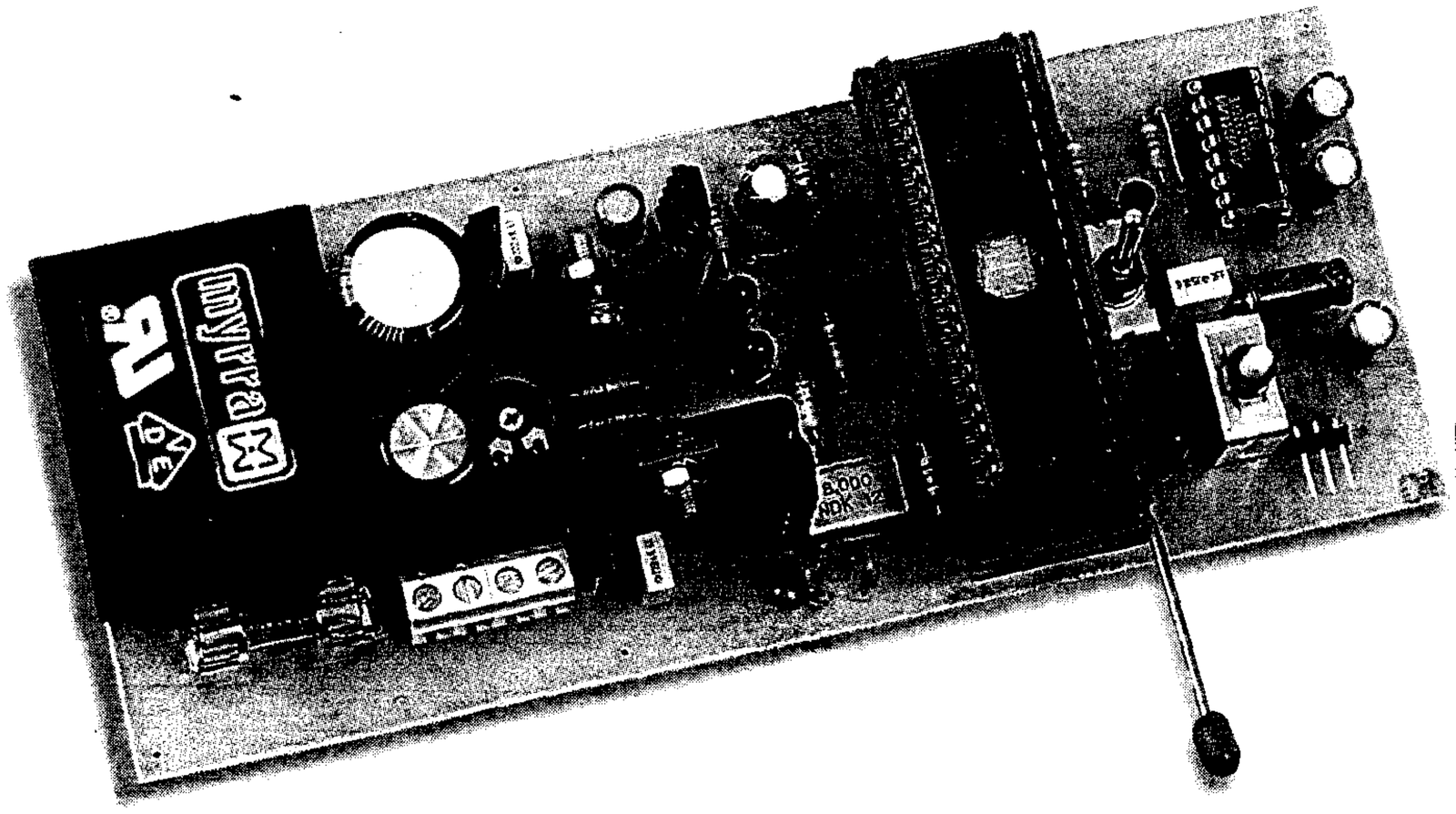
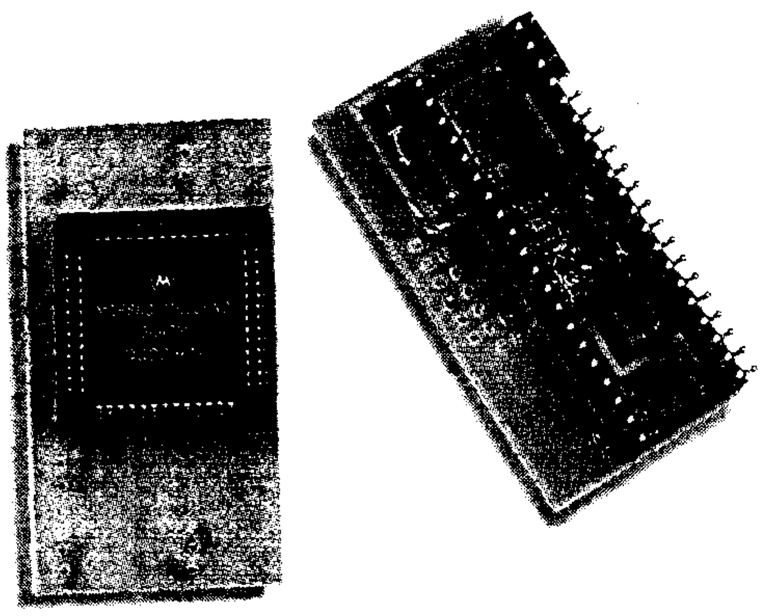
réalisation

Programmateur de 68HC11

1ère PARTIE

Bien qu'il ait été introduit sur le marché depuis près de dix ans, le microcontrôleur 68HC11 de Motorola est resté quasiment ignoré des amateurs jusqu'à ce que son utilisation dans un montage que la morale réprouve, lui assure une publicité très efficace.





Les adaptateurs à utiliser selon la version de boîtier choisie.

Ce microcontrôleur est pourtant particulièrement intéressant car il est extrêmement bien équipé en ressources internes et sa programmation est particulièrement simple. De plus, bien que ce soit un circuit 8 bits, son unité centrale supporte de nombreuses instructions 16 bits ce qui accroît de façon non négligeable sa puissance de traitement. Bien sûr, comme pour tout microcontrôleur qui se respecte, il faut disposer d'un outil de développement pour pouvoir mettre au point des applications mais, dans le cas du 68HC11, l'assembleur qui est la pierre angulaire de cet outil est disponible gratuitement. Reste évidemment à programmer le circuit une fois l'application mise au point mais cet article est là pour vous montrer comment faire avec n'importe quel compatible PC ou MacIntosh et une poignée de composants fort peu coûteux.

Précisons dès à présent qu'il n'est pas question, comme nous l'avons vu faire quelquefois, de programmer une mémoire externe et d'utiliser un 68HC11 dépourvu de ROM en mode étendu, ce qui conduit alors à un schéma lourd et à un circuit imprimé délicat. Non il s'agit bel et bien de programmer la mémoire interne de programme d'un 68HC11 afin de pouvoir l'utiliser seul, c'est à dire en mode "single chip" qui est évidemment le plus intéressant et le moins coûteux.

Présentation

Dans le cadre de cet article, il n'est évidemment pas question de réaliser un cours sur le 68HC11 et sur sa programmation vu le nombre de pages qui seraient nécessaires, deux ouvrages en français sont disponibles chez Dunod pour cela (voir bibliographie). Par contre, afin que vous soyez en mesure de bien comprendre le principe de fonctionnement du programmateur et de modifier si nécessaire son programme de commande, il est indispensable que vous possédiez quelques notions relatives à l'architecture interne du 68HC11 et à un mode de fonctionnement particulier qui s'appelle le mode "bootstrap". Rappelons tout d'abord que, selon une pratique désormais courante, il n'existe pas un circuit 68HC11 mais plusieurs dizaines de types différents. Tous ces circuits font cependant appel aux mêmes sous-ensembles de base groupés de façon différente selon les boîtiers. La différence la plus importante entre les différentes versions, dans le cadre de notre programmateur, est la mémoire morte ou ROM de programme. En effet, cette mémoire peut être présente ou absente. Lorsqu'elle est présente, elle peut être du type ROM programmable par masque, c'est à dire lors de la fabrication du circuit et il n'est alors pas question de pouvoir la programmer soi même. Mais elle peut aussi être type EPROM c'est à dire programmable électriquement. Dans ce dernier cas, cette mémoire EPROM peut être non effaçable ; elle s'appelle alors OTPROM pour One Time PROM c'est à dire ROM programmable une fois. Elle peut aussi être de type UVPROM c'est à dire programmable électriquement et effaçable aux ultraviolets.

C'est évidemment aux circuits équipés de telles mémoires (OTPROM ou UVPROM) que nous allons faire appel pour pouvoir utiliser notre programmateur ; mais que l'on se rassure tout de suite, ces circuits ne sont ni rares ni chers, tout au moins pour les versions OTPROM.

Architecture interne

La majorité des 68HC11 adopte l'architecture interne que vous pouvez découvrir en figure 1.

Bibliographie

• Note d'application AN 1060 de Motorola "MC68HC11 Bootstrap Mode".

• Microcontrôleur 68HC11 - Description, par C. Tavernier aux Editions Dunod.

• Microcontrôleur 68HC11 - Applications, par C. Tavernier aux Editions Dunod (sortie en juin 1997).

Autour de l'unité centrale 68HC11 proprement dite qui, pour les habitués des produits Motorola, est dérivée du "vieux" 6801, on trouve tout d'abord de la mémoire.
Elle se subdivise au maximum en trois blocs distincts dont la taille et la présence varient selon les références exacte du circuit. La RAM ou mémoire vive, de 192 octets au moins, est toujours présente. La ROM ou mémoire de programme peut être présente ou absente. Elle peut être réalisée selon une des technologies dont nous venons de parler et il n'y a donc pas lieu d'y revenir. Dans tous les cas, cette mémoire destinée au programme ne doit pas être confondue avec la petite EEPROM visible sur ce synoptique qui est, elle, destinée aux données à sauvegarder en l'absence d'alimentation par exemple.
Cette unité centrale est entourée d'un certain nombre de ports parallèles baptisés port A à port E (et parfois même au delà pour certains versions récentes très riches en ports parallèles) qui peuvent être bidirectionnels ou unidirectionnels selon le cas. Certaines lignes de ces ports sont également partagées avec d'autres ressources internes et ne sont donc pas nécessairement accessibles directement en permanence. Ainsi par exemple le port E est-il commun avec le convertisseur analogique/digital.
Des entrées/sorties séries sont aussi disponibles et peuvent fonctionner en mode synchrone ou asynchrone selon que l'on utilise la SPI ou la SCI. Un timer ou plus exactement un sous-ensemble timer est également disponible. Il comporte plusieurs timers très évolués ainsi qu'un accumulateur d'impulsions, une horloge temps réel et un "chien de garde" ou COP (pour Computer Operating Properly) destiné à surveiller le fonctionnement du microcontrôleur.
Un convertisseur analogique/digital 8 bits à 8 entrées complète cette panoplie de ressources internes déjà bien riche mais n'est pas présent sur toutes les versions de 68HC11.
Toute la logique nécessaire, tant au traitement des interruptions qu'à la génération de l'horloge, est évidemment intégrée dans le 68HC11 dont la mise en oeuvre matérielle est ainsi fort simple puisque, dans de nombreux cas, un quartz et une cellule R-C pour le Reset sont les seuls composants externes indispensables pour le pilotage de cette logique.
Remarquez aussi, en partie basse de la figure 1, que le circuit peut fonctionner selon deux modes : un mode "single chip" ou circuit seul et un mode "expanded" ou étendu qui lui permet alors d'adresser des circuits externes comme n'importe quel microprocesseur ordinaire. Dans ce dernier cas, les lignes des ports B et C se transforment en lignes d'adresses et de données.

