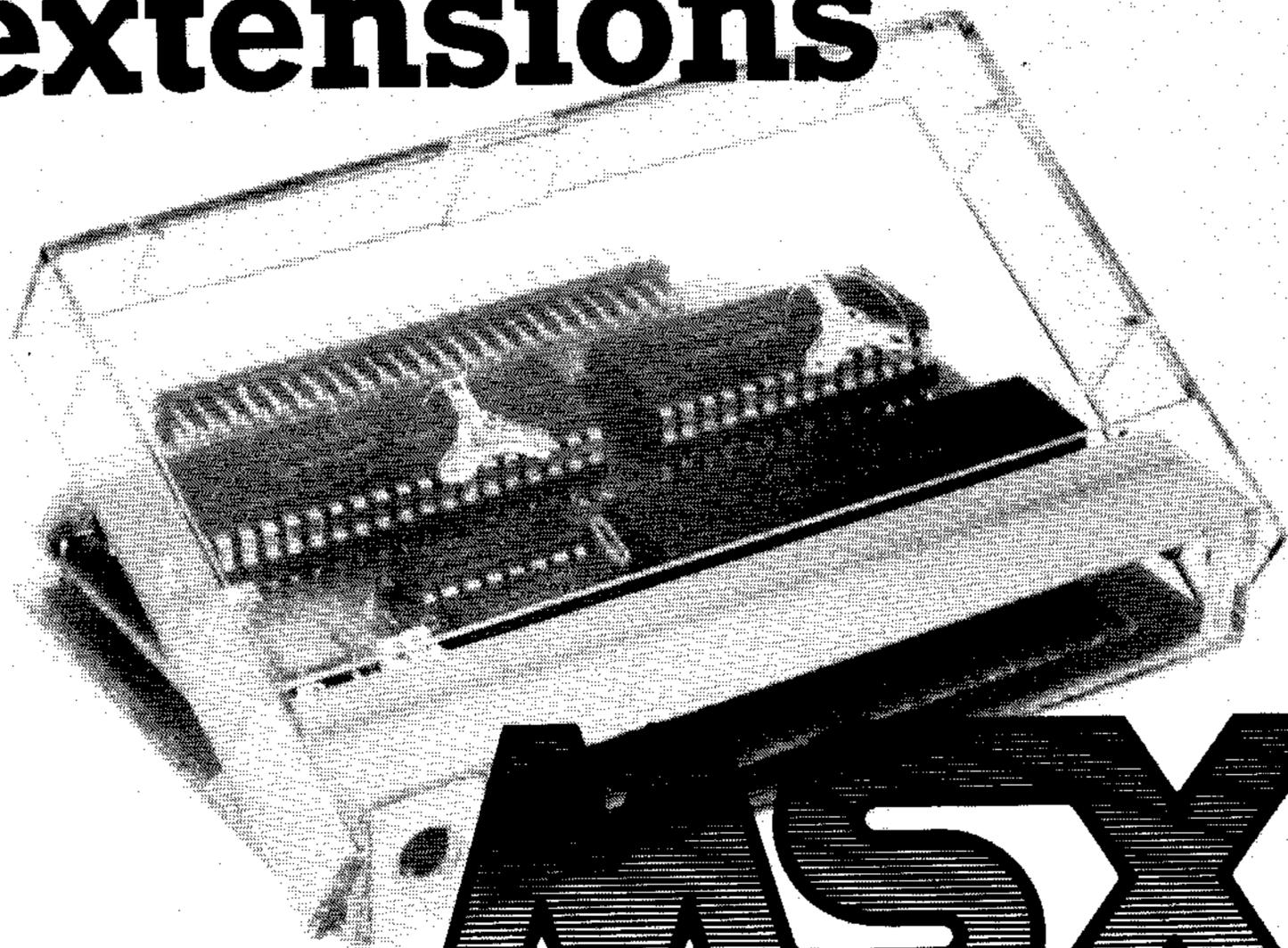


cartouche timer + interface d'E/S,
le chaînon manquant entre Microscope et MSX

extensions



MSX

(4)

Le quatrième article consacré à la réalisation d'extensions de faible complexité pour ordinateur MSX décrit un module universel de la taille d'une cartouche, assurant tout à la fois une fonction de temporisateur (timer) et celle de contrôleur d'Entrées/Sorties. Sa première application est celle de maillon indispensable reliant le Microscope (décrit en septembre et octobre derniers) à un ordinateur MSX.

Le montage que nous allons décrire dans cet article offre la possibilité aux possesseurs d'ordinateurs MSX de le doter d'une interface d'extension comportant:

- 32 (4 fois 8) lignes d'E/S;
- 4 temporisateurs programmables; et se caractérisant par:
- un décodage d'adresses définissable par l'utilisateur;
- une configuration d'interruption en noria ou en chaînage (daisy chained).

Toutes ces fonctions ont été implantées sur un circuit imprimé aux dimensions réduites au point de pouvoir l'implanter dans le boîtier plastique d'une cassette audio standard.

Bien que la fonction première de ce

montage soit de constituer une interface entre un ordinateur MSX et **Microscope**, la cartouche timer + interface d'E/S est en mesure d'assurer bien d'autres tâches. Prenez par exemple, le domaine de la robotique: la commande de moteurs pas à pas nécessite une interface les reliant à l'ordinateur (voir à ce sujet l'article "commande universelle de moteur pas à pas" ailleurs dans ce même numéro). Cette cartouche nous servira en outre d'interface entre un ordinateur MSX et le programmeur d'EPROM dont la description dans ces colonnes ne saurait tarder. Cet article-ci sera principalement consacré à la manière d'utiliser la cartouche timer + interface d'E/S 32 bits avec **Microscope**.

Les trois premiers articles consacrés à la série "extensions MSX" ont été publiés en février et mars derniers. Si vous disposez d'un ordinateur MSX, et que vous envisagiez de le doter d'une extension personnelle, nous ne saurions trop vous en recommander la (re)lecture.

Le synoptique

La **figure 1** montre les différents sous-ensembles fonctionnels présents sur la cartouche timer. Son décodeur d'adresses définit le domaine d'adresses d'E/S par lequel le processeur Z80 peut adresser la carte. Il n'est peut-être pas inutile de rappeler que les ordinateurs MSX utilisent une cartographie d'E/S comportant 255 (2 puissance 8 - 1)

ports de sortie; ce processeur ne connaît pas de domaine d'adresses spécifique réservé en RAM pour le transfert de données d'E/S et de mots de commande/d'état d'E/S.

Une fois que le processeur a choisi la cartouche par l'intermédiaire de l'instruction d'E/S convenable, le décodeur d'adresse de l'extension est validé de manière à sélectionner l'un des deux blocs d'E/S ou l'ensemble timer. Le bus de commande de l'extension donne aux blocs périphériques les informations nécessaires concernant la nature du mot présent à cet instant sur le bus de données, sachant que ce dernier est utilisé pour un transfert bi-directionnel de données et de mots de commande/d'état.

Chaque bloc d'E/S comporte deux séries de 8 lignes d'E/S et les lignes de transfert des signaux d'acquiescement (handshaking) associées; la cartouche possède de ce fait en tout et pour tout 32 lignes d'E/S, plus qu'il n'en faut pour la quasi-totalité des applications.

L'ensemble timer comporte 4 dispositifs de comptage/timer rassemblés dans un unique circuit intégré.

