FASCICULE GRATUIT

elektor

l'électronique imaginative

Nº 333
MARS 2006
mensuel
(B) 6,55 € • (CH) 11,25 FS (CAN) 8,35 SCan.
DOM surface 7,00 € • DOM avion 8,75 €
ISSN 0181-7450
www.elektor.fr

« Briques FPGA »

Module FPGA + carte d'expérimentation

AVEC POSTER

Le monde des microcontrôleurs

- *Logger* bon marché
- Kits de développement

- Le 6502 – qui s'en souvient ?
- Chargeur globe-trotter

Carte d'application pour RC8/13

L 19624 - 333 - F: 5,95 €

elektor
l'électronique imaginative

Après les µP, les µC, les CPLD et tutti quanti, voici l'ère de la FPGA…

Un coup d'oeil à la couverture et au sommaire ci-contre, il ne vous en faudra pas plus pour vous convaincre que le thème de ce numéro a pour cible le monde des microcontrôleurs, mais cette fois sous leur variété la plus récente mais aussi la plus puissante, les FPGA.

Il faut un certain courage à un magazine d'électronique pour tenter une telle aventure, essayer de vulgariser le modèle le plus complexe des microcontrôleurs modernes. Si, il y a quelques années (plus de 25 ans, ce qui ne nous rajeuni pas), nous nous étions essayés au premier modèle de microprocesseur disponible pour le grand public, le SC/MP et quelques mois plus tard nous avions tâté au 6502, au 8032, 8051, 8052AH Basic 87XX, etc. Les PIC de tout acabit et les Atmel en tous genres ont trouvé (et trouvent toujours encore) place dans nos réalisations. Nous visons plus haut cette fois, et vous proposons de vous faire les dents sur les FPGA.

Un premier article, **« Briques » FPGA flexibles**, devrait servir de sésame pour nous permettre de pénétrer dans ce monde réservé, jusqu'à tout récemment, aux professionnels dans leurs laboratoires protégés. Mais que faire d'un module FPGA aussi performant si l'on ne dispose pas des moyens de le mettre à l'épreuve ? C'est la fonction de l'article intitulé **Carte d'expérimentation FPGA**.

Nous aurons sans doute l'occasion, dans un futur plus ou moins proche, de vous proposer des applications au coeur desquelles battra notre **module FPGA**.

Nous vous proposons ensuite une sorte de vue panoramique des **Kits de développement** pour microcontrôleurs. Nous ne prétendons pas écumer le marché, cela serait une opération impossible, même une simple liste d'énumération occuperait de très nombreuses pages en caractères minuscules.

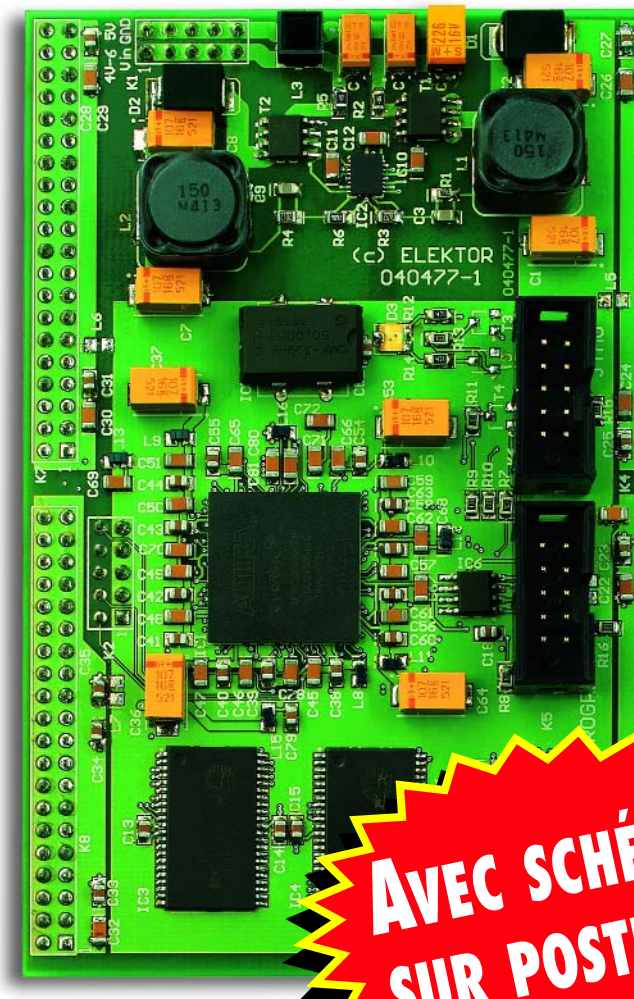
Nous vous présentons ensuite une **carte d'application pour le R8C/13**. Il nous faut reconnaître que nous avons été quelque peu débordés par le succès de cette minuscule **platine à R8C/13** de Renesas décrite dans le numéro de février 2006, alors un peu de patience lors de vos commandes.

La nouvelle rubrique lancée fin 2005, permet à **Jeroen**, un jeune Néerlandais, de nous présenter certains de ses projets nés de produits existants, un ***Logger* bon marché** cette fois.

Nous clôturons ce numéro par deux articles inhabituels, une analyse du **dépouillement de notre enquête** concernant les logiciels de CAO lancée suite à la parution du numéro de novembre (avec son DVD !). et un **Bruits de labo** né dans les parages de l'Himalaya.

Guy Raedersdorf
Rédacteur en chef

22



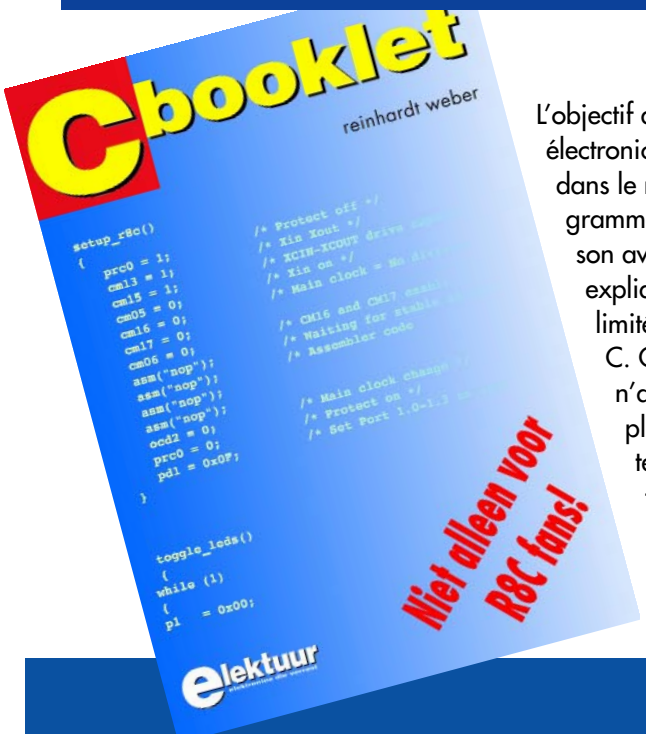
Les techniques les plus modernes pour tous !

Qui arrêtera la marche des FPGA ? Personne. Jusqu'à tout récemment, travailler avec ce type de « super-composant » était réservé aux spécialistes de développements high-tech. Les choses ont bien changé avec la chute des prix et la mise à disposition de logiciels de développement adéquats. Il nous a paru temps de vous présenter cette technologie dans les colonnes d'Elektor. Nous avons à cet effet développé un module FPGA qui constituera le coeur numérique de différents projets proposés dans les prochains numéros de ce magazine.

« Briques » FPGA flexibles

AVEC SCHÉ
SUR POST

Extra : Mini-fascicule avec cours C succinct

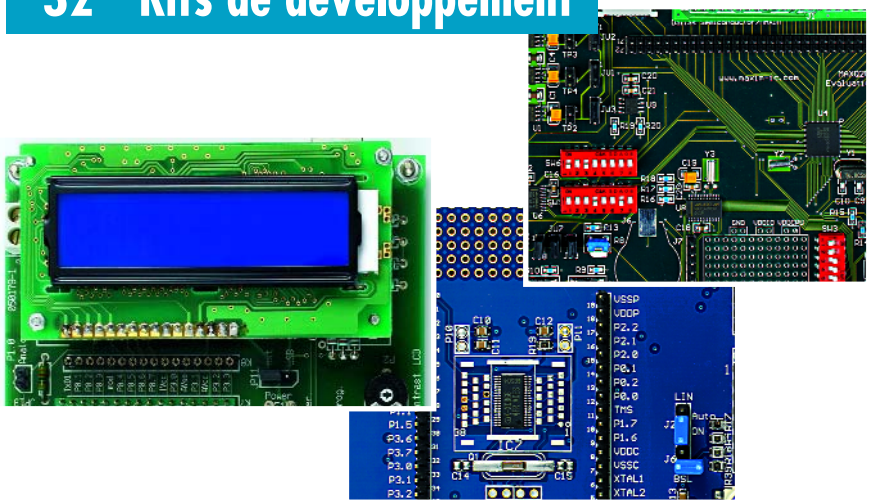


L'objectif de ce fascicule est d'aider les électroniciens dans leurs premiers pas dans le monde du langage de programmation C, surtout en combinaison avec les microcontrôleurs. Ceci explique que nous nous soyons limités aux éléments de base de C. C'est à dessein que nous n'avons pas abordé des sujets plus complexes tels que pointeurs, matrices, chaînes, structures, unions, etc. Un outil pratique pour le programmeur en C débutant…

SOMMAIRE

29e année
Mars 2006
№ 333

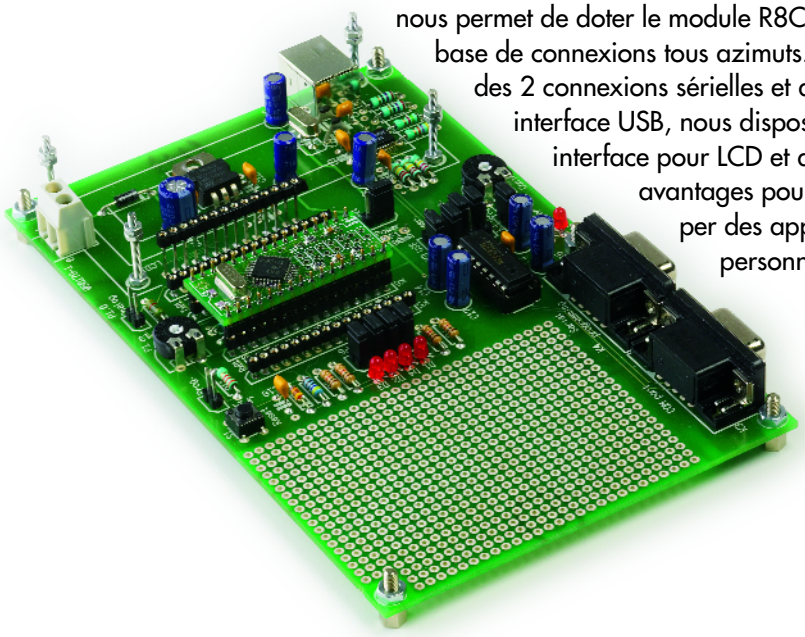
32 Kits de développement



Qu'existe-t-il, quoi prendre ?
Dans ce numéro consacré à ras bord aux FPGA et aux microcontrôleur, nous ne pouvons pas ne pas en parler, de quoi ?, des kits de développement. Nous allons nous intéresser à une sélection de kits intéressants, mais surtout abordables, en veillant à ne pas sélectionner des kits ne pouvant servir que dans un cadre professionnels mais aussi utilisables par l'amateur.

44 « Camp de base » pour débutants

Carte d'application pour R8C/13
Le numéro de février nous a déjà montré comment animer, sans peine et à peu de frais, le R8C/13. La présente carte d'application nous permet de doter le module R8C/13 d'une base de connexions tous azimuts. En plus des 2 connexions sérielles et d'une interface USB, nous disposons d'une interface pour LCD et d'autres avantages pour développer des applications personnelles.



elektor
l'électronique imaginative

29ème année, N° 333
mars 2006

ISSN 0181-7450
Commission paritaire N° 1004U8313

ELEKTOR / SEGMENT B.V.

1, rue de la Haye • BP 12910
95731 Roissy CDG Cedex
Tél. : (+33) 01.49.19.26.19
Fax : (+33) 01.49.19.22.37
Internet : www.elektor.fr

Numéro de compte : 002-007-69-901
IBAN : FR76 1873 9000 0100 2007 6990 192
BIC : ABNAFRPP
Monnaie : Euro
Branche ABN AMRO : Paris, France

Elektor désire être une source d'inspiration pour ses lecteurs, les intéresser à l'électronique, par la description de projets à faire soi-même, et les tenir au courant des développements en électronique et en micro-informatique.

Elektor paraît 11 fois, le numéro de juillet/août est un numéro double.

Il existe, sous le nom Elektor, des éditions anglaises, allemande et française, et sous celui d'Elektuur, une édition néerlandaise. Elektor est vendu dans plus de 50 pays.

Conformément à la loi "Informatique et Liberté", vous bénéficiez d'un droit d'accès et de rectification des données vous concernant. Sauf refus écrit de votre part auprès du service abonnement, ces informations pourront être utilisées par des tiers.

Rédacteur en chef international :
Mat Heffels

Rédacteur en chef France :
Guy Raedersdorf
(redaction@elektor.fr)

Rédactions :
Harry Baggen, Thijs Beckers, Jan Buiting, Ernst Krempelsauer, Jens Nickel

Secrétariat de rédaction :
Hedwig Hennekens

Rédaction technique :
Karel Walraven (chef)
Ton Giesberts (concepteur)
Paul Goossens (concepteur)
Luc Lemmens (concepteur)

Maquette et graphisme :
Ton Gulikers, Giel Dols

Directeur/éditeur :
Paul Snakkers

Responsable marketing :
Margriet Debeij

Administration des ventes :
(ventes@elektor.fr)

Publicité :
SL Régie - Sophie Lallonder
Tél : 01.53.41.07.55
Fax : 01.42.52.20.80
E-mail : sophie.lallonder@wanadoo.fr

Abonnements Suisse :
Sono Light Import
Champs-Montants 16b
CH-2074-Marin-Epagnier
Tél : 032-710.16.60
Fax : 032-710.16.63
E-mail : admin@sonolight.ch

Elektor est édité par Segment B.V.
Siège social : Peter Treckpoelstraat 2-4
6191 VK Beek (L), Pays-Bas
RC Heerlen, nr. 35306

Imprimé aux Pays-Bas par
Tijl-Offset – Zwolle

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

Il est possible de faire démarrer un abonnement à tout moment. Nous vous rappellerons en temps utile l'approche de la fin de votre abonnement. La méthode la rapide et la moins chère de vous abonner est de le faire par le biais de notre site Internet www.elektor.fr/abo, mais vous pouvez également le faire à l'aide du bon de commande se trouvant en fin de magazine.
Il est possible de commander d'anciens numéros dans la limite de leur disponibilité (cf. le bon de commande, leur prix est celui d'un numéro à l'unité).

Veuillez SVP nous fournir un changement d'adresse au moins 3 semaines auparavant en mentionnant votre numéro d'abonné (cf. le label accompagnant votre magazine), l'ancienne et la nouvelle adresse.

Le département Clients est accessible les jours ouvrables de 9h00 à 12h30 et de 13h00 à 16h30.

Si vous avez des questions concernant votre abonnement, vous pouvez appeler ce département au numéro 01.49.19.26.19

Pour le traitement de votre abonnement, Elektor vous demande des données personnelles. Conformément à la loi « Informatique et Liberté », vous bénéficiez d'un droit d'accès à ces données et vous pouvez en demander la rectification. Sauf refus écrit de votre part auprès du service Abonnement, ces informations pourront être utilisées par des tiers.

Prix au numéro

France	5,95 €
DOM Surface	7,00 €
DOM Avion	8,75 €
Belgique	6,55 €
Suisse	11,25 FS
Canada	8.35 $Can

Abonnement d'un an standard

France	62,50 €
Belgique	68,90 €
Suisse	117 FS
DOM Surface	82 €
DOM Avion	107 €
Étudiant	
France	*50 €*
Belgique	*55,12 €*

Abonnement de 2 ans standard

France	112,50 €
Belgique	124,02 €
Suisse	210,60 FS
DOM Surface	147,60 €
DOM Avion	192,60 €
Étudiant	
France	*90 €*
Belgique	*99,22 €*

Abonnement PLUS d'un an

France	72,45 €
Belgique	78,85 €
Suisse	139 FS
DOM Surface	91,95 €
DOM Avion	116,95 €
Étudiant	
France	*59,95 €*
Belgique	*65,07 €*

Abonnement PLUS de 2 ans

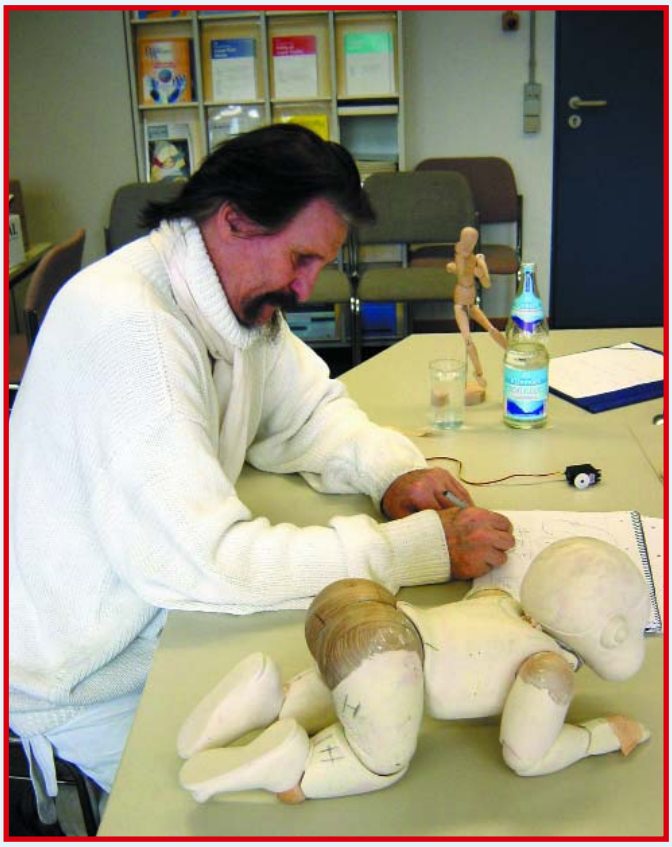
France	132,40 €
Belgique	143,92 €
Suisse	254,60 FS
DOM Surface	167,50 €
DOM Avion	212,50 €
Étudiant	
France	*109,90 €*
Belgique	*119,12 €*

Abonnements
E-mail : abonnements@elektor.fr

Commandes/Ventes
E-mail : ventes@elektor.fr

Sous réserve de modification de prix.

Un bébé à cerveau ELEKTOR



Le concepteur de réputation mondiale Luigi Colani avec le prototype du bébé (Photos : ITEC, chaire IAIM Prof. Dillmann).

S'il faut en croire les milieux bien informés, de moins en moins de jeunes optent pour des professions à caractère technique. L'électronicien type est une espèce en voie de disparition est un autre leitmotiv répété comme une mantra. Le bébé que nous vous présentons ne pouvait espérer mieux : il est né du croisement d'un concepteur technique et d'un groupe d'informaticiens portés à la technique, ne crie pas, ne salit pas de langes et n'a rien d'autre dans la tête que des puces, bits et autres octets.

Trêve de plaisanteries, tout ce qu'il y a de vrai dans le paragraphe précédent est que ce fameux bébé possède plusieurs « pères ». L'un d'entre eux est le professeur Luigi Colani [1], un concepteur de véhicules, d'appareils photo et de bien d'autres appareils technique, de réputation mondiale, et le professeur Rüdiger Dillmann, l'un des titulaires de l'ITEC (*Instituts für TEchnische Informatik* de l'Université de Karlsruhe (RFA) [2]. Comme le concepteur-étoile habite lui aussi dans cette belle ville du Bade-Wurtenberg, ils se retrouvèrent lors d'une journée Portes Ouvertes de la dite ITEC. Si le Maître trouvait les robots humanoïdes un sujet très intéressant, aucun des modèles existants ne trouvait, optiquement, grâce à ses yeux. Dillmann para l'attaque en suggérant à Colani de concevoir un Robot pour son Institut. Ceci tombait fort bien, le concepteur-étoile pouvait proposer quelque chose d'adéquat. Peu de temps auparavant il avait créé une poupée baptisée Kawa-I (soumis en japonais) qui aurait pu être un compagnon de jeu pour le fameux robot canin de Sony, AIBO [3]. L'« être » en question possédait bien les rondeurs typiques des créations de Colani, mais se déplaçait en rampant très maladroitement. Kawa-I était pour ainsi dire inanimé dans le sens propre du terme.

Aussitôt dit aussitôt fait.

L'échoppe du concepteur créa, à partir du corps de poupée massif, un moulage positif en polyester, Dilleman et ses équipiers Tamim Asfour et Tilo Gockel initièrent un bloc de travaux pratiques pour 8 étudiants qui se virent confier la mission d'apprendre au « bébé » à ramper [4].
L'équipe commença par créer un modèle de volume par CAO. Il devait être possible ainsi de vérifier par logiciel si les mouvements du « petit monstre » pouvaient supporter les critiques des parents d'enfants en bas âge de l'équipe. On ne sera guère étonné d'apprendre que les informaticiens visualisèrent des heures et des heures de vidéos de bébés !

Il était temps de passer aux choses sérieuses et de se faire la main sur la poupée de plastique. La « carrosserie » de Kawa-I constitua longtemps un casse-tête. L'épaisseur de l'épiderme n'était pas uniforme et le matériau difficile à travailler nous dit Gockel. La place à l'intérieur du corps était plus qu'exiguë, de sorte que l'on ne plaça dans les articulations des genoux et des coudes que les moteurs et les ressorts correspondants indispensables. Les informaticiens techniciens optèrent pour des servomoteurs utilisés dans le modélisme. La servo utilisée, une HS-5945MG de Hitec est pilotée en numérique et possède un couple de 130 Ncm; une structure métallique stable fut intégrée au corps pour en accroître la rigidité [5]. Les moteurs furent ensuite montés dans la poitrine, les cuisses et les hanches de Kawa-I.

Comme il fallait non seulement que le bébé se laisse télépiloter mais qu'il soit également en mesure de se « déplacer » de manière autonome, il a fallu développer un automate à microcontrôleur. Le hasard voulut que Gockel possédait déjà une bonne expérience avec la carte à 89S8252 Flash d'Elektor [6]. À noter qu'il est co-auteur d'un livre (en allemand) sur les Robots publié récemment et reposant sur une platine à base de AT89S8252 [7,8]. Ce système à microcontrôleur bien pourvu de périphériques devait fort bien pouvoir se tirer d'affaire, le seul problème était en fait sa taille qui ne permettait pas de l'intégrer dans la tête du robot (lieu privilégié de l'intelligence même si elle est artificielle). Elle fut quelque peu « rabotée » se souvient Cockel. Le bloc d'accumulateurs lui aussi n'avait pas trouvé place à l'intérieur du bébé-modèle qui fut pour cette raison doté d'un petit sac à dos.

La communication avec le cerveau du bébé se faisait, comme à l'époque de Frankenstein, par

Bibliographie

[1] www.colani.de
[2] http://wwwiaim.ira.uka.de
[3] www.aibo-europe.com
[4] T. Gockel, T. Asfour, J. Schröder, L. Colani, R. Dillmann: « Kawa-I krabbelt », document ITEC (en allemand, à télécharger depuis www.elektor.fr et voir le mois de publication)
[5] www.hitecrc.de
[6] « Carte 89S8252 Flash », Elektor 12/01, page 20 et suivantes; source : www.elektor.fr
[7] www.atmel.com
[8] T. Gockel, R. Dillmann, A. Bierbaum, A. Piaseczki, J. Schröder, P. Azad : « Embedded Robotics - Das Praxisbuch », Elektor-Verlag, Aachen 2005; source : www.elektor.de
Voir en outre : http://wwwiaim.ira.uka.de/embedded-robotics
[9] www.elektor.fr et voir le mois de publication

RS-232 à une vitesse de 19,2 Kb/s. Pour un fonctionnement autonome, les mouvements devant de permettre au robot de ramper étaient stockées dans la Flash du microcontrôleur. Le protocole ultra-simple se compose de 6 octets dont les 5 derniers donnent chacun la valeur de consigne des 5 moteurs (valeurs allant de 0 à 255). La commande des moteurs se fait en synchrone par un signal MLI (Modulé en Largeur d'Impulsion) généré par l'un des temporisateurs de l'Atmel; il est possible, en théorie, de commander un maximum de 8 actuateurs.
Les mouvements sont régis par un principe « sioux ». Toutes les servos démarrent et s'arrêtent toujours simultanément. Pour cela on calcule, à partir de la position de départ et de la position de fin des membres, le chemin à parcourir et on identifie le membre à la durée de déplacement la plus importante. La vitesse des différents autres moteurs est diminuée en conséquence de manière à ce qu'ils s'arrêtent au même moment. La réduction de vitesse diminue sensiblement les contraintes subies par les membres du bébé.

Outre une possibilité de reptation autonome, le cahier des charges requérait également un pilotage par PC. On utilisa à cet effet une carte Flash externe. Dans ce mode de fonctionnement le robot est alimenté par une sorte de cordon ombilical depuis une alimentation de laboratoire.

Et, surprise, Kawa-I rampe vraiment ! Pour le prouver, les étudiants on fait un petit film (que l'on peut télécharger depuis notre site [9].

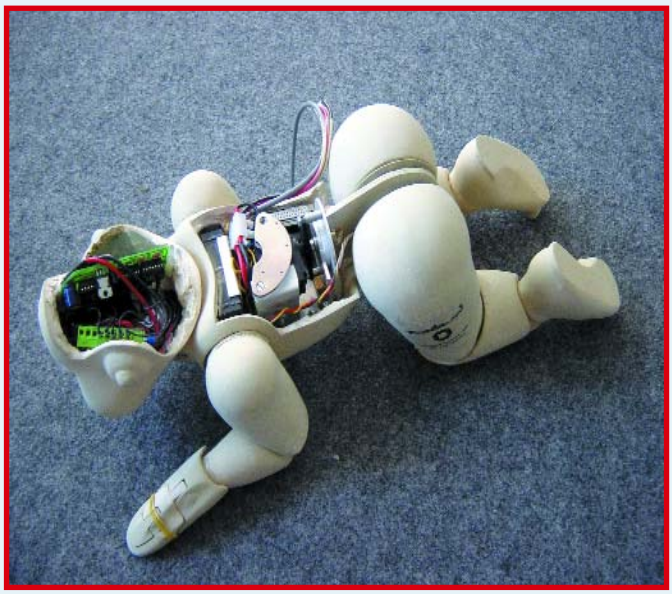
Colani fut heureux des progrès moteurs rapides de son dernier-né. Le Maître embarqua le petit pour le présenter lors d'un Salon à Kyoto au Japon, où il continue de ramper… Que les scientifiques se rassurent, notre bébé va avoir plusieurs petits frères, mais de couleur cette fois puisque faits en fibre de carbone ce qui offre plus de place dans les entrailles du bébé-robot.

Hitec a promis de nouvelles servos du type HSR-5995TG poursuit Gockel, servos dotées de moteurs hautes-performances spécialement développés pour les applications de robotique. Tout ceci donne des idées à l'équipe de l'ITEC qui envisage de doter les frères de Kawa-I de servos additionnelles pilotées par une seconde carte Flash montée en gigogne sur la première (la ligne RXD de l'interface RS-232 de la seconde étant prise en parallèle sur la ligne correspondante de la première carte, de sorte que les 2 cartes puissent réagir à des paquets de données différents tout en étant reliées à la même interface du PC.

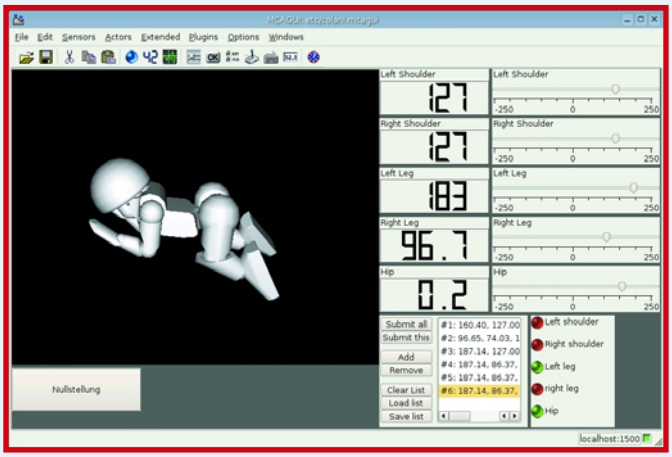
Pourquoi ne pas passer à l'étape suivante et faire marcher les bébés ? On parle même de donner à Kawa-I une petite soeur qui serait dotée d'un caméscope numérique. Et si on les faisait parler… Les bébés de Colani à cerveau Elektor feront sans doute encore (souvent) parler d'eux.

(067036-I)

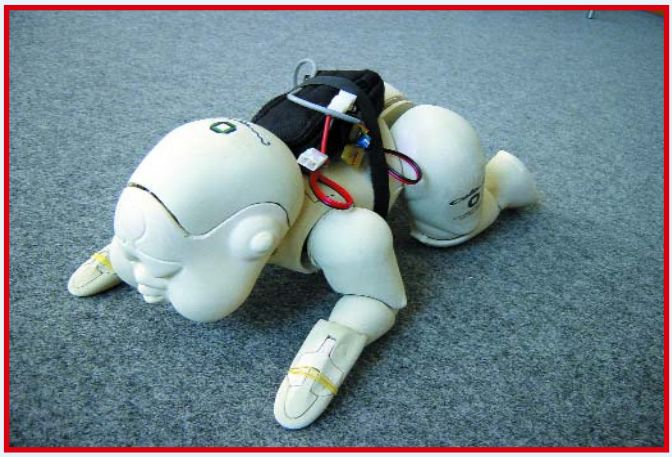
Les « pères » de Kawa-I (de gauche à droite, devant) : Lars Pätzold, Manfred Kröhnert, Julian Schill, Luigi Colani, Rüdiger Dillmann, (derrière :) Sebastian Schmidt, Anatoli Barski, Stefan Sellhusen, Günther Falk, (debout :) Markus Haas, Tilo Gockel, Tamim Asfour.



Les volumes libres du corps de plastique sont bien occupés. On voit dans la tête la carte Flash d'Elektor modifiée.

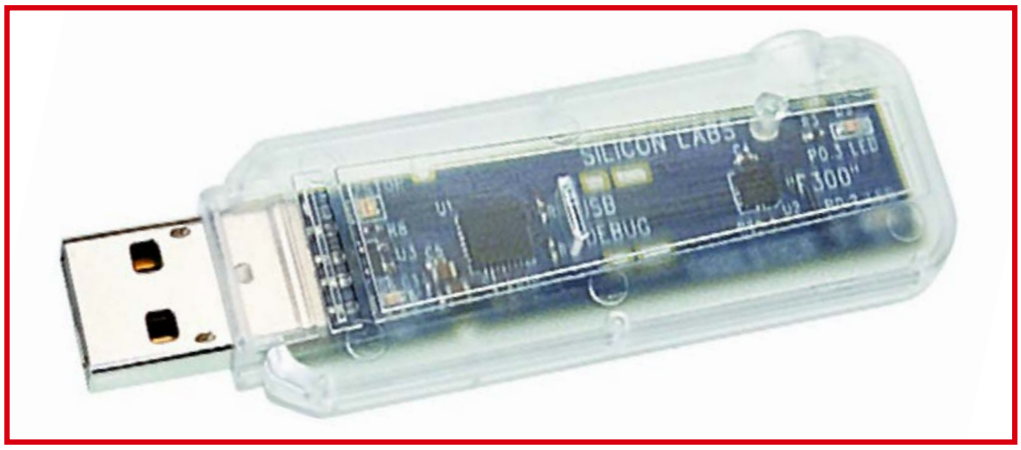


L'équipe d'ITEC créa, à l'aide d'un programme de CAO, un modèle volumétrique.



Le sac à dos contenant les accus alimente le bébé de Colani.

Clé USB = Carte d'évaluation pour IC



Le fabricant de semi-conducteurs Silicon Laboratories, spécialisé dans les circuits intégrés dits mixed-signal, a développé, pour le support de ses microcontrôleurs 8 bits, une minuscule carte d'évaluation prenant la forme d'une clé USB, outil baptisé comme on pouvait s'y attendre, le ToolStick. Tout ce dont a besoin le développeur pour se mettre au travail est d'avoir à portée de main un PC possédant (encore) un port USB de libre (heureusement qu'il existe des Hub USB pour pas très cher, les fabricants de produits détergents nous en feront bientôt cadeau avec leurs « barils de poudre »). Le concepteur peut immédiatement effectuer ses premiers pas, épaulé par l'environnement de développement de Silicon Laboratories combiné au matériel de débogage embarqué à même la puce (*on-chip*). Ce dernier permet un accès total au C8051F, un microcontrôleur mixed-signal, les périphériques et la mémoire. L'environnement de développement logiciel se compose d'un IDE (*Integrated Development Environment*) confortable, d'un éditeur, d'un débogueur, d'un programmateur Flash et d'une version d'évaluation du compilateur de Keil. Le ToolStick fait appel aux MCU ultra-compacts de la série F300 de SL. Ces microcontrôleurs peuvent être mis en liaison avec le PC par le biais de l'adaptateur-débogueur USB basé sur un microcontrôleur USB, le F321.
Dè la connexion du ToolStick au PC, celui-ci se comporte, au vu de Windows, comme une HDI (*Human Interface Device*), de sorte qu'il n'est pas nécessaire de disposer de pilotes spéciaux pour pouvoir communiquer avec la MCU du ToolStick. Deux programmes d'exemple fournis montrent comment utiliser l'environnement de développement et la MCU. Les MCU de la famille C8051F de SL intègrent un coeur de CPU 8051 (une vieille connaissance) capable d'une puissance maximale de 100 MIPS, une mémoire Flash de 128 Koctets maximum ainsi que des périphériques analogiques extrêmement sophistiqués et précis, le tout dans un boîtier ne faisant pas plus de 9 x 9 mm. Tous les MCU mixed-signal de ce fabricant sont supportés par un set d'outils de développement bon marché et facile à mettre en oeuvre qui épaulent parfaitement le matériel de débogage embarqué sur la puce.

Pour en savoir plus
www.silabs.com

(067050-I)

Système de mesure pour Pocket PC

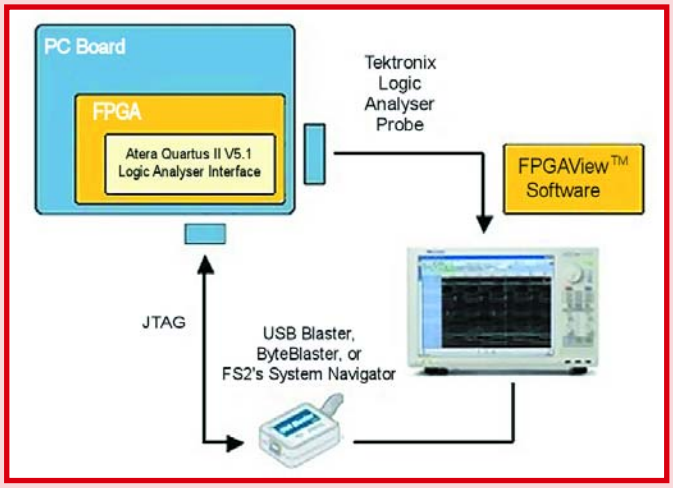
Virtins Pocket Instrument est un logiciel de mesure pour le Pocket PC puissant. Le set de programmes comporte un oscilloscope en temps réel, un analyseur en temps réel et un générateur de signal, ces différentes fonctions pouvant être utilisées simultanément. Le système connaît une technique d'acquisition de données spécialement développé à son intention, ce qui lui permet d'effectuer un suivi (monitoring) continu du signal d'entrée. On peut de cette manière garder en permanence à l'oeil le niveau de signal auquel doit se faire le déclenchement, sans qu'il ne soit nécessaire auparavant d'effectuer de conversion de données. On dispose également de fonctions de pré- et de post-déclenchement.
Bien que la majeure partie de l'écran du Pocket-PC soit consacrée à la visualisation des données on dispose aussi d'une fonction de zoom et de défilement (*scrolling*) qui permet d'examiner le moindre détail à la loupe.
Le système de mesure est doté de toute une série de possibilités, telles que la sommation ou la soustraction de signaux, la visualisation de figures de Lissajous, l'affichage d'une indication de tension, l'enregistrement de signaux, la représentation du spectre d'amplitude en RMS, la visualisation de l'amplitude relative et de la phase, celle du coefficient de relation (auto et croisée) un générateur de fonctions, un générateur de forme d'onde aléatoire, un générateur de bruit blanc et rose, sans oublier un générateur de balayage (*sweep*).
Si vous êtes intrigué, le fabricant met une version d'évaluation de son logiciel à votre disposition sur son site. Le prix de la version complète du logiciel est de 50 $US.

Pour en savoir plus :
www.virtins.com

(067029-I)

Débogeur en temps réel pour les FPGA d'Altera

Tektronix, Altera et First Silicon Solutions (FS2) ont présenté de concert FPGAView, un ensemble de logiciels développé par FS2 pour être utilisé avec les analyseurs logiques TLA de Tektronix pour un débogage en temps réel des FPGA d'Altera. Le cercle est ainsi fermé. FPGAView permet à un concepteur d'examiner les signaux internes de leurs projets à base de FPGA d'Altera et de sélecter des groupes de signaux sans que cela n'implique de recompilation du projet.



Une fois que l'on a installé le logiciel FPGAView sur un analyseur logique de Tektronix, il est possible de l'activer par le baiais de la fonction interface d'analyseur logique intégrée dans la version 5.1 du logiciel de développement Quartus II d'Altera. Cette fonction d'interface pour analyseur logique permet d'associer un grand nombre de signaux internes à un tout petit nombre de lignes de sortie par le biais d'un multiplexeur configurable par l'utilisateur. Une fois que l'on connecte l'analyseur logique au circuit FPGA par le biais de l'embase JYAG, FPGAView détermine quels signaux FPGA internes sont reliés aux lignes de sortie pour une reproduction et un débogage en temps réel. Pour un développeur, la combinaison FPGAView + la fonction d'interface analyseur logique + l'analyseur logique de Tektronix, met à sa disposition un outil plus confortable d'utilisation et moins drastique que les méthodes de débogage classiques.

FPGAView est disponible pour rous les analyseurs logiques TLA de Tektronix. IA noter que ce programme fonctionne également avec les câbles de programmation ByteBlaster et USB-Blaster d'Altera, ce qui évite d'avoir à déterminer manuellement les dénominations de signaux.

Intrigué ?

www.fs2.com
www.altera.com
www.tektronix.com

(067049-I)

Accu ultra-fin de NEC

Le fabricant japonais NEC a développé un accu d'une finesse extrême possédant un temps de (re)charge phénoménalement court puisqu'il n'est que de 30 s. De par la très faible épaisseur de cet accu, 300 lm soit 0,3 mm seulement, il devient possible de l'intégrer à des cartes à puce voire à du papier « intelligent ».

Ce nouvel accu baptisé ORB (*Organic Radical Battery*) possède une cathode faite d'une variété de plastique dite ORP (*Organic Radical Polymer*). L'état dans lequel ce trouve ce matériau est dit état gel électrolyte-perméable et c'est à lui qu'est due l'extrême flexibilité de ce nouveau type d'accumulateur. Le polymère radical organique se caractérise en outre par une réaction au niveau des électrodes très rapide, les solutions salines pouvant migrer très aisément dans le polymère-gel. Ceci se traduit par une résistance faible lors du cycle de charge et permet une durée de charge inférieure à une demi-minute.

La densité énergétique de ce nouveau type d'accu est de 1 mWh/cm2. Outre sa finesse, sa flexibilité et sa recharge rapide, ce type d'accu est également respectueux de l'environnement puisqu'il ne comporte pas de métal lourd, donc ni mercure, ni plomb ni cadmium.

Les domaines d'application de ce nouvel accu sont les terminaux extra-fin, c'est-à-dire les cartes à puce auto-alimentées et autres types d'appareils pouvant être connectés à un réseau pour constituer des terminaux. Ainsi, monté dans un dispositif RFID, cela se traduirait par plusieurs milliers de transmissions de signal avant que celui-ci ne doive être rechargé.

www.nec.co.jp/press/en/0512/0701.html

(067047-I)

Testeur de tension de poche

Fluke, l'un des grands de l'appareillage de mesure et de test au niveau mondial propose une version réactualisée de son testeur de tension alternative « VoltAltert ». Il permet, en un clin d'oeil et en toute sécurité, de vérifier si un réseau se trouve sous tension ou si la mise à la terre ne se fait pas correctement. Le testeur de tension «1AC-II VoltAlert » est doté de la fonction d'auto-test « Volt-Beat » propriétaire de Fluke qui procède à une vérification continue les réseau et l'état de la pile.

Le fonctionnement correct de l'appareil est indiqué par un double clignotement se répétant à intervalle régulier.

Cet instrument pratique teste la présence de tension en un rien de temps, permettant que l'utilisateur n'entre en contact avec des conducteurs non isolés.

Il suffit en effet pour cela, de toucher de la pointe de la sonde un bornier, une prise secteur ou un câble. Toute présence de tension est visualisée par l'allumage du rouge et un signal acoustique.

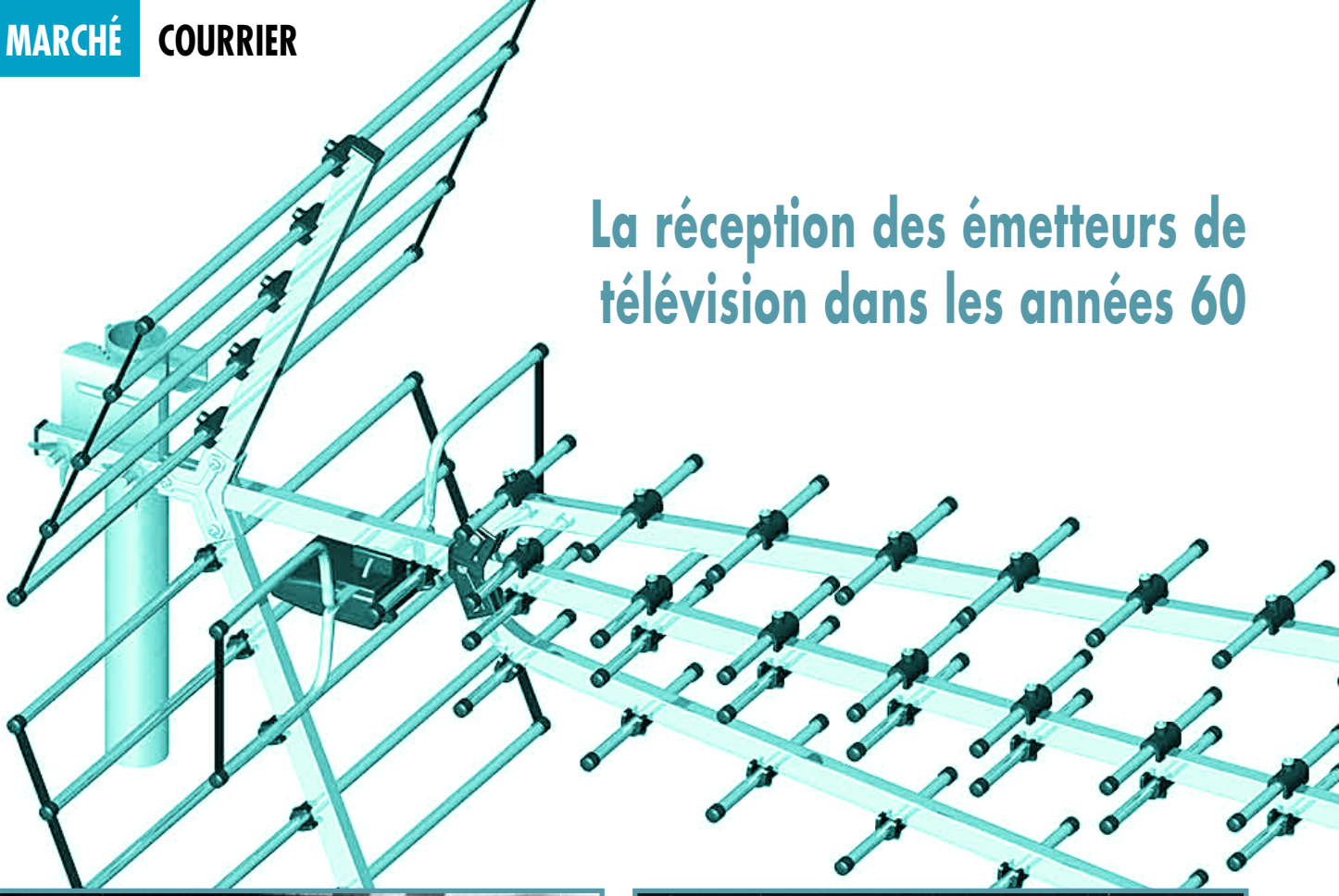
L'appareil respecte les normes sévères de la catégorie CAT IV 1000 V. La plage des tensions pouvant être testées va de 200 à 1 000 V.

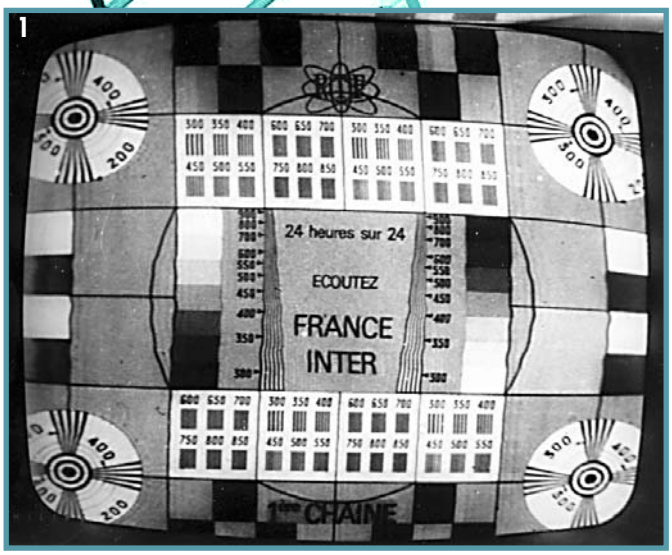
Le prix n'est que de 19 euros + TVA.

Un tour à
http://www.fluke.fr/comx/news.aspx?locale=frfr
tombe à pic (de tension).

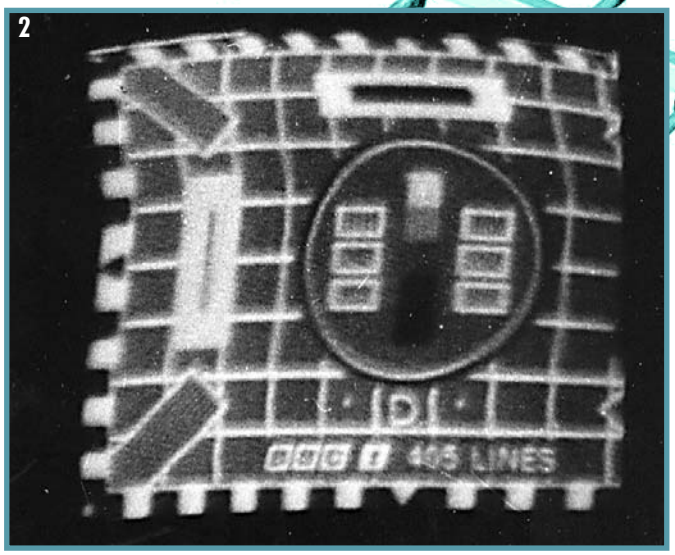
(067045-I)

La réception des émetteurs de télévision dans les années 60





1



2

François Frappé,
Dunkerque
frappef@rouvroy.com

Je vous invite à faire un sympathique retour en arrière, à l'époque où les tubes électroniques chauffaient fort, pas seulement dans les radios BF comme les nostalgiques connaissent bien, mais aussi dans les récepteurs de télévision.

Le petit écran en était à ses débuts, et moi aussi. Tout commence en 1967, quand mon frère aîné me ramena un « vieux » poste de télé, destiné à être démonté pour la récupération des pièces. Plutôt que de le disséquer et le transformer en amplificateur Hi-Fi, je préférai lui donner une chance de survie. Novice en la matière, et ce n'est pas au collège qu'on apprend à réparer les télés, je parvins pourtant à refaire balayer le spot sur le phosphore luminescent. Stupéfiant ! Je recevais la première chaîne française ! [1] Et en noir et blanc… Bientôt, je captais quelques émetteurs étrangers qui arrosaient la région. Le poste était prévu à l'origine uniquement en 819 lignes, et possédait un tuner VHF monocanal (Lille, 8A). Les émissions flamandes et britanniques étaient émises sur d'autres fréquences, et respectivement en 625 et 405 lignes [2].

La mise en place d'un rotacteur VHF et un gros bidouillage dans les bases de temps, fit de ce poste l'unique multistandard du quartier. Restait maintenant à attaquer l'UHF si je voulais voir les speakerines de la deuxième chaîne française. A l'époque, on ne trouvait guère mieux que des tuners UHF à deux tubes EC86/EC88, avec un gros condensateur variable en guise de recherche d'une éventuelle station. La sensibilité était du style poste à galène. Il valait mieux installer les antennes très haut, et avoir des câbles très courts. Cette incompatibilité engendra la réquisition du grenier qui allait devenir désormais un véritable laboratoire de recherches où des dizaines de téléviseurs s'y accumuleront…

Un jour, j'étais de passage chez mon réparateur TV préféré, et il me donna un poste « moderne ». Un grand écran de 49 cm de diagonale, à coins un peu moins ronds que les autres, mais surtout, avec un tuner UHF à transistors.

Après le remplacement des tubes pompés dans la THT (PY82, PL36, DY86), le poste fonctionnait à merveille. Il y avait là-dedans trois pentodes PL89 d'amplification FI (fréquence intermédiaire), avant un ampli vidéo musclé. Je ne vous dis pas les performances. En VHF, on pouvait recevoir à l'aise la chaîne belge francophone venant de Bruxelles La bande UHF était bien plus riche en émetteurs que je ne le pensais. La BBC2 [3], l'ITV (chaîne anglaise indépendante et deux chaînes hollandaises [4] m'y attendaient tranquillement. Seulement, ces images étaient instables, en négatif, et le son était nasillard. Après renseignements, j'appris qu'il s'agissait d'autres normes (I et BG). Il fallait inverser une diode à trouver quelque part dans le poste pour l'image, et construire un démodulateur FM pour le son. Après avoir inversé toutes les diodes du téléviseur, je parvins enfin à voir quelque chose de correct sur l'écran. Je recevais désormais une bonne dizaine de chaînes, toutes faciles à repérer, car à cette époque-là, il y avait très peu de programmes. Les iconoscopes balayaient les trois quarts du temps les fameuses mires, propres à chaque émetteur.

Un jour d'été, alors que je me connectais dans la bande I sur la fréquence de la chaîne belge flamande, je fus étonné de voir une mire que je ne connaissais pas. L'image était très perturbée, mais de temps en temps de très bonne qualité : il s'agissait de la télévision tchécoslovaque ! [5]

Cela n'a duré que cinq minutes, avant que je me prenne les pieds dans mon fer à souder et que je ne rate la marche des escaliers du grenier en voulant relater l'événement à tout le monde. Au moment où l'on se retrouvait tous coincés dans la porte, plus de Tchécoslovaquie. Rien que du brouillage, et des bips de radioamateur. J'avais l'air malin. Mais il suffisait d'attendre un peu, car devant nos yeux hagards, cette fois-ci la télévision polonaise apparut à l'écran, puis les télés russes [6], suédoises, norvégiennes… Pas de quoi suivre un film, sauf avec un bon collyre, mais la surprise était totale. La semaine suivante, toute l'Europe avait défilé sur mon écran ! Les chaînes roumaines, espagnoles, portugaises, italiennes [7]

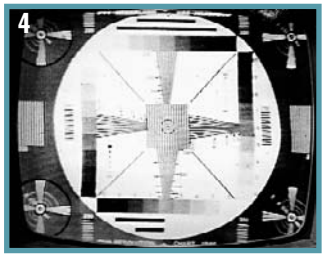
… En réalité, les ondes de la bande I, dont les fréquences sont relativement basses (40 à 60 MHz), peuvent se propager de manière sporadique selon l'existence aléatoire d'une couche ionosphérique saisonnière.
Bien entendu, pas grand monde ne me croyait quand je disais que je ne captais pas moins que 250 émetteurs jusqu'à 3 000 km de distance. C'était soi-disant impossible, surtout depuis que la terre est ronde, et pourtant… Mais, vous, respectable lecteur, devez penser que c'était tout à fait banal. Non, il n'y avait pas de satellite pour relayer l'onde capricieuse. Pour réconcilier les incrédules, rien de tel que de belles photographies des mires et des génériques. Imparable.

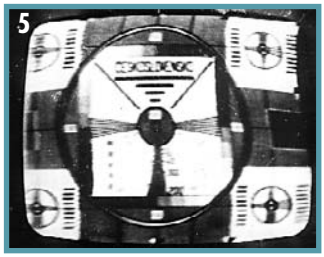
Je m'intéressais également à la propagation très particulière de l'UHF, beaucoup moins aléatoire. Celle-ci est friande de hautes pressions atmosphériques et de brouillards denses, qui forment une sorte de guide d'onde ou de loupe géante entre l'émetteur et le récepteur, pour peu que le soleil soit au rancart. La perte du signal UHF est proportionnelle au carré de la fréquence et à la longueur du câble. Le téléviseur étant déjà à un point culminant de la maison, l'idée était de placer le tuner carrément dans le dipôle et de descendre en fréquences moins vulnérables (32 à 39 MHz). Manque de pot, les tuners à diodes varicap n'existaient pas. Balèze dans la bidouille, je mis au point un système de poulies, de ficelles, contrepoids et autre frein de vélo pour mettre en rotation l'inaccessible condensateur variable à partir de mon fauteuil. Arrêtez de rigoler car les résultats étaient sensationnels : la deuxième chaîne finlandaise [8] en UHF, soit 1 500 km de distance. Magnifique.

