'ajuster ces deux éléments à l'aide de potentiomètres en façade et de contrôler leurs valeurs à l'aide d'indicateurs comme des voltmètres analogiques ou des galvanomètres (ampèremètre de type analogique). Ceci prenait la forme très distinctive de cadrans à aiguille utilisant une bobine et des aimants.

Les générations actuelles d'alimentations de laboratoire sont numériques et ceci concerne tout aussi bien le contrôle du courant et de la tension en sortie que l'indication de ces valeurs. C'est une logique embarquée, sous forme de microcontrôleur, qui impacte la transformation du courant et limite la sortie. Sur certains modèles, l'affichage des deux valeurs se fait via des leds sous forme d'afficheurs 7 segments. D'autres utilisent un écran LCD réflectif ou transreflectif numérique ou alphanumérique et enfin les modèles les plus récents et avancés reposent sur des écrans LCD ou oled ainsi que sur des interfaces graphiques utilisateur, tantôt multicolores, très complètes.

La base du contrôle cependant reste toujours la même et se résume généralement à des connecteurs type banane et des potentiomètres : un pour régler le courant et l'autre pour la tension. Ceci se complète

Un exemple typique d'alimentation de laboratoire analogique totalement dénuée d'électronique numérique (microcontrôleur). Un peu vieille, pas très chère d'occasion (prix < port), mais nécessitant souvent une calibration.



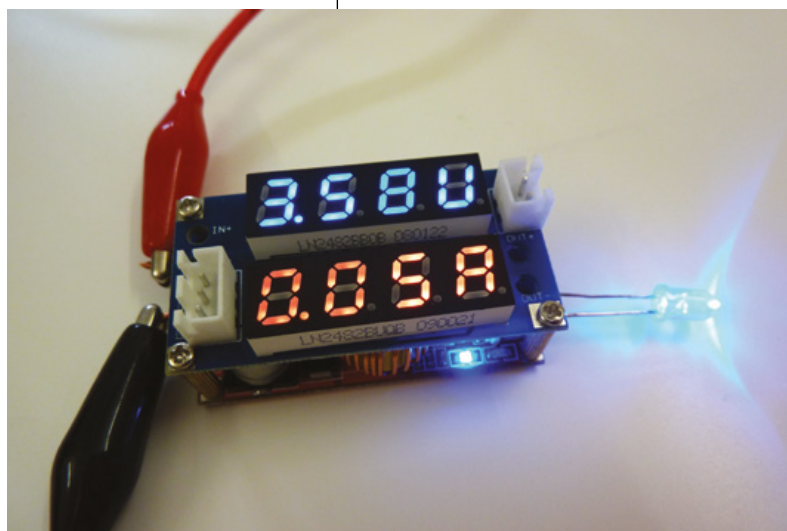
Une alimentation Protek d'entrée de gamme proposant deux sorties contrôlables en courant et en tension ainsi qu'une troisième, non réglable, fournissant 5V/3A.

souvent d'un ensemble de boutons permettant la mise sous tension, bien entendu, mais également l'activation de la ou des sorties.

2. SOLUTION MAISON

La création d'une alimentation est, dans le domaine de l'électronique, un classique presque incontournable. Je parle ici de la conception et de la construction de A à Z avec chaque « étage », de l'arrivée en 230V alternatif en passant par les sorties contrôlées, en passant par le transformateur, le redressement (alternatif vers continu),

Un petit module chinois composé d'un convertisseur DC/DC couplé à un voltmètre et un ampèremètre. Utilisé avec un bloc d'alimentation de PC portable par exemple et après quelques modifications, ceci peut être la base d'une alimentation faite maison.



la conversion de tension, la limitation de courant, les protections, etc. La conception de ce genre d'équipement nécessite de bonnes bases en électronique et les choix techniques impacteront directement les capacités et les fonctionnalités du projet final.

Une autre solution, si l'objectif est d'arriver à une alimentation simplement utilisable, consiste à baser son projet sur un élément déjà existant comme une alimentation de PC. Avec +/-12V et +/-5V et du courant à profusion (sauf en -12V qui n'est généralement et historiquement utile que pour les ports série RS232), on dispose déjà d'une alimentation efficace et très stable. Sur cette base, il suffira de trouver une solution pour décliner diverses tensions et limiter le courant (si l'on souhaite cette fonctionnalité). Pour cet ajout, il sera également possible d'utiliser un module « tout fait ». Pour ceux d'entre vous qui ont lu l'article sur l'alimentation d'une led IR de puissance dans le précédent numéro, ce type de modules se résume à un convertisseur DC/DC ajustable complété d'un afficheur numérique miniature à leds et deux micro-potentiomètres. Ce genre de modules se trouve en provenance directe de Hong Kong ou Shenzhen pour moins de 10 euros (port inclus) et via quelques modifications et un peu de bricolage pour créer un panneau pratique à utiliser, il est parfaitement possible d'obtenir une alimentation économique et fiable.

Bien entendu, ce type de créations, complètes ou sur une base existante relève plus du projet plus que de l'équipement directement et immédiatement utilisable. Si vous souhaitez un périphérique ne nécessitant aucun ajustement ou réglage, la solution est d'opter, bien sûr, pour un produit tout fait, calibré et testé.



L'alimentation ATX des PC peut être une base pour construire une alimentation de laboratoire. On retrouve généralement sur le périphérique lui-même toutes les indications techniques utiles ainsi que la nomenclature des sorties.

3. POUR QUELS USAGES ?

Bien entendu l'intérêt premier d'une alimentation de laboratoire est de disposer d'une source de courant stable et contrôlable. Peut-être qu'à ce stade de la lecture de cet article (ou même avant) vous vous demandez pourquoi donc influencer sur le courant fourni alors qu'une alimentation de PC ou un simple bloc d'alimentation de smartphone affiche un simple courant maximum.

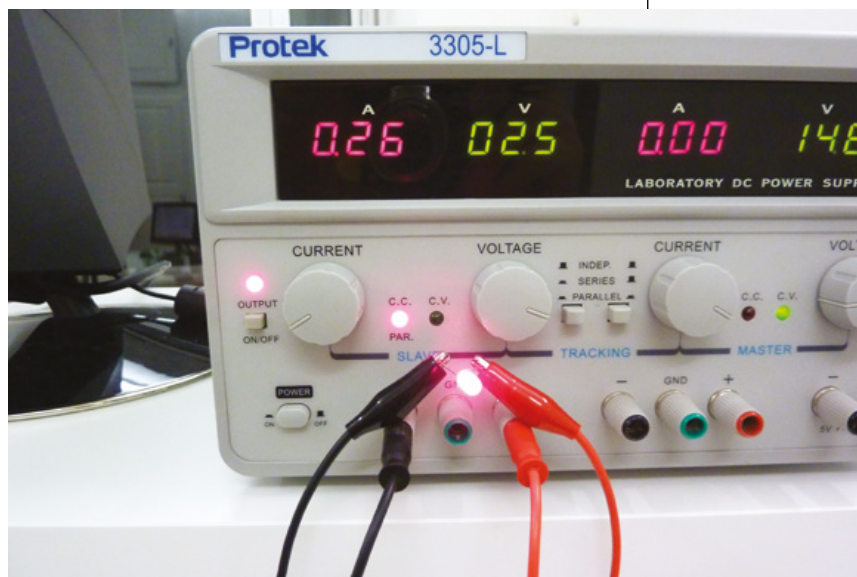
En effet, si nous prenons l'exemple du bloc d'alimentation, celui-ci est généralement libellé d'une tension en sortie, disons 5 volts, et d'un courant en ampères qui est le courant **maximum** que le bloc peut délivrer. Ceci ne signifie en rien que le smartphone que vous y branchez va effectivement utiliser ce courant. L'indication est à comprendre ainsi : « je peux fournir une tension de 5V pour un courant circulant jusqu'à 1A ». Ce qui se passera en tentant de « tirer » davantage de courant dépend du bloc et de sa conception, mais généralement ce qui se passe est un effondrement pur et simple de la tension. Dans

de nombreux cas, avec des blocs de bonne qualité un circuit de protection va détecter que le courant est supérieur à celui donné dans les spécifications et tout simplement mettre l'alimentation en défaut. Avec

un produit chinois super bas de gamme en revanche, généralement gratifié d'un logo « CE » parfaitement illégitime, le résultat risquera d'être plutôt visuel, sonore et olfactif (selon le contexte de l'odeur de plastique brûlé à la sirène des pompiers en passant par le disjoncteur qui saute). Une protection est utile en cas de court-circuit puisqu'à ce moment une très faible résistance existe et donc un maximum de courant passe.

Mais revenons à nos moutons, si c'est le circuit ou le périphérique qui détermine finalement le courant qu'il utilise pour fonctionner, pourquoi donc le limiter arbitrairement ? Souvenez-vous de la loi d'Ohm et de l'article consacré dans le premier numéro du magazine. Si nous prenons l'exemple d'une led, typiquement celle-ci se commande (ou se pilote) en courant. En d'autres

Une led de forte luminosité connectée et alimentée directement par l'appareil. Remarquez le voyant « CC » allumé. Nous avons ici un mode de fonctionnement en courant constant limité à 260 mA. Les 2,5V affichés correspondent à la tension apparaissant aux bornes de la led.



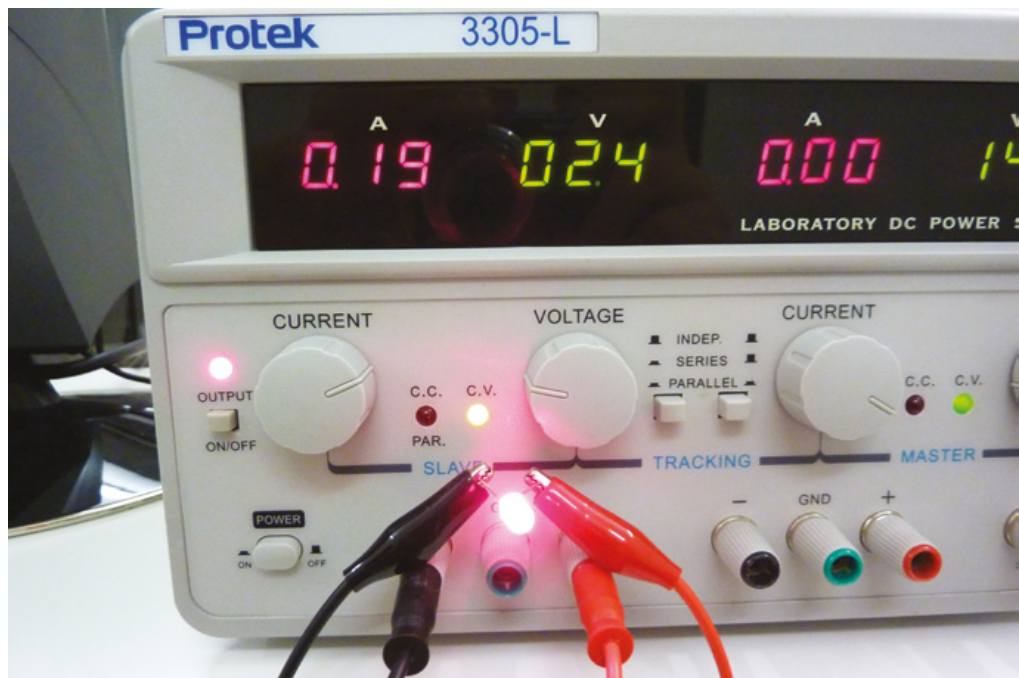


termes, le fabricant détermine un courant optimal à fournir à la led pour qu'elle fonctionne le plus efficacement possible. Résultat de l'application de cette valeur, une tension apparaît aux bornes de la led. Cette tension n'étant pas celle de l'alimentation, on utilise une résistance pour dissiper l'énergie excédentaire en calculant sa valeur grâce à la loi d'Ohm.

Une alimentation de laboratoire fonctionnera différemment étant donné qu'il est possible de contrôler précisément la limite du courant en sortie. Ainsi en connectant une led et en réglant l'appareil sur la valeur de courant indiquée par le fabricant, il n'y a que cette quantité qui va parcourir le circuit et une tension va effectivement apparaître aux bornes de la led. Tension qui s'affichera alors sur l'alimentation. Certains matériels indiquent également si l'alimentation fonctionne en mode « courant constant » (*constant current*) ou en « tension constante » (*constant voltage*).

En limitant le courant en dessous de ce que peut effectivement (dans l'absolu) accepter la led, l'alimentation est en mode courant constant et vous aurez beau tourner le potentiomètre de réglage de la tension, celle-ci ne changera pas.

Inversement, en décidant de régler le potentiomètre du courant au maximum, l'alimentation sera en mode tension constante (ampère à profusion et contrôle de la tension). Dans ce mode, le



courant affiché par le périphérique est celui parcourant le circuit et limité par le montage ainsi alimenté.

Bien entendu, le fait de contrôler tension et courant permet ainsi d'ajuster l'alimentation de tout ce qui est imaginable, bien au-delà des leds. Avec une alimentation de laboratoire, il est ainsi possible de procéder à la charge d'un accu. Il faut cependant une interaction humaine puisque les valeurs choisies et lues déterminent l'étape de la charge dans laquelle se trouve l'accu.

Vous l'aurez compris, l'alimentation de laboratoire agit donc également comme un multimètre dans certaines situations. En branchant une Raspberry Pi ou un Arduino à un tel appareil et en ajustant la tension sur 5V, la valeur du courant qui s'affiche

La même led de forte puissance sur la même alimentation, mais cette fois en mode tension constante (voyant « CV »). Ici, on impose une tension et on constate qu'un courant de 190 mA circule dans la led.

est alors celui utilisé par la carte. Vous pouvez donc calculer la consommation du montage en watts (certaines alimentations font même le calcul à votre place).

Enfin, et c'est tout de même l'objet initial d'une alimentation, ce type de produits vous permet avec une grande fiabilité et précision de fournir une alimentation à un montage exotique. Hé oui, il n'y a pas que 5V et 3,3V dans la vie. Remplacer une batterie LiPo typique consistera à fournir 3,7V (1S), 7,4V (2S), 11,1V (3S), 14,8V (4S), etc.

Vous pourrez alors non seulement remplacer une alimentation via accu par une source moins coûteuse à l'usage, mais également facilement tester la détection de la charge de la batterie par le circuit que vous alimentez. Un exemple récent qui me vient à l'esprit concerne le réglage des tensions d'avertissement et d'alerte dans un contrôleur de vol Naza Lite pour un multi-rotor DJI Flame Wheel F550 (kit ARF). En alimentant le F550 à l'aide d'une alimentation de laboratoire, on peut non seulement configurer le contrôleur sans avoir à charger un LiPo mais, en plus, on peut tester la détection des niveaux de tension et calibrer l'ensemble sans jongler avec un multimètre.

4. ALIMENTATION TOUTE FAITE : ÉLÉMENTS CLÉS

Une alimentation de laboratoire répond à un certain nombre de besoins et de critères. La plus importante à mon sens concerne les plages de tension et de courant utilisables. Pour un usage courant en électronique « généraliste », une tension réglable de 0 à 30 V est souvent suffisante. Le courant quant à lui, à hauteur de 3A est amplement adapté dans la plupart des situations. Le réglage de ces valeurs est également important. Un contrôle à 0,01 volts/ampère suffira, mais certaines alimentations peuvent proposer (parfois en option) un ajustement au millivolt et milliampère.

Le nombre de sorties ajustables est également une caractéristique importante. Beaucoup

d'alimentations peu coûteuses ne proposent qu'une seule sortie. Dans ce genre de situation, une limitation importante est à prendre en compte qui, de but en blanc, n'est pas évidente. Avec deux sorties « flottantes » (il n'y a pas de liaison à la terre en sortie), il est possible de connecter deux sorties en série (le négatif (noir) d'une sortie sur le positif (rouge) de l'autre et à la masse (connecteur GND vert)) et vous aurez alors une tension positive et une autre négative relativement à la masse. Ceci peut s'avérer très pratique dans certaines situations avec des montages particuliers ou pour alimenter, par exemple des anciens ordinateurs.

Viennent ensuite les considérations sur l'interface de l'équipement. Il est de bon ton de choisir une alimentation proposant un interrupteur dédié pour activer chaque sortie indépendamment de la mise sous tension générale.

Les connecteurs sur la totalité des sorties (ou la majorité) doivent proposer un connecteur de

L'alimentation Rigol DP832 : pas chère, programmable, ergonomique et très complète. Un produit très satisfaisant provenant d'un constructeur chinois qui redouble d'efforts depuis quelques années pour se faire une place et se tailler une réputation.





masse spécifique. Comme précisé précédemment, ceci permet de relier les sorties entre elles (voire plusieurs alimentations entre elles). Mais la qualité de ces connecteurs est également importante et ils doivent proposer une connexion de type fiche « banane » et une borne de connexion à visser. Les alimentations de qualité sont généralement livrées avec un ensemble de câbles à gaine silicone.

Le choix d'une alimentation se fera exactement comme celle de n'importe quel appareil de mesure (cf. article sur les multimètres dans un numéro précédent). Il s'agit d'un équipement destiné à une utilisation longue et fiable sur la durée. Cependant, contrairement au monde des multimètres, on assiste depuis quelque temps déjà à un mouvement de fond notable.

Le constructeur chinois Rigol, très connu et populaire depuis la commercialisation de son oscilloscope économique DS1052E et plus récemment du DS1054Z, commercialise une alimentation stabilisée programmable de laboratoire, modèle DP832, plébiscitée par une vaste majorité d'utilisateurs professionnels comme hobbyistes. En effet, le rapport fonctionnalité/prix offert par ce modèle est excellent, même si les quelques 365€ de budget nécessaires à son acquisition restent un investissement important pour un particulier.

L'interface utilisateur des alimentations de laboratoire modernes est généralement très complète tout en essayant de se calquer sur des standards issus des précédentes générations.

J'ai, comme bien d'autres, décidé de remplacer mon alimentation analogique vieillissante (devenue utilisable qu'avec l'assistance d'un multimètre) par ce modèle, non sans avoir auparavant étudié et analysé les critiques qu'on peut trouver (en anglais) sur le web (dont les excellentes vidéos de Dave Jones d'EEVBlog). Le Rigol DP832 n'est sans doute pas le seul modèle intéressant, mais il faut avouer que la volonté affichée par le constructeur pour « se faire un nom » fait que la communication avec la communauté des utilisateurs est exemplaire.

Le DP832 dans sa première génération comportait un problème de conception au niveau du refroidissement (radiateur trop petit et mal placé sur des régulateurs). Le constructeur, à l'opposé d'autres œuvrant dans le domaine de la conversion USB/série (suivez mon regard), a réagit rapidement et efficacement en annonçant prendre en compte le problème et travailler sur sa correction. Quelques semaines après, une mise à jour du modèle était disponible (celle actuellement commercialisée) et les utilisateurs étaient invités à contacter leur distributeur pour échange ou mise à jour.

À propos de distributeur justement, alors que généralement l'acquisition de ce type de matériel « tendance » de provenance lointaine s'avère être relativement difficile dans l'hexagone, les produits de marque Rigol disposent d'un distributeur agréé et officiel... et surtout accessible aux particuliers ! C'est la société Ovio Instrument (<http://ovio-scope.com/>) qui





Façade du DP832. On y retrouve deux sorties 0-30V et 0/3A, plus une 0-5V/0-3A ainsi qu'un ensemble de contrôles plus ou moins ergonomiques. L'écran présente ici une déclinaison « fun » de l'interface accessible par un bouton dédié. Quelqu'un se sert-il réellement de ce mode d'affichage absolument anti-ergonomique ?

distribue ce produit (ainsi que d'autres de diverses marques et modèles) et chez qui j'ai procédé à mon achat.

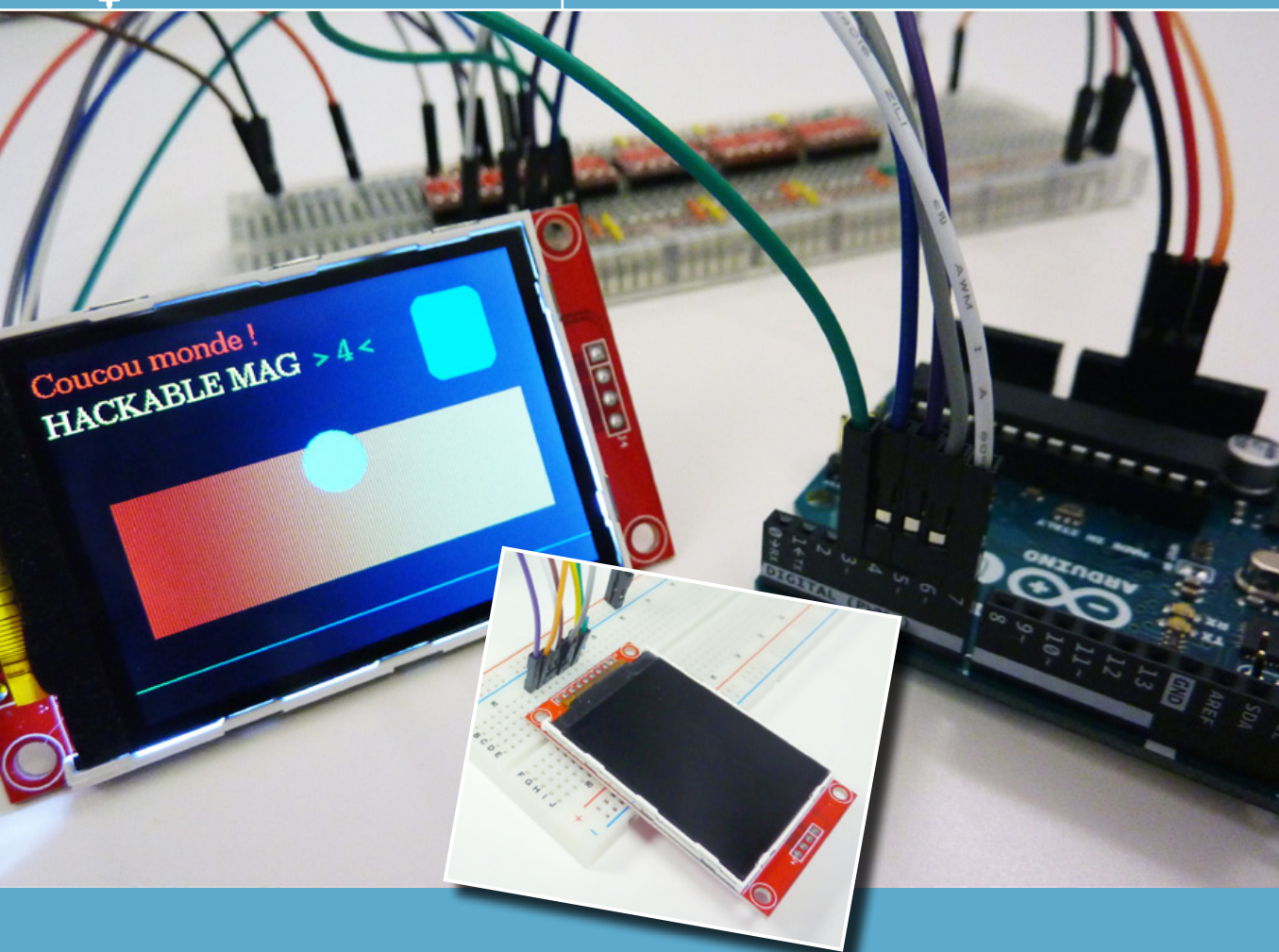
L'alimentation est programmable et dispose de fonctionnalités avancées intéressantes (en plus des fonctions standards indispensables) :

- écran 3,5 pouces,
- diverses fonctions de sécurité (température, ventilation, dépassement de courant, court-circuit, etc.),
- régulation linéaire à faible bruit,
- cycles d'alimentation programmables,
- tracé de graphes tension/courant,
- un adorable bouton de coupure de toutes les sorties en plus d'un bouton par sortie,
- interface série et USB en standard (Ethernet et GPIO en option),

- une interface utilisateur très ergonomique malgré quelques choix perturbants (interface façon cadran automobile avec un bouton dédié),
- conservation des réglages entre deux mises sous tension,
- résolution 1mV/1mA en option (environ 80 euros),
- design « rackable »,
- une documentation complète, mais en anglais.

En résumé bien que, comme doit le montrer le contenu de cet article, je sois extrêmement satisfait de mon achat, je me dois de rappeler que Rigol n'est bien entendu pas la seule marque de qualité. On pourrait citer, par exemple, B&K précision ou Agilent proposant des modèles d'entrée de gamme dans des budgets similaires (moins de 400 euros).

Rigol est une marque « qui monte » et gagne ses lettres de noblesse par une excellente communication et des réactions rapides. On ne peut qu'espérer que la révolution (ou le chamboulement) engagée dans le monde des oscilloscopes (que nous traiterons bientôt) et maintenant des alimentations programmables, en partie grâce à Rigol, se propage à d'autres types d'équipements que les amateurs d'électronique se doivent de posséder... **DB**



UTILISEZ UN ÉCRAN LCD À 5€ AVEC VOTRE ARDUINO

Denis Bodor

Dans le précédent numéro, nous avons vu comment ajouter un écran LCD économique en provenance directe de Shenzhen avec une carte Raspberry Pi. Il est temps à présent de réutiliser ce même module avec une carte Arduino afin de fournir également une sympathique capacité d'affichage à la petite carte italienne !



NIVEAU



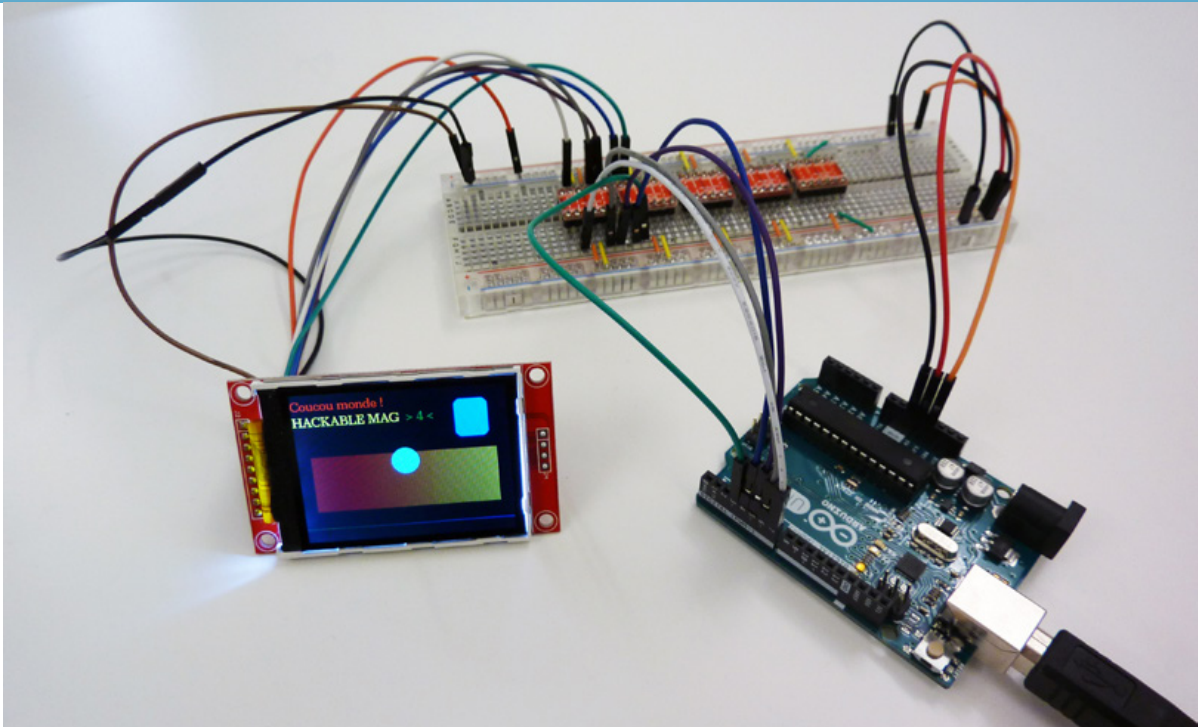
TEMPS

15
minutes



BUDGET

environ 30 €
(Arduino inclus)



Lorsqu'on parle d'afficheur LCD connecté à une carte de type Arduino, on pense généralement à quelque chose d'assez basique comme un afficheur deux fois 16 caractères « type HD44780 ». Monochrome, alphanumérique et tantôt rétroéclairé, ce type de module se connecte avec une interface 4 ou 8 lignes de données directement contrôlées via les sorties de l'Arduino. Le résultat peut être suffisant pour fournir des informations de base à l'utilisateur, mais on arrive vite aux limites même en optant pour des afficheurs similaires de 4×20 caractères. Bref, pour le dire franchement, c'est bien sympathique, mais on est relativement rapidement à l'étroit.

Dans le précédent numéro de *Hackable*, nous avons fait connaissance avec un autre type d'affichage qui généralement est bien plus coûteux : un écran LCF TFT graphique d'une résolution de 320×240 pixels. Avec la généralisation des périphériques utilisant ce

type d'afficheur comme les smartphones super bas de gamme, les baladeurs MP3, les imprimantes, ou n'importe quel appareil voulant afficher des graphismes modernes un rien colorés, les prix en sortie d'usine sont tombés à des niveaux impressionnants. Or ces usines, généralement se trouvent en Asie où, comble du hasard, résident également une tripotée de vendeurs de modules et de composants électroniques.

Si l'on y pense rien qu'un instant, la chute des prix est parfaitement compréhensible. Souvenez-vous, il y a quelques années, la « norme » en termes d'affichage, disons sur des copieurs ou des imprimantes laser par exemple, était l'afficheur LCD vert/gris présentant un texte succinct. Aujourd'hui, on a plutôt droit à un mini écran couleur, version édulcorée des interfaces de smartphone. Mettez cela à l'échelle sur le plan « gamme de produits » et immédiatement vous comprendrez que l'utilisation et donc la production d'écrans TFT LCD de faible résolution ont dû largement être optimisées

en termes de volumes et de coûts de production.

Le résultat est là, un écran LCD TFT de 2,2 pouces de diagonale géré par un contrôleur ILI9340C avec une résolution de 320 pixels par 240 (souvenez-vous un jour, dans le passé, les écrans CGA de PC utilisaient une résolution encore inférieure) en 65536 couleurs ou en 262144 couleurs. Le tout pour moins de 5 euros (4,24 euros port gratuit auprès d'un vendeur appelé *Czb Electronic* sur eBay. Non ce n'est pas un ami et je n'ai pas d'actions. C'est juste le vendeur que j'ai choisi d'après ses notes pour mes modules).

Nous pouvons connecter ce type d'afficheur sur une carte Arduino exactement comme nous l'avons fait avec la Raspberry Pi. La différence tient principalement en la façon de l'utiliser. En effet, alors que la Raspberry Pi fait fonctionner un système d'exploitation et donc permet l'utilisation de l'écran via l'installation d'un pilote, avec une Arduino, on contrôle l'écran directement (ou plus exactement via une bibliothèque dédiée).



LE PRINCIPE

L'écran dont nous parlons ici et qui repose sur une puce ILI9340C se connecte via un ensemble de lignes normalisées appelé bus SPI, plus quelques autres connecteurs pour l'alimentation, le rétro-éclairage et d'autres signaux propres à l'écran. SPI pour *Serial Peripheral Interface* est architecturée/standardisée ainsi :

- SCLK ou CLK pour *clock* (horloge) est le signal qui cadence la communication,
- MOSI pour *Master Out Slave In* ou SDI pour *Serial Data In* véhicule les données du maître vers l'esclave,
- MISO pour *Master In Slave Out* ou SDO pour *Serial Data Out* transporte les informations dans le sens opposé,
- SS ou CS pour *Slave Select* ou *Chip Select* permet à celui qui contrôle la communication (le maître) d'adresser l'esclave (lui demander d'être attentif). SPI permet d'avoir des lignes SCLK, MISO et MOSI communes pour plusieurs périphériques (esclaves) et différentes lignes CS les « activent » pour qu'ils « se sentent concernés ». Ceci évite d'utiliser une salade de câbles pour brancher plusieurs composants en SPI. Il suffit d'un bus et d'autant de lignes CS que d'esclaves.

L'écran dispose de tous ces signaux, plus d'autres, mis à disposition sous la forme d'une série de connecteurs avec la nomenclature suivante :

- SDO(MISO) : le MISO dont nous venons de parler,
- LED : la broche pour alimentation des leds de rétroéclairage (+3,3V),
- SCK : le SCK du bus SPI (horloge),
- SDI(MOSI) : le MOSI,
- D/C : une ligne qui permet au module écran de savoir s'il reçoit des données (D) ou des commandes (C),
- RESET : comme le nom l'indique, permet un reset du module,
- CS : le CS du bus SPI,
- GND : la masse,
- VCC : la tension d'alimentation de 3,3V.

Vous avez peut-être déjà deviné le problème qui va se poser à nous dans cette réalisation : le module fonctionne en 3,3 volts et ne tolère pas de tension plus élevée sur ses broches. La plupart des cartes Arduino (Uno, Leonardo, Duemilanove, Mega, etc.) fonctionnent en 5 volts et présentent donc des sorties en 0-5V qui peuvent endommager le module écran en cas de connexion directe. Le problème ne se pose pas avec une Raspberry Pi qui, elle, dispose de sorties en 3,3 volts.

Notre tâche ne consistera pas donc simplement à interconnecter Arduino et afficheur, mais nous devons nous arranger pour convertir les tensions au passage.

CE QU'IL VOUS FAUT

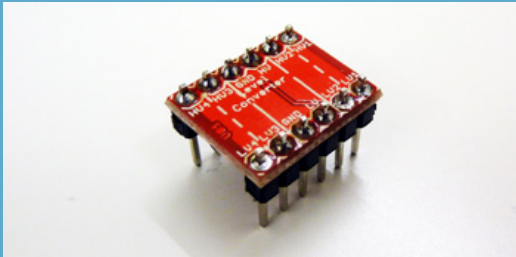


- Une carte Arduino : La bibliothèque que nous allons utiliser et la façon dont nous allons le faire nous permet de choisir n'importe quelle carte Arduino. Si cette carte est l'Arduino Due, drastiquement différente des « classiques » Uno ou Leonardo, vous n'aurez pas besoin de convertir les tensions. La Due fonctionne de toute façon en 3,3V ainsi que certains clones Arduino. Ce sera également le cas si vous optez pour une autre carte comme une Launchpad MSP430 de Texas Instruments et son environnement « compatible » Arduino appelé Energia.



- Un module écran LCD TFT provenant de la source que nous avons utilisée ou d'une autre. Des afficheurs similaires existent, comme ceux distribués par AdaFruit ou d'autres détaillants (mais pas à 5€). La bibliothèque

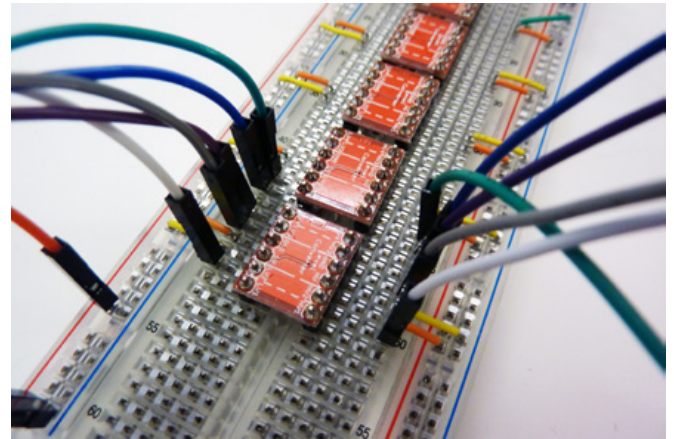
que nous allons utiliser ne supporte cependant que les afficheurs reposant sur les contrôleurs ST7735, ILI9341/ILI9340, PCF8833 et SSD1351. Il faudra donc vous assurer avant achat que le produit utilise bel et bien l'un de ces composants.



- Un ou plusieurs modules (*breakout board*) permettant la conversion de tensions. Il existe plein de solutions pour cela, y compris celles « faites maison » sur platine à essais. Nous avons cependant opté ici pour une solution toute faite et, en même temps réutilisable. Le type module en question repose sur un schéma relativement simple, que nous détaillerons plus loin dans l'article, à base de MOSFET N (BSS138), tel que distribué par Sparkfun sous le nom « *SparkFun Logic Level Converter – Bi-Directional* ». Le module permet une conversion d'une tension vers une autre de façon bidirectionnelle et sera donc utilisable non seulement pour du SPI, mais aussi pour de l'i2c (et donc connecter une RTC DS1305 5V sur une RPi en 3,3V par exemple), et ce avec des tensions non seulement de 5V/3,3V, mais aussi 1,8V, 2,5V, etc. Ce n'est pas tout, le produit étant en open hardware, les schémas sont disponibles et réutilisables et, bien entendu, des clones à base de BSS138 et d'autres MOSFET sont disponibles sur eBay par exemple. Un achat chez Sparkfun vous reviendra à 2,50€ pièce plus le port, sachant qu'un seul module comporte 4 MOSFET et est en mesure de « traiter » 4 connexions. Pour l'écran, nous en utiliserons 5 (hé oui, pas de bol).

- Le matériel classique du bidouilleur : platine à essai et câbles de connexion.