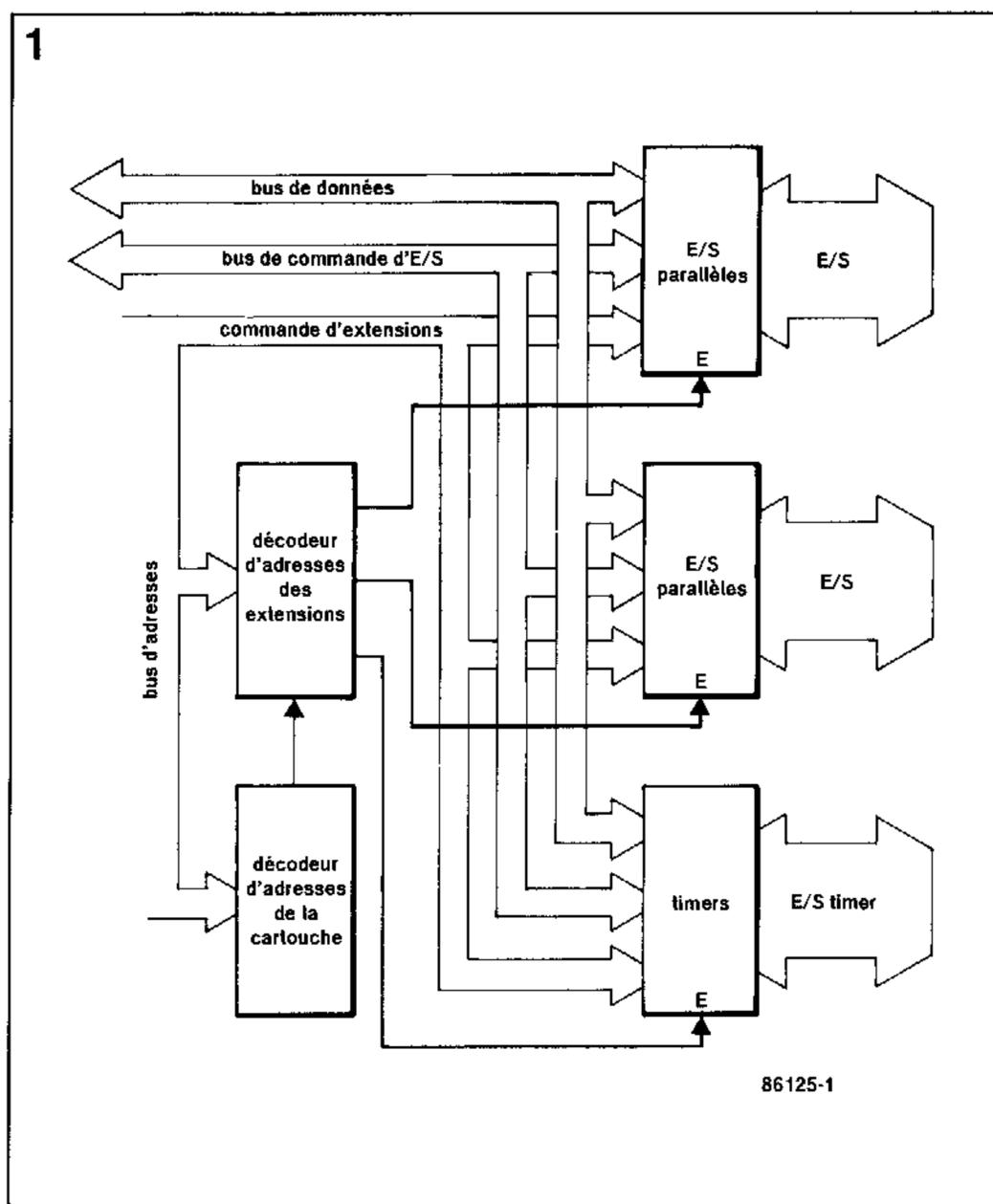


Figure 1. Synoptique de la cartouche-timer + interface d'E/S pour MSX.

Le côté matériel

Grâce à l'utilisation de trois circuits intégrés LSI de la famille des circuits périphériques pour Z80, le schéma complet de la cartouche timer + interface d'E/S (figure 2) présente de fortes similitudes avec le synoptique de la figure 1.

Le décodeur d'adresses de la cartouche, IC5, compare une adresse de 4 bits, prédéfinie à l'aide de cavaliers, aux quatre bits des adresses A4...A7 fournis par l'unité centrale (la CPU) et active sa sortie A=B lorsque les deux configurations considérées sont identiques. Les 255 ports de sortie mentionnés précédemment peuvent être adressés par l'intermédiaire de l'octet de poids faible (LSB) du bus d'adresses (A0...A7), le signal IORQ indiquant qu'il s'agit d'un cycle d'E/S et non pas d'un cycle d'accès à la mémoire. En BASIC MSX, les instructions consacrées aux Entrées/Sorties sont tout simplement INP (xxx) et OUT xxx,n respectivement, instructions dans lesquelles xxx représente le port d'E/S et n l'octet à envoyer.

Sachant que les ports d'E/S 64 à 255 sont réservés au logiciel et matériel MSX standard, les lignes A6 et A7 du quartet (4 bits ou demi-octet) d'adresses préprogrammé sont forcées à la masse (niveau logique bas) de manière à supprimer tout risque de conflit d'E/S entre notre cartou-

che et le matériel pris dans la cartographie des ports d'E/S.

Le tableau 1 donne les configurations de cavaliers à adopter pour définir le bloc de 16 ports d'E/S ouvrant l'accès à la cartouche.

Il n'est pas nécessaire d'appliquer de signal d'échantillonnage IORQ au comparateur d'adresses IC5, sachant que les circuits LSI périphériques IC1...IC3 possèdent leur propre entrée IORQ à cet effet. IC4, un double décodeur/démultiplexeur 2 vers 4, fournit aux PIO (Parallel Input/Output = circuit d'E/S parallèle) et au CTC (Counter/Timer Controller = circuit compteur/temporisateur) les impulsions de validation de circuit CE (Chip Enable). Ces trois fonctions périphériques sont sélectionnées par la configuration de bits appropriée des lignes d'adresses A2 et A3, à condition bien évidemment que la sortie A=B de IC5 soit au niveau logique haut. A

noter que la sortie 3 du décodeur 1 de IC4 (baptisée IQ3) attaque l'entrée E2 du décodeur 2 (= strobe = échantillonnage, pour info, la barre de négation placée sur un signal indique que ce dernier est actif au niveau bas); le décodeur 1 sert de ce fait uniquement à inverser le signal disponible à la sortie A=B de IC5.

En cas de sélection des PIO ou du CTC par CE, ces derniers ont accès au bus de données de la CPU lorsque la ligne IORQ passe au niveau bas. Le sens de transfert des données, CPU vers périphérique ou inversement, est défini par l'état logique de la ligne RD. Nous avons pris des dispositions pour traiter les interruptions en provenance des PIO ou du CTC, en donnant aux sorties INT de IC1...IC3 une structure de fonction OU câblée. La connexion en noria (chaînage) des signaux IEI et IEO (Interrupt Enable In et Out, respectivement entrée et sortie de validation des interruptions) est essentiellement une technique de définition de la priorité des interruptions. Dans le cas de la cartouche, IC1 possède la priorité d'interruption la plus élevée, IC3 la plus faible. Lorsque IC1 active sa sortie INT, IC2 et IC3 se voient privés de leur possibilité d'émettre des interruptions en direction de la CPU. Avec ce système, les périphériques de "haut rang" (et donc de priorité élevée) "écrasent"

Tableau 1.

bloc d'E/S cartouche (décimal)	cavaliers
0-15	a b
16-31	a d
32-47	c b
48-63	c d

Tableau 1. Définition du domaine des adresses d'E/S occupé par la cartouche.