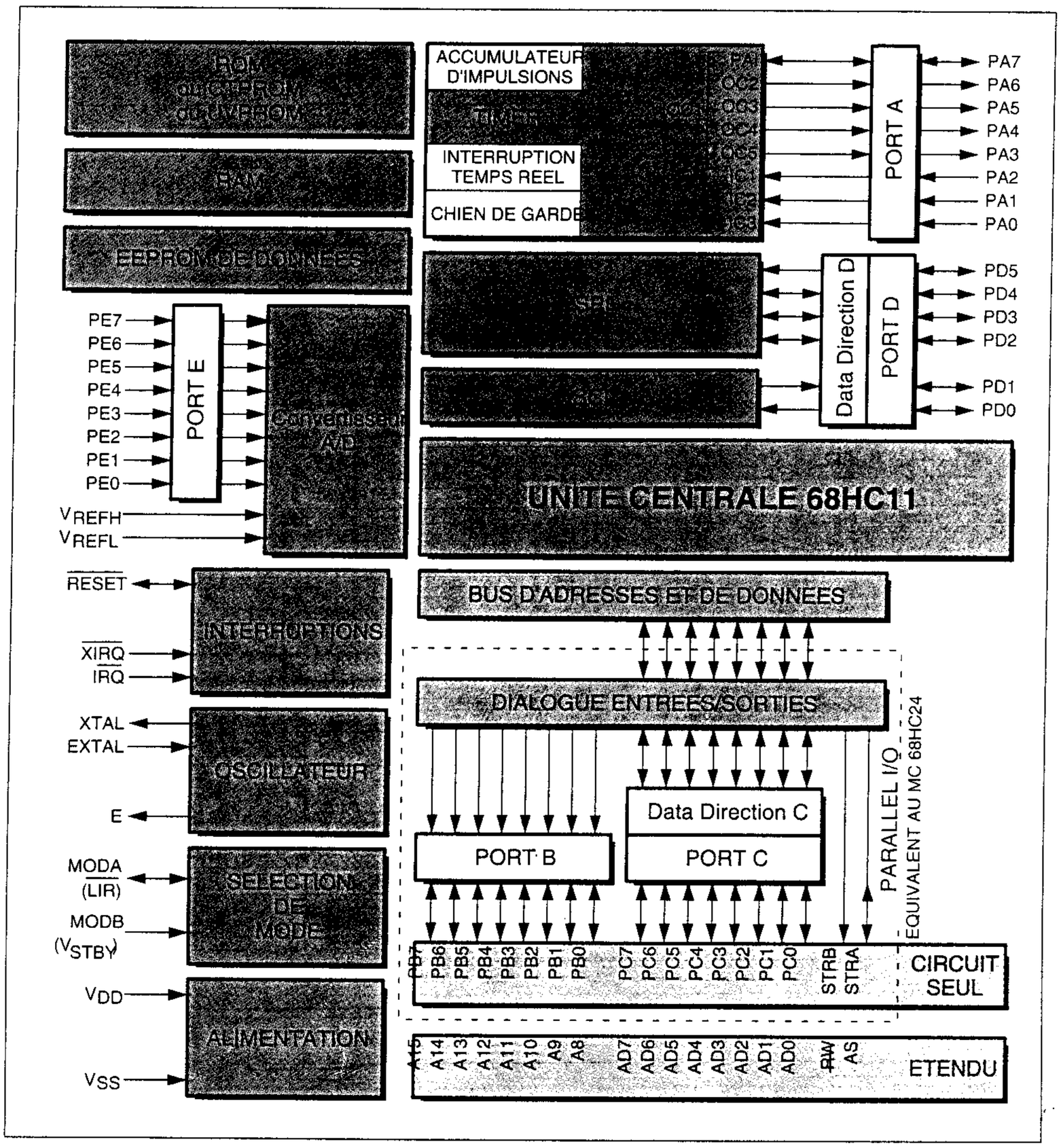


Figure 1 : Architecture interne des 68HC11.

La famille HC11

Même si la famille HC11 est aujourd'hui très riche, les versions de circuits équipées de mémoires OTPROM ou UVPROM, c'est à dire les circuits que nous pouvons programmer avec notre montage, sont au nombre de deux seulement. Fort heureusement, le choix fait par Motorola quant à l'équipement en ressources internes de ces circuits est judicieux et l'on dispose ainsi du 68HC711D3, qui est la version la plus simple mais qui permet toutefois de réaliser déjà de très belles applications, et la version 68HC711E9 qui est nettement mieux pourvue en matière de ressources internes et qui est à réserver aux applications les plus gourmandes.
Le tableau 1 précise de façon synthétique le contenu de ces deux circuits. Notez en particulier l'absence de convertisseur analogique/digital dans le 68HC711D3 et la différence de taille des mémoires de programme qui est de 4 K dans le D3 et de 12 K dans le E9. Précisons, pour terminer ce rapide tour d'horizon, que ces deux circuits coûtent entre 150 et 250 francs TTC selon la version et le boîtier à la date de publication de cet article.

Les signaux disponibles

Nous allons voir maintenant quels sont les noms et les fonctions des diverses broches que l'on peut rencontrer sur les boîtiers des circuits de la famille HC11, étant entendu que certaines pourront être absentes sur certaines versions de circuits, eu égard à leur contenu.
Précisons qu'aucun numéro de broche ne figure intentionnellement ci-après ; nous vous renvoyons pour cela à la figure 2 sur laquelle sont présentés les trois types de boîtiers dans lesquels on peut trouver les 68HC711 avec les PLCC 44 pattes et DIL 40 pattes du 711D3 et le PLCC 52 pattes du 711E9. Les noms utilisés ci-après sont ceux retenus par Motorola afin que vous ne soyez pas dépaysé si vous consultez les documentations techniques éditées par ce fabricant.
- VDD (+V) et VSS (masse) sont les pattes d'alimentation du boîtier. Bien qu'il soit réalisé en technologie HCMOS, le 68HC11 s'alimente typiquement sous 5 volts. Dans ces conditions ses entrées/sorties sont compatibles TTL-LS ou CMOS alimentés sous la même tension.
- MODA/LIR et MODB/Vstby sont des pattes à double fonction comme leurs noms le laissent supposer. Lors d'un Reset du circuit, les pattes sont à considérer comme des entrées appelées MODA et MODB et permettent de sélectionner un des quatre modes de fonctionnement du circuit comme l'indique le tableau 2. Nous reviendrons ci-après sur la signification de ces modes car c'est grâce à eux que l'on peut programmer les 68HC711. Hormis la phase de Reset, la patte MODA devient une sortie baptisée /LIR. Cette patte est au niveau bas lors du premier cycle de l'horloge E en phase d'exécution d'une instruction. Elle peut être utilisée pour synchroniser un oscilloscope ou un analyseur logique en phase de

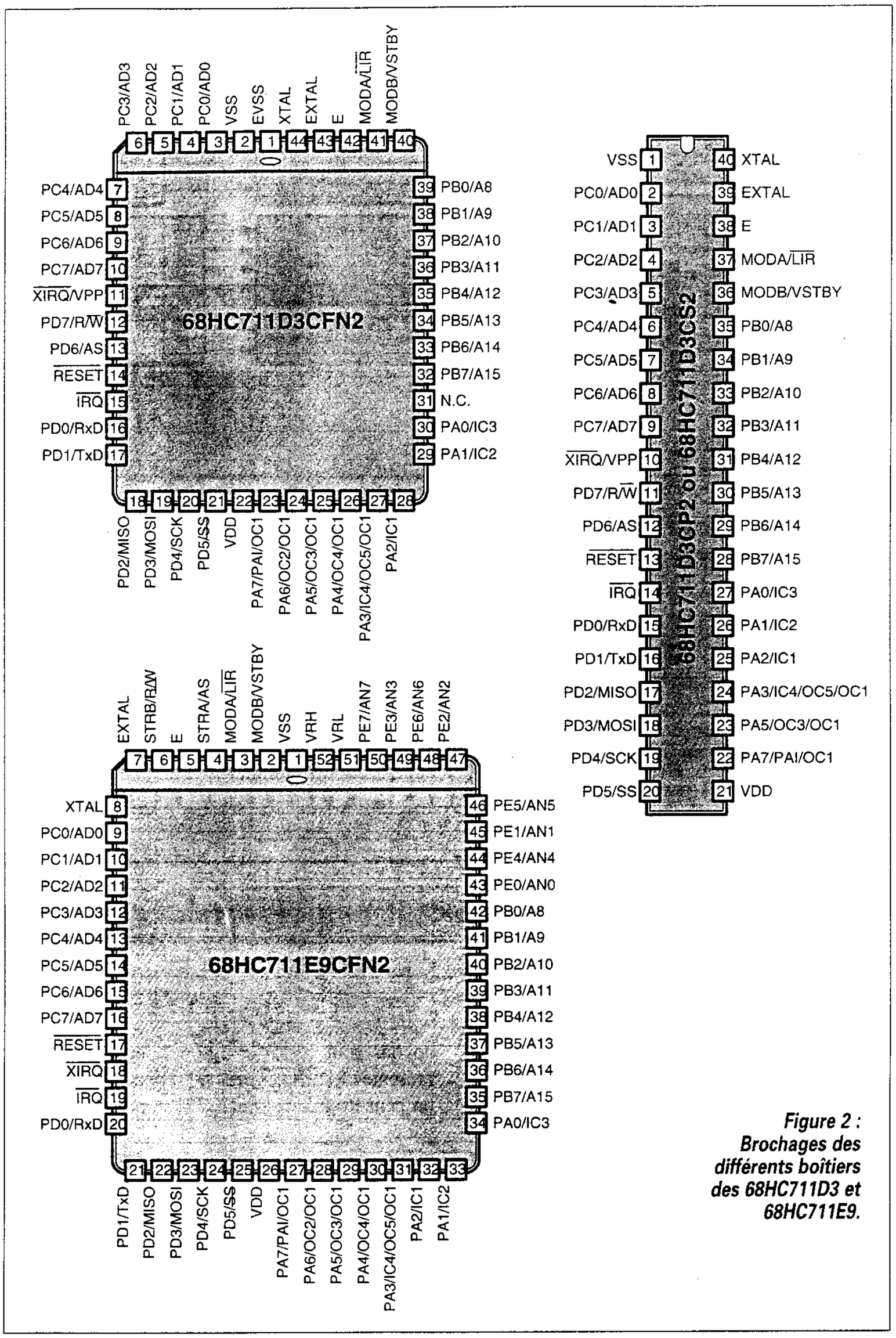


Figure 2 : Brochages des différents boîtiers des 68HC711D3 et 68HC711E9.

mise au point de programme par exemple. La patte MODB quant à elle devient une entrée d'alimentation de sauvegarde pour la RAM interne qui est activée automatiquement par un interrupteur électronique interne. - EXTAL, XTAL et E sont les broches relatives à l'horloge. EXTAL et XTAL permettent la connexion d'un quartz ou d'une horloge externe. /RESET est évidemment la broche de Reset ou de remise à l'état initial mais, contrairement à de nombreux circuits, c'est une ligne bidirectionnelle. Utilisée en entrée et forcée à la masse, elle provoque une ré-initialisation du circuit. Si par contre un événement interne anormal survient tel que, par exemple, la détection par le chien de garde ou COP d'une anomalie, ce dernier provoque un Reset interne du 68HC11 mais fait aussi générer un niveau bas à cette patte afin que cette information puisse, le cas échéant, être exploitée par des boîtiers externes.- IRQ et XIRQ sont des entrées d'interruptions externes. IRQ est une entrée d'interruption masquable alors que XIRQ est non masquable, une fois passée la phase d'initialisation du circuit. - VREFL et VREFH sont respectivement les entrées de référence basse et haute du convertisseur analogique/digital. Le convertisseur étant du type à redistribution de charge, la consommation de courant sur ces broches est quasi nulle. - PE0 à PE7 sont les entrées du port parallèle E. Ces lignes sont unidirectionnelles et ne fonctionnent qu'en entrée. De plus elles sont partagées avec les entrées du convertisseur analogique/digital mais, malgré cela, il est possible de les utiliser simultanément dans les deux modes. En effet, lors d'une entrée de donnée numérique, le buffer interne correspondant n'est mis en fonction qu'un très court instant ce qui peut introduire une légère perturbation uniquement si cela a lieu pendant une phase d'échantillonnage de l'entrée correspondante.

- PD0 à PD5 sont les entrées/sorties parallèles du port D qui sont communes avec les lignes des ports série de la SPI et de la SCI. La figure 1 vue ci-avant précise l'affectation relative de ces pattes aux différentes lignes spécifiques de ces interfaces. Il est évident qu'ici aussi toute simultanéité de fonctions est impossible sauf quelques cas particuliers. Ainsi, même avec la SPI en fonction, la ligne /SS peut être utilisée comme sortie sous réserve que l'interface fonctionne en mode maître - PC0 à PC7 sont des entrées/sorties parallèles à usage général en mode circuit seul. En mode étendu, elles deviennent les lignes où circulent les données et les huit bits de poids faibles des adresses en mode multiplexé. La ligne AS (Address Strobe) dont le rôle est alors tenu par la patte STRA indique alors si le port C véhicule des adresses (AS au niveau haut) ou si ce sont des données (AS au niveau bas). Le rôle de la ligne lecture/écriture est tenu quant à lui par la patte STRB. - PB0 à PB7 sont des sorties parallèles en mode circuit seul et véhiculent les 8 bits de poids forts des lignes d'adresses en mode étendu. Les remarques faites ci-avant pour le port C s'appliquent aussi ici. Il est évident que, si le circuit est utilisé en mode étendu, les sorties PB0 à PB7 et les entrées/sorties PC0 à PC7 sont perdues.