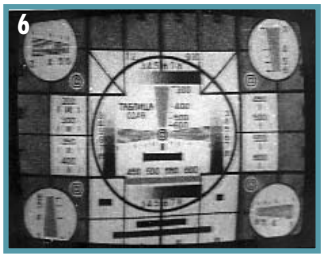
Le premier poste TV en couleurs que j'avais récupéré était aussi à tubes. Cinquante kilos d'électronique, avec 819/625 lignes (je ne vous décris pas les circuits de convergence), et PAL/SECAM à transistors. Résultats fabuleux pour les quelques chaînes UHF qui venaient de se coloriser. En tous cas, ce poste contribuait bien au chauffage du grenier l'hiver. C'était vraiment une époque très sympa, où tout était compliqué et cher. Maintenant, c'est simple… et la parabole est gratuite.

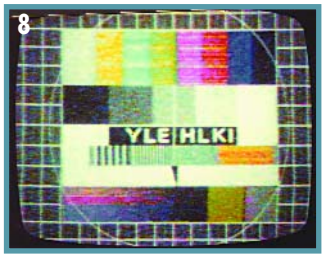
(067052-1)

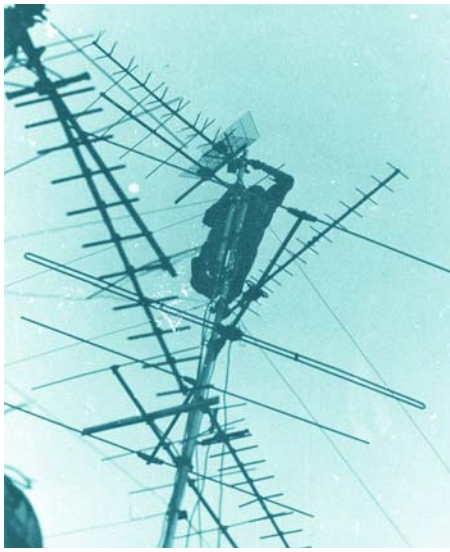
L'auteur nous a proposé bien d'autres photos d'écrans de TV de l'époque. Nous vous proposons un petit jeu. Donnez-nous leur pays d'origine, leur canal si vous le connaissez ; nous tirerons au sort, parmi les bonnes réponses qui nous seront arri-











vées à l'adresse donnée ci-dessous ou à notre E-mail (redaction@elektor.fr),

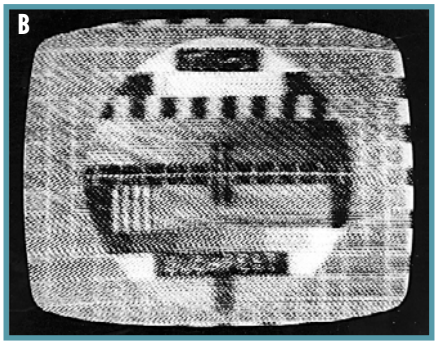
3 Starter Kit XC866 offerts par Infineon.

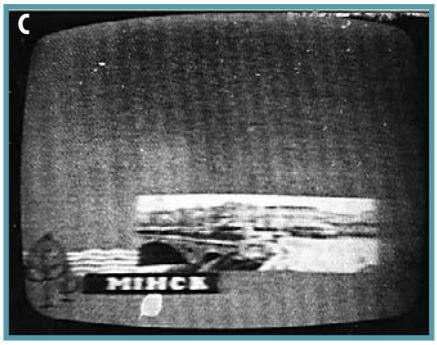
Date limite de réception des réponses, le 31 mars 2006

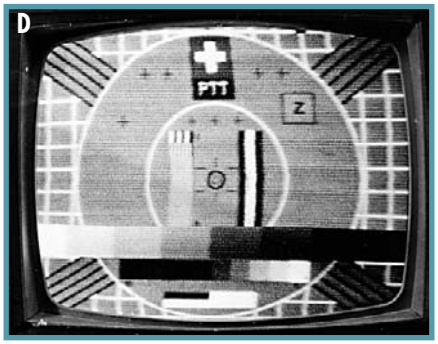
Envoyez vos réponses à l'adresse suivante :

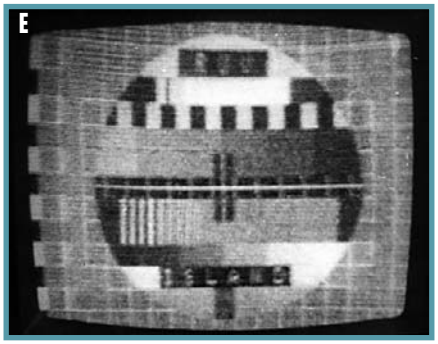
ELEKTOR / SEGMENT B.V.
Le Dôme
1, Rue de la Haye
B.P. 12910
95731 Roissy CDG

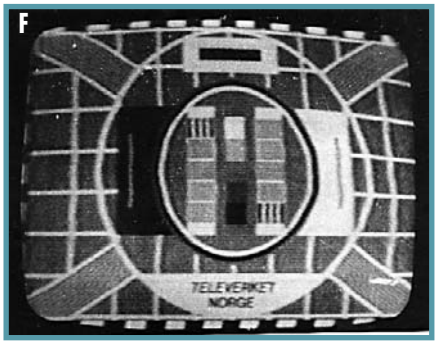
Voici les photos des écrans en question. Nous vous prions de nous excuser de leur qualité mais ni la télévision ni les appareils photos de l'époque n'étaient numériques…

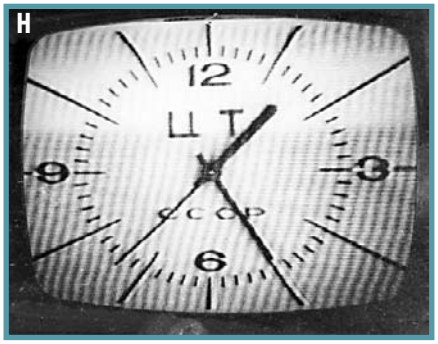


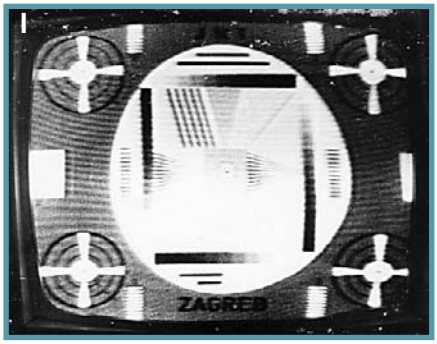


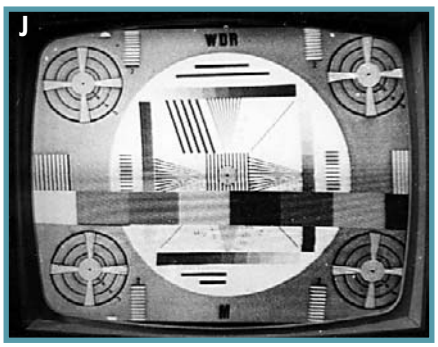












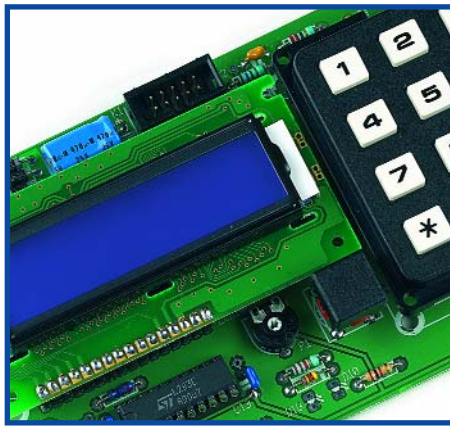
Article « contrôleur pour four à refusion »

Cher Monsieur,
J'ai lu avec intérêt l'article de M. Paul Goossens sur un contrôleur pour four à refusion publiéw dans votre numéro de Janvier 2006 dans vos différentes éditions. Avec intérêt, mais également avec une certaine stupéfaction : En effet, cet article semble reprendre, quelquefois de très près, un article que j'ai moi-même publié dans la revue Circuit Cellar en Juin 2004 ! Vous trouverezx cet article en ligne via le lien suivant : www.circuitcellar.com/library/print/0704/Lacoste_168/index.html.
Bien sûr les mêmes (bonnes) idées peuvent émerger indépendamment chez différents auteurs, mais ici la ressemblance est à mon vis (sic) vraiment forte : Les composants sont différents, mais le concept est rigoureusement le même, et même la structure générale du logiciel semble même très proche.
Étant consultant professionnel en électronique et un contributeur régulier à des revues professionnelles, vous comprendrez que cette situation puisse me causer un tord (sic) certain. **A minima il me semblerait donc juste que soit brièvement signalé dans votre prochain numéro l'antériorité de mon propre article**, et ce dans les différentes éditions dans lesquelles l'article a été publié.
Cela vous semble-t-il correct…
Robert Lacoste

Cher Monsieur Lacoste,
Il ne nous arrive que très rarement d'être soupçonnés de plagiat. Comme convenu, je fais mention de votre courrier dans le premier numéro possible, celui de mars 2006.
Je vous avais dit que je pensais, dans le cadre de ce que je savais pour le moment, qu'il y avait à mon avis coïncidence, qu'il n'y a pas dix manières d'imaginer le processus de refusion, un four domestique en valant un autre, la rédaction d'un article dans quelque magazine que ce soit, chez Elektor aussi, suivant un cheminement logique…
Voici la réponse de l'auteur de ce projet, Mr Goossens, membre de l'équipe u laboratoire d'Elektor.
« Plusieurs personnes ont déjà attiré mon attention sur le dit article. Il ressemble en effet beaucoup au nôtre, et visiblement nous ne sommes pas les seuls à avoir eu cette idée. J'ai procédé à quelques recherches depuis sur Internet et ai trouvé plusieurs pages Web qui décrivent comment transformer un petit four domestique en un four à refusion. Il semblerait que plusieurs personnes soient intéressées par cette idée.
En ce qui concerne les spécifications de la régulation, je pense qu'un coup d'?il au profil mesuré (température vs temps) de cette régulation suffit pour voir que notre régulation est plus précise. Sur la version en question il me manque également une fonction de calibration telle que la comporte notre montage.
Nous pouvons envoyer un E-mail à Mr Lacoste pour l'informer que sa régulation n'a en aucun cas servi de point de départ pour notre réalisation. Cela devrait également lui faire plaisir de constater qu'il y a d'autres personnes qui ont déjà fait quelque chose de similaire. Il a pensé et réalisé un projet d'électronique moderne ! Salutations. Paul »
P.S. Connaissant Mr Goossens depuis de nombreuses années, je ne mets pas sa parole en doute… Si vous lisez régulièrement notre magazine vous verrez d'autres articles décrivant des montages sophistiqués et hyper-actuels (venues et) venant de ses mains dans les prochains mois…
J'ose espérer que ce courrier mettra fin à l'incertitude vous rongeant…
Affaire close je suppose…

Programmateur DCC "universel"

Amateur ferroviaire, lecteur régulier de votre revue, je vous ai commandé, suite à l'article de Benoît Bouchez (Programmateur DCC "universel"), revue n°323 mai 2005, le circuit imprimé et le microprocesseur programmé. J'ai acheté les autres composants chez "Electrome" à Bordeaux (40,80 €), en particulier l'afficheur : GDM1602A de Xiamen Ocular Optics Co Ltd.
J'ai respecté le câblage, en particulier les remarques de la page 52 :



"L'afficheur LCD
Dernier point : le montage de l'afficheur LCD. Vous remarquerez que le circuit imprimé comporte une rangée de connexion à cet usage, repéré LCD1.
Ce connecteur compte 16 points alors que le connecteur LCD standard n'en compte que 14. La raison à cela est que certains modèles d'afficheurs reportent l'alimentation du rétro-éclairage sur deux broches supplémentaires, contiguës aux broches du bus de données. Nous avons à nouveau utilisé un affichage PLED en raison de l'incroyable beauté et lisibilité de son afficheur.
Si votre afficheur ne comporte que 14 broches (modèle sans rétro éclairage ou avec alimentation du rétro-éclairage séparée), la broche 1 de l'afficheur doit être positionnée sur la broche la plus à gauche (côté R9/R3) du connecteur LCD1."
Les photos confirment le texte.

MISES AU POINT

Analyseur OBD-2
(050092-1)

Bonjour, j'ai commandé chez vous un analyseur OBD-II. Le problème est que je possède plusieurs composants qui ne correspondent pas aux caractéristiques écrites dans le magazine.
R6,R7,R8,R10,R11,R14,R17 ,R22,R23 devraient valoir 10 W. Je n'ai reçu que 9 résistances (au lieu de 10) de 10 kW.
Est ce une erreur d'impression sur le magazine?
R12,R13 devraient valoir 510 W alors que j'ai reçu 2 résistances de 1 kW.
Enfin, les couleurs des anneaux de R20 de la photo ne correspondent pas à celles présentes sur la résistance que j'ai reçue.
Pour finir, le petit shunt à 2 connecteurs doit il être mis sur le circuit ou pas?
Je vous remercie d'avance pour votre réponse.
Silvia Coelho David

Il se peut toujours qu'il puisse y avoir eu une erreur quant aux composants envoyés...

Mais ici la liste des composants comporte en effet une erreur, les résistances R6 à R8, R10, R11, R14, R17, R22 et R23 sont, comme le dit le schéma, des 10 kΩ et non pas, comme le dit à tort la liste des composants, des 10 Ω.

Le schéma ne comporte lui pas d'erreur à notre connaissance...

Il est donc normal que vous ayez 9 résistances de 10 kΩ, vous devriez en avoir en outre 2 de 1 kΩ, 2 de 100 Ω, 6 de 4kΩ7, 2 de 510 Ω, une de 1kΩ1, 3kΩ9, 3 kΩ et 1 Ω...

La résistance R20 est de 3 kΩ soit orange, orange, rouge...

Le cavalier n'est à mettre que pour la programmation donc comme vous avez acheté un composant programmé il ne faut pas l'implanter...

C'est faux ...

Les datasheets du GDM1602A (1-16) et du ASI-G-162FS-GF-EWS/W (réf. de la revue) (14-1, 15,16), indiquent un câblage #. C'est la broche 14 (D7) qui doit être la plus à gauche. Aucun rectificatif dans la revue (je viens d'acheter celle de novembre), sur le site, avec ma commande. Résultat : un afficheur HS (22,10 €), et une journée de perdue (vérifications, soudures, etc.)

La réponse de l'auteur : Suite à votre email, retransmis par la rédaction d'Elektor, j'ai effectué les vérifications nécessaires.
Il y a bien une erreur dans le texte de l'article, puisque c'est la broche 14 qui est à gauche de l'afficheur et non la broche 1. Le schéma est en revanche parfaitement correct.
C'est donc la broche 14 qui doit se trouver côté R9/R3 si l'afficheur est équipé d'un connecteur 14 points.
Cette erreur est passée complètement inaperçue lors de la relecture du texte, vous m'en excuserez (je suis d'autant plus confus que cela vous a coûté un afficheur LCD). En revanche, et comme le soulignait Guy Raedersdorf, nous avons l'habitude d'utiliser cette disposition de composant "en sandwich" et cette façon de procéder ne nous a jamais posé de problèmes, l'installation mécanique se faisant naturellement.

Votre message me confirme de la nécessité d'être toujours plus vigilant sur tout ce qui concerne les afficheurs LCD : il existe désormais une pléthore de modèles totalement différents, et dont les circuits imprimés portent de moins en moins souvent d'indication quant à la position de la broche 1 (j'ai ici sur ma table pas moins de 10 afficheurs différents, seul un modèle à sa broche 1 clairement identifiée !)
Je vous réitère mes excuses pour cette erreur (à ce propos, êtes-vous sûr que votre afficheur est détruit ? Par expérience, je sais - suite à de nombreuses erreurs de manipulation de ma part lors d'essais de prototypes - que les afficheurs font parfois preuve d'une résistance étonnante et redémarrent même après un branchement inversé).

Projet lecteur

Depuis juin 2005 je fais partie des abonnés à Elektor car je cherche toujours de nouveaux exemples de circuits pour pouvoir les intégrer à des réalisations de mon cru. J'ai été surpris par la largeur de la palette des montages proposés, les montages à tubes restant l'une des rubriques phare de vos numéros. J'ai lu avec énormément d'intérêt vos articles consacrés au thème « ampli à tubes ». Les articles de discussion me rappellent quelque peu les questions qui circulent dans le monde de la musique électronique :
Les synthétiseurs analogiques virtuels outils réellement le même son que leurs prédécesseur analogiques ?
Dans le numéro double de juillet/août j'ai trouvé un circuit « Convertisseur à son de tubes » qui, en raison de sa simplicité, convenait à merveille pour un premier projet. J'ai donné de nouvelles ailes à mon vieux lecteur MP3 de 128 MB. Dans leur nouvel habit commun j'ai ainsi réalisé un appareil tout neuf que mon épouse m'a même autorisé à laisser dans le salon. Le crépuscule aidant, l'attrait optique de cet ensemble croît exponentiellement.
Dario Aeppler

Voilà ce que l'on entend sous le la dénomination d'électronique créative ;-)

Rétronique précieuse

Un petit mot au sujet de votre article Rétronique « Multimètres analogiques » du numéro de décembre 2005. Le multimètre évoqué dans cet article, le Simpson 260 est très apprécié (ce qui explique également son prix élevé sur e-Bay) parce qu'il reste toujours utilisé avec amour et plaisir lors de la révision d'aéronefs. Dans la documentation avion de Boeing, c'est très exactement ce modèle qui est évoqué de sorte que nombre de compagnies aériennes s'en servent comme instrument de mesure.
Christian Wendt

Règles du jeu

– Publication de la correspondance de lecteurs à la discrétion du Rédacteur en chef
– Les points de vue et opinions exprimées par les correspondants ne sont pas nécessairement ceux du Rédacteur en chef ou de l'Éditeur.
– La correspondance pourra, le cas échéant, être traduite ou éditée en longueur, clarté et style.
– En cas de réponse à COURRIER, veuillez S.V.P. indiquer le numéro concerné.
– Veuillez S.V.P. adresser votre correspondance :

redaction@elektor.fr ou
Rédacteur en chef Elektor
Le Dôme
31, rue de la Haye
BP. 12910 - 95731 Roissy CDG - France

Nouveau logiciel

Version 2.0 avec encore plus de possibilités

Steffen van de Vries

illustration : RailEurope

EDiTS Pro est indubitablement le système de pilotage pour ferro-modélisme à réaliser soi-même ayant rencontré le plus de succès. Des milliers d'amateurs en ont doté leur réseau ferroviaire miniature. Le « père » d'EDiTS Pro n'est pas, ces dernières années, resté les bras croisés et nous a fait la surprise d'un nouveau logiciel de pilotage : EP 2.0.

EDiTS Pro existe depuis plusieurs lustres déjà. Il n'a guère vieilli au cours des ans, de par les mises à jours plus ou moins régulières des matériels qu'il requiert et de son logiciel de pilotage. Il reste d'actualité même pour des réseaux actuels.
La partie d'EDiTS Pro dont le cycle de vie est le plus court est indubitablement son logiciel de pilotage tournant sur PC. Les développements de la micro-informatique sont tels qu'il faut impérativement prévoir une nouvelle mouture de programme tous les 3 ou 4 ans si l'on ne veut pas être dépassé. D'où la présentation ici de la version 2.0.
Un programme moderne pose inévitablement certaines exigences au niveau du système sur lequel il tourne. Pour que la version 2.0 fonctionne bien il vous faut disposer d'un PC travaillant avec Windows 98Se/Me ou XP, doté d'un minimum de 256 Moctets de RAM et tournant au moins à 1 HGz voire plus. Si vous possédez un écran à très haute résolution (1 600 x 1 200 pixels par exemple), il vous sera possible de visualiser parfaitement des projets de réseaux très développés.

Améliorations

Outre les nouvelles extensions intéressantes dont il a été pourvu, le programme a également été amélioré à nombre de points de vue de manière à mieux répondre aux exigences de l'amateur de réseaux ferroviaires miniature de 2006.
Il est possible de lire dans EP 2.0 (EP pour EDiTS Pro) des projets existants faits sous EP 1.2; il vous faudra cepen-

pour EDiTS Pro

dant en supprimer les lignes de programme.
Grâce au logiciel EP 2.0, les possibilités des contrôleurs plus rapides dotés de la version 1.2 du progiciel (*firmware*) se voient parfaitement utilisées sans que cela ne nécessite d'autres contrôleurs (pour l'instant). Il ne sera question d'adaptation du contrôleur qu'après livraison complète du nouveau Märklin Systems et que l'on en connaîtra toutes les possibilités.
Le logiciel de pilotage EDiTS Pro permet de « dessiner » un projet de réseau sur écran en se servant de toute une série de symboles. Le set de symboles disponibles suffit pour le dessin à l'écran du réseau ferroviaire le plus complexe.
L'écran de commande présente de fortes similitudes avec les panneaux commande que l'on trouve dans la réalité, mais il n'en reste pas moins facile de construire un réseau, de le corriger et de l'entretenir.
Outre cette commande par le biais de l'écran des aiguillages et des signaux, il est possible de créer des régleurs de train pour un maximum de 240 convois; il est possible en outre de définir des itinéraires et de programmer le suivi de l'exécution d'un programme par les trains (figure 1).

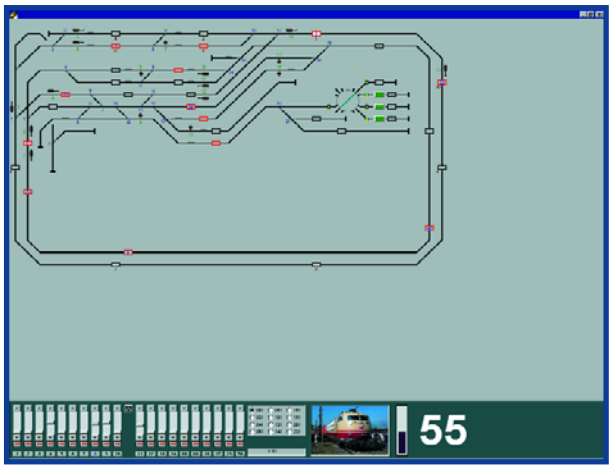


Figure 1.
Dans sa nouvelle version le logiciel EDiTS Pro offre des perspectives intéressantes pour des projets de réseaux de grande envergure.

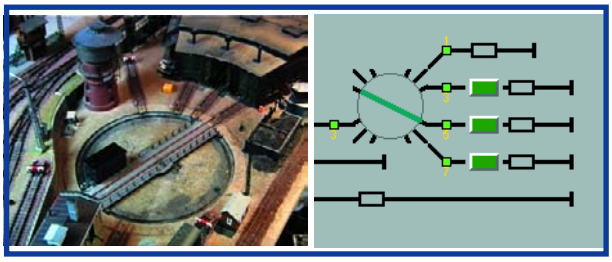


Figure 2.
Le logiciel supporte 3 modèles de ponts tournants.

Menu principal

Le menu principal du programme propose 6 options :

1. Le tableau de commande
La première étape consiste à réaliser le réseau ferroviaire miniature à l'écran à l'aide de blocs symboliques, à l'image d'une sorte de briques « Lego », que l'on saisit dans la légende.
Il existe des briques pour des sections de rails droits et en courbe, pour les aiguillages et les signaux ainsi que des boutons de détection.
Les boutons de détection sont des symboles rectangulaires implantés sur le plan du réseau et visualisant l'adresse du train, mais qui peuvent également servir de point de début ou de fin de itinéraire.
Nouveauté de la version 2.0, le support de 3 modèles de ponts tournants à 8, 12 ou 16 connexions de voies (**figure 2**). Ces ponts tournants connaissent 3 types de commande :
- Pilotage chronologique
- Pilotage par un décodeur EP avec retour
- Décodeur de pont tournant Märklin

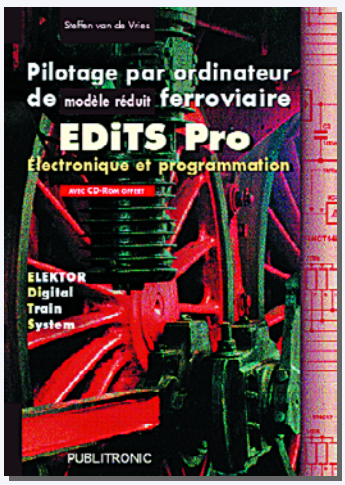
2. Adresses de décodeurs
Lors de cette étape, on procède à l'attribution des adresses (de décodeurs) des composants actifs (aiguillages, signaux) et de celles des modules de détection (d'adresse) de train.
La version 2.0 a été dotée d'une possibilité d'utilisation d'ILS (**I**nterrupteur à **L**ame **S**ouple, un relais *reed*). Les amateurs de réseaux de jardin ne manqueront pas d'apprécier cette option. Un segment de voie est identifié par la prise d'un ILS à son début et à sa fin, interrupteur activé par un aimant permanent.

3. Itinéraires

Cette option du menu permet de définir les itinéraires. On choisit 2 boutons de détection délimitant un itinéraire, boutons que l'on clique ensuite à la souris. En cliquant sur tout ce qui se trouvent entre ces 2 boutons, on

Le livre EDiTS Pro

Le livre « Pilotage par ordinateur de modèle réduit ferroviaire » EDiTS Pro décrit dans le détail la totalité du système EDiTS Pro, avec tous ses éléments, y compris de nombreux exemples d'application (seuls les circuits d'interface, l'électronique de l'adressage de régleur et la version 2.0 du logiciel n'y sont pas mentionnés, datant d'après la parution de l'ouvrage).



Ce livre constitue une source précieuse d'information pour tout amateur de réseaux ferroviaires miniature souhaitant numériser leur réseau.

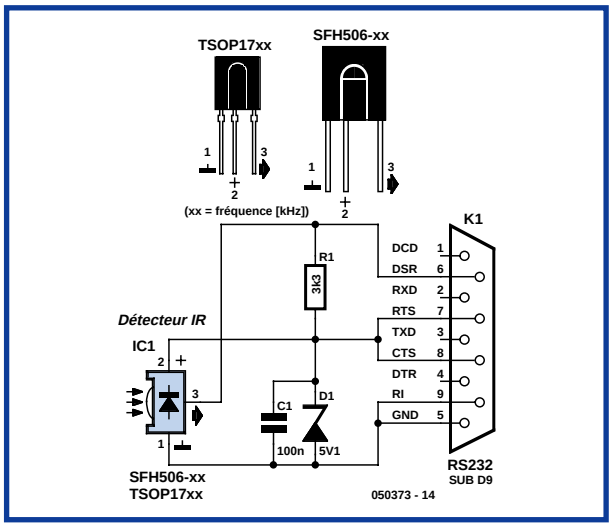


Figure 3.
C'est ainsi que l'on pourra brancher un récepteur IR à un second port RS-232.

définit un itinéraire. On pourra modifier à loisir les états des aiguillages et des signaux se trouvant le long de l'itinéraire par un simple clic sur chacun d'eux.
Il est possible, dans la version 2.0, de demander au logiciel de générer lui-même le itinéraire. Après avoir cliqué sur les points de début et de fin d'un itinéraire, le programme se mettra à la recherche d'un itinéraire les reliant. Cela fait gagner énormément de temps lors de la définition d'itinéraires.
Dans la version 2.0, la longueur des itinéraires a été portée de 25 à 35 éléments.

4. Régleurs soft
Sous cette option du menu on programme les régleurs de marche; on a en outre la possibilité de choisir un symbole correspondant pour la fonction dépendant du sens de circulation.
On peut choisir ici entre l'ancien et le nouveau format Motorola. Si l'on choisit le nouveau format, il est possible, outre la fonction dépendant du sens de circulation, d'activer 4 fonctions supplémentaires.
Le logiciel permet de programmer 240 régleurs. La visualisation de 240 régleurs sur un écran risquant d'être illisible (tout le monde n'exploite pas 240 trains), a été prévu une possibilité de sélecter des blocs de 20 régleurs à l'aide des boutons radio.

5. Programmation
Il existe maintenant un module de programme pour le traitement automatique d'un plan de circulation, module programmable par le biais de mouvements de la souris. Il permet de réaliser simplement des gares-fantôme, des plans de circulation de trains et des pilotages de blocks automatiques. Il incorpore également une protection contre les collisions latérales et frontales. Le principe de programmation est de charger le train des opérations. Pour cette raison, les lignes de programmes sont couplées aux boutons de retour ou de détection activés par le train.

Figure 4.
Les fonctions de télécommande sont visualisées en grand sur l'écran.

Jusqu'à cette version 2.0, les lignes de programme étaient couplées à un bouton de détection. Cette structure de programme était universelle de sorte qu'il suffisait de peu de lignes de programme pour piloter un nombre important de trains. L'inconvénient était que cette approche était relativement complexe et abstraite.
La version 2.0 reprend l'approche classique qui consiste à entrer la totalité du programme pour chacun des trains. Cela se traduit inévitablement par un nombre de lignes de programme plus important, mais comme la version 2.0 a été dotée de possibilités de correction efficaces (copier/coller de lignes de programmes) cela ne doit pas poser de gros problème.
Avantage additionnel de cette approche, un suivi très fiable des adresses, de sorte qu'il n'y a pratiquement pas de risque de « perdre un train ».
Comme chaque train possède son propre programme, il est possible d'activer ou de désactiver ce programme individuellement (la version 1.2 permet uniquement une activation/désactivation simultanée de la totalité des programmes elle).
Il est en outre possible, dans la version 2.0, d'activer ou de désactiver le programme des trains depuis n'importe quelle position du train sur la voie (à condition qu'il s'agisse d'un endroit pris dans le programme).
La version 2.0 montre également le itinéraire suivi par un train : un clic sur une ligne de programme se traduit par la visualisation de l'endroit où se passe l'action. Le début et la fin d'un itinéraire ouvert par une ligne de programme sont visualisés eux aussi.
La sélection de plusieurs lignes de programme produit un affichage de l'ensemble du réseau.

6. Opérationnel
Le choix de cette option permet à l'utilisateur de se mettre aux commandes de son réseau ferroviaire.
Il est possible, sur l'écran, de piloter des aiguillages et des signaux, de mettre des trains en mouvement, mais aussi de les faire changer de sens de circulation et d'activer ou de désactiver des fonctions de train. On a également une possibilité de définition manuelle de itinéraires. L'occupation des voies est signalée, la détection de trains indiquant les trains ayant passé. Il est possible d'activer ou de désactiver le programme individuel de chaque train.
On peut enregistrer sur le disque dur chacun des tableaux d'écran y compris ses spécificités, la situation dans laquelle se trouvait le réseau au moment où l'on cesse de l'utiliser et de le piloter.
Il est possible, si l'on utilise le programme à une résolution d'écran supérieure à 640 x 480, d'avoir, lors d'une sélection de régleur, visualisation à l'écran, dans la version 2.0, du train ou de la locomotive correspondant. Ce support visuel n'a pas uniquement un caractère professionnel mais il rend le pilotage du réseau plus confortable (cf. figure 1).
On pourra utiliser pour ces images ses propres photos voire utiliser celles que l'on aura trouvé sur Internet (format .jpg).
Autre fonction additionnelle très intéressante de la version 2.0 du logiciel, la télécommande par infrarouge. On pourra utiliser à cet effet pratiquement n'importe quelle télécommande IR (TV ou magnétoscope), en la combinant à un petit récepteur IR qui viendra se connecter à un autre port RS-232 du PC (figure 3).
Nous avons opté, pour disposer de cette possibilité, pour l'implémentation du logiciel Winlirc sachant qu'il nous a donné d'excellents résultats.

Voici les possibilités offertes par une telle télécommande :
– Pilotage d'un maximum de 240 trains avec toutes leurs fonctions
– Commande de 4 x 100 fonctions électromagnétiques
Comme il y a de forte chances qu'en cas d'utilisation d'une télécommande le moniteur se trouve à une distance plus importante, nous faisons apparaître les fonctions de télécommande en grand sur l'écran à titre de confirmation d'une bonne réception (cf. **figure 4**).
Dans la version 2.0, le menu de programmation du décodeur de locomotive a été modifié de façon à pouvoir définir individuellement chacun des pas de vitesse du décodeur de locomotive.
Ce qui a par contre disparu sont les possibilités de programmation des décodeurs Lenz et Uhlenbrock. Il s'est avéré qu'il n'était pratiquement plus possible de supporter ces options.
Vous pouvez obtenir le logiciel EDiTS Pro version 2.0 sur CD sous le numéro **EPS050373-81** (cf. www.elektor.fr). À noter que le CD-ROM comporte également une description du logiciel sous forme de fichier .pdf.

L'avenir

C'est après mûre réflexion que nous avons choisi de commencer par proposer la version 2.0 du logiciel et de ne venir qu'ensuite, éventuellement, avec une mise à niveau du contrôleur pour la simple et bonne raison que le contrôleur dans sa version 1.2 s'accommode fort bien de la version 2.0 du logiciel.
Comme nous le disions en début d'article, les décodeurs MFX et Märklin Systems viennent tout juste d'être mis sur le marché, mais on ne sait pas encore tout sur leur protocole. Rassurez-vous, nous suivrons à la trace les développements dans ce domaine.

(050373-1)

EDiTS Pro, l'histoire en marche

Le lecteur fervent d'Elektor passionné par le modélisme ferroviaire connaîtra sans doute par coeur l'historique d'EDiTS et d'EDiTS Pro. Epaulé par un premier article daté de 1989, EP en est à sa 4ème génération d'ordinateur et n'a pas fait de surplace, remettant à niveau les composants datant quelque peu. Il se peut ainsi que des modules qui n'ont pas évolué depuis le début soient toujours encore de la parti, alors que d'autres en sont à leur seconde ou troisième version.
De plus, certains des modules sont bien en avance sur leur époque (le module de détection de train en association avec le superdécodeur de locomotive en est l'exemple le plus marquant) qu'ils sont encore en avance sur le système MFX tout récemment remis à niveau par Märklin.

Le but que EDiTS Pro s'est fixé au départ n'a pas changé. Il vise à permettre la réalisation d'un système de pilotage de réseau ferroviaire numérique à réaliser soi-même facile à comprendre d'un point de vue technique et étant, à l'origine, compatible avec le système Märklin/Motorola (et à l'avenir avec le Märklin Systems).

Loin de nous l'idée de limiter l'utilisation de EDiTS Pro aux seuls utilisateurs de matériel Märklin et proposer un système de pilotage de train de réalisation personnelle universel pour tous les systèmes quelle que soit leur échelle.

Avec le lancement de la version 2.0 pour PC nous proposons de nouvelles fonctionnalités avec un programme totalement compatible avec Windows 98/SE/ME et XP.

Nous proposons, à l'intention de ceux de nos lecteurs auquel la notion de EDiTS Pro ne dit rien, un panorama succinct de ce qu'est ce système. Pour une description plus complète nous vous renvoyons à notre site où vous trouverez un article à télécharger gratuitement baptisé EPS050373-12F.

Unité de commande

Platine :	980085-1	Description : Livre EDiTS Pro
Contrôleur programmé (V 1.2)	010088-41	

Circuit d'adressage de régleur

Platine :	020125-1	Description : Elektor septembre 2002

Booster

Platine :	87291-6	Description du booster + interface V1.0 : Livre EDiTS Pro Description interface V1.2 (V2.0) : Elektor septembre 2002

Booster « petit budget » pour EDiTS Pro

Platine :	014098-1	Description : Elektor numéro double 2003

Décodeur d'aiguillage et/ou de signaux

Platine :	87291-1	Description : Livre EDiTS Pro

Décodeur de signal universel

Platine :	87291-4	Description : Livre EDiTS Pro

Décodeurs de locomotive

Platine du superdécodeur de locomotive H0 :	990071-1	Livre EDiTS Pro
PIC programmé superdécodeur de locomotive :	020094-41	

Il existe des sets de 5, 10 et 20 platines avec PIC programmés (020094-C5, -C10 et -C20)

Décodeur-N en CMS, PIC + platine : Site Web Edits Pro (www.editspro.com)

Module de détection de train

Platine :	87291-8	Description : Livre EDiTS Pro

Répondeur d'adresse de locomotive

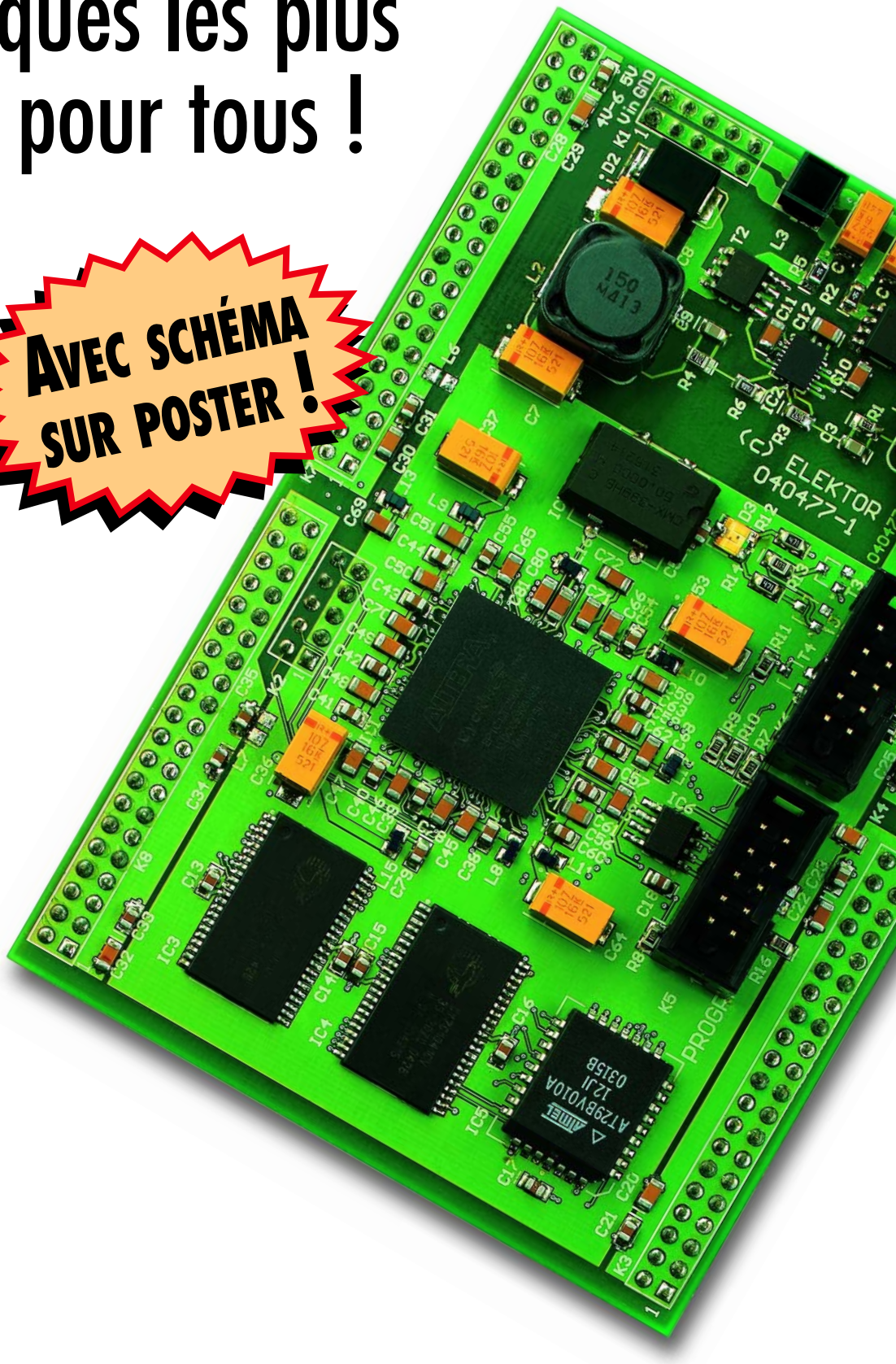
Platine :	990084-1	Description : Livre EDiTS Pro
PIC programmé :	020095-41	Description : Livre EDiTS Pro

« Briques » FPGA fl

Les techniques les plus modernes pour tous !

Paul Goossens

AVEC SCHÉMA SUR POSTER !



Qui arrêtera la marche des FPGA ? Personne. Jusqu'à tout récemment, travailler avec ce type de « super-composant » était réservé aux spécialistes de développements high-tech. Les choses ont bien changé avec la chute des prix et la mise à disposition de logiciels de développement adéquats. Il nous a paru temps de vous présenter cette technologie dans les colonnes d'Elektor. Nous avons à cet effet développé un module FPGA qui constituera le coeur numérique de différents projets proposés dans les prochains numéros de ce magazine.

lexibles

Caractéristiques techniques :

- **Altera Cyclone FPGA**
- **12 060 éléments logiques**
- **4 Moctets de mémoire de configuration**
- **8 Moctets de SRAM utilisateur**
- **1 Moctet de RAM Flash utilisateur**
- **Horloge 50 MHz embarquée**
- **Interface JTAG et de programmation**
- **Compatible Byteblaster**
- **80 signaux d'E/S utilisateur**
- **Signaux d'horloge dédiés**
- **LED d'indication**
- **Alimentation à découpage intégrée**
- **Platine (multicouche) compacte (110 x 77 mm)**
- **Disponible prête à l'emploi**

Lorsque l'on veut utiliser une FPGA dans un projet il y a un certain nombre d'éléments périphériques communs qui restent toujours nécessaires. À commencer par une interface de programmation, une mémoire de configuration, etc. Il nous a paru être intéressant, pour ne pas avoir, à chaque fois, à réinventer la roue, de mettre à disposition un module d'électronique comportant tous les éléments entourant normalement une FPGA, ce module pouvant ensuite être utilisé comme « coeur numérique » dans des réalisations très variées. L'avantage majeur de cette solution est que le concepteur peut se concentrer son application spécifique et faire confiance, en ce qui concerne la FPGA, à la qualité de son module FPGA.

Tout début est difficile

Si vous faites partie des personnes n'ayant encore jamais eu affaire aux FPGA, le présent module combiné à la carte d'expérimentation décrite dans ce même magazine, constitue un excellent tremplin pour une première prise de contact avec les FPGA.
Un montage de ce genre implique inévitablement l'utilisation de CMS. Plus délicat encore, nous avons opté pour ce module, pour une FPGA en boîtier dit BGA (*Ball Grid Array*) qu'il est tout simplement impossible de souder à l'aide d'un fer à souder. Même la soudure de son homologue (en boîtier PQFP) est une opération extrêmement délicate très risquée.
L'utilisation de ces composants « impossibles » a cependant permis de réaliser un module relativement compact.
Vous n'avez heureusement pas à vous soucier de l'aspect réalisation de ce module. Nous vous proposons en effet un module FPGA pratiquement terminé, il ne vous reste qu'à mettre en place les connecteurs et à les souder.
Il nous faut, outre ce module, également une interface de programmation faisant le pont entre le module le PC. Nous avons bien entendu développé notre propre interface pour cela.

Une seule FPGA ne fait pas le... module !

L'utilisation d'une FPGA implique, nous le disions plus haut, une électronique connexe nécessaire au fonctionnement effectif de cette puce complexe. L'un des ces éléments connexes les plus important est, ce que l'on appelle la mémoire de configuration. Cette mémoire ne perd pas, contrairement à la majorité des FPGA, l'information qu'elle contient lorsque la tension d'alimentation disparaît.
Lors de la remise sous tension du circuit, il faut commencer par reconfigurer la FPGA avant que cette dernière ne puisse remplir la fonction requise. Les fabricants de FPGA ont développé, à cette intention, des circuits de mémoire spéciaux capables, à la mise sous tension, de configurer une FPGA automatiquement. C'est ce type de mémoire que nous avons utilisé dans notre module FPGA.
Il est extrêmement souhaitable qu'il soit possible, en cours de phase de développement, de programmer tant la FPGA que la mémoire de configuration. Ceci explique, en périphérique standard, la présence d'une interface de programmation (via JTAG).
L'alimentation est une autre partie (très) importante d'un tel montage. En interne, une puce de ce genre travaille à une tension relativement faible, 1,5 V dans le cas présent. L'alimentation doit être capable de bien encaisser des brèves crêtes de tension. Les entrées et sorties requièrent elles une autre tension, que nous avons, ici, fixée à 3,3 V. Cette tension d'alimentation doit elle aussi être capable de fournir des courants relativement importants tout en restant parfaitement stable.
Nous avons en outre doté notre module d'un oscillateur, de mémoire SRAM et d'une mémoire Flash. Ces éléments sont à votre disposition pour votre propre application.
Il est facile, sur le schéma, d'identifier ces différents sous-ensembles.

Le projet

En raison de la taille de la totalité du schéma du module FPGA, nous lui avons, dans le présent numéro, consacré un poster.
Prenons les choses au début : l'alimentation. Son coeur est un TPS75003, IC2, circuit s'accommodant de toute tension d'entrée comprise entre 4,5 et 6,5 V. Ce circuit intégré a été développé spécifiquement pour les électroniques à base de FPGA. Il comporte 2 alimentations à découpage et un régulateur-série. Nous n'avons pas utilisé cette dernière fonction ici. T1, D1, L1, C1 et les composants connexes constituent un régulateur-abaisseur (*stepdown*) fournissant une tension de 3,3 V.
Périodiquement, IC2 commande la mise en conduction du FET T1 se sorte que l'on a circulation de courant, depuis la tension d'entrée, par le biais de T1, L1 et C1. Le courant traversant la self croît ainsi permettant la charge du condensateur électrochimique C1. Lorsque IC2 bloque le FET, le courant aux bornes de la self y reste un court instant. Il poursuit son chemin via C1 et D1. R1, R3 et C3 assure la contre-réaction, ce qui permet à IC2 de savoir s'il faut plus d'énergie ou, au contraire,

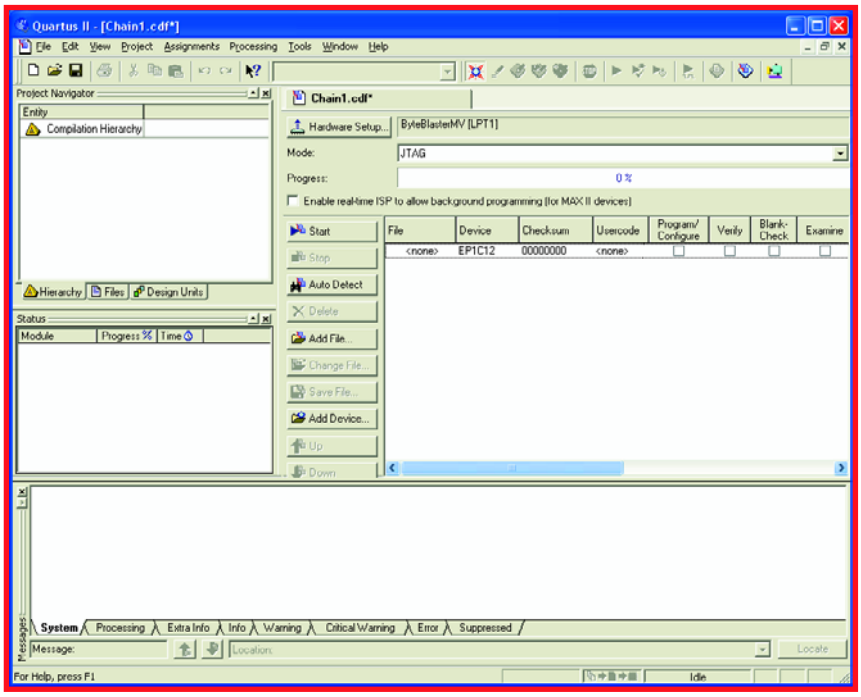


Figure 1. Le mode de programmation du programme Quartus.

moins. Les valeurs données à ces composants sont telles que IC2 fait de son mieux pour maintenir à 3,3 V la tension en sortie. R2 est une résistance de shunt. Lorsque la tension à ses bornes se met à dépasser 0,1 V, IC2 limite le courant. De par la valeur de 50 mΩ adoptée, cette limitation de courant entre en fonction à 2 A.
T2 se trouve au coeur d'une électronique identique, aux valeurs des résistances de contre-réaction près. De par leur dimensionnement, la tension de sortie est ici de 1,5 V.
Les condensateurs électrochimiques pris à l'entrée servent à tamponner la tension d'entrée. Ils ont également pour fonction de « localiser » les courants de commutation rapides. Ceci évite la naissance de parasites néfastes pour les sous-ensembles environnants.
La FPGA est alimentée par ces tensions, mais par un détour. La tension d'alimentation de chaque section de la FPGA est d'abord envoyée faire un tour par une perle de ferrite qui bloque une grande partie du rayonnement HF. À proximité immédiate de la puce, les tensions sont tamponnées une dernière fois par quelques condensateurs. On peut penser qu'il y a excès de précautions, mais on peut avoir, en fonction du circuit électronique constitué finalement par la FPGA, circulation de courants de commutation relativement importants travaillant à des fréquences élevées. Il est toujours bon de veiller à ce que ce type de courants circule de la façon la plus localisée possible. Si on n'y veille pas, un tel module peut être une source de parasites monstrueuse !

Passons au numérique

Le circuit « se trouvant sous tension », nous pouvons maintenant nous intéresser d'un peu plus près à la partie numérique. Comme nous le disions déjà à plusieurs reprises, c'est en effet important, la FPGA a besoin d'une mémoire de configuration, qui prend ici la forme physique de **IC6**, composant conçu pour travailler avec la FPGA. Lorsque les signaux MSEL de la FPGA se trouvent à la masse, cette puce sait qu'il existe une mémoire de configuration telle qu'un EPCS4. À partir de là, la FPGA prend le contrôle de ce circuit et en utilise les données pour se configurer elle-même à partir des informations qu'elles contiennent. Il est difficile de faire plus simple !
Pour pouvoir assurer la programmation de **IC6**, il est prévu une embase de programmation, **K5**. Le brochage de ce connecteur est tel que l'interface de programmation est compatible avec celle d'Altera (connue sous le nom de Byteblaster).
La programmation en circuit (ICP) de la FPGA depuis le PC se fait elle par l'embase K6. Cela évite d'avoir à commencer par une programmation du projet dans la mémoire de configuration. Cette embase peut également servir à effectuer un petit test rapide. Là encore, le brochage de l'embase est compatible avec l'interface Byteblaster. Les transistors T3 à T5 et les composants connexes constituent un indicateur qui visualise l'état de la FPGA à l'aide de la LED bicolore D3. Cette LED nous apprend si la FPGA est ou non déjà configurée. D3 nous informe également si les choses ne se passent pas comme prévu.

Des lignes d'E/S à ne plus quoi savoir en faire

Ayant prévu d'utiliser ce module pour de nombreuses applications à venir, nous souhaitons qu'il dispose d'un maximum de broches d'E/S. Ces dernières sont accessibles par le biais des embases **K3**, **K4**, **K7**, **K8** et **K2**. Ces embases sont du type 0,1 " (écartement des contacts) standard. Pas de problème de connexion pour vos propres matériels.
Les embases K3, K4, K7 et K8 relient pas moins de 80 (!) lignes d'E/S de la FPGA au monde extérieur. Ces signaux se trouvent tous sur la même rangée de contacts des embases. Sur les contacts opposés on dispose, alternativement, de la masse et du 3,3 V. Cette tension d'alimentation peut servir à alimenter vos propres circuits à condition bien entendu que leur consommation ne soit pas trop importante. Cette tension d'alimentation est elle aussi filtrée au niveau du module à l'aide d'une perle de ferrite et de 2 condensateurs.

L'embase **K2** présente des spécificités. On y trouve, outre la masse, quelques signaux spécifiquement destinés aux signaux d'horloge. Le contact 10 est une entrée par le biais de laquelle il est possible d'appliquer à la FPGA un signal d'horloge externe devant servir d'horloge. Le reste des broches est relié aux sorties des PLL internes présentes dans la FPGA.
Rien n'empêche en effet d'appliquer un signal d'horloge à une autre entrée quelconque de la FPGA, mais ces broches sont spécialement prévues à cet effet.

Quoi d'autre ?

Nombre de réalisations à base de FPGA requièrent de la mémoire. La FPGA dispose déjà de sa propre

mémoire, mais elle est souvent de capacité insuffisante. Comme il est fréquent d'utiliser ce que l'on appelle un processeur à « noyau mou » (*softcore*), une mémoire Flash peut s'avérer intéressante pour le stockage du progiciel (*firmware*) de ce processeur. Cela nous permet d'intégrer des logiciels plus volumineux dans notre projet sans que cela n'occupe trop de place supplémentaire dans la FPGA. La mémoire Flash prend ici la forme de **IC5** du schéma.
Disposer de mémoire RAM additionnelle peut être intéressant aussi. Ceci est pratique non seulement lorsque l'on intègre un ou plusieurs processeurs « softcore » dans la FPGA, mais aussi lorsque l'on procède à toutes sortes de traitements de signaux, il peut être pratique de disposer d'un « rien » de mémoire RAM additionnelle.
La mémoire RAM prend la forme de **IC3** et **IC4**. Ces composants ont une capacité de 4 Mbits organisée en 256 Kmots de 16 bits. Les signaux $\overline{\text{BHE}}$ et $\overline{\text{BLE}}$ sont utilisables comme 2 lignes $\overline{\text{CE}}$ (*Chip Enable*) distinctes, BHE étant la ligne $\overline{\text{CE}}$ pour l'octet de poids fort (MSB, D8 à D15) du bus de données, BLE remplissant une fonction similaire pour l'octet de poids faible (D0 à D7). Il est possible ainsi d'utiliser aussi cette mémoire comme une mémoire de 512 Koctets (512 Kmots x 8 bits).

