IMPROVED MONITOR FOR NASCOM 'B BUG'

Features:

1) Full software compatibility with original monitor.

2) WRITE and READ commands for I/O to cassette four times faster than DUMP and LOAD. (Extensive error checking.)

3) INTELLIGENT COPY command. Same as COPY but intelligent.

4) ARITHMETIC command. Adds and subtracts addresses. Calculates offsets.

5) HEX and NORMAL commands. Allows entry of text into the MODIFY command without having to look up ASCII tables.

6) Extended keyboard. All displayable characters available from the standard keyboard.

7) Extended Register Display. In addition to existing display, registers I, IX and IY are displayed. The flags are shown by letters, such as Z, C, etc.

8) If an NMI button is added registers displayed automatically.

9) Operation of shift key corrected.

10) Useful subroutines for: table searching, random numbers, interruptable delay, ASCII to packed BCD and vice versa, cursor movement, output of string of characters.

11) GENERATE command to write a tape which automatically loads and executes a program.

The B Bug monitor requires two 2708 EPROMS.

There is still space for future enhancement to the second EPROM, and a remarkable feature is that the first EPROM will work alone while the second EPROM is away, provided that a standard original monitor is placed in the second EPROM socket.

COMPLETE LIST OF COMMANDS

*	A	Arithmetic
	B	Breakpoint
	C	Copy
	D	Dump
	E	Execute
*	G	Generate
*	H	Hex Keyboard
*	I	Intelligent Copy
	L	Load
	M	Modify
*	N	Normal Keyboard
*	R	Read
	S	Step
	T	Tabulate
*	W	Write

* additional command

ADDITIONAL COMMANDS

A aaaa bbbb

The Arithmetic command performs simple hexadecimal arithmetic.

Three results are displayed, as follows:

SSSS DDDD JJ

SSSS is the sum of the two values.

DDDD is the difference of the two values, bbbb-aaaa.

JJ is the displacement required in a Jump Relative instruction, which starts at aaaa, to cause a jump to bbbb. If such a jump is not possible, ?? is displayed.

G aaaa bbbb eeee

The Generate command writes a tape, which, when read back in, loads a program and automatically executes it.

Data from address aaaa, up to but not including address bbbb, is written to the tape, in the same format as for the Write command.

eeee is the address at which execution is to start.

The data on the tape is as follows:

Newline

B0 Newline

E0 Newline

R Newline

Data, in the format used by the Write and Read commands

E eeee Newline

Note: Start the tape mechanism before entering the G command. The LED is only on while the data specified is being output. When reading the tape in no commands should be entered. Simply start the tape and stop it when the program has started execution.

The Hex command modifies the keyboard so that it is possible to obtain the hexadecimal equivalent of any key, instead of the value itself. To do this, hold down the space bar and press the key required. The hexadecimal value will be displayed, followed by a blank. When the space bar is not held down, the keyboard operates normally. The only hexadecimal equivalent not available is that of the space bar itself, 20H.

The command is most useful when using the Modify command to enter strings of text.

The keyboard operation is returned to normal either by pressing Reset or by the Normal (N) command.

aaaa bbbb dddd

The Intelligent Copy command is identified to the Copy command, except that data in overlapping regions is never destroyed. See the description of the Copy command.

Copying is from address aaaa to address bbbb for dddd bytes.

The Normal Keyboard command returns the keyboard to normal, after use of the Hex command.

The Read command reads data which was output by the Write command. (This is four times faster than Load.)

See the Write command for the format of the data.

As each block is read, the header data is displayed:

SSSS BBLL - start address, block number, length (0=256).

After block 0 has been read, the Read command ends. During the execution of the command the Drive LED is switched on.

...R...

...R...

The start of each block is recognised by reading the four start of block characters. All data is ignored until the start of a block. If the checksum for the header data does not agree with the data, then the message ERR is displayed, and the program waits for the start of the next block. The data following is not loaded.

If the checksum for the data does not agree with the computed total, then the message ERR is displayed, and the program waits for the start of the next block. In this case, invalid data will have been loaded, but only at the correct addresses.

If any errors are encountered, rewind the tape for about two blocks and carry on.

Do not press keys on the keyboard since this will cause errors (which will be detected).

The visual check of the display is required to ensure that all blocks have been loaded correctly.

