10/5

Mémoires d'ordinateur

Qu'elle soit morte ou vive, la mémoire d'ordinateur sera choyée par les montages de ce chapitre.

10/5.1

Un programmateur de mémoires EPROM

Comme leur nom l'indique, les mémoires EPROM (Erasable Programmable Read Only Memories) sont des *mémoires mortes* (ROM) que l'on peut programmer plusieurs fois grâce à une possibilité d'*effacement*.

Ce type de composant permet à l'amateur, à l'instar des fabricants professionnels, de « figer » des programmes ou des données dans des mémoires qui conserveront leur contenu pendant des années, même sans alimentation.

Contrairement à la ROM de l'AMSTRAD qui ne peut être programmée qu'en usine et par grandes quantités (mais en aucun cas effacée), les EPROM peuvent être programmées et effacées à l'aide d'un matériel relativement simple.

La programmation d'EPROM est une étape indispensable de la réalisation de *montages à microprocesseurs :* il peut s'agir de montages décrits dans des publications ou de créations personnelles. Il est par exemple possible de mettre au point toutes sortes d'applications sur un AMSTRAD équipé d'un bon assembleur-debugger, puis de transférer le logiciel définitif dans une EPROM que l'on installera sur une *carte microprocesseur* à Z 80.

L'AMSTRAD sera ainsi libéré pour d'autres usages, tandis que la carte autonome pourra « tourner » 24 heures sur 24 si nécessaire (centrale d'alarme, régulateur de chauffage, etc.).

On peut aussi songer à connecter au bus de l'AMSTRAD des EPROM contenant des routines d'usage fréquent, ou même des programmes entiers, qu'il n'y aura jamais plus besoin de charger à partir d'une cassette ou d'un disque, bref à fabriquer soi-même des *cartouches logicielles*.

A la limite, on pourrait même remplacer la ROM d'origine de l'AMSTRAD par une EPROM « maison » : doté d'un tout nouveau système d'exploitation, l'ordinateur pourrait être transformé en une toute autre machine (mais il s'agit là d'un travail de programmeur plus qu'averti !).

En tout cas, une foule de possibilités passionnantes attendent l'heureux possesseur d'un programmateur d'EPROM et si possible d'un effaceur (tout le monde peut se tromper ou changer d'avis !).

Les programmateurs du commerce coûtent en général largement plus cher qu'un système AMSTRAD complet, ou alors il ne s'agit que de « gadgets » aux possibilités extrêmement limitées.

Heureusement, même le plus simple des AMSTRAD peut se transformer en un très honnête programmateur, moyennant l'adjonction d'un peu de matériel et de logiciel.

Familiarisation avec les EPROM

Il existe sur le marché de nombreux types de mémoires EPROM, dont les brochages et les caractéristiques obéissent à une certaine standardisation (norme « BYTEWIDE »), et qui diffèrent essentiellement par leur capacité et leur prix.

Nous estimons que la meilleure référence à conseiller à l'amateur désireux de découvrir les EPROM est la « 2716 » : ses 2048 octets de capacité suffisent pour de nombreuses applications, tandis que son prix et sa disponibilité sont satisfaisants.

De toute façon, les méthodes et matériels décrits pourront facilement être adaptés aux versions de capacité supérieure.

La figure 1 reproduit le brochage de la 2716, qui est présentée dans un boîtier DIL céramique à 24 broches. Une fenêtre transparente (en quartz) sur le dessus permet l'exposition à la lumière ultraviolette pour l'effacement. La lumière ambiante pouvant causer à la longue une altération des

données programmées, il est indispensable d'obturer cette fenêtre avec une étiquette opaque sitôt la programmation réalisée. Il est d'usage d'y inscrire un repère identifiant le contenu.

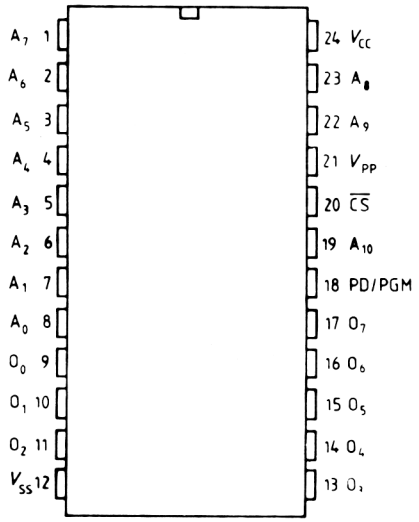

Fig. 1

On retrouve dans ce brochage les huit lignes du bus de données (utilisé d'abord en programmation puis par la suite exclusivement pour la lecture), onze lignes d'adresse (2^{11} = 2048 soit 2k-octets adressables), une broche de sélection $\overline{CS}$ comme en possède toute mémoire, une broche de masse Vss, une broche d'alimentation + 5 V Vcc, et deux broches spécifiques aux EPROM : Vpp est destinée à recevoir une tension d'alimentation spéciale pour la programmation (+25 V), alors que PD/PGM permet précisément de mettre le circuit en mode « programmation ».