IC3 est relié directement à la FPGA, **IC4** partageant son bus de données et son bus d'adresses avec la mémoire Flash (**IC5**). Ceci a été fait à dessein de manière à disposer d'un nombre suffisant de broches d'E/S de la FPGA pour les embases « utilisateur ». Ceci a cependant pour corollaire que la FPGA ne peut pas lire et écrire simultanément dans la mémoire RAM ou la mémoire Flash. Dans la pratique cela ne pose pas de gros problème, surtout pas dans le cas de processeur à « noyau mou ».
Nous en avons terminé avec la description des sous-ensembles de cette prestigieuse réalisation.
Comme nous le disions plus haut, le module FPGA est proposé « tout monté », tout ce qu'il vous reste à faire est de le doter des 8 embases qui l'accompagnent. Les connecteurs K2, K3, K4, K7 et K8 sont montés sur le dessous de la platine, les 3 connecteurs restants l'étant sur le dessus (côté composants). Vérifiez bien le positionnement correct du contact n°1 des connecteurs, la sérigraphie l'identifiant par un « 1 » !

Interface de programmation

Tel qu'il est là, le module FPGA est encore inerte. Il nous faut commencer par programmer la FPGA et/ou la mémoire de configuration, d'où le besoin d'une interface de programmation. Le poster comporte également le schéma de cette interface de programmation. Elle interconnecte la FPGA ou la mémoire de configuration au port parallèle d'un PC. L'interface de programmation est compatible avec la Byteblaster d'Altera et peut être pilotée par le biais du logiciel gratuit mis à disposition par Altera. L'électronique est la plus simple possible.
L'alimentation de l'ensemble se fait par le biais du 3,3 V en provenance du module FPGA. Les 2 circuits intégrés font office de tampon entre les signaux +3,3 V provenant du module FPGA et les signaux +5 V du PC. Les résistances de 100 Ω servent à éviter que les signaux +5 V n'entraînent une augmentation trop importante de la tension d'alimentation.
L'interface est reliée, par le biais de **K3**, au port parallèle du PC. **K2** est une embase HE-10 à 2 rangées de 5 contacts reliée, par le biais d'un morceau de câble plat à 10 conducteurs doté des connecteurs adéquat, à la platine du module FPGA.
Si nous voulons, à titre de test uniquement, programmer la FPGA, il faut que ce câble soit relié au connecteur **K6** (JTAG) du module FPGA. Si par contre nous voulons programmer le module pour lui donner son autonomie, le dit câble devra être relié au connecteur **K5** (PROGRAM) du module FPGA.

Le logiciel

Un tel projet ne saurait se passer de logiciel de bon aloi. Le fabricant de la FPGA utilisée ici, Altera, propose un beau set logiciel pour le support de ses FPGA. La version gratuite de ce logiciel s'appelle « Quartus Web Edition »; elle est téléchargeable depuis le site d'Altera (www.altera.com). *Un beau morceau puisque, à l'écriture de ces lignes, il ne « vaut » pas moins de 240 Moctets. Rassurez-vous, il existe la possibilité de demander un CD-ROM gratuit par le biais du site d'Altera.*
Une fois le logiciel installé, opération qui ne présente pas le moindre problème, il vous faut obtenir une licence; vous pouvez la demander à nouveau par le biais du site d'Altera sachant qu'elle est valable pendant 6 mois. Il faudra ensuite, une fois ce délai écoulé, renouveler la licence. Que les sceptiques soient rassurés, le fabricant nous a assuré qu'il continuerait de proposer ce logiciel gratuitement.

Essais

Bien que le circuit dans son état actuel n'ait pas encore de vraies possibilités d'entrées/sorties, nous pouvons quand même procéder à un test simple.
Une fois le logiciel installé, coupez momentanément votre PC. Ceci fait, connectez l'interface de programmation au port parallèle de votre PC. Reliez l'interface au module FPGA à l'aide du câble en nappe à 10 conducteurs accompagnant le module. Vous pouvez connecter ce câble plat au module FPGA par le biais de son embase **K6** (**JTAG**). Redémarrez le PC et alimentez le module FPGA (sans jamais dépasser 6,5 V !). Démarrez maintenant Quartus et cliquez sur l'icône de programmation présente dans le barre de menu ou encore cliquez sur le menu *Tools* puis l'onglet *Programmerr* (il n'y a pas de différence entre ces 2 approches). La fenêtre de programmation s'ouvre alors. Cliquez sur le bouton *Hardware Setup* et dans la nouvelle fenêtre sur *Add Hardware*. Sélectez le Byteblaster et indiquez à quel port imprimante l'interface de programmation est connectée (LPT1: dans la majorité des cas).
Si nous cliquons maintenant sur le bouton *Autodetect* de la fenêtre principale, le logiciel détectera automatiquement notre FPGA (EP1C12). Nous sommes assurés ainsi que la FPGA fonctionne et que la tension d'alimentation est bien présente.

(040477-1)

Le module FPGA est disponible tout monté sous le numéro EPS040477-91.

Produits au programme

- Module FPGA (prêt à l'emploi)
- Interface de programmation (prête à l'emploi)
- Câble de programmation
- Câble de liaison PC -> interface de programmation
- 8 connecteurs séparés

Carte d'expérimentation

Paul Goossens

VGA

Une sortie VGA permettant l'affichage de textes et d'images sur un écran de PC. Quelques composants standard, il n'en faut pas plus.

Cf. page 30

Ethernet

Notre connexion à Internet. La puce de ce sous-ensemble assure le couplage entre les parties analogique et numérique du montage.

Cf. page 30

USB

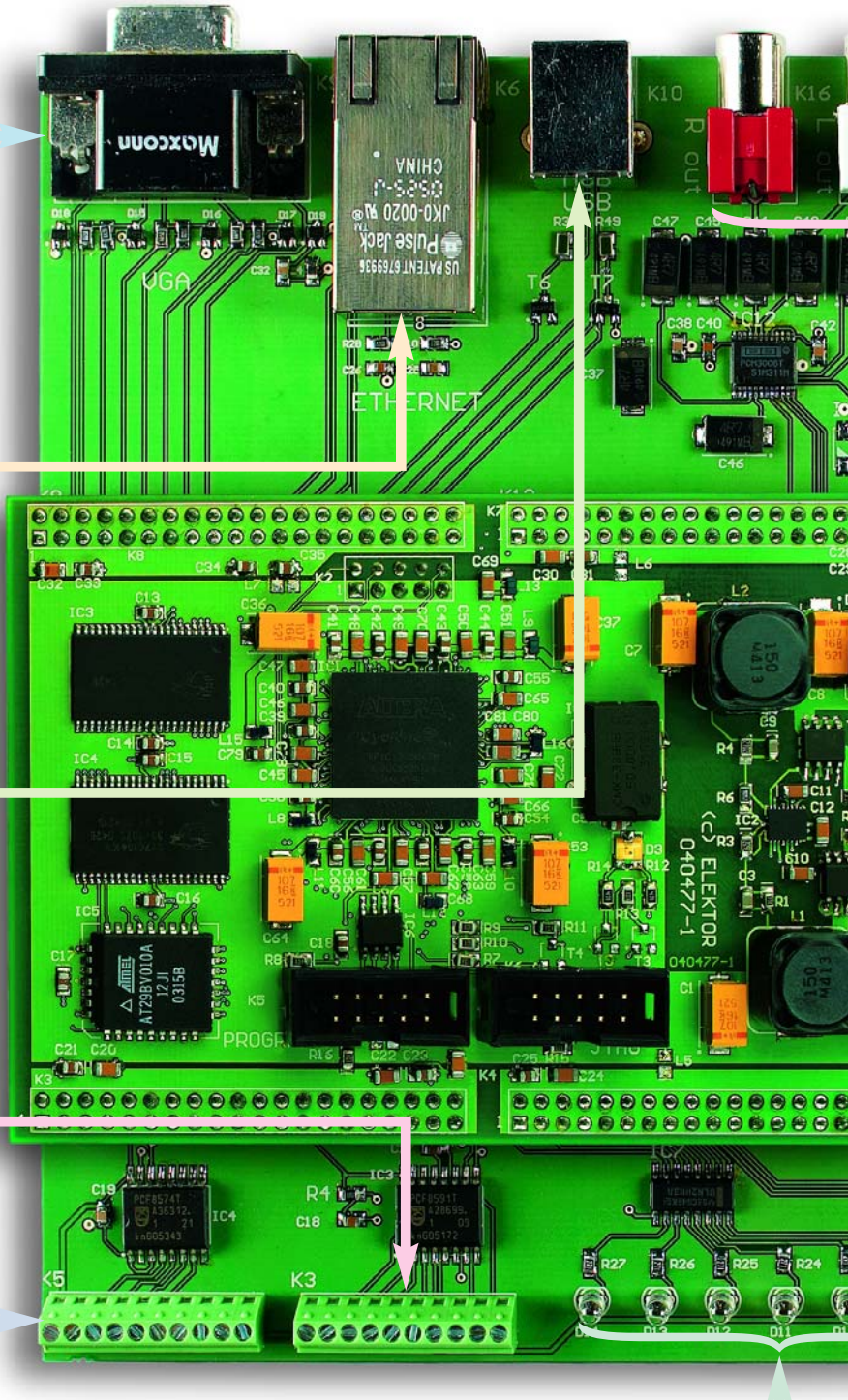
Cette interface USB nous permet d'établir une communication rapide avec le PC. Combien de composants ? 5 seulement, embase y comprise !

Cf. page 30

E/S analogiques

4 entrées analogiques + 1 sortie analogique. En dépit de son caractère numérique très prononcé, cette carte se doit de disposer d'une interface analogique. Cette E/S possède une résolution de 8 bits, ce qui suffit pour la majorité des applications.

Cf. page 30



E/S numériques

Ces embases permettent la connexion d'un montage de votre cru à cette carte d'expérimentation.

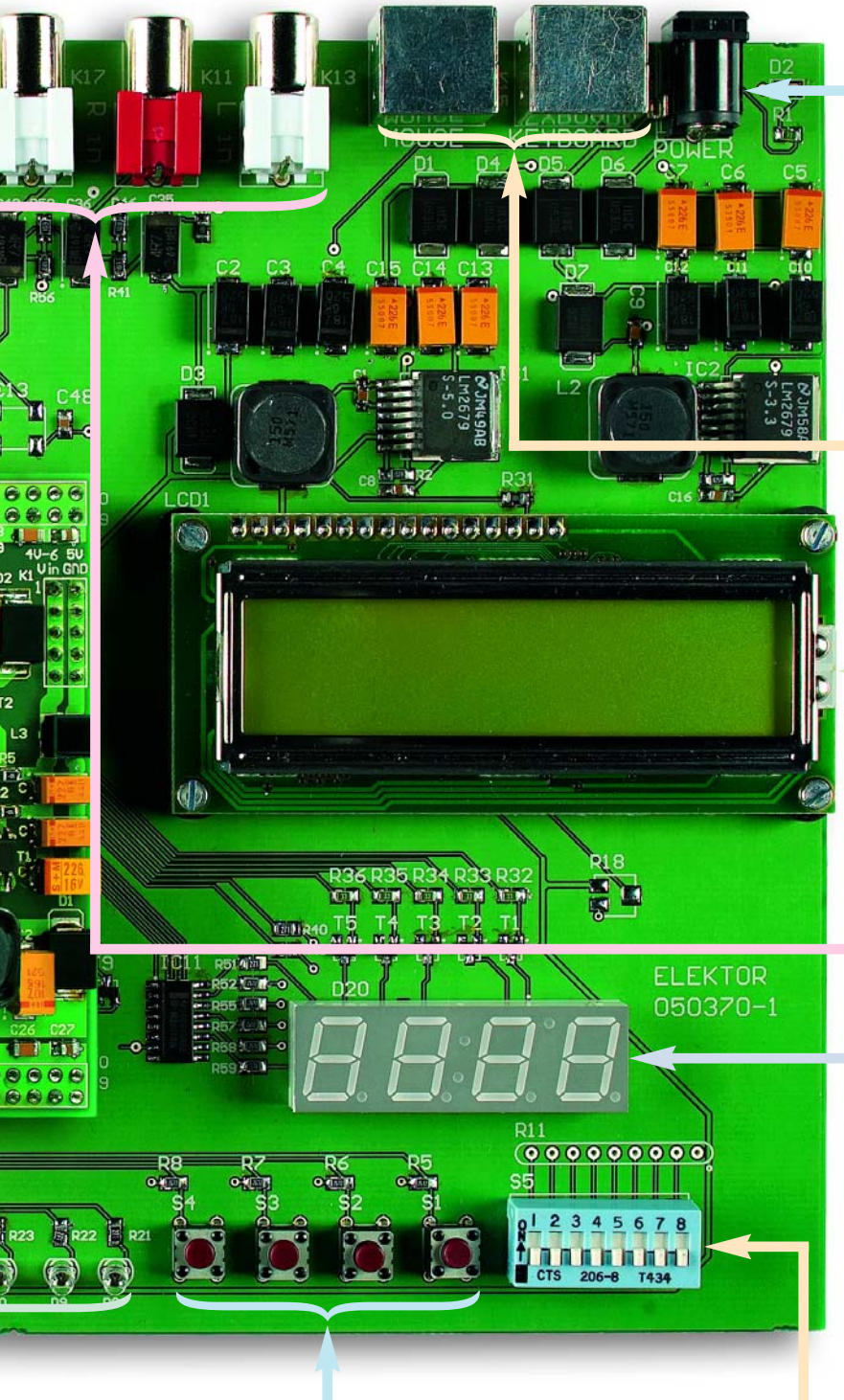
Cf. page 30

LED

Ces LED sont utilisables à de nombreuses fins et peuvent êtres utiles lors du débogage.

Cf. page 30

FPGA Souffle de vie pour les FPGA



Alimentation

Tant que la tension d'alimentation se situe entre 6 et 20 V, la régulation se charge du reste. Rien à craindre d'une inversion malencontreuse de polarité de la tension d'alimentation, l'alimentation s'en fiche.

Cf. page 31

PS/2

2 embases pour la connexion et d'un clavier et d'une souris pour PC.

Cf. page 31

LCD

Affichage LCD à 2 lignes de 16 caractères au maximum. Toute carte d'expérimentation digne de ce nom devrait en posséder un.

Cf. page 31

E/S Audio

En ce qui concerne l'audio également, cette carte d'expérimentation n'a pas à rougir. Entrée et sortie 16 bits.

Cf. page 31

Afficheurs

Il n'existe rien de plus parlant pour la visualisation de chiffres. L'idéal pour afficher une date ou une heure.

Cf. page 31

Boutons

On est souvent heureux de pouvoir disposer d'une méthode de commande simple.

Cf. page 30

Interrupteurs DIP

Permet d'activer ou d'inhiber des options. Peuvent également servir d'interrupteurs séparés.

Cf. page 30

La carte d'expérimentation présentée ici est le premier montage dont le module FPGA devient l'intelligence. Elle vous permettra de vous faire une meilleure idée des possibilités de notre module FPGA. Nous ne manquerons bien entendu pas de vous guider vos premiers pas dans le monde des FPGA à l'aide d'un cours « s'étalant » sur plusieurs numéros à venir.

La carte d'expérimentation permet d'utiliser la FPGA à différentes fins. Comme le montre le tableau des caractéristiques, cette carte est dotée des embases de connexion les plus modernes.

Concevoir une réalisation au coeur de laquelle bat une FPGA est, pour le plus grand nombre de nos lecteurs, une expérience totalement nouvelle. Ceci explique que nous ayons choisi de vous proposer dès le mois prochain, un cours « Programmer une FPGA ». Le cobaye des expériences effectuées tout au long de ce cours sera la présente carte d'expérimentation.

Tout comme c'est le cas pour le module FPGA, cette carte d'expérimentation sera disponible toute montée elle aussi. >>>

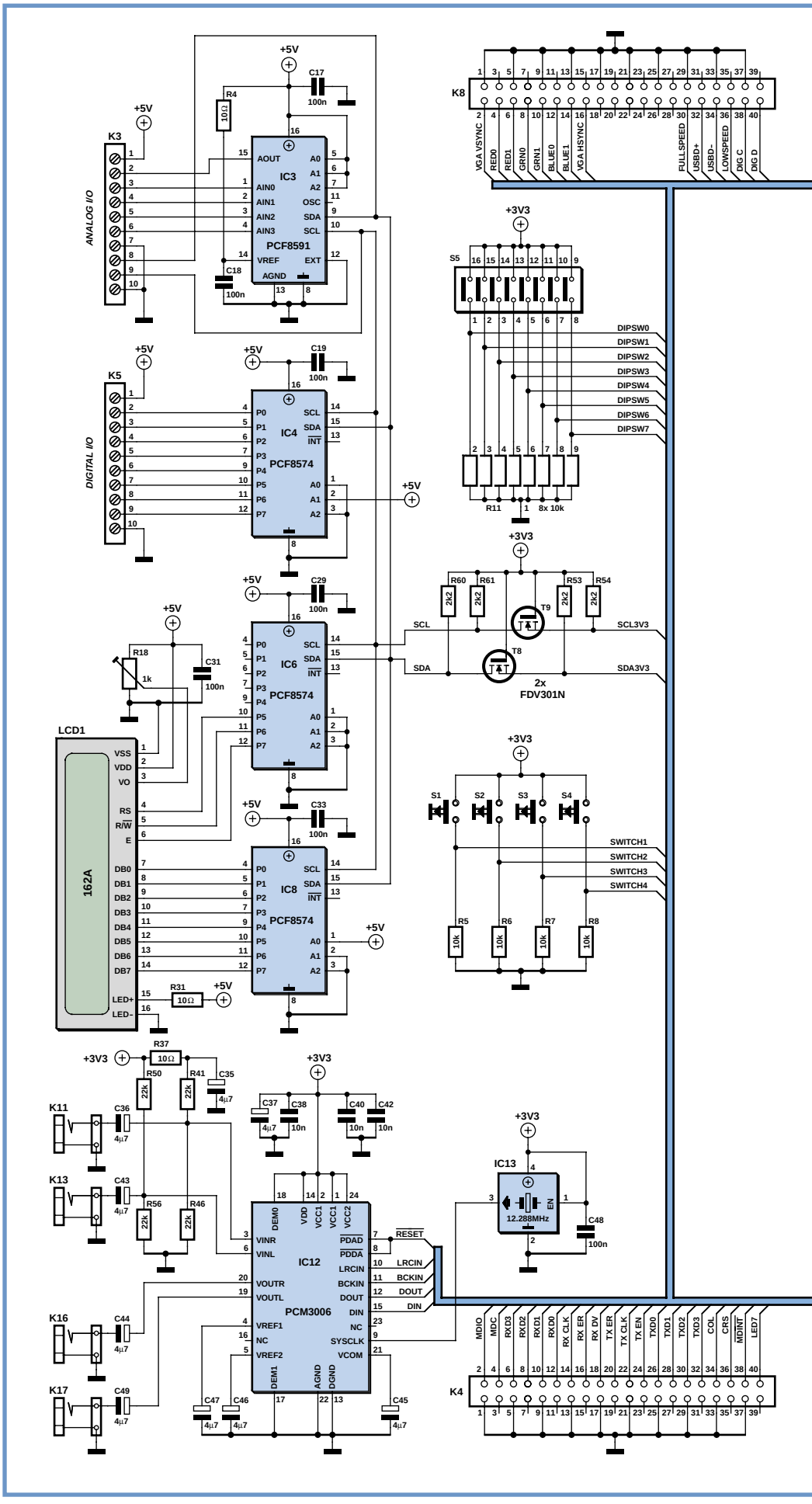
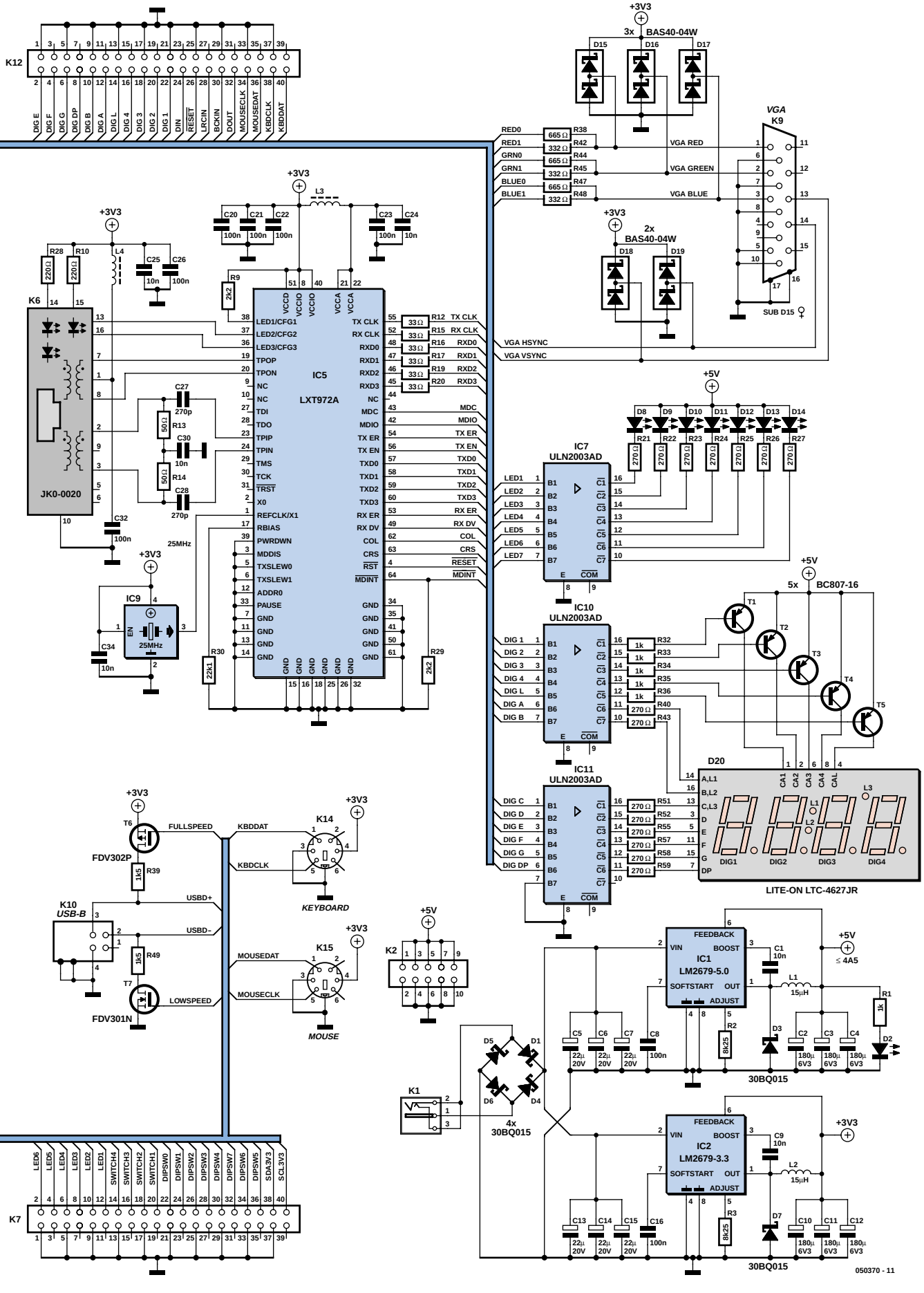


Figure 1. Schéma de l'électronique constitutive de la carte d'expérimentation FPGA. Le dessin des pistes au format .pdf est disponible gratuitement au téléchargement sur www.elektor.fr.


K12
DIG E
DIG F
DIG G
DIG DP
DIG B
DIG A
DIG L
DIG 4
DIG 3
DIG 2
DIG 1
DIN
RESET
LRCIN
BCKIN
DOUT
MOUSECLK
MOUSEDAT
KBDCLK
KBDDAT
+3V3
3x BAS40-04W
D15
D16
D17
VGA
K9
RED0
RED1
GRN0
GRN1
BLUE0
BLUE1
665 Ω
332 Ω
R38
R42
R44
R45
R47
R48
VGA RED
VGA GREEN
VGA BLUE
2x BAS40-04W
D18
D19
SUB D15
VGA HSYNC
VGA VSYNC
L3
C20
C21
C22
C23
C24
100n
10n
R28
R10
220Ω
L4
C25
C26
R9
2k2
K6
JK0-0020
IC5
LXT972A
VCCD
VCCIO
VCCIO
VCCA
VCCA
LED1/CFG1
LED2/CFG2
LED3/CFG3
TPOP
TPON
NC
NC
TDI
TDO
TPIP
TPIN
TMS
TCK
TRST
X0
REFCLK/X1
RBIAS
PWRDWN
MDDIS
TXSLEW0
TXSLEW1
ADDR0
PAUSE
GND
TX CLK
RX CLK
RXD0
RXD1
RXD2
RXD3
MDC
MDIO
TX ER
TX EN
TXD0
TXD1
TXD2
TXD3
RX ER
RX DV
COL
CRS
RST
MDINT
33 Ω
R12
R15
R16
R17
R19
R20
C27
270p
R13
50 Ω
C30
10n
R14
C28
C32
25MHz
IC9
EN
C34
R30
22k1
R29
+5V
D8
D9
D10
D11
D12
D13
D14
R21
R22
R23
R24
R25
R26
R27
270 Ω
IC7
ULN2003AD
IC10
ULN2003AD
IC11
ULN2003AD
LED1
LED2
LED3
LED4
LED5
LED6
LED7
DIG 1
DIG 2
DIG 3
DIG 4
DIG L
DIG A
DIG B
DIG C
DIG D
DIG E
DIG F
DIG G
DIG DP
B1
B2
B3
B4
B5
B6
B7
C1
C2
C3
C4
C5
C6
C7
E
COM
1k
R32
R33
R34
R35
R36
R40
R43
R51
R52
R55
R57
R58
R59
5x BC807-16
T1
T2
T3
T4
T5
D20
A,L1
B,L2
C,L3
D
E
F
G
DP
CA1
CA2
CA3
CA4
CAL
L1
L2
L3
DIG1
DIG2
DIG3
DIG4
LITE-ON LTC-4627JR
FULLSPEED
KBDDAT
KBDCLK
K14
KEYBOARD
FDV302P
T6
1k5
R39
USBD+
USBD-
K10
USB-B
R49
T7
LOWSPEED
FDV301N
MOUSEDAT
MOUSECLK
K15
MOUSE
K2
K1
4x 30BQ015
D5
D1
D6
D4
IC1
LM2679-5.0
FEEDBACK
VIN
BOOST
SOFTSTART
OUT
ADJUST
≤ 4A5
C1
10n
L1
15µH
R1
1k
R2
8k25
D3
D2
C5
C6
C7
C8
100n
22µ
20V
C2
C3
C4
180µ
6V3
30BQ015
IC2
LM2679-3.3
C9
C10
C11
C12
C13
C14
C15
C16
R3
D7
K7
LED6
LED5
LED4
LED3
LED2
LED1
SWITCH4
SWITCH3
SWITCH2
SWITCH1
DIPSW0
DIPSW1
DIPSW2
DIPSW3
DIPSW4
DIPSW7
DIPSW6
DIPSW5
SDA3V3
SCL3V3
050370 - 11

VGA

Notre carte d'expérimentation comporte même sa propre sortie VGA. Son électronique est centrée sur K9. Une demi-douzaine de résistances constituent une triplette de CNA (Convertisseur Numérique/Analogique) au coût imbattable. Les signaux de synchronisation peuvent rester numériques eux. Comme il faut s'attendre à des réflexions, nous avons, à l'aide de quelques diodes Schottky rapides, D15 à D19, protégé tous ces signaux contre d'éventuelles oscillations.

La création d'un signal VGA est bien plus facile qu'il n'y paraît au premier abord. Nous verrons dans la partie 7 du cours FPGA comment s'y prendre.

Ethernet

L'interface Ethernet repose sur IC5. Ce circuit intégré se charge de la conversion de signaux numériques en signaux analogiques et inversement, en respect du standard Ethernet. Le pilotage de cette interface est relativement complexe et fera l'objet de l'article 9 du cours FPGA dans lequel nous procéderons à quelques expériences ayant trait aux réseaux en nous servant de cette interface.

USB

L'embase K10 se trouve au coeur de cette fameuse interface USB. Nous pouvons faire en sorte que notre interface s'énumère, comme périphérique lent ou rapide. Le PC reconnaît un périphérique rapide au fait qu'il force la ligne USB D+ au + par le biais d'une résistance de forçage au niveau haut (pull-up). Un périphérique lent forcera lui la ligne D– à la masse par le biais d'une résistance de forçage au niveau bas (pull-down).

L'activation ou non, sur la carte d'expérimentation, des FET T6 ou T7 permet de piloter le mode de fonctionnement depuis la FPGA. Les 2 lignes de données sont à nouveau reliées directement à la FPGA.

Nous procéderons, dans l'article 8, à des expériences ayant trait au bus USB.

Entrées/Sorties analogiques

Outre ses capacités en numérique, notre platine comporte aussi 4 entrées analogiques ainsi qu'une sortie analogique. Ces connexions d'E/S à 8 bits prennent la forme de IC3. Ce circuit intégré aussi possède une interface I2C. Ceci explique que nous abordions ces capacités d'E/S dans la partie 3 du cours FPGA.

Entrées/Sorties numériques

Il nous semble logique que cette carte d'expérimentation soit utilisée en tant qu'unité de commande d'un dispositif quelconque. 8 lignes d'E/S numériques permettent la connexion à notre carte d'expérimentation d'une électronique que vous aurez conçue vous-même.

Le circuit intégré utilisé à cet effet, IC4, possède une interface I2C. Cette dernière va nous servir à communiquer avec un autre circuit intégré que nous ne tarderons pas à rencontrer.

La circuiterie basée sur T8 et T9 vient d'un autre projet d'Elektor. Elle convertit les signaux I2C de 3,3 V en signaux I2C de 5 V.

Le protocole utilisé par le bus I2C est un protocole sériel. Nous verrons, dans l'article numéro 3, comment utiliser une FPGA à cet effet.

LED

Les LED D8 à D14 se laissent piloter directement par le module FPGA par le biais des signaux LED1 à LED7. IC7 fait ici office de tampon pour ces signaux. Dès qu'une entrée présente un niveau logique haut la sortie correspondante est reliée à la masse. Les résistances R12 à R17 servent à limiter le courant au travers des LED.

Ce mode de pilotage est l'enfance de l'art. Dans le premier article de notre cours FPGA nous utiliserons ces LED afin de tester différentes fonctions numériques.

Boutons-poussoirs

L'un des organes de saisie d'entrée sur notre carte d'expérimentation prend la forme de boutons-poussoirs, S1 à S4. Par le biais des résistances R5 à R8, les signaux SWITCH1 à SWITCH4 restent forcés au niveau bas tant qu'ils n'ont pas été actionnés. En cas d'action sur un bouton-poussoir, le signal correspondant se trouve forcé au niveau de la tension d'alimentation. Les signaux SWITCH1 à SWITCH4 vont directement, tout comme dans le cas des LED, vers le module FPGA.

Ces boutons-poussoirs permettent de commander l'état de 4 signaux d'entrée de la FPGA. Dans le cadre du premier article de notre cours FPGA nous utiliserons ces 4 boutons-poussoirs en tant qu'organes de saisie en entrée.

Interrupteurs DIP

L'interrupteur DIP comporte en fait 8 petits interrupteurs distincts. Leur connexion est similaire à celle des boutons-poussoirs S1 à S4. Ces interrupteurs permettent à l'utilisateur de piloter 8 signaux d'entrée de la FPGA additionnels.

L'une des utilisations les plus évidentes de ce type d'interrupteur est l'activation ou l'inhibition de différentes options. Nous les utiliserons dans le cadre sur second article du cours FPGA.

Afficheurs 7 segments

La platine comporte 4 afficheurs 7 segments à LED à point décimal, qui prennent place, avec une triplette de LED individuelles, dans un même boîtier (D20). Cet afficheur comporte plusieurs broches d'anode commune (pas moins de 5).

L'application d'une tension à chacune d'entre elles permet de choisir laquelle des 5 sections de cet afficheur doit s'allumer. 4 de ces 5 broches servent aux 4 afficheurs 7 segments (avec leur point décimal). La 5ème broche sert à l'alimentation des 3 LED individuelles.

Le pilotage de l'afficheur se fait à l'aide du même type de circuit intégré que dans le cas de la commande des LED. Certaines des sorties de IC10 et IC11 servent, sous la houlette de la FPGA, à forcer les anodes à la masse. 5 des broches servent à la commutation des transistors T1 à T5. Nous avons la possibilité ainsi de déterminer, par le biais des signaux DIG1 à DIG4 et DIG L, quelle est la partie de l'afficheur devant être active.

Le pilotage de cet afficheur est un peu plus délicat, sachant qu'il faut attaquer successivement les différentes parties de l'affichage. Nous verrons comment dans le second article de notre cours FPGA.

Entrées/Sorties Audio

L'électronique centrée sur IC12 constitue un ensemble Entrée/Sortie Audio 16 bits en stéréo. De par la fréquence de IC13, la fréquence d'échantillonnage est fixée à 48 kHz.

Ce circuit intégré communique avec la FPGA par le biais de 5 signaux. Nous verrons dans le 4ème article du cours FPGA comment les choses se passent. Nous proposerons bien entendu un exemple d'application dans laquelle la FPGA pilote ce circuit intégré.

LCD

Le pilotage de l'affichage LCD à 16x2 caractères (baptisé LCD1 sur le schéma) se fait par les circuits IC6 et IC8. Il s'agit de circuits au principe de fonctionnement similaire à la puce utilisée pour les E/S numériques.

Nous mettrons cet affichage LCD à contribution dans l'article V de notre cours FPGA.

PS/2

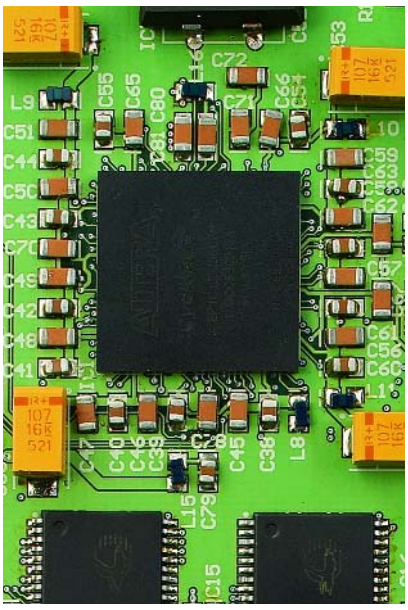
Les embases K14 et K15 permettent la connexion d'un clavier et d'une souris PS/2. Les 2 signaux présents sur ce bus attaquent directement les broches d'E/S du module FPGA. Comme pour le reste des E/S, la FPGA se charge d'un pilotage « réfléchi » de ces signaux. La FPGA est ainsi en mesure de traiter même le signal de données bidirectionnel sans nécessiter de puce de soutien externe. Cette communication se fait par un protocole sériel propriétaire auquel nous nous intéresserons dans la partie 6 du cours FPGA

L'alimentation

Il nous faut bien entendu une alimentation. C'est la seule partie du circuit qui ne soit pas pilotée par la FPGA !

IC1 et IC2 sont ce que l'on appelle des contrôleurs step-down (le fabricant a raison, on ne peut parler de régulateur de tension step-down qu'en combinaison avec les composants connexes) qui, épaulés par une self, une diode et un condensateur-tampon, constitue un convertisseur-abaisseur de tension régulateur. L'intérêt de ces convertisseurs est qu'ils ne dissipent que très peut d'énergie dans le cas de tensions d'entrée trop élevées, comme ce serait le cas avec une alimentation à régulation-série à base de 7805 par exemple.

La polarité de la tension appliquée au bornier K1 est sans conséquence. Les

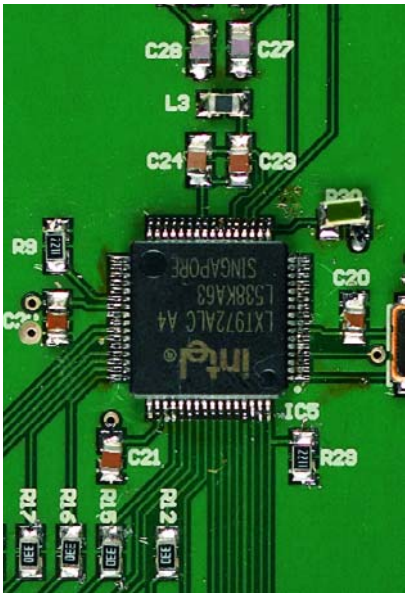


Le « moteur silencieux » du module FPGA.

diodes D1 et D4 à D6 redressent cette tension de sorte que sa polarité lorsqu'elle est appliquée à l'entrée de IC1 et IC2 est toujours correcte. Les condensateurs électrochimiques à l'entrée font en sorte que les courants de commutation rapides restent sur place, de sorte que cette alimentation ne parasite que très peu.

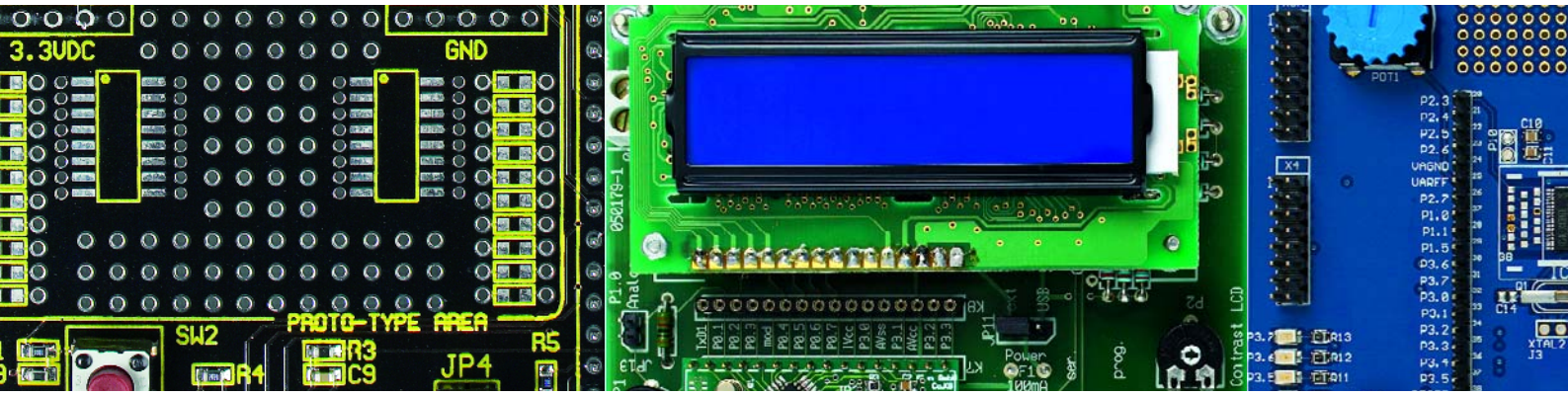
L'alimentation du module FPGA se fait par le biais du bornier K2.

(050370)



IC5 trône au cœur de l'interface Ethernet.

Kits de dévelop



Dans ce numéro consacré à ras bord aux FPGA et aux microcontrôleur, nous ne pouvons pas ne pas en parler, de quoi ?, des kits de développement. Nous allons nous intéresser à une sélection de kits intéressants, mais surtout abordables, en veillant à ne pas sélectionner des kits ne pouvant servir que dans un cadre professionnels mais aussi utilisables par l'amateur.

Les premiers pas sont toujours difficiles. Ceci est bien évidemment également vrai pour les FPGA, les microcontrôleurs et les DSP. Si les fabricants de circuits intégrés mettent à disposition ce qu'ils appellent des kits de développement (ou kits d'évaluation) ce n'est pas sans arrière-pensée : il est important de permettre à des développeurs de se familiariser rapidement avec leurs nouveaux produits. Si vous voulez vous y essayer chez vous, cela ne pose pas de problème non plus. Le distributeur local du fabricant en question se fera un plaisir de vous fournir un tel kit d'évaluation pour une somme souvent modique. Vous avez ensuite liberté de manoeuvre pour en faire ce que vous voulez. La première étape consiste cependant à déterminer quel kit il va vous falloir acheter. Ce choix peut s'avérer plus complexe qu'il n'y paraît au premier abord. Nous allons tenter, dans cet article, de vous proposer un panorama de kits intéressants et abordables, kits que nous avons classés en différentes catégories et dont nous avons récapitulé les caractéristiques techniques dans un grand tableau.

Possibilités de choix

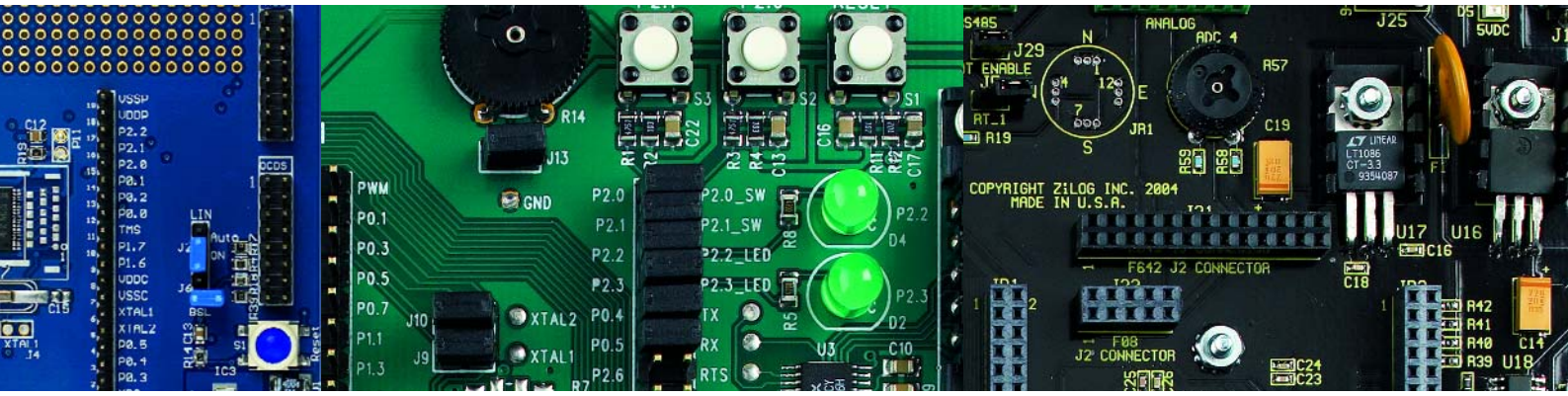
Avant de mettre à développer quoi que ce soit il vous faudra, nous le disions plus haut, faire votre choix parmi la pléthore de composants disponibles sur le marché. Il est judicieux, lors de la définition du cahier des charges, de ne pas penser uniquement à ce dont vous avez besoin dans l'immédiat mais d'effectuer une projection dans le futur pour essayer de prévoir quelles sont les extensions potentielles que vous pourriez envisager dans un proche avenir. Implanter un petit projet dans une FPGA ne pose pas de problème, mais y implémenter un projet plus important (lire : comportant un nombre de portes plus important) peut s'avérer un casse-tête insoluble. Ceci explique que souvent les kits de développement soient dotés de l'un des membres les plus puissants si ce n'est le plus puissant d'une famille. Dans le cas des microcontrôleurs aussi il vaut mieux tenir compte de souhaits futurs. Si ne pas utiliser toutes les entrées et sorties n'est pas un problème, essayer d'en utiliser plus qu'il n'en existe sur le composant est bien évidemment exclu.

Même le choix entre une FPGA et un microcontrôleur n'est pas nécessairement l'évidence même. Un microcontrôleur est moins complexe qu'une FPGA, mais cette dernière offre bien plus de possibilités qu'un microcontrôleur. Les kits de développement possède un certain nombre d'entrées et de sortie pour la communication avec des périphériques (qu'il leur faudra le cas échéant piloter). Le port RS-232, même s'il est quelque peu dépassé, reste l'une des interfaces favorites. On trouve souvent des embases un peu partout de manière à permettre un accès direct à chacune des broches du circuit intégré utilisé. L'une ou l'autre des cartes d'évaluation décrites ici possède quelques CAN (**C**onvertisseur **A**nalogique/**N**umérique) et CNA (**C**onvertisseur **N**umérique/**A**nalogique), les interfaces CAN, SPI et 1 Wire étant moins communes. Pour la programmation, chaque carte possède son propre type de connecteur. Microchip joue la carte de l'interface de programmation ICD, d'autres fabricants optant pour l'interface RS-232, JTAG voire plus spécifique encore. Ici encore il est bon de réfléchir quelques instants à ce dont on pourrait fort bien avoir besoin ultérieurement.

Le tableau récapitule certaines des caractéristiques les plus marquantes ce qui devrait vous permettre de faire un choix plus aisément. Nous allons, à l'intention du développeur en cours de formation, nous intéresser aux caractéristiques globales des microcontrôleurs, DSP et FPGA.

ppement

Qu'existe-t-il, quoi prendre ?



Les microcontrôleurs

Un microcontrôleur est une fait une sorte d'ordinateur sur une seule puce (cf. **figure 1**). L'unité de calcul est intégrée dans un circuit intégré avec toutes les entrées et sorties et les différents types de mémoire, de sorte que l'on peut se passer de puces externes additionnelles (ce qui est nécessaire dans le cas des microprocesseurs). Les domaines d'application des microcontrôleurs sont la commande d'appareils en tous genres, tel que notre four à refusion pour CMS (cf. Elektor de janvier 2006).
La différence la plus sensible entre les microcontrôleurs 8 et 16 bits se trouve dans leur vitesse. Un microcontrôleur 16 bits peut, comparé à un microcontrôleur 8 bits, pour la même puissance exprimée en MIPS (*Million Instructions Per Second*), traiter une quantité double de données.
La plupart des cartes d'évaluation sont conçues de manière à pouvoir fonctionner de façon autonome. Ceci signifie que l'on y trouve, le plus souvent, un microcontrôleur déjà programmé pour exécuter un programme (de démonstration). On a de plus une possibilité de simulation du fonctionnement de la carte dans le logiciel de support et si le fonctionnement n'est pas correct à 100%, d'éliminer les dernières petites erreurs à l'aide du mode de débogage.
Dans un cadre, nous vous proposons quelques repères pour le choix d'un type de microcontrôleur particulier.

Les DSP

Vous n'êtes pas sans savoir que DSP signifie *Digital Signal Processor*. Ce processeur est en fait une sorte de microcontrôleur spécialisé. La différence majeure entre un microcontrôleur et un DSP est que ce dernier a été optimisé pour les opérations mathématiques nécessaires pour le traitement de signaux numériques. Un DSP dispose ainsi de structures de registres spéciales et de mécanismes lui permettant d'exécuter le plus rapidement possible une FFT (Transformée de Fourier Rapide), un traitement mathématique pour l'analyse de fréquences dans un signal), et le plus efficacement possible le traitement de grosses quantités de données.
Aujourd'hui, la distinction entre les DSP et les microcontrôleurs devient de plus en plus vague, il suffit de s'intéresser d'un peu plus près à la série dsPIC de Microchip, dont les microcontrôleurs ont été dotés de certaines des fonctionnalités des DSP (nouvelles instructions donc), pour s'en convaincre.

Les FPGA

Une FPGA (*Field Programmable Gate Array*) est, comme le donne à penser son nom, une matrice de portes logiques programmables prise dans un réseau de connexions internes programmables lui (cf. **figure 2**). Un vrai mille-pattes. Il est possible de programmer une FPGA pour lui faire exécuter toutes sortes de traitements logiques, des fonctions de portes logiques de base (ET, OU, NON-OU, NON-ET, etc.) jusqu'à la logique combinatoire la plus complexe telle que fonctions mathématiques et décodeurs. Il est même possible de recréer un microprocesseur complet (un 8051 par exemple) dans une FPGA (ceci à condition que la FPGA ait suffisamment de portes (*gates*) bien entendu).
Les portes sont constituées de CLB prises dans une matrice. Chaque CLB comporte une ou plusieurs LUT

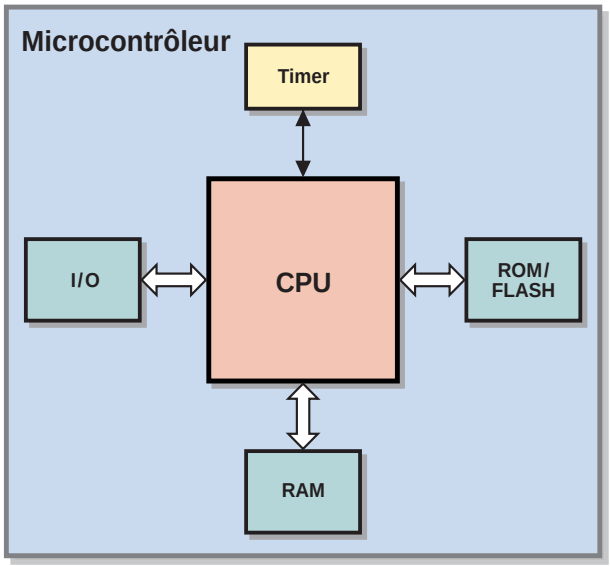


Figure 1. Structure interne d'un microcontrôleur.