- STRA et STRB sont respectivement AS et R/W en mode étendu. En mode circuit seul, ces lignes peuvent être utilisées pour divers protocoles de dialogue via les ports B et C. STRA est sensible à un front et permet de mémoriser dans un registre spécial associé au port C les données appliquées sur celles des lignes de PC0 à PC7 qui ont été placées en entrées. - PA0 à PA7 sont les entrées (PA0, PA1, PA2), les sorties (PA3, PA4, PA5, PA6) et l'entrée/sortie (PA7) du port A. Ces lignes sont partagées avec celles du timer ce qui justifie le fait que certaines d'entre-elles soient unidirectionnelles. En effet, PA0 à PA2 sont aussi les entrées de capture IC3 à IC1 du timer ; PA3 à PA6 sont aussi les sorties de comparaison OC1 à OC5 du timer tandis que PA7 peut être configurée comme entrée de capture ou sortie de comparaison. Si le timer à accumulation d'impulsions est utilisé, c'est également PA7 qui lui sert d'entrée. Il est évident que si une des pattes de sortie de comparaison est validée pour cette fonction au niveau du timer, elle ne peut être utilisée en tant que patte de sortie parallèle à usage général.

Les registres de l'unité centrale

L'unité centrale du 68HC11 est un modèle 8 bits disposant de fonctionnalités particulières lui permettant d'exécuter avec un maximum de souplesse certaines instructions, y compris arithmétiques et logiques, sur 16 bits. Elle est issue de l'unité centrale qui équipait les désormais désuets 6801 mais a été enrichie de nouvelles instructions.
De ce fait, elle est capable d'exécuter, au niveau

code objet donc, tout programme écrit pour un 6800 ou un 6801 ce qui peut être intéressant pour faire évoluer rapidement une application existante. De ce fait, les programmes sources 6800 et 6801 peuvent évidemment être soumis eux aussi à un assembleur 68HC11 pour donner du code exécutable. Quatre-vingt dix nouvelles instructions étant cependant disponibles par rapport aux jeux d'instructions initiaux de ces deux microprocesseurs, il est toutefois évident qu'il sera préférable de réécrire tout ou partie d'une application existante pour bénéficier pleinement de la puissance du 68HC11.

Les registres de l'unité centrale vous sont présentés figure 3. C'est une architecture classique chez Motorola avec des registres aux fonctions bien définies contrairement à certaines familles de microcontrôleurs où tous les registres sont parfois polyvalents. On distingue ici en premier lieu deux accumulateurs A et B qui sont des registres 8 bits. Ces accumulateurs supportent toutes les instructions arithmétiques et logiques que connaît le 68HC11. Ils peuvent être utilisés de façon totalement indépendante l'un de l'autre sans restriction. Ils peuvent également être concaténés pour former un seul accumulateur qui s'appelle alors D pour Double accumulateur. Ce dernier est alors un modèle 16 bits et supporte à ce titre un certain nombre d'opérations arithmétiques et logiques à ce format.

On trouve ensuite deux registres d'index X et Y sur 16 bits puisque la capacité d'adressage du 68HC11 est de 16 lignes d'adresses ou de 65536 octets si vous préférez. Leur rôle premier est d'être utilisés pour l'adressage indexé où leur double présence facilite énormément le balayage simultané de tables pour faire du transcodage par exemple. Quelques opérations arithmétiques élémentaires peuvent également être réalisées sur ces index. En outre, si l'adressage indexé ne les utilise pas, ils peuvent bien sûr servir de registres de stockage temporaire de données.

Vient ensuite le pointeur de pile S ou SP (pour Stack Pointer), également sur 16 bits afin de pouvoir accéder à tout l'espace mémoire adressable. Ce registre pointe sur la première adresse libre de la zone mémoire définie pour ranger la pile. Il est donc indispensable que, très rapidement suite à la sortie du Reset, ce registre soit initialisé avec une adresse de RAM valide faute de quoi, à la première interruption ou au premier appel de sous-programme toute adresse de retour sera perdue. Le 68HC11 gère automatiquement ce registre c'est à dire qu'en cas d'appel de sous-programme ou d'interruption, il range sur la pile le contenu de certains registres internes et décrémente le registre S d'autant d'unités que nécessaire de façon à ce qu'il pointe toujours sur la prochaine adresse de pile disponible. Réciproquement, au retour de sous-programme ou d'interruption, il recharge les registres internes qui avaient été sauvegardés avec les informations reprises sur la pile et incrémente le registre S d'autant d'unités que nécessaire. De plus, vous disposez d'instructions de manipulation de ce registre si nécessaire.

Le registre PC ou Program Counter n'est autre que ce que l'on devrait appeler en français compteur ordinal. Il pointe sur la prochaine instruction

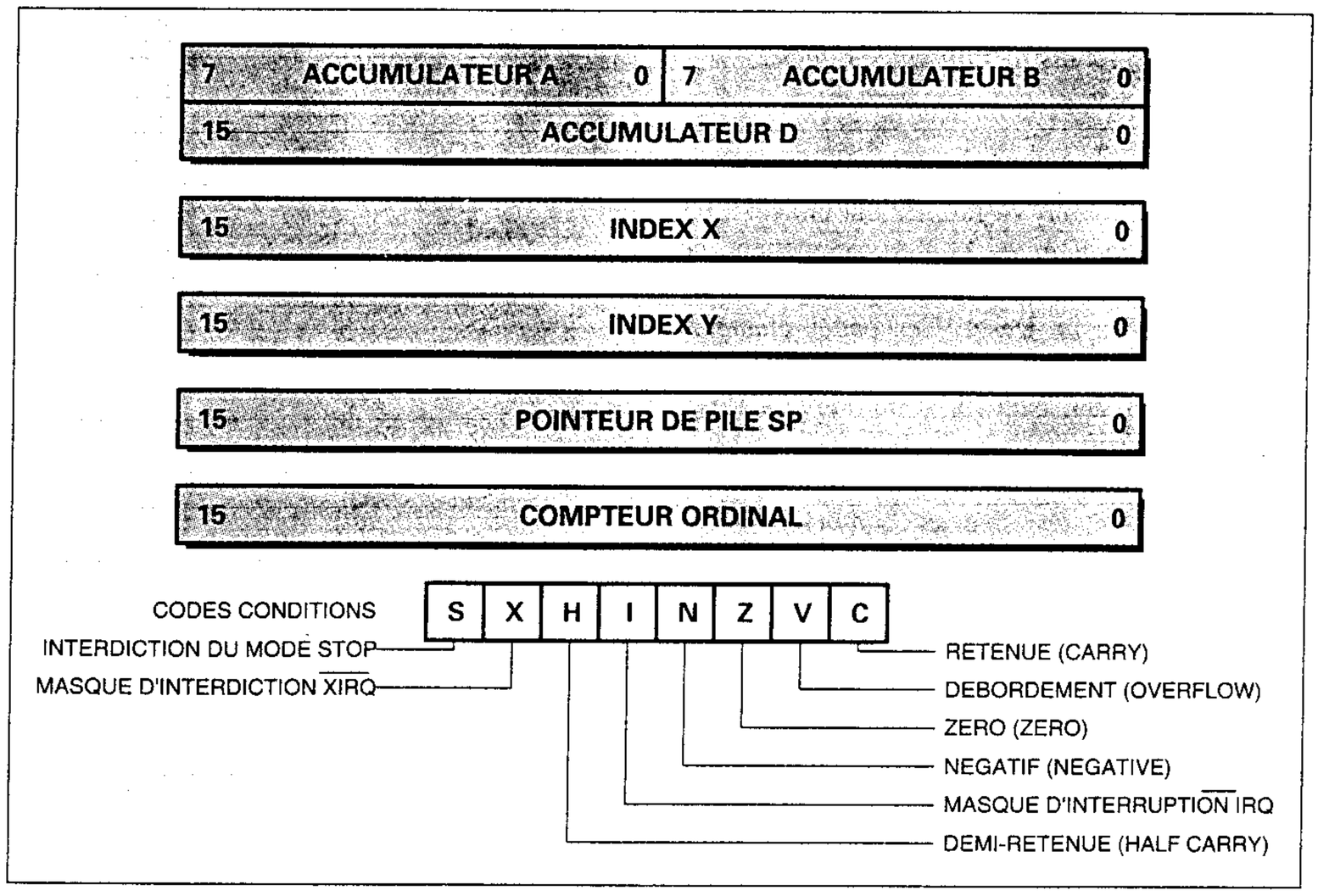


Figure 3 : Les registres de l'unité centrale du 68HC11.

à exécuter et sa taille est donc nécessairement de 16 bits pour pouvoir accéder à tout l'espace mémoire adressable. Son contenu peut être modifié indirectement par l'utilisateur lors des instructions de saut et de branchement ou lors des appels de sous-programmes ou d'interruptions.

Enfin, le dernier registre est le registre d'état ou CCR pour Condition Code Register selon une appellation propre à Motorola. Bien qu'étant un registre 8 bits, ce CCR doit être examiné bit par bit, chacun ayant une signification particulière :

- Le bit C pour Carry ou retenue est mis à 1 lorsqu'une opération arithmétique génère une retenue. Il est également utilisé comme indicateur d'erreur lors d'une multiplication ou d'une division et sert lors de certaines opérations de décalage ou rotation qui peuvent passer par son intermédiaire ou non.
- Le bit V pour oVerflow ou débordement est positionné à 1 lorsqu'une opération arithmétique a généré un débordement de l'accumulateur.
- Le bit Z pour Zero est positionné à 1 lorsque le résultat de l'instruction exécutée est nul. Contrairement aux deux bits précédents, il est affecté par de très nombreuses instructions et non seulement par des instructions arithmétiques par exemple.
- Le bit N pour Negative est positionné à 1 si le résultat de la dernière opération arithmétique réalisée est négatif c'est à dire si le bit de poids fort du résultat est à 1.
- Le bit H pour Half carry ou demie retenue est positionné à 1 lors d'une retenue entre les bits 3 et 4 d'une opération arithmétique. Il n'est affecté que par les instructions ADD, ADC et ABA et est ensuite exploité par l'instruction DAA pour réaliser de l'arithmétique DCB c'est à dire codée en fait sur 2 groupes de 4 bits.
- Le bit I pour Interrupt mask ou masque d'interruption interdit toute interruption masquable lorsqu'il est mis à 1. Les interruptions survenant pendant qu'il est à 1 peuvent cependant rester en attente si le périphérique interrupteur le permet. Dans ce cas elles seront traitées dès le retour à zéro de ce bit.
- Le bit X pour XIRQ interrupt mask fonctionne comme le bit I vu ci-avant mais pour les interruptions susceptibles d'être appliquées sur l'entrée XIRQbarre. Cette entrée, qui porte le nom de non masquable l'est donc en fait via ce bit mais uniquement suite à un Reset et tant que le pointeur de pile n'est pas initialisé.
- Le bit S pour Stop disable permet, lorsqu'il est mis à 1 d'interdire, l'exécution de l'instruction STOP. Celle-ci est alors considérée par le 68HC11 comme un simple NOP (No Operation,). Certains utilisateurs considèrent en effet que STOP est une instruction dangereuse car elle arrête l'oscillateur du 68HC11 d'où l'intérêt de ce bit.

Précisons pour en terminer avec ce CCR que même si certains de ses bits sont positionnés automatiquement par l'unité centrale en fonction des instructions exécutées, vous pouvez égale-

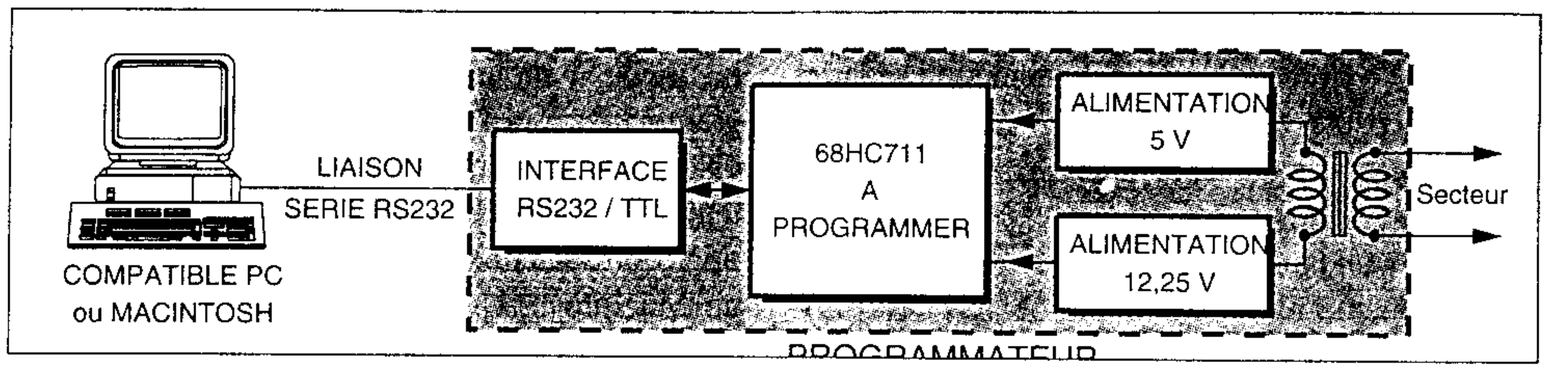


Figure 4 : Synoptique de principe de notre programmateur.

ment manipuler la majorité d'entre eux directement au moyen d'instructions spécifiques.