Le tableau 1 résume les différentes possibilités offertes pour l'utilisation pratique de la 2716, selon les tensions appliquées à ses différentes broches.

On constate que la phase de programmation terminée, la 2716 se comporte exactement comme n'importe quelle ROM, ses sorties de données n'étant actives que lorsque l'entrée de sélection est mise à la masse. Le reste du temps, l'état « haute impédance » des lignes de données permet la libre utilisation du bus par d'autres composants.

La programmation de la mémoire ressemble un peu à l'écriture dans une mémoire vive « RAM », mais en *beaucoup* plus lent : il faut appliquer une impulsion positive de 50 ms exactement sur PD/PGM ou $\overline{CS}$ (selon le branchement choisi), en présence de +25 V sur Vpp. Pendant ce temps, adresse et données doivent être maintenus sur le bus.

A titre de comparaison, quelques centaines de nanosecondes suffisent pour « écrire » dans une mémoire RAM (1 ms = 10^6 ns) ! En général, les bus d'un système à microprocesseur ordinaire sont incapables de

Tableau 1

Mode de fonctionnement 2716 / Broche		PD/PGM (18)	$\overline{CS}$ (20)	Vpp (21)	Bus de données (9-11 et 13-17)
lecture	sélectionnée	0	0	+5 V	sortie données
	non sélectionnée	sans importance	1	+5 V	haute impédance
attente		1	sans importance	+5 V	haute impédance
programmation	sélectionnée	⎍⎍ 50 ms	1	+25 V	entrée de données
		1	⎍⎍ 50 ms		
	bloquée	0	1	+25 V	haute impédance
	vérification (lecture)	0	0	+25 V	sortie de données

maintenir si longtemps adresses et données parfaitement stables, et il faut intercaler des bascules spéciales appelées « latches » pour les mémoriser le temps voulu. Avant de décrire une solution pratique, ajoutons que la tension de +25 V ne doit jamais être présente en l'absence du +5 V : il y va de la vie même de l'EPROM !

Un programmateur adaptable à l'AMSTRAD

La figure 2 expose le principe que nous avons défini pour permettre la programmation de 2716 au meilleur coût, en profitant de la puissance de traitement d'un AMSTRAD.

Les lignes de données et d'adresse de l'EPROM sont reliées à leurs homologues du bus de l'AMSTRAD à travers une batterie de bascules « latch » de type 7475. Ces bascules ont la propriété de recopier sur leur sortie l'état appliqué à leur entrée tant que leur entrée de commande « T » est à 1, et de figer leur sortie à leur état du moment dès que T passe à 0, état qui persistera tant que T ne reviendra pas à 1.

En fait, on peut laisser T à 0 presque en tout temps, et n'y appliquer que de brèves impulsions positives lorsque l'on souhaite saisir « au vol » les états très fugaces des bus de l'AMSTRAD.