W aaaa bbbb

The Write command outputs data four times faster than Dump. Data from address aaaa, up to but not including address bbbb is sent to the serial output.

Data is output in blocks, each containing up to 256 bytes of data (only the last block may have less).

The format of each block is as follows:

FF FF FF FF	4 start of block characters, 'FF' H.
SS SS	Start address, low order first.
LL	Length of data (0=256).
BB	Block number, this is one less for each block. The last block is block 0.
CC	Checksum for the start address, data length and block number.
DD DD	Data.
EE	Checksum for data.
00 00 00	3 characters, value '00' H.

...W...

DITIONAL COMMANDS

.W...

As each block is written, the start address, block number and length are displayed as follows:

SSSS BBLL

After the command is entered, the Drive LED switches on, there is a two second delay, and then the data is output.

At the end the LED goes out and the next command may be entered.

The three additional characters output at the end of each block ensure that even if up to three characters are not read in this block, the next block can still be read correctly.

OTHER FEATURES

NMI

If the necessary hardware is installed to add a button which can be pressed during program execution to cause an NMI, then the monitor will act exactly as if the program was being single stepped and will display the registers. It is then possible to continue execution or enter any other monitor command.

Subroutines

The monitor includes many useful routines. These are described in the monitor listings.

Shift Key

The action of the shift key has been corrected. Previously, if it was held down and many shifted keys, such as 1, 2, 3, were pressed very rapidly, not all the resulting characters were shifted.

Future Enhancement

There is still room in the second EPROM for future enhancement. The EPROM could be sent away for reprogramming to include the revised instructions. However, the computer can still be used during this time, simply by plugging the original monitor into the second EPROM socket. This provides the facilities of the original monitor, plus the NMI feature described above.

ADDRESSES IN SECOND EPROM

0400	WRITE	(uses TXI, SOUT)	0697	ARGS	
0453	GDS		06A3	G	(uses SOUT, WRITE)
045C	TXI		06CC	SOUT	
0466	TABLE		06D9	ERI	
047A	RND		070C	READ	(uses TXI)
*048A	EKEY		*0756	EREG	(uses ERI)
04B9	IDELAY		*0760	EPARSE	(uses TABLE)
04C6	CDA		076F	ECTAB	
04F2	RDL		0787	future enhancement	
04FC	CAD	(uses RDL)	079A	H	
0514	ICOPY	(uses ARGS)	07A1	N	
0527	ARITH	(uses ARGS)	07A6	KEX	
0557	future enhancement		07C0/07CF/07DB/07ED	MCR/MCL/MCD/MCU	

* this address is referred to in the first EPROM

ITIONAL FEATURES

ended Keyboard

displayable characters are available from the standard keyboard. diagram shows the layout of this extended keyboard.

keyboard operates exactly as normal, except that the '@' key is abled. To enter '@' hold down the shift key and press '@'.

'@' key now operates as a 'Supershift' key. When held down, ery other key, whether shifted or normal, is associated with a character. For example, if a letter key is pressed while pershift is held down it appears in lower case.

addition, keys U, V, W, X, Y and Z now provide the missing CII characters when shifted.

ample: Hold down Supershift, then hold down Shift, then press the '7' key. The result is ♀

e control keys (New Line, Back Space, Clear Screen) are not fected. In fact, characters with values 00-1F have bit 7 set, that they are 80-9F.

te: See the description of the H and N commands. These make e keyboard able to supply the hexadecimal equivalent of any aracter.