KITS DE DÉVELOPPEMENT

Microcontrôleurs	Fabricant / site Web	Type de composant fourni	Interface de programmation	Entrées/Sorties	LCD	Alimentation	Prix de base
PICDEM 2 Plus	Microchip / www.microchip.com	PIC18F452 et PIC16F877	ICD	RS-232	2 16	adaptateur 9 V CC standard	€ 85
PICDEM 4	Microchip / www.microchip.com	PIC12F1320 et PIC16F627A	ICD	RS-232, PIC16LF72 I/O-expander	2x16	adaptateur 9 V CC standard, pile 9 V	€ 110
DS89C450-K00	Maxim/Dallas Semiconductor / www.maxim-ic.com	DS89C450	RS-232	2 x RS-232, 4 x E/S 8 bits	–	adaptateur 6 à 9 V CC	€ 90
Z8 Encore! XP 4K Series Development Kit	Zilog / www.zilog.com	Z8F0XXA (à choisir)	USB ou série	RS-232, IrDA, embase pour entrée ADC (8 canaux max.)	–	adaptateur 5 V fourni	€ 50
Z8 Encore! MCU Development Kit	Zilog / www.zilog.com	Z8FXX (à choisir)	USB ou série	RS-232, IrDA, embase pour entrée ADC (8 canaux max.)	–	adaptateur 5 V fourni	€ 50
SK-XC866 Starter Kit	Infineon / www.infineon.com	XC866	JTAG	RS-232, CAN, LIN, JTAG, embase pour commande de moteur et carte SBC	–	adaptateur 8 à 18 V	€ 150
eCOG1 Development Kit	Cyan Technology Ltd / www.cyantechnology.com	eCOG1	Port Parallèle	Fast Ethernet, Interface de débogage parallèle, 2 x RS-232, I²C, SPI, IrDA	2x16	adaptateur fourni	€ 300
AVR STK500	Atmel/ www.atmel.com	AT90S8515-8PC	RS-232	2x RS-232, embases pour ports d'E/S de tout AVR	–	adaptateur 10 à 15 V	€ 130
Platine d'expérimentation R8C	Elektor / Glyn / www.elektor.fr; www.glyn.de	Renesas R8C-13 R5F21134FP#U0	RS-232	USB, 2 x port sériel, embase pour connexion LCD	–	via USB ou adaptateur	€ 90
DSP							
dsPICDEM 2	Microchip / www.microchip.com	dsPIC30F4011	ICD	RS-232, CAN, embases pour ports d'E/S	2x16	adaptateur 9 V CC standard	€ 85
Explorer 16	Microchip / www.microchip.com	PIC24FJ128GA010 et dsPIC33F128GP710DSC	ICD, JTAG et PICkit 2	USB, RS-232, JTAG, embases pour ports d'E/S, PICTail Plus	2x16	adaptateur 9 à 15 V CC	€ 110
MAXQ2000-KIT	Maxim/ Dallas semiconductor/ www.maxim-ic.com	MAXQ2000	JTAG	RS-232, JTAG, 1-Wire interface, embases pour tous les ports d'E/S	4 1/2	Via interface JTAG, 5 V et adaptateur 6 à 9 V CC	€ 75
FPGA							
High Volume Starter Kit Bundle (comprend Spartan-3 Starter Kit et CPLD Design Kit)	Xilinx/www.xilinx.com	XC3S200-4FT256C (FPGA), XC9572XL-10VQ44C et XC2C256-7TQ144 Coolrunner-II (CPLD's)	JTAG	RS-232, JTAG, PS/2 port souris/clavier, 3 x embase 40 broches	–	fourni	€ 100
MAX II Development Kit	Altera / www.altera.com	MAX II EPM1270F256C5 (CPLD)	JTAG via ByteBlaster	USB, PCI, JTAG, Altera Expansion Prototype Headers	2x16	via USB- ou bus PCI	€ 150
ADDS-21261/Cyclone Evaluation Kit	Altera / www.altera.com	Cyclone EP1C3 (FPGA) et ADSP-21261 (SHARC DSP-chip)	JTAG pour FPGA, USB pour Visual DSP++	USB, RS-232, JTAG, embase d'extension	-	via adaptateur	€ 200
EasyFPGA's EZ1KUSB Development Kit	EasyFPGA / www.easyfpga.com	Altera ACEX EP1K50TC144-3	USB ou JTAG	USB, JTAG, 58 E/S	-	via USB ou adaptateur 6 V CC fourni	€ 190
Morph-IC	Morph-IC / www.morph-ic.com	Altera ACEX EP1K10TC100-3	USB	USB, 2x20-pens header	-	via USB ou adaptateur 5 V CC fourni	€ 90

Convient aux	Manuel	Logiciels	Nécessaires pour la programmation	Divers
PIC16XXXX et PIC18XXXX à 18, 28, et 40 broches	Sur CD-ROM	MPLAB IDE, MPASM, MPLAB C18	PRO MATE II, MPLAB PM3, PIC-START PLUS ou MPLAB ICD 2	Potentiomètre 5 kW, capteur de température TC74, piézo-résonateur
PIC16XXXX et PIC18XXXX à 8, 14, et 18 broches	Sur CD-ROM	MPLAB IDE	PRO MATE II, MPLAB PM3, PIC-START PLUS ou MPLAB ICD 2	Technologie NanoWatt / circuit supercondensateur, 4 potentiomètres 5 kW, place pour transceiver LIN, place pour pilote de moteur
DS89C430, DS89C440, DS89C450, DS5000	Sur CD-ROM	Microcontroller Tool Kit (MTK)	Logiciel + PC suffisent (MTK)	64 Koctets de mémoire Flash, 128 Koctets de SRAM
Pas de substitution possible	Sur CD-ROM	ZDS II Integrated Development Environment, compilateur ANSI C	Câble sériel ou USB Smart fourni	256 bits à 1 Koctet de RAM, 1 à 4 Koctet de mémoire Flash, 2 timer 16 bits, comparateur. Optionnel : CAN à 8 canaux, résolution 10 bits, capteur de température
Pas de substitution possible	Sur CD-ROM	ZDS II Integrated Development Environment, compilateur ANSI C	Câble sériel ou USB Smart fourni	1 à 64 Koctets de Flash/ROM, 256 bits à 4 Koctets de RAM, jusqu'à 60 E/S, jusqu'à 24 interruptions, jusqu'à 4 timer 16 bits, CAN jusqu'à 12 canaux 10-bits. Optionnel : DMA, SPI et I^2C.
Pas de substitution possible	Sur CD-ROM	Versions d'évaluation de Keil uVision et Ulink ou Hitex debugger Tantino-Eco, DAvE	Tantino USB (fourni)	Compatible avec le 8051, génération MLI, CAN 10 bits, 3 timer 16 bits, 27 E/S à usage général, 768 octets de RAM, 16 Koctets de Flash, potentiomètre
eCOG1	Sur CD-ROM	CyanIDE avec compilateur ANSI C, simulateur, débogueur et Configuration Tool	Logiciel + PC suffisent	2 Moctets de SDRAM 16 bits, résonateur piézo, CAN 12 bits, capteur de température
AVR (Attiny, AT90S, ATmega) à 8, 20, 28 et 40 broches	Sur CD-ROM	AVR Studio	Logiciel + PC suffisent	2 Mbits de Flash pour les données
Pas de substitution possible	Dans Elektor / sur site Web	KD30, NC30, HEW, Flash Development Toolkit	Logiciel + PC suffisent	Timer 8 bits, CAN 12 canaux 10 bits, 5 interruptions externes et 11 interruptions internes, 4 Koctets Flash, potentiomètre 10 kW
dsPIC30FXXXX à 18, 28 et 40 broches	Sur CD-ROM	MPLAB IDE	MPLAB ICD 2	potentiomètre, capteur de température, CAN à 9 canaux 10 bits, SPI
Familles PIC24 en dsPIC33	Sur CD-ROM	MPLAB IDE	MPLAB ICD 2	capteur de température TC1047A, potentiomètre 10 kW, 256 Kbits d'EEPROM
Pas de substitution possible		MAX-IDE	Logiciel + PC suffisent	CAN/CNA MAX1407, potentiomètre, carte interface JTAG, carte LCD
Spartan-3, Coolrunner-II, XC9500XL	Sur papier et CD-ROM	Versions d'évaluation de Xilinx ISE & EDK	Câble JTAG3 fourni	Port d'affichage VGA 3 bits, 8 couleurs, 1 Moctets de SRAM
Pas de substitution possible	Sur CD-ROM	Edition Web Quartus II	Câble de téléchargement parallèle ByteBlaster II	capteur de température, potentiomètre 128 Koctets de SRAM, wattmètre embarqué
Pas de substitution possible	Sur CD-ROM	Version d'évaluation Visual DSP++, Quartus II Version Web	Logiciel + PC suffisent	64 Moctets SDRAM, 64 Moctets mémoire EE, 4 Moctets Flash
Altera ACEX EP1K10TC144-3, ACEX EP1K30TC144-3, ACEX EP1K50TC144-3	Sur CD-ROM	Edition Web de Quartus II, Pilotes USB	Logiciel + PC suffisent	
Pas de substitution possible	Sur CD-ROM	Pilotes USB, programme de chargement FPGA, DLL Windows (utilisable avec Visual C++, Visual Basic, Borland Delphi), Quartus II Software Starter Suite	Logiciel + PC suffisent	EEPROM 93C56 embarquée

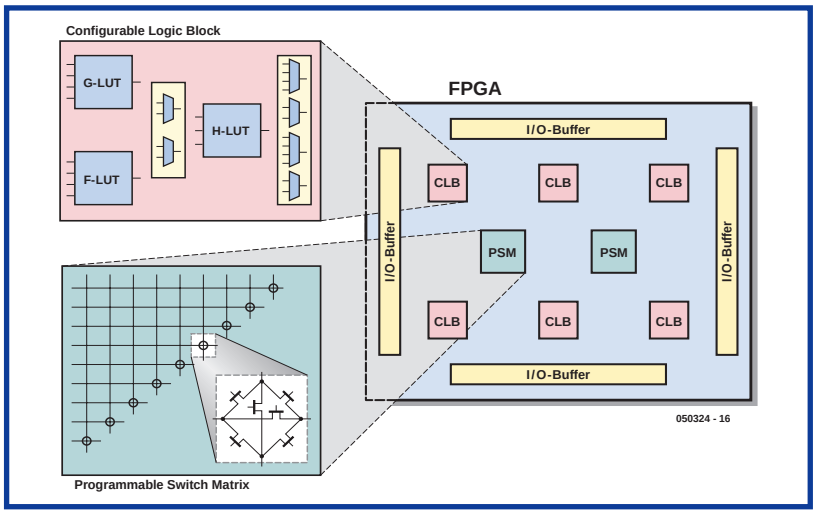


Figure 2. Architecture d'une FPGA.

(*Look-Up Table*), quelques multiplexeurs et, éventuellement, l'un ou l'autre flip-flop. Les CLB se chargent des fonctions logiques. Par une interconnexion judicieuse des commutateurs programmables, on réalise la fonction requise. Ces opérations d'interconnexion sont, en règle générale, l'affaire du programme de développement de sorte qu'il ne nous reste pas grand chose à faire. Les FPGA sont un développement des CPLD (*Complex Programmable Logic Device*). De par leur architecture interne, une FPGA a une flexibilité de développement bien supérieure à celle d'une CPLD.

Le bon microcontrôleur pour tout un chacun

Florian Schäffer

À première vue, faire ses premiers pas dans le monde des microcontrôleurs semble un jeu d'enfant. Une ribambelle de kits d'évaluation essaient d'allécher un acheteur potentiel par tous les moyens mais à y regarder de plus près, ils semblent pratiquement tous capables de la même chose. Une platine dotée d'un microprocesseur, 8 entrées/sorties au minimum, un port RS-232 pour la communication avec le PC et un affichage LCD pour la visualisation de texte, voici en gros le standard. On peut donc difficilement se tromper dans le choix du matériel. C'est alors que la question cruciale se pose : lequel choisir ?

Si l'on fait abstraction des différences de prix, il n'y a plus guère de facteurs jouant un rôle important dans le choix d'un kit de développement. Si vous êtes un débutant dans le monde des micros et que vous êtes sur le point d'acheter votre premier kit, il n'est pas mauvais de réfléchir à ce que vous voulez en faire et dans quelle mesure il est possible de le doter d'extensions. Votre seul but est de concevoir une application de commande concrète dont vous connaissez le nombre d'entrées et de sorties ? Voulez-vous vous uniquement tâter à ce monde envoûtant et vous essayer à un environnement à base de microcontrôleur et arriver, à l'aide de quelques lignes de programme, à faire clignoter quelques LED, faire réagir votre montage à des manipulations de boutons-poussoirs voire se manifester l'affichage LCD ? Ou encore, avez-vous besoin d'une fonction spéciale, telle qu'un port sériel, une interface USB ou un port I2C pour l'échange de données avec d'autres appareils ?

En fonction du cahier des charges que vous aurez établi en réponse à ces questions, il vous faudra examiner les modules d'un peu plus près. Il est délicat de ne se fier qu'aux fiches de caractéristiques succinctes des fabricants, mieux vaut faire un tour sur différents Forums Internet pour y lire les réactions d'utilisateurs. Vous apprendrez ainsi qu'une carte donnée possède bien une interface RS-232 mais que dans la pratique elle est à peine utilisable vu son tampon trop petit de 8 octets seulement et son processeur bien trop lent pour recevoir et traiter, à une vitesse de transfert de 9 600 bauds ou plus, plus de 4 caractères.

Nous en arrivons ainsi au critère suivant : set d'instructions et vitesse du processeur. Il n'est pas nécessaire, dans le cas d'un automate de chauffage central, d'avoir une CPU très rapide. Si la chaudière ne se met en route que quelques secondes plus tard, cela n'a pas de conséquences catastrophiques. Il en va tout autrement s'il s'agit, par multiplexage, de piloter quelques milliers de LED. Si la CPU n'est pas assez rapide, les LED clignotent n'importe comment. Si vous voulez que votre programme puisse réagir rapidement à des signaux externes, de confirmation par action sur une touche ou à des données transmises par une interface, il est « sympa » que la CPU soit à l'aide avec le traitement d'interruptions.

N'oubliez pas de tenir compte de la diversité des solutions de programmation. Avez-vous besoin, pour la programmation du processeur, du matériel additionnel ou est-il possible de le (re)programmer dans le système (ISP = In System Programming) et quel système préférez-vous ? ,

Quelles sont vos ambitions de bricoleur ? La plupart des kits de développement ne comportent qu'une platine. L'affichage LCD y est relié par un câble plat, l'alimentation se fait par le biais d'un adaptateur secteur classique dont la tension est régulée à bord à l'aide d'un 780X. Il arrive, sur les kits les plus simples, qu'il vous faille fournir vous-même la tension d'alimentation régulée.

Vous recherchez peut-être un automate se chargeant de tâches « domestiques ». Un module pour le système de rails DIN, utilisé dans les installations électriques, peut s'avérer très pratique. Pour pouvoir réaliser vos plan en un système réel, vous aurez peut-être besoin d'autres pièces à fabriquer vous-même ou que vous pourrez éventuellement trouver toutes faites, telles qu'un pilote de moteur pas à pas pour votre nouveau robot ou une carte à relais pour la commutation de charges importantes.

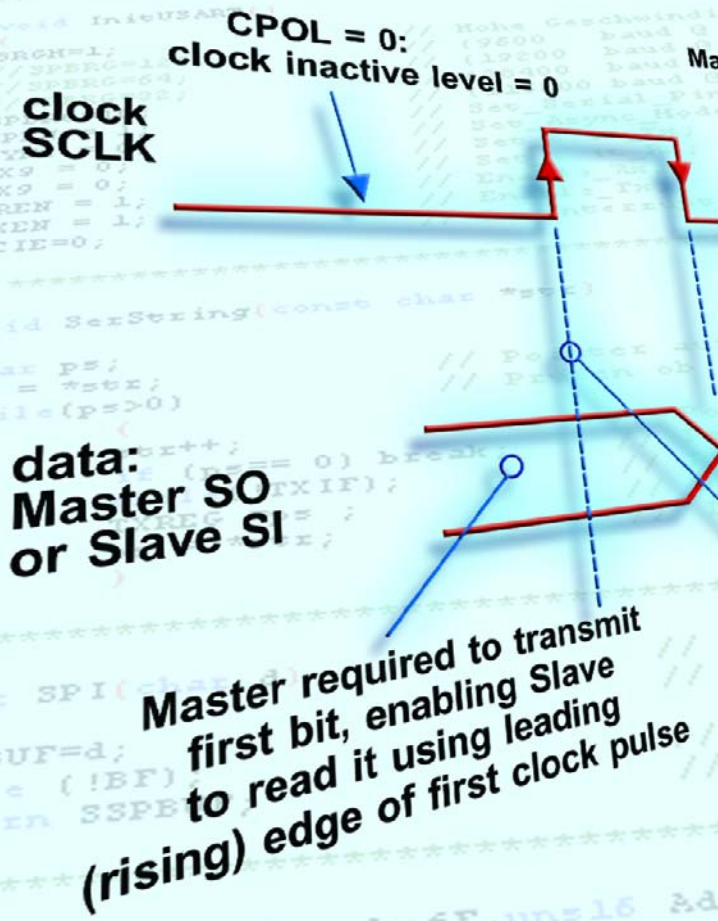
Une fois que vous avez fixé votre choix quant au matériel à utiliser, il ne faudra pas perdre de vue l'aspect logiciel. Il existe, pour la plupart des processeurs, des environnements de développement. C'est alors que se posent les questions du prix et du langage de programmation. La maîtrise de l'assembleur n'est pas donnée à tout le monde. C et BASIC sont, pour le débutant, plus faciles à manipuler, encore que les spécialistes de l'assembleur aient un sourire moqueur lorsqu'ils entendent le mot BASIC. Le code machine chargé dans le processeur n'est guère moins bon que le code écrit à la main. Comme il est probable qu'il vous faudra demander, lors de vos débuts, de l'aide à d'autres utilisateurs, il est bon d'essayer de trouver un Forum adéquat et de voir quels sont les thèmes abordés. Google devrait vous permettre de trouver des Forums consacrés à un type de contrôleur donné.

Il arrive que certains kits de développement utilisent, en combinaison avec le dit kit, des langages de programmation peu utilisés, un environnement de développement en C par exemple. Si vous vous posez des questions alors, la réponse se fera attendre vu que d'autres utilisateurs utilisent bien plus le BASIC et l'assembleur.

Vous trouverez toujours, sur un Forum, des personnes prêtes à vous aider. Elles ont bien souvent déjà répondu à la majorité des questions.

Liaisons série loca

Prof. Dr. Bernd vom Berg &
Peter Groppe Dipl.-Ing.



Les liaisons série synchrones entre un microcontrôleur et des composants périphériques externes ont gagné en popularité au cours de ces dernières années, d'autant plus que les débits dépassent entre-temps 10 Mbits/s.

Un grand nombre de puces aux fonctions les plus diverses proposées actuellement par Analog Devices, Atmel, Burr-Brown, Maxim/Dallas, Microchip et d'autres entreprises peuvent être commandées par un bus local. Cet article exhaustif décrit les bases de la transmission série des données en mode synchrone.

Quelques points fondamentaux

L'utilisateur désireux de raccorder des composants périphériques externes à un microcontrôleur se trouve confronté à une des deux possibilités suivantes :

- Le microcontrôleur dispose des broches vers l'extérieur (plus ou moins) indépendantes requises par les 3 systèmes de bus (données, adressage et commande) pour raccorder des composants externes. Ce type de microcontrôleur ouvert permet de raccorder en parallèle par les 3 systèmes de bus une multitude de composants périphériques externes.
- Le microcontrôleur est si petit est si compact que les 3 systèmes de bus nécessaires ne servent qu'à communiquer avec les unités périphériques intégrées (*on-chip*) mais pas avec l'extérieur. Les broches externes de ce microcontrôleur « fermé » ne sont donc que les E/S des modules intégrés. Il est impossible d'ajouter des extensions externes commandées en parallèle.

N'oublions pas non plus un autre point important : la distance entre le microcontrôleur et les unités externes.

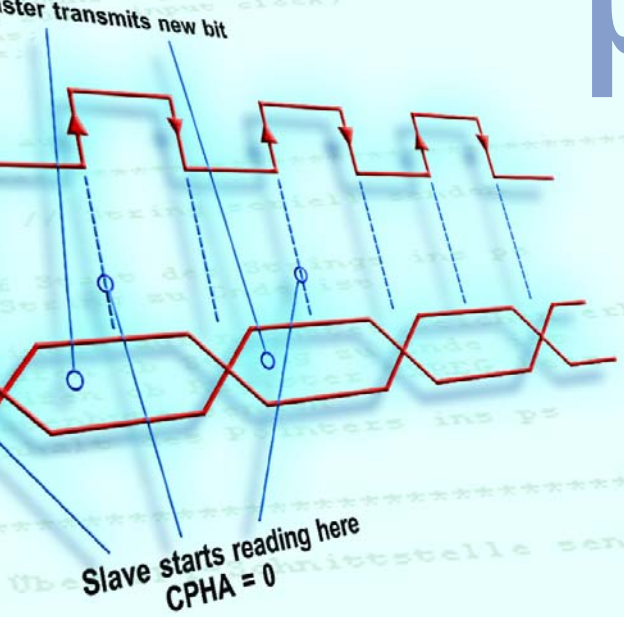
- de quelques centimètres à quelques dizaines de centimètres quand les composants sont réunis sur une carte,
- quelques mètres quand les composants se trouvent dans un appareil, une armoire ou un tableau de commande ou
- de quelques dizaines de mètres à quelques kilomètres quand les composants sont dispersés dans une halle ou un site de fabrication, dans un train ou dans un navire porte-conteneurs.

Un bus parallèle avec ses 10 à 20 conducteurs est exclu pour diverses raisons : distances infranchissables, vulnérabilité aux parasites, difficultés de réalisation, coûts prohibitifs, etc.

Transmission sérielle des données

Ces dernières années ont vu le développement d'un type de liaison « différent » dont la popularité croît toujours

les synchronisées par horloge



plus et toujours plus rapidement : il s'agit des transferts sériels point à point en mode synchrone/asynchrone et des systèmes de bus sériels synchrones/asynchrones. L'échange de données entre participants ne s'effectue plus parallèlement par octet, mais par une série de bits : les bits d'un octet sont transmis successivement.

Cette méthode élimine bien entendu un grand nombre de lignes entre les composants. L'utilisation de convertisseurs de niveau et de commandes de ligne permet d'atteindre des distances toujours plus considérables. Il suffit en fin de compte de 2 à 4 lignes normales d'E/S numérique pour réaliser un transfert de données bidirectionnel. Celui-ci peut être réalisé simplement avec un microcontrôleur ouvert ou fermé, pour autant que des E/S numériques soient encore disponibles.

Seule la vitesse de transmission de cette méthode laisse à désirer. On peut affirmer sans risque de se tromper qu'il est plus rapide de transférer 8 bits en parallèle qu'en série. Toutefois, les systèmes de bus série sont devenus si rapides entre-temps que ce problème n'en est plus vraiment un. Les disques durs modernes, par exemple, sont rapides, plus même qu'un ancien bus parallèle 16 bits.

Synchronisation du transfert des données

La synchronisation entre l'émetteur et le récepteur (**figure 1**) constitue un problème fondamental du transfert synchrone des données. L'émetteur envoie successivement les bits à une certaine vitesse par la ligne de transmission. Comment le récepteur sait-il qu'il doit lire un nouveau bit ? Il ne faut sauter aucun bit, mais ne jamais non plus lire 2 fois ou plus le même bit. Le récepteur a donc besoin de savoir exactement quand un bit correct se trouve à l'entrée et quand le lire.

La solution la plus simple consiste à transmettre des informations supplémentaires (signaux supplémentaires) de l'émetteur au récepteur indiquant sans ambiguïté quand un bit peut être lu à l'entrée du récepteur. Ce signal n'est autre qu'un signal d'horloge envoyé de l'émetteur au récepteur par une ligne supplémentaire.
Le récepteur pourrait réagir comme suit :

la présence d'un flanc montant sur la ligne d'horloge ou de synchronisation signifie toujours qu'un bit correct se trouve à mon entrée et que je peux le lire (niveau haut ou bas). Si le flanc de l'impulsion de synchronisation est descendant, j'attends que l'émetteur ait fini d'envoyer un nouveau bit par la ligne.

Les données reçues en série sont placées rythmiquement à la queue leu leu dans un registre à décalage de largeur adéquate. Une fois transférés, tous les bits sont éva-

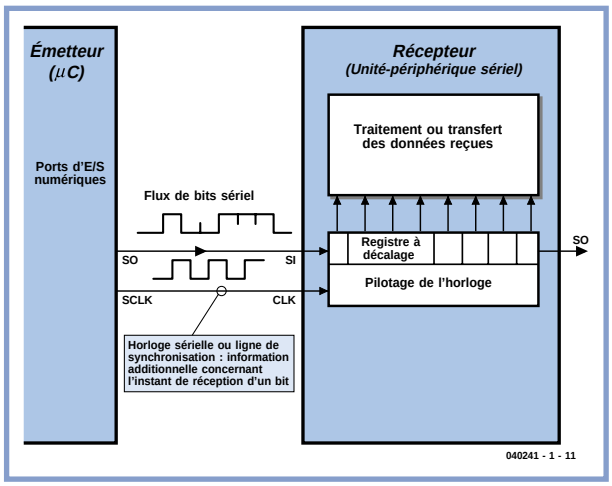


Figure 1.
Synchronisation du transfert des données entre l'émetteur et le récepteur.

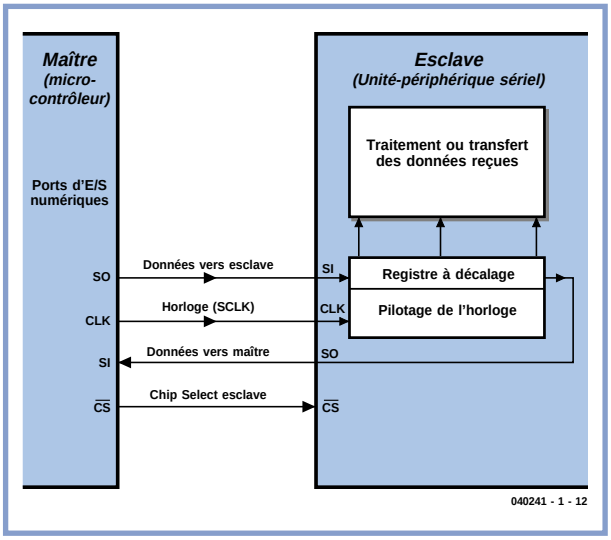


Figure 2. **Liaisons de la transmission série des données en mode synchrone.**

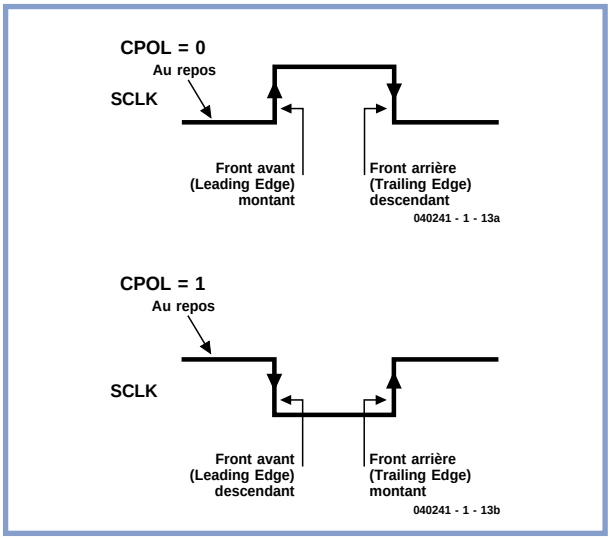


Figure 3. **Détermination du front avant et du front arrière d'une impulsion d'horloge.**

lués et traités par les étages suivants du récepteur. Il faut donc au moins une ligne de données et une ligne d'horloge pour réaliser cette transmission série en mode synchrone basée sur un signal d'horloge.
Une autre méthode de synchronisation entre l'émetteur et le récepteur est utilisée pour le transfert série des données en mode asynchrone. Ses représentants les plus connus sont les interfaces COMx-UART des PC et le bus CAN. Ce sujet sort toutefois du cadre de cet article.

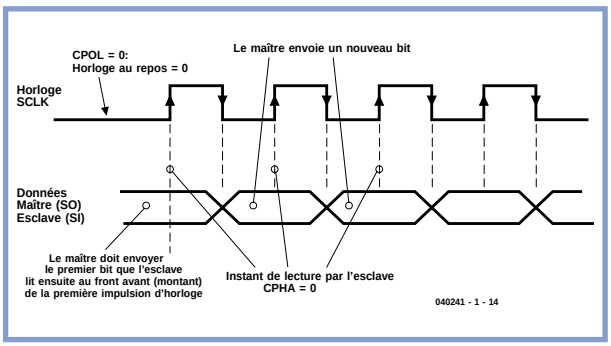


Figure 4. **Transmission série synchrone des données en mode 0.**

Transfert série des données en mode synchrone

La **figure 2** illustre l'interface généralement utilisée entre un composant maître et un composant esclave pour le transfert série de données point à point en mode synchrone. Les 2 stations peuvent procéder à un échange réciproque de données par 2 lignes séparées. Toutefois, une seule des 2 stations engendre le signal d'horloge pour les 2 directions de transfert. Cette station est donc le maître (il s'agit en général du microcontrôleur). Le signal d'horloge fourni par le maître détermine la vitesse (bits par seconde) du transfert. Elle ne doit bien entendu pas dépasser la vitesse de transfert la plus élevée de l'esclave, qui est généralement une unité périphérique.
Points saillants du processus :

- La ligne d'horloge (S)CLK (*Serial Clock*) permet de transmettre le signal d'horloge central du maître au composant esclave. Le registre à décalage de l'esclave est cadencé par ce signal.
- Les bits transmis par la connexion SO (*Serial Out*) du maître à la connexion SI (*Serial In*) de l'esclave sont placés au rythme de l'horloge dans le registre à décalage. Ils en sont extraits à la fin du transfert de données et traités par l'esclave (commandes et données).
- L'esclave, lorsqu'il doit répondre au maître, remplit son propre registre à décalage avec les bits de réponse et les envoie par la sortie SO à l'entrée SI du maître au rythme de l'horloge de celui-ci. Si l'esclave n'a pas de données particulières à envoyer au maître, il vide simplement son registre à décalage des bits (arbitraires) qui s'y trouvent. La transmission des données s'effectue donc en duplex intégral. Le maître envoie des données à l'esclave par SO tout en recevant celles de l'esclave par SI.

- La quatrième ligne $\overline{CS}$ (*Chip Select* pour l'esclave) sert à choisir l'esclave parmi plusieurs unités du système. Nous y reviendrons.

Faut-il préciser que la connexion SO de l'esclave n'existe que si celui-ci doit envoyer des données (et qu'il en est de même pour l'entrée SI du maître) ? Un petit exemple pour fixer les idées.
Une commande d'afficheur LED fonctionnant en série se contente de recevoir des données du maître, de les décoder et de les afficher. L'esclave n'a pas besoin de connexion SO car il ne renvoie aucunes données au maître. L'interface série en mode synchrone ne possède donc que 3 lignes.
Un capteur de température fonctionnant en série a, par contre, besoin d'une connexion SO dans la plupart des cas. Il doit déterminer la température sur demande du maître et la lui envoyer par sa connexion SO. L'interface série comporte alors (au maximum) 4 lignes.

CPOL et CPHA

En examinant d'un peu plus près le déroulement de la transmission, on constate qu'il existe 4 façons distinctes d'échanger les données. Les 2 paramètres CPOL et CPHA caractérisent de façon universelle ces différents modes synchrones de transmission série. Ils spécifient l'état de repos de la ligne d'horloge et l'attribution des flancs d'horloge dans l'échange des données.
Considérons la ligne d'horloge SCLK. À l'état de repos (avant le début du transfert de données), elle peut avoir exactement 2 états, décrits par le paramètre CPOL (*Clock Polarity*).

Tableau 1. Les 4 modes d'une transmission série synchrone du maître à l'esclave.		
CPOL	**CPHA**	**Mode synchrone** (aussi dénommé « mode SPI (Serial Peripheral Interface) »)
0	0	**Mode 0** L'esclave prend en charge les données avec le front avant, donc avec le flanc montant. Le maître produit les données avec le front arrière, donc avec le flanc descendant de l'impulsion précédente.
0	1	**Mode 1** L'esclave prend en charge les données avec le front arrière, donc avec le flanc descendant. Le maître produit les données avec le front avant, donc avec le flanc montant de la même impulsion.
1	0	**Mode 2** L'esclave prend en charge les données avec le front avant, donc avec le flanc descendant. Le maître produit les données avec le front arrière, donc avec le flanc montant de l'impulsion précédente.
1	1	**Mode 3** L'esclave prend en charge les données avec le front arrière, donc avec le flanc montant. Le maître produit les données avec le front avant, donc avec le flanc descendant de la même impulsion.

● CPOL = 0 : l'état de repos de la ligne d'horloge est un niveau bas (0 V). Le transfert série des données en mode synchrone commence donc toujours par un flanc d'horloge montant.

● CPOL = 1 : l'état de repos de la ligne d'horloge est un niveau haut (+5 V). Le transfert série des données en mode synchrone commence donc toujours par un flanc d'horloge descendant.

L'impulsion d'horloge possède –de façon tout à fait générale– un front avant (*Leading Edge*) et un front arrière (*Trailing Edge*) entre lesquels « quelque chose peut se passer ». Ces désignations semblent de prime abord légèrement insolites. Elles occupent toutefois une place prépondérante dans la description de la transmission synchrone des données.

Considérons pour commencer le transfert de données du maître à l'esclave par la ligne SO/SI. Il faut encore préciser certains points pour définir l'instant exact de la transmission :

● À quel flanc d'horloge l'esclave peut-il lire un bit correct à sa broche SI ?

● À quel flanc d'horloge le maître place-t-il à sa broche SO un nouveau bit qui puisse être lu par l'esclave lors du flanc d'horloge suivant ?

Il existe ici aussi 2 possibilités définies par le second paramètre CPHA (*Clock Phase*).
En tenant aussi compte de l'état de CPOL, on détermine exactement l'apparence du front avant et du front arrière de l'impulsion d'horloge (**figure 3**).
Le front avant monte et le front arrière descend lorsque CPOL = 0. CPOL = 1 signifie le contraire. En tenant compte de ce qui précède, CPHA a la signification suivante :

● **CPHA = 0**
L'esclave prend en charge les données à son entrée SI à l'instant du front avant. C'est-à-dire du flanc montant de l'impulsion si CPOL = 0 et de son flanc descendant si CPOL = 1. Il faut que le maître ait placé auparavant ce bit de données à sa sortie SO, c'est-à-dire au moment du front arrière de l'impulsion précédente.

● **CPHA = 1**
L'esclave prend en charge les données à son entrée SI à l'instant du front arrière. C'est-à-dire du flanc descendant de l'impulsion si CPOL = 0 et de son flanc montant si CPOL = 1. Il faut que le maître ait placé auparavant ce bit de données à sa sortie SO, c'est-à-dire au moment du front avant de la même impulsion.
Ces 2 paramètres CPOL et CPHA caractérisent donc les 4 cas possibles de transmission série de données en mode synchrone (**tableau 1**).

Ces 4 modes sont totalement incompatibles entre eux ! Il faut toujours être très attentif au mode de fonctionnement de l'esclave (composant périphérique) avant de programmer le maître (microcontrôleur).

Un exemple et tout deviendra plus clair. Considérons le mode 0 (CPOL = 0, CPHA = 0) de la **figure 4**. Les 3 points cruciaux sont :

1. Le niveau de repos de la ligne d'horloge est bas (0 V).

2. L'esclave reprend les données avec le flanc d'horloge montant. Le bit à lire doit se trouver à ce moment dans un état stable à l'entrée SI de l'esclave.

3. Le maître modifie les données à l'instant du flanc d'horloge descendant, ou tout au moins avant le prochain flanc d'horloge montant (qu'il engendre lui-même).

Cette description formellement exacte du transfert de données vous semble-t-elle trop ésotérique ? Formulons la même chose en termes plus pragmatiques. Pour établir une transmission série de données en mode synchrone, vérifier les points suivants concernant le composant esclave, à partir de la fiche de données si nécessaire :

1. État de repos de la ligne SCLK (niveau bas ou haut) ?

2. Avec quel flanc l'esclave prend-il en charge les bits de la ligne de données (flanc montant ou descendant) ?

3. Avec quel flanc le maître doit-il alors avoir envoyé préalablement les données à la ligne de données (flanc montant ou descendant) ?

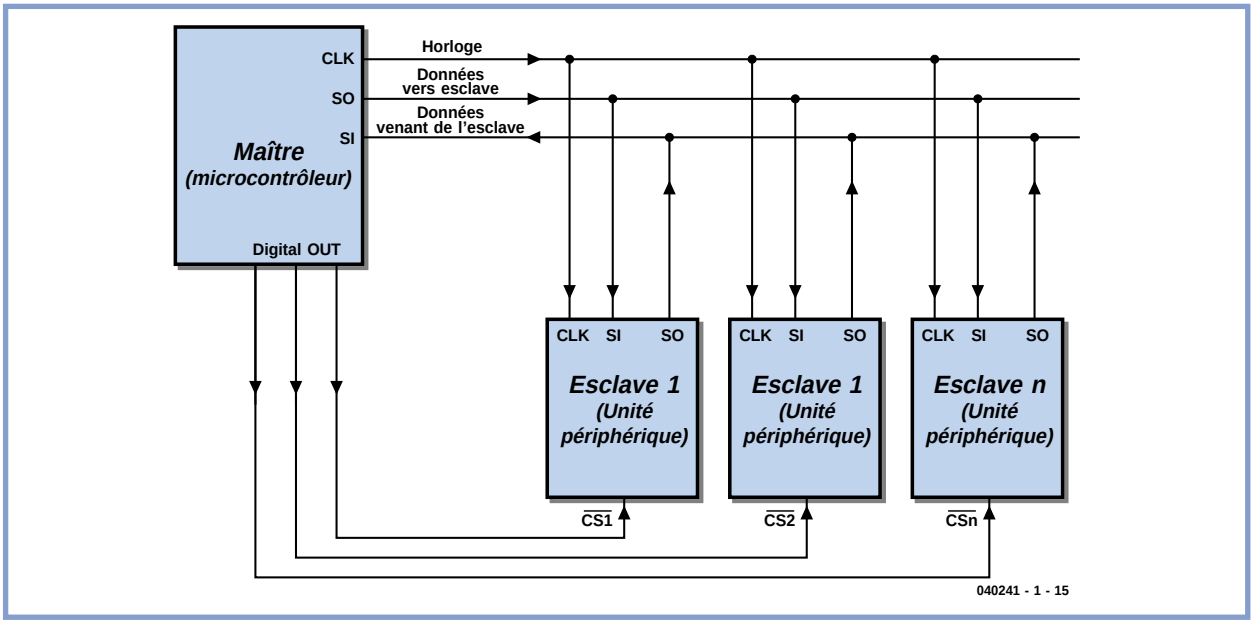


Figure 5.
Extension vers le bus sériel.

La programmation du microcontrôleur devient très simple une fois ces 3 points tirés au clair. Il faut toutefois veiller au point suivant : les désignations CPOL, CPHA, mode SPI, Leading Edge, Trailing Edge ou front avant/arrière se retrouvent très souvent dans la documentation des microcontrôleurs à interface synchrone intégrée et dans celle des composants esclaves. Il est préférable de savoir ce que cela veut dire et comment configurer le logiciel de transmission des données.
Le transfert des données de l'esclave au maître (donc sortie SO de l'esclave et entrée SI du maître) est très semblable. Les bits sont envoyés par l'esclave au rythme de chaque flanc qui suit un flanc de réception. Si le maître envoie un bit à l'esclave avec un flanc montant, l'esclave enverra un bit par la broche SO lors de l'apparition du flanc suivant, donc d'un flanc descendant. Et réciproquement.

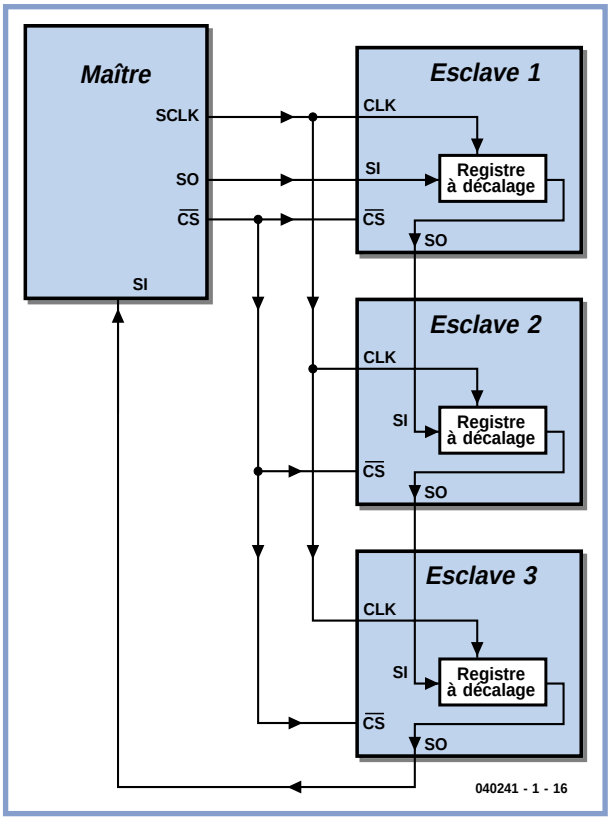


Figure 6.
Mise en cascade d'esclaves.

Bus série en mode synchrone

On peut généraliser cette liaison point à point à un système de bus série possédant plusieurs stations esclaves. Il suffit de raccorder celles-ci en parallèle aux lignes horloge et données (**figure 5**). Il faut toutefois que chaque esclave puisse être adressé individuellement pour recevoir les données qui lui sont destinées.

Le maître doit disposer d'une sortie numérique pour chaque esclave qui permette d'activer celui-ci (Chip Select $\overline{CS}$, généralement active au niveau bas). La transmission série des données ne doit avoir lieu qu'après l'activation de l'esclave. Celui-ci est de nouveau désactivé à la fin de l'échange de données.

Si l'esclave possède une sortie SO, les données sont extraites du registre à décalage interne aussi longtemps que le signal $\overline{CS}$ est actif. En l'absence de sortie SO, les bits restent dans le registre à décalage et sont simplement écrasés par les bits suivants. L'évaluation des bits de données se trouvant dans le registre à décalage ne commence que lorsque $\overline{CS}$ est désactivé.

Les données sont généralement transmises par octet. Le transfert des données vers et de l'esclave peut commencer par *l'extrémité droite* de l'octet, donc par le bit de poids faible (*Least Significant Bit, LSB*) ou par *l'extrémité gauche*, donc par le bit de poids fort (*Most Significant Bit, MSB*).

L'horloge du maître détermine la vitesse de transmission. Elle ne doit toutefois pas dépasser la vitesse d'horloge la plus élevée de l'esclave. Il peut s'avérer nécessaire d'allonger la période d'horloge (durée du cycle) au moyen de boucles de délai. Une horloge trop lente ne constitue en général pas un problème. La transmission des données est

Tableau 2. Avantages et inconvénients des esclaves à commande synchrone	
Système de bus standard	
Avantages :	Esclaves individuellement adressables. Convient aussi aux esclaves dépourvus de sortie SO. Possibilité de transmettre les données aux esclaves à la vitesse maximale sans surcharge (overhead) de transfert des données.
Inconvénients :	Le maître doit être équipé d'une ligne E/S numérique supplémentaire par esclave (ligne $\overline{CS}$) ou il faut recourir à une autre méthode pour gérer la logique $\overline{CS}$.
Système de bus en cascade	
Avantages :	Une seule ligne $\overline{CS}$ suffit pour tous les esclaves. Tous les esclaves adressables simultanément, ce qui veut dire que les données transmises sont évaluées simultanément dès que la ligne $\overline{CS}$ est désactivée. Matériel peu compliqué. Utilisable lorsque la vitesse du transfert de données n'est pas le critère déterminant.
Inconvénients :	Réservé aux esclaves possédant une sortie SO. Surcharge (overhead) considérable lorsque les données ne doivent être envoyées qu'à un seul esclave. Débit déterminé par l'esclave le plus lent.

commandée par les flancs et le temps séparant ceux-ci n'a généralement aucune influence sur l'esclave pour autant que le nombre correct de flancs (bits) soit transmis.

Esclaves en cascade

Jadis, lorsqu'il existait encore de nombreuses prairies sauvages émaillées de fleurs, les enfants, petits et grands, tressaient de jolies guirlandes de pâquerettes. L'envoi des données en cascade aux esclaves selon la **figure 6** se nomme *connexion en guirlande*, en anglais *Daisy-Chain Configuration*.

« En cascade » signifie que les mémoires de tous les esclaves reçoivent les données en série. Les esclaves sont tous raccordés à une ligne $\overline{CS}$ qui en permet simultanément l'accès. Chaque sortie des données SO est aussi reliée à l'entrée suivante SI. Les bits sont donc « poussés » successivement d'un esclave au suivant. La dernière sortie SO est reliée à l'entrée SI du maître si celui-ci doit recevoir des données. Sinon, cette liaison reste ouverte et les données se volatilisent à la dernière sortie SO. Le tout fonctionne comme un registre à décalage géant composé de plusieurs petits registres (2 ou plus).

Supposons que chacun des 3 esclaves possède un registre à décalage de 8 bits. Pour que des données parviennent au troisième esclave, le maître doit

1. rendre la ligne $\overline{CS}$ active
2. envoyer en cadence les 8 bits destinés à l'esclave 3
3. envoyer encore 16 bits pour que les 8 bits destinés à l'esclave 3 parviennent à ce dernier.
4. désactiver $\overline{CS}$

On se trouve déjà confronté à un petit problème : les nouveaux bits sont destinés à l'esclave 3. Les esclaves 1 et 2 ne doivent pas recevoir de (nouvelles) données. Pour éviter que ces données soient évaluées, on recourt à un octet de données NOP (*No OPeration*). Lorsque cette configuration particulière d'un octet se trouve dans le registre à décalage de l'esclave, il ne se passera rien lorsque la ligne $\overline{CS}$ sera désactivée : l'état de l'esclave ne subira aucune modification. Dans notre exemple, il faut transférer 24 bits de la façon suivante :

1. Activer la ligne $\overline{CS}$
2. Envoyer en cadence les 8 nouveaux bits utiles pour l'esclave 3
3. Envoyer ensuite 2 octets NOP destinés aux esclaves 1 et 2.
4. Désactiver la ligne $\overline{CS}$

Comme désiré, seul l'esclave 3 réagit à la désactivation de $\overline{CS}$ à la fin du transfert des données. Elle est sans effet sur les autres esclaves. S'il est par contre nécessaire de faire parvenir simultanément aux 3 esclaves des données à traiter, il suffit d'envoyer l'une après l'autre les configurations binaires requises.
Les esclaves en cascade sont souvent utilisés avec les afficheurs LED de grande taille (multiligne). Chaque esclave servira par exemple de circuit de commande pour une ligne de l'afficheur. Le nombre d'esclaves en cascade est, comme dans le cas d'un bus, théoriquement illimité. En pratique, un transfert de données prend d'autant plus de temps que le nombre d'esclaves est élevé. Le **tableau 2** indique les avantages et les inconvénients de l'interconnexion des puces esclaves.

(040241-1)

Pour en savoir plus

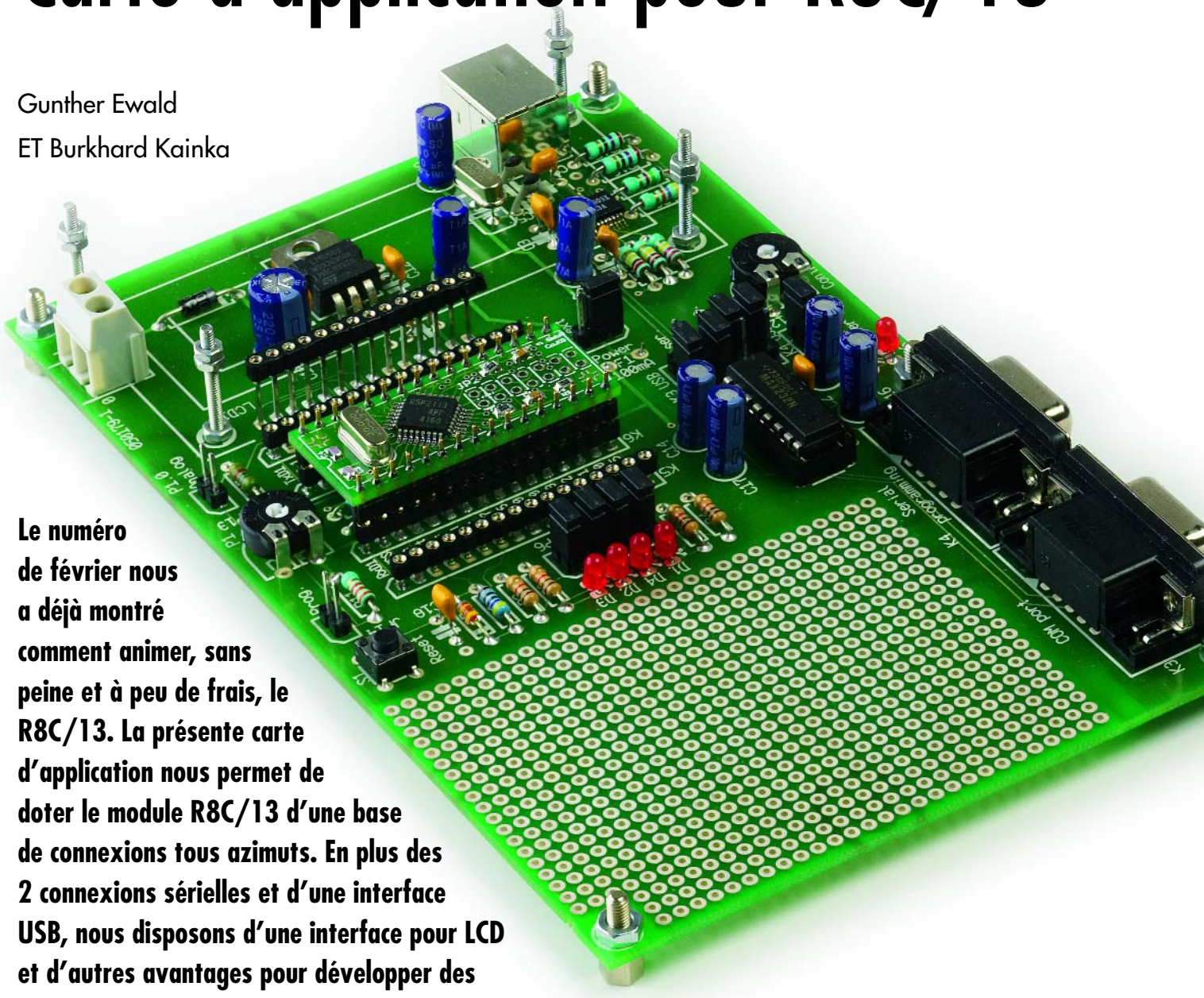
[1] Bernd vom Berg, Peter Groppe, Joachim Klein
Je programme en PASCAL les microcontrôleurs de la famille 8051 (80C537)
Éditions Publitronic/Elektor

[2] PalmTec : www.palmtec.de

«Camp de base» p

Carte d'application pour R8C/13

Gunther Ewald
ET Burkhard Kainka



Le numéro de février nous a déjà montré comment animer, sans peine et à peu de frais, le R8C/13. La présente carte d'application nous permet de doter le module R8C/13 d'une base de connexions tous azimuts. En plus des 2 connexions sérielles et d'une interface USB, nous disposons d'une interface pour LCD et d'autres avantages pour développer des applications personnelles.

La question que nous avons posée pour lancer ce développement est la suivante: de quoi l'humain a-t-il besoin pour tirer profit, avec un investissement aussi réduit que possible, des caractéristiques essentielles du contrôleur? C'est à cette question que répond la platine et son équipement, à savoir:

- 2 connexions d'interface sérielle
- un port USB, avec adaptateur USB/sériel intégré
- une connexion pour afficheur LCD
- 4 LED commutables sur les ports
- 1 potentiomètre sur l'une des nombreuses entrées analogiques
- embase pour bloc secteur et régulateur de tension 5 V
- alimentation, au choix, également par USB
- une touche de réinitialisation
- un cavalier pour l'entrée de MODE.

L'interface USB mérite une mention particulière. En effet, de moins en

our débutants

moins de PC, les portables surtout, disposent d'une interface sérielle. D'un autre côté, la communication avec le contrôleur s'effectue fondamentalement par l'intermédiaire d'interfaces sérielles asynchrones. Il était donc judicieux d'installer d'emblée un adaptateur USB/sériel sur la carte. Cette solution présentait encore sur celle d'un adaptateur USB externe, l'avantage de permettre une alimentation par l'intermédiaire du port USB. Quatre cavaliers en tout définissent quelle interface sérielle du contrôleur accède à l'extérieur par les ports RS-232 ou le port USB. Vous pouvez donc, par exemple, charger des programmes et déboguer par l'interface USB ou utiliser l'interface USB pour la communication sérielle avec le contrôleur et les programmes que vous avez conçus.

Aux ports

Le schéma (**figure 1**) pointe au c?ur de ce que le numéro de décembre a déjà mentionné comme conditions essen-

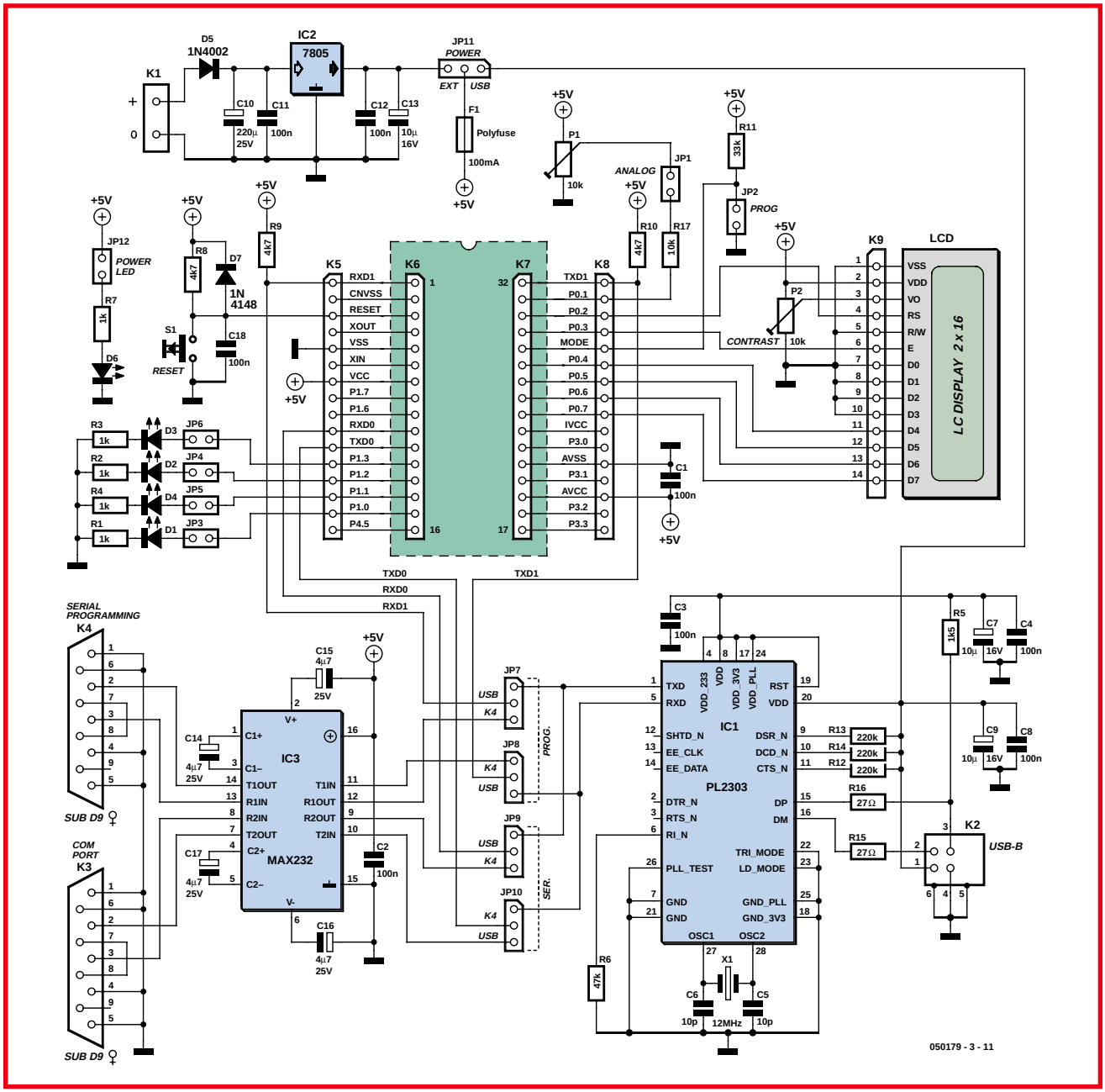


Figure 1. Le circuit de la carte d'application relie la platine enfichée R8C à deux ports sériels, une interface USB et un afficheur à cristaux liquides.