LE MONTAGE



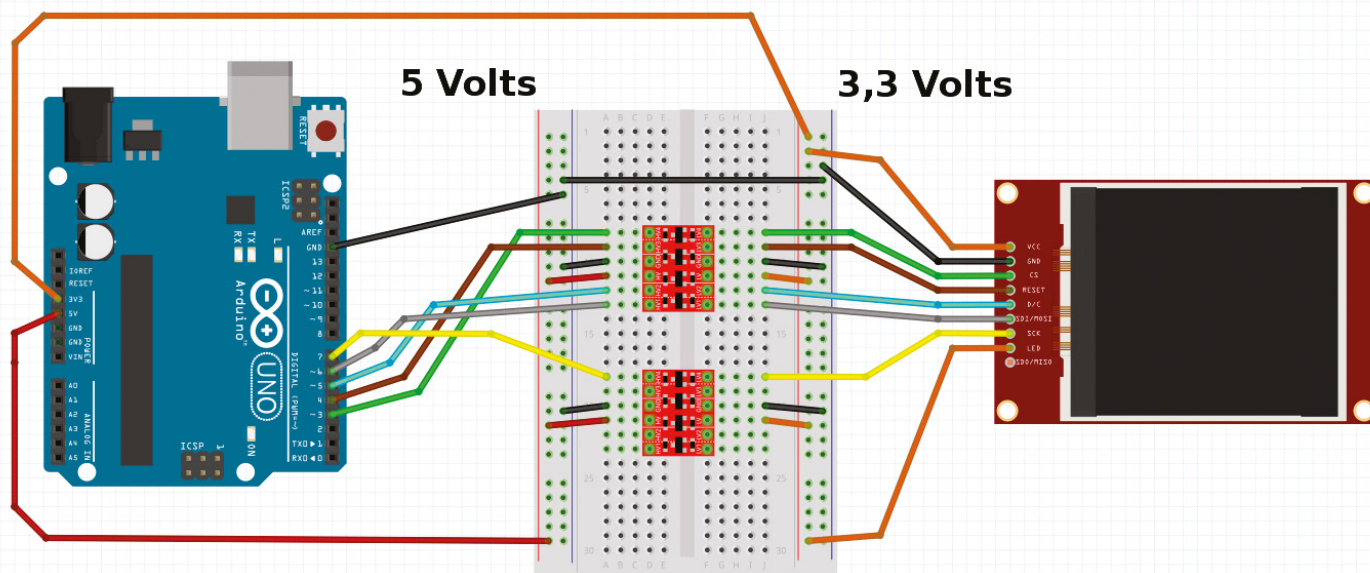
L'assemblage de l'ensemble se présente avec d'un côté le monde en 5 volts (à gauche) et l'Arduino, et de l'autre (à droite) le monde 3,3 volts avec le module d'affichage. Servant de traducteur entre les deux mondes, on trouve deux modules de conversion de niveaux composés d'un ensemble de quatre MOSFET N BSS138 chacun.

Ces modules sont alimentés à la fois en 5V désignés par HV pour *High Voltage* et en 3,3V pour le côté LV (*Low Voltage*). De la même manière, les broches d'entrée/sortie sont placées judicieusement avec d'un côté HV et d'un autre LV, ce qui est relativement intuitif.

Avant de toucher un mot sur le fonctionnement de cette conversion de tensions, détaillons l'interconnexion de l'ensemble. Sur l'Arduino, nous n'utilisons que des sorties (MISO n'est pas utilisé) tout à fait standard. Vous le verrez dans le croquis, la bibliothèque utilisée sait gérer plusieurs types d'afficheurs à la fois en reposant sur une connexion SPI matérielle ou entièrement logicielle. Cela reste du SPI, mais c'est la bibliothèque qui, à l'aide de codes, gère le protocole en contrôlant les broches. L'avantage de cette solution se résume en la possibilité de connecter l'écran où il nous chante et le contre-coût est une baisse de performance quant à la vitesse d'affichage.

L'interconnexion se résume ainsi :

- VCC sur la broche 3,3V de L'Arduino,
- GND à la masse,



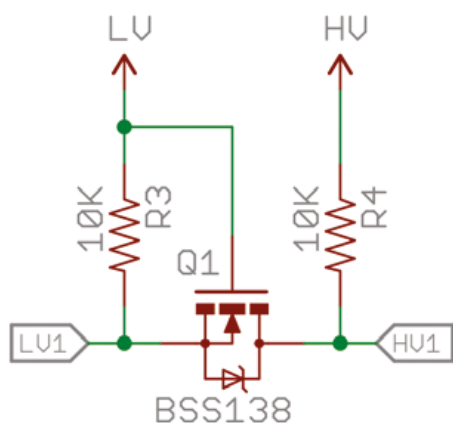
- CS sur la broche 3,
- RESET sur 4,
- D/C sur 5,
- MOSI sur 6,
- SCK sur 7,
- et enfin LED directement sur 3,3V (là, une amélioration est possible en utilisant un MOSFET comme un IRL520 ou 540 contrôlé par une sortie PWM et donc obtenir un pilotage du rétroéclairage par logiciel).

Intéressons-nous maintenant à la partie conversion avec, ci-contre le schéma correspondant à l'un des 4 BSS138 composant un module 4 voies bidirectionnelles.

Ce schéma est tiré de la documentation Sparkfun et est relativement facile à « suivre ». De plus, ceci ne sort pas de nulle part, mais fait l'objet d'une *application note* très connue de NXP (AN10441) à propos des conversions de tensions pour les bus i2c (qui sont bidirectionnels). LV et HV désignent respectivement, dans notre cas, 3,3V et 5V et le drain ainsi que la grille du MOSFET y sont reliés via des résistances de rappel (*pull up*). LV1 et HV1 désignent les connexions aux côtés 3,3V et 5V « du monde ». Voici ce qui se passe alors :

- Si aucun des deux côtés ne ramène une ligne à la masse, LV1 est amené à LV via la résistance de rappel. La grille et la source du MOSFET sont donc à 3,3V et ne présentent pas de différence de potentiel (V_{gs}). La tension de seuil n'est pas dépassée et le MOSFET ne conduit pas le courant. De ce fait, HV1 est aussi ramené à HV par sa résistance de rappel.
- Si LV1 est mis à la masse, une différence de potentiel existe alors entre la grille et la source (V_{gs}), celle-ci dépasse le seuil et le MOSFET devient conducteur. Ce faisant, HV1 est aussi ramené à la masse.
- Si HV1 est mis à la masse, le drain est ramené à la masse et V_{gs} passe le seuil rendant le MOSFET conducteur. LV1 est donc ramené à la masse tout comme HV1.

Nous avons donc là un fonctionnement dans les deux directions, ce qui non seulement est bien pratique pour un bus SPI comme dans le cas qui nous occupe, mais également pour d'autres bus comme i2c où l'un comme l'autre côté peuvent prendre le contrôle des lignes. Nous avons donc bien un fonctionnement bidirectionnel et notre conversion de niveaux de tensions est réutilisable dans un vaste ensemble d'applications et de projets.



LE CROQUIS

```

Fichier  Édition  Croquis  Outils  Aide

#include <SPI.h>
#include <Ucglib.h>

// Création de l'objet ucg
// avec dans l'ordre les broches pour
// CLK, MOSI, DC, CS, et RESET
//
// Le type utilisé désigne à la fois le modèle
// d'écran et le mode de connexion SPI
//
// Ici l'écran sur base ILI9341 240x320 pixels
// en couleurs 18 bits (262144 couleurs)
Ucglib_ILI9341_18x240x320_SWSPI ucg(7, 6, 5, 3, 4);

void setup(void)
{
    // 1s pour laisser l'écran s'initialiser
    delay(1000);

    // Initialisation de la logique interne,
    // des structures de la bibliothèque
    // et choix du mode de gestion des polices
    ucg.begin(UCG_FONT_MODE_TRANSPARENT);

    // effacement de l'écran
    ucg.clearScreen();
}

void loop(void)
{
    // choix de la police
    ucg.setFont(ucg_font_ncenR14r);

    // Définition de la palette de couleurs
    ucg.setColor(255, 0, 0);
    ucg.setColor(1, 255, 0,0);

    // On place notre "pointeur" pour tracer
    ucg.setPrintPos(0,15);

    // Dessin du texte
    ucg.print("Coucou monde !");

    // titi pause :)
    delay(500);
}

```

Arduino

À PROPOS DU CROQUIS

Les commentaires présents dans ce croquis relativement succinct (faute de place comme toujours pour le dernier article écrit pour un numéro) parlent d'eux-mêmes. Nous n'avons là, bien entendu qu'une toute petite partie des fonctionnalités offertes par la bibliothèque Ucglib qui non seulement se charge de piloter l'écran, mais de plus fournit un ensemble de primitives graphiques (textes, lignes, rectangles, surfaces, dégradés, etc.).

L'installation de la bibliothèque est relativement simple puisqu'il vous suffit de pointer votre navigateur web sur <http://code.google.com/p/ucglib/> et de cliquer sur le bouton *Download 1.01*. Vous obtiendrez, à terme, une archive contenant un répertoire **Ucglib** à copier dans le répertoire **libraries** de votre répertoire de croquis. Ceci vous donnera accès aux exemples livrés avec la bibliothèque.

Pour utiliser l'afficheur, il vous suffira ensuite d'inclure la bibliothèque dans votre croquis et de créer une instance de la classe **Ucglib_ILI9341_18x240x320_SWSPI** (définissant le type d'afficheur utilisé) nommée **ucg**. Les arguments passés au constructeur sont respectivement les ports utilisés pour CLK, MOSI, DC, CS, et RESET.

Il ne nous reste plus, ensuite, qu'à utiliser l'objet **ucg** à loisir avec l'une des méthodes décrites dans la sympathique documentation en ligne du projet : <http://code.google.com/p/ucglib/wiki/reference>.

Arrêtons là cet article déjà long (j'entends déjà l'infographiste en larmes parce que « ça rentrera jamais »). Je vous invite à lire le « *User Reference Manual* », mais aussi, et surtout les croquis des exemples pour explorer davantage de fonctionnalités intéressantes et imaginer ce que pourrait bien afficher cet économique petit écran. **DB**



RASPBERRY PI : CE QUE VOUS DEVEZ SAVOIR AVANT DE COMMENCER

Denis Bodor



L'arrivée récente de la carte Raspberry Pi modèle A+ complète la gamme avec un nano-ordinateur proposant un rapport prix/puissance sans doute inégalé pour l'instant. Ce modèle succède au modèle A et hérite des évolutions du modèle B+ qui lui-même est une évolution du modèle B. Voilà une belle histoire, mais pour celui qui n'a jamais mis la main sur une carte de ce type, un minimum d'explications est indispensable.

Raspberry Pi par ci, Raspberry Pi par là, on n'entend parler que de cela, non seulement chez les bidouilleurs invétérés, mais également de plus en plus chez les utilisateurs curieux, mais aussi chez quelques professionnels de l'embarqué (surtout pour les formations). Cette omniprésence est représentative du succès du produit, car en effet, depuis le début de la commercialisation en février 2012, ce sont des millions d'exemplaires qui ont été fabriqués et vendus. Le cap des 2 millions d'unités est passé fin octobre 2013, après moins de deux ans de commercialisation et les mots « trois millions » ont été entendus dernièrement de la bouche même du fondateur de l'initiative.

La Raspberry Pi est donc un incontournable depuis quelques années et le 10 novembre dernier la fondation annonçait le début de la commercialisation d'un modèle économique (une vingtaine d'euros) possédant quelques fonctionnalités en moins, mais héritant des évolutions faisant suite aux commentaires et critiques des utilisateurs.

L'histoire de la carte et de la fondation éponyme débute en 2006 avec les tout premiers prototypes, alors très différents des modèles actuels et des premières versions commercialisées. La motivation derrière le projet consiste à tenter de fournir à une nouvelle génération d'étudiants une plateforme leur permettant d'apprendre la programmation de manière simple et économique.

Il s'agit de reproduire le phénomène du début des années 80 et l'aventure BBC Micro. En ce temps-là, qui a vu naître le ZX Spectrum et la DeLorean (et certains d'entre vous), la BBC (*British Broadcasting Corporation*) initie un projet pédagogique et lance un appel d'offres décrivant un ordinateur avec un cahier des charges bien précis. S'en suit une lutte acharnée entre les compétiteurs et en particulier *Sinclair Research* et *Acorn Computers Ltd* qui finit par décrocher le contrat au grand dam de Clive Sinclair. En découle la vente de quelque 1,5 million de machines et une population britannique émerveillée découvre le monde de la micro-informatique. Depuis, *Acorn Computers Ltd* ne fabrique plus d'ordinateur, mais avait alors encore innové par la suite avec l'Acorn Archimedes et son processeur RISC dont le nom ne vous est sans doute pas inconnu :

REMARQUE

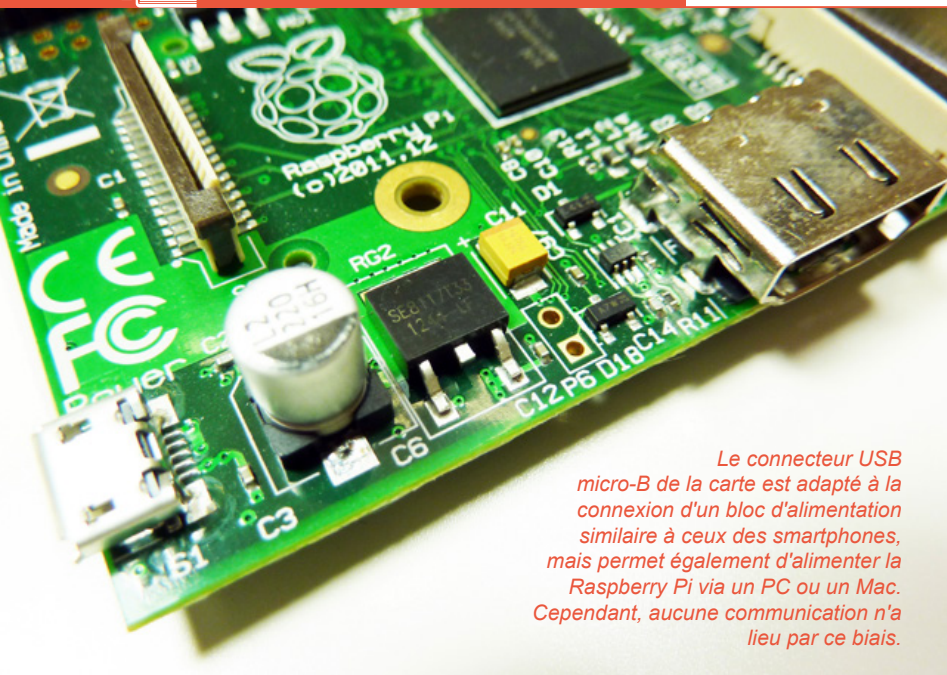
« Le » ou « La » ? Voilà l'éternel problème qui se pose lorsqu'il est question de désigner dans notre belle langue française un produit ou une marque, en particulier venant de l'espace anglophone.

Même en regardant attentivement chaque millimètre carré de la carte, il n'est pas possible d'y trouver de quelconques attributs masculins ou féminins (on a cherché, je vous passe les détails). Ainsi, certains disent « la Raspberry Pi » et d'autres « le Raspberry Pi », versions abrégées de « la carte Raspberry Pi » ou « l'ordinateur Raspberry Pi ». L'un et l'autre sont à la fois justes et faux puisque « Raspberry Pi » étant un nom propre, « on considère le plus souvent que ce nom propre hérite du genre du nom générique englobant le référent du nom propre » (merci Wikipédia (« le » ou « la » Wikipédia ?)).

Le même problème se pose avec certaines marques, dont celle d'une pâte à tartiner, et je sais de source sûre qu'en annonçant que certains disent « le Nutella » et d'autres « la Nutella », le camp opposé de celui qui se reconnaît dans l'un ou l'autre genre trouvera cela parfaitement légitime.

Tout dépend donc des habitudes de chacun et du « genre du nom générique englobant le référent du nom propre ». Si vous voyez la Raspberry Pi comme une carte, c'est juste, tout autant que si vous voyez le Raspberry Pi comme un ordinateur (on dit bien « un » PC ou « un » Mac).

Quoi qu'il en soit, pour moi, il s'agit de cartes, puisque c'est ainsi que sont vendus les différents modèles de Raspberry Pi. Si vous préférez, comme Wikipédia, considérer qu'il s'agit d'un « nano-ordinateur mono-carte à processeur ARM », libre à vous de « traduire » à la volée.



Le connecteur USB micro-B de la carte est adapté à la connexion d'un bloc d'alimentation similaire à ceux des smartphones, mais permet également d'alimenter la Raspberry Pi via un PC ou un Mac. Cependant, aucune communication n'a lieu par ce biais.

l'ARM2. Depuis, « ARM » pour *Acorn RISC Machine* est devenu « ARM » pour *Advanced RISC Machines* (1990), et la société transformée s'est spécialisée dans la conception de processeurs. Ces mêmes processeurs fabriqués par Samsung, Ti, Atmel, Freescale, NXP ou Broadcom, qu'on retrouve dans nos smartphones, nos tablettes, nos téléviseurs, nos routeurs Wifi ou encore dans la Raspberry Pi. Youpi, la boucle est bouclée !

Si l'histoire du BBC Micro vous intéresse, je vous recommande le téléfilm « *Micro Men* » avec Martin Freeman (Bilbon Sacquet, ou Watson donc) dans le rôle de Chris Curry, fondateur de *Acorn Computers Ltd* (on le trouve facilement en VOST sur YouTube). Enfin, précisons que cette « mouvance » n'était pas purement anglaise. Certains se souviendront ainsi sans doute du Thomson MO5 et du « plan informatique pour tous » lancé en 1985, faisant découvrir les joies du BASIC et du Logo à quelque 11 millions d'élèves sur plusieurs années (dont bibi).

C'est donc cette idée très intéressante qui est à la base du démarrage du projet Raspberry Pi. Après des premières ébauches sur la base d'un microcontrôleur, puis d'un SoC ARM en 2011, 50 cartes alpha sont construites et les choses s'accélérent avec une production de 10000 exemplaires plus finalisés et enfin une commercialisation en début 2012. À la fin de cette même année, un nouveau modèle B arrive et la production cesse en Asie pour être définitivement installée intégralement au Royaume-Uni dans le Pays de Galles (usines Sony de Pencoed).

La fondation Raspberry Pi est une organisation caritative ayant pour objectif de promouvoir les bases de la programmation dans les écoles. Elle est soutenue à la fois par l'université de Cambridge et la société Broadcom qui fabrique le composant principal de la carte (BCM2835) et emploie également Eben Upton, co-fondateur du projet. La fondation est divisée en deux parties : l'activité éducative et pédagogique pilotée

par Lance Howarth (ex-vice-président chez ARM), et l'activité de conception et de vente supervisée par Eben Upton (directeur technique et architecte ASIC chez Broadcom). Pour ceux d'entre vous qui en douterait, il ne s'agit clairement pas ici d'une bande de hippies rêveurs, mais d'une organisation bien structurée à l'instar, par exemple, de la fondation Mozilla supportant entre autres choses le développement du navigateur Firefox. Le temps des ingénieurs en tongs débutant une activité dans le garage parental semble bel et bien passé...

1. MAIS QU'EST EXACTEMENT UNE RASPBERRY PI ?

À présent que nous connaissons les motivations des créateurs et de la fondation, nous pouvons nous pencher sur la concrétisation de la solution qu'ils proposent pour raviver une génération de futurs développeurs et ingénieurs. De la même manière que le BBC Micro, ou le MO5 chez nous, a entraîné des légions de jeunes cerveaux vers des métiers liés aux technologies modernes, les cartes Raspberry Pi sont censées servir de base matérielle pour tous les utilisateurs curieux de comprendre et d'apprendre (jeunes ou moins jeunes).

Une carte Raspberry Pi est un ordinateur, tout comme votre PC ou votre Mac. Il y a un certain nombre de similitudes, mais aussi de différences que nous allons décrire et commenter de ce pas.

Une Raspberry Pi est un ordinateur sur une carte qui, comme ses ancêtres historiques, peut se connecter à un téléviseur en guise de moniteur. La technologie a bien changé depuis les années 80 et 90, et les différences entre votre écran de PC et votre TV de salon sont presque inexistantes. Basiquement, si vous retirez le tuner de votre TV dernier cri (ce que je vous conseille vivement étant donné le caractère lobotomisant des émissions actuelles), vous obtenez un moniteur de PC. Les cartes Raspberry Pi disposent toutes d'une connexion HDMI permettant un affichage numérique de qualité sur moniteur ou TV. C'est exactement comme votre machine de bureau ou votre ordinateur portable, à une différence près, il n'y a pas de carte graphique ni même d'emplacement pour l'insérer. Tout est directement intégré dans la puce au centre de la carte, qu'on nomme un SoC (voir article sur les différences entre Arduino et Raspberry Pi dans le présent numéro).

Les Raspberry Pi disposent également de connecteur(s) USB dont le nombre diffère en fonction

du modèle (entre 1 et 4). Ceci vous permet donc de connecter un clavier, une souris et éventuellement d'autres périphériques par l'intermédiaire d'un hub USB. Là encore, la similitude avec un PC ou un Mac est évidente.

Côté alimentation en revanche les choses sont très différentes. Alors qu'un PC sous forme d'une tour dispose d'un gros boîtier d'alimentation, une Raspberry Pi se contente d'une poignée de composants directement soudés sur la carte. Même un PC portable nécessite généralement bien plus que cela. Ceci est intimement lié à l'énergie consommée par la Raspberry Pi pour fonctionner. Alors qu'une machine classique utilise généralement une alimentation capable de fournir plus de 300 watts, un nano-ordinateur se limitera, comme un smartphone, à une poignée de watts. Le tout récent modèle A+ par exemple se contentera d'un unique watt (200 mA à 5 volts) et le modèle B+ de 3 watts.

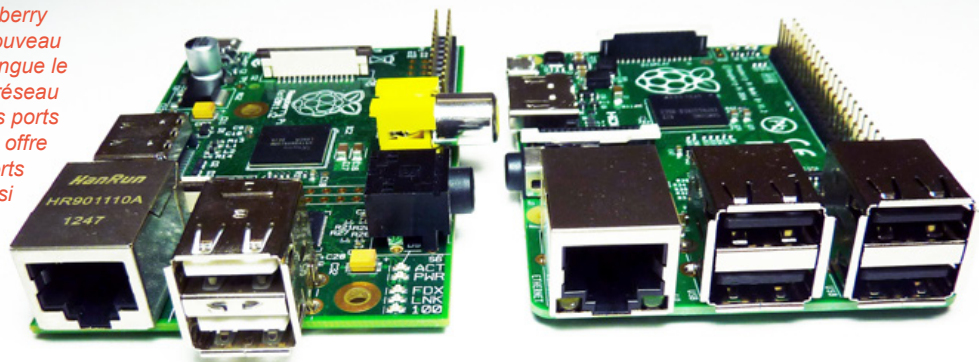
Exit également des emplacements pour des barrettes de mémoire. Une carte Raspberry Pi a effectivement de la mémoire, mais celle-ci ne peut être augmentée,

les 256 Mo des modèles A et A+, tout comme les 512 Mo des modèles B et B+, prennent la forme d'une puce directement soudée sur le SoC. Vous remarquerez au passage que la quantité de mémoire vive est bien en dessous de ce qu'on trouve actuellement sur un PC d'entrée de gamme (4 Go souvent). Ceci nous amène à une autre similarité et une autre différence importante.

Le processeur utilisé par une Raspberry Pi n'est pas visible, il se trouve à l'intérieur du SoC qui lui-même est sous la puce mémoire de la carte. Alors que dans un PC ou un Mac on a l'habitude de trouver un processeur Intel ou AMD du type Core i3, i5, i7, Celeron, Athlon ou encore Xeon ou FX, celui de la Raspberry Pi est un ARM et plus précisément un ARM1176JZF-S. Cadencé à seulement 700 Mhz, il fait partie intégrante de la puce (SoC) Broadcom BCM2835 qui contient également le processeur graphique VideoCore IV, le port USB et bien d'autres choses.

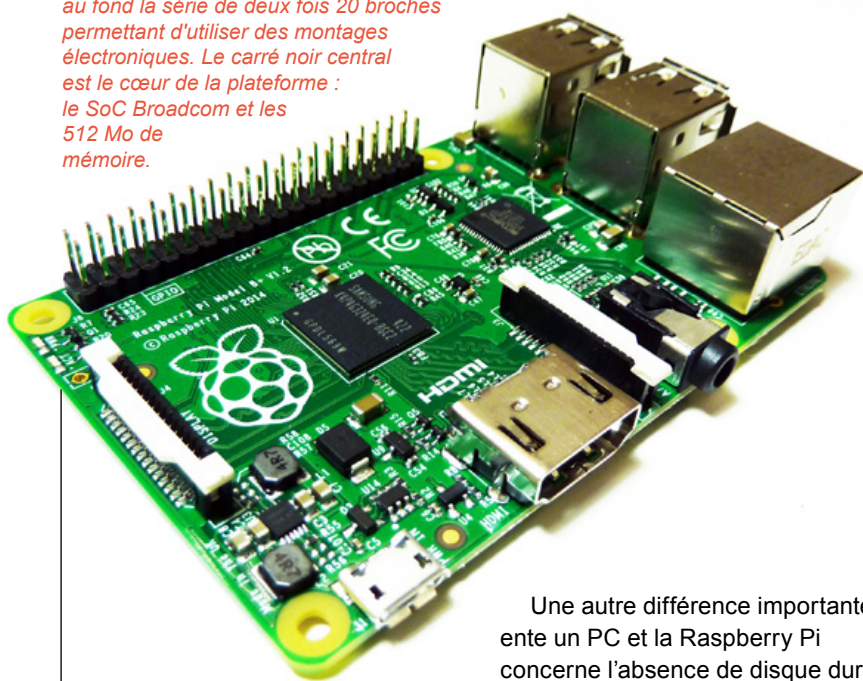
Ainsi, ce n'est pas que la taille qui est revue à la baisse. Si faire tenir un équivalent-PC dans un

À gauche une Raspberry Pi B et à droite le nouveau modèle B+. On distingue le connecteur pour le réseau filaire Ethernet et les ports USB. Le modèle B+ offre deux fois plus de ports USB permettant ainsi de brancher le clavier, la souris et d'autres périphériques.





La carte Raspberry Pi modèle B+ actuellement commercialisée. On reconnaît le connecteur USB micro-B, le port HDMI pour l'écran, le jack pour l'audio (et la sortie vidéo analogique) et au fond la série de deux fois 20 broches permettant d'utiliser des montages électroniques. Le carré noir central est le cœur de la plateforme : le SoC Broadcom et les 512 Mo de mémoire.



rectangle de 6 centimètres par 9 semble être une prouesse, ceci ne se fait pas sans concession. L'une d'elles concerne la puissance. Un processeur ARM11 à 700 Mhz avec 256 ou 512 Mo de mémoire n'arrive de loin pas à la cheville d'un PC même bas de gamme. De plus, la différence importante dans la nature même du processeur impacte directement le système pouvant être utilisé et de ce fait les applications. Oubliez Windows et Mac OS X, Word, Excel, World of Warcraft, Photoshop, etc. Le système principalement utilisé avec une Raspberry Pi est GNU/Linux, mais il est également possible d'installer RISC OS, FreeBSD, NetBSD ou Plan 9. Et si vous vous posez la question, non il n'existe pas de version stable et utilisable d'Android (ou CyanogenMod) pour la Raspberry Pi à l'heure actuelle (il semble même que le projet RazDroid soit mort).

Une autre différence importante entre un PC et la Raspberry Pi concerne l'absence de disque dur. Le support de stockage de prédilection est la carte SD ou microSD (modèles B+ et A+). Cette dernière contiendra le système, les applications et les données. Bien entendu, la présence d'un ou plusieurs ports USB permet l'utilisation de clés USB ou de disques durs externes USB, mais uniquement pour le stockage et non pour le démarrage du système. Il est possible de bidouiller, mais une carte SD sera nécessaire même si c'est en lecture seule. Si le mot Sata vous a traversé l'esprit, oubliez-le immédiatement, la Raspberry Pi n'est pas un PC et rares sont les cartes qui supportent la connexion d'un disque Sata (Cubieboard et Cubietruck/Cubieboard3, ou encore la A20-OLinXino d'Olimex le peuvent).

Ce résumé peut paraître peu réjouissant : moins de mémoire, moins de puissance, pas de

compatibilité PC/Mac, pas de disque... Mais les Raspberry Pi comme d'autres cartes nano-ordinateurs vous apportent quelque chose en plus : des connecteurs pour brancher plein de choses amusantes. On parle généralement de GPIO (*General Purpose Input/Output*), pour désigner ces entrées/sorties d'usage général qui prennent la forme d'une série de connecteurs contrôlables depuis des programmes se trouvant dans le système. Strictement parlant un (ou une) GPIO est caractérisé(e) par sa fonction générique et le fait qu'on peut soit écrire une valeur en sortie (à la masse ou au niveau logique haut, généralement +3,3V ou +5V), soit lire l'état d'une broche pour obtenir une information de l'extérieur (comme l'état d'un interrupteur ou d'un bouton poussoir).

Cependant, la définition du terme dérive peu à peu à force d'être utilisée pour désigner n'importe quel connecteur susceptible de servir au branchement d'éléments supplémentaires à une carte. Les constructeurs prévoient généralement des utilisations alternatives des connecteurs. On peut s'en servir comme GPIO ou, dans un autre mode, comme interface spécifique utilisant une norme et un protocole dédiés. Ainsi, parmi les connecteurs de la Raspberry Pi, certains peuvent faire office de ports série, SPI ou i2c. Ne vous souciez pas de la signification de ces termes pour l'instant si vous débutez, sachez simplement qu'il s'agit de standards permettant de brancher et de dialoguer avec des modules, puces et circuits utilisant ce type d'interfaces.

DÉCOUVREZ NOS NOUVELLES OFFRES D'ABONNEMENTS !

PRO OU PARTICULIER = CONNECTEZ-VOUS SUR :

www.ed-diamond.com



LES COUPLAGES PAR SUPPORT :

VERSION PAPIER



Retrouvez votre magazine favori en papier dans votre boîte à lettres !

VERSION PDF



Envie de lire votre magazine sur votre tablette ou votre ordinateur ?

Sélectionnez votre offre dans la grille au verso et renvoyez ce document complet à l'adresse ci-dessous !

Voici mes coordonnées postales :

Société :	
Nom :	
Prénom :	
Adresse :	
Code Postal :	
Ville :	
Pays :	
Téléphone :	
E-mail :	

- ☐ Je souhaite recevoir les offres promotionnelles et newsletters des Éditions Diamond.
☐ Je souhaite recevoir les offres promotionnelles des partenaires des Éditions Diamond.

En envoyant ce bon de commande, je reconnais avoir pris connaissance des conditions générales de vente des Éditions Diamond à l'adresse internet suivante : boutique.ed-diamond.com/content/3-conditions-generales-de-ventes et reconnais que ces conditions de vente me sont opposables.

HACKABLE
~ MAGAZINE ~

Édité par Les Éditions Diamond
Service des Abonnements
B.P. 20142 - 67603 Sélestat Cedex
Tél. : + 33 (0) 3 67 10 00 20
Fax : + 33 (0) 3 67 10 00 21

Vos remarques :

.....
.....

VOICI TOUTES LES OFFRES COUPLÉES AVEC HACKABLE !

POUR LE PARTICULIER ET LE PROFESSIONNEL ...

Prix TTC en Euros / France Métropolitaine

CHOISISSEZ VOTRE OFFRE !

SUPPORT

Prix en Euros / France Métropolitaine

Offre ABONNEMENT

HK
6^{no}
HK*

Réf
Tarif TTC
HK1 39,-

PDF 1 lecteur
Réf
Tarif TTC
HK12 58,-

PAPIER

PAPIER + PDF

PAPIER + BASE DOCUMENTAIRE

PAPIER + PDF + BASE DOCUMENTAIRE

LES COUPLAGES « EMBARQUÉ »

D	6 ^{no} HK*	+	4 ^{no} OS		☐ D1 65,-	☐ D12 98,-	☐ D13 85,-*	☐ D123 118,-*
E	6 ^{no} HK*	+	4 ^{no} OS	6 ^{no} MISC	☐ E1 105,-	☐ E12 158,-	☐ E13 179,-*	☐ E123 232,-*
E+	6 ^{no} HK*	+	4 ^{no} OS	6 ^{no} MISC + 2 ^{no} HS	☐ E+1 119,-	☐ E+12 179,-	☐ E+13 193,-*	☐ E+123 253,-*
F	6 ^{no} HK*	+	4 ^{no} OS	1 ^{1no} GLMF	☐ F1 125,-	☐ F12 188,-	☐ F13 229,-*	☐ F123 292,-*
F+	6 ^{no} HK*	+	4 ^{no} OS	1 ^{1no} GLMF + 6 ^{no} HS	☐ F+1 183,-	☐ F+12 275,-	☐ F+13 287,-*	☐ F+123 379,-*
G	6 ^{no} HK*	+	4 ^{no} OS	6 ^{no} LP	☐ G1 100,-	☐ G12 150,-	☐ G13 164,-*	☐ G123 214,-*
G+	6 ^{no} HK*	+	4 ^{no} OS	6 ^{no} LP + 3 ^{no} HS	☐ G+1 129,-	☐ G+12 194,-	☐ G+13 193,-*	☐ G+123 258,-*

LES COUPLAGES « GÉNÉRAUX »

H	6 ^{no} HK*	+	4 ^{no} OS	6 ^{no} LP	6 ^{no} MISC	1 ^{1no} GLMF	☐ H1 200,-	☐ H12 300,-	☐ H13 402,-*	☐ H123 499,-*
H+	6 ^{no} MISC	+	2 ^{no} HS	1 ^{1no} GLMF	6 ^{no} HS		☐ H+1 301,-	☐ H+12 452,-	☐ H+13 493,-*	☐ H+123 639,-*

Les abréviations des offres sont les suivantes : LM = GNU/Linux Magazine France | HS = Hors-Série | LP = Linux Pratique | OS = Open Sillicium | HC = Hackable

* HK : Attention : La base Documentaire de Hackable n'est pas incluse dans l'offre.

N'hésitez pas à consulter les détails des offres ci-dessus sur : www.ed-diamond.com !

Voilà donc dans les grandes lignes ce qui constitue une Raspberry Pi. En résumé, voici ce qu'il faut retenir au minimum :

- c'est moins puissant qu'un ordinateur standard,
- par contre, c'est bien plus petit et moins gourmand en énergie,
- cela offre des possibilités de bidouille intéressantes en termes d'électronique et de bricolage,
- les applications Windows et Mac OS ne fonctionnent pas avec ce type de nano-ordinateur,
- si vous n'êtes pas déjà un utilisateur de GNU/Linux, il va falloir prendre le temps d'apprendre à vous en servir.

2. CE QU'IL VOUS FAUT POUR DÉBUTER

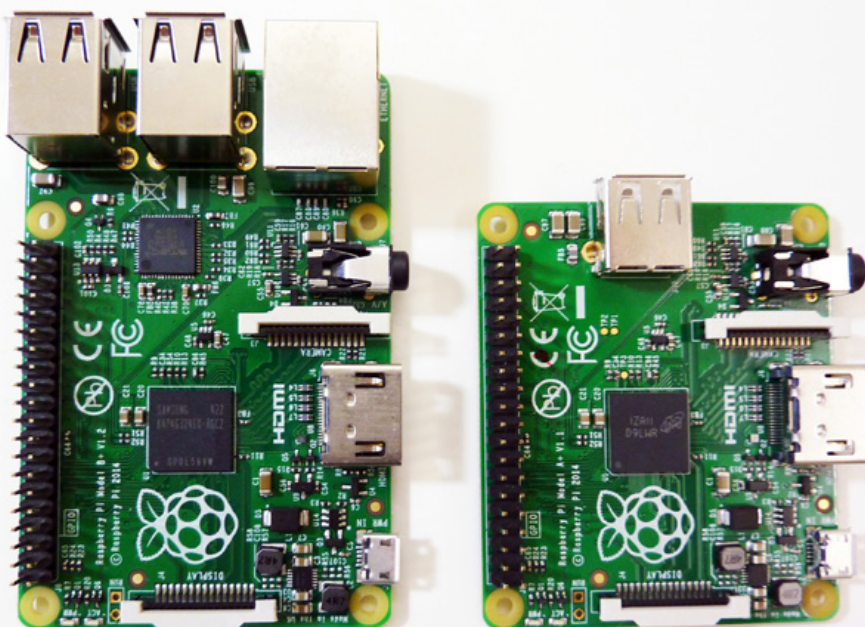
À ce stade de l'article, vous devez avoir une vision plus ou moins juste de ce qui vous attend en partant à la découverte du monde de la Raspberry Pi ou d'autres matériels de ce type (parce qu'il en existe beaucoup). En revanche, il nous paraît maintenant utile d'insister sur un point particulier très important : le coût.