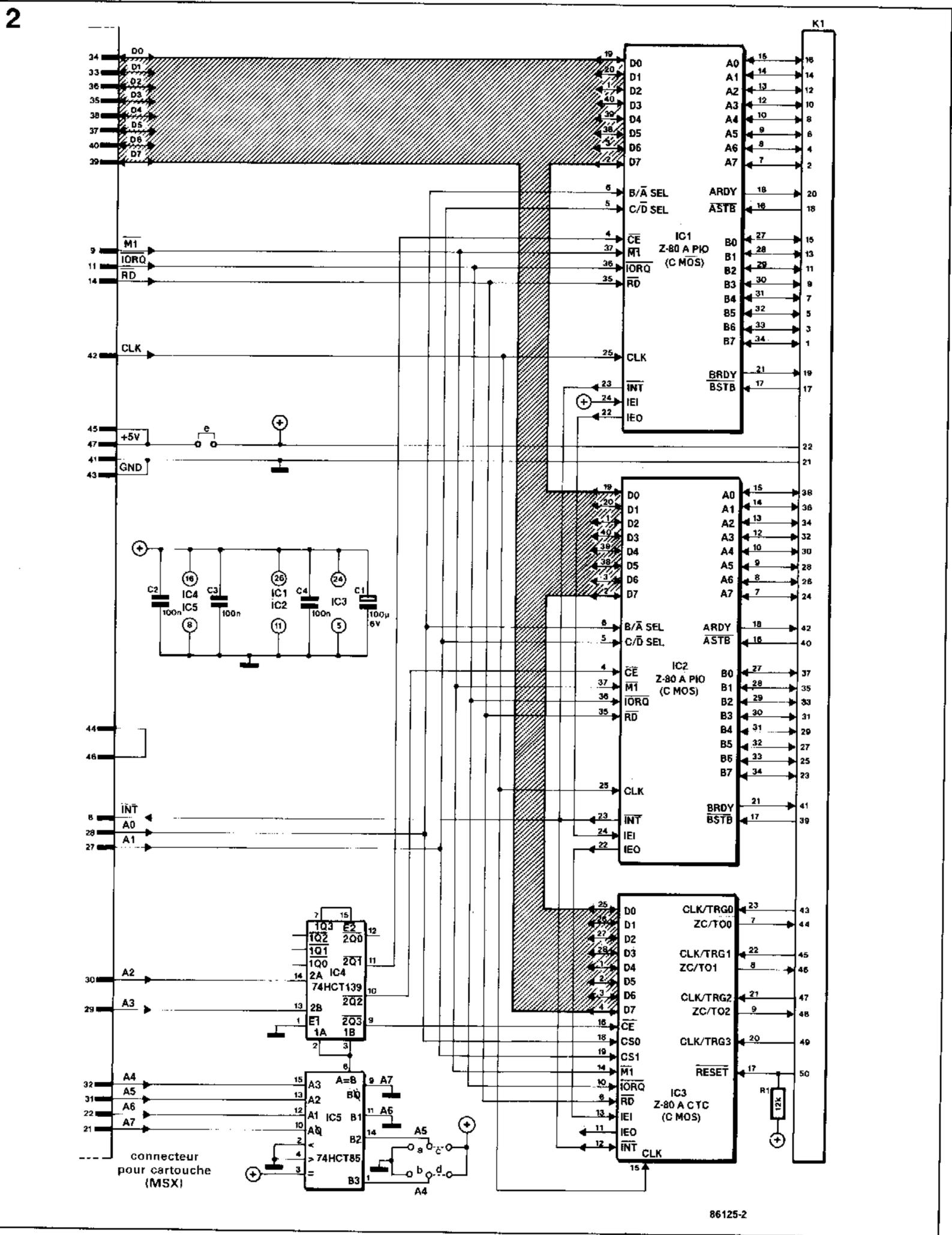


Figure 2. Schéma de l'extension timer + interface d'E/S pour ordinateurs MSX.

automatiquement les demandes d'interruption (\overline{INT}) de périphériques situés plus en aval de la noria (et ayant donc un niveau de priorité inférieur).

A la réception d'une impulsion \overline{INT} , la CPU "prend le pouls" des différents périphériques auxquels elle est reliée de manière à déterminer l'origine de cette demande d'interruption, processus qui prend la forme de l'envoi d'une impulsion \overline{INTACK} (Interrupt ACKnowledge = accusé de réception de l'interruption) réalisée par la combinaison des signaux $\overline{M1}$ et \overline{IORQ} . En réponse à ce signal, le périphérique concerné

place un octet vecteur sur le bus de données. Ce vecteur sert alors d'octet d'adresse de poids faible pour le sous-programme de traitement de l'interruption. Dans le cas d'un système à Z80, les impulsions $\overline{M1}$ et \overline{IORQ} sont utilisées pour fabriquer l'impulsion \overline{INTACK} alors que le vecteur d'interruption est chargé dans les périphériques au cours du sous-programme d'initialisation. Sur la cartouche, le niveau de priorité le plus élevé a été attribué au PIO IC1, pour la simple raison que le PIO IC2 et le CTC IC3 ne sont pas utilisés lorsque cette cartouche est utilisée avec **Microscope**.

L'alimentation de la cartouche peut se faire soit par l'intermédiaire d'une tension de 5 V prélevée sur l'ordinateur, soit à l'aide d'une alimentation externe connectée aux points 21 (GND = masse) et 22 (+5 V) du connecteur de sortie à 50 broches K1 (supprimer le strap e). Si vous utilisez la version CMOS des circuits intégrés, l'ordinateur ne devrait pas avoir de problème à fournir le courant nécessaire, une alimentation externe ne se justifiant donc pas (le strap e reste bien évidemment en place dans ce cas). Théoriquement, l'utilisation de circuits intégrés NMOS pour IC1...IC3 exigerait une ali-

mentation externe pour la cartouche, sachant que dans le cas le plus défavorable, la consommation de cette dernière peut théoriquement atteindre environ 320 mA, valeur qui dépasse légèrement le courant maximal qu'est en mesure de fournir le connecteur d'extension (300 mA). Cependant, en pratique les choses sont bien moins dramatiques, les mesures effectuées donnent une consommation de quelque 100 mA avec des circuits NMOS, courant que l'alimentation de l'ordinateur devrait être en mesure de fournir sans être en surcharge.

Ces observations montrent que la meilleure solution consiste à déterminer la consommation réelle de la cartouche en cours de fonctionnement et, au vu de ces résultats, à opter pour l'alimentation par l'ordinateur ou l'adjonction d'une alimentation externe.

La programmation des PIO

Le PIO Z80A de Zilog possède deux ports de 8 bits qui peuvent adopter 4 modes de fonctionnement différents, modes sélectionnés par logiciel en écrivant l'octet correspondant dans le registre de commande du circuit. L'état logique de l'entrée B/A SEL indique quel est le port (A ou B) que l'on veut adresser (lecture ou écriture), le bit présent à la broche C/D SEL indique s'il faut transférer, via le bus de données de 8 bits, un mot de commande/d'état ou de donnée. Les lignes d'adresses A0 et A1 attaquent respectivement les lignes B/A SEL et C/D SEL, permettant ainsi à l'utilisateur de faire fonctionner chaque PIO selon l'un des quatre modes possibles. Le MODE 0 met les ports A et B en mode sortie par octet, le MODE 1 en mode entrée par octet, le MODE 2 les positionne en mode bidirectionnel par octet, le MODE 3 les met en mode commandé (bidirectionnel programmable bit par bit).

Les modes 0, 1 et 2 sont commandés par des interruptions et ne peuvent de ce fait être utilisés qu'avec une CPU Z80 programmée pour opérer par interruption en mode 2, ce mode exigeant que l'ordinateur exécute un programme en langage machine définissant l'adresse du sous-programme de traitement de l'interruption. Dans le cas de l'ordinateur MSX, cependant, il faut commencer par invalider les interruptions générées par le VPD (Video Display Processor) à l'aide des instructions VPD(1) = VPD(1) AND 223. Après traitement de l'interruption générée par la cartouche, les interruptions d'affichage doivent être revalidées par repro-

grammation du Z80 en mode d'interruption 1 et après exécution de l'instruction VPD(1)=VPD(1) OR 32.

En raison de la complexité de la séquence de programmation évoquée plus haut, il nous a paru utile de voir de plus près le mode 3 du PIO qui permet une programmation semi-directe de la cartouche, c'est-à-dire en BASIC, sans avoir à se soucier des intrusions des sous-programmes de traitement des interruptions. Les utilisateurs de systèmes MSX désireux d'utiliser les PIO en MODE 0, 1 ou 2 ont tout intérêt à lire l'un et/ou l'autre des gros ouvrages proposés par Zilog (ou Mostek), tels que *Components Data Handbook*, (*Z80 Microcomputer Data Book*) ou le *Z80 Application Handbook*.

La séquence d'instructions suivante initialise le PIO en MODE 3:

- Octet de mode de commande = &HFF (définition du MODE 3);
- Octet de commande du registre d'E/S = &Hxx (voir l'exemple donné plus loin);
- Octet de commande de l'interruption = &H07 (invalidation des interruptions)
- Octet d'invalidation d'interruption = &H03 (n'est pas toujours nécessaire);

L'octet écrit dans le registre d'E/S du PIO détermine si les lignes individuelles sont des entrées (niveau logique bas) ou des sorties (niveau haut). Exemple: l'envoi de l'octet &HF0 au registre A d'E/S définit comme entrées les lignes de port A0...A3 et comme sorties les lignes A4...A7.