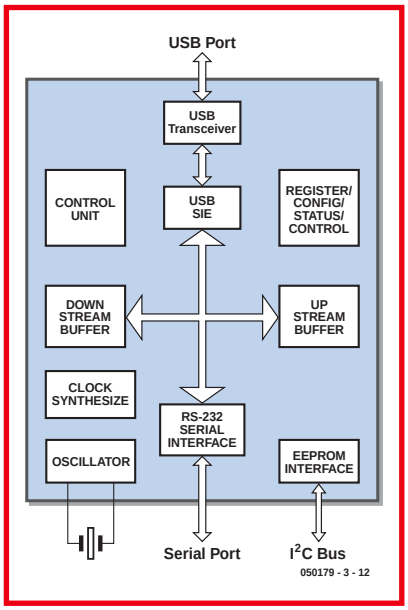


Figure 2. Schéma fonctionnel du convertisseur d'interface Prolific.

tielles pour un système R8C : le cavalier de MODE et la touche de réinitialisation pour commencer, nous avons ensuite besoin de circuits d'attaque d'interface sur l'interface de débogage RXD1/TXD1, qui conduisent ici sur JP7 et JP8, permettant de choisir entre RS-232 et USB. Suivant la position des cavaliers, on utilise le MAX232 avec la connexion K4 ou le contrôleur USB PL2303 et un câble USB.
Le R8C/13 ne dispose cependant pas seulement d'une interface sérielle de débogage UART1, mais également d'une seconde interface sérielle UART0, tout à fait indépendante, avec les lignes TXD0 et RXD0. On sait que le MAX232 est en mesure d'alimenter plus de deux lignes et il se charge des quatre. Nous avons donc encore la connexion sérielle K3 et les cavaliers JP9 et JP10. Nous

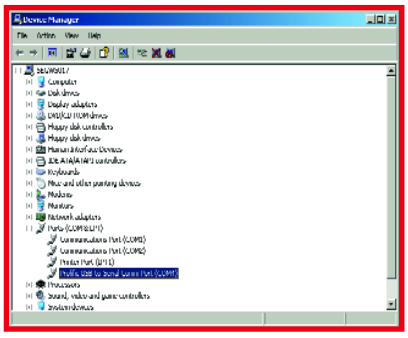


Figure 3. L'interface COM virtuelle dans le gestionnaire de périphériques de Windows XP.

pouvons ainsi choisir entre RS-232 et USB sur l'interface UART0.
Aux ports maintenant: les connexions du contrôleur sont conduites à des embases d'une rangée de contacts et offrent toutes les possibilités. L'utilisation de la plupart des ports a cependant d'abord fait l'objet d'une présélection supplémentaire. Le port de 8 bits P0 a été prévu pour la connexion de l'afficheur LCD, mais il va de soi que vous pouvez également lui confier une autre fonction si le LCD ne vous est pas utile. On peut attaquer un LCD standard alphanumérique en mode 4 bits avec six lignes de port. Les lignes P0_2 à P0_7 du LCD sont ainsi occupées. La ligne P0_0 est déjà utilisée par l'interface de débogage (TXD1) et n'est plus disponible que dans des cas exceptionnels pour des applications personnelles. Il nous reste encore P0_1 et puisque cette ligne peut également être utilisée comme entrée analogique, nous y avons connecté un potentiomètre par l'intermédiaire de JP13 pour y appliquer une tension d'entrée de 0 à 5 V. Une résistance de protection complémentaire, R17, évitera une surcharge au cas où P0_1 serait par mégarde exploité simultanément en port de sortie. Rien ne vous empêche du reste d'utiliser les deux picots de cavalier comme entrée de mesure et si vous souhaitez connecter un capteur qui délivre les deux polarités, vous pouvez en régler le zéro avec le potentiomètre au point qui vous est utile.
Les 4 bits de poids faible du port 1 sont reliés par l'intermédiaire de cavaliers et de résistances talon aux quatre LED. Cette solution est idéale pour les premiers exercices de programmation qui ont été décrits dans le numéro de décembre. Si vous n'avez pas encore essayé les premiers exemples de clignotants, il est temps de vous rattraper. En retirant les cavaliers JP3 à JP6, vous disposez de connexions pour 4 autres entrées analogiques. Rien ne s'oppose d'ailleurs plus au multimètre numérique à quatre voies à affichage LCD. Sept lignes de port indépendantes ont été laissées entièrement libres mais vous trouverez bien à les occuper. Des applications à I_C, SPI, etc. peuvent encore avoir besoin de l'un ou de l'autre port.

Ce qu'il faut de courant

Nous avons deux solutions pour l'alimentation. Suivant la position de JP11, nous utiliserons le connecteur de bloc secteur K1 et le régulateur de tension IC2 ou le port USB qui, on le sait, délivre 5 V. D'ailleurs, si vous fonctionnez sur le port USB, on conçoit facilement que vous renonciez à un bloc secteur externe. Prudence toutefois. Si l'on en croit les spécifications USB, tout port USB en réception devrait être protégé contre les surcharges par des fusibles réarmables (Polyswitch). Des expériences douloureuses montrent qu'il n'en est rien. Un simple court-circuit de la tension d'alimentation de la tension d'alimentation USB provoque un petit nuage de fumée sur la carte-mère du PC et sépare à tout jamais le port de réception du bloc-secteur. Pour éviter les frais, on a manifestement implanté que de petites résistances sur la ligne d'alimentation et elles grillent, tout simplement.
Il était donc prudent d'y remédier et de protéger le PC contre les courts-circuits en prévoyant sur la carte d'application le fusible Polyswitch F1 avec un courant de 100 mA, ce que nous avons fait. Cette précaution rassure et permet d'expérimenter de façon plus détendue.

Interface USB

Venons-en à l'interface USB. Vous connaissez peut-être le PL2303X du Chinois (Taiwan) Prolific (distribué par Glyn). Il s'agit d'un « *USB to Serial RS-232 Bridge Controller* ». Il est passé dans ces colonnes, incorporé et scellé, dans un câble de conversion d'interface décrit dans le n °329 (novembre 2005) d'Elektor. Ce circuit intégré est un périphérique USB « pleine vitesse » pour USB 1.1 (compatible avec les interfaces USB 2.0 les plus récentes). Les données sont donc transmises sur le bus à 12 Mbit/s. La résistance R5 porte la borne D+ du port USB à 3,3 V et signale ainsi au PC la présence d'un périphérique « *fullspeed* ». Le PC charge ensuite le pilote d'interface sérielle virtuelle. On peut ainsi exploiter des interfaces sérielles avec une rapidité de modulation qui peut atteindre 1 228 800 bauds (1,2 Mbauds !). Divers débits standard sont utilisés avec le R8C/13. Lors du flashage ou du débogage, l'interface sérielle virtuelle se comporte comme une vraie interface RS-232.
La **figure 2** représente la structure interne avec interface sérielle d'émission-réception USB et deux tampons de données. Les tampons de données se chargent de leur rapide transfert, sans lacune, avec des blocs de don-

nées compactés dans les différentes trames USB. Il est, de plus, possible de connecter une EEPROM I_C pour des configurations personnalisées mais nous n'en ferons pas usage ici.
Les signaux inversés RXD et TXD sont disponibles du côté sériel du convertisseur et peuvent être reliés directement au R8C. Les entrées du PL2303X sont compatibles avec les tensions de 3,3 V et 5 V. Les sorties TXD, DTR et RTS délivrent 3,3 V ou 5 V, suivant la tension appliquée sur la broche 4. Nous aurons ici 3,3 V du régulateur de tension interne du PL2303. On a donc un niveau de 3,3 V sur l'entrée sérielle du R8C/13 alors que l'on aurait pu s'attendre à y trouver 5 V. C'est pourtant admissible puisque, d'une part, le R8C/13 peut aussi fonctionner sous 3,3 V et que, d'autre part, un signal de 3,3 V suffit sur les entrées RXD du contrôleur en mode 5 V. Les entrées du R8C reconnaissent en effet comme niveau haut des tensions supérieures à 0,2 V_{CC}.

Installation du pilote

Il est possible qu'un jour vous ayez déjà installé un adaptateur USB de Prolific. Dans ce cas, l'interface est prête dès que la connexion au PC est détectée. Dans le cas contraire, ou si vous aviez des problèmes de fonctionnement avec le port USB, installez le pilote le plus récent. Vous le trouverez dans le fichier **050179-3-11.zip** à télécharger depuis notre site à l'adresse www.elektor.fr.
Cette archive contient le fichier „PL-2303 Driver Installer.exe". Lancez-le avant de connecter la platine. Lors de l'installation, les anciennes versions du pilote sont automatiquement effacées du PC. À la prochaine connexion de la platine (ou d'un autre convertisseur d'interface à circuit Prolific), Windows trouvera automatiquement le pilote convenable et le chargera. Votre PC disposera ensuite d'une interface sérielle supplémentaire qui, sinon, est utilisée comme une interface RS-232 normale.
Si vous aviez déjà installé plusieurs convertisseurs d'interface de ce type, Windows attribuerait un numéro de COM élevé à la nouvelle interface virtuelle. Il est recommandé dans ce cas, de renommer l'interface, par exemple en COM2, dans les paramètres avancés de Windows. Généralement, on peut ne pas tenir compte du message qui signale que cette interface est déjà utilisée puisqu'il concerne une autre

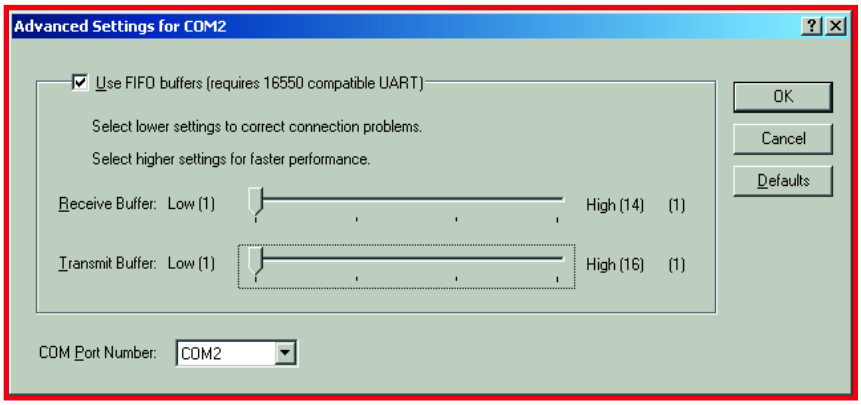


Figure 4. Paramétrage de la taille du tampon et du numéro de port COM.

interface virtuelle qui n'est pas utilisée en même temps. Il suffit en principe de savoir de quelles interfaces sérielles matérielles dispose effectivement le PC. Sur celui que nous utilisons pour les essais, la carte-mère n'offre, par exemple, que COM1, mais nous disposons en complément d'une carte d'interface avec COM3 et COM4. Il est facile de concevoir que nous désignions par COM2 l'interface virtuelle (voir **figure 3**), d'autant que beaucoup de programmes plus anciens ne connaissent que COM1 et COM2. On a également constaté que le programme de flashage FDT avait des problèmes avec les numéros de COM élevés. Il est donc conseillé d'en rester au moins aux ports à un seul chiffre.
Dans certains cas, la fiabilité des communications est mieux garantie quand la taille configurée pour le tampon FIFO de l'interface virtuelle (**figure 4**) est la plus petite. Théoriquement on peut en effet se retrouver avec de grands tampons dans une situation où le logiciel du PC attend une réponse alors que la question est encore dans le tampon. Si vous constatez, par exemple, des erreurs lors du flashage du contrôleur, vous pourrez y remédier en réduisant la taille du tampon.

Platine et composants

Si vous n'êtes pas trop sûr de vous, la platine (**figure 5**) est aussi disponible, en option, avec ses composants soudés. Vous éviterez ainsi le problème d'implantation d'IC1. Le circuit intégré de Prolific n'existe, en effet, qu'en boîtier CMS à 28 contacts (SSOP-28). Si, en

Liste des composants

Résistances :
R1 à R4,R7 = 1 kΩ
R5 = 1kΩ5
R6 = 47 kΩ
R8 à R10 = 4kΩ7
R11 = 33 kΩ
R12 à R14 = 220 kΩ
R15,R16 = 27 Ω
R17 = 10 kΩ
P1,P2 = 10 kΩ

Condensateurs :
C1 à C4,C8,C11,C12,C18 = 100 nF
C5,C6 = 10 pF
C7,C9,C13 = 10 µF/16 V radial
C10 = 220 µF/25 V
C14 à C17 = 4µF7/25 V radial

Semi-conducteurs :
D1 à D4,D6 = LED
D5 = 1N4002
D7 = 1N4148
IC1 = PL2303X (Prolific)
IC2 = 7805
IC3 = MAX232

Divers :
JP3 à JP6,JP12 = cavalier
JP7 à JP11 = embase autosécable à 1 rangée de 3 contacts
K1 = bornier encartable à 2 contacts au pas de 5 mm
K2 = embase USB de type B encartable
K3,K4 = embase sub-D à 9 contacts en équerre encartable
K5 à K8 = embase à 1 rangée de 16 `contacts
K9 = module LCD 2x16 caractères)
S1 = bouton-poussoir unipolaire à contact travail tel que, par exemple TS695
F1 = polyfuse 100 mA
X2 = quartz 12 MHz

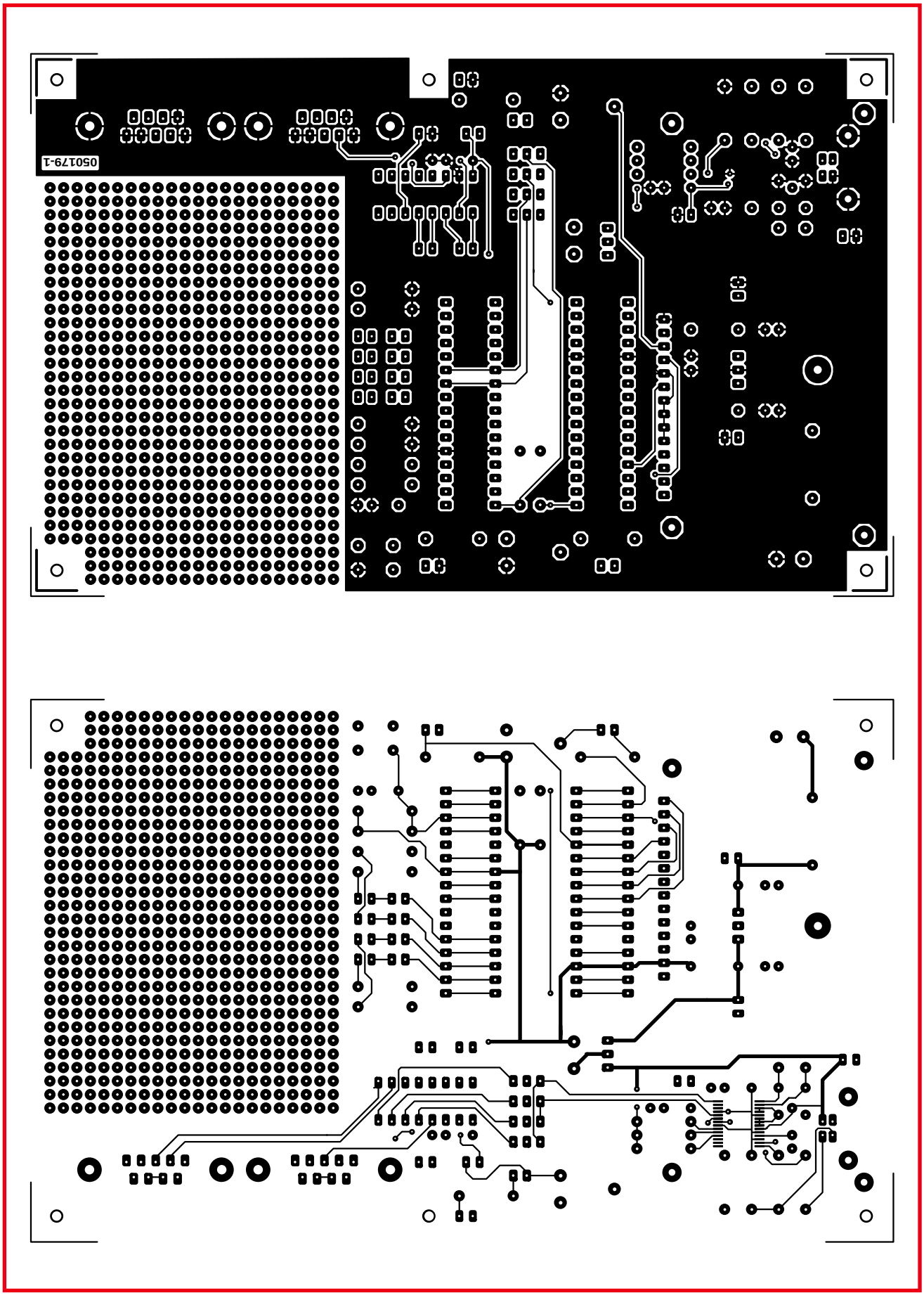


Figure 5a. Dessin du circuit imprimé ...

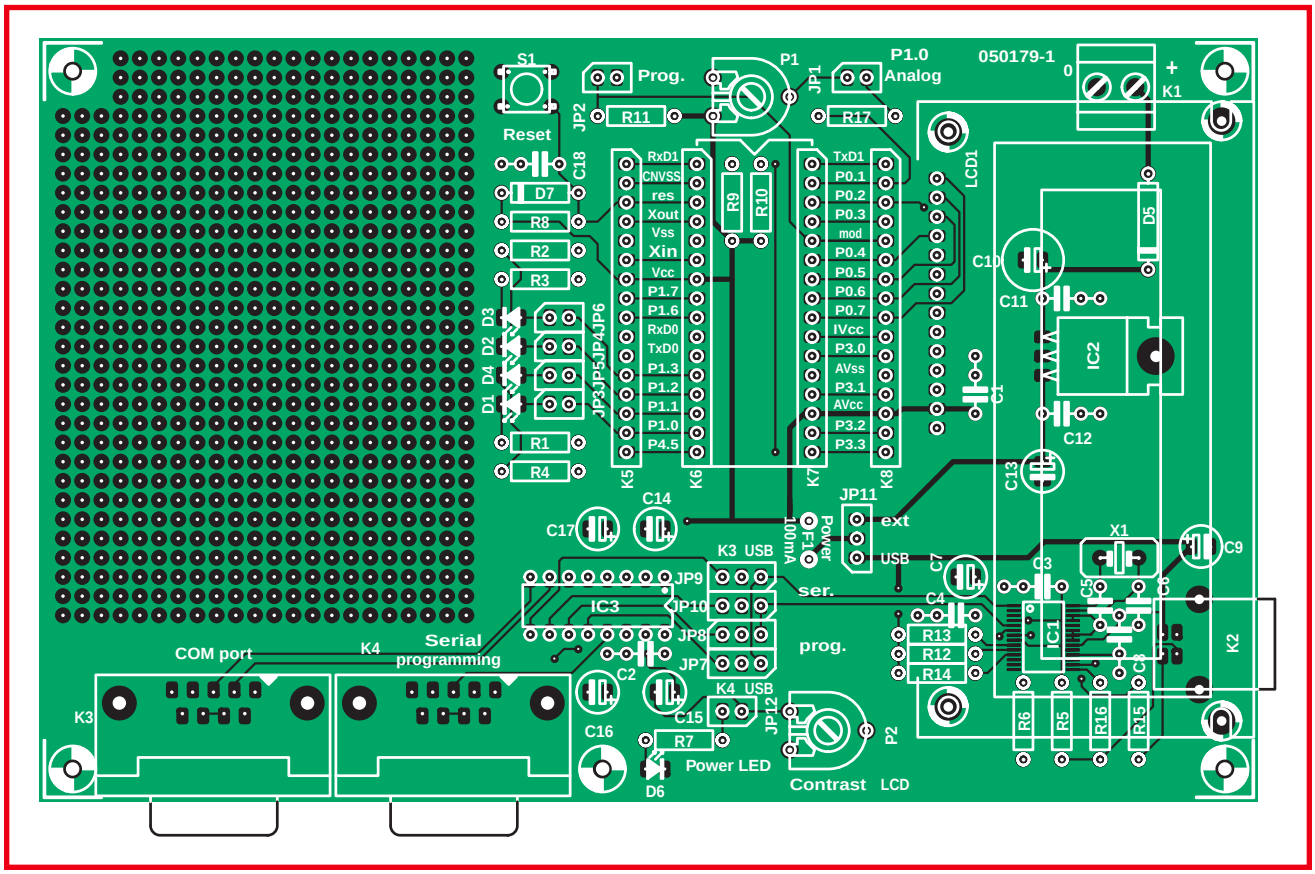


Figure 5b. ... et l'implantation des composants de la platine double face.

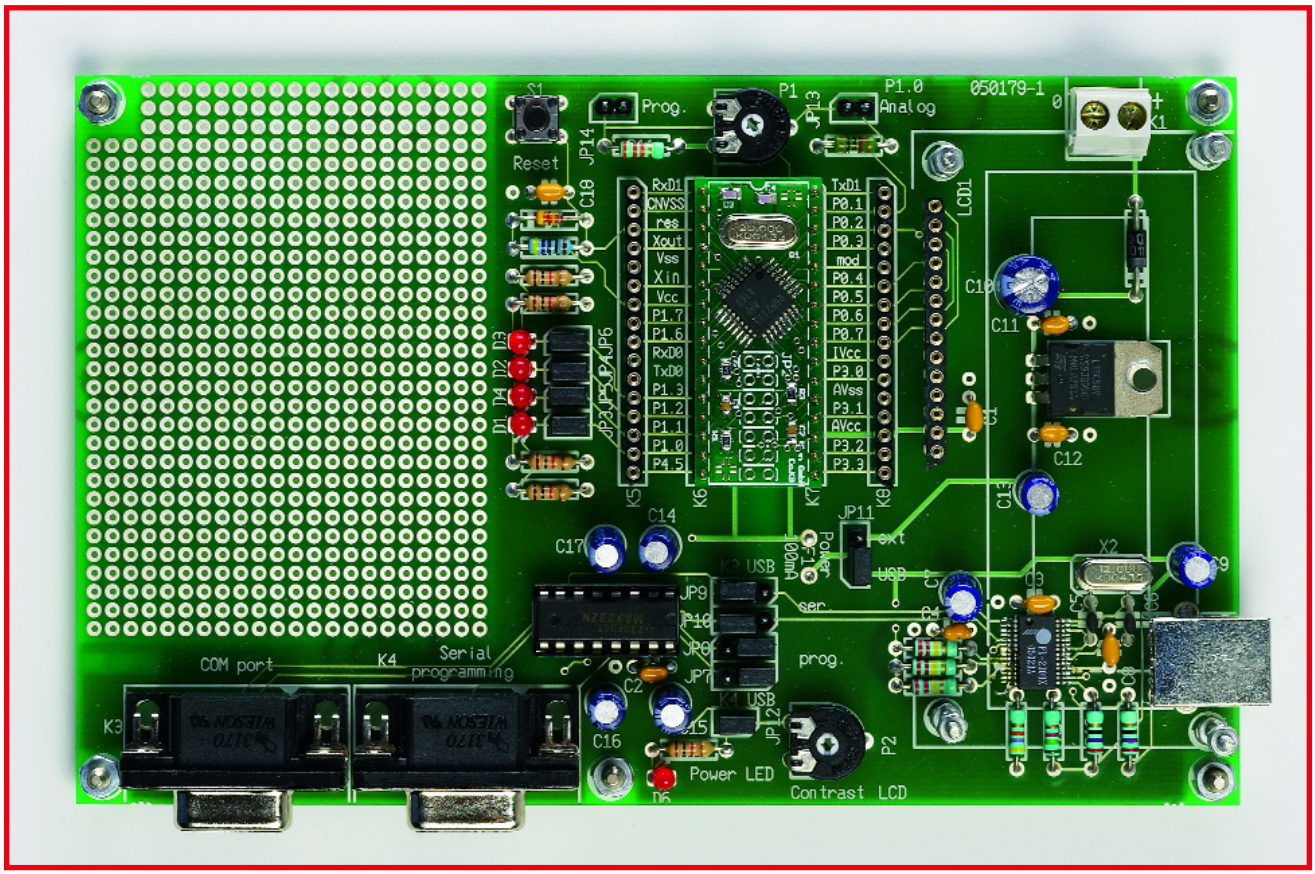
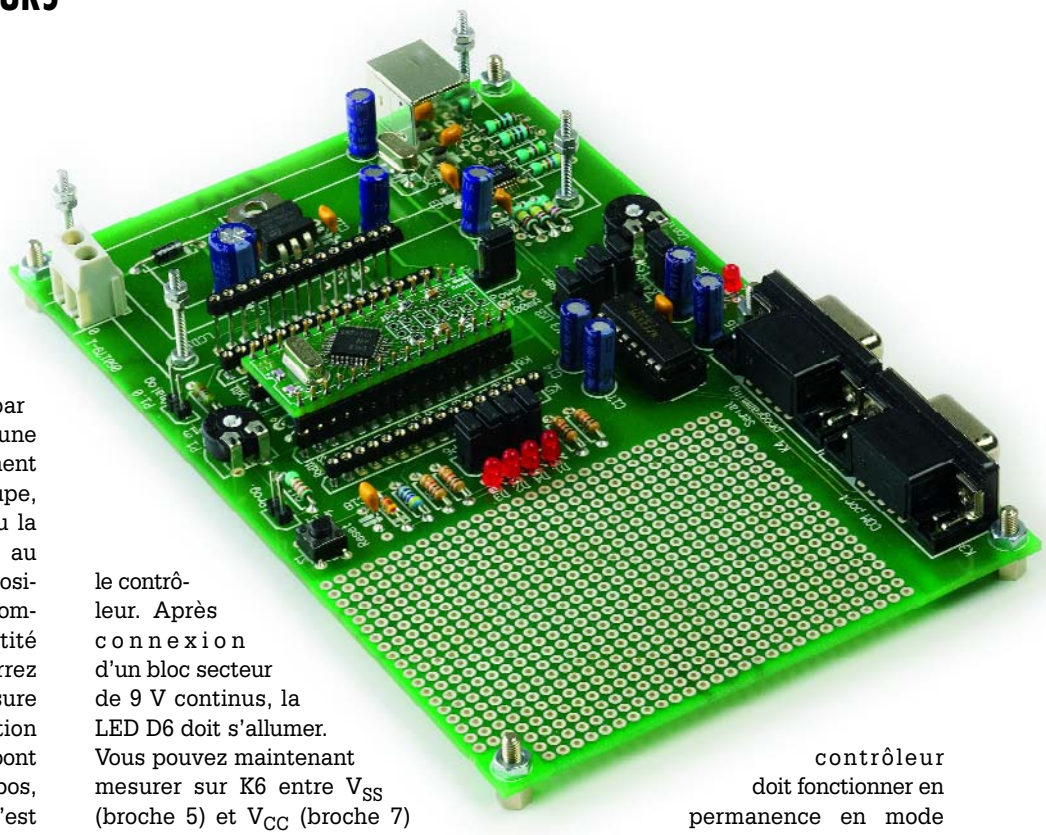


Figure 6. La platine du prototype terminée et équipée. L'implantation ne correspond pas encore tout à fait à celle de la version de série (figure 5b).



revanche, vous implantez vous-même les composants, commencez par ce circuit intégré, vous aurez moins de difficultés avec une platine encore vide. Des nombreux projets d'Elektor à CMS ont montré que l'on pouvait aussi s'en sortir sans outillage spécial pour CMS. Commencez par souder deux broches opposées d'une diagonale après avoir soigneusement aligné le circuit. Vérifiez à la loupe, vous saurez ainsi si vous avez eu la main sûre. Il est encore temps, au besoin, de rectifier, légèrement la position. Soudez ensuite une rangée complète de broches, avec une quantité suffisante de brasure. Vous pourrez ensuite facilement enlever la brasure en excès à la tresse. Si la vérification finale à la loupe ne détecte aucun pont de soudure, tout va bien. À ce propos, au lieu d'une loupe, l'un de nous s'est servi de fortes lunettes de lecture de 3 dioptries. Il s'en est fort bien trouvé. L'instrument a fait office de stéréomicroscope, à un prix sensiblement moins élevé.

Les autres composants ne posent pas de problèmes d'implantation particuliers. Il n'est pas indispensable d'implanter les barrettes K5 et K8. Si vous le souhaitez, vous pouvez par la suite connecter un câble méplat à cet emplacement ou n'effectuer que les connexions dont avez besoin sur le champ de matrice de points. Pour la connexion du LCD, nous avons préféré une barrette femelle. Garnissez ensuite le module LCD convenable de longues barrettes de broches et enfichez-le tout simplement.

Premier essai

Avant la mise en service, il est recommandé de contrôler encore une fois implantation et points de soudure et de poser les cavaliers nécessaires. Les principales positions de cavaliers sont les suivantes :

- JP 11 direction K7/K8 (position « ext », alimentation par bloc secteur extérieur)
- JP 3 à 6 posés (LED connectées)
- JP 12 posé (LED d'alimentation connectée)
- JP 2 posé (mode de débogage)
- JP 7 à 10 direction IC3 (les deux interfaces à RS-232)

Il n'est pas recommandé d'enficher la petite platine R8C/13 à cette étape Commencez d'abord par vérifier l'alimentation sans risque d'endommager le contrôleur. Après connexion d'un bloc secteur de 9 V continus, la LED D6 doit s'allumer. Vous pouvez maintenant mesurer sur K6 entre V_{SS} (broche 5) et V_{CC} (broche 7) une tension de + 5 V. Débranchez ensuite le bloc secteur et enfichez la carte support hors tension.

L'heure de vérité a sonné: rebranchez le bloc secteur de 9 V. La LED D6 d'alimentation s'allume. Reliez K4 à l'interface sérielle COM1 du PC. Comme vous avez sélectionné le mode de débogage avec JP2, vous pouvez charger tout de suite un programme. Utilisez le logiciel FDT comme le décrit le numéro 02/06 d'Elektor. Le mieux est de charger le projet « port_toggle ». L'outil de programmation signale la fin du chargement.

Retirez ensuite le cavalier JP2 et actionnez une fois brièvement la touche de réinitialisation. Les quatre LED du port P1 devraient clignoter.

Si vous souhaitez travailler par l'intermédiaire du port USB, enfichez les cavaliers JP7 et JP8 dans la position de droite (direction embase USB). Les lignes RXD1 et TXD1 sont maintenant reliées au circuit de conversion USB. Paramétrez maintenant FDT sur l'interface COM virtuelle. Pour le reste, tout fonctionne comme s'il s'agissait d'une authentique interface RS-232.

Débogueur KD30

Nous n'avons jusqu'ici flashé et lancé sur le contrôleur que des programmes tout prêts. Lorsqu'il s'agit de développer des projets plus compliqués, l'utilisation du débogueur est bien utile. Il permet d'interrompre les programmes aux emplacements voulus, d'afficher le contenu des mémoires, d'exécuter les programmes pas à pas et plus encore. Quand vous utilisez le débogueur, le contrôleur doit fonctionner en permanence en mode débogage, le cavalier JP2 reste donc enfiché. La touche de réinitialisation n'est plus nécessaire puisque votre programme se lance et s'arrête sur des commandes logicielles du débogueur.

Nous mettons à votre disposition sur le site d'Elektor www.elektor.fr un article additionnel donnant une description détaillée des premiers pas avec le débogueur. Vous trouverez ces informations sur la page du R8C accessible depuis la barre de menu de droite de la page d'accueil (juste en dessous des "E-Blocks").

(050179-3)

Liens :

Glyn GmbH & Co. KG

www.glyn.com

Mél. : elektor-r8c@glyn.de
(en anglais S.V.P.)

Questions au sujet de la R8C?

Venez faire un tour sur le Forum consacré au R8C : www.elektor.fr

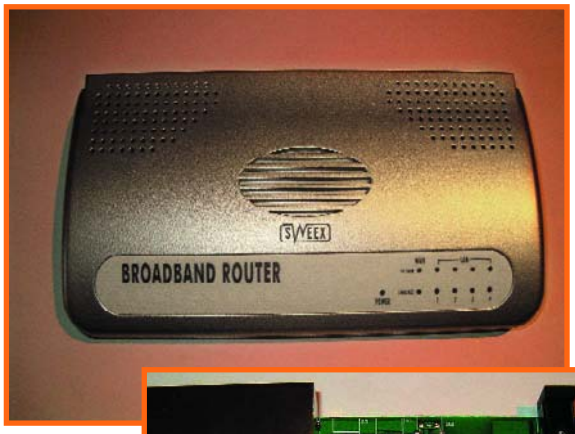
Sur le Forum allemand plusieurs spécialistes suivent le projet R8C du Forum, nous espérons trouver très bientôt également des experts en mC d'Elektor dans l'Hexagone. Venez partager vos astuces avec les autres lecteurs!

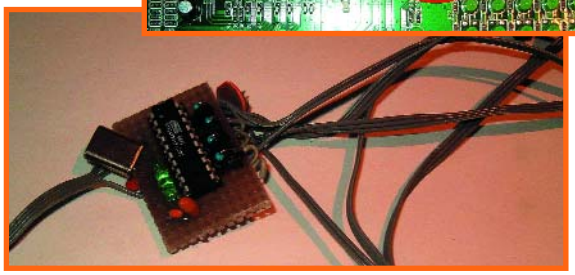
Logger bon marché

Utilisez un routeur Web pour mesurer votre consommation de gaz/eau/électricité

Jeroen Domburg & Thijs Beckers

Ce mois-ci nous allons transformer un routeur bon marché en un système d'acquisition de données (datalogger). Nous allons le doter de capteur chargés de surveiller les états des compteurs de gaz, d'eau et d'électricité. lA-côté intéressant de l'utilisation d'un routeur pour cette tâche est qu'il est possible d'interconnecter ce logger directement à notre réseau ordinateur ce qui en permet la lecture depuis n'importe quel PC doté d'un Browser Internet.





Dans le précédent numéro de cette rubrique nous vous avons présenté un routeur bon marché et avons montré comment l'utiliser à d'autres fins que celles prévues par le fabricant. Ce mois-ci, le même routeur, doté d'un rien d'électronique additionnelle, se voit transformé en instrument mesurant la consommation de GEE

Notre instrument possède 4 entrées de capteur de sorte qu'il pourra surveiller un maximum de 4 compteurs. On a, toutes les 5 minutes, lecture de l'état du compteur, ce qui permet, en fin d'opération, de dessiner de très jolis graphiques à partir des données recueillies. On peut voir ainsi à quel moment de la journée la consommation d'énergie est la plus importante et prendre les mesures requises pour la diminuer et rendre les factures plus supportables.

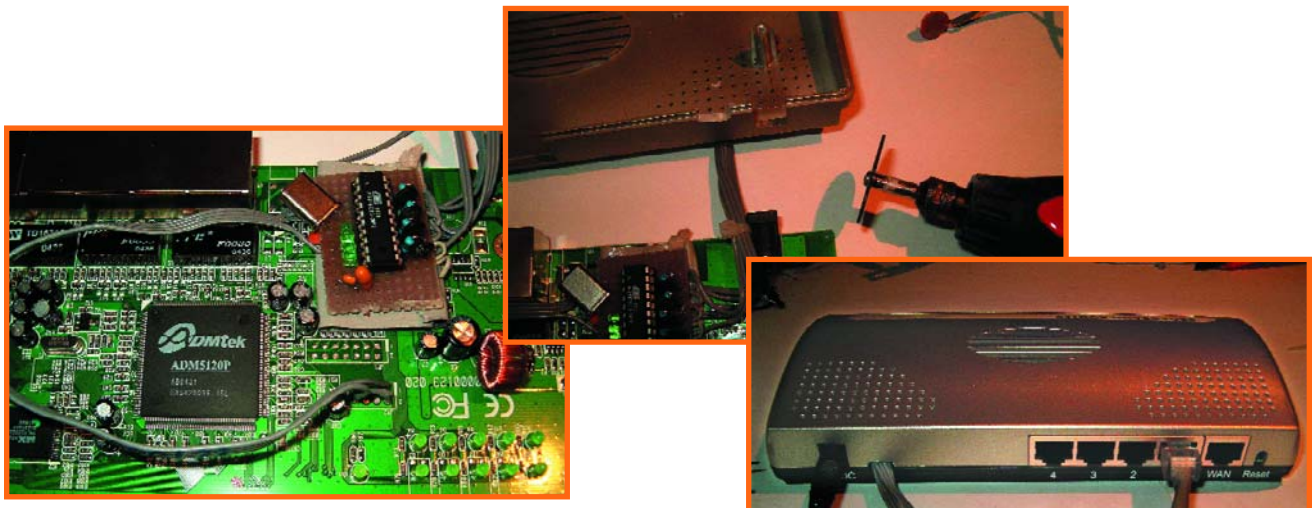
L'appareil lui-même prend place dans l'armoire à compteurs où il peut rester. Il est en effet possible d'accéder à toutes les mesure via un serveur Web intégré à cet effet. On peut donc connecter l'appareil au réseau domestique et le lire depuis son PC.

Le routeur que nous avons choisi est à nouveau le « Sweex broadband router » à 4 embases 100MBit; son numéro de

Le Sweex LB000021 reste, pour l'instant, le routeur le moins cher sur lequel tourne Linux. Lors de l'écriture de ces lignes, il est possible de trouver cet modèle sur Internet pour de l'ordre de € 30.

Après ouverture du routeur, on découvre une paire d'emplacements destinés à des embases vierges pour l'instant. Le plus compact (cerclé en rouge) est le port sériel. Nous allons l'utiliser.

Voici la platine du microcontrôleur qui sera connectée au port. Vu la simplicité de schéma, le circuit imprimé pourra prendre la forme d'un morceau de platine d'expérimentation à pastilles.



La platine est connectée au port et fixé à la platine du routeur à l'aide d'un morceau de scotch double face plié en double. Pas la peine de s'embêter avec des vis ou d'autres systèmes de fixation sophistiqués.
Rien de tel qu'un outil universel pour percer un trou dans le coffret pour permettre le passage des fils allant aux capteurs. Un serre-câble placé autour des fils fait office de dispositif anti-arrachement.
L'ensemble est connecté pour une mise à niveau (upgrade) du progiciel (firmware). Le câble réseau va directement vers le PC. Le câble UTP pourra être un câble soit 1:1 soit croisé (crosslink).

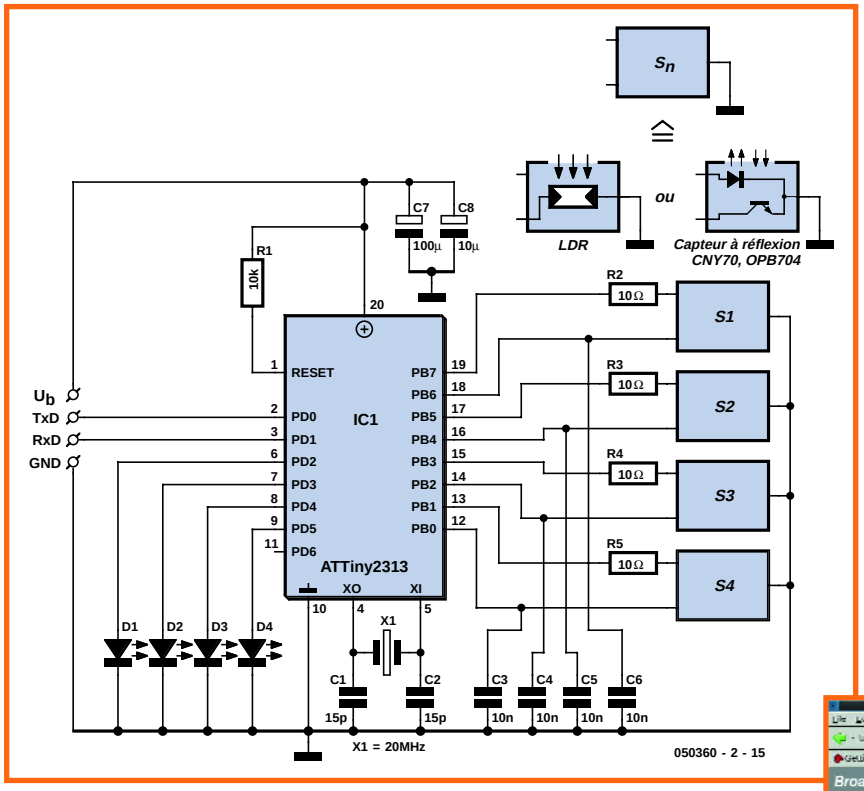
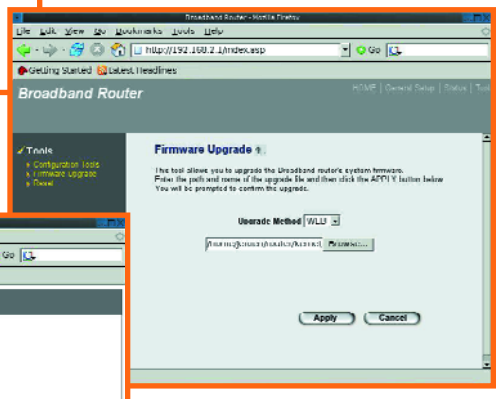
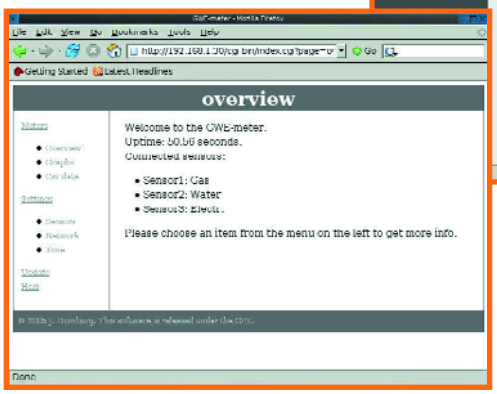


Figure 1. Le schéma au coeur duquel trône le microcontrôleur est simple; l'électronique pourra prendre place sur un morceau de platine d'expérimentation à pastilles (cf. les photos).

type LB000021 (sans WLAN). La raison de ce choix est qu'il reste le moins cher (cf [1]). Le reste du matériel ne coûte pas cher lui non plus, si l'on fait abstraction du microcontrôleur, un ATTiny2313 d'Atmel étant disponible pour quelques euros, et des capteurs.
Le système comporte 3 ensembles : le routeur lui-même, qui fait office de serveur Web et stocke toutes les données, la platine du microcontrôleur et les capteurs. Le routeur communique toutes les 5 minutes avec le microcontrôleur. Ce dernier surveille les compteurs à l'aide des capteurs.
Nous allons utiliser des capteurs de type optique, leur implantation dépendant du compteur avec lequel il est prévu de l'utiliser. Le concept sur lequel repose ce montage est de faire en sorte que le capteur « tienne à l'oeil » les



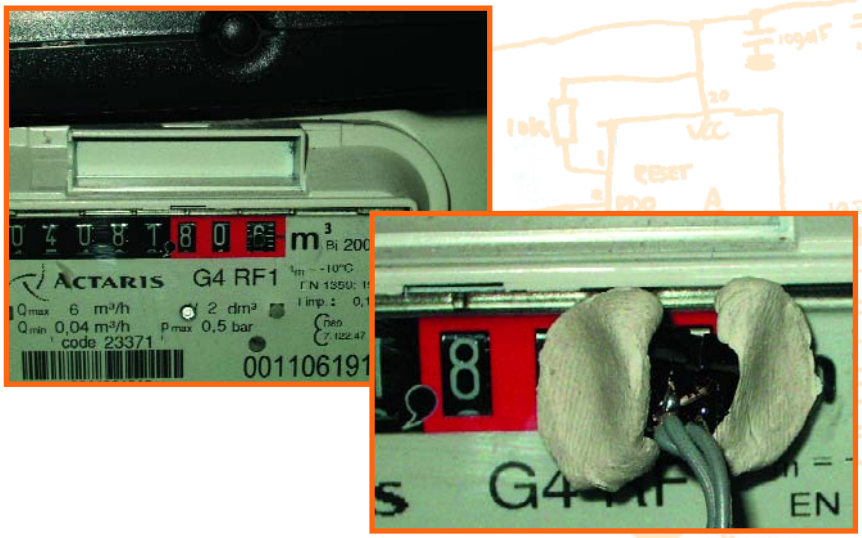


Une fois que tout est connecté, il est possible d'accéder au routeur via l'adresse IP 192.168.2.1. Par un passage, par le biais du menu « tools », à l'option « firmware upgrade », nous allons faire du progiciel un progiciel pour compteur GEE (Gaz, Eau, Électricité, GWE pour les pays anglo-saxons). Ça marche ! Une fois le progiciel mis à niveau, il est possible de connecter le routeur au réseau. L'ex-routeur se verra attribuer automatiquement une adresse IP.

Il est temps maintenant de connecter les capteurs. Un compteur d'électricité à LED clignotante moderne constitue le cobaye le plus coopérant…
Une LDR dotée d'un morceau de gaine plastique permet déjà d'obtenir une lecture de précision suffisante.

compteurs d'eau, de gaz et d'électricité. La plupart des compteurs récents sont dotés d'une LED qui s'illumine un bref instant tous les X (k)wh. On pourra lire cette impulsion lumineuse à l'aide d'une simple LDR. Les compteurs plus anciens possèdent un disque dont le bord est doté, en un certain point, d'un repère noir. Ce type de compteur se laisse lire à l'aide d'un capteur optique tel que, par exemple, le CNY70 ou (plus cher, mais meilleur) le OPB704.
Les compteurs de consommation de gaz et d'eau sont souvent dotés, sur l'un de leurs chiffres (souvent le « 0 » ou le « 6 ») d'un rond nu au niveau de leur position de chiffre le moins significatif. On pourra utiliser dans ce cas-là un capteur à réflexion.
Quelle que soit la technique de mesure utilisée, le résultat est que, de temps en temps, tous les capteurs présentent une valeur de résistance plus faible. Les condensateurs pris sur les capteurs (cf. **figure 1**) épaulés par un progiciel astucieux « gravé » dans le microcontrôleur permettent la lecture de changement de valeur de résistance.

Le progiciel comptabilise le nombre de fois que le courant au travers du capteur tombe en-dessous d'une valeur de consigne et par conséquent le nombre de variations de la résistance ayant lieu. Ce comptage constitue une bonne indication quant à la consommation sur une certaine période. Ceci permet également d'effectuer une calibration des capteurs.
Le microcontrôleur est relié à une embase du routeur, JP2, par le biais de son port sériel. Normalement le routeur n'est pas doté de cette embase qui permet d'accéder au port série du routeur (cf. **figure 2**). Le routeur a été doté d'un progiciel spécifique, progiciel dans lequel a été implémenté un programme assurant la communication avec le microcontrôleur et la chronologie correcte de certains événements tels que la lecture des compteurs, toutes les 5 mn comme nous le disions plus haut. Les données sont stockées dans la RAM interne du routeur. Ceci présente l'avantage que l'on peut stocker de l'ordre d'un an de données, l'inconvénient étant cependant une perte des données en cas de coupure inopinée du courant. Ce problème peut être contourné par la mise en place d'un accu de sauvegarde dans l'alimentation; nous y reviendrons plus loin.
Sur le routeur proprement dit il tourne un simple serveur Web et quelques fichiers script côté serveur; en fait tout ce dont on a besoin pour tout piloter, même le paramétrage du réseau et la calibration des capteurs, depuis un browser. Ce browser permet également, d'examiner la consommation de différentes manières. Les données fournies par (un ou) plusieurs canaux prédéterminés peuvent être mises en graphique pour examen. Les données brutes peuvent être transférées sous forme de fichier à extension .CSV pour être utilisées, par exemple, dans un tableur comme Excel.
On a également une possibilité de mise à jour du progiciel. Cette possibilité de mise à jour permet de reflasher dans le routeur une mise à jour officielle de Sweex pour que l'appareil retrouve sa fonction originelle de routeur. Notons que le progiciel de compteur n'empêche pas le commutateur interne de fonctionner : les 4 ports LAN et le port WAN se comportent, sous la houlette du dit progiciel, comme un grand commutateur.
Tout cela est bien joli, mais comment puis-je mettre la main sur ce progiciel ? Très simple, il vous suffit de faire un tour sur le site d'Elektor [2] ou sur celui de l'auteur [3] pour le télécharger. Tant le kernel Linux que les autres outils utilisés sont licencés sous une GPL (*General Public License*), ce qui implique qu'il faut mettre à disposition non seulement le progiciel mais également le code-source de ces programmes. Ce qu'a fait l'auteur. Il met également à disposition des outils écrits spécifiquement pour le progiciel du compteur. Vous trouverez tout ceci sur les sites mentionnés quelques lignes plus haut.
L'installation du progiciel est simple : le routeur connaît une possibilité de



Ce compteur de gaz a la caractéristique intéressante que le coeur de son « 6 » n'est pas peint, spécificité que met à profit un capteur à réflexion. Le capteur à réflexion est fixé au compteur à l'aide d'un morceau de gomme plastique du type de celle que l'on utilise pour fixer un poster au mur par exemple.

Les choses se compliquent quelque peu dans le cas de la lecture du compteur d'eau : pas de chiffre brillant, pas de LED clignotante… En principe il devrait pouvoir être possible de travailler avec un capteur à réflexion et une des petites aiguilles, en utilisant le cas échéant une LED bleue. L'auteur n'a cependant pas testé cette approche exhaustivement.

mise à jour du progiciel, ce qui permet d'y introduire en catimini notre progiciel adapté; le diaporama donne les différentes étapes du processus. Si l'on veut reflasher le routeur dans son état d'origine, ceci se fera de la façon évoquée plus haut.

Le microcontrôleur se laisse programmer par le biais d'un programmateur [4] prenant la forme d'un connecteur Sub-D à 25 contacts dotée de 3 résistances connecté au port parallèle d'un PC. Il existe un logiciel de programmation tournant sous Windows et d'autres systèmes d'exploitation.
Le positionnement physique des capteurs dépend des compteurs auxquels on a affaire. Dans le cas d'un compteur à LED flashant cette information est facile à saisir à l'aide d'une LDR plaquée au niveau de la LED à l'aide d'un morceau de scotch. Les choses sont plus épineuses dans le cas des compteurs d'eau et de gaz, mais tant que le capteur à réflexion peut être placé en regard du chiffre le plus à droite, il ne devrait pas y avoir de problème insurmontable.
L'auteur n'a pas pu tester le cas d'un compteur d'électricité sans LED (ne comportant qu'un disque rotatif), mais il est fort probable qu'il soit possible d'utiliser le repère noir ou rouge du disque pour une détection à l'aide d'un capteur à réflexion.
Une fois les capteurs placés et branchés, il va falloir les calibrer. Il existe, sur la page web du routeur, une option à cet effet. Cette fonction procède à une lecture continue de la valeur (analogique) du capteur jusqu'à ce que l'utilisateur déclare la calibration effectuée. On détermine, à partir de ces données, un minimum et un maximum, limites entre lesquelles se promène le capteur. |A chaque fois que le capteur passe au-delà des 2/3 de la différence entre le minimum et le maximum, le microcontrôleur en déduit que l'on vient d'utiliser une unité de gaz, d'eau ou d'électricité.
Il n'y a aucune raison de craindre que le routeur ait un effet sensible sur votre facture d'électricité : il ne consomme que 340 mA sous 12 V, soit une consommation de l'ordre de 5 watts. Le routeur est doté d'une alimentation à découpage : on peut donc lui appliquer n'importe quelle tension comprise entre 6 et 15 V.

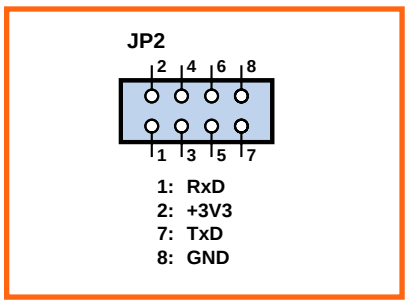


Figure 2. **Brochage de l'embase JP2 du routeur.**

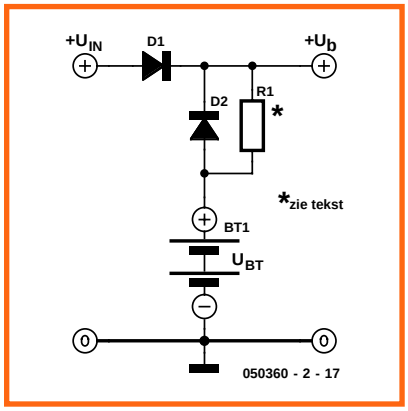


Figure 3. **L'alimentation de secours doit venir « se glisser » dans la ligne d'alimentation. Le schéma montre qu'il s'agit d'une électronique simple.**

Les infos qui suivent vous permettent de réaliser un circuit de sauvegarde (*back-up*) simple, évitant la perte de données. Un accu BiMH ou CdNi de 9 V permet au routeur de tenir le coup pendant quelque 20 mn en cas de disparition de la tension du secteur. Une demi-douzaine de gros accus NiMH permettent de ponter une absence d'alimentation secteur de l'ordre de 3 heures. Le compteur poursuit ses mesures. On ne perd pas, ainsi, les informations des compteurs de gaz et d'eau. La **figure 3** montre comment brancher l'alimentation de secours qu'il faut prendre entre l'adaptateur et le routeur. Les 2 diodes évitent qu'il n'y ait circulation de courant des accus vers l'adaptateur, la résistance permettant une charge des accus telle qu'ils soient toujours « pleins » en fonctionnement normal. Pour les diodes, on utilisera de préférence des Schottky, de sorte que la chute de tension (et donc les pertes/consommation) soit un peu plus faible. La valeur de la résistance répondant à la formule suivante :

R = (Ub – Ubat)/ (1/50 * capacité accu)

Avec un adaptateur 12 V et un accu 9 V de 800 mAh cela nous donne une résistance de 130 Ω.