Les trois modes de fonctionnement principaux

Comme nous l'avons vu rapidement ci-avant, les deux pattes MODA et MODB permettent de sélectionner le mode de fonctionnement du 68HC11 suite à un Reset. Ces pattes changent ensuite de fonction et ne permettent plus de modifier ce dernier sauf bien sûr si un nouveau Reset externe intervient.

Bien que quatre modes distincts puissent être sélectionnés par les diverses combinaisons de MODA et MODB, deux modes doivent être considérés comme principaux, un comme accessoire et le dernier comme mode de test, d'usage exceptionnel pour l'utilisateur final du circuit.

Le tableau 2 rappelle les combinaisons de MODA et MODB que nous allons maintenant commenter. En mode normal circuit seul, le circuit est configuré de façon à disposer de ses ports B et C en tant qu'entrées/sorties parallèles comme nous l'avons vu ci-avant. C'est le mode d'utilisation typique de toute application n'exploitant pas le bus externe du 68HC11.

En mode normal étendu, le circuit est configuré de façon à faire sortir ses bus d'adresses et de données sur les ports B et C selon la répartition vue ci-avant. C'est le mode d'utilisation typique de toute application exploitant un ou plusieurs boîtiers externes ayant besoin de dialoguer sur le bus du 68HC11. Le mode appelé spécial bootstrap est en fait un mode de mise au point de programme ou de programmation du circuit qui fonctionne de la façon suivante. Dans ce mode, le 68HC11 n'exécute pas le programme contenu dans sa ROM et pointé par son vecteur de Reset normal mais un "bootloader" contenu à une adresse particulière d'une ROM interne inaltérable. Il initialise alors l'interface SCI de façon à ce qu'elle puisse recevoir un programme de 256 octets de long ou moins qui est chargé en RAM à partir de l'adresse $0000. Lorsque ce chargement est terminé, le 68HC11 saute automatiquement à l'adresse $0000 et exécute donc le programme qui vient ainsi d'y être placé. En fait, en raison de la présence de mécanismes de sécurité relatifs au contenu de l'EEPROM, ce processus se déroule exactement de la façon suivante. Après un Reset dans ce mode, la SCI est initialisée avec une vitesse de transmission égale à la fréquence de l'horloge E divisée par 256 soit 7812 bauds à 2 MHz (c'est à dire pour un quartz classique à 8 MHz). Si la sécurité est validée, un $FF est envoyé et l'EEPROM est alors effacée. Si l'effacement échoue, un nouvel $FF est envoyé et l'effacement est répété. Lorsque l'effacement a réussi, la RAM est ensuite remplie de $FF et le registre CONFIG est effacé. Lorsque ces opérations sont terminées, ou directement si la sécurité n'avait pas été validée, un break est émis sur la liaison série. En retour le 68HC11 attend un $FF soit à 7812 bauds ce qui est une vitesse on ne peut plus non standardisée, soit à 1200 bauds ce qui est tout de même plus naturel. Le 68HC11 reconfigure alors sa SCI en fonction de la vitesse de réception de ce $FF et le chargement du programme peut commencer. Certaines versions de 68HC11 imposent que ce programme contienne exactement 256 octets qui seront placés les uns à la suite des autres à partir de la RAM d'adresse $0000 faute de quoi le bootloader restera indéfiniment en attente des octets manquants. Les versions 711D3 et 711E9 qui nous intéressent autorisent par contre un programme de taille variable ; le bootloader sautant à son adresse de début après un temps d'attente égal à la durée de quatre caractères à la vitesse de transmission choisie. Par ailleurs, la RAM du 711D3 ne commençant qu'en $0040, c'est évidemment à partir de cette adresse, et non à partir de $0000, qu'est placé le programme chargé par le bootloader. Le dernier mode dont nous n'avons pas encore parlé est le mode spécial de test. C'est en principe un mode réservé à Motorola lors des phases de test du circuit. L'utilisateur peut y faire appel très exceptionnellement mais, de nombreuses "bêtises" pouvant être faites à cette occasion, comme la transformation des lignes d'adresses en un compteur permanent et ininterruptible par exemple ; il est fortement déconseillé d'y faire appel.

	711D3	711E9
Mémoire de programme	OTPROM ou UVPROM 4 K	OTPROM 12 K
RAM	192 Octets	512 Octets
EEPROM		512 Octets
E/S Parallèles	32	38
SPI	Oui	Oui
SCI (Série Asynchrome)	Oui	Oui
CAN (Convertisseur A-N)		8 canaux 8 bits
Timer	16 bits 3/4 IC 4/5 OC RTI ACCU d'impulsions	16 bits 3/4 IC 4/5 OC RTI ACCU d'ilmpulsions
Boîtier	DIL 40 pattes PLCC 44 pattes	PLCC 52 pattes

Tableau 1 - Ressources internes des 68HC711D3 et 68HC711E9.

MODB	MODA	MODE
1	0	NORMAL - CIRCUIT SEUL
1	1	NORMAL - ÉTENDU
0	0	SPÉCIAL - BOOSTRAP
0	1	SPÉCIAL - TEST

Tableau 2 - Les différents modes de fonctionnement du 68HC11.

Principe de notre programmateur

La figure 4 montre le synoptique de notre programmateur qui fait appel à un micro-ordinateur disposant d'un simple port série RS 232. Compte tenu du logiciel que nous vous proposons, ce micro-ordinateur peut être un compatible PC ou un MacIntosh. Un circuit d'interface RS 232 et deux alimentations, une de 5 volts et une "haute tension" de 12,25 volts, sont associés au 68HC711 sur sa carte de programmation et c'est tout ! Pour bien comprendre le principe utilisé il faut savoir que tous les 68HC711 comportent, dans une ROM interne inaltérable, un programme de programmation de leur UVPROM ou OTPROM. Il suffit donc de se "débrouiller" pour fournir à ce programme les informations dont il a besoin, c'est à dire en fait le programme que vous souhaitez placer en UVPROM et OTPROM, et de lancer son exécution. Il faut pour cela faire appel au mode bootstrap. Le logiciel de programmation tournant sur le micro-ordinateur fonctionne donc de la façon suivante. Il fait appel au bootloader du 68HC711 pour charger en RAM un très court programme appelé chargeur. Ce chargeur contient les paramètres nécessaires à la programmation à savoir : la taille du programme à placer en UVPROM ou OTPROM et l'adresse de début de ce programme. Accessoirement il initialise aussi le port C pour la commande d'une LED d'indication d'état. Lorsque le 68HC711 a reçu ce chargeur placé en RAM, il y fait appel, ce qui lance le programme de programmation de l'UVPROM ou OTPROM qu'il contient dans sa ROM interne. Ce dernier dialogue ensuite avec le micro-ordinateur afin de suivre le processus de programmation et d'en permettre la vérification.

Conclusion

Ces quelques explications vous auront sans doute permis de comprendre que le schéma du programmateur allait être fort simple et c'est effectivement le cas. Vous le découvrirez dans notre prochain numéro avec les dessins des circuits imprimés nécessaires ainsi que le listing source du logiciel à utiliser sur le micro-ordinateur.

Nous vous indiquerons également où vous procurer un assembleur 68HC11 gratuit afin que vous soyez en mesure de réaliser avec succès vos propres applications ; à moins bien sûr que vous préfériez attendre celles que nous ne manquerons pas de vous proposer.

C. Tavernier

réalisation

Programmateur de 68HC11

2e PARTIE

Après avoir vu le mois dernier les principes qui régissent notre programmateur, nous pouvons maintenant passer à sa réalisation avec, tout d'abord, l'étude de son schéma.

Schéma théorique

Celui-ci vous est présenté dans son intégralité en figure 1 ; reconnaissez qu'il est difficile de faire plus simple. Un transformateur à secondaire à point milieu est utilisé de façon non conventionnelle de manière à disposer de deux tensions destinées aux deux alimentations stabilisées nécessaires.

La tension prélevée au point milieu du transformateur alimente le régulateur intégré IC1 qui délivre une tension stabilisée de 5 volts au reste du montage. Par contre, la tension prélevée à l'extrémité du deuxième enroulement alimente le régulateur ajustable IC2, un LM 317 en l'occurrence, qui fournit la «haute tension» de programmation.

Cette dernière devant être très précisément égale à 12,25 volts, nous avons prévu son ajustement grâce au potentiomètre P1. Afin de préserver le 68HC711, cette tension ne lui est pas appliquée directement mais passe par un interrupteur qui n'est manoeuvré qu'au moment opportun lorsque le logiciel de programmation le demande.

Le coeur du montage est évidemment le 68HC711 à programmer lui-même puisque, si vous avez suivi nos explications théoriques, vous savez aujourd'hui qu'il contient son propre logiciel de programmation.

Seules quelques pattes du boîtier sont utilisées. L'horloge tout d'abord est pilotée par quartz afin de disposer de chronogrammes internes précis, ce qui est indispensable à une bonne programmation de la mémoire, qu'elle soit de type UVPROM ou OTPROM. La fréquence de 8 MHz ne doit pas être modifiée car elle conditionne certaines durées critiques de ces chronogrammes.

Figure 1 : Schéma complet du programmateur.

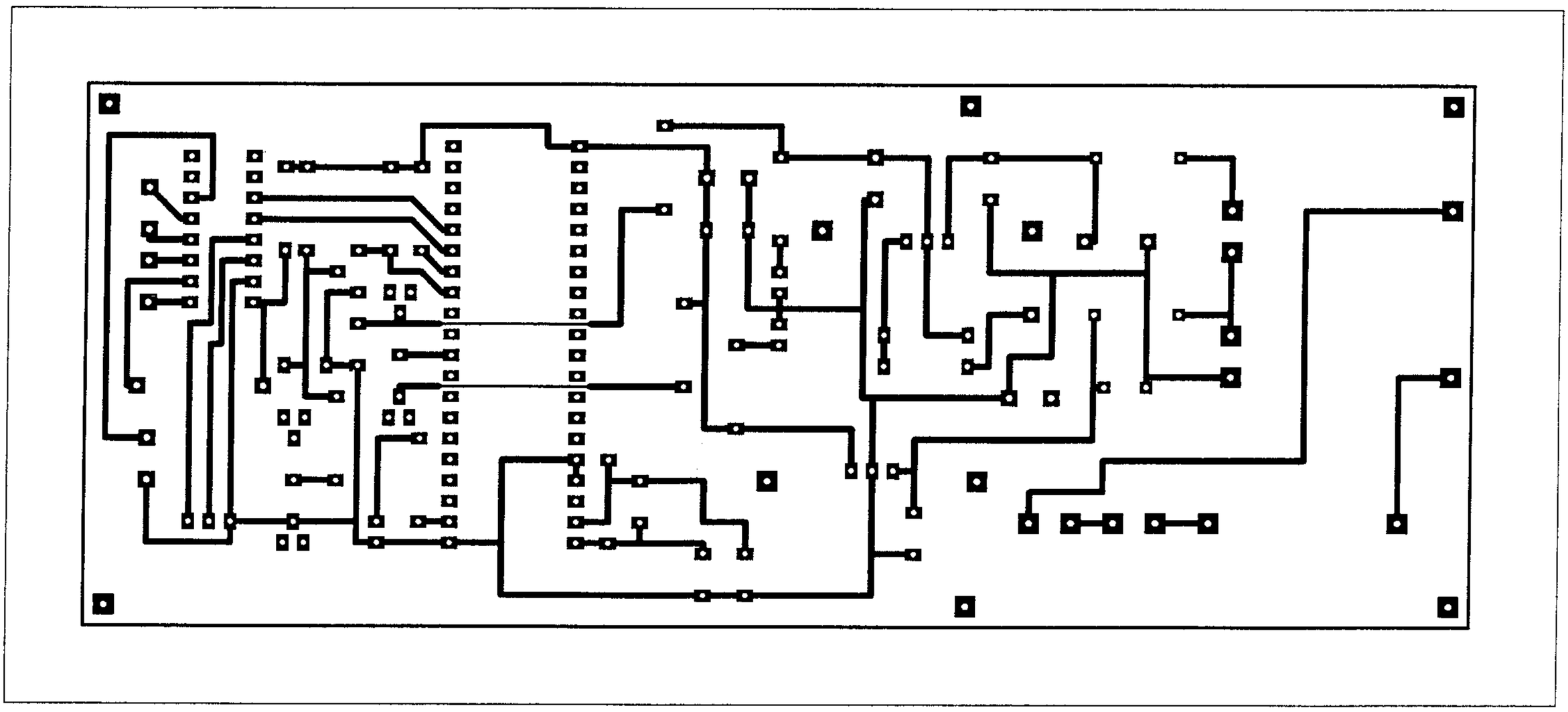


Figure 2 : Circuit imprimé principal, vu côté cuivre, échelle 1.