Après exécution de la routine d'initialisation, il est possible de recevoir ou d'émettre des données par l'intermédiaire des lignes de port. Il va

sans dire que chacun des ports doit être initialisé de la manière indiquée plus haut, ce que l'on réalise par sélection du circuit concerné (lignes d'adresses d'E/S A2 et A3), du port choisi (A/B) et de l'accès commande/donnée selon le cas. L'ensemble du processus est effectué par une série d'instructions d'écriture à des adresses situées à l'intérieur du domaine d'E/S de la cartouche.

La programmation du CTC

Le circuit CTC Z80 comprend 4 circuit de comptage/timer configurables indépendamment. La fonction de chacun des bits de l'octet de commande du CTC est indiquée dans le **tableau 2**. La constante de temps adoptée (bit D2) détermine le nombre d'impulsions nécessaire avant que la sortie ZC/TO ne passe au niveau haut. Chaque compteur/timer continue de fonctionner tant que le CTC n'a pas reçu d'initialisation logicielle (D1) ou de remise à zéro matérielle (RESET, broche 17).

La réalisation

Sachant que le module timer + E/S doit finir sous la forme d'une cartouche encartable dans le connecteur prévu pour ce type d'extension sur tout ordinateur MSX, il ne saurait y avoir de doute sur la nécessité de disposer d'un circuit imprimé double face à trous métallisés tel celui illustré par la **figure 3**. Etant donné le faible nombre de composants, il ne devrait pas y avoir de problème si tant est que l'on soigne les soudures, car la densité de pistes est

Tableau 2. Fonction bit par bit du registre de commande du Z80 CTC.

Tableau 2.

bit du CTC	fonction		note(s)
	bas (0)	haut (1)	
D0	vecteur	octet de commande	
D1	-	remise à zéro logicielle	
D2	l'octet suivant n'est pas une constante de temps	l'octet suivant est une constante de temps	
D3	déclenchement au premier flanc montant suivant le chargement de la constante de temps	démarrage du timer par impulsion d'horloge externe	en mode timer uniquement
D4	flanc descendant	flanc montant	utilise le signal d'horloge CLK
D5	facteur de division: 16	facteur de division: 256	en mode timer uniquement
D6	mode timer	mode compteur	
D7	validation de INT	non validation de INT	

Figure 3. Sérigraphie de l'implantation des composants pour la cartouche timer + interface d'E/S.

Liste des composants

Résistances:

R1 = 12 k

Condensateurs:

C1 = 100 μ /6 V

C2...C4 = 100 n

Semiconducteurs:

IC1, IC2 = Z80A PIO (CMOS)

IC3 = Z80A CTC (CMOS)

IC4 = 74HCT139

IC5 = 74HCT85

Divers:

K1 = barrette de picots tronçonnable en équerre 2 x 25 broches au pas de 2,54 mm

1 barrette de 2 x 3 picots au pas de 2,54 mm

1 barrette de 1 x 3 picots au pas de 2,54 mm

3 cavaliers femelles permettant de court-circuiter 2 picots au pas de 2,54 mm

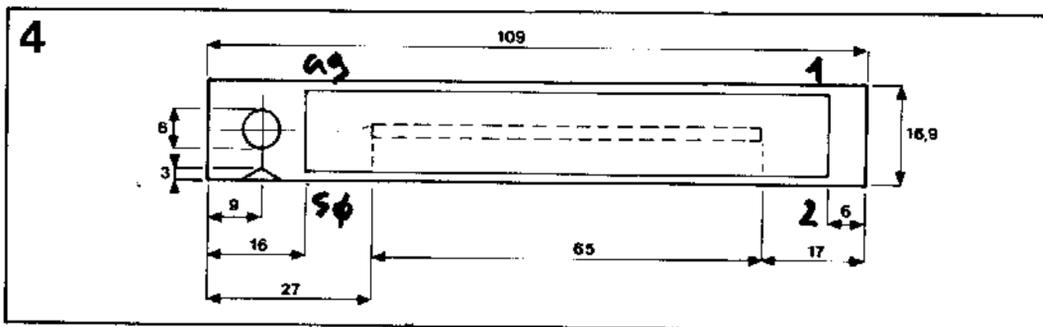
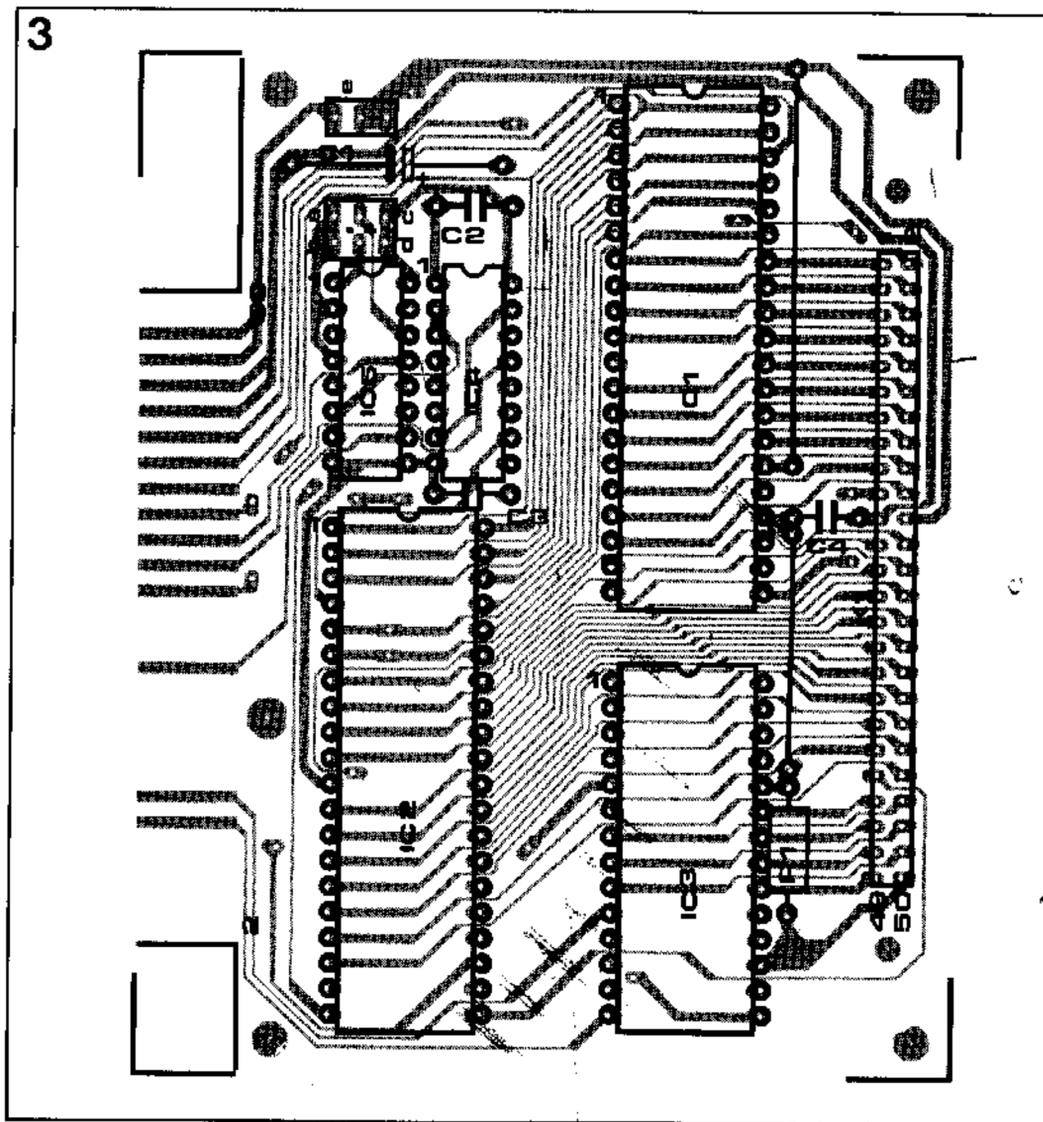


Figure 4. Dimensions de la découpe rectangulaire à effectuer dans le boîtier pour cassette audio dans le but de permettre l'implantation du circuit imprimé.

