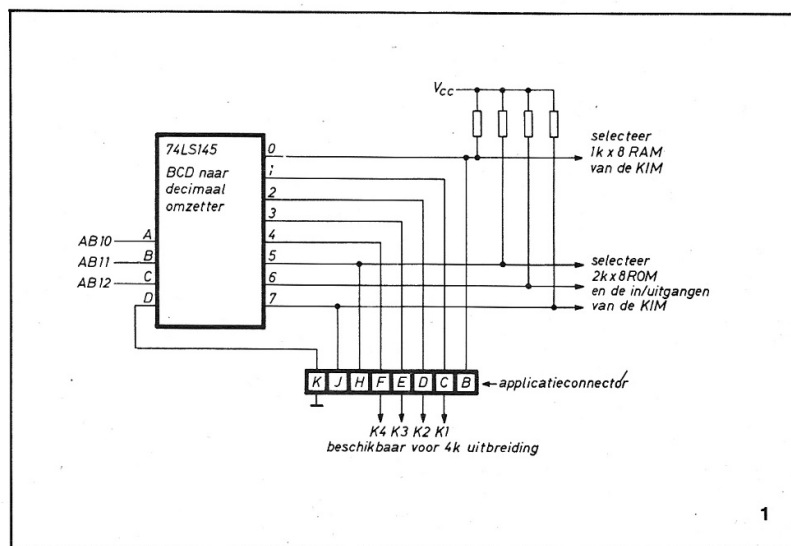




Memory Expansion for the KIM 1

D.M. de Boer

Although the 1K memory space of the KIM 1 is very generous for a beginning programmer, sooner or later the need for more memory will arise. This allows larger programs to be written, or more programs to be held in memory at the same time. There are several ways to expand the memory. You can buy a number of ICs yourself and mount them on a home-made circuit board, which is by far the cheapest solution. The nicest solution is simply to buy a ready-made memory card; you then have a professional piece of work, with the reliability that goes with it. The consequence, however, is the higher price. An attractive in-between solution is to offer such a professional memory card as a kit. This allows the selling price to be considerably lower than that of the fully assembled card, while you still have the reliability of a professional system. The manufacturer of the well-known BEM memory cards has now decided to release the BEM-1 as a kit. For RB this was reason to devote an article to this memory card in kit form.



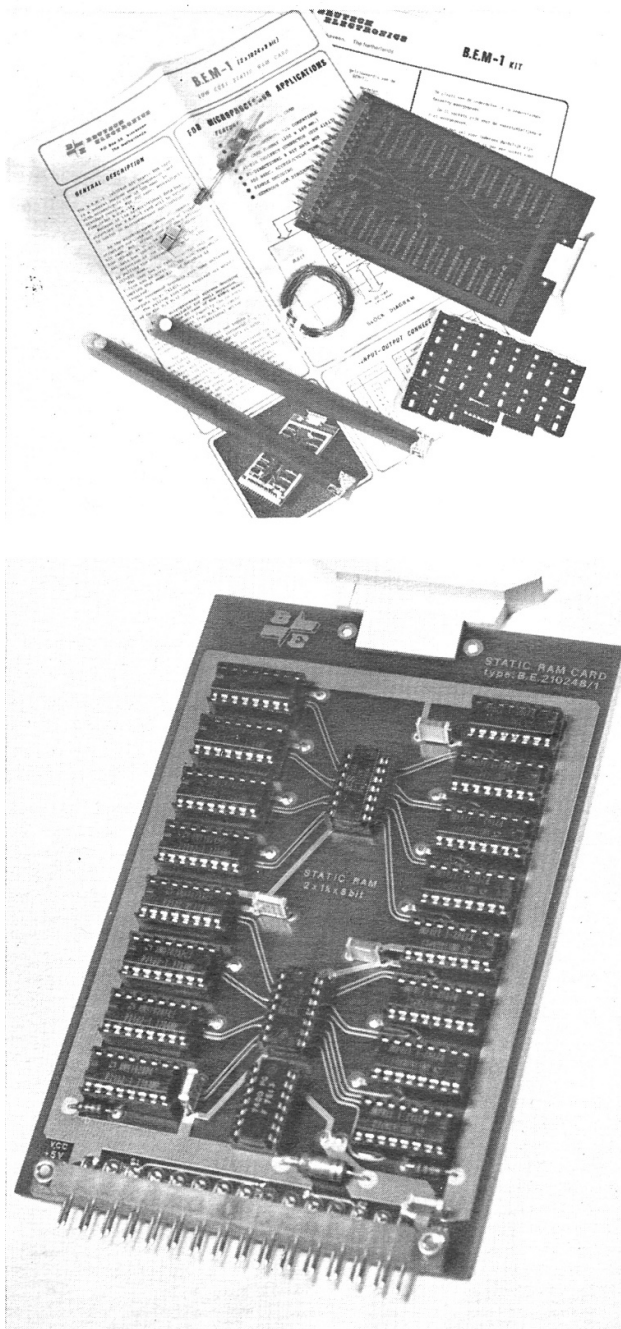
1. The KIM-1 already contains the decoding for a 4K memory expansion.