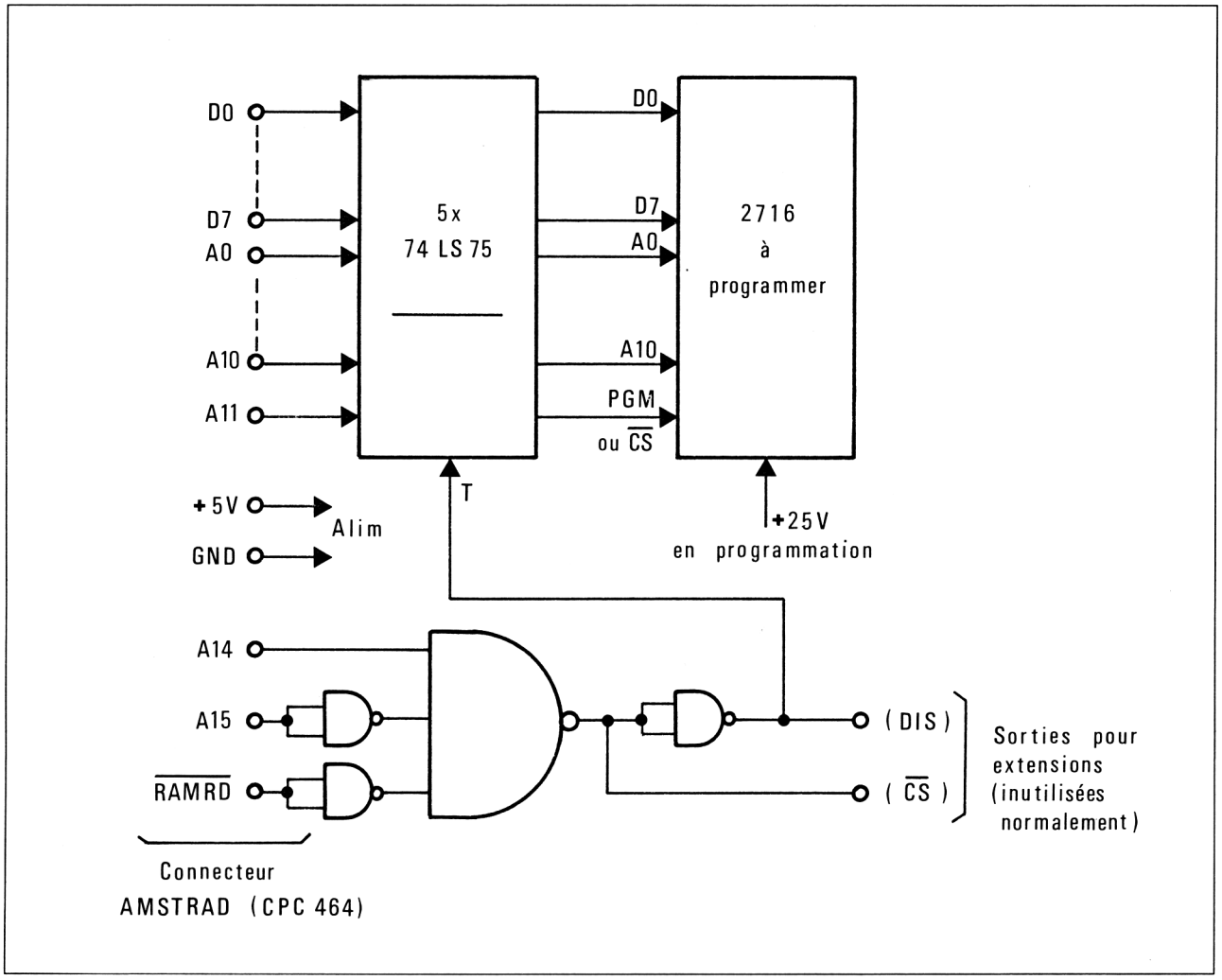

Fig. 2

Tout le secret du fonctionnement du programmateur réside dans la génération de ces impulsions exactement aux bons moments, et en synchronisme avec le déroulement du logiciel de gestion de l'appareil.

Cette technique est suffisamment souple pour que même l'impulsion de 50 ms puisse être générée par le programme, et amenée à l'EPROM (broche PD/PGM ou $\overline{CS}$) par un latch relié à la ligne A11 du bus d'adresse : grosse simplification du matériel !

Ce schéma très simple est complété par une petite alimentation + 25 V décrite à la figure 3, et qui peut être exécutée en version piles ou secteur selon l'usage plus ou moins intensif que l'on compte faire du programmateur.

Dans les deux cas, l'interrupteur « programmation » devra être manœuvré à bon escient pour éviter d'appliquer du + 25 V en l'absence de + 5 V : le logiciel y veillera...