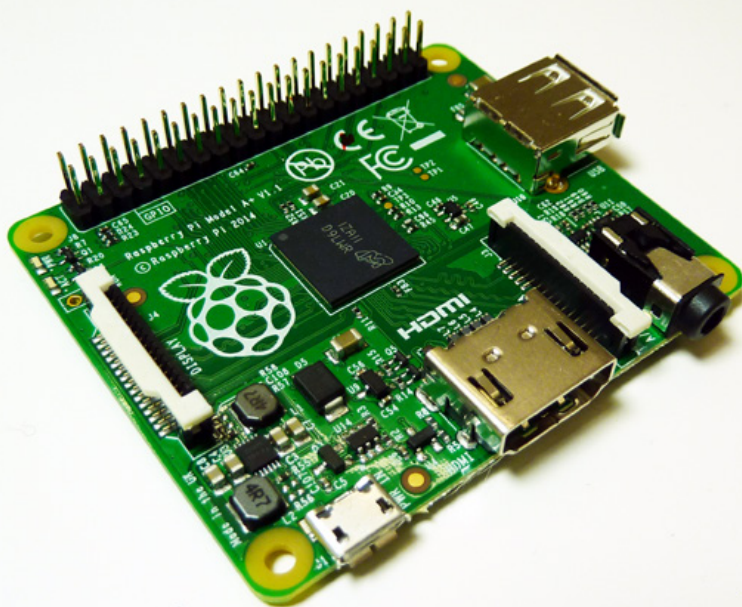
Avec l'annonce récente de la Raspberry Pi A+ à une vingtaine d'euros, ou même de la B+ à une trentaine d'euros, il est tentant de penser que votre « budget découverte » se limitera à ce petit montant. Ce n'est bien entendu

pas le cas, car à ce prix vous n'avez que la carte et rien que la carte. À ce montant, il faudra impérativement ajouter :

- Un bloc d'alimentation avec un connecteur USB micro-B ou au minimum un câble USB micro-B qui vous permettra de connecter la Raspberry Pi à un hub ou à un PC. Cette connexion ne servira qu'à alimenter la carte, aucune communication n'a lieu de cette manière. Il est cependant préférable d'utiliser un bloc d'alimentation et celui de votre smartphone pourra faire l'affaire.
- Une carte SD (modèle A et B) ou microSD (A+ et B+) d'une taille minimum de 4 Go avec une taille recommandée d'au minimum 8 Go. Choisissez de préférence une carte de bonne qualité et de marque réputée afin de garantir l'intégrité des données. Il faut savoir que le nombre de cycles d'écriture sur ce type de supports est limité et que certaines cartes bas de gamme utilisent une technologie de mémoire flash MLC (*Multi Level Cell*) ou TLC (*Triple Level Cell*) faisant usage d'une seule cellule mémoire pour stocker plusieurs bits, par opposition à la technologie SLC (*Single Layer Cell*) plus coûteuse, mais plus sûre à long terme. De plus, les cartes SD ou microSD de qualité (et donc chères) intègrent une « intelligence » qui gère automatiquement l'utilisation des cellules mémoire et donc répartit l'usure plus judicieusement. Ajoutons également qu'en toute logique, pour préparer la carte SD ou microSD, vous devez disposer

Petite comparaison des modèles actuellement vendus avec à gauche le B+ et à droite le A+. La majorité des composants des cartes est identique, mais la A+ ne dispose que d'un seul connecteur USB et ne permet pas de connexion réseau Ethernet.





Une carte Raspberry Pi modèle A+. Moins chère et plus petite, elle sera moins adaptée à une utilisation en guise de nano-ordinateur, mais conviendra parfaitement pour des usages domotiques par exemple.

d'un lecteur de ce type utilisable avec un PC ou un Mac (il existe une technique permettant d'utiliser le port USB d'une Raspberry Pi modèle A en mode périphérique pour écrire sur une SD, ceci nécessite cependant beaucoup d'efforts et certaines connaissances avancées).

Ces deux éléments représentent le strict minimum, mais il est fort probable que vous deviez aussi ajouter :

- Un couple clavier/souris afin de pouvoir contrôler directement le système sur la Raspberry Pi.
- Un moniteur ou un téléviseur avec un connecteur HDMI (ou un câble HDMI/DVI). Les modèles B et B+ permettent également de connecter un écran avec une entrée analogique composite (RCA sur B et Jack sur B+), mais ceci n'est généralement utilisé que dans des cas bien spécifiques. Il faut cependant bien comprendre qu'il n'existe pas de possibilité de réutiliser/recycler un vieux moniteur avec une entrée VGA, sauf en utilisant un boîtier convertisseur qui ne coûte souvent pas beaucoup moins qu'un écran DVI/HDMI bas de gamme.

Ceci vous permettra d'utiliser la Raspberry Pi comme un PC standard sous GNU/Linux. Il existe

des techniques plus ou moins avancées permettant de se passer de l'ensemble complet écran/clavier/souris, à commencer par l'abandon de l'interface graphique pour une utilisation en ligne de commandes seule (plus de souris). Les autres solutions, comme l'utilisation de la Raspberry Pi au travers du réseau (SSH) ou à partir d'un PC/Mac (console série) nécessiteront également des ajouts :

- Un simple câble réseau Ethernet (modèle B ou B+) avec éventuellement un *switch* Ethernet si vous n'êtes pas équipé.
- Un adaptateur Wifi USB, car la carte, quel que soit le modèle, ne dispose d'aucune connectivité sans fil (Wifi, 3G/4G ou Bluetooth).
- Un adaptateur USB/série fonctionnant en 3,3V qui permettra la connexion directe depuis un PC.

Selon votre présent niveau d'équipement et la réutilisation d'éléments actuellement dédiés au PC par exemple (clavier, écran, etc.), votre budget peut allègrement grimper à 150 ou 300 euros, le coût le plus important provenant bien entendu de la décision de dédier un écran à la Raspberry Pi. Finalement, la carte elle-même n'est financièrement qu'un petit élément de l'ensemble, ce qui est plutôt une bonne chose finalement, car si vous faites une bêtise, c'est la Raspberry Pi qui risque le plus et non les périphériques.

3. A OU B, PLUS OU PAS PLUS ?

À l'heure actuelle nous avons quatre modèles standards de cartes Raspberry Pi auxquels s'ajoute le *Raspberry Pi Compute Module* plus orienté vers les développements industriels et que nous passerons sous silence dans le présent article.

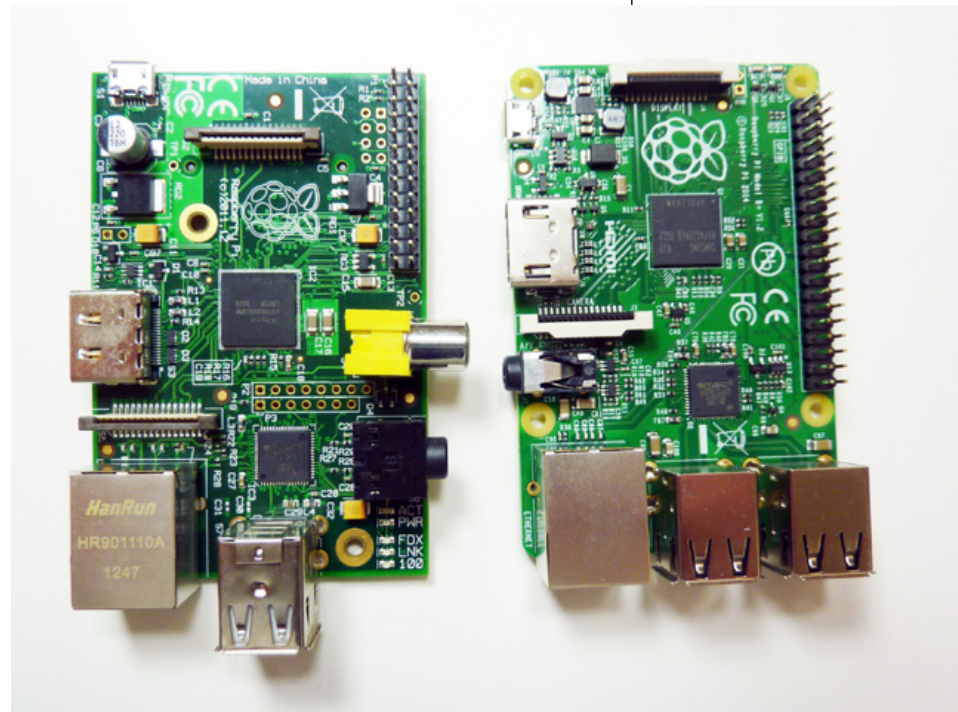
Les modèles les plus récents sont naturellement ceux qui sont recommandés pour de nouveaux achats et sont désignés sous les noms B+ et A+. Les modèles A et B sont maintenus à des fins de compatibilités mécaniques. En effet, d'une génération à l'autre, bien que le SoC soit toujours le même et qu'une majorité de composants soient identiques, les cartes ont changé de forme. Ainsi, un modèle B+ ne saurait remplacer un modèle B défectueux dans un boîtier ou dans une installation dédiée, ne serait-ce que parce que les trous de fixation ne sont tout bonnement plus aux mêmes endroits.

La question du « plus » ou « pas plus » ne se pose donc pas s'il s'agit d'un achat pour un nouveau projet ou en guise de première expérience. A+ ou B+ en revanche est une tout autre histoire. Nous pouvons résumer les différences via le tableau suivant :

Le reste des caractéristiques, comme le SoC utilisé ou les connecteurs (GPIO, LCD, caméra, alimentation) sont parfaitement identiques. Il est même amusant de remarquer qu'au niveau de la carte, la A+ est tout simplement une B+ avec 2 centimètres de circuit en moins et quelques composants déplacés pour l'occasion.

La plus importante différence concerne la disparition de la puce SMSC LAN9514 contenant à la fois un hub USB 4 ports et une interface USB/Ethernet ce qui explique l'absence du connecteur réseau filaire, mais aussi le passage de 4 ports USB à un seul (celui natif du SoC Broadcom).

Comparaison des deux modèles B (à gauche) et B+ (à droite). Les deux cartes disposent exactement des mêmes fonctionnalités et s'utilisent de la même manière. L'évolution consiste principalement en un réagencement des composants et une révision du circuit imprimé. Le modèle B+ dispose de plus de ports USB, de plus de GPIOs et d'une alimentation plus efficace.



	B+	A+
Prix	~30€	~20€
Mémoire	512 Mo	256 Mo
Vidéo	HDMI+composite	HDMI
Port USB	4	1
Réseau	Ethernet 10/100	-
Taille	8,56 cm par 5,4 cm	6,5 cm par 5,65 cm
Poids	45g	23g



Le dessous des cartes Raspberry Pi B+ et B. On y trouve surtout des composants passifs (condensateurs et résistances) et les emplacements pour les cartes SD ou microSD qui servent de supports de stockage, à l'instar des disques durs dans les PC.

Les conséquences pour l'utilisateur sont une économie d'énergie, mais aussi, et surtout le fait de non seulement ne plus avoir du tout de réseau sur la carte seule, mais aussi, et surtout de devoir impérativement utiliser une solution alternative pour le duo clavier/souris (soit en utilisant un hub USB, soit en optant pour un clavier comprenant un pad tactile). Il est évident qu'en utilisant l'unique port USB de la A+ pour un adaptateur Ethernet ou Wifi, nous avons là une solution qui se prête à un ensemble limité d'utilisations (système d'affichage réseau, contrôleur domotique, etc.).

Très sincèrement, si vous débutez avec la Raspberry Pi, mieux vaut opter pour la B+ et une utilisation immédiate en guise d'ordinateur plus ou moins standard le temps de trouver vos repères. Avec l'expérience, vous serez à même éventuellement d'envisager une utilisation spécifique. Je pense naturellement à des projets qui vont utiliser la Raspberry avec une certaine autonomie (comprendre « avec moins d'interventions d'un utilisateur »), voire avec une alimentation sur batterie. Certes, dans ce cas, la sortie HDMI perd un peu de son intérêt, mais pour 20 euros on ne va pas faire la fine bouche...

4. CONCLUSION : POUR NE RIEN VOUS CACHER...

Soyons clairs, avec les Raspberry Pi comme pour n'importe quelle carte, équipement, module, devkit ou matériel, vous aurez ce pour quoi vous avez payé.

En d'autres termes, ne vous attendez pas à ce qu'une carte de 20 ou 30 euros affiche des performances et un niveau de qualité de conception à la hauteur d'une carte pour l'embarqué ou l'évaluation d'une plateforme à 100, 300 ou 1000 euros (parfois plus encore).

Les Raspberry Pi sont loin d'être parfaites et même si cela s'améliore avec les modèles A+ et B+, un certain nombre de problèmes ou de carences existent. Les modèles A et B étaient critiqués pour les choix faits dans la conception du circuit d'alimentation (voir article suivant sur la transition A/B vers A+/B+) et les problèmes qui en découlait, en particulier au niveau de l'USB. Ceci est quelque chose de corrigé avec les modèles « plus », mais d'autres points sont toujours critiquables.

Deux d'entre eux sont généralement mis en avant et font toujours écho dans les discussions autour des Raspberry Pi. Le premier concerne l'absence d'une RTC quel que soit le modèle de carte. Comme nous l'avons

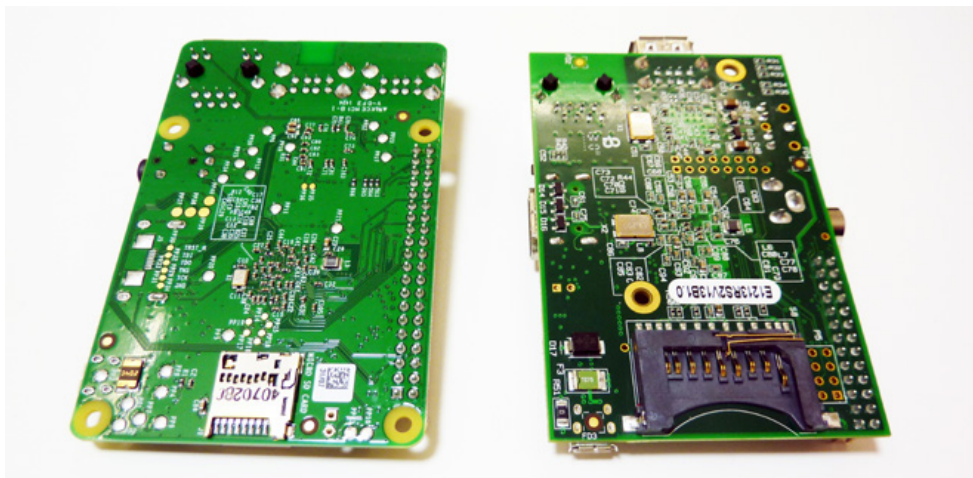
détaillé dans le premier numéro de *Hackable* (p. 56), il est nécessaire d'ajouter un module ou un composant supplémentaire pour qu'une Raspberry Pi conserve l'heure entre deux mises sous tension. En l'absence de cet ajout, la carte repose sur la disponibilité du réseau pour obtenir l'heure et la date actuelles lors de son démarrage. En cas de défaut de ce type de connexion, la date est fixée arbitrairement et doit être corrigée manuellement.

L'autre problème concerne le support USB intégré dans le SoC Broadcom et pris en charge par le noyau Linux (la base du système GNU/Linux de la Raspberry Pi). Il est communément admis que ce contrôleur USB, la façon dont il est intégré dans le SoC et son pilote au sein du noyau jouissent d'une bien triste réputation. Ceci au point que Greg Kroah-Hartman, responsable du support USB dans le noyau, avec son tact habituel, en est arrivé à publiquement conseiller le choix d'une autre plateforme

(Beaglebone) à toutes personnes « voulant faire de véritables transferts de données en USB », car « le matériel ne marche pas » (*is broken*), « le pilote ne marche pas », son code est « si mauvais que personne ne veut y toucher » et « que le fait que l'USB fonctionne tout simplement est déjà un vrai miracle » (citations tirées des réponses de Greg sur la liste des développeurs Linux suite à la question « *USB is very slow. Is this a known bug?* » (« l'USB est très lent. Est-ce un bug connu ? »)).

Certes les développeurs en charge de ces éléments n'ont pas pour habitude de prendre des gants et leur franchise choque parfois certains (cf. le « *Nvidia, Fuck you!* » de Linus Torvalds), mais les faits sont là. Il ne faudra donc pas vous attendre à des performances grandioses en termes, par exemple, de streaming vidéo d'une caméra USB via l'Ethernet...

Encore une fois, vous en aurez pour votre argent et cela s'arrête là. En reprenant le concept de nano-ordinateur de la Raspberry Pi et en lui ajoutant les éléments et composants performants d'un niveau de qualité supérieure (RTC, interface Ethernet dédiée, flash interne, etc.), vous obtenez une carte avec un prix deux ou trois fois supérieur. Ce raisonnement s'applique tout autant aux nano-ordinateurs qu'à l'équipement de mesure ou à n'importe quel matériel. Il convient simplement de garder cela à l'esprit... ce qui ne gâche en rien le plaisir de la découverte, de l'apprentissage et de l'exploration. **DB**



L'un des principaux changements entre les modèles B et B+ (ainsi que A et A+) est le passage du format SD à microSD, un format bien plus compact plus adapté à ce type de plateforme.



RASPBERRY PI « PLUS » : QUOI DE NEUF DOCTEUR ?

Denis Bodor



L'arrivée du modèle A+ est presque une conséquence directe de celle du modèle B+ destiné à remplacer le modèle B maintenant largement utilisé. Il serait faux de parler de deux véritables nouvelles cartes, mais les changements sont suffisamment importants pour ne pas parler non plus d'une simple révision mineure comme il y en a eu précédemment pour chacun des modèles. Voyons ensemble l'étendue de ce que nous nommerons donc une « évolution notable ».

Ceci est peut-être passé inaperçu pour une grande partie des utilisateurs, mais ce n'est pas la première fois que les cartes Raspberry Pi modèles A et B changent. Entre 2012 et 2014, il y eut pas moins d'une demi-douzaine de révisions par modèle au total sans compter les versions beta ou chinoises (circuit imprimé rouge). Les différences entre B+ et A+ se résument à la présence du réseau filaire (Ethernet) et au nombre de ports USB, nous concentrerons nos explications sur les différences entre les modèles B et B+.

1. CE QUI NE CHANGE PAS

Avant de parler de ce qui change, voyons ce qui reste identique, à commencer par le SoC Broadcom BCM2835 et ses 512 Mo de mémoire soudés par dessus. Les modèles A, A+, B, B+ ainsi que le *compute module* partagent le même composant central fournissant le processeur et la majorité des périphériques et fonctionnalités. Toute la partie logicielle reste donc absolument identique et compatible, à condition de garder son système à jour. Cette précision est importante, car l'un des changements faits entre le modèle B et B+ concerne un composant gérant l'USB et l'Ethernet, composant disposant d'un pilote dans le système.

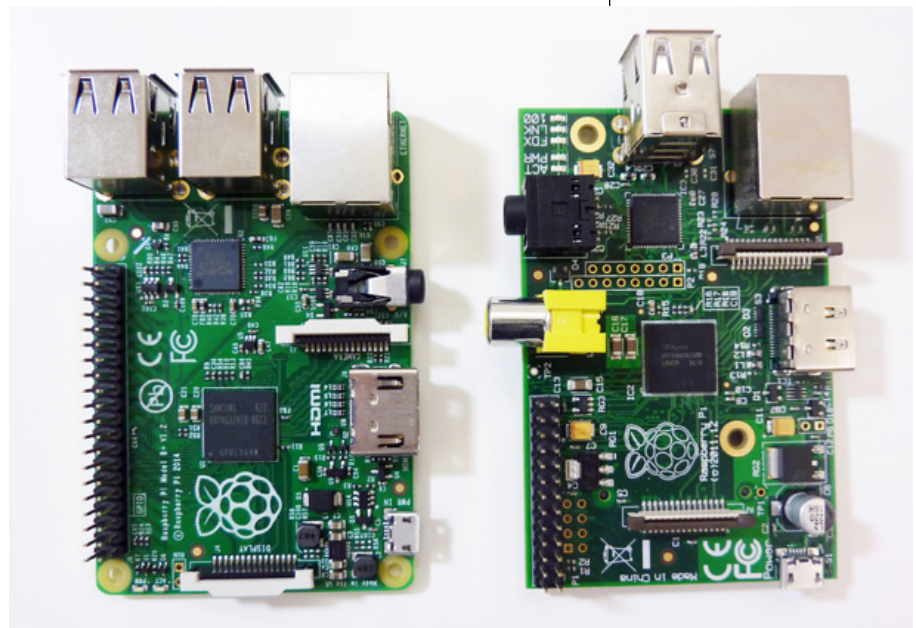
Ainsi, si vous utilisez un modèle B avec un système non mis à jour (plus ancien que juin

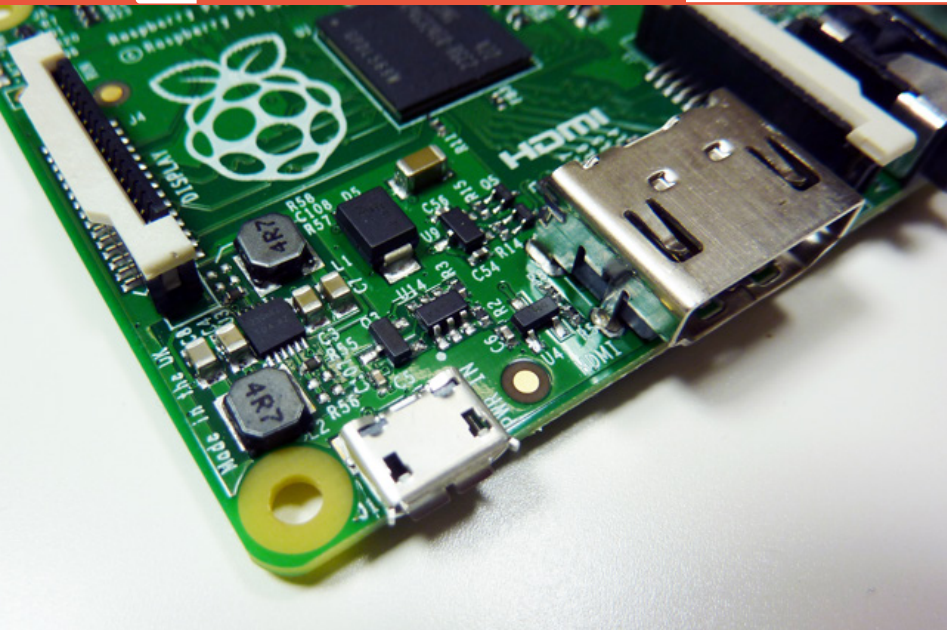
2014) et que vous tentez de migrer ce dernier sur la B+ il y a de fortes chances que les ports USB ainsi que l'Ethernet ne soient plus utilisables, et difficiles à mettre à jour. En effet, sans clavier et sans connexion Internet il devient obligatoire de jongler de manière plus ou moins cabalistique avec des copies de fichiers et/ou un émulateur. Assurez-vous donc simplement de garder votre système à jour avec les commandes **sudo apt-get update** et **sudo apt-get dist-upgrade**. Et une éventuelle migration B vers B+ n'est pas la seule raison de le faire. Les mises à jour sont importantes à la fois pour améliorer la stabilité, mais aussi la sécurité du système.

2. RÉAGENCEMENT DES ÉLÉMENTS DE LA CARTE

L'élément le plus visible des changements introduits par la B+ est sans conteste son aspect. Bien qu'exactly de la même taille que le modèle B (85,60 mm par 53,98 mm) et de la même couleur (c'est important pour certains) des caractéristiques mécaniques sont à noter. Ainsi, les trous de fixation initialement au nombre de deux

Comparaison visuelle des modèles B+ à gauche et B à droite. Au menu, réagencement des connecteurs, plus de broches sur P1, deux fois plus de ports USB, meilleure alimentation, placement des trous de fixation plus « rationnelle » et... coins arrondis.





L'un des principaux changements dans l'électronique des modèles B+ et A+ concerne le passage d'une alimentation linéaire à une alimentation à découpage bien plus efficace et en l'ajout de bon nombre de mécanismes de sécurité et de protection.

(proche du connecteur Ethernet et entre le SoC et l'alimentation), métallisés et d'un diamètre de 2,9 mm (acceptant des vis de 3 mm, type carte mère de PC) passent à 4 agencés en un rectangle de 58 mm par 49, non métallisés et d'un diamètre de 2,75 mm (oubliez la visserie PC).

Ce changement notable impacte directement l'utilisation « en production » de la carte avec d'un côté une fixation plus « conventionnelle » et fiable/solide, mais de l'autre une évidente incompatibilité avec le modèle B. L'autre changement du même type, plus cosmétique, est un passage à des coins arrondis qui permettra éventuellement l'intégration de la carte dans des boîtiers avec une forme similaire (non, la B+ ne rentre pas dans une boîte d'Altoids).

L'agencement des composants et des connecteurs a lui aussi bien changé même si on retrouve les connecteurs sur les bords identiques ou presque. Le SoC garde bien entendu son emplacement central et la puce intégrant le hub USB et l'Ethernet est toujours sur le bord Est. Les connecteurs cependant se trouvent presque tous décalés. Le HDMI et le connecteur caméra s'éloignent du connecteur Ethernet faisant de la place pour le jack qui change de bord. Le connecteur (*flat flex*) pour l'écran se rapproche du bord. Le connecteur d'alimentation micro-B passe du bord Ouest au bord Sud.

Des connecteurs apparaissent et d'autres disparaissent. Ainsi sur le modèle B+, on passe de deux ports USB à quatre, mais le connecteur RCA (jaune) pour la sortie vidéo composite n'est plus là. Le connecteur Jack initialement à trois conducteurs

passé à quatre et transporte donc maintenant les signaux audio *out* droite et gauche ainsi que la vidéo analogique. On pourra utiliser un câble spécifique comme il en existe pour certains appareils photo numériques, Jack vers 3 RCA (rouge, blanc et jaune).

Enfin, le changement le plus important en terme de confort d'utilisation concerne le passage d'un emplacement pour carte SD à friction à un emplacement microSD de type *push-pull*. L'avantage de ce format est évident puisque la carte s'en trouve bien mieux intégrée, ne dépassant que de deux petits millimètres, avec comme conséquence une impossibilité de simplement prendre le support d'un modèle B et de le placer dans une carte B+. Ceci, bien entendu reste possible, mais uniquement si :

- vous avez effectivement tenu votre système à jour,
- et avez utilisé une microSD via un adaptateur SD comme ceux souvent livrés dans des packs sous blister.

Dans ce cas précis, oui, le petit miracle s'accomplira sans le moindre problème avec un démarrage et un fonctionnement strictement identiques d'une carte Raspberry Pi à l'autre.

Tous ces changements physiques ont une conséquence relativement fâcheuse. En effet, si vous avez fait l'effort d'acquérir un boîtier plastique ou polycarbonate pour votre ou vos Raspberry Pi, il ne sera pas utilisable pour le modèle B+. Pire encore, un certain nombre de *shield* ou cartes/modules additionnels ne peuvent

tout simplement plus être utilisés (du moins sans modifications notables). Je pense notamment par exemple au module audio Wolfson proposant un DAC audio d'excellente qualité (du Wolfson quoi) et utilisant le connecteur P5 se trouvant (non peuplé) juste à côté du connecteur GPIO P1 sur le modèle B. D'autres modules n'ont pas de problème avec le connecteur lui-même, mais avec les autres éléments de la carte. C'est le cas de la Xtrinsic Sensor Board de Freescale fournissant un capteur de pression, un magnétomètre 3D et un accéléromètre. Le connecteur CN2 qui dépasse du module se retrouve ainsi en contact avec la masse du connecteur HDMI...

Ce changement d'agencement, bien que plus « normal » sur les B+ a donc fait enrager certains utilisateurs d'autant plus qu'il semblerait que certains fournisseurs aient livré des modèles B+ en lieu et place des B commandés, et ce sans avertir leurs clients. Certains n'ont d'ailleurs pas manqué de faire remarquer que des changements mécaniques sur un produit au grand dam des fabricants d'accessoires complémentaires rappelaient certaines manœuvres marketing d'une société ayant une pomme pour logo et s'amusant à changer le connecteur de ses smartphones à 600 euros d'un modèle à l'autre.

Mais n'oublions pas que tout changement a des conséquences et que la compatibilité ascendante n'est pas chose facile lorsqu'on tente de procéder à des améliorations notables. Le gain

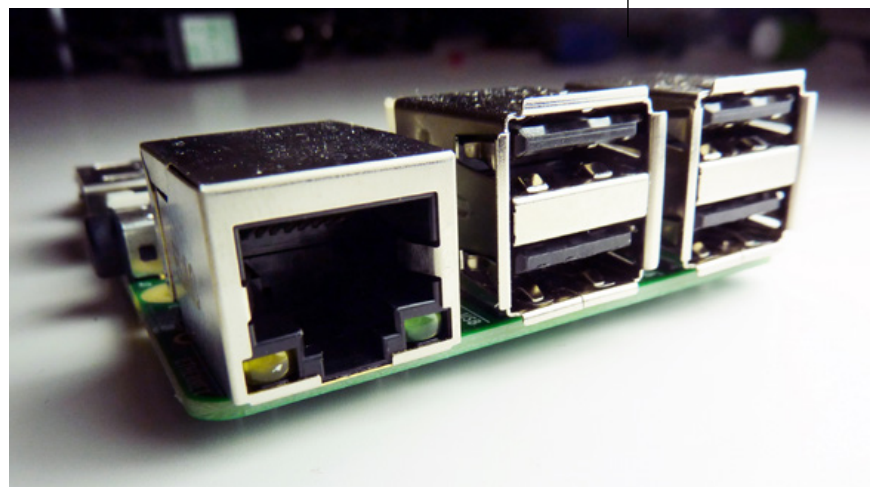
obtenu avec un passage à la B+, en particulier pour ce qui est des trous de fixation est suffisamment important, selon nous, pour accepter ce genre de désagrément. Rappelons également que Eben Upton a bien précisé que le modèle B restait disponible pour ces raisons précises de compatibilité même s'il semblerait que les détaillants fassent de regrettables raccourcis pour honorer les commandes.

3. GPIOS ET BROCHAGE

Le gros changement en termes de confort d'utilisation concerne la série de connecteurs P1 qui passe de 2×13 à 2×20 ajoutant ainsi 9 GPIO et deux connecteurs de masse (GND). La possibilité d'utiliser uniquement 17 entrées/sorties (en ne faisant pas usage de l'i2c, du port série et du bus SPI) était jugée bien trop restreinte par les utilisateurs, et ce à juste titre. En effet, en activant toutes les fonctions alternatives (un port série, une RTC en i2c et un écran en SPI, par exemple), il ne restait finalement que 7 broches pour connecter des modules et montages « maison ».

La Raspberry Pi modèle B+ porte donc ce nombre à 16 GPIOs ouvrant des perspectives bien plus intéressantes en termes de bidouille électronique. Il devient ainsi plus facile d'envisager des utilisations comprenant par exemple des écrans LCD HD44780 (en 8 bits), des relais/MOSFET/opto pour le contrôle de matériel 230V ou encore de simples boutons sans avoir recours à un module/shield extender i2c/GPIO.

Ce qui caractérise visuellement le modèle B+ est incontestablement l'arrivée de deux ports USB supplémentaires permettant ainsi de connecter le classique duo clavier/souris ainsi que d'autres périphériques.



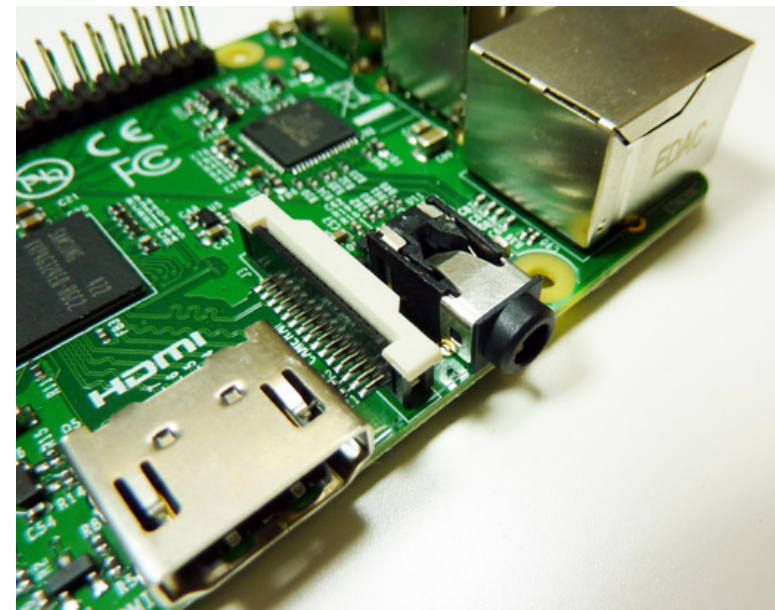


Pour une utilisation avec câble et platine à essai (*breadboard*), cette évolution du port P1 ne présente que des avantages. En effet, l'ensemble de la nomenclature des broches 1 à 26 est strictement identique entre un modèle B et B+ (et A ou A+). Nous avons, par exemple, dès réception des modèles B+, fait l'essai avec l'écran LCD SPI présenté dans le précédent numéro, et ce sans le moindre problème. De ce fait, l'ensemble des tutoriels en ligne et de nos articles est parfaitement applicable aux modèles B+ sans aucun changement.

4. CHANGEMENTS USB ET ETHERNET

Côté composant sur la carte, le gros changement est le remplacement de la puce située à proximité des connecteurs USB et Ethernet. Le SMSC LAN9512 laisse sa place à un LAN9514. De la même famille de composants, ce changement impacte le nombre de ports, car le LAN9514, en plus d'intégrer un convertisseur USB/Ethernet, comprend un hub 4 ports en lieu et place des 2 ports du modèle LAN9512.

C'est cette modification qui oblige une mise à jour du firmware et plus exactement du noyau du système GNU/Linux faisant fonctionner la Raspberry Pi. On notera également que le passage du LAN9512 au LAN9514 ne fait pas que simplement ajouter 2 ports USB, mais améliore notablement la gestion de l'alimentation USB (via un AP2553W6 piloté par le



Le connecteur Jack change de côté et trouve sa place désormais entre le HDMI et l'Ethernet. Le connecteur RCA pour la vidéo composite a disparu, mais le signal est maintenant accessible via le quatrième conducteur du Jack.

SoC afin de réduire les effets indésirables si la consommation de courant approche des limites du composant SMSC) ainsi que la stabilité liée à la présence d'un quartz pour le LAN9512 avec le modèle B (le LAN9514 reçoit son signal d'horloge du SoC à présent).

Notez cependant que ceci ne corrige pas d'éventuels problèmes rencontrés par les utilisateurs faisant un usage intensif des ports USB (streaming vidéo, acquisition de données, RTL-SDR, etc.) puisque la source de ces problèmes n'est pas imputable au hub SMSC, mais au contrôleur USB intégré au BCM2835 et son pilote difficile à améliorer pour les développeurs noyau en l'absence de documentation technique complète (*datasheet*).

5. ALIMENTATION ET AUDIO

Plus techniques et tout aussi intéressants sont les changements apportés au circuit d'alimentation de la carte elle-même. Alors que le modèle B reposait sur un ensemble de régulateurs linéaires (NCP1117-3.3, LP2980-2V5 et NCP1117-1V8), la B+ utilise ce qu'on appelle un convertisseur Buck (*Buck converter*). Il s'agit d'un composant fournissant une alimentation à découpage qui permet d'atteindre une efficacité de l'ordre de 95%. En clair, alors qu'un convertisseur linéaire (LDO) change la tension en dissipant

l'excédant d'énergie en chaleur (qui est perdue donc) un convertisseur Buck utilise un mécanisme de PWM pour découper le courant en petites tranches et, à l'aide de condensateurs, de bobines et de résistance, obtenir la tension souhaitée de manière stable (avec un mécanisme de *feedback*).

Le régulateur unique utilisé sur la B+ transforme ainsi la tension de 5V en 3,3V et en 1,8V, utilisés par le SoC et le LAN9514, de manière bien plus efficace. Il en résulte une consommation globale réduite pour l'ensemble de la carte Raspberry Pi qui passe de 3,5 watt à 3 avec le modèle B+, tout en évitant d'adjoindre un radiateur en cas d'utilisation d'un boîtier

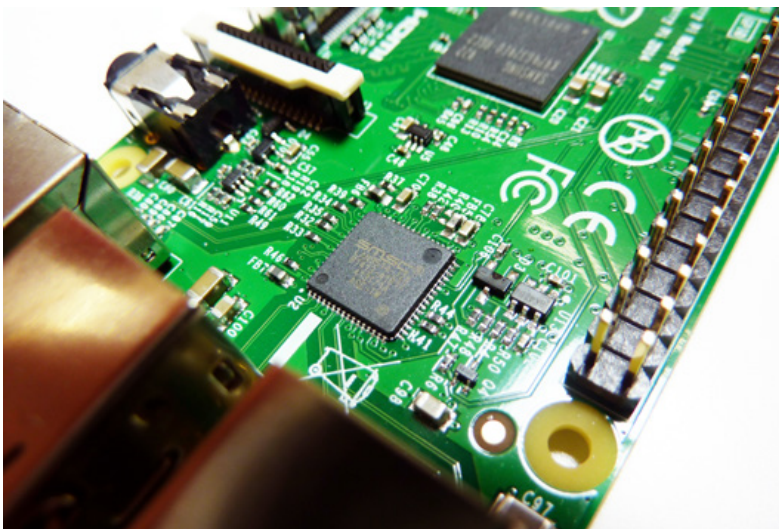
(du moins sur le régulateur, le SoC *overclocké* en ayant toujours besoin).

L'utilisation efficace de l'énergie n'est pas le seul gain des modèles « plus » au niveau de l'alimentation. Une meilleure protection est également à l'ordre du jour. Ainsi la diode Transil permettant une suppression de tensions transitoires et qui est donc une protection contre les surtensions est complétée par l'installation d'un MOSFET faisant office de protection contre l'inversion de polarité. Le *polyfuse* (fusible réarmable) de 1A est maintenant remplacé par une version 2A, un courant maintenant aisément pris en charge par le circuit complet.

Enfin, un dernier composant a été ajouté sur l'alimentation 5V afin d'éviter un reset intégral de la carte (via le déclenchement d'un *polyfuse*) en cas de connexion d'un périphérique USB problématique ou trop gourmand. Ce composant, le AP2331, limite le courant utilisable

et est spécialement conçu pour les connexions à chaud. C'est un problème très présent avec le modèle B et il était devenu de coutume de conseiller la connexion des périphériques USB *avant* la mise en fonction de la Raspberry Pi. Cette protection supplémentaire est complétée par un autre composant, le AP2553W6-7, placé à proximité de la puce SMSC (hub + Ethernet). Celui-ci est destiné à limiter le courant de manière programmable (c'est un *USB power manager*) et est directement contrôlé par le SoC. Une option **max_usb_current** placée dans le **config.txt** sur la partition FAT de la carte microSD permet de contrôler le comportement du composant. Ainsi, en ajoutant une ligne **max_usb_current=1** dans le fichier, on peut passer le courant disponible en sortie USB de 0,6A à 1,2A pour des utilisations spécifiques.

