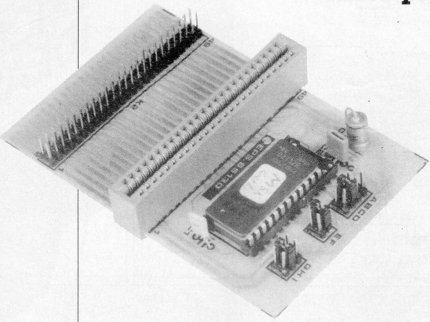
Cartridge board with user-programmable EPROM



MSX EXTENSIONS - 2

Second in the series on home-made MSX add-on units, this article presents a cartridge extension board and full details on EPROM-stored programs.

As evidenced by the first part in this series *Elektor India* February 1986), the cartridge slot available on MSX type computers may be used to effect connection of home-made extensions like the Elektor universal I/O bus.

Usually, commercially available cartridges merely contain an (E)PROM to run a program (game, utility). It is, therefore, possible to construct a device that will hold user-programmed EPROMs whilst retaining the possibility to insert existing cartridges. Our design offers the following facilities.

1. Easy connection of further hardware-extensions, like the Elektor universal I/O bus.
2. The present board may be connected to the existing 50-way output port of such MSX computers as the Spectravideo type.
3. The board may be used as an angled cartridge adapter or a versatile IC socket to hold several types of user-programmable EPROMs with 2, 4, 8, 16, or 32 Kbytes capacity.
4. The board is useful for the connection of a Yamaha synthesizer.

The MSX cartridge

As shown above, the present cartridge extension board is the sort of design that many users would undoubtedly like to see: universal, accessible for measurements and experiments and with the possibility to insert one's own EPROMs. However, before this can all come true, some knowledge is required of the 'cartridge conventions' used in MSX BASIC. We shall, therefore, first examine a typical MSX start-up procedure.

After power-on, MSX BASIC always establishes the amount of RAM (Random Access Memory) between addresses 8000 and FFFF, and activates the largest continuous area encountered. Next, BASIC examines slot address range 4000...BFFF. Each slot occupies 16 Kbytes, divided in four pages. At the beginning of every page, a sequence of codes is read to identify the slot contents. The bytes which supply this information are located in a fixed order, as shown in Fig. 1. The function of each code is as follows:

ID (identification): a two-byte code that indicates the presence of a cartridge (E)PROM. In that case, BASIC reads 41_{hex} and 42_{hex} (ASCII A and B), respectively at these locations.

INIT (initialization): a vector (address pointer) for the initialization routine associated with the cartridge function. In case this is not required, a default value 0000 is present at these locations.

STATEMENT: a vector pointing to the cartridge statement-handler, if applicable. If not, a default 0000 is present. For further details on this vector, refer to the user manual supplied with the computer or the cartridge.

DEVICE: a vector pointing to the cartridge device-handler, if applicable. If not, a default 0000 is present. Refer to computer manual for further details.

TEXT: a vector pointing at the token-coded BASIC program text in the cartridge. This pointer is of great interest to users who want to put their own BASIC programs into EPROMs. All foregoing addresses are stored in the cartridge (E)PROM with their least significant byte (LSB) first, as is customary in Z80 machine language programming.

Practical circuit

Actually, the present design, as shown in Fig. 2, is not much of a circuit at all; it is rather a truly universal and user-friendly IC socket for the 27XX series of EPROMs, ranging from the well-known Type 2716 (2 Kbytes) to the giant Type 27256 (32 Kbytes). Note that EPROM manufacturers have generally agreed on using the last two or three digits of the type indication to state the memory capacity in kilobits. Divided by eight, this will give the number of programmable bytes (one byte equals eight bits).