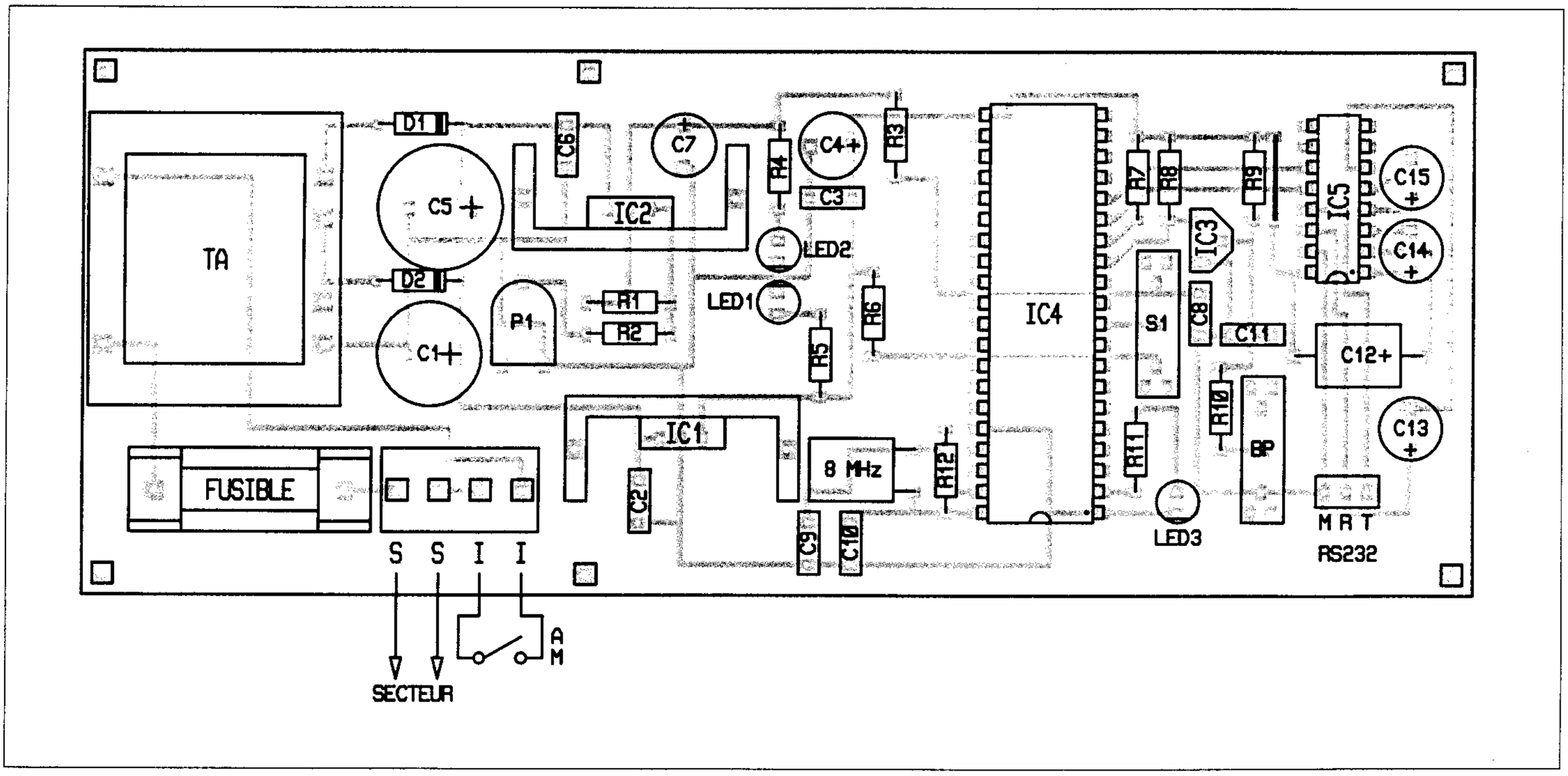


Figure 3 : Implantation des composants.

La circuiterie de Reset fait appel à un circuit spécialisé qui assure tout à la fois un Reset automatique à la mise sous tension et un Reset manuel au moyen d'un poussoir. Ce circuit, fort peu coûteux puisqu'il vaut moins de 20 francs, est préférable et de loin à la simple cellule R-C que l'on voit trop souvent sur la patte de Reset du 68HC11. En effet, cette patte n'est pas munie d'un vrai trigger de Schmitt et les Resets obtenus de la sorte ne sont pas forcément corrects.

L'entrée d'interruption externe masquable IRQ barre est reliée par une résistance de tirage au + 5 volts afin de ne pas risquer de perturber le circuit par une interruption parasite pendant la programmation. L'entrée non masquable XIRQ barre est traitée de la même façon mais, comme cette patte joue un double rôle sur les 68HC711, elle peut aussi être reliée à la «haute tension» de programmation par la manoeuvre de S1 lorsque c'est nécessaire. La ligne de poids le plus faible du port C - PC0 - est reliée à une LED qui permet de constater par son allumage que le 68HC711 a bien reçu le programme chargeur et a bien lancé son exécution (voir notre précédent article si nécessaire). Enfin, les lignes PD0 et PD1 qui ne sont autre que les entrée et sortie série asynchrones de la SCI du 68HC711 sont reliées à un circuit d'interface RS 232. Nous avons fait appel à un classique MAX 232 qui permet de disposer de vrais niveaux RS 232 sans avoir besoin d'une alimentation négative. Il comporte en effet ses propres convertisseurs de tension statiques internes à capacités commutées. Précisons à ce propos pour couper court aux éventuelles questions que les polarités de C12 et de C13 sont correctes sur le schéma. Le pôle négatif de C12 est bien relié au positif de l'alimentation et le pôle positif de C13 est bien relié à la masse !

Signalons pour terminer cet examen que le brochage du 68HC711 indiqué sur la figure 1 correspond au boîtier DIL 40 pattes ; en effet, comme nous allons le voir avec la description de la réalisation pratique, différents supports adaptateurs sont prévus pour la programmation des boîtiers PLCC à 44 et 52 pattes.

La réalisation

L'approvisionnement des composants ne pose pas de problème particulier si ce n'est peut être pour IC3 que certains revendeurs ne possèdent pas toujours en stock. Sachez que vous pouvez trouver ce circuit chez Radiospares (BP 453, 60031 Beauvais Cedex) qui pratique la vente par correspondance sans minimum de facturation.
Le 68HC711 à acheter sera fonction de ce que vous voulez faire. A l'heure actuelle, quatre versions sont utilisables sur notre programmateur :
- le 68HC711D3CP2 qui est un 711D3 en boîtier DIL plastique à 40 pattes contenant de la mémoire OTPROM (programmable une fois mais non effaçable donc).
- Le 68HC711D3CS2 qui est lui aussi un 711D3 bien sûr mais en boîtier céramique à fenêtre. Sa mémoire est de type UVPROM et peut donc être effacée aussi souvent que nécessaire avec n'importe quel effaceur à ultraviolet.
Malgré son prix (250 francs environ) c'est le boîtier que nous vous recommandons si vous voulez développer vous même vos propres applications. Sa possibilité d'effacement permet en effet tous les essais et toutes les mises au point de programmes souhaitables.
- Le 68HC711D3CFN2 qui est lui aussi un 711D3 muni de mémoire OTPROM (programmable une fois donc) mais en boîtier PLCC à 44 pattes. Il est donc plus intéressant que le 68HC711D3CP2 si vous avez des problèmes d'encombrement.
- Le 68HC711E9CFN2 enfin qui est un 711E9, c'est à dire le circuit le plus richement doté en matière de ressources internes (voir notre précédent numéro si nécessaire), muni de mémoire OTPROM (programmable une fois) en boîtier PLCC à 52 pattes.
Les supports de circuits intégrés requièrent une attention particulière.
Respectez le nombre de supports indiqué dans la nomenclature des composants même s'il vous semble anormalement élevé. Certains d'entre eux sont en effet utilisés pour la réalisation des adaptateurs PLCC.
Par ailleurs, comme ces supports seront souvent manipulés et seront donc soumis à rude épreuve, choisissez des modèles à contacts tulipes.
Pour la carte principale du programmateur, et si votre budget le permet, investissez dans un support DIL à 40 pattes à force d'insertion nulle (et non à insertion nulle comme on le lit trop souvent dans des documentations dont les auteurs ignorent le sens des mots !).
Si vous trouvez ce composant trop coûteux, oubliez-le pour l'instant mais n'achetez pas de support 40 pattes pour le remplacer, il est prévu dans le nombre de 40 pattes «normaux» indiqué.
Pour les adaptateurs PLCC, et même si des supports PLCC à force d'insertion nulle existent, nous n'avons même pas envisagé d'en acheter car le prix des plus «économiques» que nous avons pu trouver dépassait déjà les 1000 francs !
Il faut donc utiliser des supports PLCC «normaux» que vous choisirez évidemment de la meilleure qualité possible puisqu'ils ne sont en principe pas prévus pour subir des mises en place et extractions de circuits répétées.
Pensez à vous munir en même temps de la pince d'extraction «qui va bien» (on en trouve à moins de 100 francs) car le classique tournevis qui fait levier dans les angles a tôt fait de casser le support !
Ceci étant vu, vous pouvez passer au dessin du circuit imprimé de la carte du programmateur dont le tracé à l'échelle 1 vous est proposé figure 2. Il ne présente pas de difficulté si ce n'est de bien soigner les deux pistes fines qui passent entre les pattes du 68HC711.
Pour des raisons de simplicité de réalisation et de minimisation du câblage, cette carte supporte tous les composants du montage comme vous pouvez le constater à l'examen du plan d'implantation de la figure 3. Attention à l'emplacement du commutateur S1 et du poussoir P, pensez à retoucher notre dessin si les modèles que vous avez pu vous procurer n'ont pas la même taille que les nôtres.
L'implantation des composants est à faire dans l'ordre classique, composants passifs et supports en premier et composants actifs ensuite. Pour ce qui est du support 40 pattes, vous implanterez un modèle à contacts tulipes «normal», que vous ayez choisi ou non la solution du support à force d'insertion nulle.
Les régulateurs sont montés sur des petits radiateurs verticaux en forme de U de quelques cm². Une place suffisante est prévue à cet effet sur le circuit imprimé. La liaison au secteur et à l'interrupteur marche/arrêt est prévue au moyen de pastilles espacées de façon à pouvoir recevoir des borniers à vis standards au pas de 5 ou de 5,08 mm.
Si vous avez choisi d'utiliser un support à force d'insertion nulle, vous avez certainement remarqué que ses pattes sont à section rectangulaire et ne rentrent pas dans un perçage standard de circuit imprimé (sauf quelques modèles assez peu répandus).
Nous avons donc résolu le problème de la façon suivante qui présente en outre l'avantage de permettre d'utiliser ce sup-

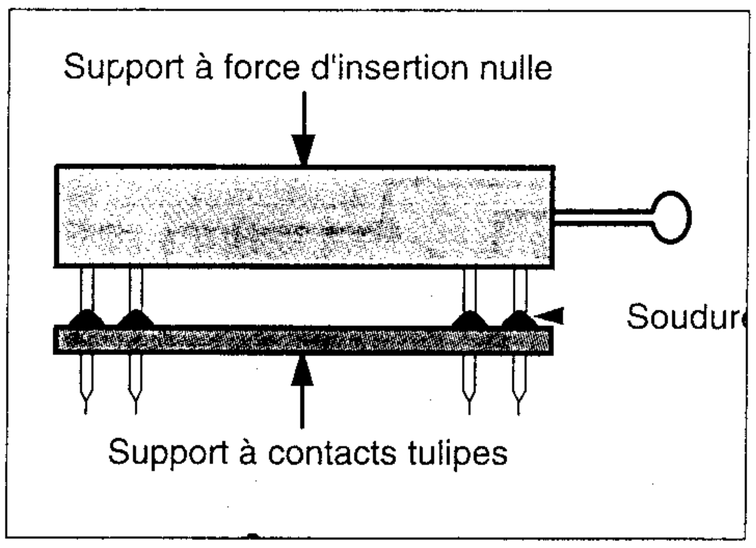


Figure 4 : Principe de montage du support DIL à force d'insertion nulle.