Le hub intégrant l'interface USB/Ethernet SMSC LAN9514 remplace le LAN9512 permettant le passage de 2 à 4 ports USB. Le composant U13 juste à sa droite est un gestionnaire d'alimentation permettant de régler de manière logicielle la puissance délivrée sur les ports.



6. MODÈLE A+

Quelques mots sur le modèle A+ en particulier sont nécessaires étant donné que les évolutions « plus » portent aussi sur ce modèle, mais qu'en vertu de sa déclinaison économique, un certain nombre de points diffèrent. Le modèle A+ profite de l'ensemble des évolutions découlant de la transition B vers B+. Ainsi, le circuit d'alimentation et le réagencement des composants ont été appliqués exactement de la même manière. Si l'on regarde attentivement ce modèle, on constate littéralement que la version A+ n'est presque rien d'autre que le circuit du modèle B+ amputé d'une paire de centimètres.



Seule la partie regroupant les connecteurs USB est différente. En effet, en l'absence de port Ethernet la puce SMSC LAN9514 manque à l'appel, et ce avec le hub USB 4 ports qu'elle intègre. Il en résulte un passage de 4 à un unique port USB qui est connecté directement au SoC Broadcom. Quelques composants passifs ainsi que le AP2553W6-7 placé à proximité du LAN9514 sont donc également absents. Il en résulte que l'utilisation de l'unique port USB de la A+ nécessitera un peu plus d'attention qu'avec une B+. Comme le port est directement lié au SoC, il faut bien comprendre qu'il n'est alors pas question d'y connecter n'importe quoi en se passant d'un hub USB actif (alimenté) supplémentaire.

Bien entendu, la nature même du modèle A+ fait qu'il est en principe peu probable que ce genre de situation survienne. La carte Raspberry Pi A+ doit être envisagée comme une solution embarquée plus que comme un nano-ordinateur. On la destina donc à être la partie « intelligente » d'un montage pour des applications comme la connexion de capteurs, la collecte de données, le contrôle de servomoteur ou encore à des domaines comme la robotique ou la domotique.

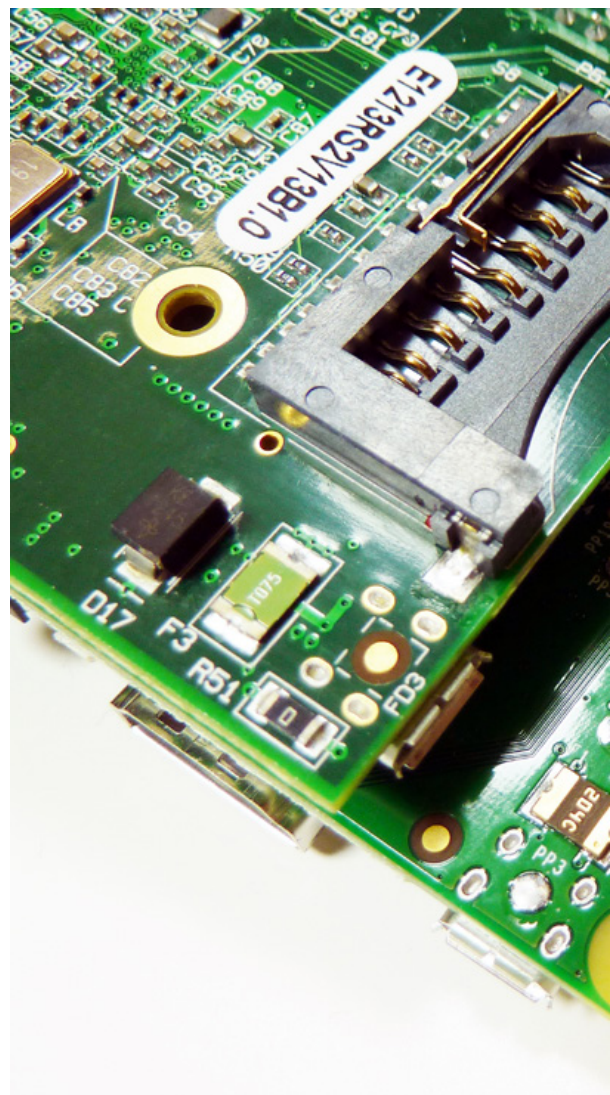
7. LES AUTRES CHANGEMENTS QU'ON AURAIT AIMÉ VOIR

Une évolution d'une plateforme aussi populaire que la Raspberry Pi est généralement une très bonne chose, d'autant plus si la

compatibilité matérielle (brochage) et logicielle (noyau Linux, système et applications) est conservée.

Pour certains constructeurs c'est également l'occasion d'apporter des modifications et des ajouts fonctionnels. Soyons clairs, la mise à jour des cartes A et B vers A+ et B+ est tout à fait appréciable en particulier au niveau de la refonte de l'alimentation. Ce sujet était très discuté concernant les précédentes versions et c'est une très bonne chose. Cependant, deux autres principales critiques étaient généralement faites à l'encontre des Raspberry Pi et celles-ci n'ont malheureusement pas fait l'objet de changements.

Le premier « problème » concerne l'absence de RTC permettant aux Raspberry Pi de conserver une base de temps lors de leur mise hors tension. Certes, le SoC Broadcom contrairement à d'autres modèles chez les concurrents ne contient pas, en interne, de RTC. Généralement, lorsque c'est le cas, le constructeur se contente de mettre à disposition une broche VBAT permettant la connexion d'une pile bouton type CR2032, et dans certains cas plus rares inclus un support pour la pile directement sur la carte. Dans le cas des Raspberry Pi, l'ajout d'une RTC et d'une pile passe par l'utilisation d'un composant particulier comme un DS1337 (cf. *Hackable n°1*, p. 56). Il aurait été appréciable que les modèles B+ et A+, pour quelques dizaines de centimes d'euros supplémentaires, intègrent ce type de composants. Après



tout, la principale raison logique de ne pas placer un composant sur le bus i2c de la Raspberry Pi est de permettre de facilement utiliser les broches dédiées à ce bus en tant que GPIO. Mais les nouveaux modèles ajoutent des GPIO et cette économie n'est plus aussi importante qu'auparavant, selon nous. Cela ne m'aurait pas dérangé de payer mes Raspberry Pi 1 euro de plus (voire 2) si elles



L'emplacement à friction pour la carte SD est remplacé par un emplacement microSD push-pull, plus compact, moins sensible aux vibrations.

avaient intégré un DS1337 ou une autre RTC du même type, avec ou sans support de pile.

L'autre point souvent critiqué concernant les Raspberry Pi est l'absence d'entrée audio. En effet, le connecteur jack placé sur les nouvelles cartes fournit des sorties audio stéréo ainsi que le signal composite vidéo, mais pas d'entrée audio. Cette sortie audio est obtenue via les lignes GPIO40 (droite) et GPIO45 (gauche) utilisées en PWM0 et PWM1. Le SoC cependant dispose d'une interface i2s (*Inter-IC Sound* ou *Integrated Interchip Sound*) permettant de connecter un périphérique compatible. Cette interface était disponible sur le modèle B sous la forme du connecteur P5 (juste à côté des GPIO (P1)) et pouvait être utilisée par des modules comme le RPi-DAC ou encore la carte son Wolfson (WM5102).

Cette interface i2s utilise des broches permettant l'envoi et la réception de données numériques encodées via des lignes PCM_CLK, PCM_FS, PCM_DOUT et PCM_DIN. Le connecteur P5 n'est plus là sur les B+, mais les lignes sont en principe utilisables via les broches standards du connecteur P1. En dehors d'un problème de compatibilité évident pour tout possesseur de carte son Wolfson ou de RPi-DAC qui passe à un modèle B+ (de nouveaux modèles spéciaux B+ sont en préparation), ceci met en lumière un problème nécessitant une solution. Certes, l'utilisation de sortie en PWM pour l'audio est une solution économique permettant de conserver les Raspberry Pi à un prix bas,

mais en sacrifiant par ailleurs une fonctionnalité qui peut être importante dans certaines situations.

L'utilisation de modules ou de périphériques USB audio est donc la seule solution permettant d'avoir une entrée audio (micro ou *line-in*). L'intégration d'un périphérique i2s augmenterait le prix des Raspberry Pi de manière non négligeable (avec des cartes à ce prix, un supplément de 5 euros représente directement un surcoût de 15% à 25%, c'est mathématique). Mais peut-être serait-il alors intéressant de réellement étudier la validité d'un modèle « deluxe » de la Raspberry Pi, appelons-le un modèle C+, intégrant RTC et une interface audio complète sur i2s. Une telle carte pourrait être vendue à 40 ou 50 euros, ce qui la rapprocherait de la gamme de prix d'une BeagleBone Black ou d'une Banana Pi, chose que la fondation Raspberry Pi ne souhaite peut-être pas.

On en conclura donc que les modèles « plus » consistent en une amélioration de l'existant et non une avancée en termes de fonctionnalités. Il reste bien entendu tout à fait intéressant d'opter pour les modèles « plus », mais il serait erroné de considérer ces cartes comme des nouveautés ou des évolutions majeures comme on le fait généralement avec d'autres produits comme les smartphones ou les PC. En remplaçant un modèle B par un B+, vous ne gagnerez rien, mais aurez simplement une Raspberry Pi plus mature. En résumé, inutile donc de se jeter sur ces nouveaux modèles si vous possédez déjà suffisamment de cartes pour vos projets. **DB**



INSTALLATION ET PREMIER DÉMARRAGE DE LA RASPBERRY PI B+

Denis Bodor



Cet article s'adresse aux débutants et aux utilisateurs de machines PC ou Mac n'ayant jamais eu affaire à une carte comme la Raspberry Pi. Nous allons couvrir ici la préparation de la carte microSD sur les trois environnements les plus courants Windows, Mac OS X et GNU/Linux. Puis nous procéderons à la configuration initiale du système.

Notez que si vous êtes déjà utilisateur d'une Raspberry Pi modèle B, rien de ce qui est décrit ici ne sera différent des étapes que vous avez suivies lors de son installation. À moins, bien entendu, que cela ne fasse tellement longtemps que les méthodes actuelles soient complètement différentes, mais dans ce cas, vous n'avez sans doute plus besoin d'explications sur l'installation du système de la Raspberry Pi.

1. CHOIX DU SYSTÈME

Avec une Raspberry Pi vous n'avez pas de disque dur. L'intégralité du système, des outils et des applications se trouve sur une carte microSD. Bien entendu, sans ce support rien ne fonctionne et il n'est pas possible d'installer quoi que ce soit. Il faut donc prévoir et préparer une carte microSD à l'aide d'un PC ou d'un Mac afin d'installer la base du système.

Ce système sera GNU/Linux. Il n'existe pas de Windows ou de Mac OS X pour Raspberry Pi, ni même d'Android. En vous rendant sur <http://www.raspberrypi.org/downloads/> vous trouverez ce qu'on appelle des « images de système d'exploitation ». Il s'agit de fichiers contenant la même chose que ce qui devra se trouver sur la carte microSD (exactement comme si vous aviez un énorme fichier avec toutes les données de votre disque dur). Parmi les images proposées sur le site, nous avons quatre « saveurs » différentes de GNU/Linux :

- Raspbian : le système le plus utilisé, basé sur la distribution Debian GNU/Linux,
- Pidora : un système basé sur la distribution Fedora,
- OpenElec : un système créé spécifiquement pour faire fonctionner un media center appelé Kodi/XBMC,
- Raspbmc : un autre système de media center, utilisant aussi Kodi/XBMC, mais reposant sur la distribution Debian GNU/Linux.

Trois de ces systèmes sont ce qu'on appelle des distributions. Il s'agit d'un ensemble de logiciels rassemblés pour former un système complet, mais permettant d'installer, désinstaller et mettre à jour les éléments qui le compose (les paquets). OpenElec n'est pas une distribution GNU/Linux, il faut plutôt le voir comme un firmware, comme ceux des smartphones. Ce système ne comprend pas de gestionnaire de paquets comme on en trouve avec des distributions sur PC comme Debian, Ubuntu, Fedora, etc.

Enfin, un dernier système est disponible : RISC OS. C'est un système totalement différent de GNU/Linux, Windows, MS/DOS, Mac OS... C'est le système initialement développé par la société Acorn pour son ordinateur *Archimedes* à la fin des années 80. Acorn à l'origine des processeurs ARM largement utilisés aujourd'hui, on peut voir cela comme une sorte d'hommage. Le nombre d'utilisateurs de RISC OS, ou plus exactement des RISC OS est relativement réduit par rapport à d'autres systèmes, y

compris GNU/Linux ou BSD. Le développement du système s'est scindé en 1998 avec la disparition d'Acorn et nous avons aujourd'hui deux principaux projets, l'un est propriétaire et développé par RISCOS Ltd et l'autre est open source, sous le nom RISC OS Open développé par la communauté (sous la forme d'une organisation appelée *RISC OS Open Limited* ou ROOL) et par la société *Castle Technology*. Le système proposé pour la Raspberry Pi est, bien entendu, RISC OS Open.

Nous nous contenterons ici de traiter de Raspbian puisqu'il s'agit du système le plus largement utilisé par les détenteurs de Raspberry Pi et donc celui pour lequel vous trouverez le plus d'informations en ligne pour vous aider si vous avez un problème.

2. LA MÉTHODE SUPER-FACILE : NOOBS

Copier l'image d'une carte SD ou microSD sur un support n'est pas chose facile pour quelqu'un qui n'a pas déjà une certaine expérience en termes de bidouille. En effet, ceci nécessite des outils pouvant accéder directement aux supports de stockage et donc possédant une marge de tolérance aux erreurs excessivement réduite.

Pour simplifier les choses, une solution particulière est disponible au téléchargement sur le site officiel (<http://www.raspberrypi.org/downloads/>). Il ne s'agit pas d'une image, ni même d'une distribution, mais d'un installateur



quasi-automatique appelé NOOBS. Ce nom n'est pas choisi au hasard et cache un jeu de mots car, si NOOBS est officiellement l'acronyme de *New Out Of the Box Software* (« nouveau logiciel tout frais déballé » en gros), le terme signifie également « débutants », *noob* étant l'abréviation de *newbie*, lui-même une contraction de *new boy*, un terme d'origine militaire (ce que dans notre belle langue française on appelle « un bleu » en référence à la tenue des nouveaux soldats au XIX^{ème} siècle (hé oui, on améliore aussi sa culture avec *Hackable*)).

La solution NOOBS consiste à télécharger une archive Zip et à décompresser son contenu sur une carte microSD fraîchement formatée. Une carte de 4Go sera suffisante pour cette opération, mais une taille de 8 Go ou plus est cependant recommandée. Précisons que si vous ne vous sentez pas à l'aise avec ce genre de choses, il est possible de commander une carte microSD (et son adaptateur SD) contenant déjà NOOBS auprès de divers revendeurs de Raspberry Pi. Remarquez cependant que cette méthode NOOBS est très accessible et consiste à simplement copier des fichiers comme vous le feriez sur n'importe quel support externe comme une clé USB.

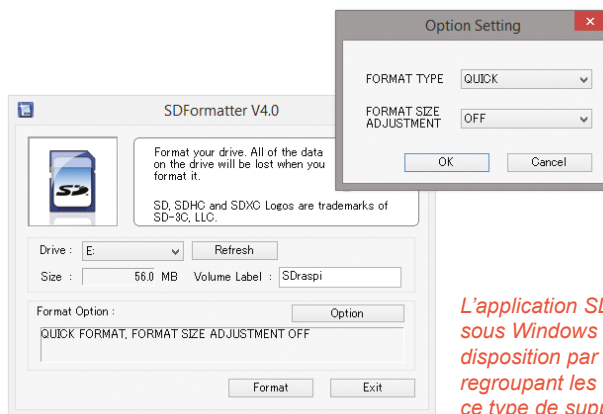
Mais avant de procéder à cette copie, il est nécessaire de formater le support. Dans certains cas, une carte neuve peut contenir des fichiers ou des répertoires. Ceci est également valable si vous utilisez une carte provenant d'un smartphone ou d'un

appareil photo numérique, même si vous la formatez dans l'appareil. Pour formater le support, il est recommandé d'utiliser l'outil mis à disposition par l'association SD (regroupant une grande majorité de fabricants) : le *SD Formatter 4.0* disponible pour Windows et Mac OS X. L'utilitaire est destiné spécifiquement aux supports SD/SDHC/SDXC et annonce formater ces cartes de manière optimale par rapport aux outils normaux intégrés dans votre ordinateur (?!). Le principal avantage cependant pour le débutant réside surtout dans le fait que l'outil ne vous propose de reformater **que** ces supports et vos disques durs n'apparaissent pas dans l'interface, évitant ainsi les erreurs de manipulation. Le formatage « optimal » lui-même n'a presque aucun intérêt puisque la procédure d'installation sur la Raspberry Pi va de toute façon reformater le support avec des outils GNU/Linux, donc sans doute moins optimaux (sarcasmes).

Rendez-vous donc sur <http://www.sdcard.org> à la rubrique **Downloads** et tout en bas choisissez de télécharger la version Windows ou Mac correspondant à votre ordinateur. Procédez à l'installation, puis lancez l'outil après avoir inséré une carte dans votre ordinateur. L'interface, en anglais, vous propose 4 étapes :

- 1 : choisissez la carte. Dans la liste ne devrait apparaître que ce type de support et non les autres disques,
- 2 : choisissez **Quick format** pour une carte neuve ou peu utilisée, ou **Overwrite Format** pour un formatage complet si la carte a un certain vécu,
- 3 : spécifiez un nom quelconque pour facilement reconnaître la carte lors de la phase suivante (genre « RASPI »),
- 4 : cliquez sur le bouton **Format**.

Le formatage rapide est... rapide, mais l'opération peut durer relativement longtemps dans le cas d'un formatage complet puisque chaque zone du support sera formatée. Le temps dépendra alors de la taille de la carte et de ses spécifications techniques (certaines cartes sont plus rapides que d'autres).



L'application SD Formatter 4.0 sous Windows 8 mise à disposition par l'association SD regroupant les constructeurs de ce type de support.

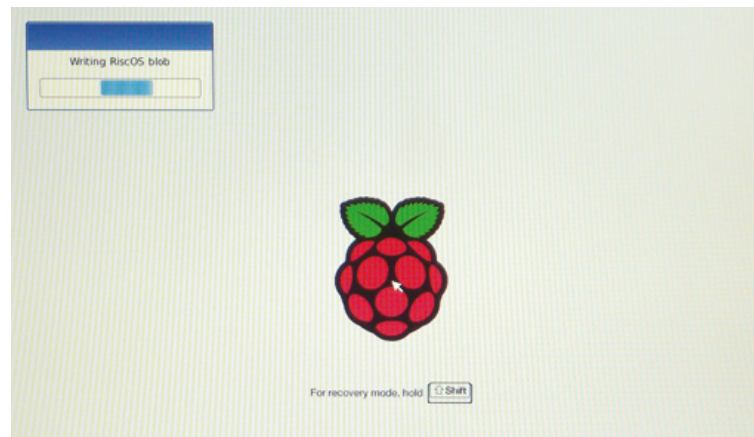
Une fois la carte formatée, rendez-vous sur **www.raspberrypi.org/downloads/** et téléchargez les quelques 738 Mo de NOOBS en cliquant sur **Download ZIP**. Notez que vous trouverez sur la page NOOBS et NOOBS LITE, une version plus petite (~20 Mo), mais qui nécessitera une connexion internet pour la Raspberry Pi lors de l'installation (ce qui n'est donc pas applicable pour un modèle A+). À cette date, la version téléchargeable de NOOBS est la 1.3.10 et prend la forme d'un fichier **NOOBS_v1_3_10.zip**. Une fois le téléchargement terminé, double-cliquez sur le fichier pour le désarchiver (ou l'ouvrir, selon votre système). Vous obtiendrez alors un dossier **NOOBS_v1_3_10** contenant ce qu'il faut copier sur la carte.

Attention : il faut copier le **contenu** du dossier et non le dossier lui-même. En « ouvrant » la carte microSD, vous ne devez pas avoir un répertoire **NOOBS_v1_3_10**, mais un ensemble de fichiers.

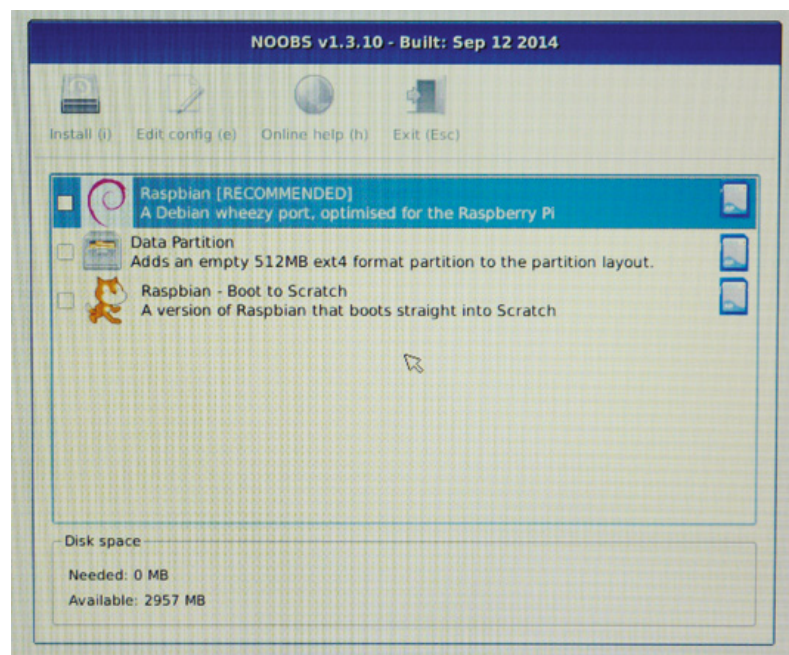
Éjectez ensuite le support et retirez-le du PC ou du Mac pour le glisser dans l'emplacement sur la carte Raspberry Pi. Connectez clavier, souris, écran HDMI, puis l'alimentation via le connecteur USB Micro-B. Le premier signe d'activité sera l'activation des lumières (leds) sur la Raspberry Pi et, selon la vitesse de réponse de votre écran, l'affichage d'un dégradé multicolore.

Le démarrage enchaîne ensuite sur un écran de démarrage permettant au système minimal

actuellement sur la microSD de se charger. Le système utilisé pour l'installateur est RISC OS, mais ne vous inquiétez pas car, ce n'est que l'installateur :

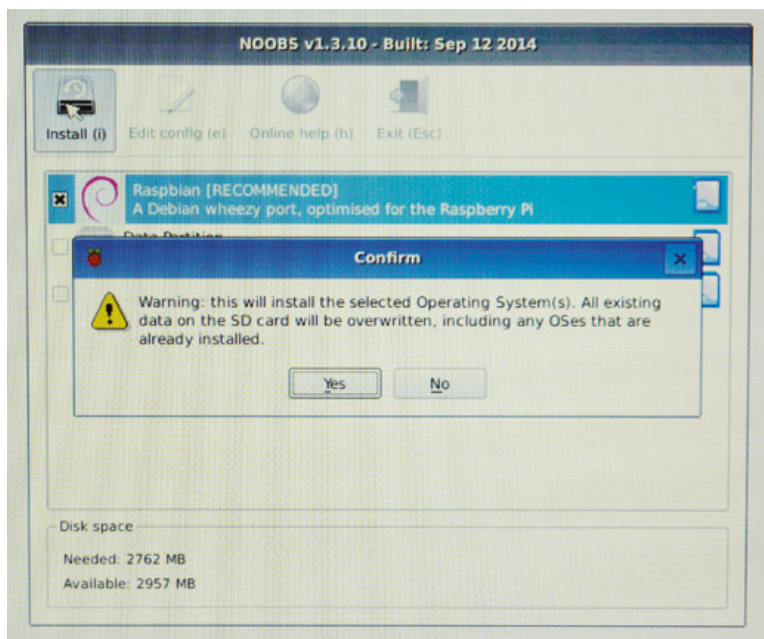


Une fenêtre s'affiche ensuite vous proposant différentes options. Vous pouvez, dans l'ordre, installer Raspbian, ajoutez une partition de données ou installer Raspbian de manière à ce qu'il démarre automatiquement avec l'interface Scratch. Cette dernière est un environnement permettant l'apprentissage de la programmation et il en résultera une installation qui n'est pas sans rappeler les premiers ordinateurs familiaux qui démarraient, eux aussi, directement en proposant un langage pour programmer (le BASIC) :



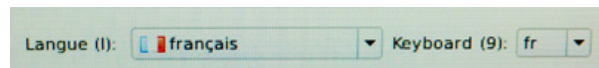


Comme recommandé dans cette sélection, choisissez Raspbian. Il vous sera ensuite demandé de confirmer votre choix :



La procédure d'installation démarre alors et durera un certain temps, soyez patient. Il n'est pas nécessaire de connecter la Raspberry Pi au réseau ou à Internet pour cette installation. La barre de progression ne représente pas un téléchargement, même si sa lenteur et l'affichage d'un nombre de mégaoctets par seconde peuvent le laisser penser. L'installateur utilise les fichiers présents sur la carte microSD pour « confectionner » le système final. Il s'agit donc d'une copie et ceci vous permettra, par la même occasion de juger de la rapidité de votre support microSD :

Comme ceci prend un certain temps, profitez-en pour porter votre attention sur le bas de l'écran. Vous pouvez ici, à l'aide de la souris, choisir une langue. Sélectionnez **français** dans la liste et automatiquement le paramètre pour le clavier passera également en **fr** :

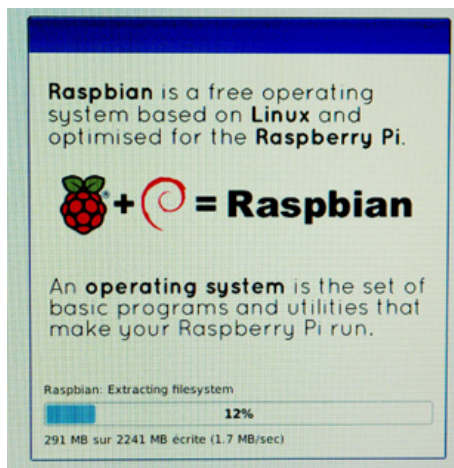


Après de longues minutes (trois cafés et demi environ), l'installation arrive enfin à son terme avec ce sympathique message :



Dès validation, la Raspberry Pi va automatiquement redémarrer. Le contenu de la carte microSD n'est désormais absolument plus celui que vous aviez créé, mais comprend un système GNU/Linux « optimisé » pour la Raspberry Pi. Votre carte microSD contient maintenant Raspbian : voir la figure ci-contre.

La suite des opérations est sensiblement identique à celle découlant d'une installation à partir d'une image Raspbian copiée sur la microSD. Vous pouvez sauter directement à la partie « Configuration lors du premier démarrage » plus loin dans l'article...



3. MÉTHODE AVANCÉE/CLASSIQUE

Cette méthode, depuis longtemps utilisée, consiste à inscrire sur le support l'ensemble du système qui sera utilisé



```

3.594822] usb 1-1.2: New USB device strings: Mfr=0, Product=2, SerialNumber=0
3.606517] usb 1-1.2: Product: USB L/3
3.651513] input: USB L/3 as /devices/platform/bcm2708_usb/usb1/1-1/1.2/1.2.1/i.1/input/input0
3.680153] hid-generic 0003:054F:0007.0001: input,hidraw0: USB HID v1.10 Keyboard [USB L/3] on usb-bcm2708_usb-1.2/input0
3.749185] input: USB L/3 as /devices/platform/bcm2708_usb/usb1/1-1/1.2/1.2.1/i.1/input/input1
3.775065] hid-generic 0003:054F:0007.0002: input,hidraw1: USB HID v1.10 Device [USB L/3] on usb-bcm2708_usb-1.2/input1
3.901262] usb 1-1.4: new low-speed USB device number 5 using dwc_otg
4.027642] usb 1-1.4: New USB device found. idVendor=046d, idProduct=c5a
4.051185] usb 1-1.4: New USB device strings: Mfr=1, Product=2, SerialNumber=0
4.076370] usb 1-1.4: Product: USB Optical Mouse
4.099400] usb 1-1.4: Manufacturer: Logitech
4.123313] input: Logitech USB Optical Mouse as /devices/platform/bcm2708_usb/usb1/1-1/1.4/1.4.1/i.1/input/input2
4.153999] hid-generic 0003:046d:0064.0003: input,hidraw2: USB HID v1.1 Mouse [Logitech USB Optical Mouse] on usb-bcm2708_usb-1.4/input2
... Starting the hotplug events dispatcher: dmdevd[ 4.435823] dmdevd[156]: starting version 175
[ ok ] Synchronizing the initial hotplug events...done.
[... ] Waiting for /dev to be fully populated...[ 6.731344] bcm2708-l2x bcm2708-l2x.0: Failed to create debugfs directory
done.
Starting fake huclock: loading system time.
Tue Sep 9 00:07:06 UTC 2014
[ ok ] Setting preliminary hupnp...done.
[ ok ] Activating uwp...done.
[ 11.599560] EXT4-fs (sdc1h06): re-mounted. Opts: (null)
[... ] Checking root file system...fsck from util-linux 2.20.1
e2fsck 1.42.5 (29-Jul-2012)
root: clean, 81573/105304 files, 59463/735084 blocks
[ 12.001879] EXT4-fs (sdc1h06): re-mounted. Opts: (null)
[warn] Creating compatibility symlink from /etc/ssh to /proc/mounts. ... (warning).
[ ok ] Cleaning up temporary files... /tmp.
[info] Loading kernel module mt-bcm2835.
[ ok ] Activating jss and rd uwp...done.
[... ] Checking file systems...fsck from util-linux 2.20.1
donefsck 3.0.13, 30 Jan 2012, FAT32, LFN
/dev/sdc1h06: 14 files, 15000/120000 clusters
done.
[ ok ] Mounting local filesystems...done.
[ ok ] Activating uwp...done.
[ ok ] Cleaning up temporary files...
[ ok ] Setting up temporary files...
[ ok ] Setting kernel variables...done.
[ ok ] Configuring network interfaces...done.
[ ok ] Cleaning up temporary files....
[ ok ] Setting up dhcpcd...done.
[info] Setting console screen modes.
[info] Skipping fast and hupnp setup. Channelled by console-setup).
[ ok ] Setting up console fast and hupnp...done.
[ ok ] Setting up ifconfig directories... /tmp.X11-unix /tmp.XCB-unix.
[info] Loading dmdevd: 2
[info] Using uwp file-origie concurrent host in realmode 2.
[info] Initializing uwp...
[warn] Kernel locale support, or any other controller not available, not starting uwp... (warning).
[... ] Applying config from /dev/usb_config.json (if it exists):Setting flosser to Saphira based on usb_config.json from HMM08. May take a while
to mount flosser. [info]
Setting flosser to fr-30-07-0 based on usb_config.json from HMM08. May take a while
[ ok ] Mounting interface flosser...this will...done.
[ ok ] Regenerating usb host keys (in background)).

```

par la Raspberry Pi. Si vous avez le moindre doute sur vos compétences techniques, ne tentez pas le diable et rabattez-vous sur NOOBS, plus simple et moins risqué. Le résultat sera le même et vous obtiendrez un système presque parfaitement identique avec l'une ou l'autre méthode (« presque » parce que la méthode NOOBS configure certaines choses à votre place, cf. la partie sur le premier démarrage dans la suite de l'article).

Dans tous les cas, comme avec NOOBS, il est recommandé de

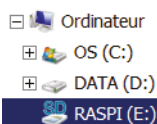
formater la carte avec *SD Formatter 4.0*, ne serait-ce que pour avoir l'opportunité d'en vérifier l'état (formatage complet) et pour lui donner un nom qui vous permettra de facilement la différencier des autres disques présents sur votre ordinateur.

3.1 Sous Windows

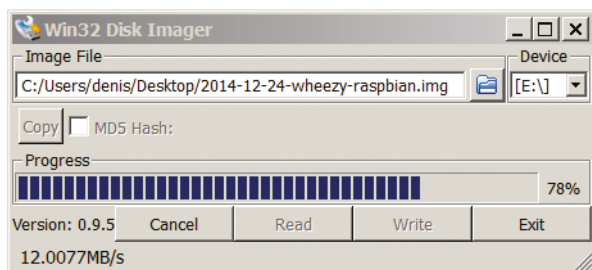
Windows n'offrant pas par défaut d'outils permettant de lire et d'écrire des images de disque, même en ligne de commandes (contrairement à GNU/Linux ou Mac OS X), il sera nécessaire

d'installer l'outil Win32DiskImager. Cette application open source (licence GPL) est spécialement destinée à cet usage qui s'étend au-delà de la simple préparation de cartes SD et microSD pour la Raspberry Pi.

Pointez votre navigateur Web sur <http://sourceforge.net/projects/win32diskimager/> et récupérez directement l'installateur pour, naturellement, exécuter le programme et installer l'outil. Placez ensuite une microSD dans le lecteur et assurez-vous de repérer et mémoriser la lettre associée au lecteur.



Lancez ensuite Win32DiskImager, sélectionnez l'image à écrire ainsi que le lecteur. Normalement, seuls les supports amovibles seront sélectionnables, il y a peu de chance donc que vous choisissiez par accident un disque dur de la machine ou un lecteur CD/DVD (notez que Win32DiskImager ne peut pas écrire sur ce type de support). Il ne vous reste plus qu'à cliquer sur le bouton **Write**, confirmer, à attendre la fin de l'opération et le tour est joué.



3.2 Sous Mac OS X

Nous ne suivrons pas ici les recommandations officielles que nous jugeons un peu trop dangereuses, car reposant sur une connaissance technique concernant l'identification des disques dans Mac OS X. Après tout, on choisit généralement Mac parce que c'est simple et intuitif, il faut donc une solution simple et intuitive (certains choisissent aussi Mac parce que ce sont les seules machines à faire fonctionner à la fois Mac OS X, Windows, GNU/Linux et BSD). Celle que nous avons choisie initialement repose sur l'utilisation de l'appli-

cation *PiWriter* de Arjan van Lent (note pour les bidouilleurs et programmeurs Mac, jetez un œil à ses projets et aux outils utilisés, c'est très intéressant). Cependant, après quelques essais couronnés de succès, PiWriter (ainsi que PiWriter2) a cessé de fonctionner sans message d'erreur (double-clic, lancement et rien). Nous n'avons pas étudié cette voie davantage ou cherché à expliquer le comportement, jugeant cela susceptible de se produire chez vous et donc d'être fort déroutant.

Les autres options sont *RPi-sd card builder* et *ApplePi-Baker*, mais il s'agit d'applications propriétaires.

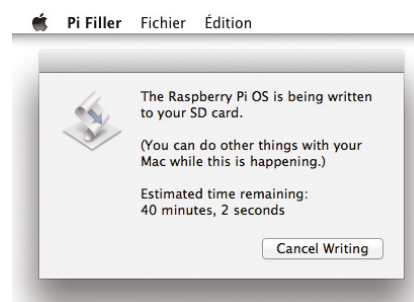
En effet, comme les sources de ces outils ne sont pas disponibles, et donc non-vérifiables par d'autres développeurs, il est légitime de douter de leur comportement sachant, de plus, qu'ils nécessitent des privilèges et permissions

« administrateur » pour fonctionner (« je ne suis pas parano, mais juste au courant »).

Il ne reste alors que *Pi Filler* sous licence WTFPL (je vous laisse le soin de vous renseigner sur celle-ci) et écrit en AppleScript. Vous pourrez l'obtenir sur <http://ivanx.com/raspberrypi/>. Une fois le fichier **PiFiller.zip** téléchargé et désarchivé, vous obtiendrez le script *Pi Filler* qu'il vous suffira de lancer. La première boîte de dialogue vous demandera d'éjecter votre carte SD/microSD si celle-ci est déjà insérée. Vous cliquez ensuite sur **Continue** et pourrez choisir l'image que vous

voulez utiliser. Le choix fait, l'outil vous demandera alors d'insérer la carte, puis de cliquer sur **Continue**. Ce faisant, il se mettra à rechercher la carte, puis doit vous afficher un nouveau message vous signalant qu'il a trouvé quelque chose.

La fenêtre qui apparaît précise le nom de volume utilisé pour la carte que vous avez spécifié à l'étape de formatage. Si tout vous semble correct, cliquez encore une fois sur **Continue**. Un dernier message de confirmation apparaît vous demandant s'il faut effectivement détruire le contenu de la carte, cliquez sur **Erase SD Card**. Votre mot de passe sera alors demandé pour passer en mode administrateur et accéder à la carte. Après saisie et validation, un petit temps de pause de quelques secondes sera marqué avant l'affichage d'une nouvelle fenêtre indiquant la progression de l'opération. Soyez patient, sur notre MacBook Pro de test la durée affichée est de l'ordre de 39 minutes (il y a plus de 3,2 Go de données à écrire sur la carte).



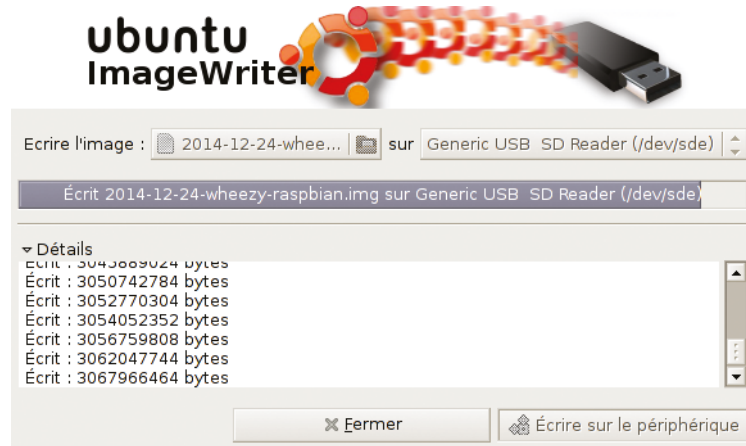
Pi Filler est un outil hyper simple écrit en AppleScript et parfaitement vérifiable contrairement à d'autres utilitaires comme RPi-sd card builder et ApplePi-Baker. C'est une affaire de goût et de paranoïa, mais nous préférons ici éviter des applications propriétaires qui nécessitent des droits administrateur dans Mac OS X.