(050360-2)

Liens Internet :

[1] http://boutique.3dchips-fr.com/product_info.php/cPath/73_79/products_id/2644

[2] www.elektor.fr

[3] http://sprite.student.utwente.nl/~jeroen/projects/gwemeter

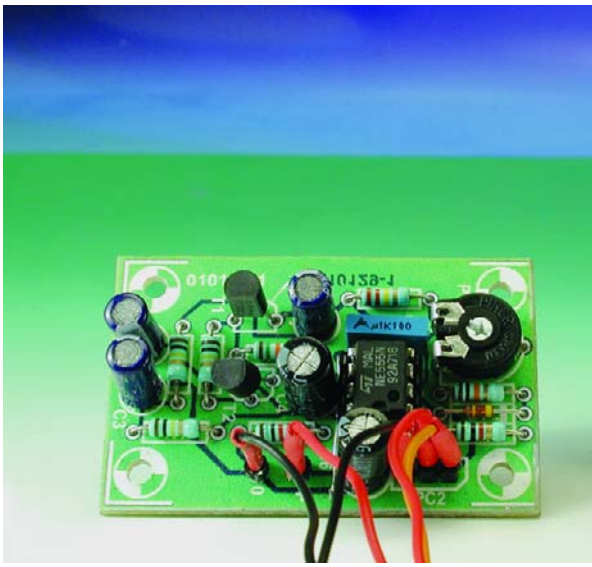
[4] http://sprite.student.utwente.nl/~jeroen/projects/progavr

L'auteur :

Jeroen Domburg est, dans sa vie de tous les jours, étudiant en Electrotechnique à IUT Saxion d'Enschede (NL). Il est un amateur éclairé et enthousiaste, qui, consacre son temps libre aux microcontrôleurs, à l'électronique et aux ordinateurs. Cette rubrique lui donnera l'occasion de proposer à nos lecteurs ses projets de conversion pour qu'ils puissent les réaliser à leur tour.

Servomatic de relax

Burkhard Kainka



Les servo-commandes ne servent, au mieux, que quelques jours par an. Dès que les aéroplanes ont retrouvé leur hangar et les bateaux leur « chantier naval », toutes les servo-commandes se trouvent au repos forcé. Pour éviter qu'elles ne perdent « leur muscle », on pourrait fort bien envisager de leur trouver quelque chose d'utile à faire.

Une servo au chômage pourrait ainsi, par exemple, piloter un grand pendule qui, par son mouvement, détendrait la personne qui l'observe, voire la plongerait dans un profond sommeil. Le point important auquel il faut faire attention est que ce mouvement soit, comme cela est le cas avec les ondulations d'un vrai pendule de grande taille (cf. le pendule de Foucault), purement sinusoïdal.

Le minuscule montage que nous avons baptisé Servomatic et dont on retrouve le schéma en **figure 1** se subdivise en 2 parties. Le générateur d'impulsions repose sur un temporisateur du type 555; il génère des impulsions positives de longueur comprise entre 1 et 2 ms, impulsions séparées par des pauses de quelque 20 ms, ce qui est très précisément le type de signaux qu'attend une servo-commande.

Le second sous-ensemble est un générateur sinusoïdal travaillant à une fréquence très faible de quelque 0,25 s, ce qui correspond à une durée de période totale de 4 secondes. Le générateur sinusoïdal fait appel à un réseau de déphasage et fournit ainsi des signaux sinusoïdaux caractérisés par une distorsion faible. Il suffira, si l'on veut

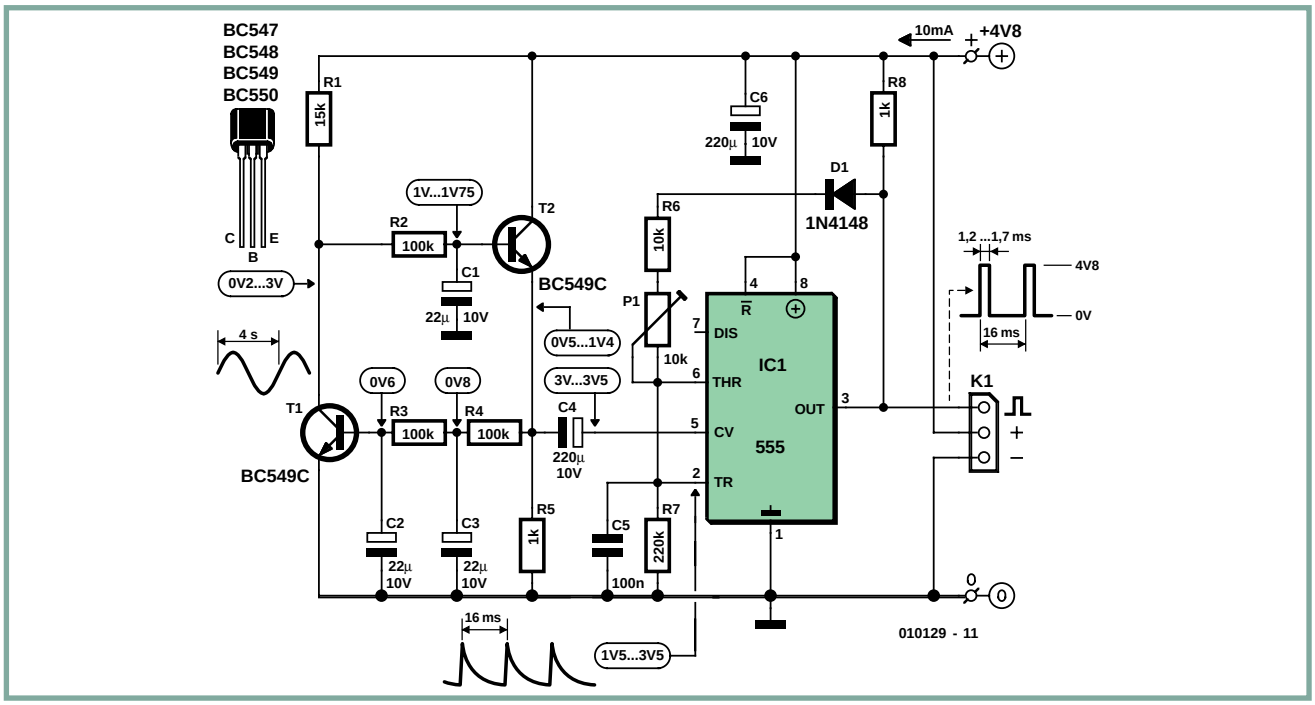


Figure 1. L'électronique de Servomatic de relaxation.

xation pour mise en sommeil

allonger cette durée de période, d'augmenter la valeur des condensateurs électrochimiques C1 à C3.
Au travers d'un condensateur électrochimique le générateur attaque l'entrée de commande du circuit intégré temporisateur (il faudra utiliser ici sa version CMOS 7555 ou un 555C). C'est ainsi que se fait le pilotage de la longueur de l'impulsion fournie par le temporisateur, la durée des pauses ne bougeant pas elle.
Nous avons dessiné une platine à l'intention de la Servomatic de relaxation (cf. **figure 2**). Elle est disponible au près de The PCB Shop (www.elektor.fr).
Il est difficile, lors de l'implantation des composants, de faire une erreur, pour peu que l'on respecte au pied de la lettre la sérigraphie de l'implantation des composants et la liste des composants.
Il ne faudra pas oublier le pont de câblage à proximité de IC1, pont à réaliser à l'aide d'un conducteur isolé en raison de la proximité des pattes du circuit intégré.
On pourra utiliser, pour T1 et T2, n'importe quel transistor NPN si tant est qu'il un facteur d'amplification (gain) suffisant (type C, h_{fe} >300, tel que, entre autres, les BC547C, BC548C, BC549C, BC550C).
P1 sera réglé de manière à ce que la servo batte bien vers la gauche et vers la droite sans cependant arriver en butée.
Si l'élongation (déplacement d'un mobile par rapport à sa position d'équilibre dit le Petit Robert) est trop importante ou trop faible, on pourra expérimenter en jouant quelque peu sur la valeur de R1.
Si R1 est trop petite (<10 kΩ), l'oscillateur ne démarrera même pas. Le condensateur C4 exerce lui aussi une influence sur l'amplitude. La consommation de courant du circuit est de 10 mA environ, hors consommation de la servo-commande elle-même.
L'embase à 3 contacts K1 devrait pouvoir recevoir directement la plupart des servo-commandes, certains modèles requérant cependant un petit adaptateur.

Une servo-commande connectée correctement présente maintenant un mouvement sinusoïdal très lent. Un pendule d'une longueur de quelque 4 mètres aurait une fréquence d'oscillation similaire. Vous pouvez aussi fixer à l'axe de la servo-commande un petit pendule, une rondelle de carton coloriée ou quoi que ce soit d'autre si tant est que l'adjonction ne soit ni trop grande ni trop lourde (en raison de la charge que peut supporter une servo). Ce système possède un effet très relaxant. À cela s'ajoute le bruit du moteur de la servo qui, à ce mouvement lent, tient plus d'un « ronronnement ». Cela contribue indiscutablement à la relaxa... uchrrr chrrrr… uchrrr chrrrr...

(010129-1)

Liste des composants

Résistances :
R1 = 15 kΩ
R2 à R4 = 100 kΩ
R5, R8 = 1 kΩ
R6 = 10 kΩ
R7 = 220 kΩ
P1 = ajustable 10 kΩ

Condensateurs :
C1 à C3 = 22 µF/10 V
C4, C6 = 220 µF/10 V
C5 = 100 nF

Semi-conducteurs :
D1 = 1N4148
T1, T2 = BC549C
IC1 = 555C ou 7555 (CMOS)

Divers :
K1 = embase auto-sécable à 1 rangée de 3 contacts
2 picots

Platine 010129-1 (cf. The PCB Shop)

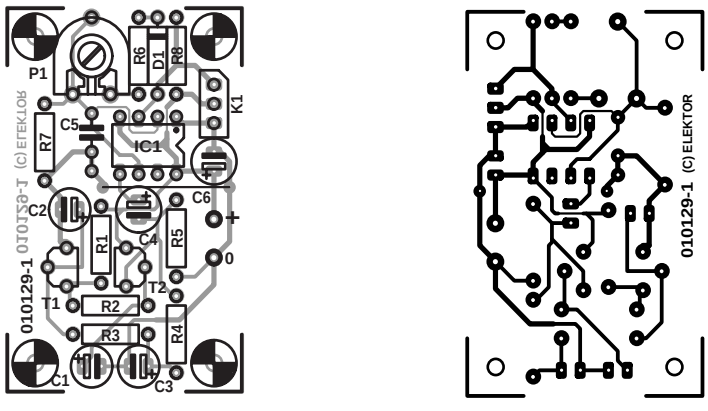


Figure 2. Dessin des pistes et implantation des composants.

Rênes en main

Un environnement de développement moderne au

Thijs Schoonbrood

Vous voulez vous lancer dans un nouveau projet de programmation. Vous en avez clairement la structure à l'esprit et vous savez déjà quel langage de programmation vous allez utiliser. C'est alors que se pose le dilemme : quel environnement de programmation utiliser ? Si l'un est trop complexe pour l'écriture d'un petit programme, l'autre est dépassé dans le cas d'un projet d'une certaine importance.

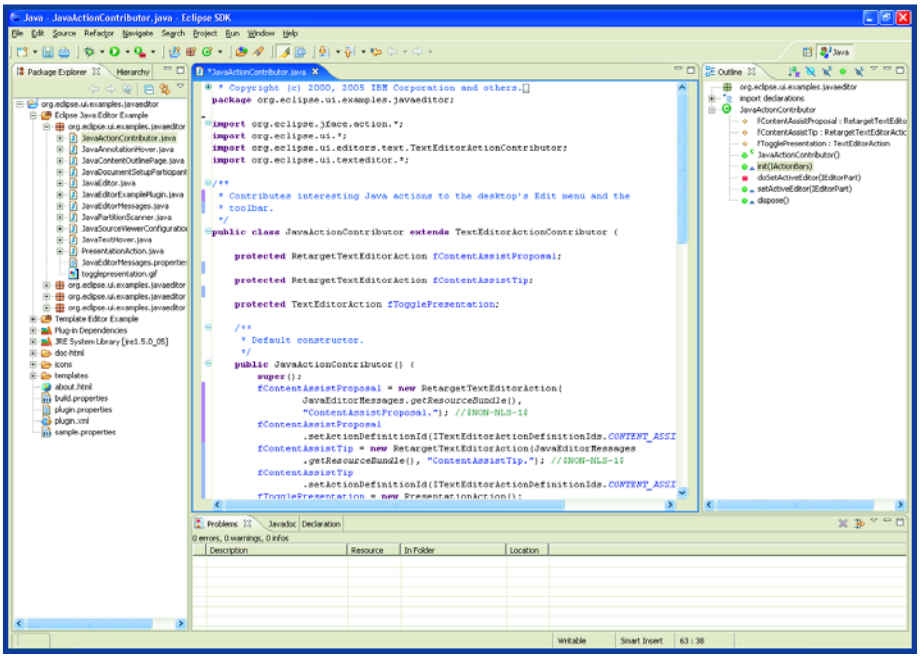


Figure 1. Voici à quoi ressemble Eclipse la plupart du temps.

Eclipse est un environnement de développement moderne facile à apprendre, se laissant aisément doter de nouvelles possibilités et convenant aux langages de programmation les plus divers. De plus, il s'agit d'un logiciel gratuit et open-source !

Eclipse est un cadre logiciel sans lien de plateforme destiné au développement d'applications rich-client. Le programme est souvent utilisé en IDE (*Integrated Development Environment*) Java et considéré comme l'un des meilleurs de son genre. Mais Eclipse convient également fort bien à d'autres langages de programmation. Bien que ce soit IBM qui soit à l'origine du développement d'Eclipse, ce programme est entretenu depuis quelques années déjà par la Eclipse Foundation, une association à buts non lucratifs épaulée par, entre autres, IBM et Borland.
La version (stable) la plus récente d'Eclipse est sa version 3.1. On pourra la trouver, tout comme les versions précédentes, à l'adresse www.eclipse.org. La taille du programme est, si on la compare à celle d'autres IDE, (relativement) faible : un peu plus de 100 Moctets. Cette compacité tient au fait que nombre de fonctionnalités d'Eclipse prennent la forme de Plugins, aspect auquel nous reviendrons plus loin.

Dès le téléchargement terminé, vous pouvez immédiatement vous mettre au travail vu qu'Eclipse ne requiert pas d'installation. On décompacte le fichier .zip et au travail. Il faut, lors du démarrage du programme, paramétrer le workspace, le dossier dans lequel seront placés nos pro-

grâce à Eclipse

ux nombreuses possibilités

jets. Nous arrivons ensuite dans l'écran de bienvenue qui nous offre le choix entre différents didacticiels ou exemples soit nous mettre immédiatement au travail.

Perspectives en tous genres

Nous nous retrouvons maintenant dans le workbench (l'établi). Celui-ci est constitué de plusieurs perspectives. L'idée à la base de cette mise en module est qu'il faut, lors du développement d'un programme, distinguer différentes tâches, dont, par exemple, la programmation (produire le code-source), le débogage (élimination des erreurs) et l'adjonction et la gestion de code de parties tierces. Comme quantité d'informations concernant une tâche donnée ne présentent pas d'intérêt pour une autre tâche, celle-ci n'est disponible que pour la tâche en question (lire la perspective correspondante). Ceci se traduit par un écran lisible affichant l'information nécessaire et suffisante. Une perspective est, à son tour, subdivisée en plusieurs Visus (Views) et éditeurs.
Un éditeur se trouve le plus souvent au centre de l'écran et sert à modifier le code. Une Visu donne, sous forme graphique, des informations supplémentaires au sujet d'un ensemble donné. Si, au premier abord, l'utilisation de perspectives peut paraître abstraite, on s'y habitue rapidement une fois que l'on s'y met. Une perspective est très malléable, on peut en modifier pratiquement tout, y compris le choix, la position et la distribution des différents visus et éditeurs. On dispose ainsi toujours de l'information la plus utile.

Au travail avec Java

Eclipse convient à merveille pour le développement d'applications en Java. Dans ce cadre, la perspective Java joue un rôle central. Tout à gauche on trouve le Package Explorer. Il s'agit d'une sorte d'explorateur qui visualise le contenu de tous les projets se trouvant à l'intérieur d'un workspace déterminé. Pensez au code-source, aux fichiers graphiques et de configuration, mais aussi aux bibliothèques importées et au JRE (*Java Runtime Environment*) utilisé. Depuis le Package Explorer vous choisissez les fichiers à visualiser ou à modifier.
Si l'on veut avoir plus d'informations au sujet du fichier ouvert dans l'éditeur on peut faire appel à l'outline-view. Ce dernier comporte 3 parties : tout en haut le package dans lequel se trouve la classe. En dessous on voit une liste des déclarations d'importation, que l'on peut faire disparaître si on le désire pour disposer de plus d'espace pour le reste. La dernière partie, la plus intéressante, donne une vue détaillée de la structure de la classe en question, structure intégrant les variables et les méthodes avec l'information correspondante.

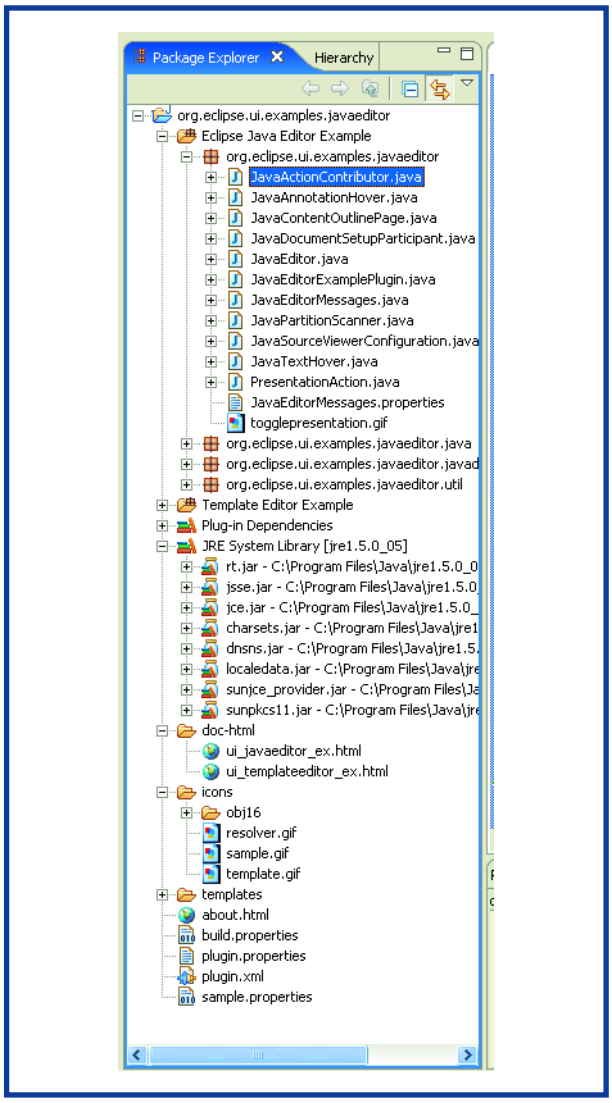


Figure 2. Package Explorer permet de se faire une idée sur les éléments constitutifs d'un projet.

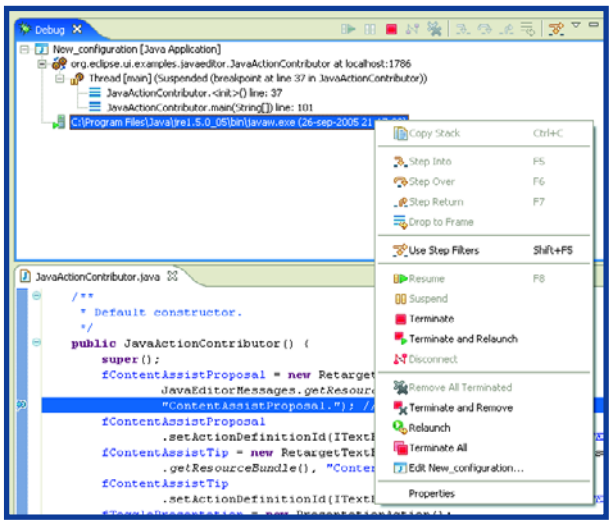


Figure 3. Vous êtes, en cours de débogage, tenu au courant de tout ce qui se passe.

Figure 4.
Sur le seul site Web d'Eclipse on trouve déjà près de 1 000 plugins.

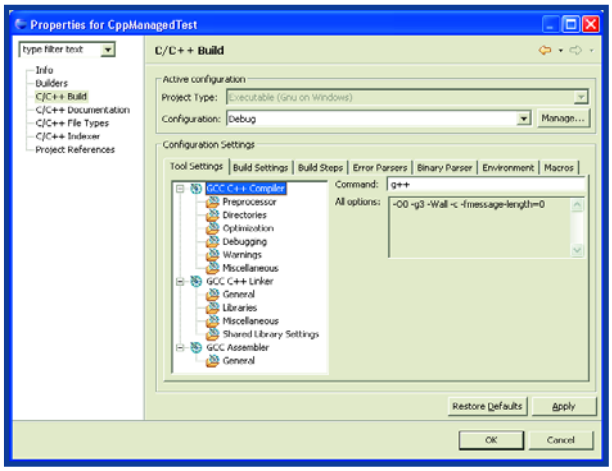


Figure 5.
Le compilateur connaît une pléthore de paramètres.

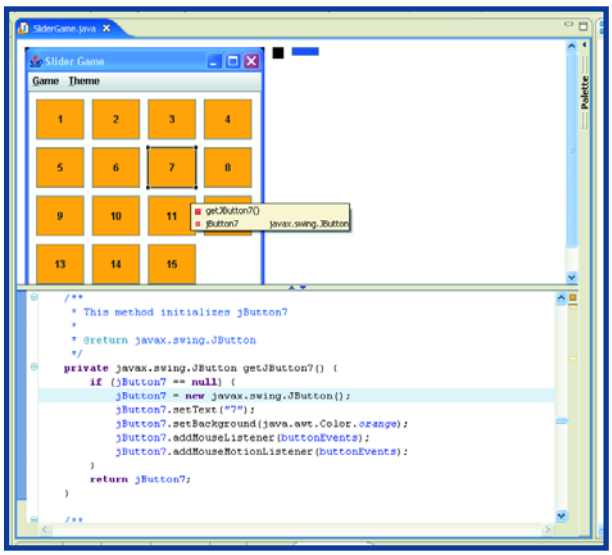


Figure 6.
Visual Editor permet de savoir, instantanément, à quoi ressemblera une application graphique.

Un coup de main

Eclipse comporte une option baptisée « auto completion » qui visualise toutes les possibilités intéressantes dans le cadre d'une classe. Il suffit ainsi d'entrer le mot « get » et d'appuyer simultanément sur la touche Ctrl et la touche Espace pour voir apparaître toutes les méthodes commençant par « get ». Et mieux encore, on voit apparaître la documentation Java correspondante dans une petite fenêtre popup.
Cette petite fenêtre d'auto-complément sert également dans un autre contexte, l'option correspondante s'appelant alors « quick assist ». Ceci est très pratique si l'on veut modifier rapidement du code en cas de message d'erreur ou d'avertissement du compilateur. La fenêtre « Problems » affiche toutes erreurs constatées et les avertissements donnés. Il suffit de faire « Ctrl+1 » pour qu'Eclipse propose quelques options pour la solution du problème.

Recherche d'erreurs

Quelle que soit l'intelligence d'un environnement de développement, tout code d'une certaine longueur comporte toujours des erreurs. En cas de détection d'une telle erreur, il faut en découvrir la raison, l'environnement de débogage tombant alors à point nommé. Vu l'importance du débogage, il s'est vu attribuer sa propre perspective. Cela permet de visualiser l'information indispensable lors du débogage.
On y trouve bien évidemment tout ce qui a trait aux valeurs actuelles de variables, aux points d'arrêt (breakpoint), etc. On y présente également les threads en cours et leur état. Il est possible, à loisir, de terminer des threads, les poursuivre ou les redémarrer.
Les choses deviennent intéressantes lorsque nous nous mettons:à utiliser des points d'arrêt conditionnels. Un point d'arrêt dans une boucle qui n'entre en action qu'après la 7ème itération ? Pas de problème ! Ou uniquement lorsqu'une variable a une valeur prédéterminée ? Cela aussi est possible.

Coopération avec suivi de version

Les programmeurs travaillant à des projets avec d'autres développeurs utilisent, en règle générale, un système de suivi de version. Il est indispensable de disposer de différentes versions de code-source en un même point central. Le système le plus souvent utilisé s'appelle CVS (*Concurrent Versions System*). Eclipse le supporte de pas sa nature même; il existe une perspective spécifique pour la gestion de reposoirs CVS (*CVS-repository*). Cette perspective sert, outre à la classique vérification et synchronisation de projets, également à la gestion de tags et de branches. Inutile de s'embêter avec une application diff externe, Eclipse disposant d'origine de cette fonctionnalité ainsi que d'une visu pour l'examen de l'historique des ressources d'un projet. La gestion de tags et de branches en devient un jeu d'enfant.

Fonctionnalités supplémentaires par plugins

Comme nous le disions, l'extension des fonctionnalités d'Eclipse se fait par plugins. Il en existe une quantité industrielle que l'on peut, grossièrement, diviser en 2 groupes. Le premier, placé sous la houlette de l'organisation Eclipse, n'en comporte que relativement peu. Citons, au nombre d'entre eux, les plugins C/C++ et Visual Editor auxquels nous reviendrons plus loin. On les trouvera à l'adresse www.eclipse.org/tools. Le second groupe, bien plus étoffé,

comprend les plugins de tierce partie. Nombre de ces derniers sont en mesure de se tenir eux-mêmes à jour grâce au gestionnaire de mise à jour d'Eclipse intégré.

C 4 yourself

Pour nombre de développeurs, un IDE n'est vraiment intéressant que lorsqu'il est en mesure de s'accommoder de plusieurs langages de programmation, sans que cela n'aille (trop) au prix de leur fonctionnalité.
Ceci explique que la fondation Eclipse ait lancé le projet CDT (*C/C++ Development Tools*). Pour en savoir plus à son sujet, allez à www.eclipse.org/cdt. Comme le donne à penser son nom, ce plugin met Eclipse en mesure de travailler avec du code C et C++. Ceci signifie que nous pouvons alors mettre à contribution les possibilités de CVS, le débogueur, l'auto-complément, etc. Contrairement à ce qui est le cas en cas de développement sous Java, nous ne disposons pas avec ce plugin, directement d'un compilateur intégré. Ceci est plus délicat avec C et C++ en raison de leur grande dépendance de la plate-forme sur laquelle ils tournent. Pas de problème sous Linux. La quasi-totalité des distributions connaissent une version de GCC (*GNU C Compiler*), utilisable sans problème par Eclipse. Par le biais de CygWin (http://cygwin.com) GCC devient utilisable, y compris nombre d'autres fonctions Linux, sous Windows. Si vous n'avez que faire de ces fonctionnalités Linux, MinGW (www.mingw.org) constitue une excellente alternative.

Interface utilisateur sur la sellette grâce à Visual Editor

Si vous créez de nombreuses applications en Java, il vous faudra, souvent, développer et implémenter une GUI (Graphical User Interface). Vous pouvez bien entendu vous y mettre vous-même et entrer tout le code en vérifiant régulièrement le résultat en démarrant l'application. Dans le domaine de l'implémentation des modules graphiques en particulier, la méthode WYSIWYG (*What You See Is What You Get*) est très prisée. Cette technique permet une « composition » graphique d'une GUI, tandis que l'on a, en arrière-plan, génération du code-source correspondant. Dans le cas d'Eclipse, cette fonctionnalité se trouve dans un plugin baptisé Visual Editor (VE) (cf. www.eclipse.org/vep).

En conclusion

Eclipse est donc utilisable avec différents langages de programmation, sans même parler de COBOL, PHP, développements sous UML/UML2 ou création de bases de données. Tous ces projets sont gratuits et ne cessent d'être réactualisés. Il se trouve ainsi pratiquement toujours une configuration d'Eclipse répondant à vos souhaits. Si cela ne devait pas être le cas, il ne vous reste plus qu'à créer votre propre plugin...

(060018-1)

AIDES À LA RÉALISATION

Elektor ne fait pas la vente de composants. Ceux-ci sont normalement à trouver chez un revendeur de composants. Il nous a cependant semblé nécessaire, suite à de nombreuses lettres, de résumer sur cette demi-page les informations cruciales pour la lecture et la compréhension des articles publiés dans Elektor. Nous utilisons, pour l'indication des valeurs de composants, les préfixes (classiques) suivants :

E (exa) = 10^{18}	a (atto) = 10^{-18}
P (peta) = 10^{15}	f (femto) = 10^{-15}
T (tera) = 10^{12}	p (pico) = 10^{-12}
G (giga) = 10^{9}	n (nano) = 10^{-9}
M (mega) = 10^{6}	µ (micro) = 10^{-6}
k (kilo) = 10^{3}	m (milli) = 10^{-3}
h (hecto) = 10^{2}	c (centi) = 10^{-2}
da (deca) = 10^{1}	d (deci) = 10^{-1}

Dans certains schémas et dans la liste des composants nous préférons utiliser, contrairement aux recommandations IEC et BS, le préfixe + symbole comme caractère délimiteur en remplacement de la virgule. 2 exemples :

3kΩ9 = 3,9 kΩ 4µF7 = 4,7 µF

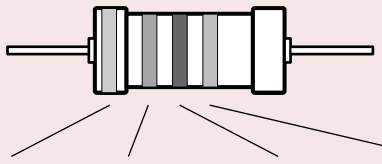
Sauf mention contraire, la tolérance des résistances est ±5% et leur wattage 1/3 à 1/2 watt. La tension de service des condensateurs est de ≥ 50 V.

Lors de **la mise en place des composants** on commencera en règle générale par l'implantation des composants passifs de la taille la plus faible, c'est-à-dire les ponts de câblage, les résistances et les petits condensateurs; on passera ensuite aux supports pour circuits intégrés, aux relais, aux condensateurs de forte capacité tels que les électrolytiques et aux connecteurs et autres embases. Les semi-conducteurs vulnérable et les circuits intégrés fragiles seront montés en dernier.

Le soudage. On utilisera un fer à souder d'une puissance de 15 à 30 W doté d'une pointe fine et de la soudure à âme de résine (60/40). On enfiche les connexions du composant concerné dans les orifices prévus à cette intention, on les replie légèrement, on les coupe à la bonne longueur et on procède à leur soudure; on attend de 1 à 2 secondes jusqu'à ce que l'alliage étain/plomb devienne liquide et vienne souder relier la connexion au métal de l'orifice. On peut alors enlever le fer à souder. Attention à éviter de surchauffer le composant en particulier les circuits intégrés et les semi-conducteurs. S'il faut désouder un composant on utilisera de préférence un fer à dessouder à pompe aspirante ou un appareil spécialement prévu à cet effet.

Le dépannage. Si le circuit ne fonctionne pas correctement, il faudra comparer soigneusement les composants mis en place sur la platine avec la sérigraphie de l'implantation des composants et vérifier leurs caractéristiques à l'aide de la liste des composants. Tous les composants se trouvent-ils à leur place (celle prévue sur la sérigraphie)? Les polarités des composants en ayant une a-t-elle bien été respectée. N'avez-vous pas fait d'erreur dans le branchement des lignes d'alimentation ? Toutes les soudures faites sont-elles « saines » ? N'avez-vous pas oublié de pont de câblage ? Si le schéma de la réalisation en cause comporte des valeurs de mesure, les éléments mesurés sur le circuit imprimé correspondent-ils à ces valeurs – on peut accepter une dérive de ±10% des dites valeurs.

La valeur d'une résistance est indiquée à l'aide d'un code de couleurs qui défini comme suit :



noir	--	0	--	--
marron	1	1	x10^{1}	±1%
rouge	2	2	x10^{2}	±2%
orange	3	3	x10^{3}	--
jaune	4	4	x10^{4}	--
vert	5	5	x10^{5}	±0,5%
bleu	6	6	x10^{6}	--
violet	7	7	--	--
gris	8	8	--	--
blanc	9	9	--	--
or	--	--	x10^{-1}	±5%
argent	--	--	x10^{-2}	±10%
rien	--	--	--	±20%

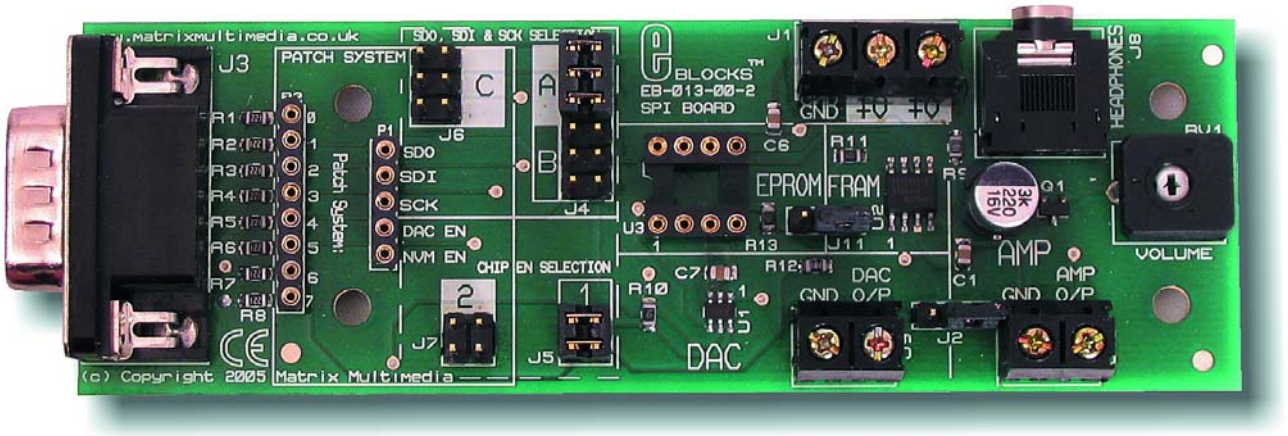
Exemples :

marron-rouge-marron-or = 120 Ω, ±5%
jaune-violet-orange-marron = 47 kΩ, ±1%

Il arrive que nous ayons à publier des corrections éventuelles concernant l'une ou l'autre réalisation, ce que nous faisons dans les plus brefs délais dans l'un des magazines publiés ultérieurement. On notera que la rubrique « le coin du lecteur » contient de temps à autre des commentaires ou/et des informations additionnelles concernant des montages publiés dans un numéro précédent.

E-blocks fait des V

John Dobson



Carte bus SPI de la famille E-blocks avec MMN et CAN

Nombre de nos lecteurs savent mettre à contribution les entrées A/N du microcontrôleur PICmicro, l'utilisation des N/A est moins courante. Ce court article va montrer comment créer à l'aide d'E-blocks, sous la houlette de Flowcode, un oscillateur sinusoïdal rudimentaire pouvant servir au test de circuits audio.

Il nous a été demandé à de nombreuses reprises un générateur sinusoïdal simple utilisable pour le test de circuits audio. Le système E-blocks comportant un CNA (**C**onvertisseur **N**umérique/**A**nalogique) pourquoi ne pas s'en servir pour réaliser un générateur de formes d'ondes à usage général ?
Il vous faut, pour commencer, décider la plage des valeurs à générer et procéder à quelques calculs sur les paramètres généraux pour voir si le projet est réalisable. La plage de sortie du CNA s'étend de 0 à 5 V en 255 pas. Il n'est pas judicieux de se rapprocher de trop près des limites définies par les rails d'alimentation au cas où l'on aurait besoin d'un peu de marge pour un tampon, raison pour laquelle nous allons nous limiter à une sortie de 4 V crête à crête au maximum. Il est possible, par logiciel, de diviser cette valeur pour travailler avec des signaux plus faibles. La fréquence doit se situer aux alentours de 1 kHz, valeur se trouvant en plein dans la plage audio. Un coup d'oeil rapide aux spécifications du CAN MAX5385 nous apprend qu'il possède un temps d'établissement de 20 µs pour une précision de sortie de la moitié du bit de poids faible. Si nous disposons de 256 échantillons par forme d'onde, cela nous donnerait une fréquence de sortie maximale de 200 Hz. Théoriquement cette fréquence n'a pas une hauteur suffisante, mais comme nous utilisons une sinusoïde dont les échantillons sont séparés par un incrément très faible, le système a toutes les chances de s'en sortir mais nous pouvons toujours nous résigner, si l'approche ne fonctionne pas comme souhaité, de réduire le nombre d'échantillons par forme d'onde.

Mettez la main sur les CNA

La bonne nouvelle : les CNA sont relativement simples à mettre en oeuvre. Vous les attaquez à l'aide d'un nombre et le niveau de la sortie analogique correspond à la tension de sortie à pleine échelle multipliée par un rapport, N, égal au nombre appliqué à l'entrée sur le nombre correspondant à la pleine échelle de l'entrée. Le graphique de la **figure 1** montre un croquis de départ reposant sur une sortie à pleine échelle de 5 V et une entrée à 8 bits (2^8 = 256 ou de 0 à 255). En réalité, la sortie du CNA se trouve à 0,9 V_{CC} seulement, ce qui signifie que l'on aura une erreur de 10% sur le niveau de tension de la vraie sortie, mais cela n'a rien de rédhibitoire.
Le microcontrôleur doit générer une série de valeur correspondant à un signal de sortie sinusoïdal. 256 pas impliquent 256 valeurs distinctes. Heureusement, les ondes sinusoïdales sont symétriques et, de plus, périodiques de sorte qu'il nous suffit de générer le quart d'une forme d'onde et d'utiliser ensuite une programmation simple en vue de réaliser une réflexion des données horizontale et verticale. Nous souhaitons une sortie de 4 V crête

...agues Flowcode prend le contrôle de CNA

à crête - pour éviter des nombres complexes il est préférable de n'utiliser que 200 des 256 bits disponibles. Ceci nous donnera pratiquement 4 V de la plage (théorique) totale de 5 V. La forme d'onde centrale est représentée en bas à gauche de la figure 1 : elle comporte 64 valeurs en *x* battant une plage numérique allant de 0 à 100, soit de l'ordre de 2 V.
En étant arrivé là, nous allons utiliser un tableur pour calculer les valeurs de sortie numériques pour les données de base. La formule d'une onde sinus est $y = \sin(x)$, formule dans laquelle *x* varie de 0 à 360 degrés ou de 0 à 2π radians. Comme nous ne savons pas ici, chez Elektor, faire travailler Excel en degrés, nous utilisons tout simplement des radians. Nous nous sommes contentés de représenter les 3 premières et les 3 dernières des 64 valeurs de base. Il vous suffit, pour obtenir les valeurs qu'il vous faut envoyer au CNA, de multiplier $\sin(x)$ par la plage de base de 100. OK, nous connaissons la théorie.

Dans la pratique

Commençons par jeter un coup d'oeil au matériel. Nous allons utiliser la **carte de bus SPI des E-blocks** que l'on trouve sur la photo en début d'article. Elle est connectée à la carte Multiprogrammateur USB dotée d'un PIC16F877. Nous aurions pu utiliser n'importe quel PIC possédant un USART. La carte SPI attaque le portC, un affichage LCD le portB et le clavier le portD.
La carte bus SPI comporte un minuscule CNA MAX5385 ainsi qu'une mémoire non volatile (NVM de type FRAM) à bus SPI de 64 Koctets. La première étape consiste à transférer les données vers la NVM. Lors d'une étape suivante du projet on pourra envisager de réaliser un portfolio de plusieurs formes d'ondes stockées dans la NVM pour ensuite les appeler indépendamment en s'aidant du clavier et de l'affichage LCD pour leur sélection.
Pour transférer les données dans la NVM, nous avons écrit un programme pour y placer les 64 valeurs de base. Il fut défini un compteur allant de 0 à 63, suivi d'une routine écrivant les différents octets à des emplacements séquentiels en NVM. Pour faire cela en Flowcode il suffit d'initialiser le bus SPI et d'écrire ensuite un programme à 64 icônes, chacune de ces icônes écrivant un octet de donnée, comme l'illustre la **figure 2**.
L'étape suivante consiste à écrire un programme distinct pour s'assurer que les valeurs de base génèrent bien le premier quart de la sinusoïde. Aussitôt dit aussitôt fait. On définit un compteur, chargé de lire successivement le contenu (la valeur) des emplacements 0 à 63, suivi d'une routine transférant les données de la NVM vers le CNA. La **figure 3** montre le Flowcode chargé de cette opération. L'oscillogramme de la **figure 4** montre la première forme d'onde en sortie pour le premier quart de cycle. Il manque quelque peu d'arrondis, mais est identifiable comme étant une quart de sinusoïde. Malheureusement

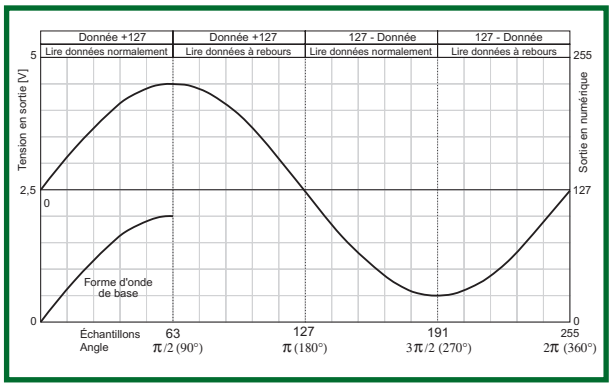


Figure 1. **Croquis de la spécification initiale.**

la durée de période pour un quart de cycle atteint 3 ms, ce qui se traduit par une fréquence finale inférieure à 100 Hz, trop faible pour ce que nous voulions faire. Nous avons heureusement un plan de secours.

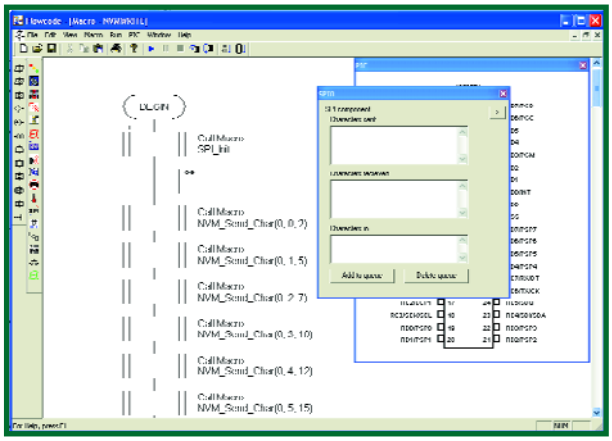


Figure 2. **Programme de gavage de la NVM.**

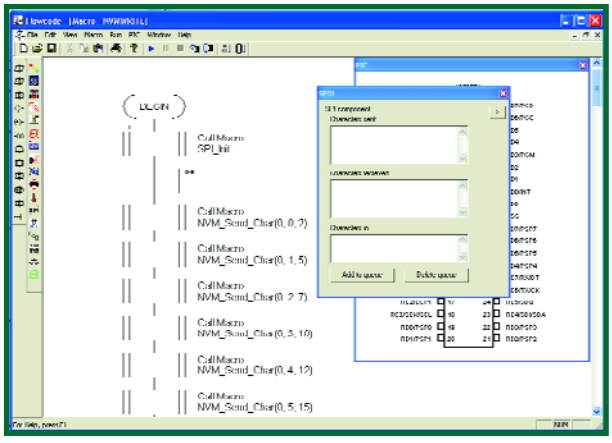


Figure 3. **Programme à 2 sous pour la saisie de données de la NVM et leur transmission vers le CNA.**

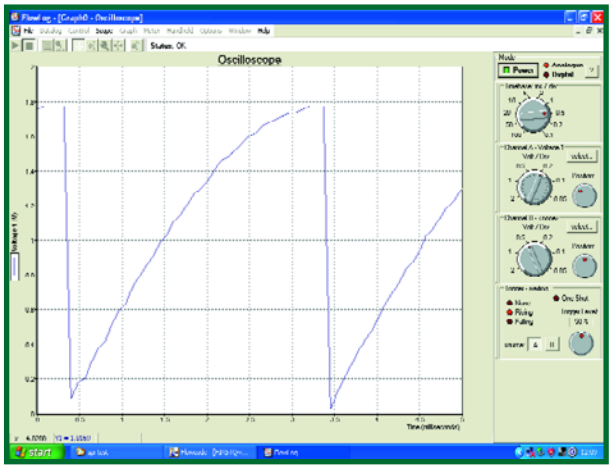


Figure 4.
Première forme d'onde en sortie montrant des valeurs de base répétitives expédiées vers la sortie.

Programmes disponibles au téléchargement

SPI memory stuffing.fcf

First quarter.fcf

Sine wave gen.fcf

SPIscreen.FCF

Numéro du fichier : 065031-11.zip

Position : MAGAZINE → Mars 2006 → 'E-blocks fait des vagues'.

Tableau 1. Valeurs en sortie

Échantillon n°.	x	sin(x)	100sin(x)
0	0,024544	0,024541	2
1	0,049087	0,049068	5
2	0,073631	0,073565	7
…	…	…	…
61	1,521711	0,998796	100
62	1,546255	0,999699	100
63	1,570798	1	100

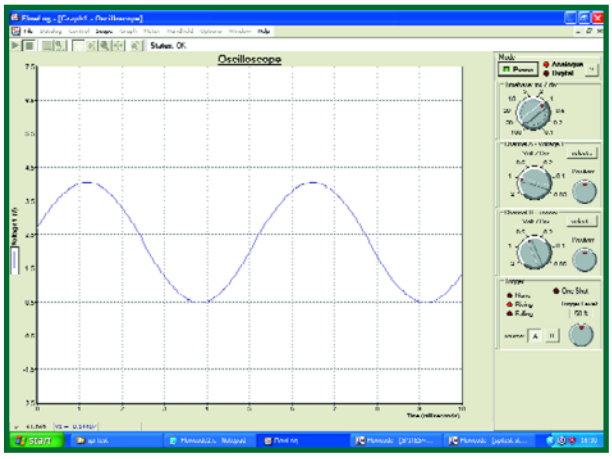


Figure 5.
Le résultat final en sortie.

Ciblons

Le processus est ralenti par le besoin de lire sériellement les données en provenance de la NVM externe. Le PIC '877 dont nous nous servons possède quant à lui 256 octets d'EEPROM en interne ! Il est facile de mettre nos données dans l'EEPROM interne du PIC en espérant que cela va accroître sensiblement la vitesse du processus.
Flowcode a été doté d'une nouvelle macro utilisateur baptisée LOADEEPROM. Elle lit la NVM externe et charge les 64 octets de donnée dans l'EEPROM interne. Ceci a impliqué une modification du programme pour une lecture en avant du premier quart et en arrière du second quart de la forme d'onde. Nous avons ensuite ajouté un offset de 127 et les données de la seconde moitié du cycle subirent le même traitement. Nous avons ensuite mesuré les résultats qui sont présentés en **figure 5**.
Le signal final possède une période de 2,5 ms, soit une fréquence de 400 Hz, et une tension crête à crête de quelque 3,5 V. La forme d'onde n'est pas trop distordue, les points de passage à zéro présentant cependant une petite discontinuité. Ceci est probablement dû aux délais du programme entre chacun des quarts de cycle de données prises en compte. On peut résoudre cette imperfection par un chargement de la totalité des 256 octets de données dans l'EEPROM interne et un démarrage de la forme d'onde à 90 degrés, endroit où la distorsion aura un effet moindre. N'hésitez pas à nous faire savoir (ainsi qu'à nos autres lecteurs) comment vous avez amené ce projet à perfection, par le biais de la rubrique E-blocks démarrée dans le Forum en ligne d'Elektor.

NdlR :
Il nous est souvent demandé quelle était la différence entre la version étudiant/particulier et la version Pro de Flowcode.

La différence majeure entre la version étudiant/particulier et la version Pro de Flowcode est que la première ne connaît pas les systèmes virtuels ; il s'agit en quelque sorte de blocs fonctionnels permettant de remplir des fonctions spécifiques telles que la communication par RS-232, par IrDA, le contrôle d'un bus SPI, la communication avec une EEPROM et bien d'autres choses encore, cette liste ne cessant de s'allonger au fur et à mesure des développements.

En outre, la licence de la version étudiant/particulier n'en autorise pas l'utilisation dans les environnements de l'enseignement et commercial.

(065031-1)

Articles précédents de cette série

- **Un meccano pour électroniciens**, n°329, novembre 2005, page 64 et suivantes
- **E-blocks et Flowcode**, n°330, décembre 2005, page 64 et suivantes
- **E-blocks & Cyberespace**, n°331, janvier 2006, page 60 et suivantes
- **E-blocks ? Passons au CAN**, n°332, février 2006, page 54 et suivantes

S'ils vous intéressent et que vous ne les possédez pas, ces articles sont tous téléchargeables depuis notre site.

Z8 Encore MC™

Kit de développement pour commande de moteur à base de microcontrôleurs de la série Z8FMC16100

L'installation de l'environnement de développement est un modèle du genre et se fait sans le moindre problème. Une fois le programme installé, on dispose de toute la documentation technique nécessaire.

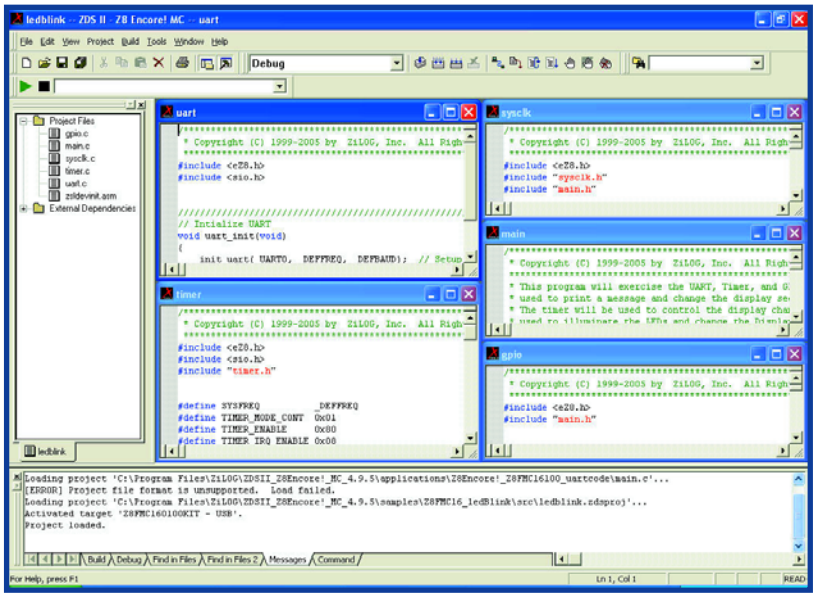
Un coup d'oeil au matériel nous montre que l'on a affaire à un ensemble gigogne, une carte-fille sur laquelle se trouve le microcontrôleur étant solidement enfichée dans une embase présente sur la carte-mère, qui est en fait la carte de puissance. Un câble plat à 6 conducteurs interconnecte la carte-fille à un boîtier baptisé « Opto-isolation Smart cable » et doté d'un beau logo « Zilog XTools », boîtier qui, si on l'ouvre, se révèle être aussi riche en électronique que la carte-fille, et qui fait office de liaison USB entre le PC et le kit de développement. Lors de la connexion du câble USB, le système s'installe de lui- même, si tant est que l'on dispose de Windows XP. Il est temps maintenant de mettre les mains à la pâte, ce n'est en effet pas tous les jours que l'on a la possibilité de piloter un moteur de 30 W pouvant tourner à 3 200 tr/mn.

Pour l'amateur d'expérience(s fortes), ce kit requiert, de disposer d'une alimentation de laboratoire capable de fournir 24 V. Bien évidemment si vous êtes un spécialiste des moteurs vous conviendrez qu'il s'agit là d'une exigence parfaitement normale dès lors que l'on veut se faire la main sur les moteurs. et par conséquent, cela ne devrait pas être un obstacle insurmontable.

Une fois l'alimentation 5 V appliquée, la LED verte d'allume, la LED jaune se met à clignoter, tout va bien jusqu'à présent. Après avoir mis la main sur une alimentation de laboratoire capable de fournir 24 V, et l'avoir coupée après l'avoir à zéro, et l'avoir branchée aux points prévus à cette intention (P5 pour la masse et P4 pour le +) on peut la mettre en fonction. Il ne doit rien se passer pour le moment. On augmente ensuite progressivement la tension de sortie de l'alimentation et au bout d'un certain temps, le moteur devrait se mettre à tourner. Un inverseur à glissière permet de mettre le moteur en route et de le couper, un autre d'en inverser le sens de rotation. Un potentiomètre permet de jouer sur la vitesse. Comme on le voit, il ne faut pas grand chose pour piloter un moteur.

À noter la présence d'une aire de prototypage où l'on pourra ajouter quelques composants additionnels.
Ce kit est une excellente entrée en matière si l'on veut s'adonner à une nouvelle discipline, celle du pilotage de moteurs et cela en s'aidant d'un microcontrôleur prévu à cet effet, le Z8FMC16100.

(067017-1)



Tour d'honneur p

30 ans après, il inspire encore des milliers

Roelf Sluman

Les microprocesseurs à huit bits appartiennent-ils au passé ou peuvent-ils encore servir à quelque chose ? Elektor a fouiné et trouvé que le bon vieux 6502 attirait encore, à l'ère des processeurs à double c?ur et du logiciel arborescent, une multitude de fans.

Dans les années 70 et 80, trois microprocesseurs dictaient leur loi au marché : le 6809 de Motorola, le Z80 de Zilog et le 6502 de MOS. Mais le plus populaire d'entre eux, et de loin, c'était le 6502. Lors de son introduction, il coûtait à peine 25 dollars et son design avancé pour l'époque a fait de lui un conquérant, puisqu'il était le cerveau d'ordinateurs individuels aussi populaires que le Commodore 64 et l'Apple II.

Quelque trente ans plus tard, le marché est dominé par des processeurs qui tournent des milliers de fois plus vite que le 6502 et pourtant… il reste quantité d'applications pour cet élégant huit-bits. Des dizaines de milliers d'enthousiastes, dans le monde entier, travaillent encore quotidiennement avec le 6502 et réalisent avec lui des choses que l'on aurait tenues pour impossibles en 1975.