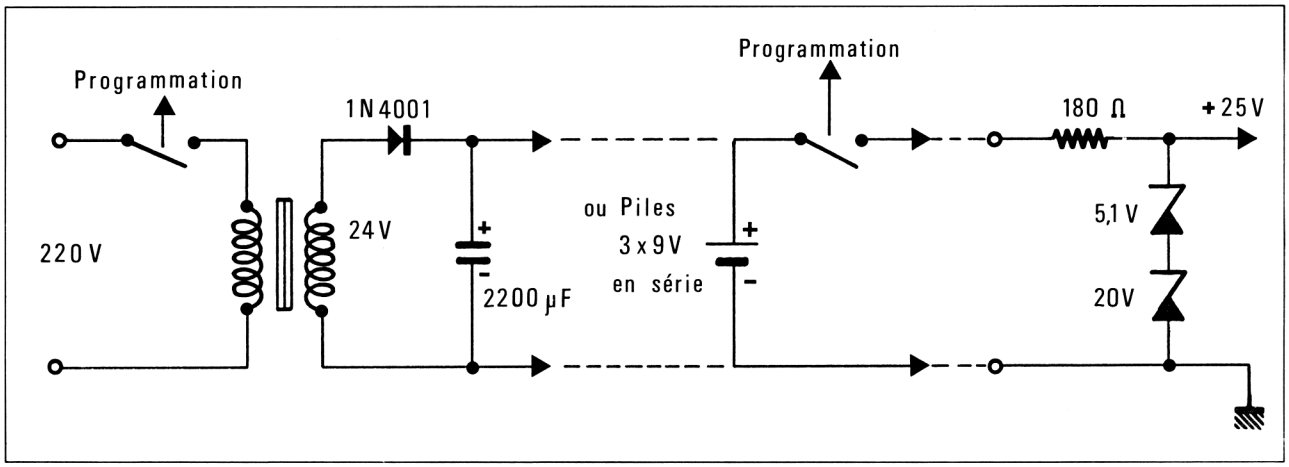

Fig. 3

La conception du « circuit de sélection » attaquant l'entrée « T » de tous les latches appelle quelques commentaires : c'est elle qui fixe, en effet, les instructions qu'il faudra faire exécuter à l'AMSTRAD pour qu'il programme l'EPROM.

Les portes NAND utilisées sont interconnectées de façon à ce que les entrées « T » ne passent à 1 que lorsque les états suivants seront obtenus *simultanément* sur le bus de l'AMSTRAD :

— A14 à 1

— A15 à 0

— $\overline{\text{RAMRD}}$ à 0

Cela signifie que les états des lignes de données D0 à D7 et des lignes d'adresse A0 à A11 ne seront mémorisés par les latches que lorsque sera effectuée une *lecture* (PEEK) d'une adresse mémoire de l'AMSTRAD supérieure à 16383, mais inférieure à 32768 (en décimal), ce qui tombe juste dans la zone « utilisateur » non perturbée par ailleurs.

Le tableau 2 montre comment sont construites les adresses « stratégigues » que devra utiliser le logiciel d'exploitation.

En résumé, il devra effectuer une série de PEEK sur les adresses 16384 à 18431 dans lesquelles on aura au préalable « rangé » tout ce qu'il faut « recopier » dans les 2k-octets de la 2716.

Quelle que soit la destination du résultat de PEEK, l'octet ainsi prélevé en RAM transitera forcément sur le bus de données et sera donc « intercepté » par les latches du programmateur.

Cette « mémorisation » effectuée, il faut encore générer l'impulsion de programmation de 50 ms exactement, *sans modifier en rien* l'état des lignes d'adresse et de données de l'EPROM.

Cela peut être fait en lançant un second PEEK sur une adresse supérieure de 2048 à la précédente, puis après 50 ms d'attente, réitérer le premier PEEK.

Tableau 2

	2048 adresses de la 2716 à programmer											**Impulsion 50 ms**	**Base adresse du programmateur**			
	1 A_0	2 A_1	4 A_2	8 A_3	16 A_4	32 A_5	64 A_6	128 A_7	256 A_8	512 A_9	1024 A_{10}	2048 A_{11}	4096 A_{12}	8192 A_{13}	16384 A_{14}	32768 A_{15}
16384	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
18432	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
18431	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	0
20479	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0
+ 8192	X	X	X	X	X	X	X	X	X	X	X	X	0	1	X	X
+ 4096	X	X	X	X	X	X	X	X	X	X	X	X	1	0	X	X
+ 12288	X	X	X	X	X	X	X	X	X	X	X	X	1	1	X	X
32767	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Cela suppose évidemment que le « modèle » de l'EPROM soit présent deux fois à la suite dans la RAM de l'AMSTRAD.

En réalité, cette méthode suppose un énorme gaspillage de RAM : 16 K d'espace mémoire sont monopolisés pour 2 K à programmer !

En fait, compte tenu de la simplicité du logiciel d'exploitation du programmateur, cette place mémoire serait de toute façon restée inutilisée : mieux vaut en profiter pour simplifier les circuits du programmateur et pour réserver des possibilités d'extension à des EPROM de capacité supérieure (il faudrait mettre à contribution A12 et A13, inutilisées pour l'instant).

Reste à expliquer l'utilité des points « DIS » et « $\overline{CS}$ » appelés « sorties pour extensions » sur la figure 2.

Une fonction courante des programmateurs du commerce consiste à *dupliquer* des EPROM existantes, et ces deux connexions permettent au nôtre d'en faire autant très simplement.