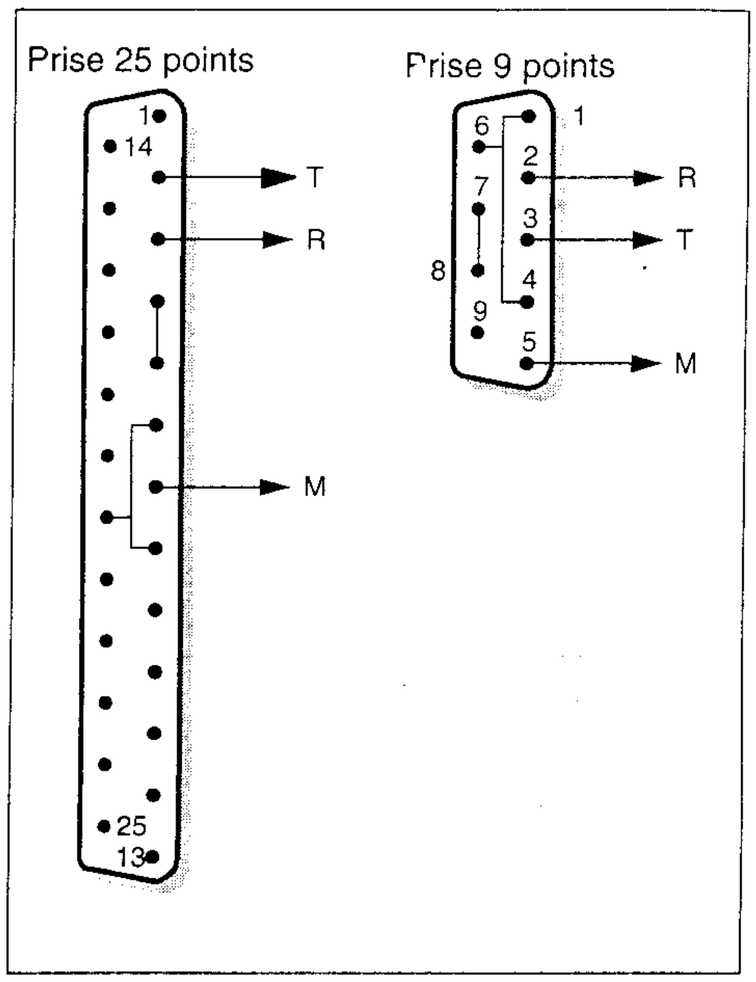
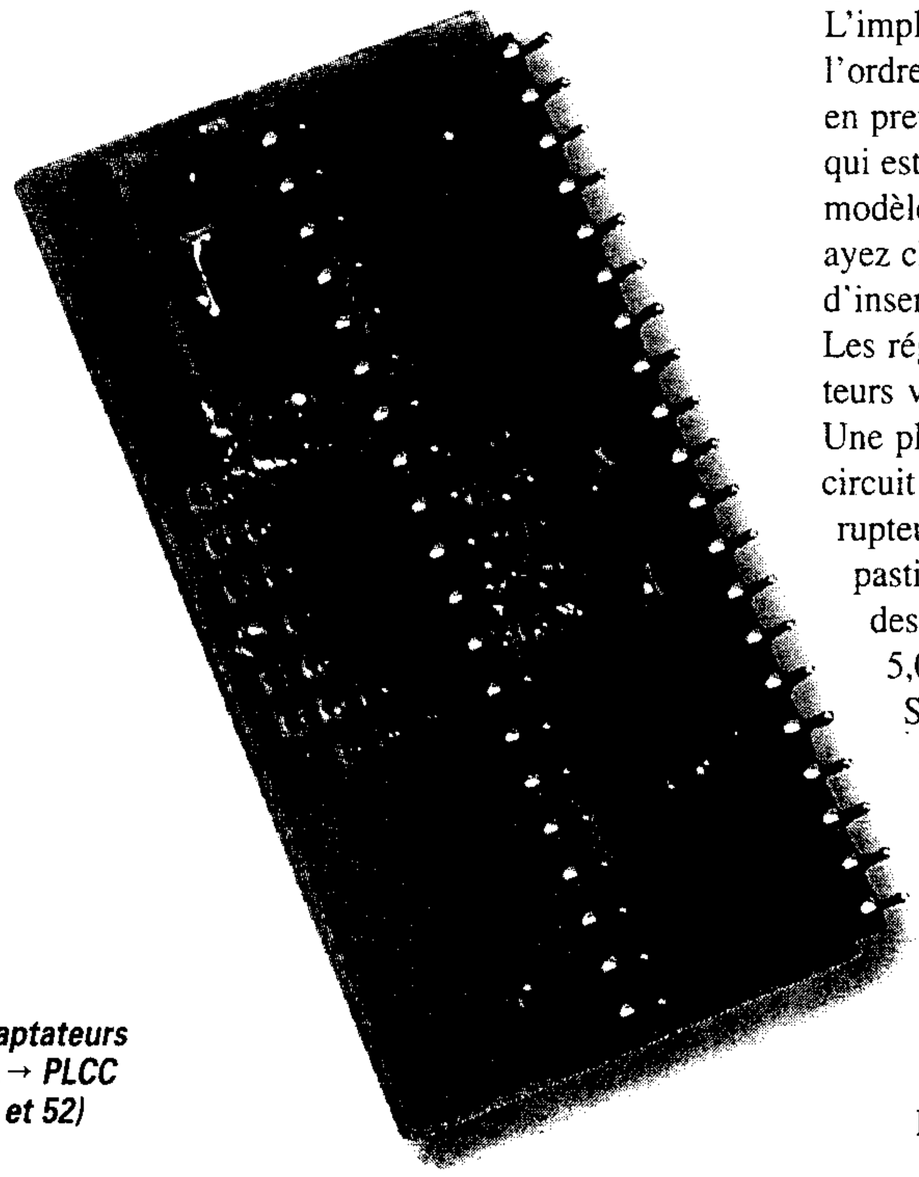


Figure 5 : Brochages des connecteurs séries à 9 et 25 points des compatibles PC et straps à réaliser.



Adaptateurs DIL → PLCC (44 et 52)

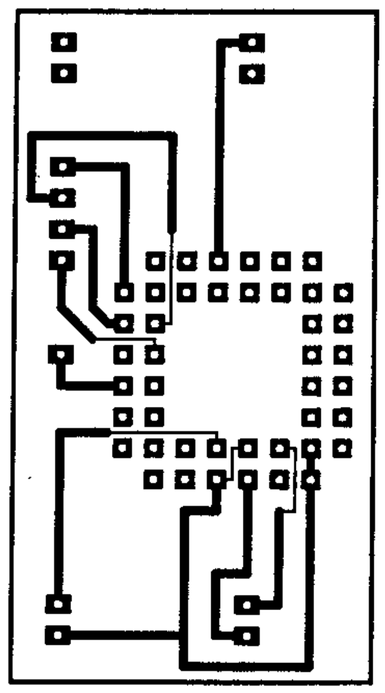


Figure 6 : Circuit imprimé de l'adaptateur PLCC à 44 pattes, vu côté cuivre, échelle 1.

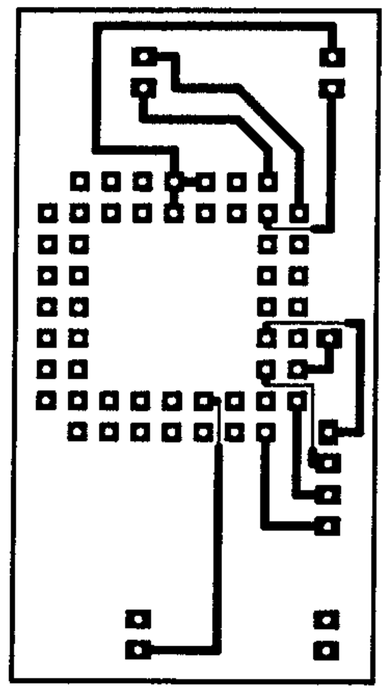


Figure 7 : Circuit imprimé de l'adaptateur PLCC à 52 pattes, vu côté cuivre, échelle 1.

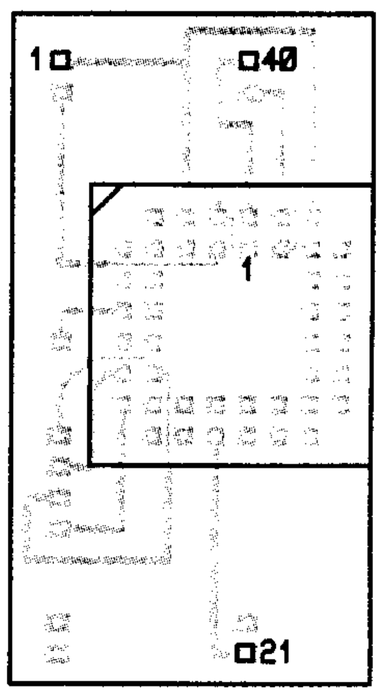


Figure 8 : Implantation des composants sur l'adaptateur PLCC à 44 pattes.

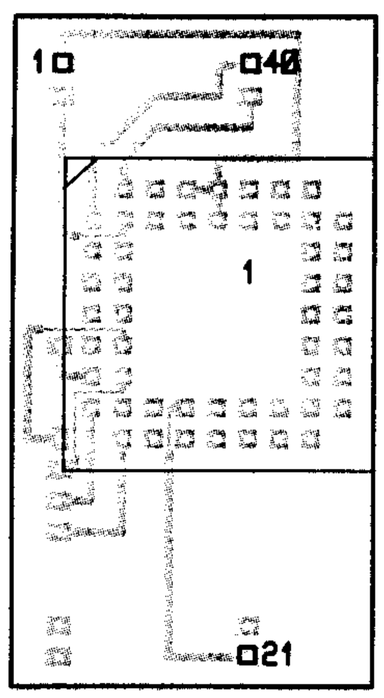


Figure 9 : Implantation des composants sur l'adaptateur PLCC à 52 pattes.

port ailleurs (sur la maquette recevant votre HC711 pendant sa mise au point par exemple).
Le support à force d'insertion nulle est tout simplement soudé, patte par patte, sur un support tulipe à 40 pattes comme cela est schématisé figure 4.
Cet ensemble peut ainsi s'enficher ensuite très facilement dans n'importe quel type de support DIL à 40 pattes.
Il faut juste faire attention, lors de cette opération de soudure, à ne pas faire de ponts entre pattes voisines et à ce que chaque patte du support à force d'insertion nulle soit bien soudée dans le contact tulipe correspondant.

Les essais

A ce stade de la réalisation, les essais se bornent à mettre le montage sous tension et à vérifier que les deux LED vertes LED1 et LED2 s'allument.

Constatez que vous avez bien 5 volts en sortie de IC1 et ajustez le potentiomètre P1 pour avoir 12,25 volts en sortie de IC2.
Préparez ensuite le câble de liaison au micro-ordinateur. Comme seules les lignes Rxd et Txd de l'interface RS 232 sont utilisées, il faut «tromper» les signaux de contrôle en réalisant des straps au niveau des connecteurs à 9 ou 25 points de la liaison série.
La figure 5 précise les brochages à respecter et les straps à mettre en place dans le cas des compatibles PC.
Ceci étant vu, vous pouvez alors passer à la phase logicielle qui consiste tout simplement à saisir le listing présenté dans ces pages.
Il est écrit en Quickbasic et a été testé avec succès sur un compatible PC et sur un MacIntosh.
Compte tenu de sa structure, seules deux lignes changent selon le type de machine utilisée :
- pour un compatible PC vous saisirez le programme tel quel et éliminerez les lignes de REM 1465 et 1505.
- pour un MacIntosh, vous enlèverez les REM des lignes 1465 et 1505 ; lignes qui resteront bien sûr dans le programme et, soit vous placerez un REM au début de 1460 et de 1500, soit vous enlèverez purement et simplement ces deux lignes.
Une fois ce programme saisi et vérifié, vous pouvez relier le programmateur à un port série de votre micro-ordinateur (1 ou 2, le programme

NOMENCLATURE DES COMPOSANTS

Semi-conducteurs
- IC1 : 7805 (régulateur + 5 volts, 1 ampère en boîtier TO 220)
- IC2 : LM 317
- IC3 : MC 33164
- IC4 : 68HC711 à programmer (voir texte)
- IC5 : MAX 232
- D1, D2 : 1N 4004
- LED1, LED2 : LED vertes
- LED3 : LED rouge

Résistances 1/4 de watt 5%
- R1 : 220 Ω
- R2 : 1,8 kΩ
- R3 : 100 Ω
- R4 : 560 Ω
- R5 : 180 Ω
- R6, R7, R8, R9, R10 : 4,7 kΩ
- R11 : 1 kΩ
- R12 : 10 MΩ

Condensateurs
- C1 : 1000 µF 25 volts chimique radial
- C2, C6 : 0,22 µF mylar
- C3, C8 : 10 nF céramique
- C4, C7, C13, C14, C15 : 10 µF 25 volts chimique radial
- C5 : 1000 µF 40 volts chimique radial
- C9, C10 : 22 pF céramique
- C11 : 1 µF mylar
- C12 : 10 µF 25 volts chimique axial

Divers
- P1 : potentiomètre ajustable horizontal de 470 ohms
- TA : transformateur moulé 220 volts 2 fois 9 volts 3,2 VA environ
- S1 : commutateur à implanter sur CI, 1 circuit 2 positions
- BP : poussoir à implanter sur CI, 1 contact travail (en appuyant)
- Quartz 8 MHz en boîtier HC 18/U ou équivalents
- Supports de CI DIL (voir texte) : 1 x 16 pattes, 4 x 40 pattes
- Supports de CI PLCC : 1 x 44 pattes, 1 x 52 pattes
- Support à force d'insertion nulle (facultatif) : 1 DIL à 40 pattes
- Picots à souder mâle - mâle (26)
- Support de fusible pour CI
- Fusible T20 0,1 ampère retardé
- Bornier à vis pour CI à 4 contacts
- Radiateurs pour IC1 et IC2
- Prise 9 ou 25 points mâle ou femelle selon port série du micro-ordinateur.

vous laissera le choix) et passer à des essais plus démonstratifs que nous vous proposons de découvrir en même temps que le mode d'emploi du programmateur.