Pi Filler est un outil relativement basique, mais il fait le travail correctement et ne nécessite pas l'installation d'applications (c'est un simple AppleScript). Si cela vous paraît trop peu évolué, libre à vous de vous tourner vers *RPi-sd card builder* ou *ApplePi-Baker* (ou vers la solution proposée officiellement et utilisant, en partie, la ligne de commandes et **dd**). Encore une fois, personnellement, dans la mesure du possible, je n'accorde aucune confiance et n'utilise aucune application non open source qui requiert un fonctionnement en mode administrateur.

3.3 Sous GNU/Linux

Si vous êtes utilisateur de GNU/Linux, il est fort probable que votre niveau de compétences ne nécessite pas beaucoup d'explications quant à l'écriture d'une image sur un support. Ce type de manipulations, faites avec les privilèges super-utilisateur **root** via **sudo** repose généralement sur la ligne de commandes et en particulier sur l'outil **dd** qu'on utilisera avec la syntaxe **dd if=source of=destination**.

Il existe cependant une alternative moins spartiate : Ubuntu Image Writer. Il s'agit d'une interface écrite en Python et proposant une version très édulcorée des fonctionnalités de **dd** sur lequel il repose (Ubuntu Image Writer n'est en réalité qu'une interface graphique lançant **dd**). La raison d'être initiale de cet outil est la possibilité de produire des clés USB contenant un système comme Ubuntu ou Debian GNU/Linux.



*Ubuntu Image Writer est une interface graphique écrite en Python pour la commande classique **dd**, bien plus facile à utiliser et moins risquée.*

Nous pouvons cependant faire usage de cet utilitaire en l'exécutant en super-utilisateur et en choisissant successivement le fichier image du système Raspbian et un périphérique cible. Il est fort probable que vous deviez « démonter » (et non éjecter) le support afin que **dd** puisse y accéder. Ce faisant, Ubuntu Image Writer sera en mesure d'y accéder et de procéder à l'écriture de données. Notez que l'outil ne vous propose pas, en principe, de choisir un support actuellement utilisé par le système (monté). Vous êtes donc, en principe (encore), à l'abri de fausses manipulations risquant d'écraser les données de votre système en cours d'utilisation. Prenez cependant le temps de bien lire ce que l'interface vous propose pour vous assurer qu'il s'agit bien de la bonne cible.

Les utilisateurs de Debian GNU/Linux peuvent également bénéficier de cet utilitaire à condition de l'installer manuellement depuis un fichier **.deb**. Les utilisateurs d'Ubuntu le trouveront simplement dans la liste des applications disponibles.

Nous vous recommandons cependant d'apprendre à vous servir des outils classiques comme **dd** et de la ligne de commandes. Le stress induit par l'utilisation de telles commandes a le bénéfice de vous rendre très attentif au fonctionnement du système, ce qui est forcément une bonne chose. De plus, la ligne de commandes vous permettra également d'exploiter (et explorer) bien davantage votre Raspberry Pi que n'importe quelle interface graphique...

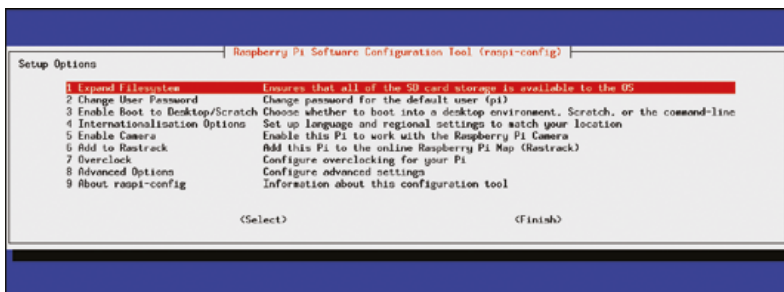
4. CONFIGURATION LORS DU PREMIER DÉMARRAGE

Que vous ayez procédé à l'installation du système avec NOOBS ou par copie de l'image sur le support, le premier démarrage effectif du système vous affichera, après un long défilement de lignes sur fond



noir (ce sont les messages détaillant le démarrage de GNU/Linux), un écran bleu/cyan présentant un certain nombre de menus. Ceux-ci sont destinés à vous permettre de configurer le système pour sa première utilisation.

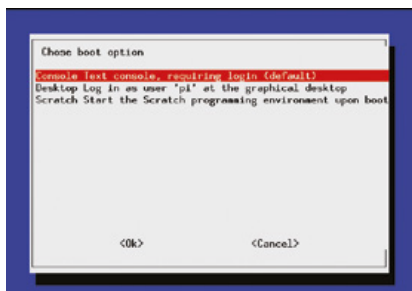
Nous allons parcourir les menus utiles pour configurer votre Raspberry Pi. La navigation se fait dans les menus avec les flèches de direction et les boutons au bas des écrans sont accessibles avec la touche de tabulation :



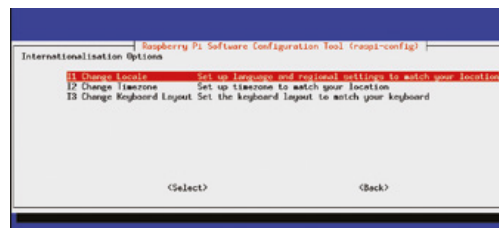
La première chose à faire si vous n'avez pas utilisé NOOBS est d'étendre la taille de l'espace utilisable à la totalité de la carte microSD. En effet, l'image Raspbian ne fait que 3,2 Go et dans l'état actuel, l'ensemble du système fait cette taille, que vous ayez utilisé un support de 4 Go, 8 Go 16 Go ou plus. Avec NOOBS, cette opération n'est pas nécessaire, car elle fait déjà partie du processus d'installation standard. Utilisez ainsi le menu 1 et au bout de quelques instants un message de confirmation apparaîtra :



Nous pouvons maintenant choisir la façon de démarrer la carte Raspberry Pi. Pour l'instant, par défaut, celle-ci propose un environnement en mode texte avec une ligne de commandes. Ceci n'est cependant pas du goût de tout le monde et en particulier si vous découvrez l'environnement sans aucune expérience dans l'utilisation de GNU/Linux. En choisissant le menu 3, vous pourrez alors définir au choix, respectivement, le mode texte, le mode graphique ou le démarrage automatique avec l'environnement de programmation Scratch. Faites votre choix et validez :



De retour au menu principal, nous choisissons le menu 4 concernant l'internationalisation. Là encore, si vous avez utilisé NOOBS et choisi un pays durant l'installation ces paramètres devraient déjà être corrects. Un coup d'œil pour s'en assurer ne coûte cependant pas grand-chose. Le sous-menu qui vous est proposé permet de changer plusieurs éléments en rapport avec la localisation de votre système : les locales qui dictent aux applications vos préférences linguistiques, le fuseau horaire (*timezone*) dans lequel vous vous trouvez et l'agencement du clavier :



Commençons par les locales qui regroupent un ensemble d'éléments autour d'une dénomination codifiée. Dans la longue liste qui se présente à vous, vous pouvez cocher plusieurs locales qui seront alors utilisables dans le système. En France, nous utilisons **fr_FR.UTF-8** (ou historiquement **fr_FR@euro** pour un jeu de caractères ISO-8859-15) avant l'arrivée d'UTF-8). Une petite explication s'impose ici. Contrairement à certains autres systèmes d'exploitation, GNU/Linux est internationalisé. Ceci signifie que l'ensemble du système peut très simplement changer de langue et ceci inclut les messages à l'écran, mais également des choses comme le format de la date par exemple. Il n'y a pas une version française du système

et une autre américaine, allemande, suédoise, etc. Pour la plupart des logiciels, les traductions sont déjà présentes lors de l'installation. L'affichage de ces traductions dépend d'un paramètre du système : les locales.

Vous pouvez très simplement avoir la démonstration de cette fonctionnalité. Si votre système est configuré pour le français et que vous essayez d'invoquer la commande **tar** sans argument, vous obtiendrez un message d'erreur en français :

```
$ tar
tar: Vous devez choisir une option parmi
« -Acdrux » ou « --test-label »
Pour en savoir davantage, faites : « tar
--help » ou « tar --usage ».
```

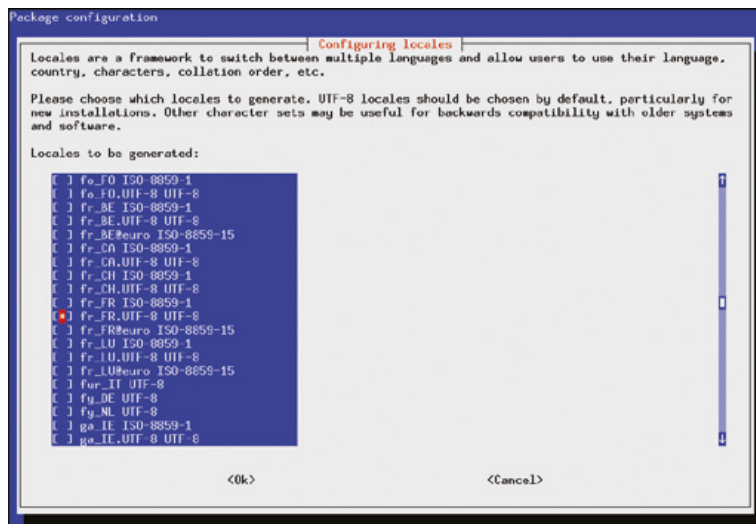
Mais, avec les bonnes locales, rien ne vous empêche de préciser que vous êtes allemand :

```
$ LANG=de_DE.UTF-8 tar
tar: Eine der Optionen „-Acdrux“ oder
„--test-label“ ist notwendig.
„tar --help“ oder „tar --usage“ gibt weitere
Informationen.
```

ou anglais :

```
$ LANG=en_GB.UTF-8 tar
tar: You must specify one of the '-Acdrux'
or '--test-label' options
Try 'tar --help' or 'tar --usage' for more
information.
```

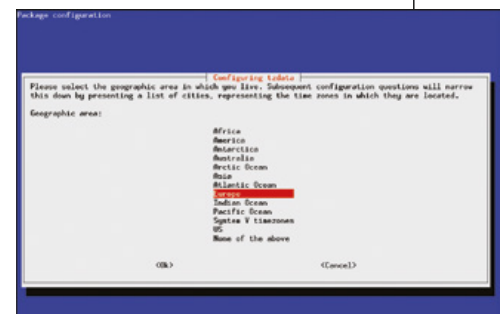
Notez que non seulement le texte est traduit, mais en plus la typographie elle-même est différente (guillemets). Ainsi, choisissez dans la longue liste la ou les locales que vous êtes susceptible de devoir utiliser :



C'est à l'étape suivante que vous pourrez, dans cette sous-sélection choisir celle utilisée par défaut :



Nous passons ensuite à la configuration du fuseau horaire avec tout d'abord un choix de continent :



Puis celui d'une ville. Notez que « Paris » est utilisé pour toute la France, car nous n'avons qu'un seul fuseau horaire pour notre pays contrairement, par exemple, aux USA :

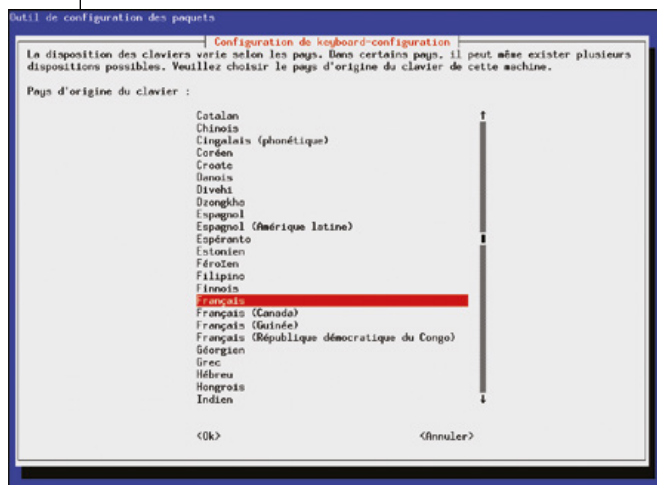




Enfin, est ce sera un peu plus long en termes de succession de menus, nous avons la configuration du clavier. Notez au passage, si vous êtes un utilisateur avancé et que vous procédez à la configuration via SSH, que ceci ne fonctionne QUE si un clavier est effectivement connecté. Le menu retourne automatiquement au départ sans rien faire dans le cas contraire. On vous demandera en premier lieu le modèle de clavier utilisé. « PC générique 105 touches » est le modèle standard, mais il est possible que votre périphérique soit explicitement listé (comme les claviers Apple ou Microsoft par exemple) :

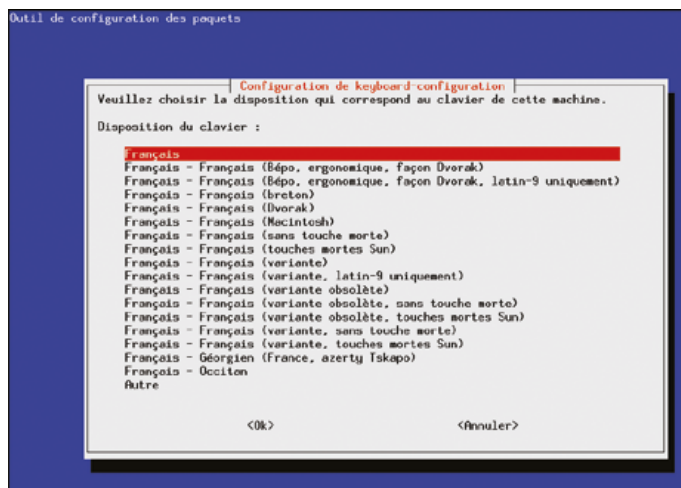


Il existe plusieurs variations de chaque modèle en fonction des pays, vous devrez alors choisir le vôtre dans la liste :

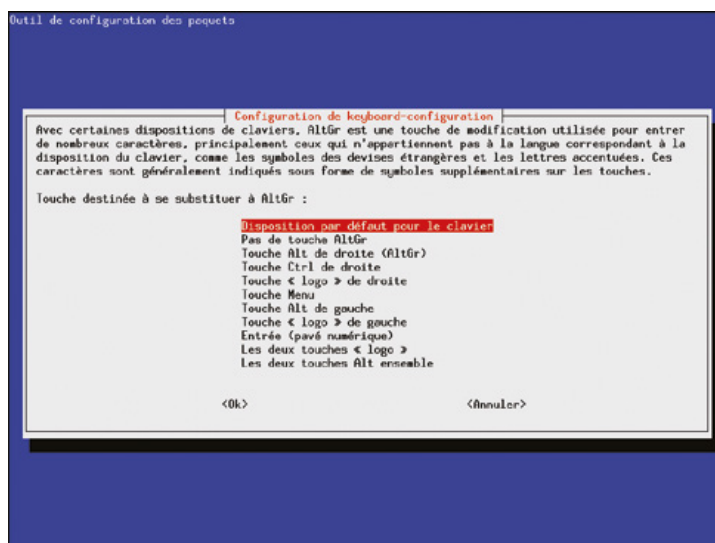


Vous pourrez ensuite choisir la disposition spécifique du clavier avec, par défaut **Français**, mais

sachez qu'il existe des variantes bretonne et occitane par exemple, mais pas alsacienne (zut alors !) :



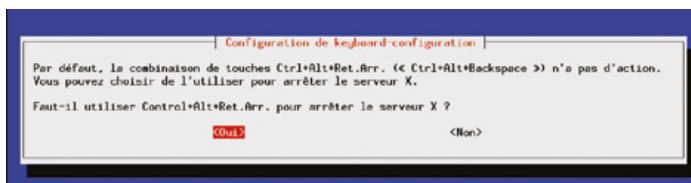
Certains claviers et variantes de claviers ne possèdent pas la touche AltGr (vous savez, celle pour faire un « @ »), la configuration vous propose donc d'en définir une autre pour palier à cela ou laisser celle par défaut :



Certains caractères spéciaux, dans certains pays ou pour certaines utilisations particulières, nécessitent l'utilisation d'une touche « compose » qui n'est sans doute pas présente sur votre clavier français AZERTY standard. Vous pouvez, si vous en avez l'utilité, définir une telle touche :



Enfin, dernier point pour le clavier, il est traditionnellement d'usage dans le monde UNIX d'avoir à disposition un raccourci permettant de tuer le serveur graphique X (le système qui gère l'affichage dans l'interface graphique). Ce raccourci, CTRL+ALT+Backspace est de moins en moins configuré par défaut et vous pouvez ici choisir si oui ou non vous désirez l'avoir à disposition :



La configuration de base est maintenant terminée et vous pouvez utiliser la touche de tabulation et les flèches pour valider *Finish*. Comme vous avez procédé à d'importants changements, il sera nécessaire de redémarrer le système :



Validez simplement le message et votre Raspberry Pi va redémarrer une dernière fois pour appliquer les modifications. Notez que dans la suite de l'utilisation de la carte et du système, vous pouvez à tout moment relancer ce système de configuration en utilisant les commandes **sudo raspi-config**. Seront alors présentés, à nouveau à l'écran, les mêmes menus et les mêmes options.

Nous n'avons pas couvert ici des choses spécifiques et techniques se trouvant dans le menu 8. Ici, vous pourrez configurer différents paramètres d'affichage, de sortie audio ou encore l'activation de certaines fonctionnalités de la carte comme les bus SPI et I2C. Ceci relève d'une utilisation plus avancée que le premier démarrage d'une Raspberry Pi et une exploration de ce qu'elle contient...

POUR CONCLURE

Cet article ne se veut pas être un guide complet pour appréhender tous les mystères de la Raspberry Pi, mais uniquement une introduction pour que vous puissiez mettre le pied à l'étrier. Nous n'avons fait qu'effleurer une partie de la surface de l'océan de fonctionnalités et de possibilités qu'offre une carte comme la Raspberry Pi. Si vous êtes un utilisateur GNU/Linux vous retrouverez rapidement vos marques. Si vous êtes un utilisateur expert, coutumier de la ligne de commandes et du shell Windows (ou Mac OS X) ainsi que de tous les aspects techniques d'une machine PC/Mac, prendre en main plus avant la carte ne vous demandera pas beaucoup d'effort. GNU/Linux a été créé et est maintenu par une communauté de développeurs, d'administrateurs et d'utilisateurs faisant généralement preuve de bon sens et d'une implacable logique (la plupart du temps). Si vous savez comment fonctionne un ordinateur, vous n'aurez pas trop de problèmes. La logique est celle de l'ingénierie informatique.

Si vous êtes novice et n'avez pas l'habitude de trifouiller là où il est généralement recommandé de faire appel à « votre administrateur système », choisissez de démarrer la Raspberry Pi en mode graphique. Ceci vous permettra de découvrir l'environnement tout en obtenant rapidement des résultats. Vous aurez ensuite tout loisir de vous pencher sur la ligne de commandes et les méandres du système. Sachez que, tant que vous ne touchez pas à l'aspect matériel (brancher des modules, etc.), vous ne pouvez rien casser. Au pire, le système sera inutilisable et il vous suffira d'effacer la carte mémoire pour repartir à zéro avec un système tout neuf en reproduisant ce qui est détaillé ici. **DB**



TESTEUR DE PILES RECHARGEABLES

Thierry Monin



Les piles rechargeables présentent une alternative très économique et durable par rapport aux piles alcalines. Pour autant que nous en prenions soin en tenant compte de leurs caractéristiques, elles alimenteront nos appareils énergivores durant de nombreux mois. Quelques composants électroniques et un Arduino UNO pour superviser le tout nous permettront non seulement de mieux comprendre le fonctionnement de ces batteries, mais aussi de tester et évaluer leur fiabilité.

1. AVANT-PROPOS

En milieu de saison automnale, quand je commence à nouveau les entraînements de jogging ou randonnées de nuit, je charge « à fond » les accus de ma lampe frontale pour courir avec une bonne visibilité et sécurité. Dans mon cas, l'avantage des piles rechargeables est qu'elles offrent une luminosité plus forte et surtout constante dans le temps. Sauf qu'au milieu de ma première sortie, je me suis retrouvé « dans le noir » après quelques minutes seulement. Victime d'une batterie défectueuse ou mal chargée, mais aussi d'une caractéristique des accus : leur faible résistance interne... Contrairement aux alcalines dont la luminosité diminue progressivement (résistance interne plus élevée qui augmente en cours de décharge), la résistance interne est (très) faible, et une fois déchargée, la tension et, en conséquence le courant, chutent rapidement. Là où une pile alcaline traditionnelle annonce qu'elle faiblit, une rechargeable s'éteint.

Au cours des années, de nombreux accus s'accumulent dans nos tiroirs. Dans mon cas, plus d'une quarantaine de piles rechargeables de marques et de capacités différentes retrouvées dans des jouets, manettes de consoles de jeu, postes de radio (trans)portables et autres appareils grands consommateurs d'énergie. Un tri s'impose afin de jeter, ou plus exactement d'envoyer en filière de recyclage, les éléments qui ne répondent plus à nos attentes, ou qui sont tout bonnement défectueux.

Si un premier tri nous permet d'isoler rapidement les batteries qui présentent des traces de fuites d'électrolytes ou autres détériorations apparentes, il sera plus complexe de vérifier si les éléments apparemment intacts sont réellement fonctionnels. En effet, le simple fait de mesurer la tension aux bornes d'une pile – qu'elle soit alcaline ou rechargeable – est tout à fait inutile sauf dans le cas d'une pile alcaline dont la tension serait si faible qu'il n'y aurait aucun doute sur son état de décharge.

Première règle importante : la tension aux bornes d'une pile doit toujours être mesurée avec une charge appliquée à ses bornes.

2. GÉNÉRATEUR DE TENSION OU DE COURANT

Au quotidien, nous utilisons des alimentations ou des chargeurs pour faire fonctionner nos équipements électroniques. Les alimentations de laboratoire peuvent fournir des tensions continues pour alimenter nos montages. Ces tensions sont réglables sur une plage de 0 à 12V voire même de 0 à 30V, tensions indispensables pour certains projets. Pour nos projets d'électronique purement numérique, nous utiliserons une alimentation stabilisée de 5V ou de 3,3V selon les cas.

En électricité, nous pouvons définir ces générateurs :

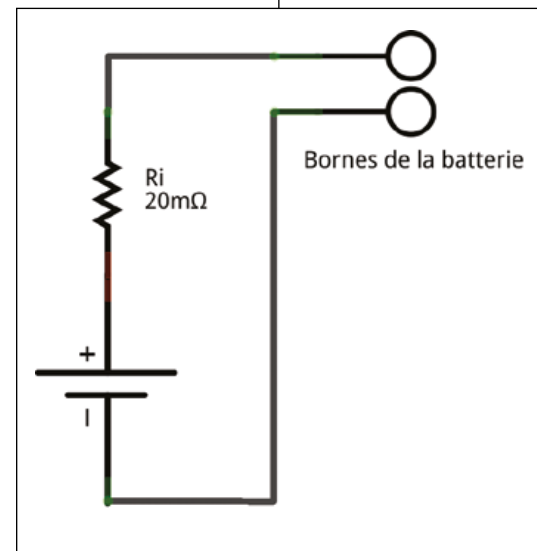
Générateur idéal de tension

Par définition, un générateur de tension idéal est un élément – ou dipôle – qui maintient sa tension constante, quel que soit le courant qu'il débite. La résistance interne d'un tel générateur est nulle et l'intensité de son courant de court-circuit est théoriquement infinie.

Générateur idéal de courant

Ce générateur fournit un courant constant, quelle que soit la tension à ses bornes. En cas de court circuit, son courant est égal au courant qu'il fournit, quelle que soit sa charge.

Figure 1 : Schéma interne d'une batterie rechargeable. Générateur idéal de tension avec une R_i en série.





Une pile – ou une batterie – devrait se rapprocher d'un générateur idéal de tension. Quelques mesures avec un multimètre mettront en évidence que nous sommes loin d'un tel générateur. En effet, lorsque nous faisons varier la charge aux bornes d'une pile ou d'un accu, nous constatons inmanquablement que la tension chute quand la résistance de la charge diminue – c'est-à-dire que le courant demandé augmente. Cela met en évidence l'existence d'une résistance interne (« R_i ») au sein même de la batterie ou de la pile alcaline (Figure 1).

La valeur de cette résistance interne se situe aux alentours de quelques dizaines de milliohms pour un accu et de quelques centaines de milliohms pour une pile alcaline.

Concrètement, la résistance interne d'un accu est de l'ordre de 10 à 20 fois inférieure à celle d'une pile alcaline. Ce qui explique pourquoi la tension aux bornes d'un accu est beaucoup plus stable au cours du temps, mais aussi pourquoi son courant de court-circuit est beaucoup plus élevé ! Intensité de courant de court-circuit très élevée qui peut l'endommager ou le détruire. Un risque de surchauffe, d'incendie ou d'explosion existe. Dans les montages qui suivent, nous vous invitons à la prudence et au soin dans la réalisation des schémas.

Pour plus d'informations sur les valeurs réelles des résistances internes, vous trouverez les spécifications techniques des accus sur les sites web des fabricants.

Tableau de comparaison pile / accu :

	Pile Alcaline	Pile rechargeable
Tension nominale	1,5V	1,2V
Résistance interne	0,2 Ohms	0,02 Ohms
Cycles recharges	0	500 à 1000 fois

Capacité des batteries

La capacité nominale des batteries, représentée par la lettre C, s'exprime en milliampères heure. Par exemple, une batterie de 2000 mAh pourrait théoriquement alimenter un appareil qui consomme 100 mA pendant 20h, ou 200 mA pendant 10h et ainsi de suite.

Il ne faut pas confondre les notions de capacité d'un accumulateur, de puissance électrique, ou d'énergie consommée.

La **capacité** d'un accumulateur « C » est exprimée en Ah (ampères heure). Elle représente la quantité de courant

qu'une batterie complètement chargée peut restituer au cours de sa décharge complète.

La **puissance** électrique « P » est exprimée en Watts. La puissance représente la quantité d'énergie échangée par unité de temps (pour 1 seconde).

L'**énergie** consommée est exprimée en Watts heures, ou en kilowatts heures. Par exemple, un appareil électrique d'une puissance de 1 kW qui fonctionne trois heures aura consommé en tout 3 kW-h.

3. CHARGE...

Avant toute utilisation, il faut charger les batteries. La charge consiste à faire circuler un courant continu en sens inverse dans l'accumulateur. Pour un cycle de charge normal, le courant de charge doit être limité à environ 10% de la capacité de l'accumulateur pour une durée de 16 heures. Concrètement, pour une batterie de 2500 mAh, le courant de charge sera de 250 mA et la durée de 16h. Des cycles de charges plus courts, avec des courants plus forts, sont possibles, mais ils peuvent dégrader les performances et réduire la durée de vie des accus.

Différentes technologies plus ou moins complexes existent pour les chargeurs : régulation en tension, en courant, découpage, impulsions... Les chargeurs les plus évolués sont dotés de microcontrôleurs, de sondes de température. Il est important d'arrêter le courant en fin de charge pour éviter d'endommager les accumulateurs.

La fin de charge peut être déterminée par une minuterie, par la température des accus, ou encore les variations de la tension aux bornes des accus en fin de charge.

La plupart des chargeurs ou testeurs d'accus du commerce imposent de placer les batteries par paire(s). C'est une généralité pour les chargeurs d'entrée et de milieu de gamme. Les rares modèles qui permettent de tester ou de charger individuellement une batterie sont vendus à des prix prohibitifs. Si l'avantage de charger les accus par paire est avant tout économique – en effet, la fabrication d'un chargeur pour quatre batteries en série « deux par deux » est moins coûteuse que la fabrication d'un chargeur de quatre circuits d'une seule batterie. Il faut deux fois moins de composants électroniques pour un tel appareil et le fait de brancher deux accus en série permet de « charger » les deux en série sur 2,4 V au lieu de 1,2 V, ce qui est un peu moins complexe pour la gestion de l'alimentation.

Les inconvénients de ces chargeurs ou testeurs sont nombreux. Il faut brancher deux piles strictement identiques – en capacité et marque – par circuit. Il faut aussi que ces batteries soient dans un niveau de décharge similaire.

En effet, si le niveau de (dé) charge des deux batteries est très différent, par exemple si nous mettons en série deux accus identiques, l'un complètement déchargé et l'autre au contraire presque complètement chargé, le chargeur va fournir un courant qui va traverser les deux accus.

Comme ils sont en série, c'est la même intensité de courant qui traverse les deux éléments. Et quel sera le résultat de ce cycle de charge ? Il sera mauvais ! L'accu complètement à plat sera à peine chargé et celui qui était déjà chargé risque d'être détérioré par ce courant de charge important et inutile. Ce type d'opération risque à terme de réduire la durée de vie des accus.

Autre inconvénient : certains appareils utilisent un nombre impair de piles. C'est le cas de la lampe frontale donnée en exemple au début de cet article qui accepte 3 piles micro LR03 AAA. Pour de tels appareils, nous achetons les accus par 4, 6 ou 8 pièces et les utilisons en alternance. Cela pose un problème parce qu'au final on se retrouve avec 3 accus presque complètement déchargés qui traînent quelques semaines dans un tiroir et 3 ou 5 autres accus dont nous sommes en fin de compte incertains de l'état de charge et en conséquence de la fiabilité que nous pouvons leur accorder.

Avec de tels chargeurs, si l'un des deux accus est défectueux, il sera plus difficile de l'isoler rapidement.

Nous avons suffisamment de bonnes raisons pour nous pencher sur le fonctionnement des accus en général et sur leur décharge en particulier.

Attention : la charge ne s'applique qu'aux accumulateurs. Il ne faut en aucun cas essayer de recharger une pile alcaline !

4. ... ET DÉCHARGE

Lorsqu'elle est utilisée dans un appareil, la batterie restitue l'énergie qu'elle a emmagasinée et se décharge plus ou moins rapidement en fonction de l'intensité du courant qui lui est demandé. Un phénomène d'auto-décharge se produit lorsque les batteries ne sont pas utilisées durant une longue période. Les batteries stockées et inutilisées doivent être rechargées régulièrement.

Les anciens accumulateurs NiCd – « Nickel - Cadmium » – souffraient d'un effet mémoire. La capacité de ce type de batterie diminuait significativement si on rechargeait la batterie alors qu'elle n'était pas complètement déchargée. Les accumulateurs récents – par exemple NiMH – peuvent toujours souffrir de ce problème, mais dans des proportions bien moindres.

La plupart des fabricants qui publient les spécifications techniques de leurs produits donnent les courbes de décharge à une température donnée pour des courants 10 % ou de



20 % de la capacité de l'accumulateur. Ces courants correspondent à une utilisation relativement intensive des batteries.

Exemples pour une batterie de 2500 mAh :

- Courant de 0,1 C = 250 mA et temps de décharge environ 10 heures.
- Courant de 0,2 C = 500 mA et temps de décharge environ 5 heures.

Pour nos tests, nous travaillerons dans cette fourchette de valeurs. Tester avec des courants plus faibles prendrait trop de temps, et tester avec des courants plus forts nous éloignerait de la réalité d'utilisation.

La température (°C) est un facteur qui influence considérablement le fonctionnement d'une batterie. Quand la température diminue, la résistance interne de la batterie augmente et les performances se dégradent. Ce paramètre température prend toute son importance pour les projets qui fonctionnent à l'extérieur. Pour mettre en évidence cette dépendance à la température, il suffit de stocker un accu quelques heures dans un endroit frais, puis de le brancher sur un des montages décrits ci-après dans une pièce plus chaude. Pour plus de précision, on peut définir un intervalle de mesure à 10 ou 30 secondes.

Attention, bien évidemment, à rester dans les normes de températures de stockage définies par les fabricants. Mettre une batterie au surgélateur n'est définitivement pas une bonne idée.

La température d'une batterie qui fournit un courant augmente. Plus le courant demandé est important et plus cette augmentation de température est perceptible.

5. MONITORING DE DÉCHARGE

Après ces quelques notions théoriques, nous pouvons passer aux tests de fonctionnement dans le monde réel.

Avec un Arduino, une platine de montage « bread-board », quelques composants bien choisis et une bonne calculatrice, nous pouvons étudier et réaliser différents montages de complexité variée qui pourront répondre aux questions que nous nous posons sur l'état et le fonctionnement de nos batteries.

Tous ces montages pourront par la suite s'adapter à nos besoins ou à nos envies de satisfaire notre curiosité. En changeant les valeurs de quelques résistances – composants bon

marché par excellence –, nous pourrions adapter ces schémas à d'autres contraintes.

6. PROCÉDURE DE TEST

Nous devons définir une procédure bien précise pour tester objectivement l'état des accus.

- Classer les batteries rechargeables par marque, modèle et capacités. Quelques étiquettes autocollantes pour les numéroter peuvent se montrer utiles pour assurer le suivi.
- Décharger toutes les batteries individuellement à l'aide de l'un des schémas expliqués plus loin.
- Recharger complètement les batteries. En suivant bien entendu les instructions du manuel utilisateur de votre chargeur.

La décharge préliminaire est importante pour s'assurer d'un niveau de décharge suffisant avant de placer les piles dans un chargeur.

Plusieurs approches sont envisageables pour la surveillance de la décharge et l'évaluation de la capacité d'un accu avec Arduino.

- L'Arduino effectue tous les traitements de données : la mesure des tensions, le calcul des courants et l'évaluation de la capacité des batteries. Le résultat de ces mesures est transmis en fin de test sur le port moniteur série, ou sur un afficheur LCD.

- L'Arduino prend en charge uniquement la mesure des tensions sur ses entrées analogiques et transmet ces mesures à intervalles réguliers – par exemple toutes les minutes – sur le port moniteur série.

La première solution présente l'avantage d'un montage totalement indépendant et autonome. Le prix à payer sera un croquis Arduino plus complexe qui nécessitera plus de temps de développement et de maintenance en cas de modification ou de correction.

La seconde solution nous fournira un journal/historique de mesures simple que nous pourrions traiter par la suite dans d'autres logiciels. Sur la base des données reçues, nous pourrions générer des courbes et calculer la capacité de l'accu dans LibreOffice. Un programme écrit en Python, en PHP ou tout autre langage pourrait tout aussi bien reprendre ces données pour des traitements plus ciblés.

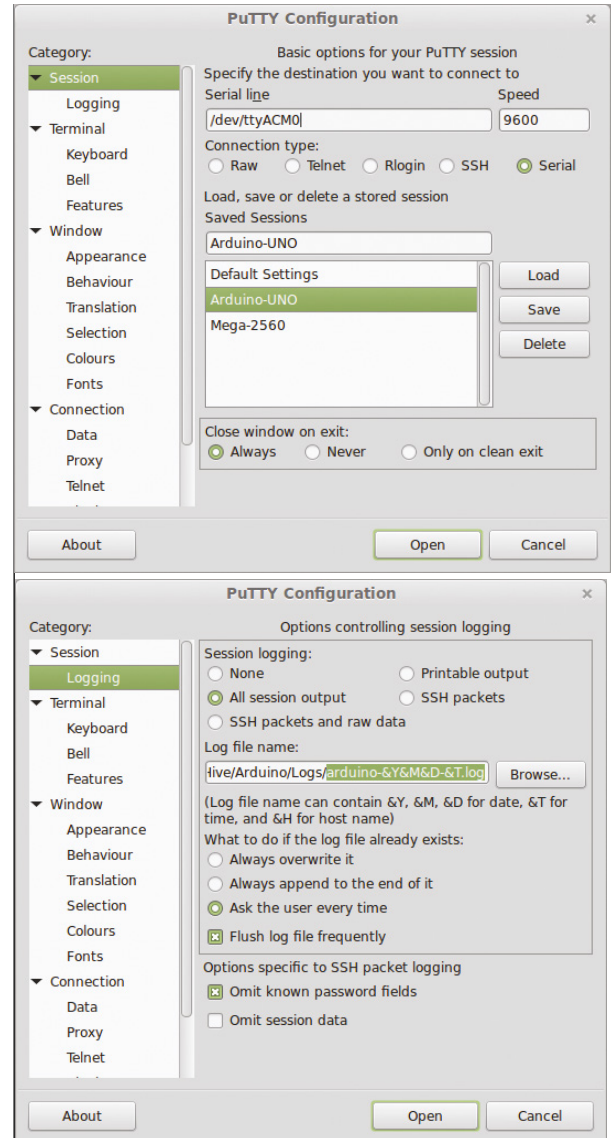
Ce principe de collecte d'information pure, sans calculs à la source, pourra s'adapter facilement à d'autres projets. Par exemple, nous pourrions utiliser un capteur de température pour mesurer la température d'une pièce en fonction de l'heure et adapter les plages de fonctionnement d'un thermostat d'ambiance sur la base de ces courbes.

En raison de sa souplesse, cette seconde approche sera favorisée pour la suite. L'Arduino transmettra la valeur des tensions en entrée sur le port moniteur série à intervalles réguliers et nous utiliserons la fonction de trace de l'émulateur de terminal PuTTY pour sauver dans un fichier les données reçues du port série de l'Arduino.