Si nous relions DIS du programmateur à RAMDIS de l'AMSTRAD, la RAM de l'ordinateur ne sera pas sélectionnée pour les adresses utilisées par le programmateur : on peut donc relier aux bus de l'AMSTRAD une mémoire externe (notamment une EPROM), et la sélectionner par la sortie $\overline{CS}$ du programmateur. C'est son contenu et non plus celui de la RAM qui sera « brûlé » dans l'EPROM vierge !

La figure 4 donne le détail de cette adaptation, qui peut tout aussi bien servir à équiper l'AMSTRAD de « cartouches » EPROM enfichables sur son connecteur arrière : des routines et/ou des données (bref du « firmware » accessibles dès la mise sous tension de la machine comme si elles se trouvaient en RAM !

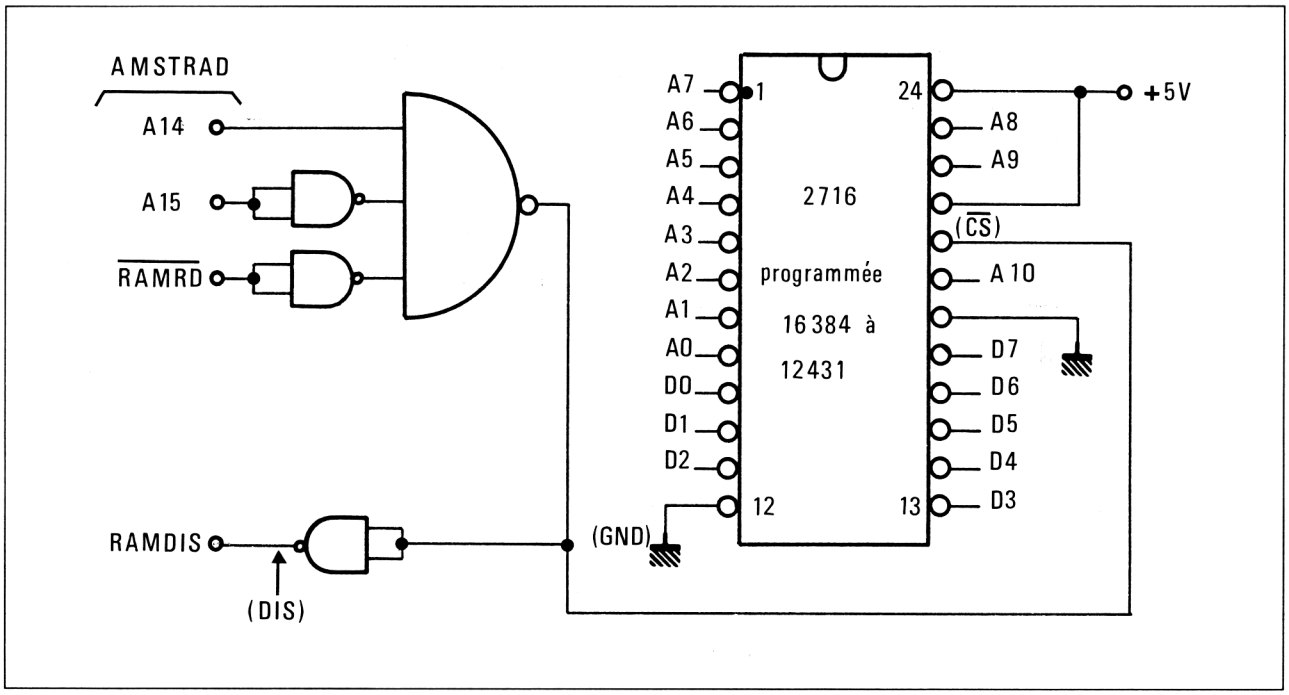

Fig. 4

Réalisation pratique

A part l'alimentation + 25 V qui, en version « secteur », est décrite séparément à la figure 5, tout le montage tient sur un unique circuit imprimé dessiné à la figure 7.

Le câblage d'après la figure 8 consiste à monter les sept circuits intégrés, le support de 2716 (si possible à force d'insertion nulle ou tout au moins d'excellente qualité), cinq straps en fil rigide, et une longueur de câble plat à 25 conducteurs (en version de base) rejoignant les broches de mêmes noms d'un connecteur compatible avec la prise d'extension à 50 points de l'AMSTRAD (voir 8/1 page 3).