To accommodate every member of the 27XX family, the present extension board has a number of jumpers, which will have to be installed or removed as follows:

jumper A selects between Types 27128 and 27256 EPROMs and should be installed with the latter type inserted.

jumper B connects terminal 27 of a Type 27128 to +5V. Thus: jumper A for a 27256, jumper B for a 27128.

jumper C connects V_{cc} terminal 24 of 24-pin Types 2716 and 2732 to +5V.

jumper D connects address line A_{13} to terminal 26 of 28-pin Types 27128 and 27256. For the 2764, jumper C must be installed (pin 26 to +5V, not **both** jumpers C **and** D).

jumper E connects terminal 23 (28-pin types) or terminal 21 (2732) to A_{11} and must be installed for all EPROMs except Type 2716.

jumper F connects V_{pp} terminal 21 of a Type 2716 EPROM to +5V.

jumpers G, H, and I connect the EPROM $\overline{CE}$ terminal (chip enable) to MSX signal $\overline{CS1}$, $\overline{CS2}$ or $\overline{CS12}$ in that order, $\overline{CS1}$ being the ROM select signal valid for address range 4000...7FFF, $\overline{CS2}$ for 8000...BFFF, and $\overline{CS12}$ for both ranges, i.e. 4000...BFFF. Up to and including a Type 27128 EPROM, either $\overline{CS1}$ or $\overline{CS2}$ is used; a Type 27256 requires the $\overline{CS12}$ signal. Table 1 summarizes all available jumper configurations in order that any user can readily find and set the jumper combination as required for the EPROM in use.

So far, only EPROMs have been mentioned because these are most readily available and programmable. However, it will be evident that pin-compatible proprietary PROMs or ROMs will work just as well.

If fitted in the MSX computer, the in-

1

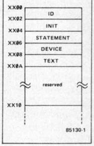


Fig. 1 These codes at the beginning of every slot address-block form a software 'visiting card' of the cartridge, for identification by MSX BASIC.

2

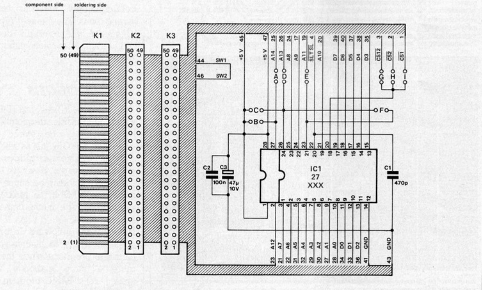
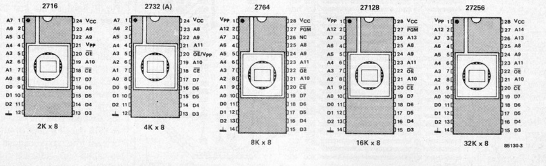


Fig. 2 Practical circuit of the cartridge extension board. The jumpers are set to suit the type of EPROM used (2...32 Kbyte).

Fig. 3 Pin designations of the popular 27XX series of EPROMs, arranged in order of memory capacity.

3



Listing 1 This memory dump program may be used to analyse larger BASIC programs as they reside in RAM; it provides hexadecimal presentation of any given memory area and may be put in EPROM to function as a stand-by utility cartridge.

Listing 1.

```
DUMP
10 CLS
20 INPUT"start";A
30 INPUT"end";B
40 FOR C = A TO B
50 LPRINTUSING"\   \";HEX$(C);:LPRINT"  ";
60 FOR D=0 TO 15
70 LPRINTUSING"\\";HEX$(PEEK(C+D));:LPRINT" ";
80 NEXT
90 C=C+15:LPRINT" ":LPRINT" "
100 NEXT
110 END
```

Parts list

Capacitors: C_1 = 470p
C_2 = 100n
C_3 = 47μ;10V

Semiconductors:
IC_1 = 2716,2732;2764; 27128;27256 or corresponding pin-compatible (P)ROMs.