The organisation of the BEM-1 card

The BEM-1 is a 2K static RAM card. It is divided into 2 sections of 1K and has 10 address connections (A0 . . . A9), enough to decode 1024 locations. Together with the memory already present in the KIM, the total available RAM is thus divided into blocks of 1K. The

KIM-1 board carries a BCD-to-decimal decoder (Fig. 1). This decoder decodes addresses A10 . . . A12 and in this way determines which 1K block is being addressed. Depending on the code presented on address lines A10 . . . A12, one of the outputs of this IC goes low. For example, for addresses 0000 . . . 03FF output K0 goes low (A10 . . . A12 are '0'). K0 selects the 1K RAM of the KIM-1. So we must never use this output to select another 1K block. Outputs K1 through K4 are free to use; K5 . . . K7 are used to select the 2K RAM timers and the I/O of the KIM. With outputs K1 through K4 we can therefore add a total of 4K RAM, corresponding to 2 BEM-1 cards. If we want to expand the memory further, separate circuits must be added to decode addresses A13 . . . A15;

in addition, buffers must then be placed in the address lines so as not to load the microprocessor too heavily. We will return to this in due course. For the time being we limit ourselves to a maximum expansion of 4K.



The kit

The BEM-1 kit (Fig. 2) contains all the parts needed to assemble the 2K memory card. The Eurocard-size board is a double-sided epoxy board with plated-through holes. In addition, a socket is supplied for every IC, so that any defective ICs can easily be replaced. As instructed in the assembly manual, we checked the board for any imperfections before soldering. The board looked well made, however, and we found no faults. For soldering, a 15 to 20 Watt iron with a small tip is recommended. Irons with a higher power rating (e.g. 40 Watt) have far too much capacity, which can damage the board!! If you don't have a small soldering iron yet, it is certainly worth buying one. You will badly need it in the future as well. Next, the connector, the IC