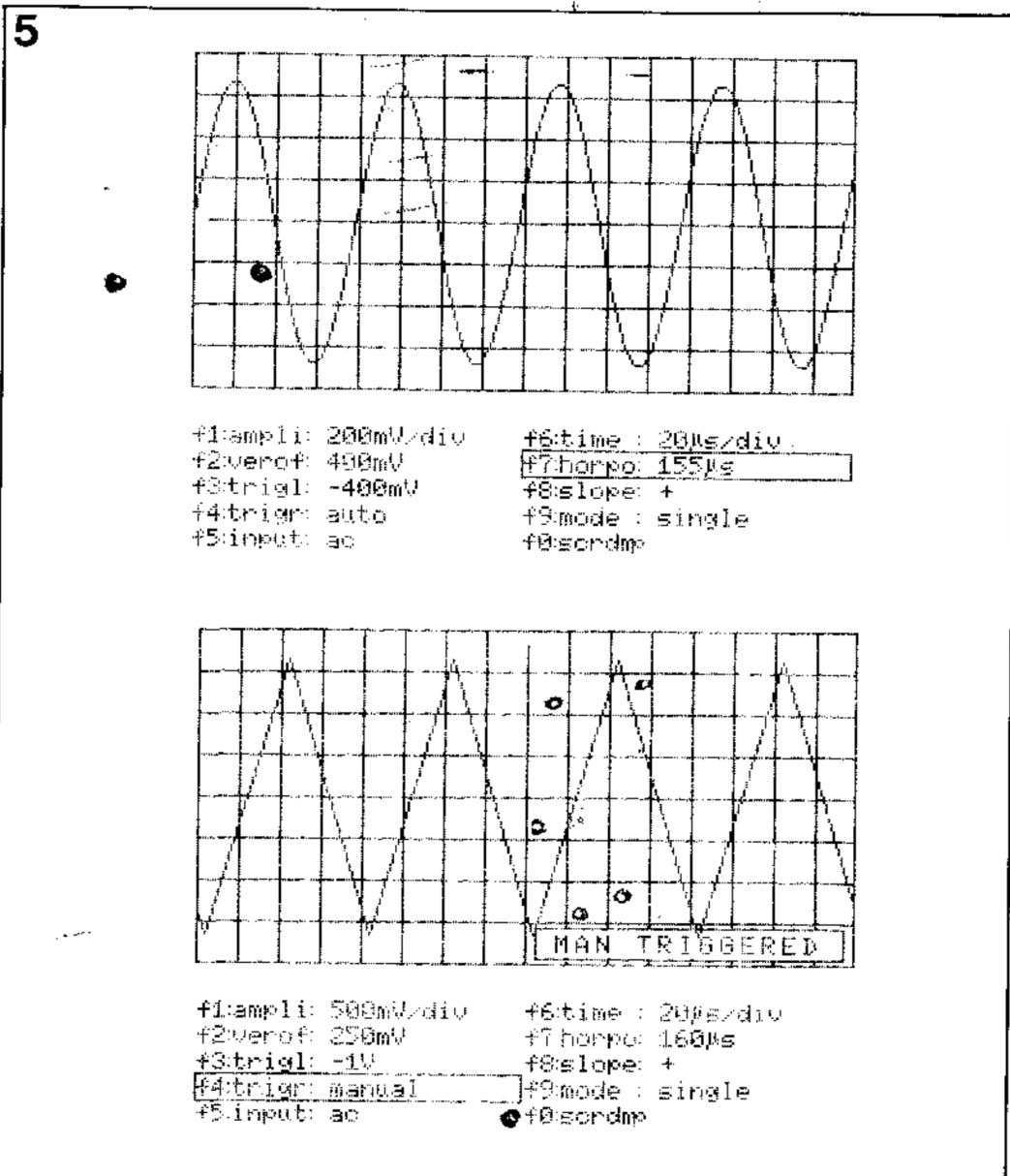


Figure 5. Exemples de deux signaux analysés par le Microscope rendus par impression sur papier.

relativement élevée (le masque de soudage supprimant tout risque de court-circuit à ce niveau, à condition bien évidemment de ne pas faire de pâtes). Il est un point important dont il faudra s'assurer avant de commencer l'implantation des composants sur le circuit imprimé: vérifier que ses dimensions en permettent l'implantation dans le boîtier d'une cassette audio. Il est plus que probable que l'utilisation de supports ordinaires risque de poser des problèmes insurmontables dans ce cas et il y a de grandes chances qu'il faille souder les circuits intégrés directement sur le circuit imprimé si l'on veut pouvoir fermer convenablement le boîtier. Il faudra en outre supprimer les ergots de centrage des bobines et autres "éléments contondants" que pourrait comporter le boîtier et, comme l'illustre la **figure 4**, percer une fente permettant le passage du connecteur.

De cette manière, on réalisera une cartouche de fabrication maison capable de supporter de nombreux enfichages dans le connecteur sans poser de problèmes de contact côté connecteur encartable de la cartouche.

Le logiciel MSX pour le Microscope

Les techniques de programmation nécessaires au fonctionnement de **Microscope** décrites dans les numéros de septembre et d'octobre derniers pour les ordinateurs des types Electrons C64 ou BBC restent valables pour le logiciel MSX fourni avec le circuit imprimé de **Microscope**. Cependant, la résolution limitée de l'écran d'un ordinateur MSX implique une disposition légèrement différente pour les explications des commandes exécutées par les touches de fonctions de l'ordinateur MSX (voir **figure 5**). Ces touches permettent d'invoquer les différentes commandes du scope, les touches de positionnement du curseur permettent, quant à elles, de donner la valeur désirée à chacun des paramètres.

En raison de la limitation imposée par la résolution maximale d'un écran MSX (192 x 256 pixels), il n'a pas été possible de doter les ordonnées ou les abscisses de repères chiffrés.

Passons rapidement en revue les fonctions exécutables par actions sur les touches F1 à F9:

F1 définit l'amplitude que l'on veut donner au signal à visualiser.

F2 et **F3** servent respectivement à définir l'offset vertical et le niveau de déclenchement, ce qui implique la

visualisation d'un niveau de tension absolu; sachant en outre que le niveau de déclenchement est pris dans l'octet échantillonné, la modification de l'offset vertical entraîne aussi un changement du niveau de déclenchement. L'ordinateur visualise le niveau de déclenchement ainsi défini sous la forme d'une petite barre clignotante.

La division de l'écran (le graticule) peut être définie par incréments de 1 pixel (curseur haut/bas) ou par incréments de 8 pixels (curseur gauche/droit), un fonctionnement similaire ayant été adopté pour la touche de fonction F7.

F4 sélectionne le mode de déclenchement: automatique, manuel ou externe. Le mode de déclenchement automatique donne à l'ordinateur la faculté de définir le niveau de déclenchement après une action de l'utilisateur sur la barre "espace". Pour les modes manuel et externe, l'ordinateur attend une seconde action sur cette barre, action indiquant une impulsion de déclenchement manuel ou une impulsion de validation de déclenchement (EXT). F5 sélectionne le mode d'entrée: tension alternative (AC), tension continue (CC) ou masse (GND = 0 V).

F6 définit la base de temps.

F7 détermine la position horizontale de l'instant de déclenchement.

F8 choisit un déclenchement sur un flanc soit montant soit descendant.

F9 sert à définir le mode de visualisation: affichage unique (+ effacement), affichage continu (+ effacement) ou affichage continu. Une action sur la touche DEL provoque l'effacement de l'écran avant la visualisation d'un nouveau signal.