La guerre des prix

Lors de son introduction en 1975, le 6502 coûtait environ 25 dollars. Il constituait donc un redoutable concurrent pour le processeur dont il était issu, le 6800 qui, lui, affichait un prix s'élevant à 179 dollars. Rien d'étonnant à ce que des fabricants tels que Apple ou Commodore aient préféré le 6502. Steve Wozniak de chez Apple avait une prédilection pour le 6800, mais c'est l'énorme différence de prix qui força la décision.

Un peu d'histoire

Alors que le 6502 fête cette année son trentième anniversaire, on se souviendra qu'à l'époque, il avait fait scandale. Ses concepteurs avaient rêvé à un autre processeur : le 6501. Mais voilà, il ressemblait comme deux gouttes d'eau au 6800 du concurrent Motorola. Rien d'étonnant, il avait été imaginé par les mêmes ingénieurs ! Immédiatement après l'introduction du 6800, un conflit opposa Motorola aux concepteurs du 6800, lequel entraîna la démission simultanée de presque tous les membres de ce groupe. Après quoi, ils furent immédiatement engagés par MOS Technology, le plus grand concurrent de Motorola dans les années 70. MOS s'était rendu compte du potentiel du 6800 et demanda à ses concepteurs de réaliser un processeur compatible broche à broche avec le 6800. Ce fut le 6501 qui était un peu moins cher que le 6800, puisqu'il n'avait occasionné que peu de coûts liés au développement.

Motorola ne s'en tint pas là et porta l'affaire devant les tribunaux. La réaction de MOS, ce fut le 6502, tout à fait identique au 6501, mais au brochage différent. Il ne pouvait donc pas s'implanter sur les platines prévues pour le 6800, précisément ce que Motorola voulait obtenir en justice.

Comme il n'y avait pas de carte susceptible de recevoir le 6502, il fallait que MOS développe quelque chose pour attirer l'attention des fournisseurs de logiciel. On vit donc apparaître le KIM-1 (cf. **figure 1**), un ordinateur à

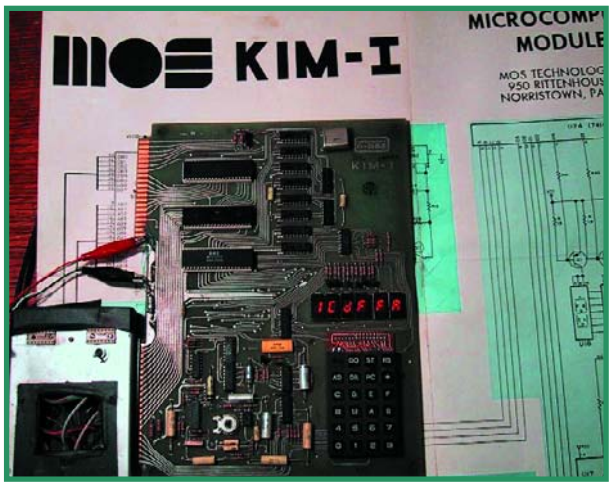


Figure 1. L'ordinateur KIM-1 développé par MOS Technology. La saisie s'opérait sur un clavier hexadécimal, sortie sur un afficheur à LED de six chiffres.

pour le

de concepteurs

une seule platine avec 1 Ko de mémoire RAM.
Très vite, les fabricants sentirent les énormes possibilités offertes par le 6502. En Amérique, il opéra une percée fulgurante avec les Atari 400 et 800, tandis qu'en Europe, Commodore volait de succès en records de vente avec le VIC-20, bientôt avec le Commodore 64 qui recelait une version améliorée du 6502, le 6510. Le Commodore 64 s'est vendu à plus de 25 millions d'exemplaires, le nombre total de processeurs 6502 vendus dans le monde est estimé à 100 millions !
Elektor a aussi publié de nombreux projets de construction basés sur le 6502. Le Junior Computer, dès 1980, a été un système d'étude particulièrement prisé. À partir de 1983, une série de carte ont permis à nos lecteurs de se fabriquer un ordinateur complet centré sur le 6502. Finalement, le résultat s'est mué, en 1985, en un système du nom d'Octopus 65.

Un peu de technique

Le 6502 est un processeur à huit bits alimenté sous 5 V et doté d'un domaine d'adresses à 16 bits, ce qui permet d'atteindre, de 0x0000 jusqu'à 0xFFFF, au maximum 64 Ko de mémoire. Il intègre 4 300 transistors. Sa fréquence d'horloge est de 1 MHz, mais comme le 6502 ne doit pas s'occuper d'instructions en microcode pour chaque code d'opération du processeur, il va aussi vite, en pratique, que le Z80 à 4 MHz qui, lui, doit traiter le microcode.

Le jeu d'instruction n'en compte pas plus de 56 et comme à l'époque la mémoire RAM était plus rapide que l'horloge du processeur, les concepteurs avaient épargné en registres internes. Le 6502 n'en a que trois, chacun de huit bits : l'accumulateur (le seul capable de réaliser des opérations arithmétiques) et les registres d'index X et Y. Le nombre relativement élevé de modes d'adressage du 6502 permet de gérer de manière efficace 64 Ko de mémoire avec des instructions à deux octets (un code d'opération et un opérande). C'est ainsi qu'il suffit de 19 octets de code pour remplir 64 Ko de mémoire. Si vous ne voyez pas comment procéder, envoyez donc un courriel à l'auteur : rsluman@gmail.com pour le découvrir.
Il existe un certain nombre de variantes du 6502, avec plus ou moins d'options. Les plus célèbres sont le 6507, celui utilisé dans l'ordinateur de jeu Atari VCS 2600, et le 6510 dans le Commodore 64. Celui-ci a été le tout premier ordinateur individuel de 64 Ko de mémoire à offrir l'astuce de pouvoir la commuter à son gré d'une banque à l'autre (sous la mémoire dans laquelle s'opérait la gestion des ports d'E/S se trouvait par exemple une partition de mémoire RAM que l'on pouvait rendre accessible pour y loger provisoirement des données puis la recacher pour récupérer l'accès aux ports d'E/S).

Travailler avec le 6502

La meilleure façon de faire connaissance avec le 6502, c'est de le (re)mettre à l'ouvrage.

Par simulateur

Le plus simple, c'est d'utiliser un simulateur, un logiciel qui imite le comportement du 6502. Un exemple de très bon programme dans ce domaine, c'est le *6502 Simulator* disponible gratuitement au téléchargement entre autres sur http://home.pacbell.net/michal_k/6502.html (**figure 2**).

Par émulateur

Un émulateur est un logiciel pour ordinateur qui crée une machine virtuelle, un processeur dans l'ordinateur. Comme il s'agit d'une réplique logicielle du processeur authentique, tout programme écrit pour ce processeur fonctionnera également sur l'émulateur. Même les défauts éventuels du matériel d'origine sont reproduits sur l'émulateur !
Pratiquement tous les processeurs des années 70 et 80 ont été émulés. Dans bien des cas, les concepteurs du

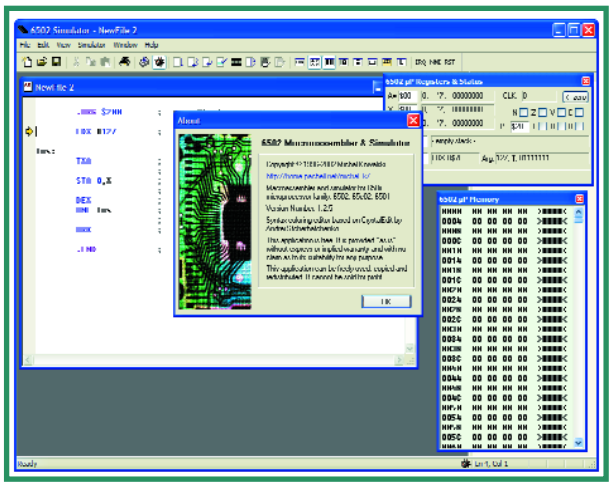


Figure 2.
L'idéal pour ceux qui veulent travailler avec le 6502 : le macro-assembleur et le simulateur de Michal Kowalski.

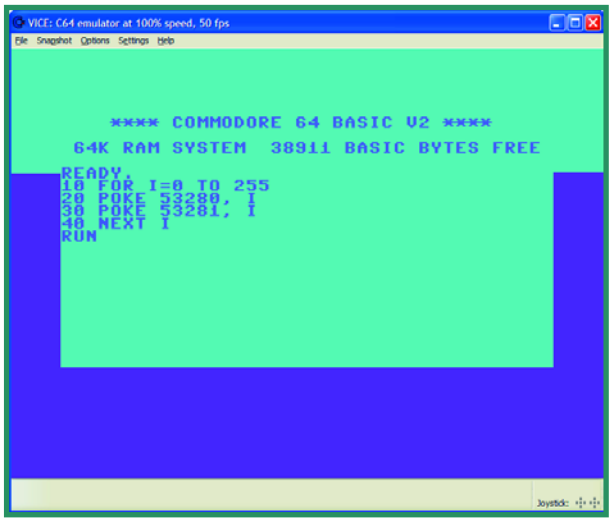


Figure 3. Une vieille connaissance, l'écran du Commodore 64, se retrouve comme émulateur sur un PC moderne.

logiciel de ces ordinateurs populaires ont libéré les droits aux profit des utilisateurs de ces émulateurs. S'il vous prend encore de temps en temps la nostalgie des heures passées devant votre Apple, Atari ou Commodore, vous pourrez ainsi les faire revivre !

L'un des programmes d'émulation les plus connus est VICE (un sigle tiré – légèrement par les cheveux — de Versatlle Commodore Emulateur). On le trouve sur www.viceteam.org. Il est destiné à émuler tous les ordinateurs individuels connus de Commodore, et parmi ceux-ci le 64. L'émulation par VICE est à ce point parfaite que pratiquement tout logiciel jamais écrit pour le Commodore 64 tourne sans difficulté sous VICE.

Dès le lancement de VICE apparaît l'écran bleu (**figure 3**) bien connu et vous avez sous les doigts un vrai Commodore 64. Même le clavier se réarrange différemment pour correspondre à celui du Commodore 64.

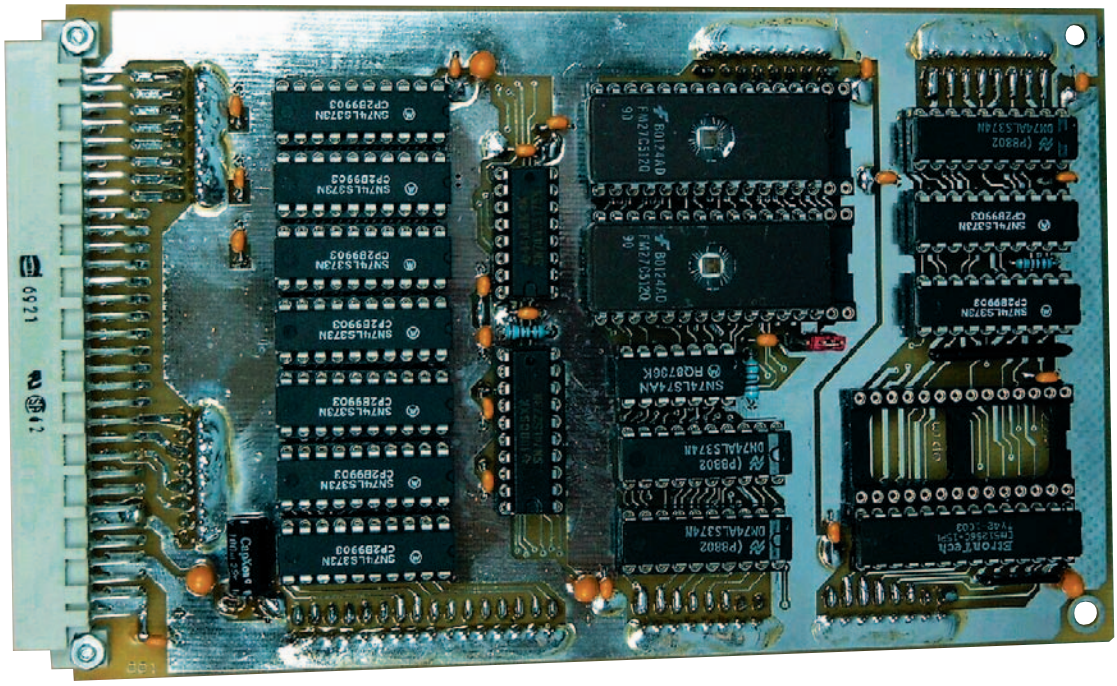


Figure 4. Une des deux platines qui composent le 6502 à composants discrets de Dieter Müller. (source : le site web Dieter's Hobby Projects)

Figure 5. Sur FPGA Arcade, on trouve différents façons d'imiter le 6502 en FPGA pour le plaisir du jeu.

Le lecteur de disquettes et le deck à cassettes avec éventuellement le chargeur turbo sont émulés à 100 %.

Par contrefaçon

Vous pouvez aussi « imiter » un 6502 à l'aide de composants discrets comme des puces 7400, RAM et EPROM. C'est ce que Dieter Müller a fait avec d'anciennes pièces qu'il avait chez lui. Il a pratiquement réussi en mettant en jeu 40 puces réparties sur deux platines (**figure 4**). La concession principale porte sur la fréquence de l'horloge, mais son processeur discret reconnaît toutes les instructions du 6502 (cf. http://freenet.de/dieter.02/m02.htm).
Une autre solution consiste à démarrer partir d'un FPGA et lui faire apprendre toutes les fonctions du 6502. Dans ce genre de FPGA, on trouve toujours quantité de blocs logiques programmables et dans un exemplaire

moderne, on peut aisément introduire la circuiterie complète d'un 6502, et même en loger plusieurs dans la même puce ! Il existe une organisation qui s'occupe de développer en FPGA différentes unités centrales. Elle s'appelle Opencores (http://opencores.nnytech.net). Le code du 6502, après recherche, se trouve sous le nom de T65. Il est également disponible sur http://www.fpgaarcade.com/ où l'on trouve aussi plusieurs variantes de jeux pour le 6502 (**figure 5**).

Applications

Comme le 6502 est un processeur très doué et bon marché et qu'il se combine aisément à d'autres matériels, il reste un composant favori pour de nombreux projets personnels. On trouve sur Internet de nombreuses réalisations qui utilisent le 6502. La plupart concernent un ordinateur construit à partir d'un 6502, mais on y découvre aussi des applications qui manifestent une bonne dose d'inspiration.

Le 6502 visite le casino

Vous pouvez gagner beaucoup d'argent avec le 6502, c'est ce que deux étudiants californiens, sous le pseudo de Eudaemons, ont démontré. Ils ont fabriqué deux circuits basés sur le 6502 pour les installer dans une de leurs chaussures. Ils se sont alors présentés au casino et ont choisi une table de roulette. Le premier promenait son soulier sous le trajet du plateau tournant de la machine, après quoi le processeur 6502, muni d'un ingénieux algorithme, calculait la fin de la trajectoire de la bille, dans quel trou elle allait tomber. L'information était alors immédiatement transmise à la chaussure de l'autre étudiant sous la forme de vibrations qui lui indiquaient alors dans quelle case miser.

Procurez des yeux à votre robot

Mike Nabereznv a utilisé une combinaison de quatre capteurs d'infrarouge GP2D02 de Sharp pour doter d'yeux son robot. Le programme se contente d'interroger continuellement chacun des quatre capteurs (une haute valeur signifie « près » et une petite, « loin ») et à passer ces informations au robot.
Sur Internet (entrée via www.6502.org) vous pourrez trouver nombre de projets semblables basés sur le 6502.

(050316-I)

2 Ko de mémoire ? Plus qu'assez !

Le premier ordinateur équipé du 6502 fut le célèbre ordinateur de jeu Atari VCS 2600. Bien que le 6502 soit le processeur le meilleur marché, Atari a encore comprimé le prix en commandant une version spéciale du 6502, le 6507 qui ne pouvait adresser que 8 Ko au lieu de 64 Ko. Aucun souci à se faire, selon les concepteurs de l'Atari 2600. En fin de compte, la mémoire était tellement chère que les jeux sur cet ordinateur n'ont jamais dépassé le taille de 2 Ko !

Où trouver encore un 6502 ?

Gunther Ewald

Difficile à croire, pourtant on produit encore aujourd'hui des microcontrôleurs avec le coeur du 6502.

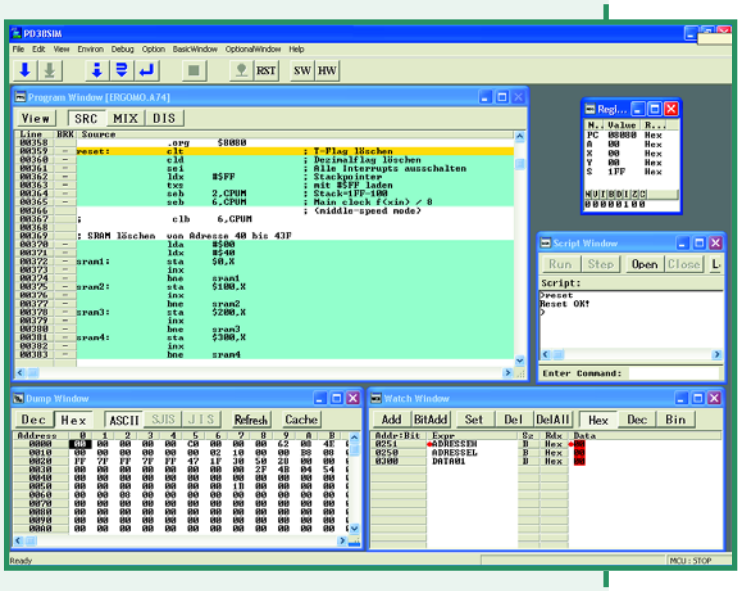
C'est ainsi que Micronas et Renesas fabriquent des microcontrôleur peu onéreux pour des tâches simples.

Vous verrez par exemple chez Micronas le WDC65C816 utilisé principalement sur automobile : http://www.micronas.com/products/by_function/cdc_1607f-e/product_information/index.html

Renesas utilise le coeur du 6502 dans toute la famille 740. Il est même compatible avec les codes d'instruction d'origine, et comporte encore quelques suppléments.

Malheureusement, leur nombre a diminué après la conversion aux produits sans plomb, mais on trouve encore chez ce fabricant 37 dérivés munis du processeur 6502 :

www.renesas.com/fmwk.jsp?cnt=740_family_landing.jsp&fp=/products/mpumcu/740_family/



En définitive, ces produits ne conviennent pas à des applications d'amateur parce qu'il faut disposer d'un émulateur du microcontrôleur. Le fabricant fournit un simulateur gracieusement pour une durée d'utilisation de quatre mois. Il offre la simulation des E/S, des interruptions, la mesure du cycle d'exécution, un moniteur afficheur de RAM et une mesure de la couverture. Ce programme est disponible sur la page d'accueil de Renesas.

Chargeur en voyage

Énergie gratuite sur les cimes

Karel Walraven

Si vous choisissez de passer des vacances dans des contrées (relativement) inhospitalières, une opération aussi simple que la (re)charge d'accus (pour un caméscope numérique par exemple) peut devenir un vrai casse-tête. Impossible en effet de trouver une prise secteur où que ce soit ! Ceci n'a pas de quoi rebuter un concepteur de l'équipe d'Elektor, qui résoudra ce problème à l'aide d'un mini-panneau solaire et d'un chargeur d'accus Li-Ion de fabrication-maison.

Ayant besoin de vacances, je proposais à la direction de m'absenter 6 mois. Tous comptes faits, cela se transforma en un mois à grenouiller aux alentours de l'Annapurna au Népal. Pour ma propre alimentation en énergie, j'emportais un petit panneau solaire et un petit chargeur Li-Ion réalisé en technologie discrète. N'ayant pas le temps pour le concevoir moi-même, je choisis, pourquoi vouloir réinventer la roue, d'utiliser un schéma d'une firme chinoise fort connue.

Je ne vais pas vous apprendre que le chargement de cellules Li-Ion est l'enfance de l'art : vous réalisez une alimentation bien régulée que vous verrouillez à 4,1 ou 4,2 V très précisément (toujours vérifier la valeur en vous aidant des données du fabricant), la dotez d'une limitation de courant de façon à ce que les courants ne puissent pas devenir trop importants. Plus la cellule se (re)charge, plus l'intensité du courant diminue (cf. la courbe de caractéristique de la **figure 1**). Lorsque le courant a chuté à 1/20 voire moins de son intensité en début de charge, on peut supposer que la cellule est pleine. N'ayons pas peur de le souligner : la tension de charge est critique, la tolérance admise est de 1% au maximum, ce qui n'est pas beaucoup, 1% de 4,2 V ne représente que 42 mV ! Il est impératif pour cette raison de toujours vérifier la tension de sortie pour s'assurer qu'elle reste à l'intérieur de la tolérance.

L'intérêt premier du schéma en question (**figure 2**) est qu'il est facile à reproduire vu qu'il ne requiert pas de composant exotique. Le circuit de référence de tension, le TL431 se trouve encore facilement. On utilise en outre quelques transistors ordinaires et un transistor de puissance que l'on peut d'ailleurs remplacer par pratiquement n'importe quoi. En ce qui concerne la diode Schottky, tout type capable de supporter 1 A fait l'affaire, comme une classique 1N4001 par exemple. Pas de problème si la tension d'entrée est un peu trop haute, à un moment donné il faudra bien évidemment doter T1 d'un radiateur.

Philosophie du concept

Il est toujours intéressant de jeter un coup d'oeil critique sur un concept qui vous est étranger pour en découvrir la philosophie de concept.

Tension de référence

On constate tout d'abord l'utilisation d'une référence de tension stable et réputée, IC2, un TL431 de Texas Instruments. Un choix très judicieux car si l'on veut garantir une tension de charge précise à 1%, une référence d'excellente qualité est un préliminaire indéniablement important. De manière à disposer d'un courant supérieur aux quelques milliampères dont est capable ce circuit intégré, il est doté en aval d'un émetteur-suiveur, T4. Normalement ceci se traduit par une diminution de quelque 0,6 V de la tension de sortie, la stabilité en devenant catastrophique; cependant, grâce à la contre-réaction (R21/R23) en aval du transistor, cela n'est pas le cas. Le circuit intégré modifie sa tension zener de façon telle qu'il y ait 2,5 V sur son entrée *adjust* (ADJ). Ceci donne une tension de référence de 3,3 V utilisée un peu partout sur le circuit. Il est important, pour le fabricant du chargeur qu'il soit possible, en cours de fabrication, de modifier la valeur de R21 ou R23 au cas où la tension du TL431 dérivait de façon trop importante.

S'il devait apparaître au cours d'un contrôle que vous aussi vous ayez un exemplaire à la tolérance trop importante, corrigez la tension en prenant une autre résistance en parallèle sur R21 ou R23. Ceci est une méthode pratique puisque l'on peut, par « *trial and error* » (essai et erreur) essayer à chaque fois une nouvelle valeur jusqu'à ce que la tension ait la valeur requise, la résistance pouvant alors être soudée définitivement en place.

Le choix d'une tension de référence de 3,3 V est en effet arbitraire. On aurait pu la prendre un peu plus faible ou un peu plus élevée. Pas trop élevée car dans ce cas-là la marge de jeu du TL431 devient trop faible. Plus important encore, la tension de référence doit se situer dans la plage de tension d'entrée en mode commun de l'amplificateur opérationnel utilisé. Nous y reviendrons lors du choix de l'ampli op. La prendre trop faible (2,5 V par exemple, la tension de référence sans diviseur) n'est pas souhaitable non plus vu que cela complique le pilotage de la LED D1 et qu'en outre les tolérances aux entrées des amplis op (l'offset) prennent de plus en plus d'importance.

La partie de régulation

Passons au circuit de régulation proprement dit. Nous voulons :

- Une tension de sortie stable
- Une limitation de courant

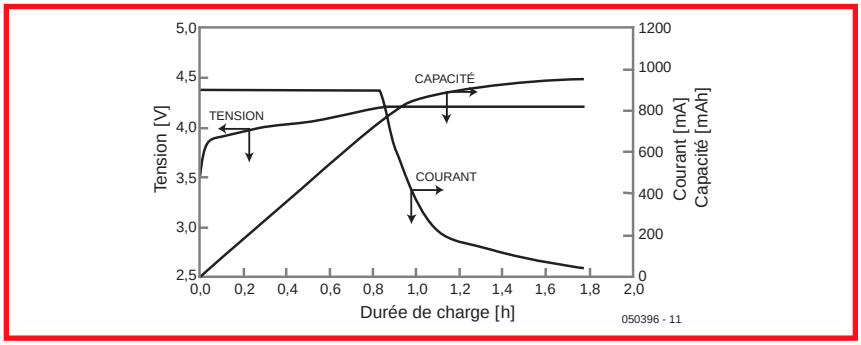


Figure 1. Caractéristique de charge d'une cellule Li-Ion (source : Panasonic).

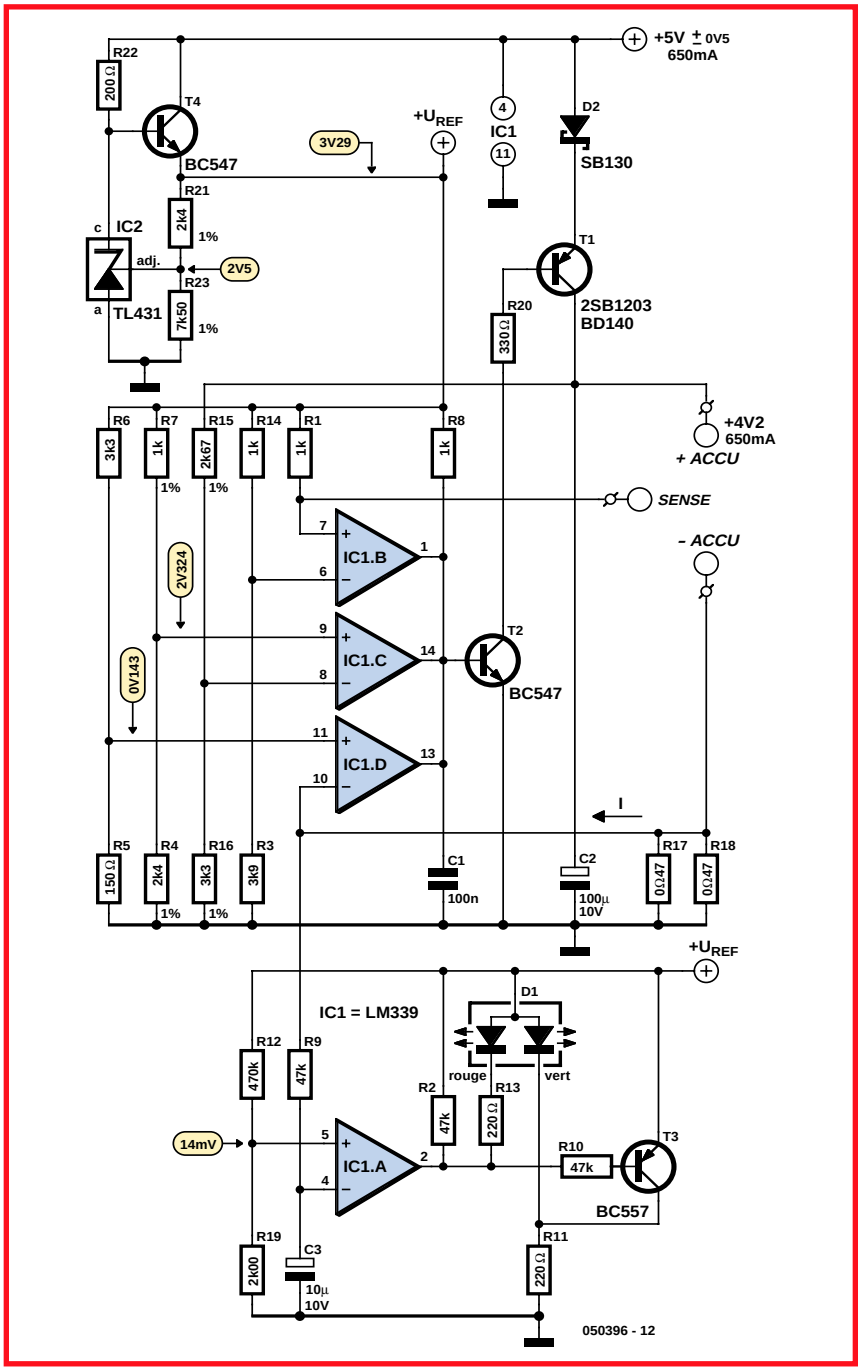


Figure 2. Le schéma du chargeur fait appel à des composants standard, se passant de circuit intégré de charge spécial.

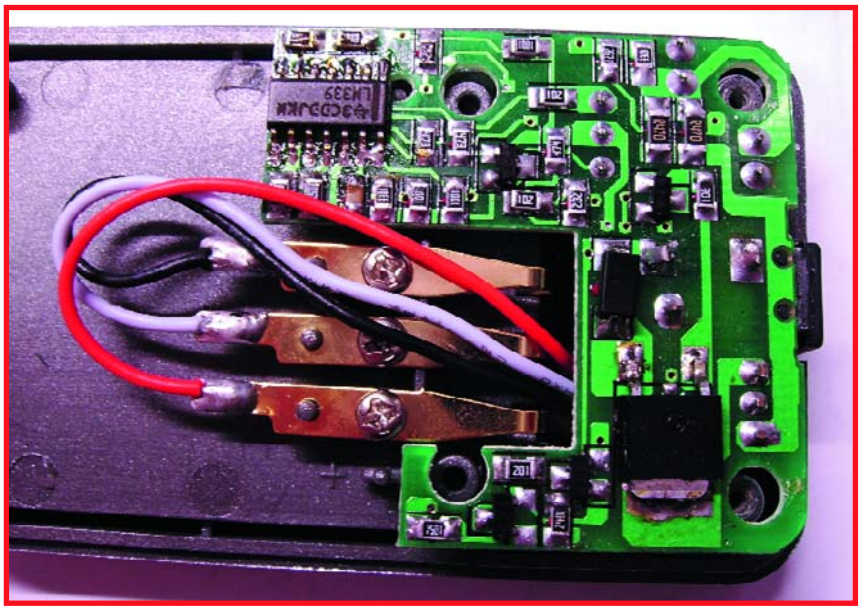


Figure 3. La platine du circuit de charge : on découvre à gauche es contacts à ressort pour le pack d'accus.

- Une surveillance de l'accu (température)
- Une indication claire plein/pas plein

Les 3 premiers points ont tous une influence sur la tension de sortie. Nous pouvons diminuer le courant en réduisant la tension de sortie, et si la température de l'accu devient trop élevée, il nous faut là encore réduire la charge en abaissant à nouveau la tension de sortie. Nos optons pour cette raison pour une topologie à base de 3 amplis op (IC1.B, IC1.C ou IC1.D) qui peuvent influer simultanément sur une sortie. Cela est possible en utilisant une sorte de porte OU. Le schéma montre que les sorties sont purement et simplement interconnectées. Il va sans dire que dans des conditions normales cela risquerait de poser des problèmes lorsque l'un des amplis op tentait d'augmenter la tension de sortie alors que l'autre essaie de la diminuer. Il est possible de résoudre ce problème par la prise de diodes ou de transistors en aval des amplis op. Nous avons opté pour une solution plus élégante consistant à utiliser un ampli op doté de sorties à collecteur ouvert. De ce fait chacun des amplis op peut uniquement abaisser la sortie commune. C'est la résistance R8 qui est chargée du rehaussement. Elle fournir le courant de base à T2 qui, à son tour, entraîne la mise en conduction de T1 de sorte que le plus de l'accumulateur se trouve forcé à la tension d'alimentation. Nous avons ainsi un système dans lequel chaque paramètre (tension, courant et température) possède son propre ampli op capable d'abaisser la tension de sortie.

La régulation de courant : On peut admettre, comme approximation, que le courant de charge maximum pour un accu Li-Ion standard se situe aux alentours de 0,7 à 1 fois sa capacité. Le maximum que ce chargeur puisse fournir est 0,65 A; il convient de ce fait à des accus d'une capacité minimum de 0,65 Ah (650 mAh).
Dans le cas d'un accu vide, le courant a tendance à dépasser les 0,65 A évoqués plus haut. Il est facile de mesurer le courant en prenant une petite résistance en série. Dans le cas présent ce sont les résistances R17 et R18 (2 résistances en raison de l'intensité du courant et de la puissance). Le courant produit une tension aux bornes de ces résistances, tension que l'ampli op IC1.D compare à la tension de référence divisée. Lorsque le courant de charge produit une tension plus élevée, IC1.D entre en scène en abaissant la sortie commune. Ceci produit une diminution de la tension de charge de l'accu, chute qui se traduit par une diminution du courant de charge à la valeur requise.
Une fois l'accu plein, la tension dépasserait, en l'absence de régulation introduite par IC1.C, les 4,2 V fatidiques. La tension de référence divisée (R7/R4) est comparée à la tension d'accu divisée (R15/R16) et si nécessaire, IC1.C abaisse la sortie commune.
En ce qui concerne le suivi de la température faite sous la houlette de IC1.B, le principe est similaire. L'accu est doté d'une NTC qui voit sa résistance diminuer lorsque la température augmente. De pair avec R1, cette NTC constitue un diviseur de tension, IC1.B comparant cette tension avec la tension de référence divisée (R14/R3) et freine le processus de charge lorsque la température augmente de trop. Cette protection thermique n'est pas infaillible, en cas de mauvais contact ou de rupture de ligne, la charge se poursuit tout simplement.
S'il vous est arrivé de monter, en aval d'un ampli op, une paire de transistors (amplifiant tous les deux !), vous savez que cela ne peut que poser des problèmes. On est en effet certain d'une mise en oscillation. Ce risque est éliminé ici par un ralentissement de la régulation par la mise en place d'un gros condensateur, C1. Un condensateur plaqué: à la sortie, C2, se charge, épaulé par le diviseur de tension R15/R16 (qui introduit une charge minimale), de la stabilité de l'ensemble à l'état non chargé.

Indication

Nous n'avons pas encore évoqué un dernier sous-ensemble du circuit : l'information par IC1.A. Cette électronique a une fonction qui bien que secondaire n'en est pas moins importante. L'utilisateur aime en effet savoir où il en est. L'attrait de ce dispositif indicateur est qu'il réagit au courant de charge. On est certain ainsi qu'il y a effectivement charge et cela évite les erreurs d'absence d'adaptateur secteur connecté au système ou d'un mauvais contact au niveau de l'accu.
Une nouvelle fois, on a division de la référence, cette fois par le biais de R12/R19, vers de l'ordre de 14 mV. Si la chute de tension due au courant de charge est, aux bornes de la paire R17/R18 plus faible (accu absent ou accu plein), le transistor de sortie interne de IC1.A bloquera vers la masse. La LED rouge D1 ne s'allume pas, la verte s'allumant elle vu qu'elle se trouve à 3,3 V et qu'elle est reliée à la masse au travers d'une résistance de 220 Ω.
Si on a la situation inverse et qu'il circule un courant de charge suffisant, IC1.A met sa sortie au niveau bas. La LED rouge s'allume alors, le transistor T3 entrant en outre en conduction. Ce dernier court-circuite la LED verte qui reste par conséquent éteinte. D1 est une LED bicolore de sorte que l'information qu'elle fournit est très claire.

Avez-vous pressenti les conséquences du choix d'une référence de tension de 14 mV pour IC1.A. Le plein courant de charge de 650 mA produit une chute de tension de 153 mV sur R18/R17. Ces 14 mV représentent environ 1/11ème, de sorte que dès que le courant de charge est tombé au 1/11ème de l'intensité maximale, l'accu est considéré comme étant plein. Le processus de charge se poursuit tout simplement, la LED se contente d'indiquer que l'accu est plein.

Nous pourrions ajouter un étage additionnel à l'électronique : lorsqu'un accu est déchargé au point que sa tension est inférieure à 2,9 V, il est en effet préférable de limiter le courant de charge au 1/10ème de sa capacité. Cette option est facile à réaliser par l'ajout d'un ampli op supplémentaire et la prise d'une résistance en parallèle sur R5, ce qui fait que la tension de référence divisée passe, pour la limitation de courant, à 14 mV. Le fabricant ne l'a pas prévu, voulant sans doute ne pas ajouter de circuit intégré additionnel.

Choix des composants

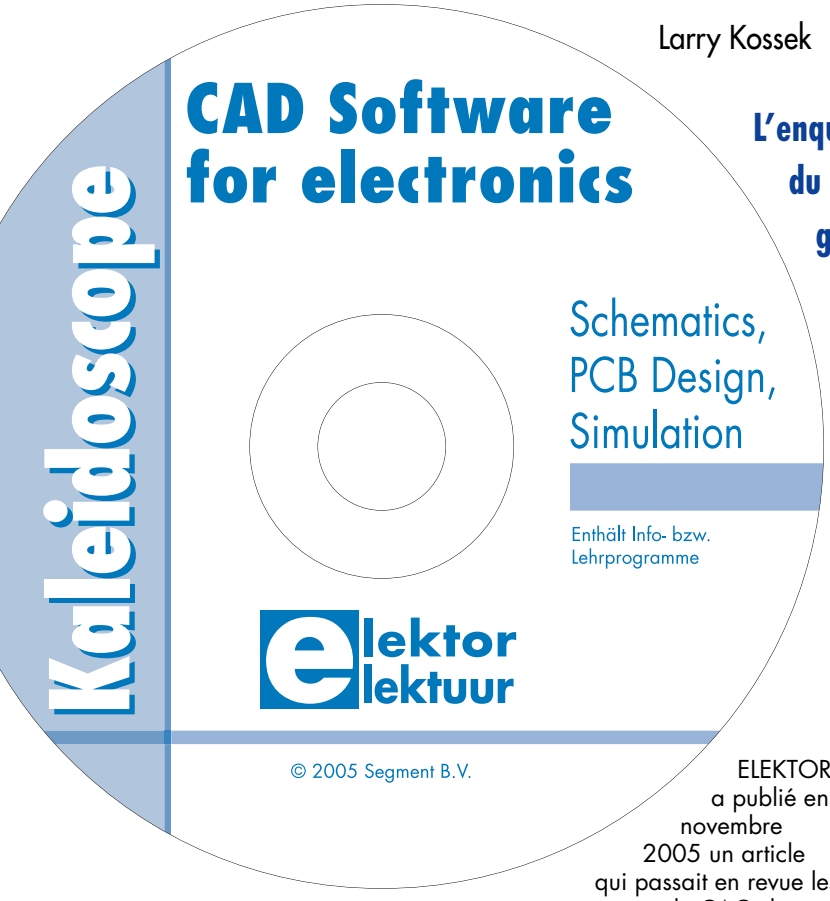
Un mot quant au choix de l'ampli op. On a, lors de la conception d'un circuit, le choix entre plusieurs centaines de composants de ce type. Cela ne fait que compliquer les choses. Nous pouvons utiliser ici un ampli op standard sans caractéristiques spéciales (précision, plage de température, bruit, vitesse, etc.). Examinons-en 2 aspects : La plage en *mode commun* : il est important, lors du choix d'un ampli op, que toutes les tensions pouvant être appliquées à l'entrée restent dans la plage de ce que l'on appelle le mode commun, soit encore d'opter pour un ampli op pouvant traiter les tensions pouvant se présenter. La fiche de caractéristiques nous apprend que dans le cas d'un LM339, zéro volt constitue la limite inférieure, la tension d'alimentation diminuée de 1,5 V représentant elle la limite supérieure. Si nous avions opté pour un LM741, la limite inférieure aurait été de 1,5 V de sorte que des tensions telles que 14 mV n'auraient pas été acceptables. Il est crucial, pour ce projet, d'opter pour un ampli op capable de travailler correctement à des tensions proches de zéro volt.
La *tension d'offset* : La fiche de caractéristiques nous apprend que la tension d'offset de l'entrée est de 2 mV. (Le fabricant propose cet ampli op sous différentes versions aux valeurs d'offset de plus en plus petites, mais les meilleures spécifications se paient). Ceci signifie que le circuit a une tolérance de 2 mV environ à l'entrée. On pense (espère) que l'indication de courant s'active à 14 mV, mais en fait elle peut le devenir à 16 ou 12 mV. Ceci représente une erreur supérieure à 10%, appréciable donc ! Si cela n'a pas de conséquence pour l'indication de courant, nous ne pouvons pas accepter une telle tolérance dans le cas de la tension de charge ! Ceci explique que la valeur de la référence pour la tension de charge choisie soit bien plus élevée, 2,4 V environ. Une dérive de 2 mV sur une tension de 2 235 mV représente de l'ordre de 1‰ (1 pour mille), étant de ce fait négligeable.

(050396-1)

En quête de votre

Un électronicien gâté est un électro

Larry Kossek



L'enquête ouverte sur le site d'Elektor à la parution du numéro de novembre, avec son fameux DVD gratuit, a permis à de nombreux lecteurs de nous donner leur avis sur cette opération exceptionnelle et surtout sur le contenu du DVD. Merci à tous ceux qui ont répondu à cette enquête dont voici les résultats les plus marquants.

ELEKTOR a publié en novembre 2005 un article qui passait en revue les programmes de CAO de schémas et de platines dont l'intérêt principal résidait dans le fait que tous les programmes étaient réunis sur un DVD joint gratuitement au numéro. Un peu plus tard nous avons ouvert sur le site internet www.elektor.fr une enquête pour obtenir l'avis de nos lecteurs. Les réponses se sont un peu fait attendre, car il y avait du pain sur la planche, mais nous avons fini par réunir des informations suffisamment intéressantes pour en faire profiter tout le monde.

Cool

Les commentaires varient assez peu, on devine surtout l'âge de leur auteur : de « *remarquable tour d'horizon* » ou « *excellente initiative* » à « *trop cool* » en passant par « *superbement génial* » et « *la classe, merci* ». La plupart des réactions étaient gratifiantes. Le DVD gratuit a été perçu comme un service rendu au lecteur : « *Quel gain de temps ! Je n'ai pas à chercher moi-même.* » Ou « *Sans prétendre à l'exhaustivité, ce DVD réunit une belle collection de logiciels à tester.* »
Qu'Elektor offre ainsi un DVD gratuit a donc franchement surpris plus d'un lecteur. « *Une bonne vue du marché en quelques clics. On voit bien que finalement Elektor connaît la musique.* » Certains avouent qu'ils n'ont acheté ce numéro que pour le DVD. Et ils en ont eu pour leur argent. « A *lui seul le DVD valait largement le prix du magazine. Je vais me lancer dans un essai comparatif. J'y verrai plus clair après pour choisir le programme sur lequel je me perfectionnerai* ». Le temps, le temps, le temps, si précieux et si élastique : « *Je n'ai qu'une connexion analogique à l'internet. Avec le DVD je n'aurai pas à me taper des heures de téléchargement...* »

Les étudiants ont particulièrement apprécié l'offre. Leurs professeurs aussi. Certains nous ont signalé qu'ils utiliseraient « *cette collection pour la présenter à leurs élèves ou leurs étudiants.* » Que « *l'un des programmes au moins soit aussi utilisable sous Linux* » n'a pas échappé à l'attention de ceux qui ont essayé KiCAD, le seul programme qui ne fonctionne pas seulement sous Windows. Idem pour les utilisateurs d'ordinateurs Apple : « *Incroyable, il y a même un programme pour MAC sur ce DVD !* »

> **"J'utilise PCad pour mon travail et je trouve ça nul"**

Bref, il n'y a pas de mal à se faire du bien comme l'a résumé un de nos lecteurs épicuriens : « *C'est ainsi que ma vie d'électronicien me plaît !* »

Le fait de pouvoir comparer des programmes aura-t-il finalement eu sur l'avis des utilisateurs l'impact que l'on imaginait ? On peut en douter. Avant que quelqu'un change d'avis, il en faut davantage. L'alchimie qui fait que l'on se sent bien dans un programme et mal à l'aise avec un autre échappe en partie aux critères rationnels C'est du moins ce que l'on peut déduire de commentaires

e avis

nicien satisfait

> **Je fais des schémas détaillés et j'aimais bien Workbench que je connais comme ma poche; pour moi tous ces programmes sur le DVD sont bien compliqués, mais j'en essaye un chaque soir. C'est loin de marcher à tous les coups, ne serait-ce que pour faire apparaître tout bêtement une résistance à l'écran, mais je m'accroche. "**

tels que : "*Intéressant! Enrichissant! Mais je reste un fervent de [xxx]! Un tel choix engendre la confusion. Je ne doute pas qu'il existe des programmes meilleurs que celui-là, mais je ne vois pas ce que je gagnerais à en changer.*" N'est-il pas admirable finalement que notre DVD, au lieu de faire changer d'avis, ait contribué à renforcer des fidélités inébranlables ?
Parmi les effets inattendus, un autre lecteur nous signale que c'est par le DVD que lui est venue l'idée d'utiliser un programme de dessin de faces avant d'appareils. Après avoir essayé *Frontdesigner* il a acheté ce programme et s'en sert régulièrement.

L'aigle

Eagle est le programme le plus populaire, toutes nations confondues. C'est le programme le plus apprécié sur le DVD mais aussi le plus utilisé. Nos lecteurs anglais ont plébiscité KiCAD mais nous avons de bonnes raisons de penser qu'il y a eu bourrage d'urnes, car dans les forums consacrés à ce programme, les fans s'étaient donné le mot. C'est de bonne guerre.
Le **tableau** et les graphiques des **figures 1 et 2** donnent une vue d'ensemble assez complète des résultats. Ce qui n'y apparaît malheureusement pas, c'est l'extraordinaire diversité des programmes cités par nos lecteurs dans leurs réponses à l'enquête. La résolution des graphiques est insuffisante pour la myriade de logiciels qui se disputent vos faveurs et que nous étions loins de connaître tous.
Une tendance très claire se dessine toutes langues confondues : environ 65% des lecteurs d'Elektor utilisent leur programme préféré à des fins privées. Plusieurs utilisateurs nous ont signalé que le programme employé à la maison n'est pas le même que celui du travail. Le critère de choix le plus cité est le prix du logiciel qui doit rester raisonnable. La recommandation faite par un ami ou un collègue vient en deuxième position. La facilité d'accès d'un logiciel est le troisième critère de choix. Plus on apprend vite à s'en servir, plus le programme est apprécié.

La plupart des participants à l'enquête se sont dits satisfaits, voire très satisfaits de leur programme. Seuls 10 à 18% se disent conscients de l'existence probables de meilleures solutions, mais ils n'ont pas encore réussi à mettre la main dessus. On peut espérer que ces personnes-là auront trouvé leur bonheur sur le DVD du numéro de novembre d'Elektor.

Remarques & suggestions

Le but visé n'était pas l'exhaustivité. Nous sommes conscients du fait qu'il manquait quelques programmes intéressants. D'après nos lecteurs ce sont **Altium – Protel**, **AutoTRAX EDA**, **DipTrace**, **DouglasCAD**, **Express PCB**, **PADS**, **SupermaxECAD**, **TINA simulator**, **Virtual Breadboard** parmi les plus demandés. Un lecteur regrette par exemple l'absence de **Scooter**, une autre la version à 100 € de **BAE**. L'absence de logiciel de simulation n'est pas passée inaperçue non plus, mais ce n'était pas le sujet. Peut-être une idée pour un prochain DVD...

Beaucoup de lecteurs bons joueurs signalent aussi qu'ils ont trouvé sur le DVD des programmes dont ils ignoraient jusqu'à l'existence.

Un père nous dit sa déception de n'avoir pas trouvé de programme adopté aux besoins et aux capacités de son fils de 12 ans, débutant passionné d'électronique. Une idée à transmettre aux éditeurs de logiciels.
Un des fans de KiCAD demande à Elektor de proposer une série d'initiation à son programme favori. Bonne idée. Il n'y a plus qu'à l'écrire. Nous cherchons un auteur... ça pourrait être vous, non ?

	DE	NL	UK	FR
Eagle	43%	43%	17%	21%
Pas de réponse	17%	13%	7%	21%
SPlan & SPrintLayout	14%	8%		7%
Target 3001!	9%	3%		5%
OrCAD	4%	8%	11%	11%
Proteus	2%	3%	16%	9%
KiCAD	0%		24%	4%
Layo PCB		7%		4%
Design Suite		4%		

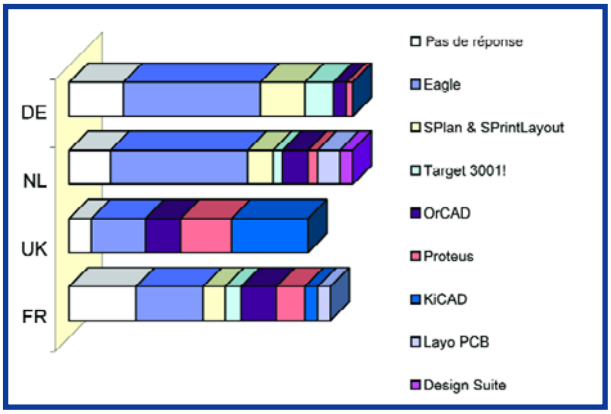


Figure 1.
Les programmes de CAO les plus souvent cités par les lecteurs d'Elektor dans les quatre zones linguistiques européennes.

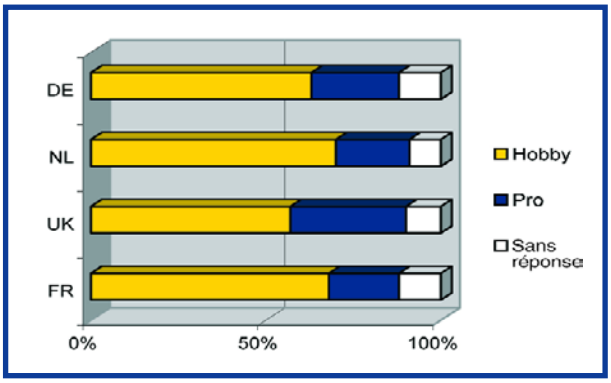


Figure 2.
Environ un tiers des électroniciens qui ont participé à l'enquête utilisent leur programme de CAO pour leur travail.

Certains programmes suscitent des critiques, parfois vives. Quelqu'un affirme : « *Avec ses nouvelles versions, Protel passe visiblement à côté des besoins pratiques.* »
Un autre lecteur trouve « *Boardmaker Win trop cher pour les particuliers !* » Mais le plus souvent, c'est un concert de louanges pour le programme favori (par exemple EASY PC) de celui qui écrit.
C'est aussi le cas de SPlan et Sprint Layout, très cotés. « *C'est SprintLayout qui répond le mieux à mes besoins, mieux en tout cas que tous les autres programmes.* »

Un lecteur nous suggère de refaire une opération de ce genre, cette fois avec des programmes de mesure et de simulation pour PC. Pour d'autres, il faudrait faire la même chose avec du matériel, des cartes, des appareils etc. Quelqu'un nous a demandé de faire une revue des *programmes de programmation* (sic). A quoi pensait-il ? Des éditeurs, des assembleurs et des compilateurs sans doute. C'est une bonne idée.

Une critique récurrente portait sur le fait que le DVD proposait essentiellement des versions de démonstration gratuites. L'absence d'information sur le prix des versions complètes interdirait toute comparaison sérieuse. « *Comment voulez-vous comparer un programme à 200 avec un programme à 10 000 . Si vous ne tenez pas compte du prix. Il aurait fallu indiquer les prix, même approximatifs.* » Eh bien nous ne sommes pas de cet avis.
C'est justement en s'affranchissant des préjugés induits par le prix (« tout ce qui est bon est cher, ce qui est bon marché est médiocre ») que l'on peut évaluer sereinement les capacités d'un programme et son adéquation à nos besoins.
Ceci dit, un tableau avec les prix n'aurait peut-être pas été superflu.

Mécontent

Le DVD gratuit n'a pas fait l'unanimité. Le plus gros reproche est venu de ceux de nos lecteurs qui ont reçu leur numéro de novembre d'Elektor… sans DVD. Nous avons envoyé à ces malchanceux le DVD manquant (à condition qu'ils n'aient pas oublié de donner leur adresse).
Le second reproche concernait de prétendus dysfonctionnements du DVD. Dans tous les cas portés à notre connaissance il ne s'agissait finalement que d'erreurs de manipulation. Il ne fallait pas s'attendre à des miracles si vous mettiez le DVD dans un lecteur de CD-ROM ! Il n'est pas étonnant non plus qu'il ne se passe rien si vous mettez le DVD dans un lecteur de DVD de salon.

Les quelques DVD abîmés durant le transport ont également été remplacés à nos frais.
Il y a eu quelques énigmes aussi, comme par exemple celle du numéro arrivé chez un lecteur dans son enveloppe cachetée mais sans DVD. Un cas de magie noire heureusement isolé, et sur lequel nous ne nous étendrons donc pas.
Les critiques les plus dures – mais heureusement rares – ont qualifié le DVD de « *totalement inutile* » voir de « *sujet le plus naze jamais publié dans Elektor* ». D'autres se sont contentés de le trouver « *maigre* ».
Un reproche plus nuancé concernait la *Windomanie* généralisée. Une maladie fréquente en effet, mais qu'on ne saurait attribuer à ELEKTOR. Dans le même ordre d'idées, les plus rigoureux d'entre nos lecteurs ont observé que la plupart des programmes testés, une fois désinstallés, laissent sur l'ordinateur des traces de leur passage. Pour les effacer, il existe heureusement de très bons programmes.

Au nombre des remarques subtiles nous retenons la distinction faite par certains entre le DVD d'une part et l'article d'autre part. Ce dernier a moins marqué que le DVD lui-même. « *Beaucoup de programmes exigent un long apprentissage. A quoi bon faire cet effort si c'est pour une version de démonstration ?* » On comprend, entre les lignes, que cette personne aurait aimé que la rédaction d'Elektor en dise davantage sur sa propre expérience avec ces programmes, ses propres choix, et donne le conseil ultime, le tuyau en or massif. Ce dont nous nous sommes évidemment bien gardés, à dessein.
Le plus culotté de nos critiques affirme que le DVD aurait dû ne contenir que des versions complètes au lieu de versions de démonstration. Pourquoi pas ? Il suffirait qu'un des éditeurs de logiciel commence par proposer une version complète pour que d'autres suivent. Qui commence ?