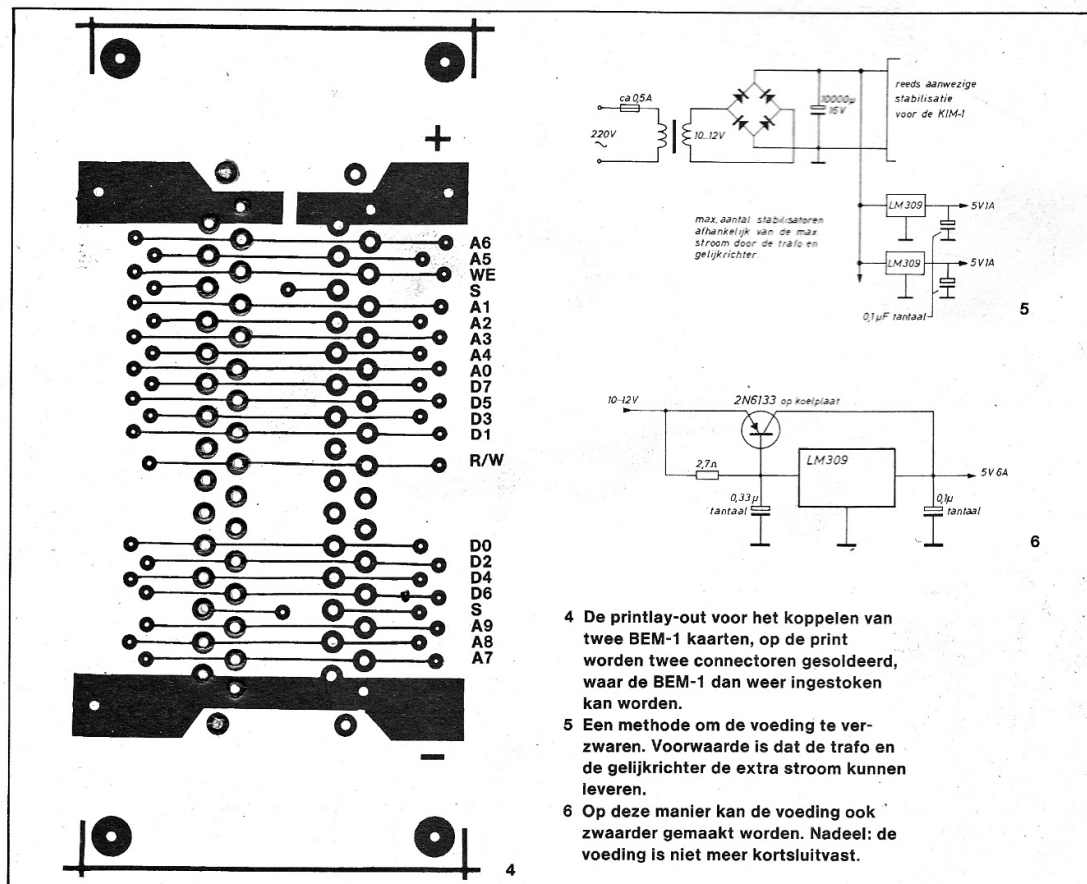
sockets and the passive components are soldered onto the board, in that order. The assembly manual gives some useful tips for this. Thanks to the plated-through holes, the board only needs to be soldered on one side, which with 345 solder joints saves a considerable amount of time. The supplied solder was more than enough to solder the whole board.

The final step is inserting the ICs. Here we must take care that the ICs are placed the right way round. With the RAMs we must also remember that they are sensitive to static charges.

The connections

The BEM-1 kit is now fully assembled and ready to be connected to the KIM-1. On the BEM-1 board, each pin is clearly marked with the signal it carries. Table 1 lists these connections once more. Signals A0 . . . A9 are the ten address lines, signals D0 . . . D7 are the eight data (information) lines. In addition we have ground (Vss) and +5V (Vcc). By making S1 or S2 low we can enable one of the two 1K sections. If, for example, we connect S1 of the BEM to K3 of the KIM, the 1K RAM section will be located at addresses 0C00 . . . 0FFF. Table 2 shows which addresses the 1K RAM sections get when we connect one of the S lines of the BEM to K1, K2, K3 or K4 of the KIM. It will usually be most convenient to connect S1 to K1 and S2 to K2.

Together with the RAM already on the KIM, we then have a continuous address range from 0000 to 0BFF. A second BEM-1, if any, can then be connected to K3 and K4, in which case the address range runs from 0000 . . . 13FF. Table 3 shows once more which points of the KIM-1 must be connected to the BEM-1.



4. The board layout for linking two BEM-1 cards; two connectors are soldered onto the board, into which the BEM-1 cards can then be plugged.

5. A method of making the power supply heavier. The condition is that the transformer and the rectifier can supply the extra current.

6. The power supply can also be made heavier this way. Disadvantage: the power supply is no longer short circuit protected.

Mini-bus for 2 BEM-1 cards

When 2 BEM cards are connected to the KIM, the board from Fig. 4 can be put to good use. 2 connectors are soldered onto the board. In addition, 4 card guides must be screwed onto the board. The connections have remained the same (Table 3). In this way the 4K memory becomes a complete unit.