tended Register Display

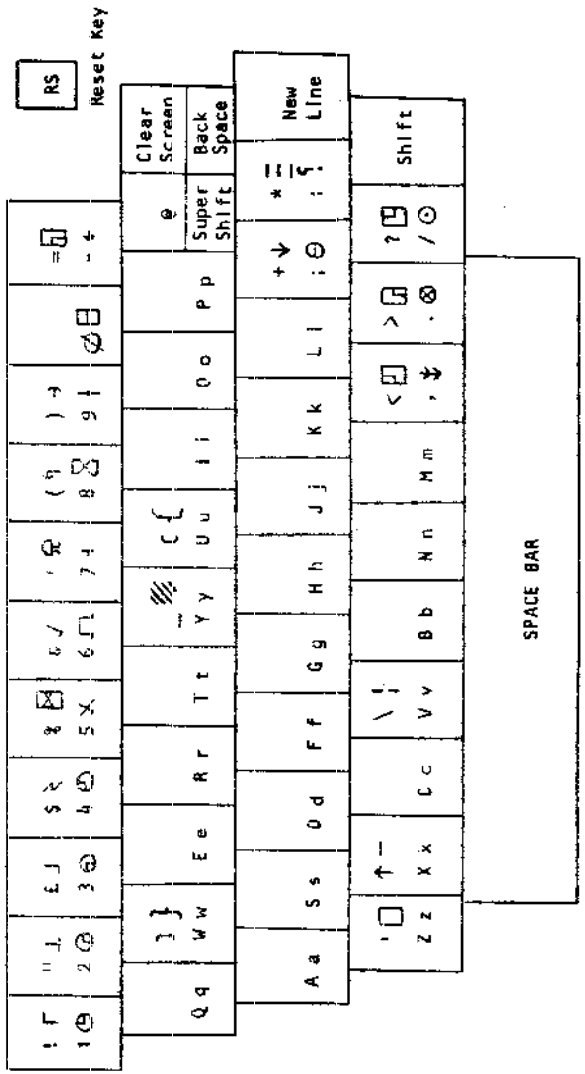
en the Step command is used, a breakpoint is encountered, or the de E7 (RST 20H) is executed in a program, the program registers e displayed.

is display has been extended and the following information is now own:

SP PC AF HL DE BC I IX IY Flags

e Flags are a decoded representation of register F. The llowing characters may be displayed, indicating which flag bits ave been set.

S Z H P N C



NASCOM 1 KEYBOARD LAYOUT

(showing SHIFT and 'SUPERSHIFT' character allocations)

PROGRAM LISTING Sheet 1 of 2

GRAM: MONITOR
TINE: WRITE

:.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
00	CD	51	00		WRITE	CALL	MOTFLP		) Switch on LED
03	06	00				LD	B,	0	) and wait
05	CD	35	00		W2	CALL	KDEL		) for two
08	10	FB				DJNZ	W2-$		) seconds.
0A	2A	0C	0C			LD	HL,	(ARG1)	) Get parameters
0D	ED	5B	0E	0C	W4	LD	DE,	(ARG2)	) calculate
11	EB					EX	DE	HL	) length -1.
									)
12	37					SCF			) If this is negative,
13	ED	52				SBC	HL,	DE	) return, switching off
15	DA	51	00			JP	C,	MOTFLP	) LED. HL is now the
18	EB					EX	DE,	HL	) start address. DE is length -1.
19	06	04				LD	B,	4	)
1B	3E	FF			W5	LD	A,	0FFH	) Output start of
1D	CD	5D	00			CALL	SRLOUT		) block characters
20	10	F9				DJNZ	W5-$		)
22	AF					XOR	A		) If block 0, then
23	BA					CP	D		) set length to
24	20	02				JR	NZ,	W6-$	) E+1 instead of
26	43					LD	B,	E	) to 0(256).
27	04					INC	B		)
28	58				W6	LD	E,	B	) Set E to length of data.

e description of W command.

PROGRAM LISTING Sheet 2 of 2

PROGRAM: MONITOR
ROUTINE: WRITE

P.C.		MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
04	29	7D					LD	A,	L	)
	2A	CD	5D	00			CALL	SRLOUT		) Output start
	2D	7C					LD	A,	H	) address.
	2E	CD	5D	00			CALL	SRLOUT		)
	31	7B					LD	A,	E	) Output length
	32	CD	5D	00			CALL	SRLOUT		) of block.
	35	7A					LD	A,	D	) Output block
	36	CD	5D	00			CALL	SRLOUT		) number.
	39	0E	00				LD	C,	0	) Display start address, block
	3B	CD	5C	04			CALL	TX1		) number, and length, and
	3E	79					LD	A,	C	) output checksum for header.
	3F	CD	5D	00			CALL	SRLOUT		)
	42	CD	40	02			CALL	CRLF		)
	45	CD	CC	06			CALL	SOUT		Output the data.
	48	06	04				LD	B,	4	) Output checksum followed by
	4A	79					LD	A,	C	) three '00's, to allow for
	4B	CD	5D	00		W9	CALL	SRLOUT		) dropouts in this block.
	4E	AF					XOR	A		)
	4F	10	FA				DJNZ	W9-$		)
	51	18	8A				JR	W4-$		Next block.