Utilisation du programmateur

Le programmateur est supposé relié à un port série du micro-ordinateur au moyen du câble que vous avez réalisé.

Placez alors sur son support le 68HC711 à programmer, soit directement si c'est un modèle en boîtier DIL, soit au moyen d'un adaptateur (décrit ci-après) si c'est un boîtier PLCC. Assurez-vous que la haute tension est bien coupée au moyen de S1 (levier côté opposé à la LED rouge pour la majorité des interrupteurs).

Mettez le programmateur sous tension. Les deux LED vertes doivent être allumées et la LED rouge doit être éteinte.

Lancez alors le programme sur le micro-ordinateur. Il vous demande tout d'abord le type de processeur à programmer (D3 ou E9) et vous rappelle ensuite les valeurs par défaut qu'il va utiliser (taille de programme maximum débutant à son adresse basse).

Le nom du fichier à programmer dans le circuit vous est ensuite demandé. Ce fichier doit être dans le même répertoire disque que celui contenant le programme Basic et doit être au format normalisé par Motorola dit format S1-S9. Tous les assembleurs pour microprocesseurs et microcontrôleurs Motorola savent produire ce format, vous n'avez donc aucune crainte à avoir. Le fichier est alors converti en binaire ce qui peut prendre de quelques secondes à quelques minutes sur les machines les plus lentes.

Un message vous signale que la conversion est en cours. Le numéro du port série utilisé vous est ensuite demandé puis ce port est ouvert en sortie.

Le fait de frapper ENTREE en réponse à la question posée fait alors envoyer le chargeur au 68HC711 qui le renvoie en écho sur l'écran à des fins de contrôle. En outre, à la fin de ce chargement, et s'il s'est bien passé, la LED rouge s'allume. Le programme vous demande alors d'appliquer la haute tension, ce que vous ferez en basculant S1, et la programmation commence dès la frappe de ENTREE ; un abandon du processus est encore possible à ce stade du programme mais, après la frappe d'ENTREE, c'est trop tard !

Pendant la programmation, les octets sont relus au fur et à mesure qu'ils sont programmés et la ligne la plus haute de l'écran affiche le numéro d'octet en cours de traitement, la valeur à programmer et la valeur effectivement relue dans le circuit (ces deux données doivent être identiques si tout se passe bien).

En fin de programmation, le logiciel vous demande de couper la haute tension, de faire un Reset puis d'éteindre le programmateur avant de retirer le circuit de son support. Si vous avez investi dans un 68HC711D3CS2 c'est à dire dans un modèle à fenêtre, vous pourrez tenter une «vraie» programmation, avec n'importe quoi ou avec un petit programme de votre choix par exemple, afin de voir si tout se passe comme prévu.

Il vous suffira ensuite d'effacer le circuit pour pouvoir l'utiliser à nouveau.

Si par contre vous n'avez sous la main que des versions OTP, il est inutile de gâcher un boîtier uniquement pour voir si «ça marche».

Pour tester tout de même votre programmateur, suivez exactement le mode d'emploi précédent mais, lorsque le programme vous demande d'appliquer la haute tension, ne touchez pas à S1 et répondez lui tout de même comme si c'était fait.

La programmation n'aura pas physiquement lieu dans le circuit mais le programme et la vérification se dérouleront cependant normalement. Compte tenu du fait qu'un 68HC711 vierge contient FF, tous les octets différents de FF que vous tenterez de programmer seront indiqués comme étant faux bien sûr mais cela vous permettra de tester de façon certaine tout votre logiciel de programmation en laissant votre 68HC711 parfaitement vierge.

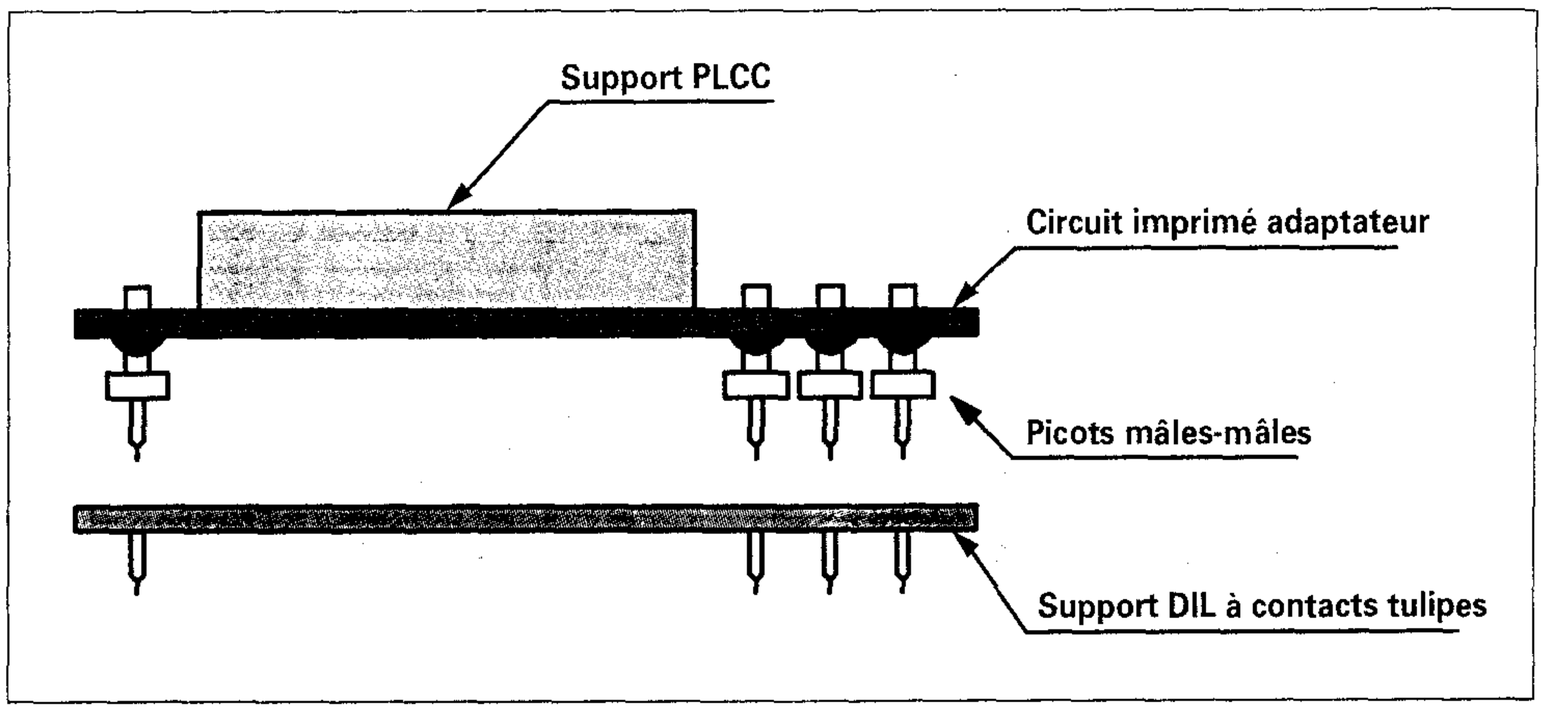
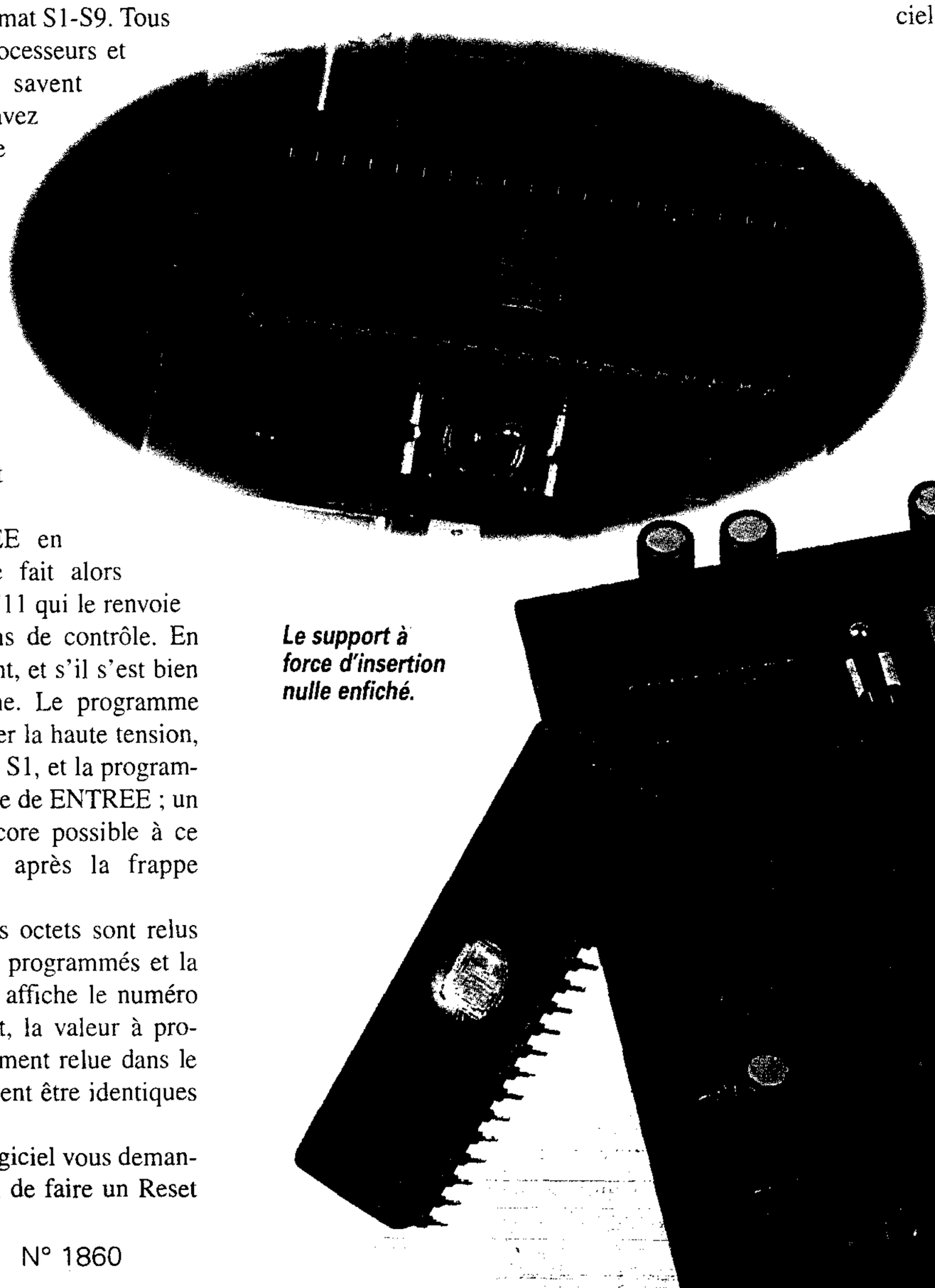


Figure 10 : Principe de montage des adaptateurs PLCC.

Les adaptateurs PLCC

Pour que notre ensemble soit complet, il ne reste plus à réaliser que les adaptateurs PLCC, ce qui ne présente aucune difficulté mais demande juste un peu de soin.



Le support à force d'insertion nulle enfiché.