Miscellaneous:
K_2 = 50-way (2 × 25) male PCB connector
K_3 = 50-way (2 × 25) MSX female slot connector, 0.1 inch pitch
18-way (2 × 9) male locking plug assembly for female jumpers (e.g. Minicon Latch PI 18w).
4 jumpers
PCB 85130 (65.5 × 98mm)

4

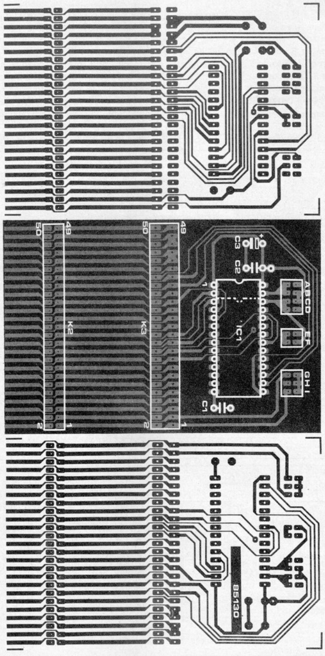


Fig. 4 This through-plated PCB is small but effective when it comes to plugging in existing cartridges, hardware extensions, or EPROMs holding user programs.

sert/remove protection circuitry will detect the connection between SW1 and SW2 as present on the extension board.

Three connectors are provided on the board; K_1 is simply the edge of the extension board with connecting copper tracks on both sides for insertion in the computer cartridge slot; PCB connector K_2 is a standard male 50-way type (2 rows of 25 pins); while K_3 is a cartridge slot connector with 0.1 inch pitch contacts, just as the one inside the computer.

Construction

Track layout and component mounting plan of the cartridge extension board are shown in Fig. 4. The ready-made PCB is a moderately sized, through-plated type, available as usual through our Readers Services. The soldering islands and slot connecting tracks have been pre-tinned to guarantee stable contacts. Use of a 28-way ZIF (zero insertion force) socket is highly recommended because sooner or later EPROMs will have to be taken out, erased with a UV source, programmed again, debugged, etc., and this perhaps several times. The cheaper types of IC socket will inevitably develop bad terminal contacts after prolonged use...

Applications

Now that a neat, universal (E)PROM socket is available, frequently used programs may be stored in a dedicated EPROM, just as with commercially available cartridges, but a good deal cheaper. However, before user programs may be successfully stored in EPROM, the MSX BASIC program storage method needs to be unravelled.

Note that the following description does not apply to machine-coded cartridge programs, since these require a more elaborate vector system. For a BASIC program, then, the ID and TEXT vectors are essential; they are located at XX00–XX01 and XX08–XX09 respectively (see Fig. 1). Because the first 16 bytes of the cartridge (E)PROM are reserved for program identification and system vectors, the token-coded BASIC program itself may be stored from location XX10 onwards.

MSX BASIC programs are generally stored in memory from address 8000 onwards, so the value 80 may be read for XX from now on.

At 8010, the CPU must invariably read byte 00. The next locations contain a so-called link address (two bytes) and a line number (also two bytes);

next comes a token-coded line of BASIC text, terminated with a byte 00. This procedure is repeated for the following text lines.

To find out the hexadecimal codes that constitute a program, it is necessary to run the DUMP program of Listing 1, preferably with a printer connected to the computer. In case a printer is not readily available, the bytes may be put on the screen by changing all LPRINT commands into PRINT and next changing value 15 into 7 in lines 60 and 90 to allow for the reduced number of printable characters per line. Note that the DUMP program may be 'attached' to any user program in memory by entering it from, say, line 10000 onwards.

After RUN 10000, the program prompts for a start and end-of-program address; the former is always &H8000, the latter depends on the actual size of the program, which

	A	B	C	D	E	F	G	H	I
27256	o—o			o—o	o—o		o—o		
27128		o—o		o—o	o—o			o*o	o*o
2764		o—o	o—o		o—o			o*o	o*o
2732			o—o		o—o			o*o	o*o
2716			o—o			o—o		o*o	o*o

85130-T1

o—o = jumper
* = select either H or I (see text)

Table 1 Summary of the necessary jumper configurations for every type of EPROM in the 27XX series. The choice between jumpers H and I depends on the selected memory area (see text).