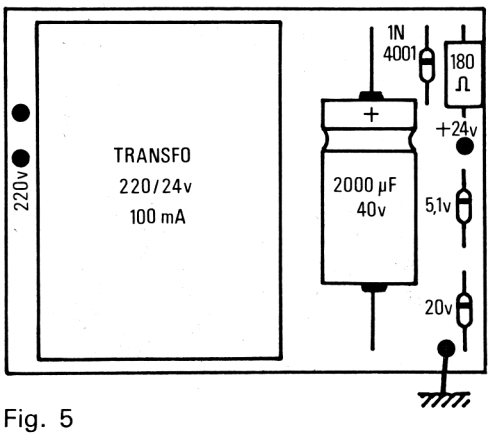

Fig. 5

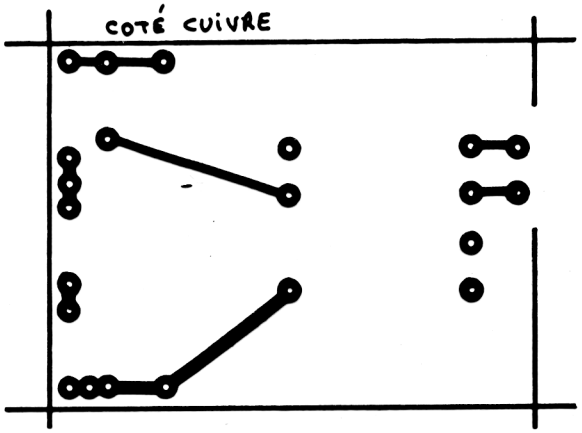

Fig. 6

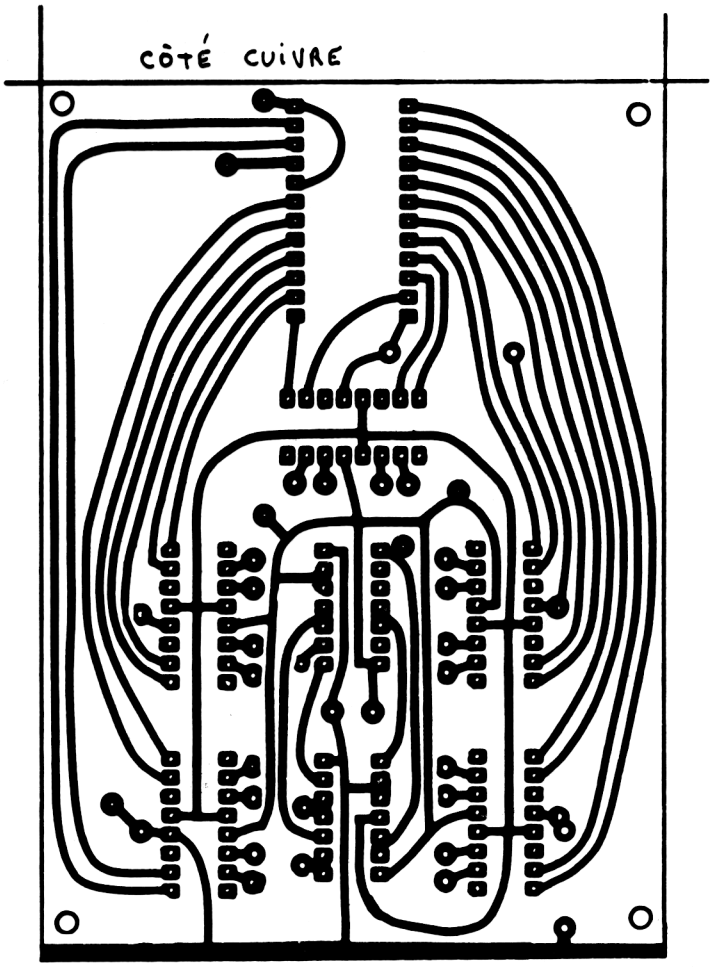

Fig. 7

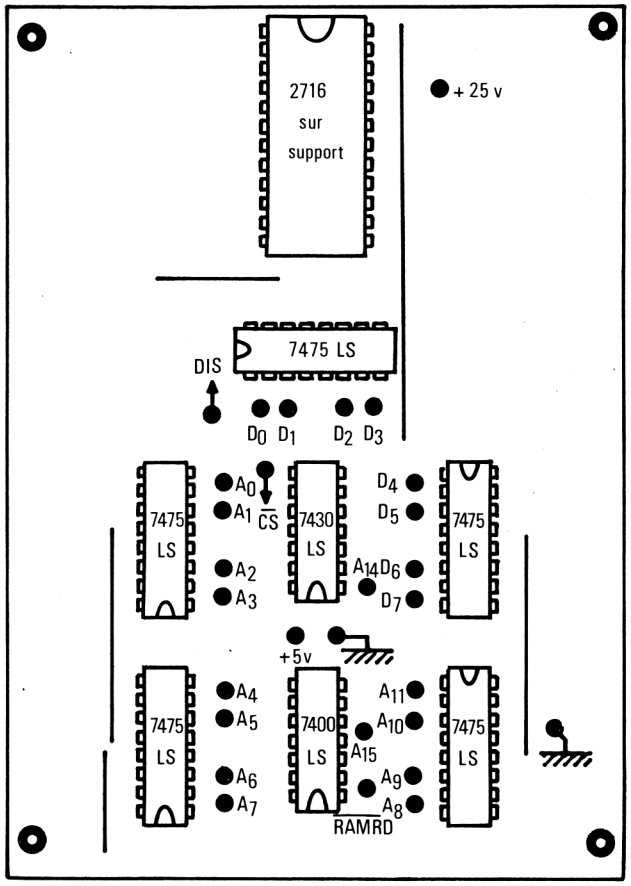

Fig. 8

On fera aussi court que possible, une longueur de plus de 30 cm risquant de créer des problèmes.

Le programmateur peut se loger dans un boîtier plastique 115 PP de marque MMP, à condition de prévoir une découpe en regard du support d'EPROM. Bien entendu, tout autre coffret fera aussi bien l'affaire, par exemple un pupitre TEKO ou RETEX, qui pourra abriter en même temps le support pour une EPROM « modèle » en vue de travaux de duplication.

On ne recommandera jamais assez le plus grand soin au cours de la construction, ainsi que de sérieux contrôles : ce montage est en prise directe avec les circuits internes de l'AMSTRAD, ce qui rendrait toute erreur ou fausse manœuvre lourde de conséquences. Prenez vos précautions, car notre responsabilité ne saurait être engagée.

Le logiciel d'exploitation

Nous n'avons pu réduire la partie matérielle à aussi peu de chose qu'en reportant sur le logiciel la majorité des tâches délicates. Les possibilités du programmateur seront donc celles de son programme d'exploitation.

Bien que court, le logiciel ci-dessous suffit pour « brûler » dans une 2716 les 2048 octets supposés présents en RAM à partir de l'adresse 16384. Il suffit de se conformer strictement aux consignes apparaissant sur l'écran.

```
1 FOR a=16384 TO 18431
10 POKE a+2048,PEEK(a)
20 NEXT a
30 a=PEEK(16384):CLS
40 PRINT"Alimenter le Programmateur"
50 PRINT"    Puis Presser ENTER"
60 INPUT a$
70 FOR f=16384 TO 18431
80 a=PEEK(f)
85 a=PEEK(f+2048)
90 FOR t=1 TO 50:NEXT t
100 a=PEEK(f):PRINT f-16384,a
110 NEXT f