F10 permet de transférer le contenu de l'écran vers une imprimante (screen dump mode). La routine d'initialisation implantée dans la version MSX du logiciel pour **Microscope** est spécifique des imprimantes de la famille Smith Corona; son utilisation avec une imprimante d'un type différent peut nécessiter la réécriture de cette partie du programme pour la rendre compatible avec le mode image bit par bit et la disposition des aiguilles de la tête d'impression de l'imprimante concernée. Pour peu que l'on ait quelques connaissances de programmation en langage machine, écrire sa propre routine de visualisation sur imprimante ne devrait pas être une tâche impossible, d'autant plus que la version pour Smith Corona constitue un exemple très parlant.

Le **tableau 3** donne un programme de test rudimentaire permettant de vérifier le fonctionnement de la cartouche et du circuit de **Microscope**, ce programme étant de la même

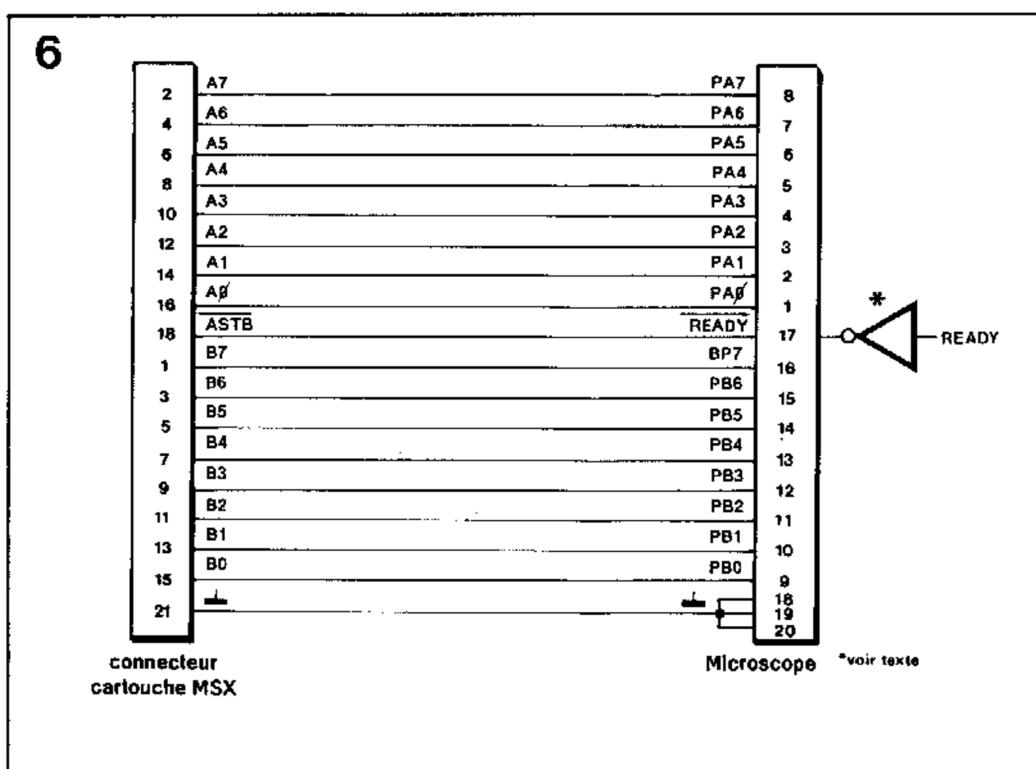


Figure 6. Plan des connexions à effectuer entre la cartouche et Microscope.

veine que ceux donnés pour le C64, le BBC et l'Electron (octobre 1986, page 70). La cartouche est connectée au **Microscope** de la manière illustrée par la **figure 6**. On y voit que les lignes de communication avec le PIO, ARDY (port A prêt) et \overline{ASTB} (strobe du port A), ne sont pas utilisées dans la disposition de base. Cependant, pour améliorer la vitesse de communication entre l'ordinateur et **Microscope** on pourra envisager d'implanter l'un des inverseurs N16-N19 inutilisés pour inverser le signal READY disponible à la sortie du **Microscope**. Il est à noter cependant que le logiciel MSX fourni avec le circuit imprimé repose sur un fonctionnement du PIO en MODE 3, comme nous l'avons vu en détail plus haut et qu'il ne supporte pas les signaux d'acquiescement (handshaking). L'écriture d'une routine de traitement d'interruption permettant l'utilisation de \overline{ASTB} tout en maintenant intacte la chronologie de

l'affichage d'écran MSX (VDP) peut constituer un challenge pour un programmeur expérimenté. Quoi qu'il en soit, l'écriture d'un tel sous-programme est une tâche réservée aux spécialistes et si vous ne vous sentez pas en mesure d'arriver à la réaliser, il vous suffira de laisser la ligne \overline{ASTB} en l'air, le transfert des informations restera suffisamment rapide.

Chaque semaine voit l'arrivée sur le marché de nombreux ouvrages nouveaux destinés aux propriétaires d'ordinateurs MSX, la plupart d'entre eux étant malheureusement en anglais (internationalisation oblige), trop peu en français.

Le prochain article consacré à un montage pour ordinateur MSX décrira un programmeur d'EPROM pour MSX, appareil qui utilisera comme interface la cartouche timer + interface d'E/S que nous venons de décrire.

Tableau 3. Version MSX du programme de test pour le Microscope.

Tableau 3.