Nous allons commencer par installer les outils nécessaires. Pour une distribution Linux basée sur Debian, nous pouvons très facilement installer les paquets nécessaires (*NDLR : PuTTY est également disponible gratuitement pour Windows, les mêmes explications peuvent donc s'appliquer avec ce système. Pour Mac OS, vous pouvez utiliser CoolTerm, par exemple*).

```
$ sudo apt-get install arduino putty
```

Une fois les paquets et leurs dépendances installés, nous pouvons brancher l'Arduino, lancer l'environnement de développement - ou « IDE » - et vérifier si tout fonctionne bien. Au passage, notez le device du port série/USB sur lequel l'Arduino est branché, nous aurons besoin de cette information pour créer un profil dans



Configuration de PuTTY.



PuTTY. Sous Linux, ce port série pour les Arduino récents sera habituellement `/dev/ttyACM0`.

PuTTY supporte différents protocoles – ssh, telnet, rlogin, ... – et permet aussi d'établir des connexions séries. C'est cette dernière fonctionnalité que nous allons utiliser pour recevoir les données de l'Arduino.

Voici la procédure de configuration de PuTTY :

- Sélectionner le type de connexion « série ».
- Encoder le nom du port série auquel l'Arduino est connecté.
- Vitesse de transfert des données 9600 bauds.
- Dans **Logging** : Sélectionner **All session output** et spécifier un répertoire et un nom pour le fichier de traces. Par exemple : `accu-&Y-&M-&D_&T.log`. Un nouveau fichier avec date et heure sera créé à chaque ouverture de ce profil.
- Cocher la case **Implicit CR in every LF** dans terminal.
- Revenir au premier onglet **Session**, donner un nom au profil et le sauvegarder.

Important : La connexion série doit être maintenue tout au long du cycle de monitoring. Si on interrompt la connexion série et qu'on la rétablit ensuite, la conséquence en sera un redémarrage complet de l'Arduino.

7. PRINCIPE GÉNÉRAL

Pour évaluer la capacité d'une batterie rechargeable, une charge résistive sera branchée à ses bornes. L'évolution de la tension de la batterie sera mesurée à intervalles réguliers tout au long de la décharge. Pour cette mesure, une entrée analogique de la platine Arduino sera utilisée. Le croquis écrira le résultat de ces mesures de tensions sur la console série. Ces données seront « capturées » vers un fichier texte au format CSV. Pour le traitement et l'analyse des mesures, le CSV nous ouvre la possibilité de charger les données dans LibreOffice Calc pour, par exemple, calculer la capacité ou afficher une courbe de la tension en fonction du temps de décharge.

L'Arduino UNO dispose de 6 entrées analogiques. Nous pouvons donc, en reproduisant à l'identique les schémas, tester simultanément jusqu'à 6 batteries.

8. CHARGE RÉSISTIVE

Quelques calculs préalables seront nécessaires pour déterminer la valeur et la puissance de la résistance de décharge. La batterie doit fournir un courant entre 0,1 C et 0,2 C. La loi d'Ohm établit la relation entre le courant « I » la tension « U » et la résistance « R » (cf. article sur la loi d'Ohm dans le n°1 de *Hackable*).

$I = U / R$ qu'on peut mettre sous la forme $R = U / I$ pour calculer la résistance.

Pour une batterie 2500 mAh :

Courant de décharge à 0,2C = $2500 * 0,2 = 500$ mA.

Pour obtenir une valeur de résistance en Ohms, il faut obligatoirement exprimer la tension en Volts et le courant en Ampères.

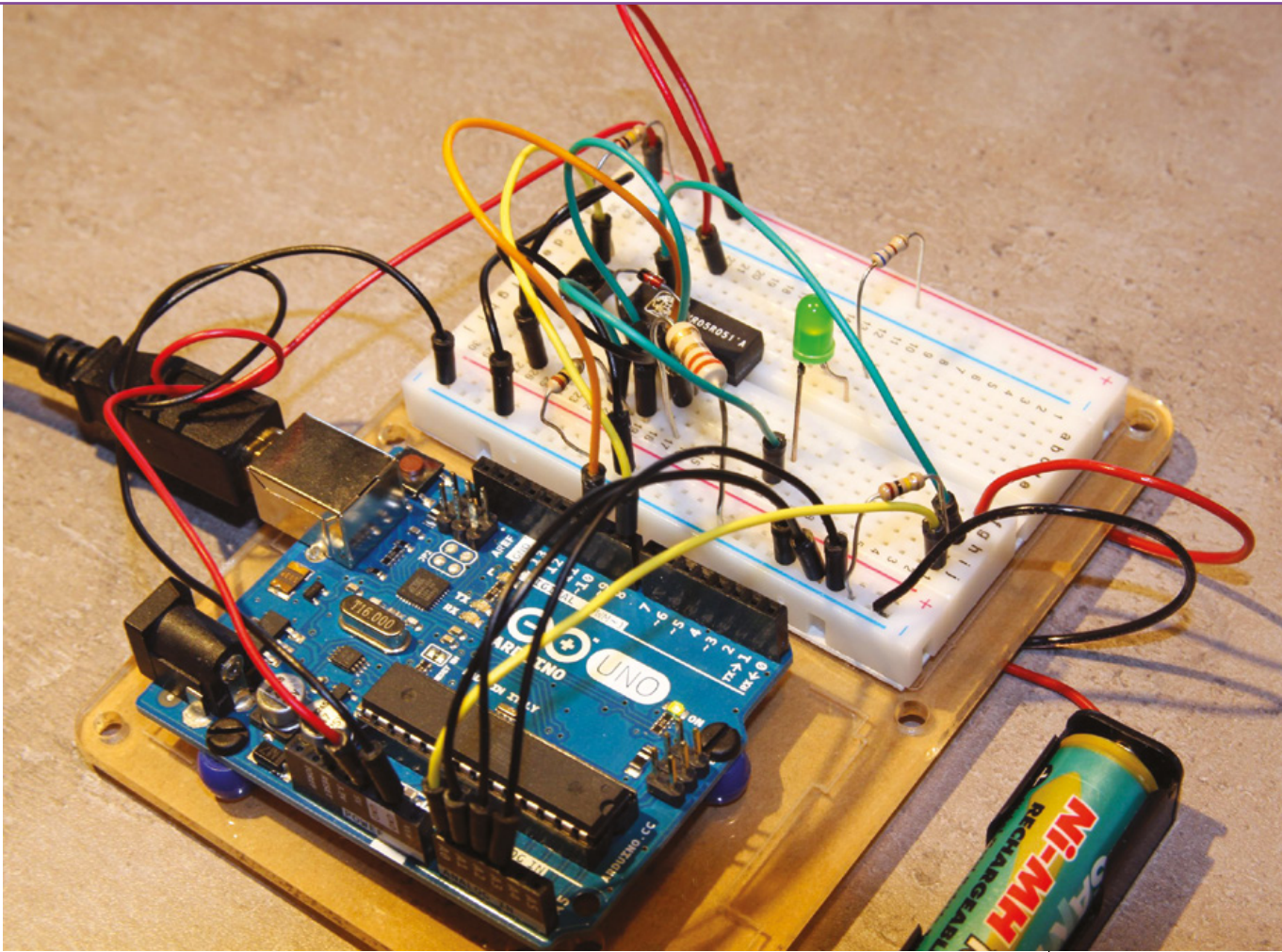
$$R = 1,2 \text{ V} / 0,5 \text{ A}$$

$$R = 2,4 \text{ Ohms}$$

En électronique grand public, on trouve deux normes principales qui définissent les valeurs des résistances. Les normes E12 et E24 (cf. liens). Les résistances de la série E12 ont habituellement une précision de +/- 10% et celles de la série E24 de +/- 5%. La série E12 est un sous-ensemble de la série E24.

Si nous trouvons bien une résistance de 2,4 ohms dans la série E24, par contre cette valeur n'existe pas dans la gamme E12. Les valeurs E12 les plus proches sont 2,2 ohms ou 2,7 ohms.

N'oublions pas dans le cas d'une résistance plus faible, le



courant sera plus important et la décharge de la batterie prendra moins de temps.

Nous avons donc plusieurs valeurs de résistance pour notre montage. Au choix 2,2, 2,4 ou 2,7 ohms.

La résistance branchée sur la batterie va donc « consommer » de l'énergie. Cette consommation d'énergie va se dissiper sous forme de chaleur dans la résistance, mais aussi dans une moindre mesure dans la résistance interne de la batterie. Une batterie qui donne un courant important chauffe. La puissance dissipée est proportionnelle au carré du courant qui traverse la résistance.

$$P = U \cdot I$$

$$P = R \cdot I^2$$

Nous devons calculer la puissance qui sera dissipée dans la résistance. Par sécurité, nous prenons une tension de 1,5V pour les calculs. Cette valeur tient compte du fait qu'un accu bien chargé peut fournir une tension supérieure à 1,2V pendant quelques minutes. Sans oublier que nous pourrions aussi tester des piles alcalines.

1) Pour 2,2 ohms

$$I = 1,5 / 2,2 = 0,682 \text{ A}$$

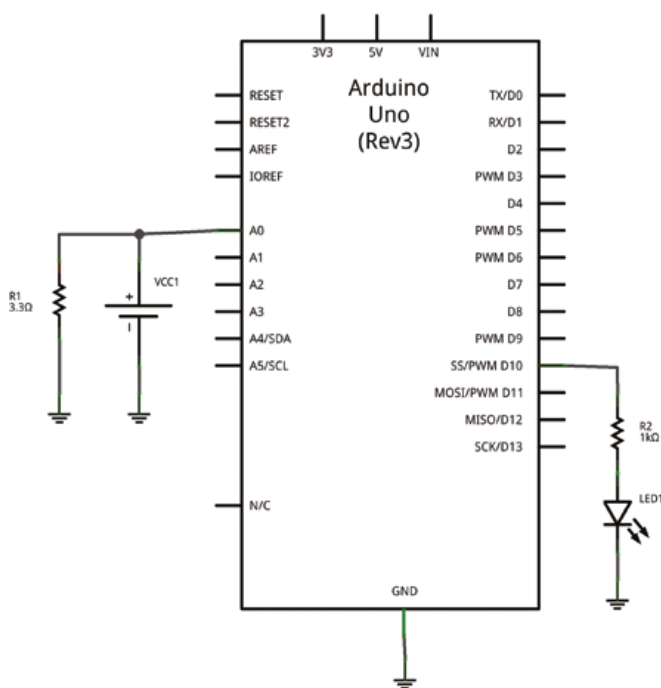
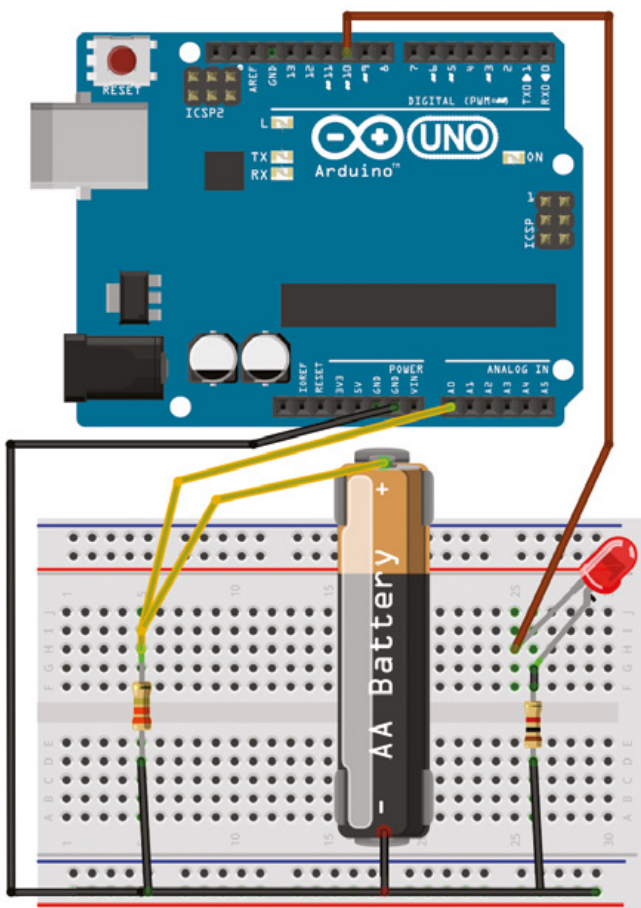
$$P = R I^2 = 1,02 \text{ W}$$

2) Pour 2,7 ohms

$$I = 1,5 / 2,7 = 0,556 \text{ A}$$

$$P = R I^2 = 0,85 \text{ W}$$

L'utilisation d'une résistance usuelle, très largement sous-dimensionnée, de 1/4 de Watt risque donc de transformer le montage en un joli feu d'artifice. Si nous souhaitons tester



avec de telles intensités de courant, il faudra utiliser une résistance qui peut au minimum dissiper 2 Watts. En augmentant légèrement la valeur de la résistance et donc en diminuant le courant de décharge, nous pourrions nous contenter de résistances de 1W, qui sont un peu moins volumineuses.

Exemple pour 0,15C :

$$I = 2500 \times 0,15 = 375\text{mA}$$

$$R = U / I = 1,2 / 0,375 = 3,2 \text{ ohms}$$

Valeur de résistance la plus proche = 3,3 ohms !

$$P = UI = U^2/R$$

$$P = 1,5^2 / 3,3 = 0,680 \text{ W}$$

Dans ce cas, une résistance de 1 W est largement suffisante.

9. PREMIER SCHÉMA

Ces quelques calculs terminés, nous pouvons commencer un premier câblage très simple qui pourra être mis en œuvre rapidement. Ce schéma couvre un canal individuel de décharge.

Les entrées analogiques non utilisées, mais lues par le croquis doivent toujours être câblées à la masse, soit directement, soit par l'intermédiaire d'une résistance de « pull-down ». Cette résistance de pull-down, généralement d'une valeur élevée de l'ordre de quelques centaines de kilo-ohms, a pour but de ramener la tension d'entrée au niveau de la masse, soit 0V. En effet, la valeur d'une tension lue sur une entrée analogique « volante » – c'est-à-dire non connectée – sera totalement aléatoire !

Un croquis de quelques lignes transmet la valeur des tensions mesurées au format CSV. L'Arduino UNO possède six entrées analogiques, et nous pourrions donc décharger simultanément jusqu'à six batteries. Le croquis ci-dessous est prévu pour suivre la décharge sur quatre canaux maximum. Ce nombre de canaux peut être très facilement augmenté ou diminué en modifiant les premières lignes du programme.

```

/* -----
 * Analyseur de piles rechargeables
 * Croquis "schéma simple"
 * ----- */

// 4 canaux de test, possibilité d'étendre à N canaux
const int NBCHANNELS = 4;
const char strCsvHeader[] = "Millis;Num;Analog0;Volt0;Analog1;Volt1;
Analog2;Volt2;Analog3;Volt3;";

int analogInputs[NBCHANNELS] = {0, 1, 2, 3}; // Entrées analogiques
des 4 canaux
int digitalOutputs[NBCHANNELS] = {10, 11, 12, 13}; // Sorties
numériques (LED d'avertissement)

long meteringDelay = 60000L; // 60000 ms = 1 minute entre deux
mesures (réduire cette valeur pour les tests)
int loopCounter; // Compteur de boucles

int cutOffVoltage = 163; // Tension de seuil pour allumer la LED =
0.8V ( 1023 / 5 * 0.8 = 163 )

// -----
// Procédure d'initialisation
// -----
void setup() {

    loopCounter = 0;

    // Sorties numériques, LED éteintes
    for(int i=0; i<NBCHANNELS; i++) {
        pinMode(digitalOutputs[i], OUTPUT);
        digitalWrite(digitalOutputs[i], LOW);
    }

    // Ouverture COM console ("moniteur série")
    // et envoi entête CSV
    Serial.begin(9600);
    Serial.println(strCsvHeader);
}

// Lecture des infos sur le canal "channelNum"
void processChannel(int channelNum) {
    int iAnalogValue;
    float fVoltage;

    if (channelNum >=0 && channelNum < NBCHANNELS) {
        iAnalogValue = analogRead(analogInputs[channelNum]); // Valeur sur
10 bits
        fVoltage = 5.0 * iAnalogValue / 1024; // Tension en Volts (0..1023
<=> 0..5 V)
        if (iAnalogValue < cutOffVoltage) {
            // Tension lue sous le seuil défini. On allume la LED.

```



```
        digitalWrite(digitalOutputs[channelNum], HIGH);
    } else {
        digitalWrite(digitalOutputs[channelNum], LOW);
    }
}
// Envoi des données sur la console
Serial.print(iAnalogValue);
Serial.print(";");
Serial.print(fVoltage);
Serial.print(";");
}

/* -----
   Boucle principale
   ----- */
void loop() {
    int i;
    loopCounter++;

    Serial.print(millis()); // Référence temps en millisecondes.
    Serial.print(";"); // Séparateur CSV
    Serial.print(loopCounter); // Compteur de boucles
    Serial.print(";");
    // Traitement des NBCHANNELS canaux analogiques
    for(i=0; i < NBCHANNELS; i++) {
        processChannel(i);
    }
    Serial.print("\n"); // Saut à la ligne après traitement de tous les canaux
    delay(meteringDelay); // Attente entre deux mesures
}
```

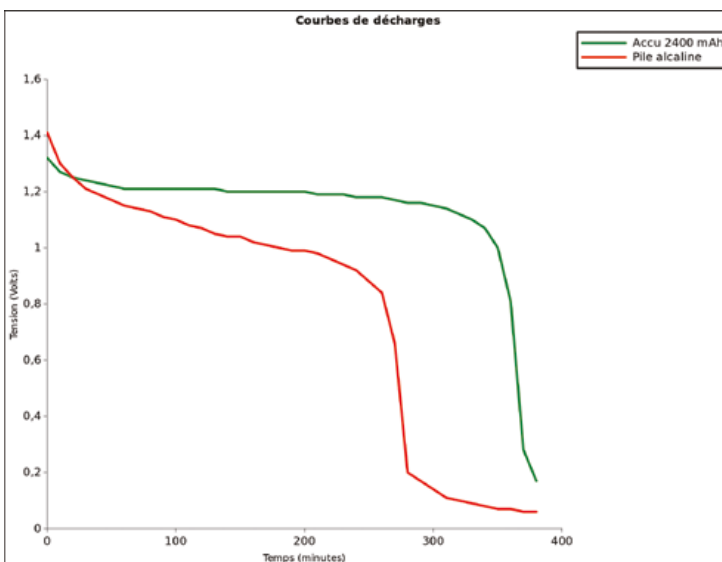


Figure 2 : Courbes d'exemples de charges et décharges piles accus.

L'inconvénient majeur de ce premier schéma est qu'il n'y a aucun contrôle de la fin de la décharge. La résistance reste en permanence branchée sur la batterie et consomme tout le courant qu'elle lui fournira jusqu'à son épuisement total si nous n'interrompons pas le processus. La tension de la batterie va donc descendre à 0 V.

Ce n'est certainement pas une bonne idée de vider à ce point une batterie. Cela présente même le risque de l'endommager ou de réduire ses performances. Aucun intérêt non plus d'un point de vue pratique : la majorité des appareils alimentés par des accus ne fonctionneront plus si la tension – par accu – descend sous les 0,8V.

D'un point de vue « expérimental », la décharge maximale d'un accu nous intéresse. Cf. la courbe de décharge obtenue qui est très intéressante (Figure 2). À mettre en relation avec la même courbe d'une pile alcaline classique sacrifiée sur l'autel de la science. Pour une pile alcaline, la mesure de la capacité est une expérience sans retour !

Comme l'arrêt de la décharge n'est pas possible dans ce schéma, nous avons ajouté une résistance R2 et une diode électroluminescente rouge LED1 pour avertir de la fin de la décharge. Nous utiliserons une sortie numérique de l'Arduino pour allumer cette LED lorsque la tension aux bornes de la batterie descend sous un certain seuil.

Quelques composants supplémentaires – transistor, résistance et buzzer piézoélectrique – permettraient tout aussi bien l'ajout d'une alarme sonore.

10. DEUXIÈME SCHÉMA

Nous allons ajouter quelques composants pour « prendre soin » de nos accus en leur évitant une décharge excessive.

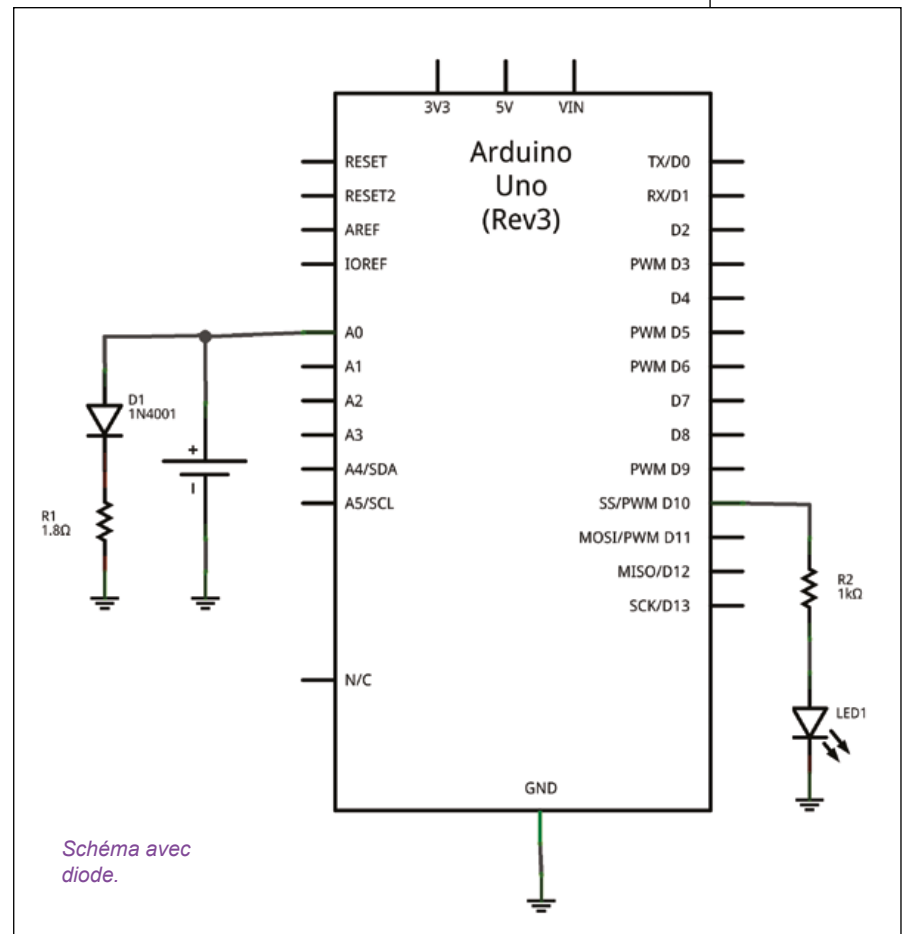
Quelles sont les options pour arrêter une décharge ? Un petit inventaire rapide des composants dont nous disposons... Diodes, transistors à jonction et à effet de champ, relais (reed ou classiques)... Dans le choix des

composants, nous devons obligatoirement tenir compte de leurs spécifications techniques. Hors de question de demander à un transistor de véhiculer un courant de 1 A si ses spécifications précisent un courant collecteur/émetteur maximum de 500 mA.

10.1 Première option

La plus simple : ajouter une diode en série avec la résistance. La diode est composée d'une anode A et d'une cathode K. Ce semi-conducteur devient conducteur quand la tension AK est positive et suffisante – en pratique supérieure à +0,6 Volt. La tension d'un accu est suffisante pour que la diode devienne conductrice.

En fin de décharge, dès que la tension sera sous les 0,6V, la diode ne sera plus conductrice et se comportera comme un circuit ouvert.





Nous devrions aussi dans ce cas calculer une nouvelle valeur pour la résistance de charge afin de maintenir en courant d'environ 350 mA.

$$R = (1,2V - 0,6V) / 0,350 A$$

$$R = \pm 1,8 \text{ Ohms}$$

Malheureusement, la chute de tension aux bornes de la diode est loin d'être stable et varie de quelques dizaines de millivolts en fonction de facteurs physiques ou électriques. Par exemple : l'intensité du courant qui la traverse, la température... Nous sommes donc dans une situation où nous mesurons une tension de l'ordre de 1,2 V à laquelle nous devons retrancher environ 0,6 V pour calculer l'intensité du courant qui traverse la résistance. La chute de tension aux bornes de la résistance sera d'environ 0,7 V.

D'un point de vue de la précision des mesures, c'est catastrophique ! Nous pourrions imaginer mesurer la tension aux bornes de la résistance. Cela nous obligera à utiliser deux entrées analogiques pour déterminer cette tension V_{res} qui sera de l'ordre de :

$$V_{res} = V_{accu} - V_{diode}$$

$$V_{res} = \sim 1,2V - \sim 0,6V$$

Si nous tenons compte de l'incertitude de la mesure de tension sur l'entrée Arduino ($5V / 1024 = 4 \text{ mV}$) et de la contrainte d'utiliser deux entrées analogiques par canal – donc maximum 3 canaux pour un Arduino UNO cela représente un inconvénient majeur. Sans oublier que la valeur de notre résistance a elle-même aussi une tolérance de 5 % voire 10 %.

10.2 Seconde option

Un transistor, autre composant bon marché, pourrait-il venir à notre secours ? Il existe deux grandes familles de transistors. Les transistors à jonctions et les transistors à effet de champ.

Avec ces composants, nous rencontrerons des problèmes similaires à ceux que nous avons rencontrés avec la diode. La chute de tension aux bornes collecteur/émetteur (ou drain/source) est trop variable, mais avant tout trop significative par rapport aux 1,2 volts que nous souhaitons mesurer. Suivant les modèles de transistors, la chute de tension entre CE – ou DS pour les FET – varie de 0,2V à 0,4V. Ce schéma fonctionnera très mal de toute

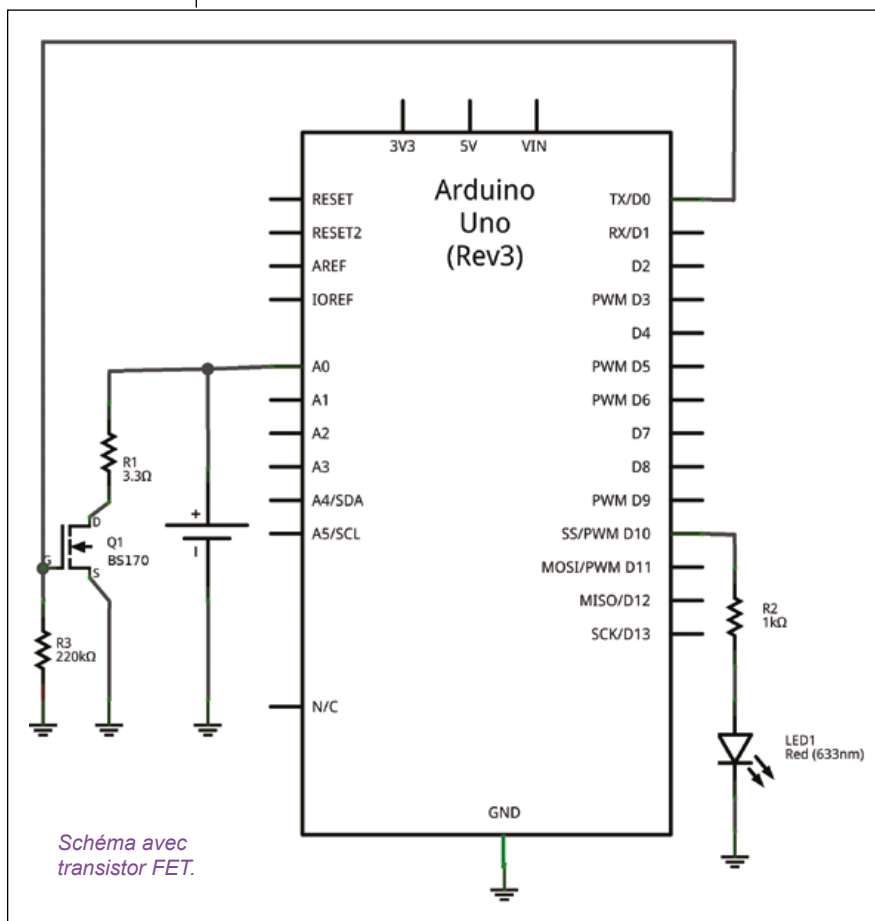
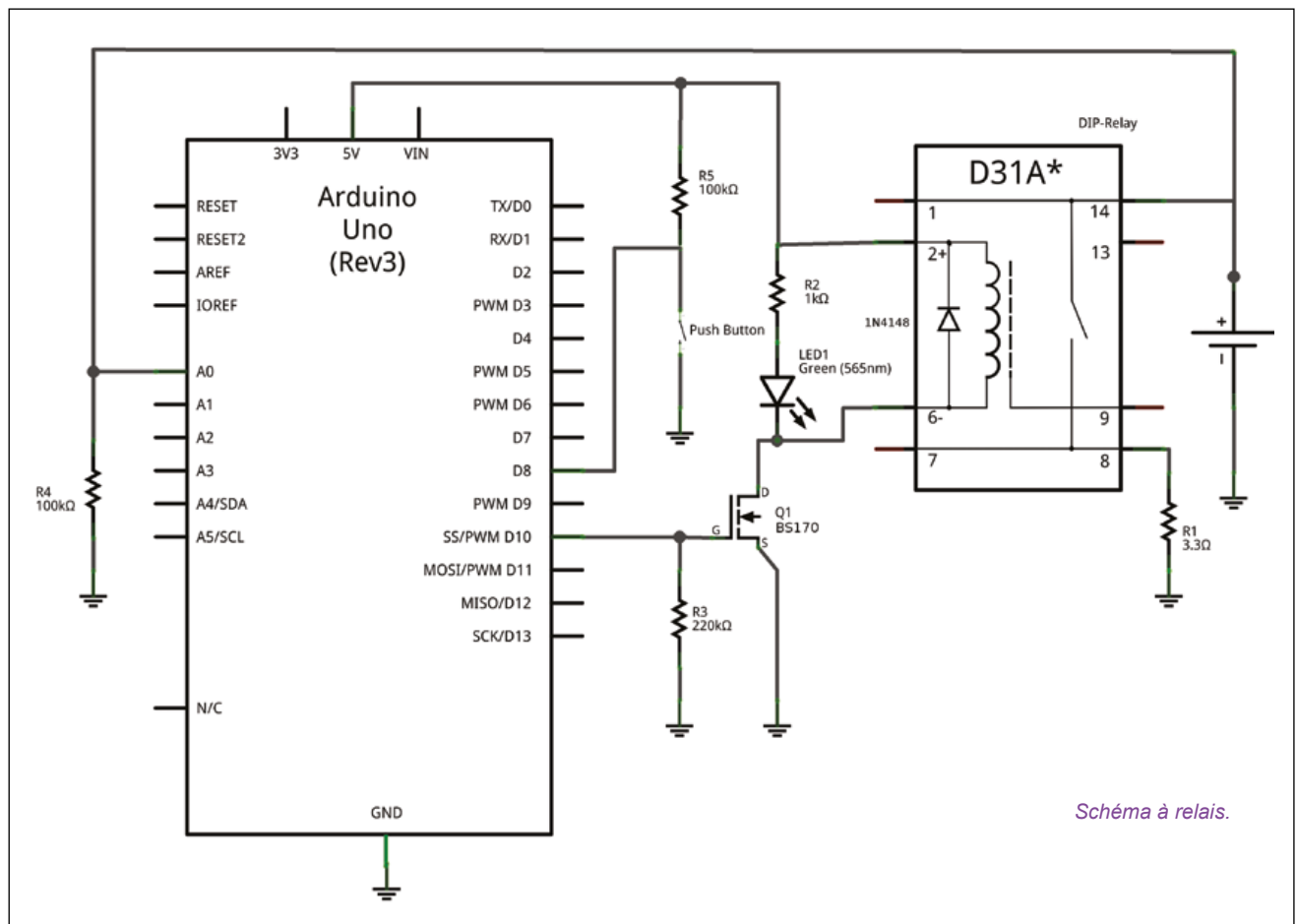


Schéma avec transistor FET.



façon : la tension appliquée de 1,2V est trop faible pour assurer un bon fonctionnement des transistors classiques.

10.3 Troisième option

Les deux schémas précédents ont mis en évidence les problèmes que nous rencontrons en pratique avec les semi-conducteurs. Ce fut aussi l'occasion d'introduire quelques notions de base en électronique.

Que nous reste-t-il comme solution ? Une dernière idée serait d'utiliser un relais pour « fermer » le circuit de décharge. Le contact d'un bon relais a une résistance

très faible. La mesure de tension aux bornes de la résistance sera donc plus précise que dans les cas précédents. Nous pouvons utiliser une sortie numérique de l'Arduino pour fermer le relais au début du test. Une fois la tension sous une valeur que nous fixons – par exemple 0,8V, il suffit de couper le relais (ouvrir le circuit de décharge) pour arrêter le processus de décharge et préserver notre batterie. Le relais sera plus coûteux que les composants envisagés précédemment. Il faut compter +/- 2,50 euros pièce. Ce qui reste au final raisonnable.

Dans ce schéma, nous avons ajouté un bouton poussoir qui

permettra de lancer, ou de relancer, la procédure de test de décharge. Le transistor à effet de champ Q1 fonctionne en commutation pour commander la fermeture du relais. La valeur de la tension sur la sortie numérique D10 commande l'ouverture ou la fermeture du relais. Si D10 passe à 5V – ou « HIGH » – le transistor devient conducteur et le relais se ferme. Si D10 est à 0V – ou « LOW » – le transistor n'est plus conducteur et la bobine du relais qui n'est plus alimentée ouvre le circuit. La diode électroluminescente LED1 et sa résistance R2 sont en parallèle sur la bobine du relais. Cette LED est allumée quand le relais est fermé.



La résistance de « pulldown » R4 qui relie l'entrée analogique à la masse joue à nouveau un rôle important. Sans cette résistance, dans le cas où aucune pile ne serait branchée, l'entrée analogique serait volante et la valeur de la tension lue sur cette entrée aléatoire. Pour se convaincre de l'utilité de cette résistance, il suffit de la retirer et d'exécuter le croquis sans accu inséré dans le support de batterie.

La bobine d'un relais est une charge inductive. Il faut obligatoirement placer une diode de roue libre aux bornes de cette bobine (entre les bornes 1 et 6 du relais sur le schéma) pour éviter toute surtension inverse lorsqu'on coupe le courant qui la traverse. Cette surtension peut très rapidement détruire le transistor qui commande le relais ! Cette diode est parfois intégrée directement dans le relais en lui-même (cf. spécifications techniques du fabricant). Si ce n'est pas le cas, elle doit être ajoutée directement au montage.

11. LE CROQUIS ARDUINO

Ce croquis est un peu plus complexe que le précédent. Un tableau de booléens mémorise l'état des canaux et des relais qui leur sont associés. Dès que la tension sur un canal est inférieure au seuil défini en constante (0,8V), le relais coupe le circuit de décharge pour préserver la batterie. Quand tous les canaux sont désactivés, le croquis envoie une chaîne de fin de traitement qui pourrait, par exemple, être interprétée par un programme que nous pourrions écrire en alternative à PuTTY.

```
/* -----
 * Analyseur de piles rechargeables
 * Croquis "schéma relais"
 * ----- */

// 4 canaux de test, possibilité d'étendre
// à N canaux
const int NBCHANNELS = 4;
const char strCsvHeader[] = "Millis;Num;Ana
log0;Volt0;Analog1;Volt1;Analog2;Volt2;Anal
og3;Volt3;";
const char strEndOfFile[] = "===EOF===";

boolean channelState[NBCHANNELS]; // Canal
ON ou OFF
long millisOffset;
int analogInputs[NBCHANNELS] = {0, 1, 2,
3};
int digitalOutputs[NBCHANNELS] = {10, 11,
12, 13}; // Sorties numériques de commande
des relais
int pushButtonIn = 8; // Entrée numérique
du bouton poussoir pour démarrer le test

float resistors[NBCHANNELS] = {3.3, 3.3,
3.3, 3.3}; // Résistances de (dé)charge
long meteringDelay = 60000L; // 60000 ms =
1 minute entre deux mesures (réduire cette
valeur pour les tests)
int loopCounter; // Compteur de boucles
boolean testRunning = false; // Procédure
de test en cours d'exécution ?
int cutOffVoltage = 163; // Tension d'arrêt
du test 0.8V ( 1023 / 5 * 0.8 = 163 )

// ---
// Initialisation
void setup() {
    int i;
    // Communication sur la console /
    "moniteur série"
    loopCounter = 0;

    Serial.begin(9600); // Ouverture COM
    console ("moniteur série")

    // Entrée du bouton poussoir
    pinMode(pushButtonIn, INPUT);
```

```
// Sorties relais et états des canaux de test
for(i=0; i<NBCHANNELS; i++) {
    pinMode(digitalOutputs[i], OUTPUT);
    digitalWrite(digitalOutputs[i], LOW);
    channelState[i] = false;
}

// ---
// Lecture des infos sur le canal "numChannel"
void processChannel(int numChannel) {
    int iAnalogValue;
    float fVoltage;

    if (numChannel < NBCHANNELS) {

        if(channelState[numChannel]) {
            // Canal ouvert
            iAnalogValue = analogRead(analogInputs[numChannel]); // Valeur
sur 10 bits
            fVoltage = 5.0 * iAnalogValue / 1024; // Tension en Volts
(0..1023 <=> 0..5 V)
            // Faut-il arrêter le test pour ce canal ?
            if (iAnalogValue < cutOffVoltage) {
                // On ouvre le relais et arrêt de la procédure
                digitalWrite(digitalOutputs[numChannel], LOW);
                channelState[numChannel] = false;
            }
        } else {
            // Canal fermé.
            iAnalogValue = 0;
            fVoltage = 0.0;
        }

        // Que le canal soit ouvert ou non, on imprime la valeur lue (0 si
fermé).
        Serial.print(iAnalogValue);
        Serial.print(";");
        Serial.print(fVoltage);
        Serial.print(";");
    }
}

// ---
// Attente appui du bouton poussoir pour démarrer le test
void waitPushButton() {
    int inValue;
```



```
do {
    inValue = digitalRead(pushButtonIn);
} while(inValue == HIGH);

testRunning = true;
millisOffset = millis();
loopCounter = 0;
}

boolean performMonitoring() {
    boolean bChannelOpened = false;

    if (! testRunning) {
        // Attente du signal pour commencer
        waitPushButton();

        // Début d'un nouveau test !
        // On ferme tous les relais
        for(int i=0; i<NBCHANNELS; i++) {
            digitalWrite(digitalOutputs[i], HIGH);
            channelState[i] = true;
        }
        Serial.println(strCsvHeader);
    }

    // Test commencé ou en cours
    // Si tous les canaux sont fermés, on arrête le test !
    for(int i=0; i<NBCHANNELS; i++) {
        bChannelOpened = bChannelOpened || channelState[i];
    }
    testRunning = bChannelOpened;

    return testRunning;
}

/* -----
   Boucle principale
   ----- */
void loop() {
    int i;
    // Conditions remplies pour commencer ou poursuivre le monitoring ?
    if( performMonitoring() ) {
        loopCounter++;
        Serial.print(millis() - millisOffset); // Référence temps en
millisecondes
        Serial.print(";"); // CSV pour import dans LibreOffice
        Serial.print(loopCounter); // Compteur de boucles
        Serial.print(";");
    }
}
```

```
for(i=0; i < NBCHANNELS; i++) {
    processChannel(i);
}
Serial.print("\n"); // Saut à la ligne après traitement de tous
les canaux
}
if (testRunning) {
    delay(meteringDelay);
} else {
    Serial.println(strEndOfFile);
}
}
```

12. INTERPRÉTATION DES DONNÉES

Les données des fichiers de traces sont au format CSV. Calc, le programme tableur de LibreOffice, peut ouvrir ces fichiers CSV et présenter le résultat des relevés de tensions en colonnes. Les données une fois importées dans Calc pourront être analysées et interprétées soit sous forme de graphiques, soit quantitativement pour calculer la capacité des accus.