120 PRINT:PRINT
130 PRINT "Arreter le Programmateur"
140 REM (c) 1987 Patrick GUEULLE
```

Programme réalisé par Patrick Gueulle

Tout au long de la programmation, les données programmées défilent sur l'écran avec leur adresse réelle dans l'EPROM (de 0 à 2047) : on sait donc à tout moment où on en est, ce qui permet d'interrompre l'opération par anticipation si on ne recopie que quelques dizaines ou centaines d'octets.

Sachez qu'il est toujours possible de programmer de nouvelles données par-dessus des bits à « 1 » (octets valant 255 en décimal soit FF en hexa), mais qu'un « 0 » ne peut être transformé en « 1 » que par effacement complet de l'EPROM et reprogrammation.

Veillez bien à couper le + 25 V avant d'arrêter l'AMSTRAD ou son moniteur ! Il peut arriver que l'ordinateur se « plante » lors de la mise sous tension du programmateur : cela est dû à un fort parasite provoqué par l'interrupteur secteur du programmateur. Utilisez un modèle de qualité, ajoutez un condensateur d'antiparasitage entre ses bornes (0,22 μF 400 V) ou... travaillez sur piles !

Le programme de la page suivante est destiné à nos lecteurs ne disposant pas d'autre moyen de préparer en RAM les données à recopier en EPROM (moniteur, assembleur, debugger, etc.) : il offre toutes les commodités souhaitables pour écrire, lire et modifier des octets en tout point de la zone de RAM utilisée.

Pour « sortir » de la procédure d'acquisition, il faut presser ENTER sans donner de valeur : dans le mode « contrôle et modification » ainsi appelé, la touche ENTER permet d'avancer d'une adresse à la fois, tandis que les touches de flèche haute et basse permettent de reculer et d'avancer avec répétition automatique, ce qui est bien plus rapide.