Listing du programme en Quickbasic pour compatible PC ou MacIntosh.

```
1 ' ************************************
2 ' *    Programme de programmation d'un
3 ' *    68HC711 en utilisant le mode bootstrap
4 ' *    Ce programme a été testé sur un compa-
5 ' *    tible PC avec QUICKBASIC
6 ' *    Copyright C.TAVERNIER pour la ver-
7 ' *    sion française actuelle
8 ' *
10 '************************************
11 H$ = "0123456789ABCDEF"
12 DEFINT B, I
13 CLS
14 PRINT "Type de processeur 1 = 711D3, 2 =
711E9 ";
15 INPUT TP$
16 IF TP$ = "1" THEN 19
17 IF TP$ = "2" THEN 24
18 GOTO 13
19 PRINT "Programmation avec les valeurs par
défaut : taille PROM 4 K octets"
20 PRINT "                                    début
PROM en $F000"
21 CODESIZE% = 4096: ADRSTART = 61440!
22 RESTORE 9500
23 GOTO 35
24 PRINT "Programmation avec les valeurs par
défaut : taille PROM 12 K octets"
25 PRINT "                                    début
PROM en $D000"
26 CODESIZE% = 12288: ADRSTART =
53248!
27 RESTORE 9600
35 BOOTCOUNT = 25
40 DIM CODE%(CODESIZE%)
45 BOOTCODE$ = ""
50 FOR I = 1 TO BOOTCOUNT
55 READ Q$
60 A$ = MID$(Q$, 1, 1)
65 GOSUB 7000
70 TEMP = 16 * X
75 A$ = MID$(Q$, 2, 1)
80 GOSUB 7000
85 TEMP = TEMP + X
90 BOOTCODE$ = BOOTCODE$
   + CHR$(TEMP)
95 NEXT I
100 ON ERROR GOTO 0
101 LOCATE 5, 1
105 PRINT "Nom du fichier au format S1-S9 à
programmer";
107 INPUT Q$
110 IF Q$ = "" THEN PRINT CHR$(7): GOTO
101
115 FILNAM$ = Q$
116 ON ERROR GOTO 200
120 OPEN FILNAM$ FOR INPUT AS #1
130 PRINT : PRINT "Conversion de "; FIL-
NAM$; " en binaire"
140 GOTO 1000
200 PRINT CHR$(7);
210 PRINT "Nom de fichier incorrect ou fichier
introuvable";
220 FOR J = 1 TO 20
230 FOR I = 1 TO 10000
240 NEXT I
250 NEXT J
260 RESUME 100
1000 GOSUB 6000
1010 IF FLAG THEN 1250
1020 IF A$ <> "S" THEN 1000
1022 GOSUB 6000
1024 IF A$ <> "1" THEN 1000
1030 GOSUB 6000
1040 GOSUB 7000
1050 BYTECOUNT = 16 * X
1060 GOSUB 6000
1070 GOSUB 7000
1080 BYTECOUNT = BYTECOUNT + X
1090 BYTECOUNT = BYTECOUNT - 3
1100 GOSUB 6000
1102 GOSUB 7000
1104 ADDRESS = 4096 * X
1106 GOSUB 6000
1108 GOSUB 7000
1110 ADDRESS = ADDRESS + 256 * X
1112 GOSUB 6000
1114 GOSUB 7000
1116 ADDRESS = ADDRESS + 16 * X
1118 GOSUB 6000
1120 GOSUB 7000
1122 ADDRESS = ADDRESS + X
1124 ARRAYCNT = ADDRESS - ADRSTART
1130 FOR I = 1 TO BYTECOUNT
1140 GOSUB 6000
1150 GOSUB 7000
1160 Y = 16 * X
1170 GOSUB 6000
1180 GOSUB 7000
1190 Y = Y + X
1200 CODE%(ARRAYCNT) = Y
1210 ARRAYCNT = ARRAYCNT + 1
1220 NEXT I
1230 GOTO 1000
1250 CLOSE 1
1400 REM ***** Envoi le code du chargeur au
circuit *****
1410 PRINT "Quel port série utilisez-vous (1 ou
2) ";
1420 INPUT NP$
1430 IF NP$ = "1" THEN 1460
1440 IF NP$ = "2" THEN 1500
1450 LOCATE 8, 1: GOTO 1410
1460 OPEN
"COM1:1200,N,8,1,CD0,CS0,DS0,RS" FOR
RANDOM AS #2
1465 REM OPEN "R",#2,"COM1:1200,N,8,1"
pour un Macintosh
1470 GOTO 1510
1500 OPEN
"COM2:1200,N,8,1,CD0,CS0,DS0,RS" FOR
RANDOM AS #2
1505 REM OPEN "R",#2,"COM2:1200,N,8,1"
pour un Macintosh
1510 INPUT "Port COM ouvert, frappez
[ENTREE] pour continuer "; Q$
1512 WHILE LOC(2) > 0
1513 GOSUB 8020
1514 WEND
1515 PRINT : PRINT "Envoi du chargeur au cir-
cuit"
1520 A$ = CHR$(255) + BOOTCODE$
1530 GOSUB 6500
1540 PRINT
1550 FOR I = 1 TO BOOTCOUNT
1560 GOSUB 8000
1564 K = ASC(B$): GOSUB 8500
1565 PRINT "Caractère n°"; I; " reçu = "; HX$
1570 NEXT I
1590 PRINT "Appliquez la haute tension puis
frappez [ENTREE] "
1591 PRINT "Frappez n'importe quelle autre
touche pour abandonner ";
1592 INPUT Q$
1595 IF Q$ <> "" THEN 5000
1596 CLS
1597 WHILE LOC(2) > 0
1598 GOSUB 8020
1599 WEND
1600 XMT = 0: RCV = 0
1610 A$ = CHR$(CODE%(XMT))
1620 GOSUB 6500
1625 FOR I = 1 TO CODESIZE% - 1
1630 A$ = CHR$(CODE%(I))
1635 GOSUB 6500
1640 GOSUB 8000
1650 RCV = I - 1
1660 LOCATE 2, 1: PRINT "Vérification octet
n°"; I; "    "
1664 IF CHR$(CODE%(RCV)) = B$ THEN
1670
1665 K = CODE%(RCV): GOSUB 8500
1666 LOCATE 1, 1: PRINT "Octet n°"; I; "    ",
" Programmé - "; HX$;
1668 K = ASC(B$): GOSUB 8500
1669 PRINT "      Lu en retour - "; HX$;
1670 NEXT I
1680 GOSUB 8000
1690 RCV = CODESIZE% - 1
1700 LOCATE 2, 1: PRINT "Vérification octet
n°"; CODESIZE%; "    "
1710 IF CHR$(CODE%(RCV)) = B$ THEN
1720
1713 K = CODE%(RCV): GOSUB 8500
1714 LOCATE 1, 1: PRINT "Octet n°"; CODE-
SIZE%; "    ", " Programmé - "; HX$;
1715 K = ASC(B$): GOSUB 8500
1716 PRINT "      Lu en retour - "; HX$;
```

Listing du programme en Quickbasic pour compatible PC ou MacIntosh. (suite)

```
1720 LOCATE 8, 1: PRINT : PRINT "***
Programmation terminée  ***"
1730 PRINT "*** Coupez en premier la haute
tension puis faites un RESET ***";
1740 PRINT "*** Coupez ensuite l'alimentation
avant d'enlever le circuit du support ***"
1750 PRINT CHR$(7)
4900 CLOSE
4910 INPUT "Frappez [ENTREE] pour quitter ",
Q$
5000 END
5900 '*********************************
5910 '*  LECTURE D'UN OCTET DEPUIS UN
5930 '*  FICHIER DISQUE L'OCTET LU EST
         DANS A$
5940 '*********************************
6000 FLAG = 0
6010 IF EOF(1) THEN FLAG = 1: RETURN
6020 A$ = INPUT$(1, #1)
6030 RETURN
6490 '*********************************
6492 '*  ENVOI DU CONTENU DE A$ DANS
6494 '*  LE PERIPHERIQUE OUVERT EN #2
6496 '*********************************
6500 PRINT #2, A$;
6510 RETURN
6590 '*********************************
6594 '* CONVERSION DE L'HEXA
         CONTENU DANS A$ EN ENTIER
6596 '*********************************
7000 X = INSTR(H$, A$)
7010 IF X = 0 THEN FLAG = 1
7020 X = X - 1
7030 RETURN
7990 '*********************************
7992 '* LECTURE D'UN OCTET VIA
         LE PORT COM OUVERT EN #2
7994 '* ATTEND INDEFINIMENT
         LA RECEPTION DE L'OCTET
7996 '* SOUS PROGRAMME ABANDONNE
         PAR FRAPPE AU CLAVIER
7998 '* OCTET RECU DANS B$.
         UTILISE Q$.
7999 '*********************************
8000 WHILE LOC(2) = 0          'Attente d'un
entrée
8005 Q$ = INKEY$: IF Q$ <> "" THEN 4900
'Clavier actionné -> abandon
8010 WEND
8020 B$ = INPUT$(1, #2)
8030 RETURN
8490 '*********************************
8491 '*  CONVERSION DECIMAL HEXA
8492 '*  ENTREE:  K - ENTIER A CONVER-
8493 '*  TIR SORTIE: HX$ - DEUX
         CARACTERES EN HEXA
8494 '*********************************
8500 IF K > 255 THEN HX$ = "Trop grand":
GOTO 8530
8510 HX$ = MID$(H$, K \ 16 + 1, 1)
8520 HX$ = HX$ + MID$(H$, (K MOD 16) +
1, 1)
8530 RETURN
9499 '**** CHARGEUR POUR 711D3 ******
9500 DATA 86, 03          'LDAA  #$03
9510 DATA B7, 00, 02       'STAA  OPT2
Port C en Push Pull
9520 DATA 86, FF          'LDAA  #$FF
9530 DATA B7, 00, 03       'STAA  PORTC
Allume 1 LED sur PC0
9540 DATA C6, FF          'LDAB  #$FF
9550 DATA F7, 00, 07       'STAB  DDRC
Port C en sortie
9560 DATA CE, 0F, A0       'LDX   #4000
2msec à 2MHz
9570 DATA 18, CE, F0, 00    'LDY   #$F000
Début de la ROM
9580 DATA 7E, BF, 00       'JMP   $BF00
9590 '*********************************
9595 '*****CHARGEUR POUR 711E9 ******
9600 DATA 86, 03          'LDAA  #$03
9610 DATA B7, 10, 02       'STAA  OPT2
Port C en Push Pull
9620 DATA 86, FF          'LDAA  #$FF
9630 DATA B7, 10, 03       'STAA  PORTC
Allume 1 LED sur PC0
9640 DATA C6, FF          'LDAB  #$FF
9650 DATA F7, 10, 07       'STAB  DDRC
Port C en sortie
9660 DATA CE, 0F, A0       'LDX   #4000
2msec à 2MHz
9670 DATA 18, CE, D0, 00    'LDY
#$D000  Début de la ROM
9680 DATA 7E, BF, 00       'JMP   $BF00
9690 '*********************************
```

Leurs circuits imprimés vous sont présentés figures 6 et 7 tandis que les plans d'implantation correspondants sont visibles figures 8 et 9.
Veillez à orienter correctement les supports PLCC ; ils peuvent en effet entrer dans tous les sens sur les circuits imprimés mais le HC711 n'apprécierait pas vraiment une rotation de ses pattes !
Le pan coupé, visible dans un angle des supports PLCC, doit donc impérativement être placé comme indiqué sur le plan d'implantation.
Pour que ces adaptateurs puissent être mis en place facilement sur le support DIL à 40 pattes, qu'il soit «normal» ou à force d'insertion nulle, nous avons procédé de la façon suivante, schématisée figure 10.
Côté cuivre du circuit des adaptateurs nous avons soudé des picots à souder mâles - mâles ; le côté de plus gros diamètre de ces picots étant soudé sur le circuit imprimé pour des raisons de rigidité mécanique.
Les adaptateurs pourraient alors être enfichés comme cela dans le support DIL du programmateur mais, vu le faible nombre de picots et leur relative fragilité, nous avons préféré enficher une fois pour toute chaque adaptateur dans un support DIL à 40 pattes à contacts tulipe.
C'est ensuite celui-ci qui vient s'enficher dans le support DIL du programmateur.
On ne court ainsi aucun risque de tordre ou de casser les picots mâles - mâles des adaptateurs.
Si vous le désirez, vous pouvez même coller à la colle époxy, les picots des adaptateurs sur les supports DIL intermédiaires mais, même après une utilisation intensive de l'ensemble, nous ne l'avons pas jugé nécessaire.
Ces adaptateurs doivent évidemment être enfichés dans le bon sens sur le support DIL du programmateur mais il est quasiment impossible de se tromper ; en effet, la dissymétrie volontaire du circuit imprimé de ces adaptateurs fait qu'ils butent dans le levier de commande de S1 si on tente de les placer à l'envers !

L'assembleur gratuit

Nous vous l'avons annoncé dans notre article théorique et nous tenons parole. Vous pouvez en effet vous procurer un assembleur pour 68HC11 tout à fait honorable constituant ainsi avec notre programmateur un outil de développement complet. Rendons à César ce qui lui appartient ; cette gratuité n'est pas notre fait mais celui de Motorola qui propose sur son serveur Internet (adresse HTTP://WWW.MOT.COM) un produit appelé AS11.EXE qui n'est autre qu'un assembleur pour 68HC11 tournant sur compatible PC. Ce produit est fourni avec sa notice complète et il ne s'agit pas d'une quelconque version de démo mais bien d'un programme complet.
Et si vous n'êtes pas un Internaute convaincu, ou tout simplement si vous ne disposez pas d'un abonnement vous permettant d'accéder à Internet, vous pouvez aussi télécharger cet assembleur sur le serveur de la revue sur lequel nous avons fait le nécessaire pour qu'il soit présent.

Conclusion

Muni de ces deux outils les portes du développement d'applications à base de 68HC(7)11 vous sont ouvertes mais, si la programmation n'est pas votre tasse de thé, sachez déjà que dans les mois qui viennent nous vous proposerons quelques idées d'utilisations honnêtes de ces petites merveilles que sont les 68HC11.

C. Tavernier

BIBLIOGRAPHIE

- **Note d'application AN 1060 de Motorola** "MC68HC11 Bootstrap Mode".
- **Microcontrôleur 68HC11** - Description, par C.Tavernier aux Editions Dunod.
- **Microcontrôleur 68HC11** - Applications, par C.Tavernier aux Editions Dunod (sortie en juin 1997).