Pour calculer la capacité en mAh d'une batterie avec LibreOffice, nous pouvons nous baser sur le courant moyen que la batterie a fourni sur toute la durée du test.

Voici, en résumé, la procédure à suivre :

- Importer le fichier CSV dans LibreOffice Calc.
- Ajouter une cellule « **R décharge** » avec la valeur de la résistance de décharge utilisée (Ohms).

- Calculer « **iMoyen** » le courant moyen. Pour déterminer le courant moyen, faire la moyenne de la tension et diviser par la résistance ($I = U / R$).
- Prendre la valeur maximum de la colonne « Millis » et la convertir en heure. « **temps** » = $\text{Max}(\text{Colonne Millis}) / 1000 / 3600$.
- Et finalement calculer la capacité en multipliant le courant moyen et le temps en heures. Multiplier par 1000 pour obtenir le résultat en mAh.

« **capacité** » = « **iMoyen** » * « **temps** » * 1000 mAh

CONCLUSION

Au fil de ces pages, nous avons abordé tout un ensemble de concepts en relation avec les batteries rechargeables. Batteries rechargeables qui n'ont finalement constitué qu'un prétexte pour élaborer quelques schémas électroniques basiques et aborder l'utilisation des entrées et sorties numériques ou des entrées analogiques de l'Arduino dans le but de satisfaire notre curiosité. L'approche souple des traces dans des fichiers CSV ouvre des perspectives pour des applications dans d'autres projets. **TM**

LIEN

Séries de valeurs normales pour résistances et condensateurs sur Wikipédia :

http://fr.wikipedia.org/wiki/CEI_60063



LES BONNES LEÇONS DU PASSÉ OU LA RÉSURRECTION D'UNE NEXTSTATION TURBO

Denis Bodor



On ne se rend pas facilement compte de la vitesse à laquelle le temps passe. Pourtant les ordinateurs avec des volumes de mémoire et des espaces disques exprimés en mégaoctets ne sont pas si loin de nous. Même si aujourd'hui 1,44 Mo est le volume courant d'un gros fichier, fut un temps, c'était la taille d'un support de stockage complet. Disposer d'un disque dur de quelques 120 Mo était un luxe, 540 Mo un rêve. En ce temps, une machine faisait rêver de nombreux utilisateurs, car tout simplement inaccessible à leur budget : un NeXT.



20 ans après, une machine de la marque NeXT est toujours une légende, mais celle-ci est enfin accessible pour qui est prêt à y passer du temps, attendre le bon moment et y mettre le prix. En ce qui me concerne, posséder une telle machine a toujours été un rêve et c'est aujourd'hui une chose que je peux rayer sur la fameuse liste des « un jour je... ».

Pourquoi ces machines sont-elles passées de la production à la légende ? Ceci tient à leur histoire et à celle de celui qui a créé la marque : un certain *Steve Jobs*, un dirigeant sans doute aussi doué que tyrannique, mais ayant toujours su s'entourer de personnes qui étaient de véritables génies de la technologie. En résumé, *Steve Jobs* a créé avec *Steve Wozniak* (alias « le Woz ») la société Apple en 1976, dont les produits ont révolutionné le monde de l'informatique personnelle. L'Apple II succéda à l'Apple I, puis il y eut le rarissime Apple III, le Lisa et surtout le Macintosh. Et puis *Steve Jobs* devint indésirable chez Apple pour diverses raisons et prit la porte en 1985... Il créa alors la société NeXT Inc avec pour objectif de concevoir et vendre le meilleur ordinateur au monde doté du meilleur système d'exploitation. Ceci donna naissance au NeXT Computer, puis à la NeXTstation et au NeXTcube avec, côté système, NeXTSTEP puis OPENSTEP.

Les caractéristiques de ces machines étaient littéralement impressionnantes pour l'époque



(et leur prix aussi). Ainsi, une NeXTstation Turbo pouvait avoir jusqu'à 128 Mo de mémoire, un affichage de 1120×832 pixels, un système audio intégré avec DSP dédié, une connectivité Ethernet 10base-2 (topologie en bus avec câble coaxial) et 10base-T (connecteur RJ45 actuel), des disques SCSI, et surtout un système d'exploitation proposant du développement orienté objet avec un créateur d'interfaces,

Le NeXT, la machine mythique qui a fait rêver bon nombre d'utilisateurs n'ayant de loin pas les moyens de se payer ce genre de technologie dans les années 90.



Une NeXTstation avec un clavier AZERTY et sa poussière d'époque... Combien en reste-t-il dans le monde en état de marche actuellement ?

un système d'affichage en Postscript et une vision « objet » complète. C'est sur un NeXTcube que Tim Berners-Lee (alias TimBL) inventa et implémenta le concept du *World Wide Web* (le web) au CERN au début des années 90. Mais les machines NeXT étaient hors de prix et la société, après avoir vendu quelque 50000 machines, se tourna entièrement vers le logiciel avec une version d'OPENSTEP pour plateforme Intel (plus Sparc et PA-RISC).

Et puis ce fut le retour de Steve Jobs chez Apple qui racheta NeXT en 1997 pour plus de 400 millions de dollars, absorbant par la même occasion la technologie NeXT. Ce renouveau technologique fut l'origine d'un point de rupture important chez Apple : le vieillissant Mac OS 9 laissa la place à Mac OS X, une évolution d'OPENSTEP pour architecture PowerPC, désignée pendant un temps sous le nom de projet Rhapsody. Il faut bien comprendre que Mac OS X n'est pas une évolution de Mac OS, mais d'OPENSTEP. Aujourd'hui, les utilisateurs Apple qu'il s'agisse d'iMac, de MacBook, d'iPod ou d'iPhone (et de l'Apple Watch) utilisent tous un système initialement basé sur les travaux, les recherches et les produits NeXT. Ceci touche aussi bien l'architecture globale du système, reposant sur UNIX, mais également une partie de la logique de l'interface. Si vous vous demandez pourquoi dans Mac OS X il y a un *Terminal.app* ou pourquoi un affichage « en colonne » est disponible dans le Finder, la réponse tient en un mot de quatre lettres : NeXT. Et ce ne sont

là que deux exemples « visibles » de cet héritage, la majorité de ce que l'utilisateur ne voit pas provient directement de NeXTSTEP/OPENSTEP. C'est également valable pour d'autres composants actuels du monde Apple comme WebObjects sur lequel repose Apple Store, iCloud et iTunes Store. Il s'agit là également d'une technologie héritée de NeXT.

Ainsi, c'est cette histoire, le peu de machines NeXT encore disponibles actuellement, leur prix de l'époque et l'avance considérable de la technologie qu'elles concentraient qui font que posséder une NeXTstation (dite *NeXT Slab* ou *NSlab*) ou un NeXTcube aujourd'hui, c'est littéralement posséder un morceau d'histoire, de notre histoire...

Les rares exemplaires qui ne se trouvent pas dans des musées refont tantôt surface sur des sites d'enchères en ligne. C'était le cas pour ma récemment acquise NeXTstation Turbo, vendue avec son écran mono (noir et blanc plus deux niveaux de gris) MegaPixel de 17 pouces, sa souris et son clavier... AZERTY ! La NeXTstation est une station de travail de seconde génération chez NeXT. C'est la version « économique » (vendue plus de 50000 francs HT à l'époque) accompagnant les offres basées sur le NeXTcube. La machine de première génération, le NeXT Computer, également en forme de cube (en magnésium SVP), est aujourd'hui quasiment introuvable. Cette station de travail qui est maintenant la mienne était équipée de 18 Mo de mémoire et d'un disque de 400 Mo, avec

NeXTSTEP 3.2 installé, un CD-ROM système (mais pas de lecteur) et un lecteur de disquette 2,88 Mo intégrés.

1. DÉBALLAGE, INSPECTION ET PREMIÈRE ERREUR

L'impatience est souvent cause de problèmes, mais, tantôt, par effet de bord, ceux-ci s'avèrent bénéfiques ou du moins éducatifs. La réception et le déballage de la machine, suivi de la mise en situation et des branchements fut l'occasion de tester le matériel pour un premier démarrage. La merveille, bien que construite il y a plus de 20 ans démarre sans le moindre problème, charge le système, permet un login et une exploration de l'interface NeXTSTEP 3.2.

Mais la tentation de regarder sous le capot et constater par soi-même le soin et l'attention apportés par ses concepteurs est trop grande. Le retrait d'une simple vis à l'arrière du boîtier est suffisant pour mettre les entrailles en évidence. Le retrait de la pile au lithium pour test au multimètre, en revanche, était une regrettable erreur. Celle-ci permet la sauvegarde des paramètres de configuration matérielle.

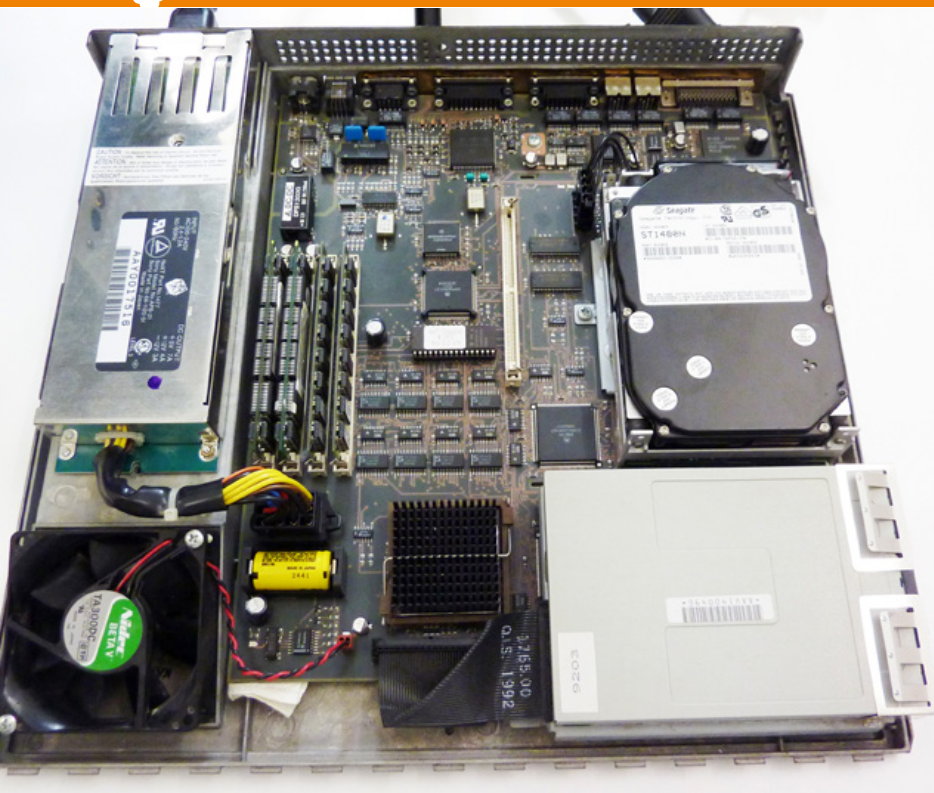
Les ordinateurs NeXT, comme les Mac, les SPARCstation ou encore des machines SGI n'ont pas cette horreur très simpliste qu'on appelle BIOS sur les PC. Bien sûr, aujourd'hui enfin cela est en train de changer avec la norme UEFI qui permet aux machines d'avoir

un système de boot en flash (et sur le disque) plus modulaire et plus fonctionnel qu'un BIOS « statique » (voire psychorigide). C'est une nouveauté sur PC, mais ceci existe de longue date sur d'autres architectures. Mais en retirant la pile assurant la sauvegarde des informations de configuration, on retourne aux valeurs par défaut. Or, sur la NeXTstation, une telle configuration n'indique pas un boot sur le disque SCSI. S'en sont donc suivi un démarrage, une tentative de boot en réseau et une grosse frayeur à la simple idée que le disque dur ait pu survivre deux décennies avant d'agoniser sous mes yeux suite au transport. Moralité, si vous mettez la main sur un *ordinosauire*, mitigez vos ardeurs et votre curiosité et agissez posément.

Mais c'est finalement ainsi que l'on peut faire connaissance avec le *NeXT ROM Monitor*, programme en ROM chargé de gérer le démarrage et la configuration initiale. Pour entrer dans le ROM Monitor, il faut faire usage de la combinaison de touches [Command]+[~] (tilde) qui, sur un clavier français est la touche [</>] du pavé numérique. On arrive alors dans l'interface du ROM Monitor, l'ex-futur OpenFirmware des Mac, sous la forme d'une fenêtre et d'un interpréteur de commandes. Là, **p** nous permet d'invoquer un outil de configuration, les éléments de démarrage et en particulier le périphérique de boot qui doit être utilisé, **sd** pour *SCSI Disk* et non **od** pour *Optical Disk*. Dans la succession de questions/réponses, on en profite pour activer les tests et le démarrage en mode verbeux.

L'arrière de la NeXTstation « pizza box » ou Nslab pour les intimes avec, de gauche à droite, le moniteur (alimenté par +12V-12V directement), le connecteur série pour l'imprimante et les deux types de connexion réseau 10baseT et 10base2 (coax) à cette époque très populaires.





L'intérieur du NSlab admirablement agencé avec sur la gauche le bloc d'alimentation, à droite le disque du SCSI et le lecteur de disquette et au centre en bas le processeur sous son radiateur.

L'entrée dans le ROM Monitor est également l'occasion de s'assurer de la configuration de la mémoire. Il suffit d'utiliser la commande **m** pour voir le résultat de la détection des barrettes de RAM installées dans le boîtier. S'en suit donc un accès à la « boîte à RAM » contenant toute les barrettes ayant été glanées au fil des années précédentes (quand on vous dit de ne jamais rien jeter, c'est précisément pour pouvoir faire face à ce genre de situation). Après essai de plusieurs combinaisons, la station NeXT se trouve gratifiée de quatre barrettes IBM 70ns de 16 Mo chacune permettant de joyeusement faire passer la configuration des 18 Mo de base à 64 Mo (un vrai monstre pour l'époque). Que du bonheur.

Le ROM Monitor permet bien plus que la simple configuration et l'affichage des mémoires détectées. Il est possible d'examiner le contenu de la mémoire et des registres, de gérer les interruptions, de procéder à des tests, etc. Il fait office d'outil de configuration, mais également de débogage et de solution de repli en cas de problème (*system panic*). Vous trouverez davantage d'informations sur le sujet sur <http://n-1.nl/next/hardware/info/rom-monitor.html>.

2. UTILISATEUR ET RÉSEAU

La machine m'a été livrée apparemment fraîchement installée et possède comme seul utilisateur **root**, qu'il n'est bien entendu pas raisonnable d'utiliser pour un

usage courant. Après login, on s'empresse donc de créer un utilisateur standard qu'on placera non seulement dans son propre groupe, mais également dans le groupe **wheel** (oui, oui, c'est toujours ce groupe qui correspond aux administrateurs dans Mac OS X). Ceci nous permettra d'utiliser la commande **su** si nécessaire pour les opérations super-utilisateur en ligne de commandes.

La première chose qu'on souhaite faire pour redonner vie à la station est, bien entendu, de la connecter au monde d'aujourd'hui : Internet. Une interface en mode graphique permet de configurer le réseau. Surprise, il n'est pas fait mention de résolution d'adresse via DHCP. En réalité, NeXTSTEP 3.2 ne dispose même pas de client DHCP, mais ce n'est guère étonnant, la première RFC qui parle de DHCP est la 1531 (octobre 1993) et il s'agit simplement d'une extension du protocole BOOTP permettant le démarrage réseau. Ont suivies la RFC1541 (1993) pour corriger les erreurs de la 1531 puis la 2131 (1997) qui a finalement bien installé le protocole dans les mœurs. Mais NeXTSTEP n'était déjà plus et avait été remplacé par OPENSTEP. Pas de solution miracle dans notre cas, il convient d'utiliser une configuration réseau statique et donc de définir manuellement une adresse IP dans l'interface. Les changements sont alors répercutés dans le fichier de configuration **/etc/hostconfig** qui représente une autre voie pour ce type de paramétrage :

```
#
# /etc/hostconfig
#
# This file sets up shell variables used by the various rc scripts to
# configure the host. Edit this file instead of rc.boot.
#
# Warning: This is sourced by /bin/sh. Make sure there are no spaces
# on either side of the "=".
#
# There are some special keywords used by rc.boot and the programs it
# calls:
#
#      -AUTOMATIC-      Configure automatically
#      -YES-            Turn a feature on
#      -NO-             Leave a feature off or do not configure
#
HOSTNAME=next1
INETADDR=192.168.10.163
ROUTER=192.168.10.1
IPNETMASK=255.255.255.0
IPBROADCAST=-AUTOMATIC-
NETMASTER=-NO-
YPDOMAIN=-NO-
TIME=-NO-
```

On n'oubliera pas, également, de régler le DNS via une modification du fichier `/etc/resolv.conf` :

```
# DNS Google
nameserver 8.8.8.8
```

3. AMÉLIORATION LOGICIELLE ET INSTALLATIONS COMPLÉMENTAIRES

À présent, le NeXT communique sur le réseau et répond aux **ping**. Il est même possible de naviguer sur le Web avec l'application dédiée, appelée OmniWeb. Bien entendu, le navigateur ne supporte que les standards qui étaient d'usage au début et au milieu des années 90. Oubliez donc les CSS, HTML5, Flash et les autres gouffres à ressources CPU qui peuplent le Web actuel, même les résultats d'une recherche Google ne s'affichent pas correctement. Il est amusant de constater qu'OmniWeb existe toujours (version 5.11.2 en 2012), mais repose maintenant sur un moteur HTML identique à celui de Safari (KHTML Webcore) et est, bien entendu, très différent.

Il y a plus intéressant à faire que de simplement tenter d'utiliser la machine telle qu'elle est. Nous allons compléter son offre logicielle de quelques éléments afin de lui fournir quelques fonctionnalités actuelles. Mais avant toute chose, un petit coup de scan de port (**nmap**) pour voir quels sont les services qui fonctionnent sur le système. Oh surprise, nous avons un service telnet et un FTP à notre disposition par défaut. On critique souvent le laxisme des machines et des systèmes embarqués actuels, mais force est de constater qu'à l'époque la situation n'était



Ce composant contient le programme de démarrage de la machine. À l'intérieur se trouve un code qui a maintenant 22 ans et qui pourtant dépasse en fonctionnalités encore aujourd'hui la plupart des BIOS des PC actuellement en service.

guère mieux. De tels services lancés par défaut sur une machine étaient et sont toujours une très mauvaise idée (voilà de quoi couper l'herbe sous le pied de quiconque souhaiterait lancer un « dans le temps ce n'était pas aussi catastrophique que maintenant »).

On se plie donc d'un petit **telnet** suivi de l'adresse de la machine depuis un autre poste et après avoir utilisé son nom d'utilisateur et son mot de passe (circulant en clair sur le réseau), nous voici avec une ligne de commandes du shell C. Voilà l'occasion de lancer une petite commande pour se faire plaisir :

```
next1> hostinfo
Mach kernel version:
  NeXT Mach 3.2: Tue Sep 14 19:25:36 PDT 1993;
  root(rcbuilder):mk-149.30.12.obj~2/RC_m68k/RELEASE_M68K

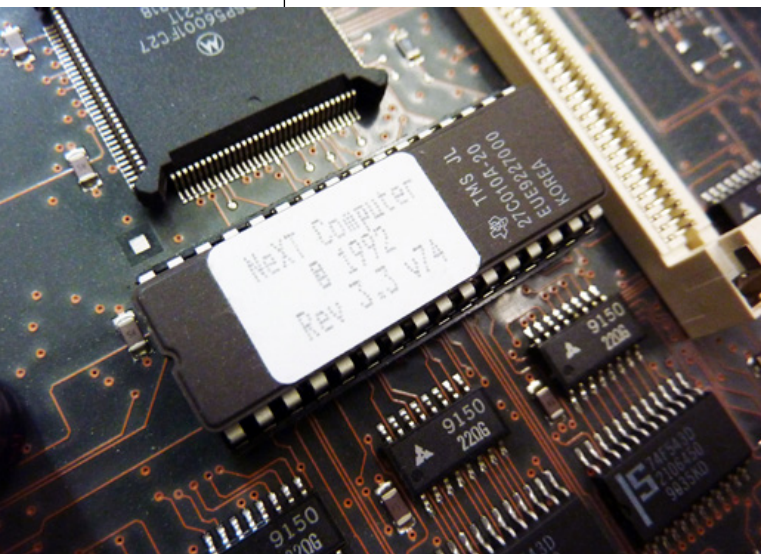
Kernel configured for a single processor only.
1 processor is physically available.
Processor type: MC680x0 (68040)
Processor speed: 33 MHz
Processor active: 0
System type: 4
Board revision: 0xf
Primary memory available: 64.00 megabytes.
Default processor set: 43 tasks, 71 threads, 1 processors
Load average: 0.00, Mach factor: 0.99
```

Sous nos yeux, un système reposant sur un noyau Mach compilé il y a tout juste 21 ans, fonctionnant sur un magnifique processeur 68040 cadencé à 33 Mhz. C'est beau, au point d'en avoir les larmes aux yeux. Ce duo va nous permettre de manipuler des éléments récupérés par une autre machine (sous GNU/Linux), envoyée au NeXT via FTP, et installée via la ligne de commandes obtenue via telnet. Un sympathique autre « restaurateur » de machines NeXT a eu la gentillesse de mettre à disposition une collection d'outils directement installable sur le système : <http://www.nleymann.de/Nextstep/>.

Notre premier élément sera **wget**, un outil permettant de récupérer des pages HTML et des fichiers en HTTP. Ceci nous permettra d'éviter l'étape FTP et de télécharger les archives directement sur le NeXT. On télécharge donc le fichier **wget.1.8.1.N.b.tar.gz** qu'on s'empresse de transférer en FTP dans **/denis/DNL**. S'en suit l'installation via telnet, avec :

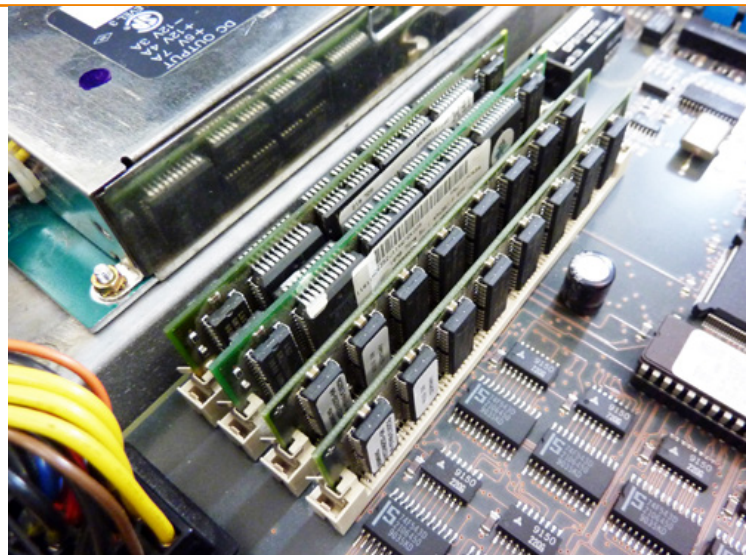
```
next1> su
next1> cd /usr/local
next1> gzip -dc /denis/DNL/
wget.1.8.1.N.b.tar.gz | tar xvf -
```

L'astuce consiste à se placer dans le répertoire **/usr/local** et à y décompresser l'archive dont les fichiers trouveront leur place dans **bin/**, **share/**, etc. Ceci étant, on dispose maintenant d'une commande **wget** et il devient possible de plus rapidement récupérer le reste des binaires et des



bibliothèques avec **wget** suivi de l'URL du fichier qui nous intéresse. On récupère ainsi :

```
bc.1.06.N.b.tar.gz
libiconv.1.7.N.b.tar
make.3.79.1.N.b.tar.gz
openssh.3.1p1.N.b.tar.gz
openssl.0.9.6c.N.b.tar.gz
sed.3.02.N.b.tar.gz
vim-61-nextstep33-N-bin.tar.gz
wget.1.8.1.N.b.tar.gz
zlib.1.1.4.N.b.tar.gz
```



Une petite fouille dans la collection de barrettes de mémoire de récupération et quelques essais et voici la machine gratifiée de quelques mégaoctets supplémentaires totalement gratuits...

Il suffira alors d'installer tout cela comme précédemment avec **gzip** et **tar** dans **/usr/local**. Quelques outils à jour, de quoi développer plus facilement et un éditeur de texte/code digne de ce nom (Vim) c'est une chose, mais le plus important c'est de disposer de SSH. L'archive **openssh.3.1p1.N.b.tar.gz** installera le client et le serveur, mais il nous reste de la configuration à faire.

La première étape concerne la connexion cliente (du NeXT vers une machine GNU/Linux moderne). Pour ce faire, il faut générer une paire de clés permettant de nous identifier de manière convenable et accessoirement nous rappeler que ce qui prend quelques secondes sur un PC moderne est une tâche titanesque sur une machine qui a plus de 20 ans. La génération de la paire de clés DSA se fait avec la commande **ssh-keygen -t dsa** et beaucoup, beaucoup, beaucoup de patience. Non, le système n'a pas planté, il est tout simplement à genoux et il faut simplement attendre !

Le serveur SSH, quant à lui, nécessite également la génération de clés dites clés d'hôte. Cependant, comme on ne sait pas quel type de client va se connecter au serveur, il est préférable de générer les trois types de paires de clés. On utilisera donc la même commande de génération, mais en spécifiant les types **rsa1**, **rsa** et **dsa** (via l'option **-t**) et on précisera respectivement les fichiers **ssh_host_key**, **ssh_host_rsa_key** et **ssh_host_dsa_key** (dans **/usr/local/etc**). Dès lors, tout est en place pour le premier essai.

On teste simplement en lançant le serveur à la main avec **/usr/local/sbin/sshd -d**, l'option spécifiant le mode debug nous permettant de garder la main et de voir tous les messages d'information apparaître sur l'écran. Attention, le lancement du serveur prend du temps pour la génération de la clé temporaire, mais le port est ouvert avant cette génération. De ce fait, si on tente de se connecter depuis une autre machine immédiatement après avoir validé la commande, la première connexion prend un temps non négligeable.

Normalement, tout se passe bien et après authentification on obtient une ligne de commandes via SSH comme sur les systèmes modernes. Nous pouvons donc intégrer le lancement du serveur au démarrage système en éditant **/etc/rc**. Vers la fin, on ajoute ces quelques lignes pour le lancement du serveur OpenSSH :

```
if [ -f /usr/local/sbin/sshd ]; then
    (echo -n 'Starting OpenSSH server:') >/dev/console
    fbshow -B -I "Starting OpenSSH server" -z 97
    /usr/local/sbin/sshd
    (echo '.') >/dev/console
fi
```



It's alive ! (PS : le capharnaüm entourant la machine n'est absolument pas représentatif de l'état courant du lab de Hackable. C'est une mise en scène afin de vous faire déculpabiliser (je suis crédible là ??))

fbshow est l'outil chargé de composer la barre de progression au démarrage, le paramètre **-z** permet de donner la position de la barre, ici 97%, entre les 95% de « standard daemons » et les 100% de « Startup complete ». Ceci ne s'affiche qu'en mode de démarrage standard (graphique). En mode verbeux, **fbshow** n'est pas utilisé, mais c'est la commande **echo** qui informe de la progression du démarrage des différents services. Il est très agréable de constater que le script de démarrage (**/etc/rc**) est parfaitement commenté et clairement conçu pour être personnalisé de cette manière. Les choses sont conçues proprement et intelligemment, c'est l'esprit UNIX qui forme la base de NeXTSTEP.

4. UN PETIT BONUS : LE CD-ROM D'ORIGINE

À ce stade, on dispose d'une NeXT Station Turbo avec 64 Mo de mémoire, ayant une connectivité Internet et offrant un accès SSH, le tout avec un éditeur Vi digne de ce nom. Bien entendu, l'idée n'est pas de dénaturer la machine ou le système, mais simplement de lui apporter les quelques éléments modernes plus ou moins indispensables pour un usage agréable. Cependant, l'installation par défaut de NeXTSTEP sur la machine possède une lacune : le paquet

du langage français n'est pas installé et de ce fait l'interface graphique dans notre langue n'est pas disponible. On dispose bien du CD-ROM d'installation, mais la machine ne dispose pas de lecteur et trouver un lecteur CD-ROM SCSI en étant de marche et à un prix raisonnable aujourd'hui risque d'être une tâche presque aussi difficile que l'acquisition du NeXT lui-même (si vous voulez vous débarrasser d'un tel matériel, l'adresse de la rédaction est en début de magazine ;)).

Nous devons donc lire le CD-ROM sur une autre machine et transférer le paquet nécessaire sur le NeXT pour installation. La fonctionnalité de lecture d'un CD-ROM UNIX (système de fichiers UFS) élimine d'office l'utilisation de Windows. Mac OS X serait une option via l'utilisation de MacFuse (ce système ne lit pas nativement les supports de son aïeule). La solution la plus simple, toutefois, consiste à utiliser GNU/Linux, disposant de tout ce qu'il faut pour lire presque tout et n'importe quoi.



Par prudence et afin d'éviter toutes manipulations non nécessaires du CD-ROM, on commence par en faire une image avant de ranger la précieuse relique à l'abri des rayures. Ceci ne nécessite pas de prendre en compte le type de données sur le support, on copie simplement octet par octet toutes les données du disque dans un fichier avec **dd** :

```
% sudo -s
% dd if=/dev/sr0 of=~ /NEXT/Nextstep32.iso
```

Il nous suffit ensuite de passer en **root** pour charger le module permettant le support d'UFS et de monter l'image dans un répertoire en utilisant les options adéquates :

```
% sudo -s
% modprobe ufs
% mount -t ufs -o loop,ro,ufstype=nextstep-cd \
~/NEXT/Nextstep32.iso /mnt/cdrom/
```

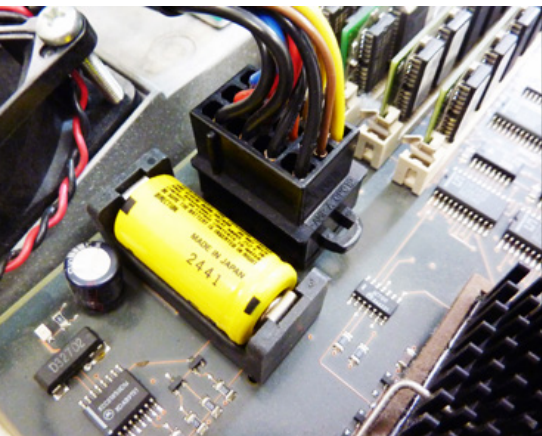
Et voilà, nous avons accès aux données du disque comme avec n'importe quel autre support, via **/mnt/cdrom**. Il s'agit d'un CD-ROM d'installation bootable, mais les paquets sont bel et bien présents, dont **NextCD/Packages/French.pkg** qui est précisément le paquet pour la langue française. Il nous suffit de copier cet élément depuis la machine GNU/Linux vers le NeXT via **scp -r** (c'est un répertoire, le **-r** est donc nécessaire). Il nous faudra ensuite nous loguer en **root** sur le NeXT, puis double-cliquer sur ce paquet pour lancer **Installer.app** et procéder à l'installation. On passera également dans le gestionnaire d'utilisateurs afin de définir le français comme langue principale. Après redémarrage et connexion avec l'utilisateur « normal », l'interface est en français.

L'installation de paquets est relativement aisée, mais pour faire un peu de place, on peut également supprimer des paquets. Pour cela, il suffit d'aller, avec le Finder, dans **/NextLibrary/Receipts** et cliquer sur les paquets et, dans **Installer.app** qui se lance alors, choisir l'option de désinstallation.

5. CE QU'IL FAUT RETENIR

Pourquoi donc non seulement mettre son nez dans ce genre de vieilleries (en dehors d'un fantasme de jeunesse), mais en plus en détailler la résurrection ? Tout simplement, car il s'agit d'une leçon d'histoire et une magnifique occasion de relativiser la puissance, la qualité et l'évolution des technologies que nous utilisons actuellement. En accédant ainsi physiquement à notre passé technologique, nous avons appris beaucoup de choses :

- Ne pas se précipiter et tester les éléments arbitrairement alors que la machine est en parfait état de marche. On se documente d'abord et on agit ensuite.
- Les erreurs peuvent conduire à des découvertes et il ne faut pas tirer de conclusion hâtive si un dysfonctionnement apparaît soudainement (le disque dur n'était subitement pas mort).
- La sécurité de l'époque n'était guère plus importante qu'elle ne l'est maintenant, au contraire.
- Il faut savoir être patient avec ces vieilles machines et respecter leur âge. On a désormais trop l'habitude que tout soit instantané et il est facile de perdre de vue que la machine travaille en coulisse.



La fameuse pile permettant de conserver les paramètres de démarrage et de vous faire une grosse frayeur après l'avoir retirée pour test au multimètre. Juste au-dessus, le connecteur de l'alimentation absolument non standard.

La souris NeXT. Carrée, noire, et avec deux boutons !



- Le matériel de l'époque était étrangement fait pour durer. C'est une constatation générale valable pour les NeXT, mais également pour n'importe quel ordinateur ayant quelques décennies au compteur (C64, Atari ST, Amiga, IBM PC, etc.). Je ne pense pas que dans 20 ans votre actuel PC, votre laptop, ou votre smartphone, pourrait se comporter de même sans pièces de remplacement (ventilateurs, disques, condensateurs, etc.).
 - L'évolution et la croissance des besoins en ressources sont globales. Même si le Web utilise toujours des protocoles et des standards qui n'ont que très peu changés, la spirale de l'évolution est sans fin : plus on a de ressources, plus on compose des sites lourds, plus les sites sont lourds, plus les machines ont besoin d'être puissantes. Ceci se constate déjà avec une machine comme un Pentium 4 sur le web d'aujourd'hui, c'est encore plus flagrant avec une machine de plus de 20 ans.
 - Les bonnes idées et l'intelligence dans la conception n'ont pas de lien avec la modernité. NeXTSTEP est bien pensé et il nous a été possible d'intégrer facilement un logiciel créé dix ans plus vieux que le système. Il devrait même être possible de compiler et installer une version actuelle de ce même logiciel avec un peu de ténacité. En d'autres termes, l'intelligence n'a pas d'âge et une bonne conception traverse les années sans encombre.
 - Le respect de la machine est primordial. On a trop souvent le réflexe de se penser plus rusé ou du moins de partir du principe que, parce qu'un système est ancien, il peut répondre à des principes qu'on juge maintenant comme parfaitement admis. Paradoxalement, c'est précisément l'état de fait inverse qui est constaté, ce sont les systèmes actuels qui reposent encore sur des principes de fonctionnement de systèmes dont ils héritent la logique.
 - Le bug de l'an 2000 a réellement existé. En effet, le NeXTSTEP installé est incapable de gérer une date au-delà du 31 décembre 1999 ce qui pose de nombreux problèmes quant à son utilisation et surtout des petits soucis lors des tentatives de compilation des sources comportant des dates de création et de modification de fichier après cette date fatidique (la commande **touch** est votre amie tout comme l'option **m** de **tar**).
- J'espère que cet article vous aura donné envie de repenser aux vieilles machines inaccessibles qui vous faisaient rêver plus jeune et pourquoi pas vous poussera à réaliser ce rêve avec votre regard et votre expérience d'adulte. Qu'il s'agisse d'une machine mythique ou commune, mais d'un autre âge, j'espère que vous ressentirez comme moi un certain plaisir doublé d'une bonne dose d'humilité en ravivant ce que nombreux considèrent comme des déchets ou des artefacts destinés à l'oubli... **DB**

PROFESSIONNELS !