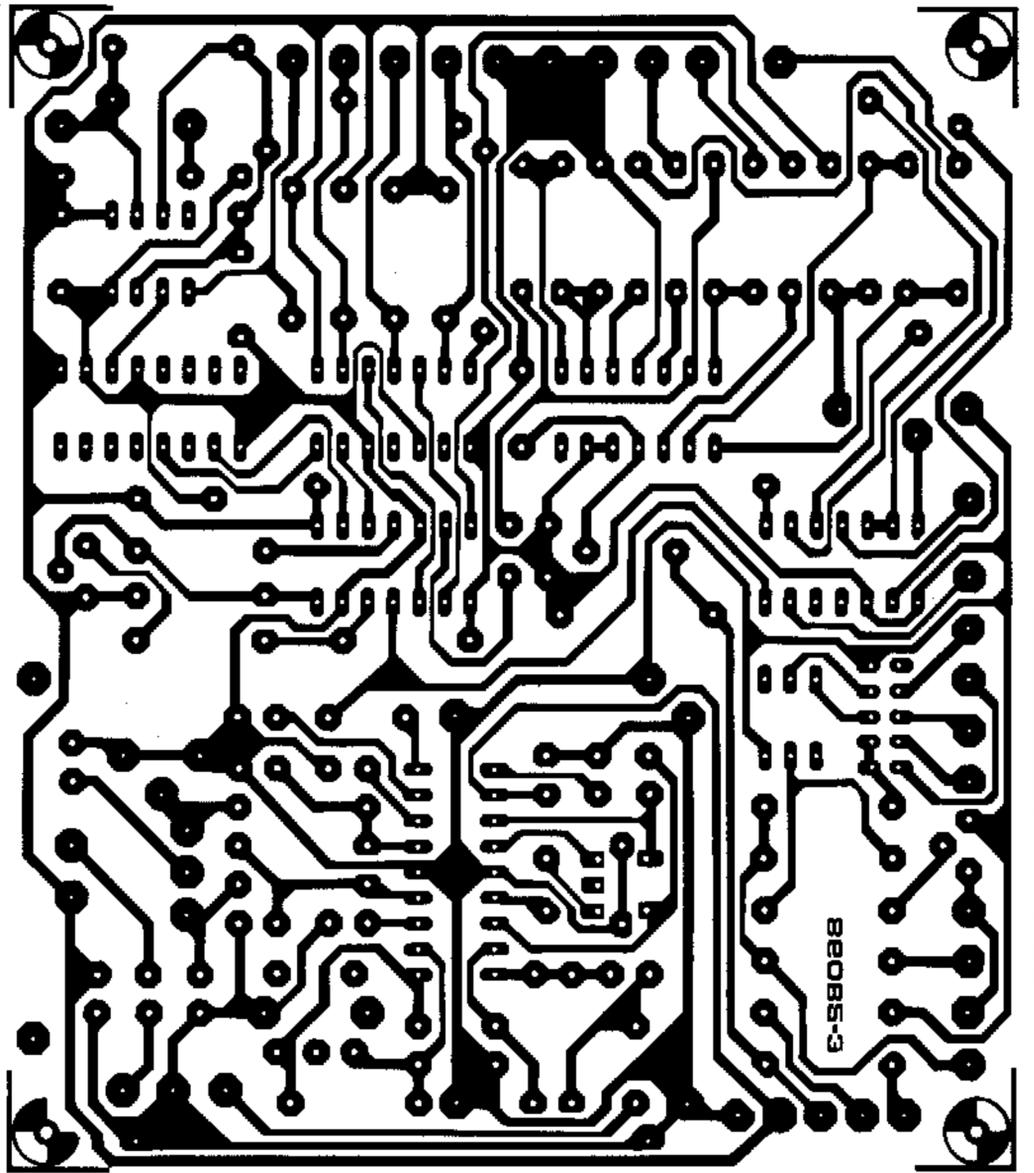
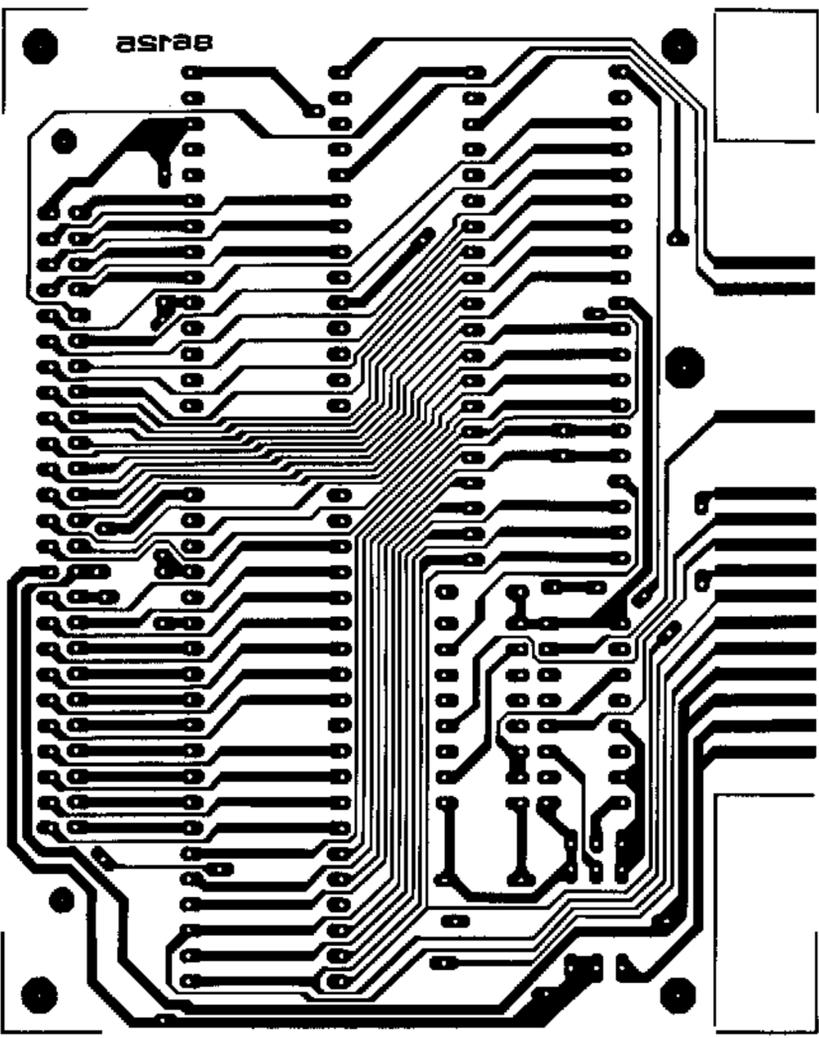
```

10 SCREEN 2
20 A=3*16
30 DA=A+4: DB=A+5: CA=A+6: CB=A+7
40 OUT CA,255: OUT CA,0: OUT CA,7: OUT DA,&H10
50 OF=0: IN=1: NI=0: TH=0: TB=8: AM=8: TR=0
60 OUT CB,255: OUT CB,0: OUT CB,7
70 OUT DB,(OF+64+128*IN): OUT DA,&H14
80 OUT DB,(NI+64+128*TH): OUT DA,&H12
90 OUT DB,(TB+16*AM): OUT DA,&H11
100 OUT CB,255: OUT CB,255: OUT CB,7: OUT DA,0: OUT DA,&H40: OUT DA,&H10
110 HO=TIME+(TB+1)*50
120 IF HO>TIME THEN 120
130 IF TR=0 THEN OUT DA,&H30
140 IF TR=1 THEN OUT DA,&H38
150 IF TR<>2 THEN 160 ELSE IF INKEY*=" " THEN OUT DA,&H90 ELSE 140
160 HO=TIME+3*(TB+1)*50
170 OUT DA,0: OUT DA,&H20: OUT DA,0
180 CLS
190 PSET(0,65)
200 FOR I=0 TO 255 STEP 2
210 LINE -(I/2,150-INP(DB)/2)
220 OUT DA,&H40: OUT DA,0
230 OUT DA,&H40: OUT DA,0
240 NEXT
250 OUT DA,&H20
260 FOR I=256 TO 512 STEP 2
270 LINE -(I/2,150-INP(DB)/2)
280 OUT DA,&H60: OUT DA,&H20
290 OUT DA,&H60: OUT DA,&H20
300 NEXT
310 GOTO 50
    
```

SERVICE

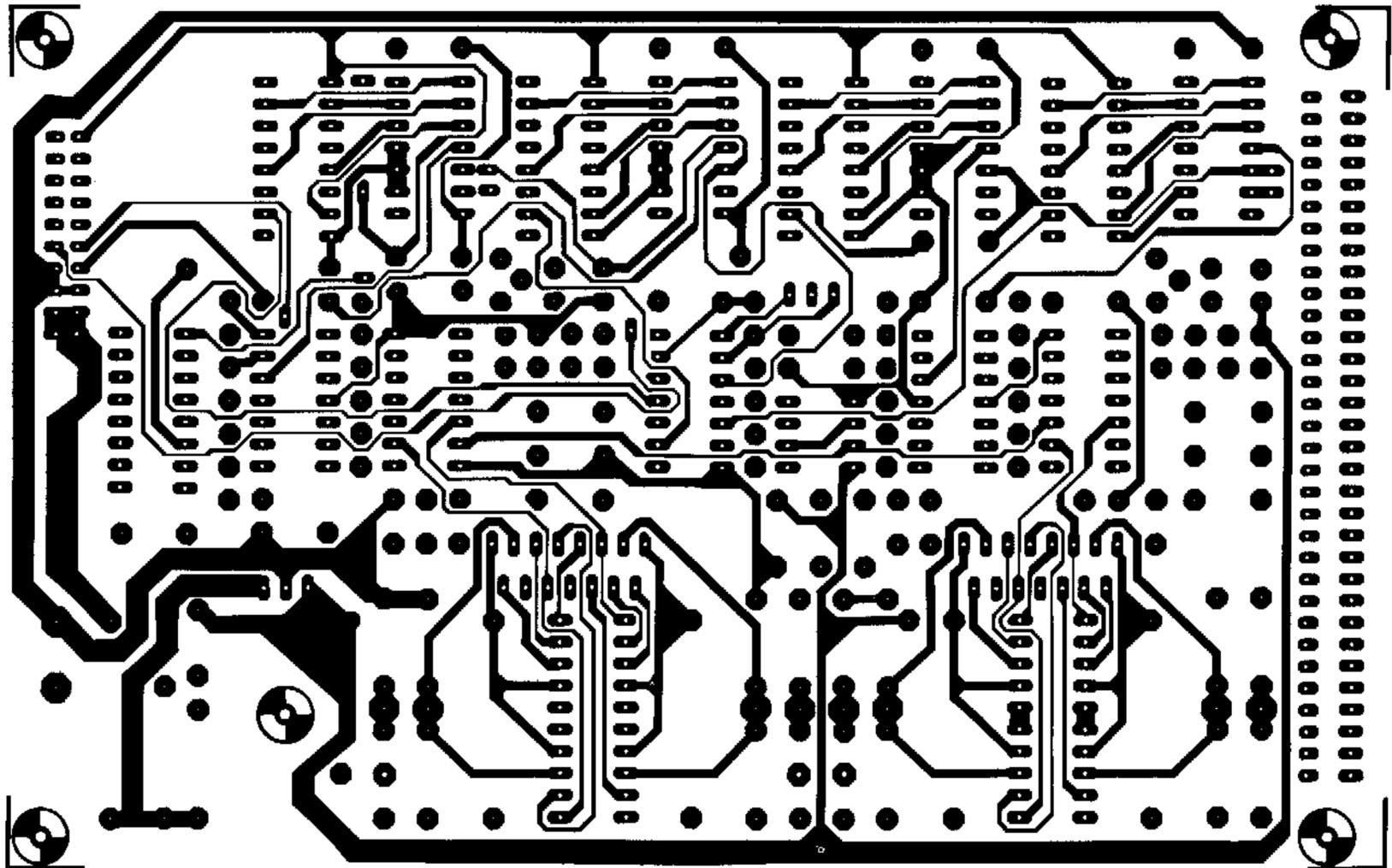
réception TV par satellite: les accessoires