La touche DEL permet le retour à la procédure d'acquisition en restant sur la même adresse, ce qui permet toute correction. L'exécution s'arrête au bout des 2048 octets, ou à tout moment sur un BREAK. On peut alors charger le programme de recopie en EPROM sans pour autant effacer le modèle qui vient d'être saisi.

```
5 CLS
10 CLS:l=16384
20 PRINT HEX$(l),
30 INPUT n$
40 IF n$="" THEN 110
50 n=VAL(n$)
60 POKE l,n
70 l=l+1
80 IF l>18431 THEN 100
90 GOTO 20
100 l=16384
110 PRINT l;" ";HEX$(l);"    ";
120 n=PEEK(l)
130 PRINT HEX$(n);" ";n
140 k$=INKEY$
150 IF k$="" THEN 140
155 k=ASC(k$)
160 IF k=241 OR k=13 THEN 200
170 IF k=127 THEN 20
180 IF k=240 THEN 250
190 GOTO 140
200 l=l+1
210 IF l>18431 THEN STOP
220 GOTO 110
250 l=l-1
260 IF l<16384 THEN STOP
270 GOTO 110
300 REM (c)1987 Patrick GUEULLE
```

Programme réalisé par Patrick Gueulle

Contrôle et effacement

Le contrôle d'une mémoire EPROM que l'on vient de programmer est facile : il suffit de la comparer adresse par adresse avec son modèle, en la lisant grâce au montage de la figure 4.

On peut aussi, pour gagner du temps, se contenter de définir un octet de « checksum » obtenu par addition sans retenue de tous les octets de la mémoire (c'est facile et très rapide en assembleur). Si le checksum de la copie diffère de celui du modèle, alors il y a erreur. Seuls de très rares cas peuvent conduire à des checksums identiques en présence d'erreurs.

L'effacement d'une EPROM nécessite par contre un matériel particulier. Les effaceurs du commerce coûtent cher, les « kits » un peu moins, mais rien ne vaut le « système-D » !

Un effaceur est une simple boîte munie d'ouvertures d'aération ne laissant pas trop passer la lumière, dans laquelle les EPROM à effacer sont piquées dans de la mousse de plastique conductrice (noire) en face d'une ampoule émettant des ultraviolets. La fenêtre de l'EPROM doit être bien en face de la lampe, à une distance de 3 à 5 cm. Pour éviter un échauffement excessif, l'EPROM doit être plutôt en-dessous de l'ampoule qu'au-dessus (la chaleur monte...).

La durée d'effacement varie grossièrement de 10 mn à 1/2 heure selon la puissance de la lampe et la distance. Il est prudent de majorer de 10 à 20 % la durée minimum déterminée par essais, ou de pratiquer systématiquement une lecture de contrôle après effacement (tous les bits doivent être à 1, c'est-à-dire tous les octets à 255 ou FF). Tout excès d'exposition risque d'abréger la vie de l'EPROM.

Le rayonnement ultraviolet le plus efficace pour l'effacement est celui produit par les tubes à vapeur de mercure (longueur d'onde de 2537 Angströms) à une dose de 15 W.s/cm².

C'est précisément ce que produisent les tubes « germicides » qui peuvent être obtenus sur commande chez les bons électriciens. Le TUV 6W PHILIPS est le plus pratique, puisqu'il fonctionne directement en 220 V sans autre accessoire qu'une douille E27. Pour effacer plusieurs EPROM à la fois, on préférera le TUV 15 (46 cm de long) qui s'utilise exactement comme un tube fluorescent à ballast de 15 watts.

Attention, ce rayonnement UV est nocif pour les yeux et la peau : concevez votre boîtage avec soin, et prévoyez une minuterie ou au moins un interrupteur pour la mise hors tension !

Nomenclature

Résistances			
R1	180Ω	R	R
Condensateurs			
C1	40V 2200µF	C	C
Circuits intégrés			
CI1	74LS30	CI7 7475LS	CI
CI2	74LS00	CI8 2716 à programmer	CI
CI3	7475LS	CI	CI
CI4	7475LS	CI	CI
CI5	7475LS	CI	CI
CI6	7475LS	CI	CI
Autres Semiconducteurs			**Divers**
1N 4001 zener 20 V zener 5.1 V			Transfo 220/24 V ou 3 piles 9 V inter unipolaire Support pour 2716 Circuits imprimés Coffret genre 115 PP Câble plat 25 conducteurs connecteur pour AMSTRAD (50 contacts 2.54 mm DF) Cordon secteur