See description of W command.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: GDS

P.C.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
53	1F				GDS				N L
54	42	30	1F						S N L
57	45	30	1F						E N L
5A	52	1F							R N L

e description for routine G.
ing of characters output by routine G.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: TX1

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
04	5C	CD	5F	04		TX1	CALL	TX2		) Display
										) HL and DE.
	5F	CD	32	02		TX2	CALL	TBCD3		) Also
										) accumulate
	62	EB					EX	DE,	HL	) into
	63	C9					RET			) checksum.
	64	00	00				NOP	NOP		Spare.

The routine is used by the Read and Write routines.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: TABLE

P.C.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
66	D5				TABLE	PUSH	DE		Save DE.
67	5F					LD	E,	A	Put value into E
68	7E				TB1	LD	A,	(HL)	Get table value.
69	23					INC	HL		) Increment position.
6A	B7					OR	A		) Jump out if end
6B	28	07				JR	Z,	TB3-$	) of table.
6D	BB					CP	E		) Jump out if
6E	28	04				JR	Z,	TB3-$	) match.
70	23					INC	HL		) Increment over address and
71	23					INC	HL		) repeat for next value.
72	18	F4				JR	TB1-$		)
74	5E				TB3	LD	E,	(HL)	)
75	23					INC	HL		) Put address
76	56					LD	D,	(HL)	) into HL.
77	EB					EX	DE,	HL	)
78	D1					POP	DE		Restore DE.
79	C9					RET			

e routine performs a table search and returns an address in HL. HL must be the start the table. A must be the value to be compared to the table. Each line of the table st consist of 3 bytes - the comparison value and the address. The last line of the

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: RND

P.C.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
04	7A	C5			RND	PUSH	BC		Save BC.
	7B	47				LD	B,	A	Set B to maximum.
	7C	ED	5F			LD	A,	R	)
	7E	86				ADD	A,	(HL)	) Generate
	7F	38	01			JR	C,	R2-$	) and store
	81	3D				DEC	A		) random byte.
	82	77			R2	LD	(HL)	A	)
	83	90			SUB	SUB	B		) Obtain
	84	30	FD			JR	NC,	SUB-$	) result by
	86	80				ADD	A,	B	) repeated
	87	3C				INC	A		) subtraction.
	88	C1				POP	BC		Restore BC.
	89	C9				RET			

The routine generates a pseudo-random number, between 1 and A. HL must be the address of a one byte work area. A is the maximum value and must be in the range 1-255 (01-FF). The result is returned in A. No other registers are modified. For best results the

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: EXTENDED KEYBOARD

C.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
8A	30	0D			EKEY	JR	NC,	KE-$	) Normal return if) no key pressed.
8C	21	09	0C			LD	HL,	0C09H	
8F	FE	40				CP	'@'		) Jump to K3 unless
91	20	0A				JR	NZ,	K3-$	) '@' entered.
93	B7					OR	A		Reset carry.
94	CB	66				BIT	4,	(HL)	1=shift down.
96	28	01				JR	Z,	KE-$	) 0, so '@') disabled.
98	37				KN	SCF			Set carry.
99	E1				KE	POP	HL		) Restore
9A	D1					POP	DE		) registers
9B	C1					POP	BC		) and
9C	C9					RET			) return.
9D	FE	21			K3	CP	21H		) If blank or less
9F	38	F7				JR	C,	KN-$	) than don't modify
A1	FE	55				CP	'U'		) If 'U' or
A3	38	06				JR	C,	K5-$	) more and
A5	CB	66				BIT	4,	(HL)	) shift is
A7	28	02				JR	Z,	K5-$	) down,
A9	C6	06				ADD	A,	6	) add 6.
AB	CB	6E			K5	BIT	5,	(HL)	) 1='@'
AD	28	E9				JR	Z,	KN-$	) held down.
AF	C6	20				ADD	20H		)
B1	FE	60				CP	60H		) '@' held down
B3	30	E3				JR	NC,	KN-$	) so modify
B5	C6	40				ADD	40H		) all keys
B7	18	DF				JR	KN-$		)

s extension to the first EPROM extends the keyboard routine. The routine provides displayable characters, using the standard keyboard. Characters 00-1F have in fact bit 7 set, to distinguish them from the control characters.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: INTERRUPTABLE DELAY