The power supply

According to the specifications, the maximum current drawn by a BEM card is 890 mA. The power supply you built or bought for the KIM-1 is probably just not heavy enough to supply the extra current. If the transformer and the rectifier can supply this current, a voltage regulator can simply be added for each expansion (Fig. 5). The maximum current a regulator IC can supply can also be increased by adding an extra transistor (Fig. 6).

This last method has the disadvantage that the regulator's internal overload protection no longer works. If the transformer you use is not heavy enough, you had best take a second transformer and build an entirely separate power supply. In that case the ground connections of the different supplies must be connected together.

Testing

Once all connections have been made and the power supply has been connected (mind the correct polarity), we can start testing the board.

We must check whether all the new memory locations can indeed be written to, and whether the numbers written are indeed retained. A little job the KIM can do excellently. A test program is shown in Listing 1. The start address of the memory area to be tested goes into addresses 00 and 01, the end address into 02 and 03. If we have connected the S lines of the BEM-1 to K1 and K2 of the KIM, the memory to be tested lies between addresses 0400 and 0BFF. We enter this as follows:

```
00 00 start address
04 01
FF 02 stop address
0B 03
```

We can start the program at address 0100. When an error is detected, the address of the error appears on the display. Pressing one of the keys (not RS or ST) makes the test program continue. When the program has finished (this takes quite a while), address 0000 appears on the display. If an error is detected, you should check the board carefully for short circuits.

By swapping the RAMs you can determine whether a RAM is faulty (the error then gets a different address). If you cannot find the fault, the supplier of the kit will be glad to carry out the repair for you. If the fault was not caused by, for example, bad soldering, wrong voltages or other incorrect handling, no repair costs will be charged.

lijsten:

- 1a Hoofdprogramma voor het testen van de geheugenuitbreiding.
1b De subroutines en geheugenplaatsen welke het hoofdprogramma gebruikt.

tabellen:

- 1 Aansluitingen van de BEM-1.
2 Adresruimte voor de geheugenuitbreiding.
3 Verbindingen tussen de KIM-1 en de BEM-1.

tabel 1

Pin	Functie	Pin	Functie	Pin	Functie
1	VSS	11	N.C.	21	A 0
2	VSS gnd.	12	N.C.	22	A 4
3	A 7	13	N.C.	23	A 3
4	A 8	14	N.C.	24	A 2
5	A 9	15	R/W	25	A 1
6	S 2	16	N.C.	26	S 1
7	D 6	17	D 1	27	WE
8	D 4	18	D 3	28	A 5
9	D 2	19	D 5	29	A 6
10	D 0	20	D 7	30	VCC + 5V
				31	VCC

tabel 2

S1 of S2 aansluiten op	adresruimte waarin de 1k-sectie komt
K1	0400 ... 07FF
K2	0800 ... 0BFF
K3	0C00 ... 0FFF
K4	1000 ... 13FF

KIM-1 expansionconnector		BEM-1
naam	pen	pen
AB7	J	3
AB8	K	4
AB9	L	5
DB6	9	7
DB4	11	8
DB2	13	9
DB0	15	10
R/W	V	15
DB1	14	17
DB3	12	18
DB5	10	19
DB7	8	20
AB0	A	21
AB4	E	22
AB3	D	23
AB2	C	24
AB1	B	25
RAM/R/W	Z	27
AB5	F	28
AB6	H	29
applicatieconnector		
naam	pen	
K1 ... K4	C, D, E, F	6,26