Table 2.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8000	0	7	80	A	0	9F	0	16	80	14	0	85	22	87	74	61
		L8007		*10		Tk	EOL	L8016		*20		Tk	"	s	t	a
8010	72	74	22	3B	41	0	23	80	1E	0	85	22	65	6E	64	22
	r	t	"	;	A	EOL	L8023		*30		Tk	"	e	n	d	"
8020	3B	42	0	33	80	28	0	82	20	43	20	EF	20	41	20	D9
	;	B	EOL	L8033		*40		Tk	sp	C	sp	Tk	sp	A	sp	Tk
8030	20	42	0	4E	80	32	0	9D	E4	22	5C	20	20	5C	22	3B
	sp	B	EOL	L804E		*50		Tk	Tk	"	\	sp	sp	\	"	;
8040	FF	9B	28	43	29	3B	3A	9D	22	20	20	22	3B	0	5D	80
	Tk	Tk	(	C	)	;	:	Tk	"	sp	sp	"	;	EOL	L8050	
8050	3C	0	82	20	44	EF	11	20	D9	20	F	F	0	7B	80	46
	*60		Tk	sp	D	Tk	0	sp	Tk	sp	Tk	15	EOL	L807B		*
8060	0	9D	E4	22	5C	5C	22	3B	FF	9B	28	FF	97	28	43	F1
	70	Tk	Tk	"	\	\	"	;	Tk	Tk	(	Tk	Tk	(	C	Tk
8070	44	29	29	3B	3A	9D	22	20	22	3B	0	81	80	50	0	83
	D	)	)	;	:	Tk	"	sp	"	;	EOL	L8081		*80		Tk
8080	0	96	80	5A	0	43	EF	43	F1	F	F	3A	9D	22	20	22
	EOL	L8096		*90		C	Tk	C	Tk	Tk	15	:	Tk	"	sp	"
8090	3A	9D	22	20	22	0	9C	80	64	0	83	0	A2	80	6E	0
	:	Tk	"	sp	"	EOL	L809C		*100		Tk	EOL	L80A2		*110	
80A0	81	0	0	0	8	41	0	C5	32	76	80	0	0	0	0	8
	Tk	EOL	[end of program]			A										
80B0	42	0	C5	32	51	20	0	0	0	0	8	43	0	C5	32	59
	B											C				
80C0	60	0	0	0	0	8	44	0	41	90	0	0	0	0	0	0
							D									
80D0	3A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
80E0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
80F0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

* : line number *nn*
L : Link address *llhh*
Tk : Token byte
sp : space
EOL : end of BASIC line
[box] : end of program

Table 2 This table is a hexadecimal dump of the DUMP program as it resides in MSX computer RAM memory. All bytes have been analysed, and it may be useful to reconstruct program Listing 1 from it!

Table 3 These data are burned into an EPROM to function as a utility cartridge called DUMP. Compare the shaded addresses with those in Table 2 to note the move up by 1Øhex and the correspondingly adapted LSBs.

Table 3.

	Ø	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8ØØØ	41	42	Ø	Ø	Ø	Ø	Ø	Ø	1Ø	8Ø	Ø	Ø	Ø	Ø	Ø	Ø
8Ø1Ø	Ø	17	8Ø	A	Ø	9F	Ø	26	8Ø	14	Ø	85	22	73	74	61
8Ø2Ø	72	74	22	3B	41	Ø	33	8Ø	1E	Ø	85	22	65	6E	64	22
8Ø3Ø	3B	42	Ø	43	8Ø	28	Ø	82	2Ø	43	2Ø	EF	2Ø	41	2Ø	D9
8Ø4Ø	2Ø	42	Ø	5E	8Ø	32	Ø	9D	E4	22	5C	2Ø	2Ø	5C	22	3B
8Ø5Ø	FF	9B	28	43	29	3B	3A	9D	22	2Ø	2Ø	22	3B	Ø	6D	8Ø
8Ø6Ø	3C	Ø	82	2Ø	44	EF	11	2Ø	D9	2Ø	F	F	Ø	8B	8Ø	46
8Ø7Ø	Ø	9D	E4	22	5C	5C	22	3B	FF	9B	28	FF	97	28	43	F1
8Ø8Ø	44	29	29	3B	3A	9D	22	2Ø	22	3B	Ø	91	8Ø	5Ø	Ø	83
8Ø9Ø	Ø	A6	8Ø	5A	Ø	43	EF	43	F1	F	F	3A	9D	22	2Ø	22
8ØAØ	3A	9D	22	2Ø	22	Ø	AC	8Ø	64	Ø	83	Ø	B2	8Ø	6E	Ø
8ØBØ	81	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø