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
04	B9	1B				IDELAY	DEC	DE		)
	BA	7A					LD	A,	D	) Loop until
	BB	B3					OR	E		) DE is 0.
	BC	C8					RET	Z		)
	BD	CD	4D	0C			CALL	$KBD		
	C0	D8					RET	C		) Return if) keyboard entry.
	C1	CD	35	00			CALL	KDEL		Delay 7.5 ms.
	C4	18	F3				JR	IDELAY-$		

DE	APPROX. DELAY
00 01	0
00 02	7.5ms.
00 20	1/4s.
01 00	2s.
02 00	4s.
04 00	9s.
08 00	18s.
10 00	36s.
20 00	1min. 11s.
40 00	2min. 22s.
80 00	4min. 44s.
00 00	9min. 28s.

The routine provides a variable delay which depends on the value of DE. If a key is pressed on the keyboard, then $KBD returns with carry set, and the routine is terminated at once. Register DE must be set before the routine is called. Registers AF and DE are modified by the routine. If the routine returns with carry set, then it

PROGRAM LISTING Sheet 1 of 1

RAM: MONITOR
INE: CDA - CONVERT BCD TO ASCII

.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
C6	7E				CDA	LD	A,	(HL)	) Save (HL).
C7	F5					PUSH	AF		)
C8	CD	DE	04			CALL	CD14		First half.
CB	CD	DE	04			CALL	CD14		Second half.
CE	F1					POP	AF		) Restore (HL).
CF	77					LD	(HL),	A	)
D0	23					INC	HL		) Increment BCD) position.
D1	10	F3				DJNZ	CDA-$		Next BCD byte.
D3	CB	41				BIT	0,	C	) Return normally) if value was
D5	C0					RET	NZ		) output or if) single zero
D6	CB	49				BIT	1,	C	) suppression) requested.
D8	C8					RET	Z		)
D9	1B					DEC	DE		) Put in the
DA	3E	30				LD	A,	30H	) single zero if
DC	18	11				JR	CD18-$		) value was zero.
DE	AF				CD14	XOR	A		) Get next 4 bits.
DF	ED	6F				RLD			)
E1	20	08				JR	NZ,	CD16-$	) If value is 0
E3	CB	41				BIT	0,	C	) and no value yet
E5	20	04				JR	NZ,	CD16-$	) output, then
E7	3E	20				LD	A,	20H	) then output
E9	18	04				JR	CD18-$		) a blank.
EB	CB	C1			CD16	SET	0,	C	) Set bit to show) no suppression.
ED	C6	30				ADD	A,	30H	) Convert to) ASCII.
EF	12				CD18	LD	(DE),	A	) Set ASCII value) and increment
F0	13					INC	DE		) output position.
F1	C9					RET			

routine converts a packed BCD value to ASCII. (Both stored with high order first.) must be the address of the BCD field. DE must be the address of the ASCII field. ust be the length of the BCD field, in bytes. (The ASCII value will be twice as g.) C is used to control the conversion:
0=0 Suppress leading zeroes. Bit 0=1 Do not suppress leading zeroes.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: RDL - ROTATE PACKED BCD LEFT

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
04	F2	C5				RDL	PUSH	BC		) Save registers.
	F3	E5					PUSH	HL		)
	F4	2B				DL2	DEC	HL		) Rotate the) packed BCD value
	F5	ED	6F				RLD			) to the left by) half a byte, and
	F7	10	FB				DJNZ	DL2-$		) add the decimal) value in A.
	F9	E1					POP	HL		) Restore
	FA	C1					POP	BC		) registers.
	FB	C9					RET			

The routine rotates a packed BCD value to the left by half a byte, and adds a decimal digit to the rightmost, low order position. HL must be the address to the right of the last byte of the packed BCD field. B must be the length of the packed BCD field, in bytes. A must be the decimal digit to be added. Only registers AF are modified by the

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: CAD - CONVERT ASCII TO BCD