tabel 3

0100	20 74 01	start	JSR-abs	floop	wacht tot toets is losgelaten
0103	20 56 01		JSR-abs	init	geheugens in de begintoeestand
0106	A0 00		LDY-imm	\$00	te testen
0108	A9 00	loop1	LDA-imm	\$00	geheugen op \$00
010A	91 04		STA-ind,y	ptrlo	verhoog testadres
010C	20 4F 01		JSR-abs	incptr	was dit het laatste testadres?
010F	A5 04		LDA-Z page	ptrlo	zo nee, terug naar 'loop 1'
0111	C5 02		CMP-Z page	toplo	reset testadres
0113	D0 F3		BNE-rel	loop1	onhoudd 'mask' verhoog 'mask' oude waarde
0115	A5 05		LDA-Z page	ptrhi	'mask' in A
0117	C5 03		CMP-Z page	tophi	controleer of de oude waarde nog op het testadres staat
0119	D0 ED		BNE-rel	loop1	zo nee, naar 'fout' nieuwe waarde 'mask' in A
011B	20 5A 01	loop3	JSR-abs	reset	zet A op het testadres
011E	A6 06		LDX-Z page	mask	controleer of de nieuwe waarde goed geschreven is
0120	E6 06		INC-Z page	mask	zo nee, naar 'fout' verhoog testadres
0122	8A	loop2	TXA-impl	mask	was dit het laatste testadres?
0123	D1 04		CMP-ind,y	ptrlo	zo nee, terug naar 'loop 2'
0125	F0 03		BEQ-rel	goed1	laatste waarde van 'mask'?
0127	20 63 01		JSR-abs	fout	zo nee, terug naar 'loop 3'
012A	A5 06	goed1	LDA-Z page	mask	voer adres \$0000 in naar monitor-programma
012C	91 04		STA-ind,y	ptrlo	
012E	D1 04		CMP-ind,y	ptrlo	
0130	F0 03		BEQ-rel	goed2	
0132	20 63 01		JSR-abs	fout	
0135	20 4F 01	goed2	JSR-abs	incptr	
0138	A5 04		LDA-Z page	ptrlo	
013A	C5 02		CMP-Z page	toplo	
013C	D0 E4		BNE-rel	loop2	
013E	A5 05		LDA-Z page	ptrhi	
0140	C5 03		CMP-Z page	tophi	
0142	D0 DF		BNE-rel	loop2	
0144	A5 06		LDA-Z page	mask	
0146	D0 D3		BNE-rel	loop3	
0148	85 FA		STA-Z page	pointl	
014A	85 FB		STA-Z page	pointh	
014C	4C 22 1C		JMP-abs	resvec	

1a

014F	E6 04	incptr	INC-Z page	ptrlo	hoog ptrlo op moet ptrhi ook opgehoogd worden
0151	D0 02		BNE-rel	end	hoog ptrhi op terug naar hoofdprogramma
0153	E6 05		INC-Z page	ptrhi	
0155	60	end	RTS-impl		
0156	A9 00	init	LDA-imm	\$00	mask 'mask' leeg
0158	85 06		STA-Z page	mask	
015A	A5 00	reset	LDA-Z page	botlo	kopieer het beginadres, en zet dit op ptrlo en ptrhi
015C	85 04		STA-Z page	ptrlo	terug naar hoofdprogramma
015E	A5 01		LDA-Z page	bothi	
0160	85 05		STA-Z page	ptrhi	
0162	60		RTS-impl		
0163	A5 05	fout	LDA-Z page	ptrhi	zet het adres van de defecte geheugenplaats in de geheugens voor het display
0165	85 FB		STA-Z page	pointh	
0167	A5 04		LDA-Z page	ptrlo	
0169	85 FA		STA-Z page	pointl	
016B	A5 06		LDA-Z page	mask	
016D	85 F9		STA-Z page	\$F9	
016F	20 1F 1F	Loop4	JSR-abs	scands	naar display-subroutine
0172	F0 FB		BEQ-rel	loop4	toets ingedrukt?
0174	20 FE 1E	floop	JSR-abs	AK	zo ja, toets losgelaten?
0177	D0 FB		BNE-rel	floop	zo nee, terug naar floop
0179	A6 06		LDX-Z page	mask	oorspronkelijke X terugzetten
017B	CA		DEX-impl		'Y' weer '00' maken
017C	AD 00		LDY-imm	\$00	terug naar hoofdprogramma
017E	60		RTS-impl		
		botlo	\$0000	beginadres te testen geheugendeel	
		bothi	\$0001		
		toplo	\$0002		
		tophi	\$0003		
		ptrlo	\$0004	door het programma gebruikte geheugenplaatsen	
		ptrhi	\$0005		
		mask	\$0006		
		pointh	\$00FB		
		pointl	\$00FA		
		scands	\$1F1F		
		AK	\$1EFE		
		resvec	\$1C22		

1b