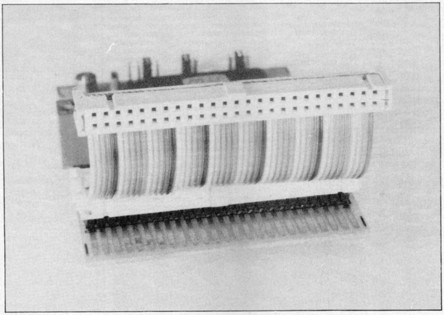


Fig. 5 The Spectravideo MSX computer may be connected to the cartridge extension board with a short length of 50-way ribbon cable and two suitable sockets.

is lengthened by some 160 bytes because of the addition of DUMP.
After this first aquaintance with the hexadecimal dumping format and use of DUMP in practice, the computer memory may be cleared (NEW) and DUMP entered as shown in Listing 1, i.e. from line 1Ø onwards. Run DUMP, enter &H8ØØØ as the start address and &H81ØØ as the end, and have a look at the machine code that constitutes this little program. With the use of Table 2, try to retrace the familiar BASIC lines to understand the MSX memory storage principle. Note that the link addresses and line numbers are in reverse order, that is with their LSBs first. All standard BASIC commands have a corresponding token-byte, and it will not be difficult to spot some of them:

82h=FOR; 9Dh=LPRINT; EFh="=" (equal sign); 83h=NEXT; F1h="+"; E4h=USING; etc.

If this is all sufficiently clear, we will now consider the EPROM data.

EPROM data

It will be evident that the computer does not consider the machine code currently present in locations 8ØØØ and up as located in a cartridge, because the identification group of bytes as already discussed is not present at the beginning of the program (8ØØØ. . .8ØØF). To obtain factual EPROM data, the whole machine code program will have to be moved up by sixteen (1Øh) bytes, the link addresses changed accordingly, and the identifiers placed at the beginning as outlined above.
A practical example of how this may be accomplished is shown in Table 3; this is the DUMP program again, but this time as present in an EPROM; compare the data with those of Table 2 to gain an insight into cartridge EPROM operation with MSX BASIC; program an EPROM with these data, plug it into the cartridge ZIF socket, and run your own utility cartridge.
Finally, a word about lengthier, more complicated BASIC programs and their storage in EPROM. As already suggested, the DUMP program may be attached to them at a suitable high line number, e.g. 1ØØØØ. With the main program fully debugged and operational, run DUMP, spot the link addresses, add 1Øhex to them, move the program up by 1Øhex addresses, and write a suitable sequence of identification bytes. The link addresses always point to the next one, and are thus easily picked out for modification. Program end is marked by a link address reading ØØØØ, but the **real** end, that is **without** the added DUMP program, may be found by looking for the hexadecimal equivalent of 1ØØØØ, bytes 1027 in that order; next, change the preceding link address into ØØØØ. Finally, note that programs run from cartridge may, of course, not be edited because they reside in read-only memory.

Spectravideo connection

The extension board need not always be inserted into the computer's cartridge slot; the Spectravideo MSX computer, for instance, features a 'real' 50-way expansion connector for receiving an appropriate flat ribbon type socket. The present extension board is then connected with a short length of 50-way flat ribbon cable with such a socket on either end of it, as shown in Fig. 5. Note that there is a slight oddity with the Spectravideo output expansion connector; the tiny arrow on it does not indicate pin 1 as usual practice, but pin 50. However, no problems should be encountered if the example given by Fig. 5 is followed.
This finishes the present article on MSX extensions; a further instalment will deal with the construction of a bus-board for this type of computer.

GD:BL