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
4	FC	78				CAD	LD	A,	B	) Store length of) packed BCD field.
	FD	36	00			CA2	LD	(HL),	0	) Initialise
	FF	23					INC	HL		) packed BCD field
5	00	10	FB				DJNZ	CA2-$		) to 0.
	02	47					LD	B,	A	) Set B back to) BCD field length
	03	87					ADD	A,	A	) Set C to twice) BCD field length
	04	4F					LD	C,	A	) equals max ASCII) field length.
	05	1A				CA6	LD	A,	(DE)	) Get ASCII byte) and increment
	06	13					INC	DE		) position.
	07	06	30				SUB	30		) Convert from) ASCII to decimal
	09	D8					RET	C		) Return if) invalid,
	0A	FE	0A				CP	0AH		) ignoring the) rest of the
	0C	D0					RET	NC		) ASCII field.
	0D	CD	F2	04			CALL	RDL		) Update the) BCD value.
	10	0D					DEC	C		) Repeat until the) length of the
	11	20	F2				JR	NZ,	CA6-$	) ASCII field) processed is
	13	C9					RET			) twice length of) BCD field.

The routine converts a decimal value in ASCII to a packed BCD value. (Both stored with high order first.) Any non-numeric character delimits the ASCII number. HL must be the address of the packed BCD field. DE must be the address of the ASCII field. B must be the length of the BCD field, in bytes. The ASCII value is not modified. After

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: ICOPY

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
05	14	CD	97	06		ICOPY	CALL	ARGS		Get arguments.
	17	B7					OR	A		) Compare HL to DE) without
	18	ED	52				SBC	HL,	DE	) modifying them.
	1A	19					ADD	HL,	DE	)
	1B	D2	FA	03			JP	NC,	COP5	) If HL≥DE go to) standard copy.
	1E	0B					DEC	BC		)
	1F	EB					EX	DE,	HL	)
	20	09					ADD	HL,	BC	) Set to end
	21	EB					EX	DE,	HL	) instead of start,
	22	09					ADD	HL,	BC	)
	23	03					INC	BC		)
	24	ED	B8				LDDR			) Copy, starting) at end.
	26	C9					RET			

See description of the I command. The "Intelligent Copy" routine is identical to "Copy", except that copying to a higher address does not destroy overlapping data

PROGRAM LISTING Sheet 1 of 1

GRAM: MONITOR
TINE: ARITH

.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
27	CD	9B	06		ARITH	CALL	ARG2		) Get
2A	EB					EX	DE,	HL	) parameters
2B	E5					PUSH	HL		) and save them.
2C	19					ADD	HL,	DE	) Display sum.
2D	CD	32	02			CALL	TBCD3		)
30	E1					POP	HL		)
31	B7					OR	A		) Display
32	ED	52				SBC	HL,	DE	) difference.
34	CD	32	02			CALL	TBCD3		)
37	2B					DEC	HL		) Adjust
38	2B					DEC	HL		) for instruction length.
39	7C					LD	A,	H	) If H is FF.
3A	FE	FF				CP	0FFH		)
3C	20	0A				JR	NZ,	A2-$	) Then bit 7
3E	CB	7D				BIT	7,	L	) of L must
40	20	0D				JR	NZ,	AOK-$	) be set.
42	EF				ANG	RST	28H		)
43	3F	3F	1F	00		'??'	CRLF	NOP	) No good.
47	C9					RET			)
48	B7				A2	OR	A		) If H is not FF
49	20	F7				JR	NZ,	ANG-$	) or 00, then no good. If it is, 00.
4B	CB	7D				BIT	7,	L	) Then bit 7 of L
4D	20	F3				JR	NZ	ANG-$	) must be 0.
4F	7D				AOK	LD	A,	L	)
50	CD	44	02			CALL	B2HEX		) Display offset
53	CD	40	02			CALL	CRLF		) and return.
56	C9					RET			)

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: ARGS

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
06	97	ED	4B	10	0C	ARG	LD	BC	(ARG3)	)
	9B	ED	5B	0E	0C	ARG2	LD	DE,	(ARG2)	) Get arguments.
	9F	2A	0C	0C			LD	HL,	(ARG1)	)
	A2	C9					RET			)

The routine loads the registers with the arguments.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: G