DÉCOUVREZ NOS NOUVELLES OFFRES D'ABONNEMENTS ...

PDF COLLECTIFS

		PROFESSIONNELS					
		1 - 5 lecteurs		6 - 10 lecteurs		11 - 25 lecteurs	
OFFRE	ABONNEMENT	Réf	Tarif TTC	Réf	Tarif TTC	Réf	Tarif TTC
PROHK	6 ^{n°} HK	☐ PRO HK2	156,-	☐ PRO HK2	312,-	☐ PRO HK2	624,-

Prix TTC en Euros / France Métropolitaine

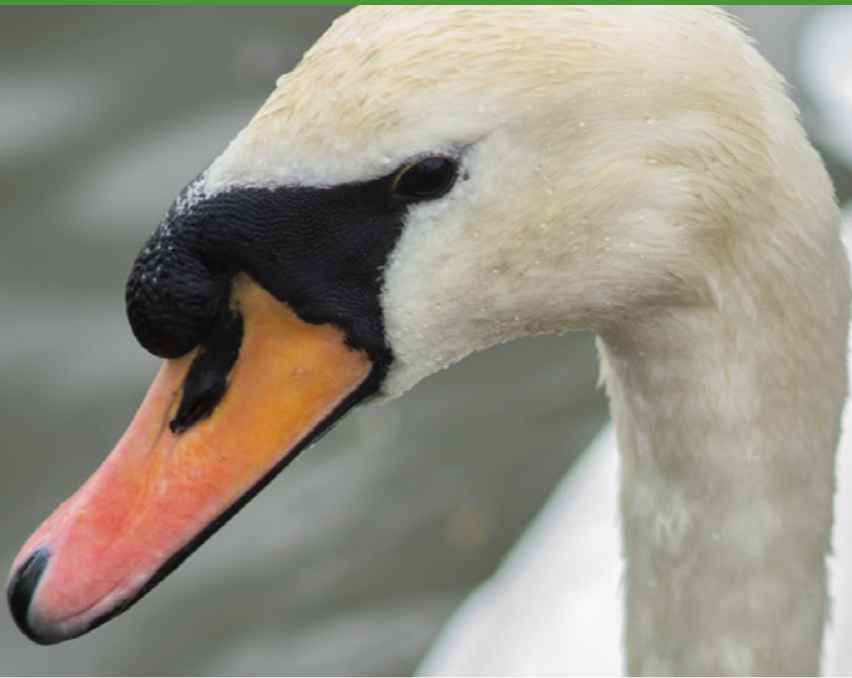
PROFESSIONNELS :
N'HÉSITEZ PAS À
NOUS CONTACTER
POUR UN DEVIS
PERSONNALISÉ PAR
E-MAIL :
abopro@ed-diamond.com
OU PAR TÉLÉPHONE :
03 67 10 00 20

ACCÈS COLLECTIFS BASE DOCU

		PROFESSIONNELS					
		1 - 5 connexion(s)		6 - 10 connexions		11 - 25 connexions	
OFFRE	ABONNEMENT	Réf	Tarif TTC	Réf	Tarif TTC	Réf	Tarif TTC
PROOS	OS	☐ PRO OS+3	90,-	☐ PRO OS+3	180,-	☐ PRO OS+3	360,-
PROH+	GLMF + HS + LP + HS + MISC + HS + OS	☐ PRO H+3	447,-	☐ PRO H+3	894,-	☐ PRO H+3	1788,-

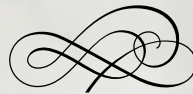
Prix TTC en Euros / France Métropolitaine

...EN VOUS CONNECTANT À L'ESPACE
DÉDIÉ AUX PROFESSIONNELS SUR :
www.ed-diamond.com



SSH OU COMMENT AVOIR UNE LIGNE DE COMMANDES VIA LE RÉSEAU

D. B.



Les utilisateurs GNU/Linux vous le diront, il est simple en ligne de commandes de « prendre la main » sur un système avec la magie de SSH. C'est quelque chose qui est également possible sur une Raspberry Pi ou n'importe quel nano-ordinateur ou carte utilisant GNU/Linux en guise de système. Que diriez-vous, avec Cygwin, de prendre ainsi le contrôle de votre Windows à distance ?

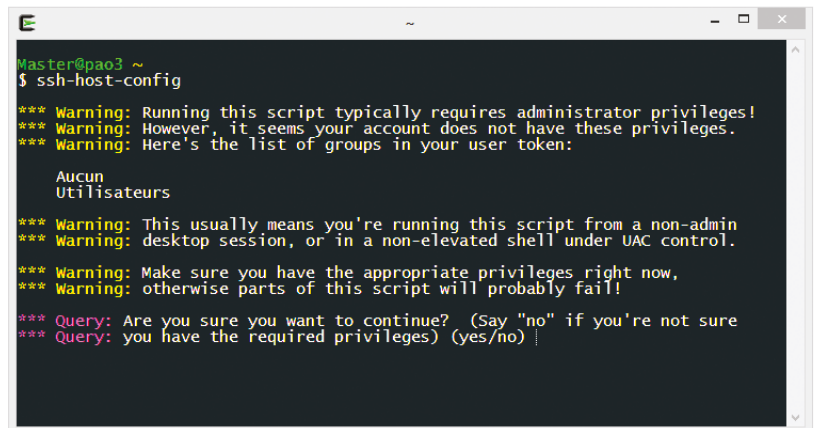
Nous l'avons vu, nous pouvons nous connecter avec la commande **ssh** aussi bien à un PC sous GNU/Linux qu'à un nano-ordinateur de type Raspberry Pi. Ceci offre des avantages non négligeables lorsqu'on ne dispose pas forcément du couple écran/clavier sur son nano-ordinateur, mais également lorsqu'on ne veut ou ne peut simplement pas se déplacer jusqu'à la machine.

Traiter de l'ensemble des options qui s'offrent à nous avec SSH sort du cadre de cet article et du magazine tout entier. Sachez simplement qu'en plus de prendre la main sur une machine, vous pouvez transférer des fichiers (**scp**), sécuriser des échanges, créer des tunnels, utiliser SSH comme relais vers un autre ordinateur et même utiliser des applications graphiques entre deux systèmes différents. Je vous recommande la lecture des nombreux tutoriels en ligne sur l'utilisation de SSH ainsi que des articles sur le sujet qui ont été publiés dans nos autres publications (*GNU/Linux Magazine* et *Linux Pratique*). Les explications données, généralement pour GNU/Linux, s'appliqueront parfaitement à Cygwin puisque, je vous le rappelle, les outils sont strictement identiques.

Ce qui nous intéresse ici est plus atypique. Alors que sous GNU/Linux et Mac OS X, permettre l'accès à la machine depuis un autre poste en ligne de commandes se fait par une simple case à cocher, Windows lui, n'offre par défaut absolument aucun service de ce genre. Bien entendu, il vous est toujours possible d'utiliser la « Connexion Bureau à distance », en mode graphique, ou encore VNC qui fonctionnera admirablement entre différents

systèmes. Mais on parle ici de rendre accessible le système, non pas en mode graphique, mais en ligne de commandes et ce, via le réseau. Pourquoi ? Hé bien déjà parce que c'est possible et ensuite parce que cela rend votre système accessible, de manière sécurisée (SSH utilise des connexions chiffrées), aussi bien pour obtenir une ligne de commandes que pour échanger des fichiers (oui, ça marche dans les deux sens).

Vous devrez dans un premier temps installer OpenSSH via l'installateur Cygwin. Ensuite, vous devrez lancer le terminal Cygwin en tant qu'administrateur (clic droit, **Exécuter en tant qu'administrateur**). Dans le cas contraire, ce qui suit vous afficherait ceci :



```

Master@pao3 ~
$ ssh-host-config
*** Warning: Running this script typically requires administrator privileges!
*** Warning: However, it seems your account does not have these privileges.
*** Warning: Here's the list of groups in your user token:

        Aucun
        Utilisateurs

*** Warning: This usually means you're running this script from a non-admin
*** Warning: desktop session, or in a non-elevated shell under UAC control.
*** Warning: Make sure you have the appropriate privileges right now,
*** Warning: otherwise parts of this script will probably fail!
*** Query: Are you sure you want to continue? (Say "no" if you're not sure
*** Query: you have the required privileges) (yes/no) |
  
```

En gros, ceci vous annonce brutalement « *apparemment vous n'avez aucun droit, la configuration ne fonctionnera pas, mais si vous êtes sûr de votre coup, vous pouvez toujours essayer. Oui ?* ». Et bien entendu, cela ne fonctionnera pas puisqu'on parle ici de modifier le comportement du système et qu'un utilisateur non privilégié n'a absolument pas le droit de faire ce genre de choses.

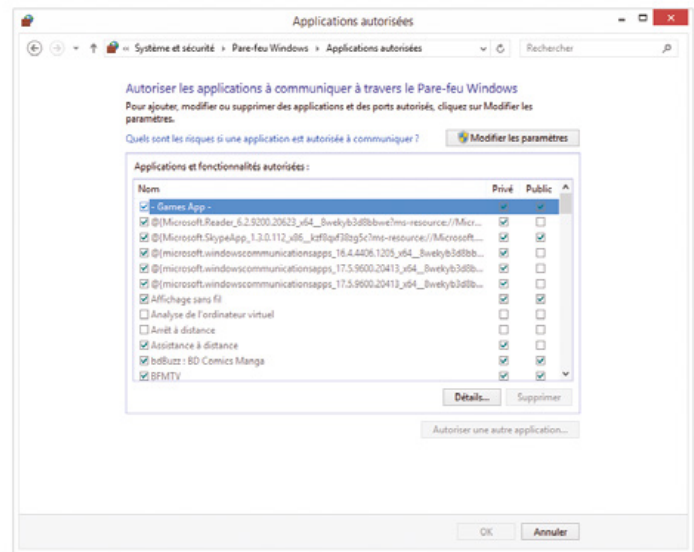
Une fois le terminal lancé correctement en tant qu'administrateur, tapez simplement la commande **ssh-host-config --yes**.

S'en suit, normalement, un interrogatoire en bonne et due forme. Fort heureusement, nous avons dépouillé tout cela pour vous et pouvons vous recommander de simplement répondre **yes** à chaque question. L'option **%--yes** permet d'ailleurs de faire cela automatiquement et de vous simplifier la vie. L'outil de configuration va faire plein de choses à votre place :

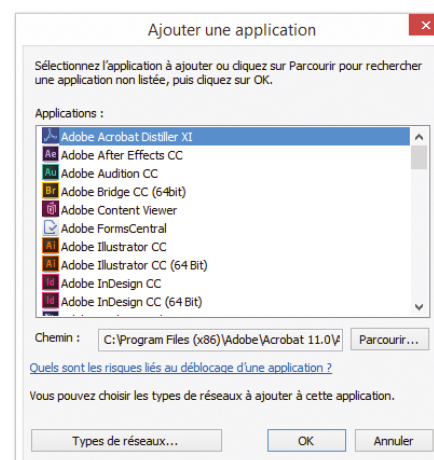
- Créer un nouvel utilisateur appelé **sshd** qui aura des permissions réduites. L'idée est celle de la séparation de privilèges. Le serveur SSH n'a pas besoin de tous les droits qu'à l'administrateur, mais uniquement ceux qui lui suffisent pour fonctionner. C'est typique de la philosophie Unix : ne jamais faire plus que nécessaire, surtout lorsqu'il s'agit de sécurité.
- Créer un utilisateur **cyg_server**. Cygwin a besoin d'une identité pour faire fonctionner les services (des programmes tournant en tâche de fond). Il vous demandera d'ailleurs de saisir un mot de passe pour cet utilisateur, deux fois. ATTENTION : vous ne verrez pas ce que vous tapez à ce moment et c'est tout à fait normal (tout comme sous GNU/Linux). La philosophie Unix se caractérise aussi par une paranoïa aiguë. Les interfaces graphiques cachent souvent les mots de passe en remplaçant chaque caractère par un * ou un point. Ceci permet toujours, à quiconque en train d'observer l'écran, de connaître le nombre de caractères du mot de passe et lui donne donc une piste pour le deviner. Dans le monde Unix, on préfère ne rien afficher du tout, c'est plus sûr.
- Générer la configuration et les fichiers (les clés) du serveur SSH.
- Installer le service dans Windows.

Si tout se passe bien, vous n'avez qu'à saisir les mots de passe et l'outil de configuration s'occupera de tout le reste, avant de terminer en vous signalant que vous n'avez plus qu'à démarrer le service en tapant **cygrunsrv -S sshd**. Chose que vous ferez dans la foulée.

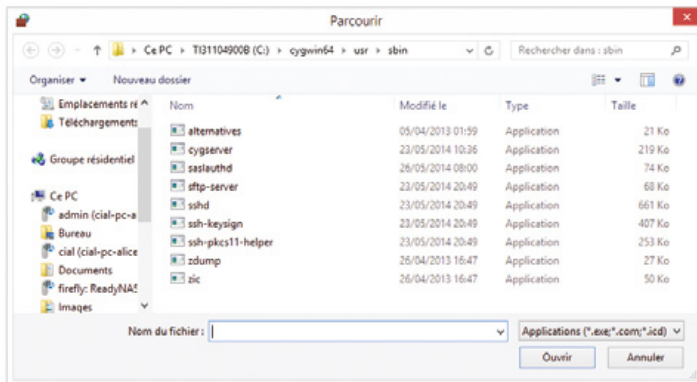
Mais ce n'est pas tout, Windows 8.1 est assez pointilleux sur la sécurité (du moins plus que Windows 7 ou XP) et il faut autoriser l'application à utiliser le réseau comme elle l'entend. Un tour dans la configuration du pare-feu Windows est nécessaire, ainsi qu'un clic sur **Modifier les paramètres**, puis **Autoriser une autre application...**



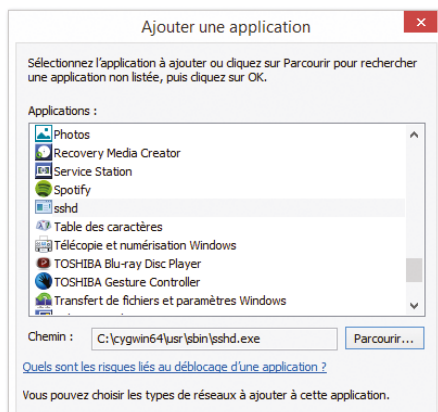
Dans la liste qui s'affiche il n'y a pas SSH, il faut donc utiliser le bouton **Parcourir** :



Là, vous devez vous rendre dans **C:\cygwin64\usr\sbin** (ou **C:\cygwin\usr\sbin**) et choisir « sshd » (qui est en fait le programme **sshd.exe**).



Après validation, « sshd » trouve place dans la liste des applications et il suffira simplement de faire **Ok** pour fermer les différentes fenêtres.



Le serveur OpenSSH fonctionne et est maintenant accessible depuis une autre machine ou par exemple une Raspberry Pi. Vous pouvez simplement tester dans un premier temps avec la commande **ssh 127.0.0.1**, ce qui aura pour effet de vous connecter à la machine elle-même. Un message s'affichera à la première tentative vous signalant que l'authenticité de l'hôte ne peut être établie. C'est normal.

SSH utilise un mécanisme complexe pour s'assurer que l'endroit où l'on se connecte est bien celui qu'il annonce être. Il vous affiche une longue suite de lettres et chiffres qui vous indique l'identité de la machine (son empreinte) et vous demande si c'est correct.

En principe, vous devez alors vérifier cette information pour être sûr de cette identité et contacter le responsable de la machine (vous même souvent) pour qu'il confirme cette empreinte (il utilisera une commande comme **ssh-keygen -lf /etc/ssh_host_ecdsa_key.pub** pour vous donner la longue

suite de lettres et de chiffres qui doit correspondre à celle qui s'affiche sur votre écran). Vous serez alors absolument certain que personne ne se fait passer pour cette machine et tente de vous tromper. On ne rigole pas avec la sécurité avec les outils du monde Unix, même quand on est sous Windows.

Vous pourrez également tester depuis un PC sous GNU/Linux ou depuis Mac OS X en précisant l'adresse de la machine, mais également depuis une Raspberry Pi ou n'importe quel nano-ordinateur sous GNU/Linux. Sous Windows, vous pouvez retrouver l'adresse de la machine en fouillant dans le panneau de configuration, mais, plus simplement, en utilisant la commande **ipconfig**, l'adresse s'affiche sous **Adresse IPv4**.

Fort de cette information, il vous suffit d'utiliser la commande **ssh** suivie du nom d'utilisateur que vous utilisez sur la machine Windows (attention les majuscules/minuscules comptent), un **@** et l'adresse IP de la machine (celle qu'on obtient avec **ipconfig**). Il est également possible de faire **ssh -l**, espace, le nom d'utilisateur, espace et l'adresse. Ceci nous donne quelque chose comme **ssh Master@192.168.10.102** ou **ssh -l Master 192.168.10.102**.

Si tout fonctionne correctement, après saisie correcte du mot de passe (que vous ne voyez pas, ici non plus, en le tapant) vous aboutirez sur la ligne de commandes Cygwin de la machine Windows. Exactement la même que celle que vous obtenez lorsque vous double-cliquez sur l'icône Cygwin.

Bien entendu, comme il s'agit d'une connexion à distance, non graphique, il n'y a pas de fenêtre à fermer pour se déconnecter. Vous devrez utiliser la commande **exit** ou le raccourci clavier **Ctrl+D** pour cela.

1. QUELQUES EXEMPLES

1.1 Copies de fichiers

Il est possible d'imaginer bien des scénarios où le fait de disposer d'une machine Windows ainsi accessible peut être intéressant. Un cas évident peut être le fait de chercher à récupérer ou à placer des fichiers sur cette machine depuis, par exemple une Raspberry Pi. Avec SSH, est installée la commande **scp** qui permet justement de copier des données au travers du réseau de manière relativement simple et surtout sécurisée.

Ainsi si vous utilisez votre Raspberry Pi et, disons, souhaitez y copier quelques images de la machine Windows, il devient totalement inutile d'éteindre la Pi, sortir la carte SD, la placer dans le lecteur de la machine Windows, faire la copie, éjecter la carte, la placer dans la Pi et démarrer à nouveau.

Tout ce qu'il vous faut pour remplacer toutes ces étapes se résume, depuis la Pi, à taper la commande **scp** suivie du nom d'utilisateur, un @, l'adresse de la machine Windows, un :, l'emplacement (le chemin) vers vos fichiers, un espace et un point. Ce qui nous donne, par exemple : **scp moimeme@192.168.0.12:/cygdrive/c/Users/moimeme/Pictures/P1020280.JPG ..** Ou, en français : copier **P1020280.JPG** du répertoire des images de l'utilisateur **moimeme** de la machine Windows à l'endroit où je me trouve sur la Raspberry Pi.

Bien entendu, de cette manière on peut également copier des répertoires complets, dans un sens comme dans l'autre. La syntaxe générale de la commande est : **scp source destination**. Dans le cas de la copie d'un répertoire, on ajoutera l'option **-r** comme « récursif ». Exemple :

```
$ scp -r Master@192.168.10.102:/cygdrive/c/Users/Master/
Pictures/* .
desktop.ini          100% 174      0.2KB/s  00:00
cygwin1.PNG          100% 47KB    47.5KB/s 00:00
cygwin10.PNG         100% 42KB    42.4KB/s 00:00
[...]
cygwin5.PNG          100% 13KB    12.9KB/s 00:00
desktopWIN8.png      100% 589KB   588.7KB/s 00:00
desktop.ini          100% 504      0.5KB/s  00:00
```

Nous venons, en une commande, de transférer depuis la Raspberry Pi tout le dossier « Images » de la machine Windows à la SD de la Pi et ce, avec les sous-dossiers et tous les fichiers qu'ils contiennent. En inversant, **scp -r .* Master@192.168.10.102:/cygdrive/c/Users/Master/Pictures/**, on procédera au transfert dans l'autre sens, du répertoire courant vers le dossier « Images » de Windows.

1.2 Comment éviter de taper un mot de passe

Si vous procédez souvent à des opérations de copie, peut-être vous lasserez-vous de devoir saisir, à chaque fois, le mot de passe. Tout aussi embêtant, le mot de passe en question est celui de l'utilisateur Windows, ce qui signifie qu'en le changeant, il faudra aussi utiliser ce nouveau mot de passe lors de la connexion SSH. Enfin, et c'est plus problématique, si vous voulez aller un peu plus loin et automatiser ces échanges, la demande de mot de passe demandera une intervention humaine indispensable. Sans vous, ça ne marche pas.

Il existe une solution sécurisée pour pallier à ces problèmes : ne pas reposer sur un mot de passe pour valider la connexion, mais utiliser un autre mécanisme. Sans entrer dans les détails, il s'agit d'utiliser un secret que seul le compte utilisateur de la Raspberry Pi possédera. Cette solution se compose de deux clés, une publique et une secrète, liées entre elles par une relation mathématique.

On commence donc, sur la Raspberry Pi par créer une paire de clés avec :

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/pi/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/pi/.ssh/id_rsa.
Your public key has been saved in /home/pi/.ssh/id_rsa.pub.
The key fingerprint is:
6b:5e:3d:24:5b:e4:6f:b1:9d:18:0a:ca:13:59:50:75 pi@raspberrypi
The key's randomart image is:
+--[ RSA 2048 ]-----+
|          . . . . . E          |
|          . .          |
|          . .          |
|         o o          |
|        oS.. +..       |
|       . o..*..o+.     |
|      +o o.o.o.+..    |
|       o.. o          |
|          .          |
+-----+-----+
```

On vous demandera un mot de passe lors de cette opération. Il ne s'agit pas du mot de passe de votre compte sur la Pi ou sous Windows. C'est un mot de passe qui protège l'accès à la clé secrète (un peu comme un mot de passe qui protège un fichier Zip). Si vous comptez pouvoir vous connecter à l'autre machine sans mot de passe, vous pouvez simplement valider deux fois pour créer une clé qui ne sera pas protégée. Le contrecoup de cette « facilité » est le suivant : si on vous prend votre carte SD et que la clé n'est pas protégée par un mot de passe, on pourra accéder sans mot de passe à la machine Windows. Gardez cela à l'esprit et protégez donc votre carte SD. Bien entendu, si vous la perdez ou avez un doute, vous pouvez parfaitement annuler cette configuration et supprimer cet accès sans mot de passe, même sans la carte SD en question.

Sur votre Pi, vous voici en possession de deux clés. L'idée est de confier l'une d'elles, la clé publique, à la machine Windows. En procédant de la sorte, le SSH sur la Pi devra utiliser la clé privée qui correspond à la clé publique copiée sous Windows pour se connecter. La relation mathématique entre les deux clés est l'élément qui assure la sécurité. Il est impossible à partir de la clé publique de déduire ou retrouver la clé privée, du moins pas en un temps raisonnable et avec la puissance de calcul actuellement disponible pour l'espèce humaine.

Il n'y a rien à configurer pour mettre cela en place, une commande spécifique existe. Ainsi, sur la Raspberry Pi, utilisez :

```
$ ssh-copy-id Master@192.168.10.102
Master@192.168.10.102's password:
Now try logging into the machine, with
"ssh 'Master@192.168.10.102'", and check in:
  ~/.ssh/authorized_keys
to make sure we haven't added extra keys
that you weren't expecting.
```

Le mot de passe de l'utilisateur (ici c'est « Master ») sur la machine Windows vous sera demandé (une dernière fois) pour cette connexion afin d'envoyer la clé publique. Si, à présent, vous tentez une nouvelle connexion, SSH par défaut essaiera en premier le mécanisme de clés :

```
$ ssh Master@192.168.10.102
Last login: Tue Jun 24 14:32:11 2014
from 192.168.10.160
Master@pao3 ~
```

Plus besoin de mot de passe ! Comme l'utilisateur sur la Pi possède la clé secrète correspondant à la clé publique installée sous Windows, la connexion est autorisée. Bien entendu, si on essaie depuis ailleurs (un PC GNU/Linux, un Mac, une autre Pi, une autre carte SD, un autre compte), ceci ne fonctionnera pas et le mot de passe sera demandé. Il en va de même si vous effacez la SD pour réinstaller le système, la clé privée sera perdue et il faudra recommencer en utilisant le mot de passe à nouveau.

Si vous n'avez pas défini de mot de passe pour protéger la clé privée, il devient possible non seulement de se connecter rapidement à la machine Windows, mais également d'intégrer une telle connexion ou un transfert de fichiers dans un script sur la Raspberry Pi. Ceci fonctionnera sans problème puisque l'intervention de l'utilisateur n'est plus nécessaire.

Rappelons encore une fois qu'une clé privée non protégée par un mot de passe est quelque chose à garder à l'abri des regards indiscrets : ne passez JAMAIS votre carte SD à quelqu'un, ne laissez personne d'autre utiliser ce compte sur la Pi et, si vous effacez la SD pour un autre usage, utilisez un outil d'effacement sécurisé, sinon même avec un formatage, les données peuvent être potentiellement récupérées et donc votre clé privée aussi.

Pour retirer cette configuration et interdire l'utilisation de la connexion sans mot de passe, il vous faudra éditer le fichier `.ssh/authorized_keys` sur la machine Windows et supprimer la ligne correspondant à la Raspberry Pi. Dans le doute, vous pouvez également, tout simplement, effacer le fichier, mais ceci retirera alors toutes les autorisations pour tout le monde.

Bien entendu, ce que nous venons de décrire pour procéder à une connexion depuis la Pi vers Windows fonctionne parfaitement dans l'autre sens si on inverse les rôles et qu'on procède de même pour autoriser les connexions sans mot de passe depuis Windows (ou GNU/Linux ou Mac OS X) vers la Raspberry Pi.

1.3 Et si on simplifiait un peu

Se passer de saisir le mot de passe est une chose, devoir à chaque fois spécifier le nom d'utilisateur et l'adresse de la machine en est une autre. Nous pouvons simplifier cela en créant un « profil » de configuration. Pour ce faire, toujours sur la Pi, mais là aussi cela marche dans les deux sens, il nous suffit de créer un fichier `config` dans le répertoire `.ssh` de l'utilisateur courant, contenant quelques informations utiles :

```
Host win
  Hostname 192.168.10.102
  HostKeyAlias win
  Port 22
  User Master
```

Vous pouvez utiliser n'importe quel éditeur de texte pour créer ce fichier et si vous voulez faire cela « à la Unix », vous pouvez simplement utiliser `cat > ~/.ssh/config` sur la ligne de commandes. C'est une solution simple et rapide (sans éditeur), mais vous ne pouvez pas revenir sur une ligne validée et vous devez terminer la saisie en utilisant la combinaison Ctrl+D.

Dans ce fichier, on trouve :

- **Host** suivi d'un nom qui sera celui du profil créé, puis, décalé avec des espaces ou une tabulation :
- **Hostname** qui précise l'adresse de la machine « cible »,
- **HostKeyAlias** est un nom utilisé pour que SSH associe l'identité de la machine, vous savez, le message d'avertissement de la première connexion,
- **Port** qui par défaut sera 22,
- **User** pour spécifier le nom d'utilisateur de la machine cible.

Vous pouvez parfaitement vous limiter à **Host**, **Hostname** et **User** si vous le souhaitez.

Ceci fait, vous pouvez immédiatement utiliser le nom du profil avec **ssh** ou **scp**. Inutile de spécifier l'adresse de la machine ou le nom d'utilisateur, tout ceci est dans le profil. En combinant cela avec la connexion sans mot de passe, vos accès à la machine Windows se feront simplement avec, par exemple ici, **ssh win** sans plus de formalité. Le nom **win** remplaçant avantageusement **Master@192.168.10.102**. SSH vous demandera, lors de cette première connexion avec le profil, de confirmer encore une fois l'identité de la machine cible. Ceci vient de l'utilisation de **HostKeyAlias**. SSH connaît bien la machine si vous vous y êtes déjà connectée, mais pas sous ce nom.

1.4 Jour, nuit, jour, nuit...

En ayant tout cela en place, il devient possible de faire un geste pour la planète (si, si). Une Raspberry Pi est équipée d'un fusible réarmable ou *polyfuse* en anglais. Celui-ci est présent afin de limiter le courant traversant la carte de manière à rester dans les

caractéristiques définies par le constructeur. Ce fusible est limité à 700mA (*hold current*) et il est donc conseillé de ne pas dépasser cette valeur, car le fusible, si davantage de courant le traverse, va chauffer et augmenter en résistance. Au-delà de 1,1 A, celui-ci se bloquera. On peut donc, au sens large du terme, en conclure la puissance maximum possible qu'il sera jamais possible d'atteindre avec une Pi : 1,1 ampères x 5 volts = 5,5 watts.

Un PC standard, quant à lui, consomme joyeusement de 300 à 500 watts (parfois plus pour des configurations de *gamers*), soit un rapport de 100. En dehors de considérations écologistes, il est peu judicieux, financièrement, de laisser allumer un PC qui n'est pas utilisé, même si des solutions d'économies d'énergie existent dans les systèmes modernes. La meilleure solution est encore d'éteindre la machine lorsqu'elle ne sert pas. Pourtant, tantôt, il peut être plaisant d'accéder à certains fichiers sur un PC alors que celui-ci est à l'arrêt : la solution c'est une Pi, Cygwin et SSH.

Il existe une fonctionnalité peu connue souvent présente dans la configuration des PC et plus particulièrement dans le BIOS (le *setup*) de nombreux PC : le *Wake on LAN* ou WoL. Ceci ne date pas d'hier et l'idée est la suivante : l'interface réseau d'une machine peut rester alimentée, consommant très peu de courant, alors que le reste de l'ordinateur est éteint. Avec la configuration appropriée, il est possible d'envoyer un « paquet magique » à l'interface au travers du réseau. Quand celle-ci recevra le paquet, elle démarrera l'ordinateur dans son ensemble. Notez qu'il ne s'agit pas d'une sortie de veille, mais bien d'un démarrage alors que la machine est physiquement à l'arrêt.

Pour envoyer un tel « paquet magique », comme le système est éteint, on ne peut utiliser l'adresse IP. Dans cet état, il n'y a pas de réseau à ce niveau de fonctionnalité. On utilise alors la vraie adresse matérielle (adresse physique) propre à l'interface : c'est l'adresse MAC, qui prend la forme d'un ensemble de six valeurs présentées ainsi : **AA:BB:CC:DD:EE:FF**. Pour envoyer le paquet depuis notre Raspberry Pi, nous utilisons l'outil **etherwake** en spécifiant cette adresse physique.

Pour connaître l'adresse sous Windows, lancez le terminal Cygwin (ou le **cmd.exe** de Windows) et utilisez la commande **ipconfig /all**. Dans la masse de données qui s'affiche, vous trouverez quelque chose comme :

```
[...]
Carte Ethernet Ethernet :
  Suffixe DNS propre à la connexion. . . : ed-diamond.com
  Description. . . . . : Contrôleur Realtek PCIe GBE Family
  Adresse physique . . . . . : 00-8C-FA-62-C0-D4
  DHCP activé. . . . . : Oui
  Configuration automatique activée. . . : Oui
  Adresse IPv6 de liaison locale. . . . : fe80::2c6a:9b35:7072:3a4a%4 (préféré)
  Adresse IPv4. . . . . : 192.168.10.102 (préféré)
  Masque de sous-réseau. . . . . : 255.255.255.0
[...]
```

C'est la ligne « Adresse physique » qui nous intéresse. Fort de cette information, vous pourrez utiliser **etherwake** pour provoquer le démarrage, après configuration dans le BIOS. Cette configuration est accessible via une touche spéciale au démarrage, dépendante de la marque et du modèle de votre ordinateur, [suppr], [F2] ou encore [F12]. Consultez la documentation de votre machine pour la connaître. Farfouillez ensuite dans la configuration pour trouver l'option adéquate et surtout, ne changez rien d'autre !

Ceci fait, éteignez votre PC normalement, puis depuis la Raspberry Pi faites :

```
$ sudo etherwake 00:8C:FA:62:C0:D4
```

00:8C:FA:62:C0:D4 est, bien entendu, l'adresse physique (MAC) de l'interface réseau de la machine Windows. Peu après la validation de cette commande, la machine devrait démarrer, et ce, quel que soit son système d'exploitation. Après le temps classique nécessaire au démarrage de Windows, vous voici en mesure d'accéder, via SSH à votre PC.

Là vous avez accès à tout le système comme précédemment. Vous pouvez par exemple envoyer ou récupérer des fichiers, etc. Une fois vos différentes manipulations terminées, peut-être souhaiteriez-vous ne pas laisser le PC allumé. Rien de plus simple, vous pouvez l'éteindre avec une commande Windows : **shutdown**. Il ne vous est même pas nécessaire de la saisir après connexion via SSH, vous pouvez directement spécifier la commande et ses options avec **ssh** :

```
$ ssh win "shutdown /s"
```

Ce qui revient exactement au même qu'une connexion avec **ssh win**, la saisie et validation de la commande et une déconnexion avec **exit**. L'option utilisée ici avec **shutdown** est **/s** pour un arrêt (**/r** pour un redémarrage). Notez qu'il s'agit là d'une commande Windows qui n'a rien à voir avec Cygwin (sous GNU/Linux on utiliserait **sudo halt**).

Avec ces quelques informations, il devient possible de faire des choses très amusantes comme, par exemple, prendre des photos avec la caméra de la Raspberry Pi à intervalles réguliers, puis une fois par jour, démarrer la machine Windows, copier toutes les photos dans un répertoire et l'éteindre.

2. POUR FINIR

Cygwin vous offre l'opportunité de mélanger Windows avec des outils puissants et une vraie ligne de commandes. Ceci vous permettra à la fois d'améliorer vos compétences dans ce domaine et de vous ouvrir à des possibilités qui, sans Cygwin, vous seraient totalement inaccessibles. Même si ce que nous venons de voir est relativement technique et puissant, nous n'avons fait qu'effleurer le sujet. Grâce à Cygwin, la quasi-totalité de ce qu'il est possible de faire sous GNU/Linux est réalisable sous Windows et ce nativement, sans émulation et gratuitement... **DB**

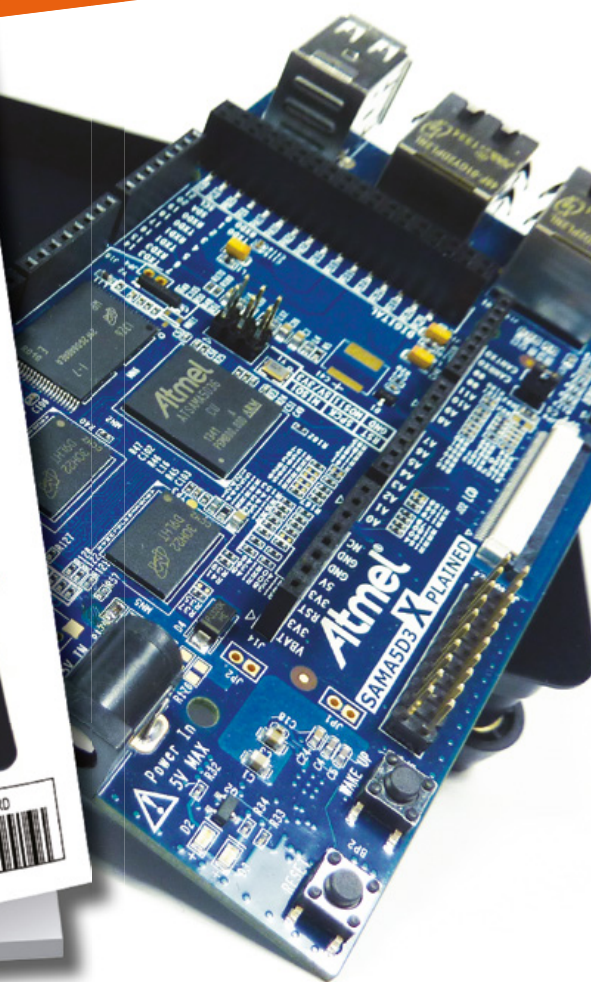
NE MANQUEZ PAS OPEN SILICIUM N°13 !

FOCUS :

DÉMARREZ AVEC YOCTO/POKY

...sur Atmel

SAMA5D3 Xplained



**DISPONIBLE CHEZ VOTRE
MARCHAND DE JOURNAUX ET SUR :
www.ed-diamond.com**



innorobo

1-3 Juillet
Lyon - France

Les technologies et systèmes robotiques apportent des solutions à nos enjeux sociétaux planétaires et contribuent à l'amélioration de notre qualité de vie. Sur cette conviction partagée, nous avons construit et développé **un sommet robotique de 3 jours** qui réunit **un vaste écosystème d'affaires** et qui a pour objectif **la mise sur le marché des innovations robotiques** le plus rapidement possible.

Innorobo™ 2015 développera 6 thématiques majeures d'applications robotiques

MEDICAL & HEALTH



SMART CITIES



SMART HOMES



FACTORIES OF THE FUTURE
SERVICE TO INDUSTRIES



TECHNOLOGIES & FORESIGHT



Artificial Intelligence
Materials, Mechatronics
Nano Technologies & Electronic
Software and Components
Big data, HRI
Cloud robotics

FIELD ROBOTICS



Nouveautés 2015

Village des Développeurs et Espace Makers

Avez-vous déjà pensé à **programmer des robots** ? Vous maîtrisez le C, Java, ou bien des environnements types comme **ROS, URBI, Matlab ou bien Chorégraphe** ? Venez montrer vos talents et **échanger avec des experts** !

Passionnés d'électronique et de mécanique ? Venez exposer à Innorobo pour montrer votre savoir faire, dans l'unique lieu en Europe réunissant imprimantes 3D et robots, sociétés robotiques, labos de recherche, startups, développeurs et passionnés.

On vous attend !

www.innorobo.com



SYROBO

ONLY LYON
GREATER LYON

Rhône-Alpes
Région

GDR
robotique

INNOECHO