Les plus désespérés sont les chants les plus beaux

Pour conclure nous ne renonçons pas au plaisir de citer ce témoignage dont l'authenticité ne fait aucun doute.
Citation : "Depuis des années j'utilise Capture et Layout. Dès le premier jour j'ai maudit ce programme, non sans espérer secrètement, de mise à jour en mise à jour, qu'il s'améliorerait. Quelle erreur ! Des fonctions au point dans une version donnée ont disparu dans la version suivante. Chacune des versions successives résolvait des problèmes créés par la précédente, mais à quoi bon puisqu'ils réapparaissaient obstinément dans une version ultérieure. Trois mises à jour successives n'ont rien apporté d'autre que des retouches cosmétiques et quelques fonctions superflues. La fonction d'annulation multiple a été intégrée dans Capture 10.0, mais elle ne concerne que des commandes dépourvues d'intérêt. Dans Layout il n'y a toujours pas de fonction d'annulation multiple, ce qu'Elektor avait d'ailleurs déploré lors d'un test il y a déjà pas mal de temps. […] Voilà ce que je peux dire de mon expérience avec un programme réputé *made by USA*."
Qui eût cru que le dessin de circuits imprimés puisse devenir un tel chemin de croix !

(050323-1)

Les citations de nos lecteurs sont anonymes mais les noms de leurs auteurs sont connus de la rédaction qui remercie vivement tous les participants.

Hexadoku

Puzzle pour les électroniciens

Nous vous présentons ici notre troisième Hexadoku, le casse-tête de nos lecteurs électroniciens… et de leurs familles.

Les instructions pour la résolution de ce puzzle sont enfantines. Le Hexadoku utilise les chiffres du système hexadécimal, à savoir de 0 à F. Remplissez le diagramme de 16 x 16 cases de façon à ce que **tous** les chiffres hexadécimaux de 0 à F (0 à 9 et A à F) ne soient utilisés **qu'une seule et unique fois** dans chaque rangée, colonne et carré de 4 x 4 cases (identifiés par une ligne plus grasse). Certains chiffres sont déjà placés dans le puzzle et en définissent ainsi sa situation de départ.
La solution de ce puzzle vous permettra de gagner un joli prix. Il vous suffit de nous envoyer la **série de chiffres** en grisé.

(065043-1)

Où envoyer ?

Envoyez votre réponse (les chiffres de la section grisée) par E-mail, télécopie ou courrier avant le **27 mars 2006** à

Elektor c/o Regus Roissy CDG
Le Dôme
1, rue de la Haye
BP 12910
01.49.19.26.19
95731 Roissy CDG
Email: hexadoku@elektor.fr

Tout recours légal est exclu de même que le sont les personnels de Segment B.V. et leur famille.

Les gagnants

La bonne solution de l'Hexadocu du numéro 331 (janvier) est : **EA639**
Le gagnant du **E-blocks Starter Kit Professional** est une gagnante :

Melle Corinne Bonnin

Les **3 bons Elektor** d'une valeur de **€50** chacun vont à :
Mme Chantal André de Fontaine, Mr Christian Fromentin de Rennes et Mr Daniel Muffato de Roubaix.
Nos félicitations aux lauréats !

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

Participez et gagnez !

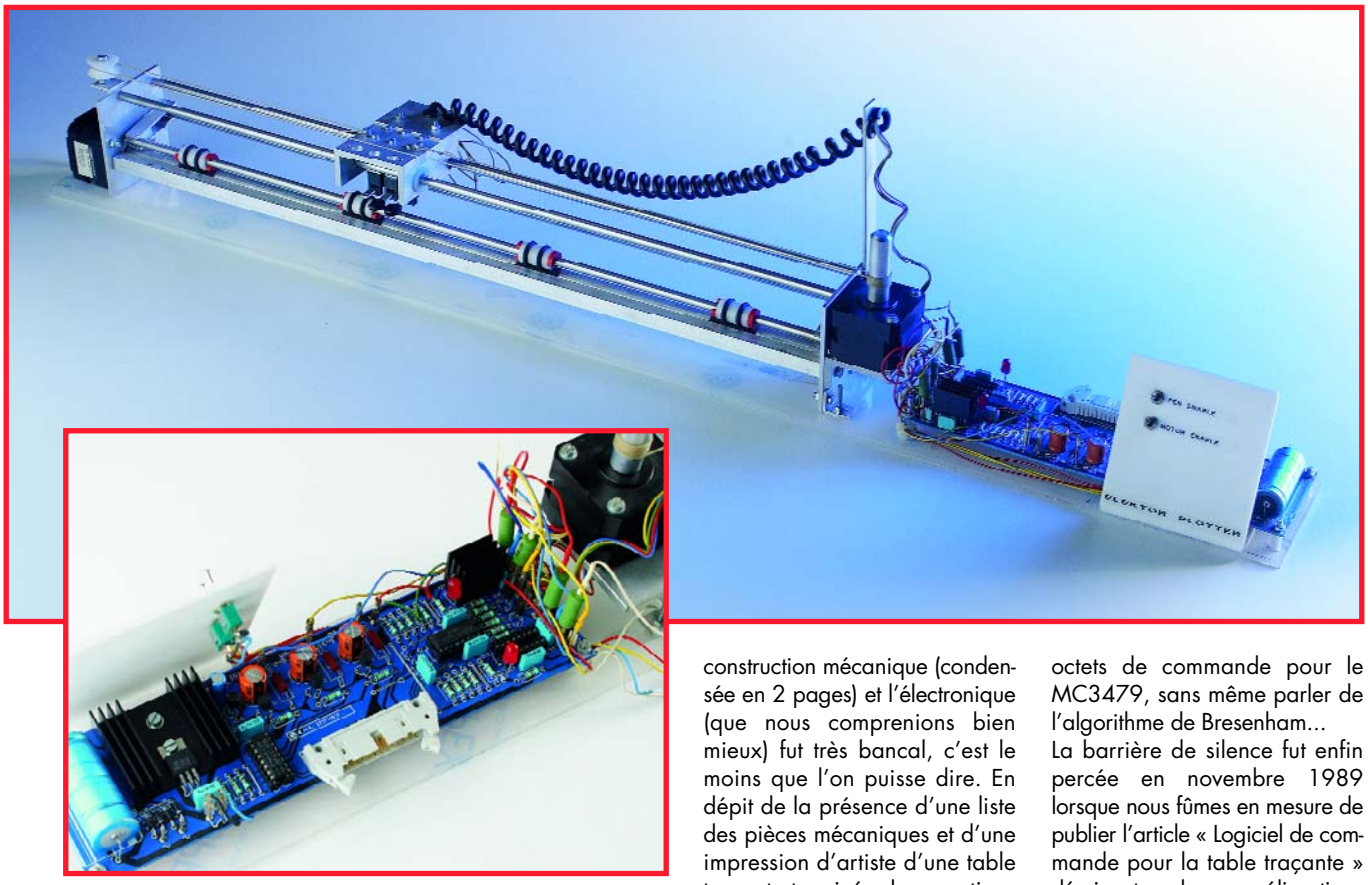
Nous tirerons au sort l'une des réponses correctes qui nous seront parvenues; son auteur recevra un

E-blocks Starter Kit Professional

d'une valeur de **€ 365,75**;

nous offrirons en outre
3 bons Elektor
d'une valeur de **€ 50,-**
chacun. Faites vos jeux !

Tables traçantes d'Elektor (1988-1991)



Jan Buiting

Cette histoire démarre en mai 1988, avec la publication d'un article dont le titre était à la délicieusement simple et parfaitement objectif : table traçante (1). La réalisation était basée sur un croquis coté épaulé par une lettre de l'un de nos lecteurs néerlandais, J. Arkema. C'était à l'époque où les tables traçantes étaient encore un périphérique pour PC lent, bruyant et coûteux que l'on ne risquait de rencontrer que dans les bureaux de dessin travaillant avec AutoCAD ou un autre programme de ce genre. L'amateur d'électronique avait bien entendu toutes sortes d'idées de ce qu'il ferait avec une table traçante s'il pouvait mettre la main sur un tel bijou : reproduire ses schémas et ses dessins de platine ! Malheureusement, il est extrêmement rare de trouver une table traçante haut de gamme, en particulier de marque Hewlett Packard, qui fonctionne encore.

La table traçante proposée à l'époque était une simple X-Y, le mouvement sur les axes des Y étant obtenu par le déplacement du plateau sur lequel était fixé la feuille de papier (A2 au maximum !), le déplacement en X étant l'affaire d'un « porte-plume » fixé à un axe de guidage de près de 50 cm de long. L'article de janvier 1988 comportait une jolie page de croquis cotés des parties, en aluminium pour la plupart, à découper, percer et limer avant de pouvoir rêver de l'assemblage de la table traçante. Ces dessins étaient faits à la main sur une grande table de dessin. Une tâche toute neuve pour notre bureau de dessin auquel il fallut plusieurs ébauches avant que la version définitive que pourrait comprendre un « laïc » ne soit finalement au point. Nous ne disposions pas à l'époque de quelque table traçante que ce soit ni même d'un programme ou d'un PC.

Malheureusement, en dépit de l'énergie consacrée à la production des dessins, l'équilibre entre la construction mécanique (condensée en 2 pages) et l'électronique (que nous comprenions bien mieux) fut très bancal, c'est le moins que l'on puisse dire. En dépit de la présence d'une liste des pièces mécaniques et d'une impression d'artiste d'une table traçante terminée, les questions fusèrent de toutes parts. Comment assembler « La Chose » ?, où trouver les pièces Skiffy, les aimants Binder, les moteurs pas à pas Berger et les recharges pour stylo Rotring ? Heureusement, l'une des sociétés qui faisait de la pub dans le numéro néerlandais, Meek-it de La Haye (NL), sauta le Rubicon et proposa un kit des pièces nécessaires à la réalisation de la table traçante.

Une fois que la poussière fut retombée autour de la construction mécanique, le silence se fit au sujet de l'aspect logiciel publié dans ce même article long de 12 pages (un record que nous n'avons pas encore battu…) consacré aux algorithmes qui devraient permettre, pensions-nous, aux lecteurs talentueux d'écrire leurs propres pilotes pour la commande de la table traçante depuis un PC. Nous nous trompions lourdement, et il apparut qu'il n'y avait personne à posséder les notions de ce que nous entendions par octants, tableaux matriciels pour moteur pas à pas, octets de commande pour le MC3479, sans même parler de l'algorithme de Bresenham…

La barrière de silence fut enfin percée en novembre 1989 lorsque nous fûmes en mesure de publier l'article « Logiciel de commande pour la table traçante » décrivant quelques améliorations mécaniques pour le concept d'origine mais plus important un pilote « partiellement » compatible HPGL pour le PC. Le programme de B. Lewetz fut baptisé Mondriaan, du nom d'un peintre néerlandais (1877-1944). Bien qu'il ne connaisse que 6 des instructions de base du set HPGL bien plus étoffé, le programme donnait enfin vie à un projet en hibernation depuis près de 2 ans. L'enthousiasme de nos lecteurs fut délirant, car ils attendaient depuis si (trop ?) longtemps ce lien indispensable entre le matériel et le logiciel. Les croquis fameux à l'époque du (feu) « Columbia » et du « réacteur » étaient fournis avec le programme, épaulés par quelques fichiers de test servant à la calibration des plumes et de la table traçante. Les programmes maîtres du marché à l'époque, AutoCAD et Autosketch n'utilisaient eux aussi que 6 instructions HPGL, de sorte que la table traçante d'Elektor se débrouillait fort bien connectée au port Centronics.

(065015-1)

Rétronique est une colonne mensuelle s'intéressant à de l'électronique du siècle dernier y compris des montages de légende décrits dans Elektor. Nous sommes ouverts à toutes les contributions et suggestions. N'hésitez pas à nous envoyer un E-mail à redaction@elektor.fr, sujet : Rétronique Elektor

Démagnétiser à l'aide d'une pièce recyclée

Klaus Rohwer

Il s'agit ici plus d'une astuce pratique que d'un secret de développement. Ce qui est arrivé à l'auteur lors d'un déménagement pour également arriver dans le quotidien d'un environnement de laboratoire : un millivoltmètre (antique) encombrant, relativement lourd, a été placé au-dessus d'un oscilloscope. Il y resta un certain nombre de jours, voire quelques semaines le temps passe si vite...
À un moment ou à un autre on se rend compte qu'un tel instrument de mesure à aiguille comporte un aimant au magnétisme très important. Mais il était déjà trop tard à ce moment-là. L'oscilloscope avait la gueule à laquelle on pouvait s'attendre, à savoir bien de travers. Le seul remède, une démagnétisation, mais comment s'y prendre ?

Dans l'armoire à bricoles (toute collection prend de la place !) j'ai trouvé quelques bobines provenant d'une vanne magnétique d'une vieille machine à laver la vaisselle ou le linge. Les bobines étaient encore en bon état, seule leur membrane devait sans doute être défectueuse.
Il est connu qu'il vaut mieux expérimenter que plonger dans les livres. D'où l'idée de connecter, avec les précautions requises (on se trouve en effet en présence de la tension du secteur) un câble secteur à cette bobine à embase AMP (on veillera à isoler parfaitement les connexions ainsi établies !). Après avoir placé quelques feuilles de papier sur l'oscilloscope, la bobine sous tension fut déplacée un certain nombre de fois au-dessus du coffret de l'oscilloscope qui, à l'approche de l'électro-aimant alimenté en courant alternatif, s'était mis à gron-

der fortement. Après cette thérapie de champ magnétique l'écran de l'oscilloscope avait heureusement retrouvé sa forme première... Petit conseil en guise de conclusion : il est recommandé de doter le câble secteur allant à la bobine d'un interrupteur pour câble ou de relier la bobine à une multiprise dotée d'un interrupteur général.

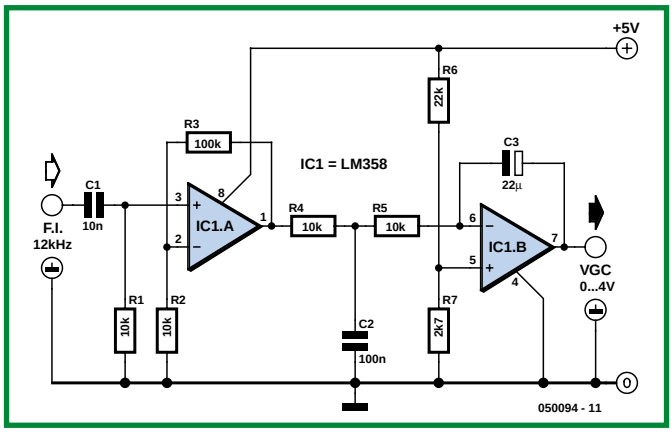
(050285-1)

CAG auto pour récepteur DRM

Burkhard Kainka

Un récepteur DRM délivre un signal de FI (**F**réquence **I**ntermédiaire) de 12 kHz à la carte-son du PC. Le programme se charge de la démodulation, opération au cours de laquelle il lui faut faire face à des différences de niveau importantes. Par la mise en oeuvre d'une CAG (**C**ommande de **G**ain **A**utomatique = AGC pour *Automatic Gain Control* de l'autre côté de la Manche) on peut faire en sorte que la carte-son se voit toujours appliquer la tension de signal optimale. C'est en particulier dans le cas de variations extrêmes de la puissance de champ, que l'on risque ainsi moins de décrochage lors du décodage. Certains appareils du commerce, le récepteur DRM DRT1 (www.schneider.de) par exemple, comportent une entrée de commande permettant de jouer sur le gain. Il est possible, par application à l'entrée VGC IN présente sur le dit modèle d'une tension de commande, de jouer sur le gain total du récepteur sur une plage de 25 dB/V environ. Le récepteur DRM d'Elektor peut lui aussi être doté d'une CAG (nous travaillons sur une solution à vous proposer dans un prochain secrets du concepteur).

Un amplificateur de réglage pour DRM doit respecter un cahier des charges spécial si l'on veut qu'il n'ait pas d'influence sur le signal qu'il fausserait sinon. Il faut commencer tout d'abord par veiller à ce qu'il n'y ait pas de variations de niveau trop rapide. La présente électronique traite le signal de sortie de 12 kHz du récepteur DRM et en dérive la tension de régulation. Le premier étage est un redresseur demi-onde. On pourrait penser, au premier abord, qu'il lui manque une diode. Le redresseur travaille en amplificateur avec limitation de la plage dans laquelle peut se mouvoir la tension de sortie. Les demi-ondes positives subissent un gain de 10x environ. Dans le cas de demi-ondes négatives, la tension de sortie reste à zéro, car comme le LM358 est alimenté à l'aide d'une tension asymétrique de 5 V, il ne peut pas être forcé à travailler dans la plage négative. Le résultat final est un redresseur demi-onde extrêmement simple



ne possédant pas le seuil d'activation classique dans le cas de redresseurs à diodes.
Le signal de sortie du redresseur attaque un intégrateur inverseur. Une tension continue moyennée supérieure à 0,5 V à l'entrée produit une tension en chute à la sortie, une tension inférieure à ce niveau se traduit par une tension en croissance en sortie de l'intégrateur. La boucle de régulation constituée par le paramétrage de gain du récepteur DRM et de l'amplificateur de régulation force la tension de sortie du récepteur à un niveau constant de l'ordre de 100 mV environ. Le critère important est la constante de régulation relativement importante qui évite des variations de niveau brutales. Ce circuit a fait ses preuves non seulement avec des récepteurs DRM mais aussi pour la réception de signaux en MA (**M**odulation d'**A**mplitude) et BLU (**B**ande **L**atérale **U**nique = SSB pour *Single Side Band*). Dans le cas d'une utilisation avec le DRT1 évoqué plus haut, la plage de réglage atteint de l'ordre de 115 dB de sorte que l'on a toujours le gain optimal.

(050094-1)

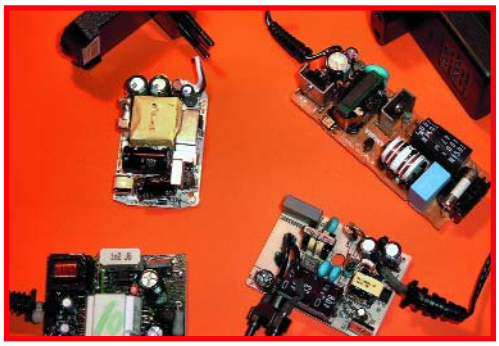
SPI-BOX

Nombre de microcontrôleurs et circuits périphériques modernes sont dotés d'une interface SPI (pour Serial Peripheral Interface) servant à la programmation ou à la commande du circuit intégré situé en aval dans le premier cas et en amont dans le second. La structure de cette interface est simple et ne requiert que peu de matériel pour sa commande depuis un PC. Souvent, on utilise à cet effet un port RS-232, mais cette approche présente quelques inconvénients. La solution que nous vous présentons dans le numéro d'avril évite ces écueils, garantissant en outre des temps de programmation courts, même si l'on utilise un adaptateur USB/série dans le cas où l'ordinateur de bureau ou portable ne possèderait plus d'embase RS-232.

L'ÉLECTRONIQUE À LA LESSIVE !

Existe-t-il encore un appareil grand public ne comportant pas un (ou plusieurs) microcontrôleur(s) ? Les fabricants des grandes marques consacrent ainsi énormément de temps (et d'argent, time is money) au développement tant matériel que logiciel d'un produit donné. Comme certains produits sont fabriqués à des millions d'exemplaires, on ne peut pas se permettre d'erreur de conception ou de programmation. Comme nous étions curieux de savoir comment un grand fabricant s'atèle à une telle « opération » nous avons fait un tour chez Miele, l'un des grands de la machine à laver et jeté un coup d'oeil par-dessus l'épaule des concepteurs.

AUTOPSIE DES BLOCS SECTEUR À DÉCOUPAGE



Les appareils mobiles tels que téléphones GSM, Assistants Personnels (PDA), lecteurs MP3 et les ordinateurs portables utilisent de plus en plus souvent, pour leur alimentation, une variété d'alimentation à découpage au lieu de la version à base de transformateur + régulateur. Il existe aujourd'hui des dizaines de modèles différents de ce type d'alimentation sur le marché. Nous en avons autopsié un certain nombre pour voir ce qu'elles avaient « dans le ventre ». Nous aborderons le moins prochain l'aspect technique du concept de ces alimentations, verrons comment elles sont construites et quelques autres approches pour réaliser une alimentation. Nous mettrons en pratique l'expérience acquise pour réaliser un petit convertisseur CC/CC.

Attention, le numéro d'avril 2006 devrait être en kiosque aux environs du 15 mars 2006.

WWW.ELEKTOR.FR WWW.ELEKTOR.FR WWW.ELEKTOR.FR

Le site Elektor - du sang neuf !

De par l'approche adoptée **« Projet par Projet »** lors de la construction, le visiteur de cette nouvelle mouture du site trouvera sur la même page, tout ce qui a trait à un projet donné : téléchargement de l'article au format .pdf, du logiciel, commande (platine et composants), mais aussi informations additionnelles et mises à jour.

Magazine : fait apparaître le sommaire du numéro le plus récent. Un clic sur le titre concerné permet de lire le début de l'article concerné.

Collection : Permet de remonter le temps grâce aux archives. Pour le moment, tous les articles depuis l'année 2000 sont téléchargeables, un moteur de recherche permettant de travailler par année et/ou par mot-clé.

Quoi de neuf sinon sur www.elektor.fr :

- Un Forum lecteur
- Petites Annonces Gratuites
- Nouvelles vous concernant
- Courriel Hebdomadaire Gratuit
- FAQ
- e-CHOPPE, pour tous vos achats

C booklet

reinhardt weber

```
setup_r8c()
{
    prc0 = 1;           /* Protect off */
    cm13 = 1;           /* Xin Xout */
    cm15 = 1;           /* XCIN-XCOUT drive capa
    cm05 = 0;           /* Xin on */
    cm16 = 0;           /* Main clock = No divi
    cm17 = 0;
    cm06 = 0;           /* CM16 and CM17 enable
    asm("nop");         /* Waiting for stable o
    asm("nop");         /* Assembler code
    asm("nop");
    asm("nop");
    ocd2 = 0;           /* Main clock change */
    prc0 = 0;           /* Protect on */
    pd1 = 0x0F;         /* Set Port 1.0-1.3 be
}
```

```
toggle_leds()
{
while (1)
{
p1   = 0x00;
```

pas exclusivement pour les fanas de la R8C

elektor
l'électronique imaginative

Préface

Pourquoi le C ?

Beaucoup de fans de l'électronique obtiennent d'excellents résultats avec les microcontrôleurs et n'hésitent pas à les programmer en assembleur. L'ampleur de la tâche et la complexité du logiciel assembleur les incitent à faire appel à un environnement de programmation plus puissant. S'il vous est arrivé de devoir résoudre en assembleur des fonctions mathématiques comme 1/x, le sinus ou d'autres, vous connaissez la difficulté. Dans cette optique, le langage évolué C, un standard industriel dans les domaines du microcontrôleur et du microprocesseur, offre des avantages décisifs. Les programmes en C sont portables, ce qui signifie qu'une structure logicielle rédigée reste transférable sur un autre type de processeur. Il n'y aura plus qu'à adapter l'arrangement des ports et les spécificités de ce qu'on appelle les registres à fonction spéciale.
Les professionnels considèrent qu'un logiciel qui demanderait deux semaines de développement en assembleur sera finalisé en C dans les deux ou trois jours. En outre, les fabricants de semi-conducteurs sont toujours plus nombreux à proposer des environnements de développement gratuits très performants. Raison de plus pour emboîter le pas au C !

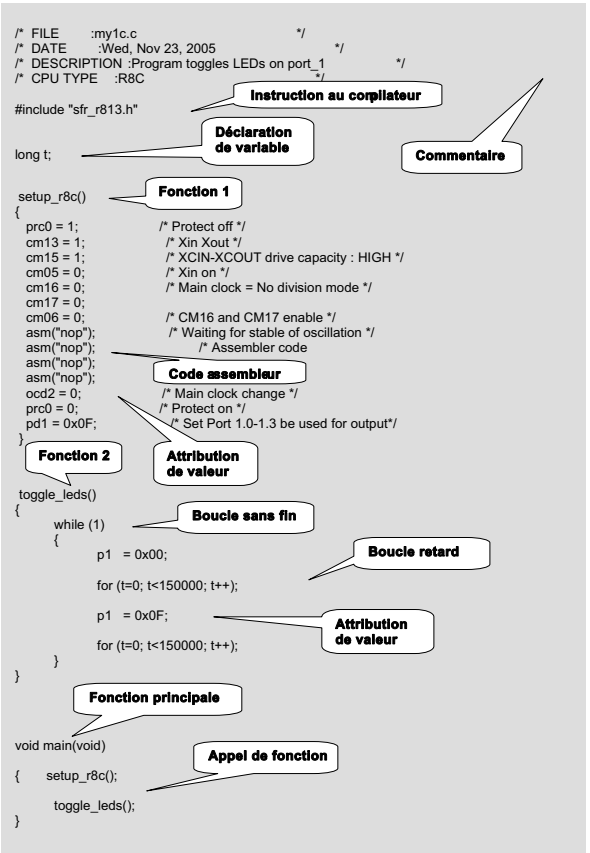
Minute ! Avant la couronne de laurier, il faut mouiller sa chemise. Pour se mettre au courant, on trouve des initiations au langage C et des programmes de démonstration dans les périodiques professionnels. Mais écrire ses propres programmes demande du temps. Et puis il faudra aussi se référer à des ouvrages spécialisés disponibles uniquement en anglais. Les questions et réponses que nous trouvons dans les forums de discussion sur le microcontrôleurs nous ont appris que de nombreux amateurs d'électronique considèrent qu'il ne faut surtout pas sous-estimer ces difficultés.

Cette brochure se limite aux éléments fondamentaux du langage C. Nous avons délibérément fait l'impasse sur les structures complexes du C telles que *pointer*, *array*, *string*, *structure*, *union* etc. Ce fascicule a pour but de servir de guide au débutant. Il n'a pas l'ambition de remplacer un cours approfondi.

Table des matières

La structure des programmes en C

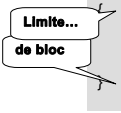
Tout logiciel en C se compose de différentes parties comme les commentaires, les instructions de prétraitement, les déclarations, définitions, expressions, assignations et fonctions. L'illustration vous en présente un exemple simple.



La fonction principale

Tout programme en C doit comporter au moins une fonction, la principale, dite **main function** Elle s'exécute toujours en premier et est appelée dès que le programme démarre. La meilleure façon de composer un fonction principale, c'est de lui réserver autant que possible l'appel des autres fonctions, plutôt que de lui confier la totalité du code du logiciel. Le programme en devient alors plus clair, plus aisé à faire évoluer et, si le travail est volumineux, facile à partager entre plusieurs programmeurs. On déclare une *main function* comme n'importe quelle autre fonction.

```
void main(void)
{
        /* votre code de

        programme ici */
}
```




```
main()
{
        /* votre code de

        programme ici */
}
```

Toutes les instructions et fonctions de la fonction principale sont entourées d'accolades {…} (*curly brackets* en anglais). Il s'agit de former des blocs.
Void signifie « vide » et indique que la *main function* ne comporte aucun paramètre d'entrée et qu'après exécution, elle ne renverra aucun résultat en sortie. Ces deux mots-clés sont optionnels, on peut les omettre.

Les commentaires en C

Les ***comments***(commentaires) sont des textes ou des paragraphes qui n'entrent pas dans la programmation proprement dite, que le compilateur ne traduira pas et qui n'occuperont donc pas d'espace en mémoire. Ils revêtent pourtant une grande importance parce qu'ils expliquent aux autres personnes le sens du programme ou de ses parties essentielles. Ils sont aussi d'un grand secours au rédacteur du logiciel, parce qu'après quelques jours, on ne se souvient pas forcément pourquoi on a formulé le code de la sorte plutôt qu'autrement. Le temps gagné à les négliger risque de se perdre après en recherches fastidieuses.

```
/*
Les commentaires
sont délimités
par barres
obliques et
astérisques…
*/
```

```
// sauf pour une ligne de commentaire
```

Les commentaires d'une seule ligne débutent par deux barres obliques et s'arrêtent automatiquement en fin de ligne.
En assembleur, on utilise le point-virgule (;) comme indicatif de texte explicatif.
Mais en langage C, le point-virgule sert à terminer une instruction !

#include

La norme ANSI du langage C ne retient pas de nombreuses déclarations et fonctions qui, sans être indispensables, sont cependant très utiles. On les dissimule dans ce que l'on appelle des bibliothèques (libraries) . Dès lors, pour que le compilateur les trouve, au cours de la traduction des codes sources C, il faut les déclarer comme fichiers d'en-tête, appelés header files à inclure (include). On reconnaît ces fichiers à leur suffixe « **.**h ». Exemples :

```
#include "stdio.h"
```

Parmi les programmes en C, on utilise ceux qui ont été rédigés pour le PC : [Standard-input-output, contient e. a. la fonction d'impression printf()]

```
#include "sfr       _r813.h"
```

de la bibliothèque de Renesas. Ici, les noms et les bits des registres du contrôleur R8C sont définis comme p1, pd1, p1_7 …

Les mots-clés en C

Dans la norme ANSI du langage C, il existe 32 mots-clés définis, appelés keywords . Ces mots sont réservés au compilateur. Tous les keywords doivent être écrits en minuscules et on ne peut pas les utiliser à d'autres fins, comme nom de variable, par exemple.

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Plusieurs compilateurs C ajoutent aux définitions ANSI d'autres mots-clés pour mieux mettre à profit les possibilités du processeur. Voici les termes supplémentaires applicables comme keywords pour le microcontrôleur R8C :

_asm	asm	near
_far	_Bool	restrict
_near	far	inline

Les constantes et variables

Les systèmes de numération

Le langage C connaît différents systèmes de numération (number base , base de numération) : le nombre décimal, binaire, octal ou hexadécimal.
Les données chiffrées sans spécification de la notation sont interprétées par défaut comme des nombres décimaux. Toutes les autres bases doivent être identifiées. Les nombres en octal sont précédés de 0 (zéro), ceux en hexadécimal commencent par 0x et les binaires par 0b.

Base	Formulation	Chiffres permis	Exemples
Decimal(10)	-	0123456789	5
Octal(8)	0…	01234567	05
Hexadecimal(16)	0x…	0123456789ABCDEF	0x5
Binary (2)	0b	0 1	0b11110000

En C, il faut faire attention à la notation américaine des nombres. Alors qu'en français nous utilisons systématiquement la virgule pour séparer la partie entière de la partie décimale, les nombres en virgule flottante (float) s'écrivent avec un point décimal. La virgule (comma), en C, sert de séparateur dans les listes entre nombres ou entre variables. Le double point (colon) caractérise une suite numérique ininterrompue.

Exemples :

USA	F
3.14159	3,14159
3,4	3 (et) 4
0:3	0->3, donc: 0,1,2,à 3

Les types de données

Avant d'utiliser des données en C, il faut en déclarer le type. Sans quoi, le compilateur ne peut pas savoir combien de mémoire il doit leur réserver. En principe, on choisit toujours le type qui suffit pour un usage donné, sans occuper inutilement de place en mémoire. Voici les principaux types de données :

Type	espace mémoire	Domaine
_Bool	8	0, 1
char	8	0 -> +255
unsigned char	8	0 -> +255
signed char	8	-128 -> +127
int, short	16	-32768 -> +32767
double	64	$2.22..e^{-308}$ -> $1.79..e^{+308}$
unsigned int	16	0 -> +65535
long	32	-2147483648 -> +2147483647
float	32	$-1.17..e^{-38}$ F-> $+3.4..e^{+38}$F

Exemples :

```
_Bool bouton_stop // bouton à 2 positions  marche/arrêt
unsigned int _an  // assez pour un millésime  0 à 65 535
float _volume     // réel, virgule flottante pour calculs
```

Les constantes

Une constante (constant) est un nombre que le programme ne devra pas modifier. Tous les nombres « normaux » peuvent en faire partie. Un nombre entier (integer) s'écrit sans virgule. Un nombre réel à virgule flottante (float) peut comporter des chiffres derrière le point décimal. Les signes du clavier (character), on les enferme entre guillemets simples ' …' (single quotes).
On déclare les constantes à l'aide du mot clé : #define .

```
#define    <label>          value
```

pas de ; c'est seulement pour le compilateur

Exemples :

```
#define true  1                  // true = vrai
#define false 0                  // false = faux, pas vrai
#define pi 3.14159               // rapport circonf/diamètre
#define lettre_1  'A'            // touche A
```

On peut choisir à son gré les noms des constantes, variables et fonctions, à condition de ne pas y mettre de mot-clé ni de signe d'opération. En principe, on n'utilise que les lettres de l'alphabet anglais, les chiffres et la barre de soulignement _. Choisissons de préférence des noms parlants, bouton_alerte en dira davantage que t1, par exemple !

Les variables

Par variable, on entend une place en mémoire pour un nombre, une lettre ou un texte que le programme pourra modifier. En C, toutes les variables doivent avoir été déclarées avant l'emploi. Les variables font partie des « statements » et se terminent pas un point-virgule. Voici comment déclarer une variable :

```
type       <label>          ;
```

; ici, c'est pour le processeur !

Exemples :

```
_Bool        action_poussoir ;

long         nombre ;

float        rayon ;
```

… et voici comment indiquer la valeur (value) de la variable :

```
<label>    =      value        ;
```

Exemples :

```
action_poussoir = 1 ;              val_min = nombre - 50 ;

action_poussoir = false ;        val_min = nombre * nombre ;

nombre = 100 ;                        _circonf = rayon * 2 * pi ;
```

Les commandes et les instructions qui s'adressent au processeur -- on les appelle ici « *statements* » -- sont terminées par un point-virgule (;).

Les opérateurs en C

Les opérateurs arithmétiques

Les signes des opérations arithmétiques correspondent à ceux bien connus des calculettes :

```
+    Addition              // Exemples    y = x + 3
-    Soustraction          //              y = x - b
*    Multiplication        //             y = a * b
/    Division              //              Y = a / b
```

Le signe égal n'a pas la même fonction en C que dans les mathématiques traditionnelles, il sert d'opérateur d'assignation. Cela veut dire que le membre à droite du signe égal représente un calcul à effectuer, dont le résultat sera attribué à la variable inscrite à gauche. Aussi, les expressions suivantes sont-elles admises en C, mais n'auraient pas la même portée en mathématique normale :


```
x = x+y ;      // calculer x+y et mémoriser la réponse dans x
x = -x ;       // changer le signe de la variable x
```

Les opérateurs relationnels

Les opérateurs relationnels servent à comparer des variables. Ils fournissent ensuite un résultat true (vrai) ou false (faux).

```
>     plus grand que          ==    égal
>=    plus grand ou égal      !=    non égal
<     plus petit que
<=    plus petit ou égal
```

Les opérateurs logiques

A l'aide des opérateurs logiques ET, OU et NON, on peut exprimer les associations bien connues en technique numérique.

b	a	AND a && b	OR a || b	NOT !a
0	0	0	0	1
0	1	0	1	0
1	0	0	1	1
1	1	1	1	0

Exemple :

```
If(_prix <= prix_max && compte > 1000)
        _acheter();

/* La fonction _acheter() ne sera exécutée que si _prix est
plus petit ou égal à prix_max et que _compte est supérieur à
1 000 € */
```

Le compilateur C de Renesas pour le R8C compte d'autres opérateur logiques destinés à l'association de variables au niveau du bit :

		b	a	a^b
&	pour opérateur ET bit par bit	0	0	0
|	pour opérateur OU bit par bit	0	1	1
		1	0	1
^	pour opérateur XOR bit par bit	1	1	0

Exemples :

```
a   = 10011010
b   = 11000011
a&b = 10000010
```

```
a   = 10011010
b   = 11000011
a|b = 11011011
```

```
a   = 10011010
b   = 11000011
a^b = 01011001
```

Les raccourcis

Les États-uniens sont passés maîtres dans la composition d'abréviations (shortcuts) . La remarque s'applique particulièrement aux concepteurs du langage C, Dennis Ritchie et Brian Kerningham. Voilà qui permet de se simplifier la tâche à la saisie du codage.

Shortcut	Normal	Shortcuts	Normal
a*=b	a = a*b	a<<=b	a = a<<b
a/=b	a = a/b	a>>=b	a = a>>b
a+=b	a = a+b	a&=b	a = a&b
a-=b	a = a-b	a|=b	a = a|b
a%=b	a = a%b	a^=b	a = a^b
a++	a = a+1 (increment)		
a--	a = a-1 (decrement)		

Exemple :

```
for(t=0, t<100000, t++);        /* boucle retard  */
```

Les fonctions en C

La notion de fonction

Les fonctions sont l'alpha et l'oméga de la langue de programmation C. Une fonction peut, à partir du programme principal ou de n'importe où, appeler une autre fonction. Chaque programme en C doit au moins contenir une fonction : main() . Elle sera appelée automatiquement au lancement du programme.
Les fonctions sont des parties de programme indépendantes (blocks) qui remplissent une tâche déterminée (operation) comme nous les connaissons bien sur les calculettes :

CE efface la mémoire de saisie de la calculette. En C, nous dirions qu'il s'agit d'une fonction qui ne réclame aucune entrée (paramètre) et ne restitue aucun nombre.

1/x attend l'entrée d'un nombre pour en calculer l'inverse. En C, on parle alors d'une fonction à un paramètre d'entrée.

+ - * / en revanche demandent l'introduction de deux paramètres et

O# la fonction somme même davantage.

sin et log sont par contre des fonctions qui exigent l'entrée d'un paramètre et fournissent aussi un résultat (result) en sortie.

Déclaration d'une fonction

La forme générale d'une fonction C est :

```
type  function_name(type var1,type var2,type var3,...)                    ;
```

;

Type de sortie

Paramètre d'entrée avec son type

Exemples :

Une fonction sans paramètre d'entrée ni de sortie

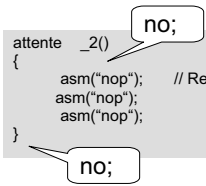
```
void    attente    _1(void)
    asm("nop");              //Appel de «       no  operation » en assembleur
```

Le mot void (vide) dit au compilateur que la fonction _attente_1 n'a pas besoin de paramètre d'entrée et qu'elle n'enverra pas de résultat non plus. Les deux mots-clés void peuvent d'ailleurs être omis :

```
attente    _1()
  asm("nop");
```

;

Quand une fonction contient plus d'une instruction, il faut les grouper entre accolades (curly brackets). Le point-virgule final doit alors s'éclipser !



Une fonction avec entrée et pourtant pas de paramètre de sortie

```
int t;

attente _3(int autant_de_fois )
  for(t=0;t= autant_de_fois;t++);
```

/* la boucle for compte t de zéro en croissant, par incréments de 1, jusqu'à la valeur indiquée par autant_de_fois (délai) */

une fonction avec paramètre en entrée et en sortie

```
float _volume(float long, float large, float haut)
  return long*large*haut;
```

/* Cette fonction attend trois entrées à stocker dans les variables long, large et haut. Ensuite, elle calcule le produit des trois et renvoie (return) à l'expéditeur un nombre en notation à virgule flottante (float) */

Appel de fonction

On appelle une fonction simplement par son nom et on peut le faire à n'importe quel endroit d'un programme. Dès que la fonction a terminé sa tâche, ce qui se remarque par le point-virgule (;) s'il n'y a qu'une instruction ou l'accolade (}) quand il y en a plusieurs, le programme renvoie automatiquement la main à la partie appelante. Le mot-clé return a en C une autre signification qu'en assembleur. En C, il sert à communiquer le résultat trouvé et pas à marquer la fin de la routine.

Les fonctions peuvent être imbriquées (*nested*), ce qui veut dire qu'une fonction peut en appeler une autre et ainsi de suite.

Exemples :

Appel d'une fonction à partir du programme principal, sans paramètre d'entrée ni de sortie.

```
void  main(void)
{
        attente    _1();
}
```

Appel d'une fonction à partir du programme principal, avec paramètre d'entrée mais pas de sortie.

```
void main(void);
{
        attente    _3(100);
}
```

// expédition de la constante 100 à la fonction attente_3

Appel d'une fonction à partir du programme principal, avec paramètre d'entrée et de sortie.

```
void main(void);
{
        nombre_de_litres        = _volume(a,b,c);
}
```

/* On envoie à la fonction _volume les valeurs des variables a, b et c. La fonction s'en sert pour calculer le volume de l'objet et passe la réponse sous la forme de la variable nombre_de_litres */

Les commandes du programme

If

Il arrive souvent qu'une instruction (statement) ou un bloc d'instructions ne doive s'exécuter que si (if) une certaine condition (condition) est remplie.
Une condition est considéré comme remplie si le test de la condition renvoie la valeur « vrai ». Un zéro pour cette valeur signifie « faux » (false), n'importe quel autre chiffre est pris pour « vrai » (true).

La forme générale est :

;

```
if    (condition)      statement ;
```

S'il faut plusieurs instructions pour définir la condition, alors on doit les grouper entre accolades.

no;

```
if (condition)
{
        statement_1
        statement_2
        statement_3
        // . . .
}
```

no;

Exemples :

==

```
if(touche == 3)
        led_rouge = _on;
```

```
if(touche == 3)
{
        led_verte = _off;
        led_rouge = _on;
}
```

If-Else

S'il vous faut faire exécuter une ou plusieurs instructions quand la condition est réalisée, mais aussi d'autres instructions si elle ne se réalise pas, la commande dont vous avez besoin, c'est if – else (si – autrement).

La forme générale en est :

;

;

```
if    (condition)      statement_1 ;    else    statement_2 ;
```

et s'il y a plusieurs instructions à exécuter, il faut aussi les grouper par des { }.

Exemples :

```
if(touche == 3)
        led_rouge = _on;
else
        led_verte = _on;
```

;

;

```
if(touche == 3)
{
        led_verte = _off;
        led_rouge = _on;
}
else
        led_verte = _on;
```

==

```
if(touche == 3)
{
        led_verte = _off;
        led_rouge = _on;
}
else
{
        led_verte = _on;
        led_rouge = _off;
}
```

Bloc

Switch

Si, dans une décision, il y a plus de deux options à envisager, l'utilisation de la commande if-else devient laborieuse. On préfère alors faire appel à l'alternative multiple switch – case. On dispose en effet dans cette version du logiciel d'une sorte de commutateur (switch) rotatif à plusieurs positions (case).

La forme générale est :

no;

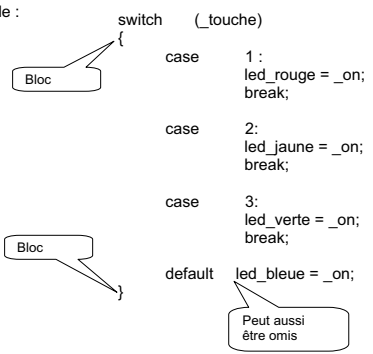
```
switch      (variable)
{
        case        constante_1 :
                    instruction_1;
                    break;

        case        constante_2:
                    instruction_2;
                    break;

        case        constante_3:
                    instruction_3;

        case        // . . .
                    break;

        default:  instruction_x;
}
```

no;

La fonction switch compare le contenu de la variable à celle de la constante dans les différents cas (case) prévus. Quand le résultat de la comparaison est positif, la ou les instructions correspondantes sont exécutées.

Quand le mot-clé break (arrêt) est atteint, l'exécution du programme retourne chez le demandeur de la décision. Si aucun des cas prévus n'a été rencontré, c'est l'instruction sous default qui s'exécute. Si celle-ci n'est pas nécessaire, on peut omettre cette partie.

Exemple :




```
/* si la valeur de _touche = 1 la LED rouge s'allume, pour 2
c'est la jaune et pour 3, la verte. Si la variable _touche ne
contient ni 1, ni 2, ni 3, mais par exemple 4, alors la LED
bleue est activée.
```

For

Si la même partie d'un programme doit s'exécuter plusieurs fois, on utilise la boucle (loop) for .

En voici la forme générale :

```
for(val_initiale; val_test; val_pas)
        instruction_1;
                      ;
                      ;
```

A l'appel de for loop , la valeur initiale est assignée au compteur. A chaque itération de la boucle, on ajoute (ou retranche) au compteur la valeur du pas jusqu'à atteindre la valeur test et que le test résulte en une valeur logique true .

Si plusieurs instructions doivent être exécutées dans la boucle, elles seront encadrées d'accolades.

Exemples :

```
int  t;
```

Variable com pteur

```
for(t=0; t < 10; t++)
      cligno_led();
```

/* lors de l'arrivée dans la boucle for, on attribue à la variable t la valeur 0. Alors, on appelle la fonction cligno_led. Après la première itération dans la boucle, la variable t se voit incrémentée (t++) et prend la valeur 1. Comme 1 est plus petit que 10, on continue le carrousel jusqu'à 9. La boucle sera ainsi bouclée 10 fois */

Variable

Variable com pteur

```
a = 2;
b = 10;
c = 4;

int i;

for(i = a; i < b; i+ = c)
{
      led_rouge = _on;
      led_rouge = _off;
}
```

/* cette boucle-ci ne sera parcourue que 2 fois */

While

On se sert de la boucle while quand l'exécution des instructions incluses est soumise à une condition.

Sa forme générale est la suivante :

no;

```
while(_condition)
{
      instruction_1;
      instruction_2;
      // . . .
}
```

no;

Lors de l'appel de la boucle while, la condition est d'abord testée. Si la réponse est affirmative (true) l'instruction_1 s'exécute aussi longtemps que le test ne renvoie pas un résultat false .

Exemple :

```
#define true 1

while(i == true)
{
        i = bouton_poussé();
        cligno_led();
}
```

==

/* Aussi longtemps que la fonction bouton_poussé retourne la
variable i avec la valeur 1, la fonction cligno_led s'exécute
*/

Do-While

Une boucle while ne sera pas exécutée si au moment de son lancement la condition n'est pas remplie.
Si au moins une des instructions doit être exécutée, l'instruction de test doit la suivre. Dans ce cas, on utilise la boucle do-while .

Sa forme générale est :

```
do
{
        instruction_1;
        instruction_2;
        instruction_3;
        // . . .
}
while(_condition);
```

no;

no;

;

Exemple :

```
#define true 1

do
{
        i = bouton_poussé();
        cligno_led();
}
while(i == true);
```

==

/* Dans ce cas-ci, la fonction cligno_led s'exécute au moins
une fois */

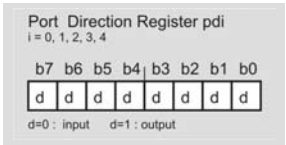
Le fichier d'en-tête « sfr_r813.h »

Le Header File sfr_r813.h donne accès aux SFR (*special function register*) du microcontrôleur R8C. Ces registres contiennent les réglages de base du microcontrôleur tels que la direction des ports (entrée/sortie), les temporisateurs, le convertisseur A/N, les UART et autres.
Le microcontrôleur R8C compte plus de 50 SFR, raison pour laquelle il convient, à ce stade, d'opérer une sélection des SFR principaux. Vous trouverez toutes les informations nécessaires sur les SFR dans le manuel R8C/13 Group Hardware.

Registres des ports P0, P1, P2, P3 et P4

Par ports , on entend les positions de mémoire dont les lignes sortent sur les broches du boîtier du microcontrôleur. On utilise les ports pour entrer et sortir des données. Un port peut être configuré en entrée (input) ou en sortie (output). Lors de la mise sous tension, tous les ports sont des entrées. C'est le registre de direction du port (pd) qui permet d'en changer la destination.

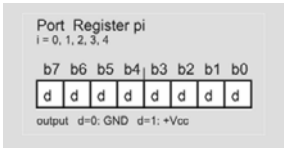
Voici comment définir la direction d'un port :



Beispiele

```
pd1 = 0x0F;
/* port1, bits 0->3 = output
   bits 4->7 = input */

pd2_3 = 1;
/* port2, bit3 = output */
```

… et voici comment fournir les données au port.



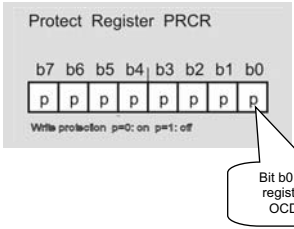
Exemples

```
p1 = 0x0F;
/* port1, bits 0->3 = 1
   bits 4->7 = 0 */

pd2_3 = 0;
/* port2, bit3 = 0 */
```

/* Si l'on inscrit dans un registre de port un 1, la tension d'alimentation (p.ex. +5 V) apparaîtra sur la broche correspondante de la puce. Un zéro ramène la broche à la tension de masse (gnd = 0 V) */

Protection Register PRCR

Il est possible, grâce au PRCR, de protéger le contenu d'autres registres importants contre l'écriture intempestive, par exemple si le programme se plante.

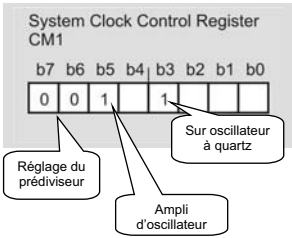


Exemples

```
prc0 = 1;
/* Protection mise */

prc0 = 0;
/* Protection enlevée */
```

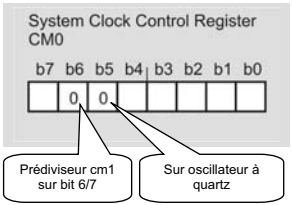
System Clock Control Register CM1, CM2 et OSD

Le microcontrôleur R8C contient deux oscillateurs pour scander l'unité centrale. L'un d'eux est embarqué « on-chip-oscillator », l'autre, appelé « main clock », fonctionne à l'aide d'un quartz extérieur relié aux broches Xin et Xout. Le registre CM détermine la manière de scander(clock) le microcontrôleur. En outre, on peut programmer un diviseur préalable (prescaler) pour ralentir le processeur.



Exemples

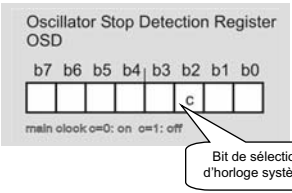
```
cm13 = 1;
/* Xin-Xout aux ports p46 et p47
= Quartz ext. */

cm15 = 1;    Exemples
/* tampon pour Xin-Xout */

cm16 = 0;
/* prédiviseur d'horloge */

cm17 = 0;
/* prédiviseur d'horloge */
```



Exemples

```
cm05 = 0;
/* brancher Xin-Xout aux ports
p46 et p47 */

cm06 = 0;
/* brancher le prédiviseur */
```

On trouve encore un registre OSD (*Oscillator Stop Detection*) responsable de la sélection d'horloge et la supervision de son régime.



Exemples

```
osd2 = 0;
/* Oscillateur externe (quartz)
comme horloge système */

osd2 = 1;
/*   débrancher le quartz */
```

Le fichier d'en-tête « math.h »

Le Header File math.h est la bibliothèque des fonctions mathématiques du R8C. Il suffit à présent d'y puiser les fonctions principales nécessaires à la bonne exécution du programme.

Fonctions du type
double f_name(double x);

```
sin();
cos();
tan();

asin();
acos();
atan();

sinh();
cosh();
tanh();

sqrt();

exp();
log();
log10();

mod();

fabs();
floor();
ceil();
```

Fonctions du type
float f_name(float x);

```
sinf();
cosf();
tanf();

asinf();
acosf();
atanf();

sinhf();
coshf();
tanhf();

sqrtf();
powf();

expf();
logf();
log10f();

fabsf();
floorf();
ceilf();
```

Fonctions du type
double f_name(double x, double y);

```
pow();

fmod();

atan2();
```

Fonctions du type
float f_name(float x, float y);

```
powf();

atan2f();

fmodf();
```

elektor
l'électronique imaginative

Programmation in situ

de fonctions numériques et analogiques

JTAG / SPI • GAL, PAC, CPLD
In System Programmable

L'histoire des circuits intégrés programmables trouve aujourd'hui un aboutissement extraordinaire avec la programmation en circuit et même en fonctionnement. Le mot magique, c'est isp, pour in system programmable et concerne des fonctions numériques, mais aussi des fonctions analogiques.

17 x 23,5 cm
256 pages + CD-ROM
ISBN 2-86661-140-3
39 €

Elektor
BP 12910
95731 Roissy CDG
Tél. : (+33) 01.49.19.26.19
Fax : (+33) 01.49.19.22.37
E-mail : ventes@elektor.fr

Pour la Suisse
Sono Light Import
Champs-Montants 16b
CH-2074 Marin-Epagnier
Tél. : 032.710.16.60
Fax : 032.710.16.63
E-mail : admin@sonolight.ch

www.elektor.fr

elektor
l'électronique imaginative

Carte de programmation et d'expérimentation en kit monté prêt à l'emploi !

Kit de la carte Flash 89S8252 (avec livre) pour 125,80 €
Livre seul : 26,80 €

Pour vous faciliter le démarrage, nous proposons à l'occasion de la parution du livre « Programmation de microcontrôleurs », le kit de la carte Flash 89S8252 d'Elektor (n°282 décembre 2001, page 20) monté et prêt à l'emploi, avec bloc d'alimentation et câble de connexion, ainsi que le logiciel sur CD-ROM.

Elektor
BP 12910
95731 Roissy CDG
Tél. : (+33) 01.49.19.26.19
Fax : (+33) 01.49.19.22.37
E-mail : ventes@elektor.fr

Pour la Suisse
Sono Light Import
Champs-Montants 16b
CH-2074 Marin-Epagnier
Tél. : 032.710.16.60
Fax : 032.710.16.63
E-mail : admin@sonolight.ch

www.elektor.fr