cartouche timer + E/S 32 bits: côté pistes



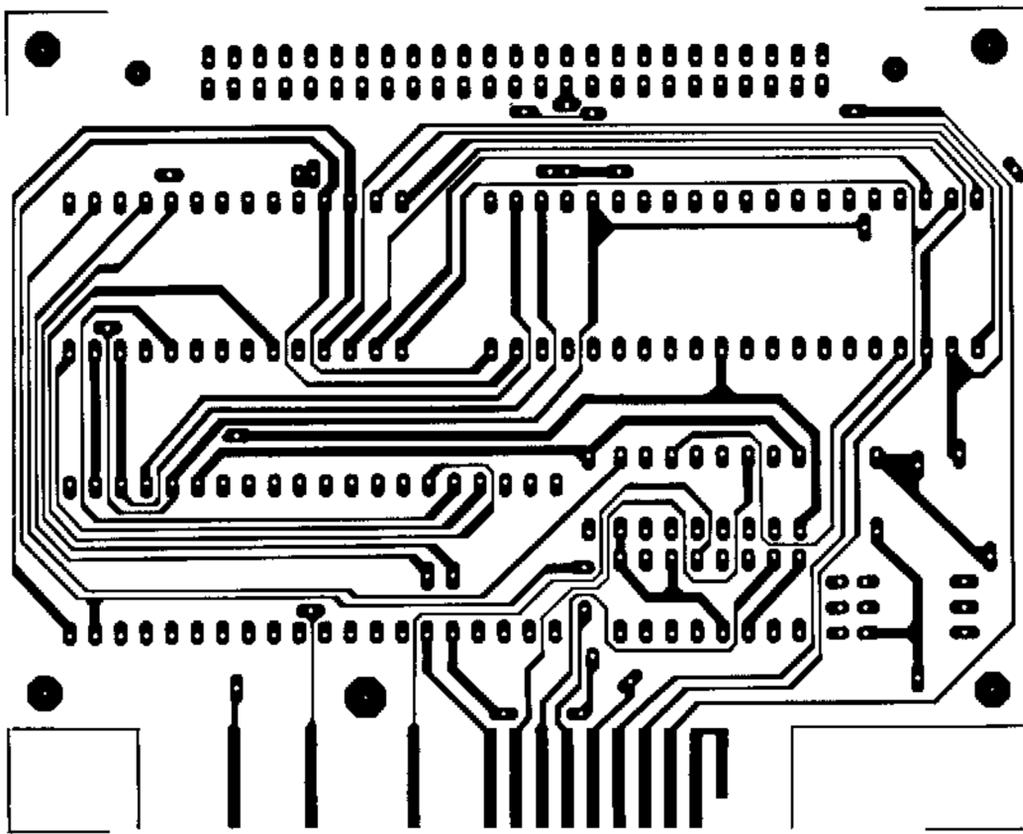
Le manque de place ne nous permet pas de donner les dessins recto/verso des pistes de la platine principale de "the preamp". A noter que les platines de la cartouche timer + E/S 32 bits et de la commande de moteur pas à pas sont des circuits double face à trous métallisés.

commande de moteur pas à pas: côté composants

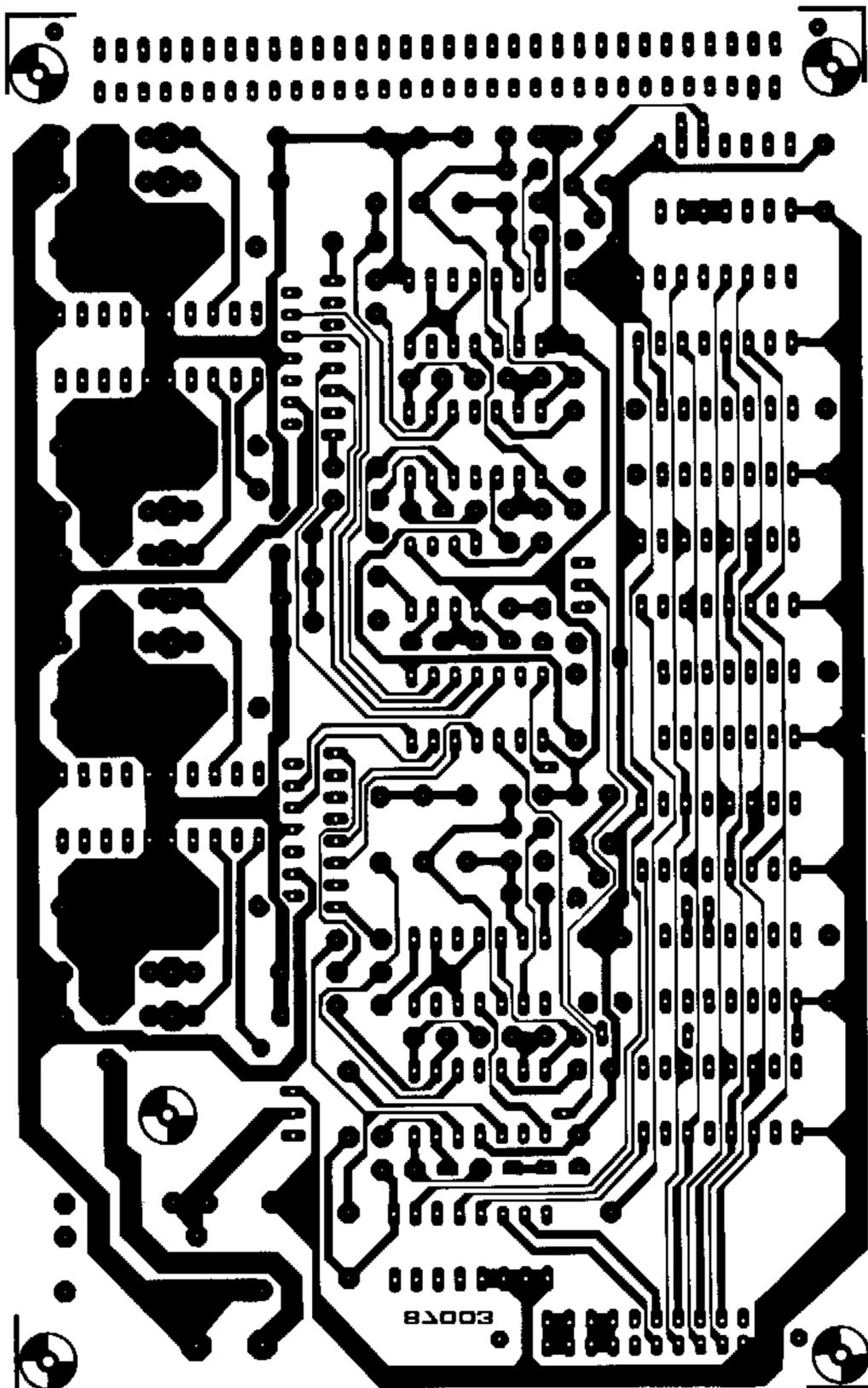


SERVICE

cartouche timer + E/S 32 bits: côté composants



commande de moteur pas à pas: côté pistes



sinus numérique