C.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
A3	21	53	04		G	LD	HL,	GDS	) Output initial
A6	06	09				LD	B,	9	) string of
A8	CD	CC	06			CALL	SOUT		) characters.
AB	CD	00	04			CALL	WRITE		Output the data.
AE	3E	45				LD	A,	'E'	) Output an 'E'.
B0	CD	5D	00			CALL	SRLOUT		)
B3	2A	4B	0C			LD	HL,	($CRT+1)	) Save current
B6	E5					PUSH	HL		) address of CRT.
B7	21	5D	00			LD	HL,	SRLOUT	) Set address of) CRT routine to
BA	22	4B	0C			LD	($CRT+1)	HL	) that of SRLOUT.
BD	2A	10	0C			LD	HL,	(ARG3)	) Set HL to third) parameter.
C0	CD	32	02			CALL	TBCD3		) Output HL to the) tape.
C3	E1					POP	HL		) Set address of) CRT back to
C4	22	4B	0C			LD	($CRT+1)	HL	) original value.
C7	3E	1F				LD	A,	1FH	) Output a CRLF
									) and return.
C9	C3	5D	00			JP	SRLOUT		)

description of the G command.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: SOUT

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
06	CC	0E	00			SOUT	LD	C,	0	) Set checksum) to 0.
	CE	7E				S01	LD	A,	(HL)	) Accumulate
	CF	81					ADD	A,	C	) into checksum.
	D0	4F					LD	C,	A	)
	D1	7E					LD	A,	(HL)	)
	D2	CD	5D	00			CALL	SRLOUT		) Output character
	D5	23					INC	HL		) and branch back
	D6	10	F6				DJNZ	S01-$		) for the next.
	D8	C9					RET			Return.

The routine outputs a string of up to 256 characters to the serial output. It also accumulates the sum of the values into C. HL must be the address of the string of characters. B must be the length of the string. Registers HL, B, C and AF are modified by the routine.

PROGRAM LISTING Sheet 1 of 2

OGRAM: MONITOR
UTINE: ER1

.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
D9	ED	57				ER1	LD	A,	I	) Display
DB	CD	44	02				CALL	B2 HEX		) I register.
DE	CD	3C	02				CALL	SPACE		)

e next sheet.

PROGRAM LISTING Sheet 2 of 2

PROGRAM: MONITOR
ROUTINE: ER1

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
06	E1	DD	E5				PUSH	IX		)
	E3	E1					POP	HL		) Display IX.
	E4	CD	32	02			CALL	TBCD3		)
	E7	FD	E5				PUSH	IY		)
	E9	E1					POP	HL		) Display IY.
	EA	CD	32	02			CALL	TBCD3		)
	ED	21	39	0C			LD	HL,	0C39H	) Set A to stored
	F0	7E					LD	A,	(HL)	) user flags.
	F1	11	03	07			LD	DE,	0703H	) Start of string -1.
	F4	06	08				LD	B,	8	8 bits.
	F6	13				ER4	INC	DE		) Next position in string.
	F7	17					RLA			Examine next bit
	F8	F5					PUSH	AF		) Save shifted flags in A.
	F9	1A					LD	A,	(DE)	Set output value
	FA	38	01				JR	C,	ER6-$	) No output if
	FC	AF					XOR	A		) bit not set.
	FD	CD	4A	0C		ER6	CALL	$CRT		Output character
07	00	F1					POP	AF		) Restore shifted flags in A.
	01	10	F3				DJNZ	ER4-$		
	03	C9					RET			
07	04	53	5A	00	48					) String of characters'SZ.H'
	08	00	50	4E	43					'.PNC'

The routine provides an extension to the normal display of registers. Registers SP, PC, AF, HL, DE and BC have already been output. Registers I, IX and IY are output, followed by a character representation of the F register. Letters S, Z, H, P, N and C

PROGRAM LISTING Sheet 1 of 2

PROGRAM: MONITOR
ROUTINE: READ

P.C.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
0C	CD	51	00		READ	CALL	MOTFLP		Switch on LED.
0F	06	04			R1	LD	B,	4	
11	CD	3E	00		R2	CALL	CHIN		Check for start
14	FE	FF				CP	0FFH		of block
16	20	F7				JR	NZ,	R1	characters.
18	10	F7				DJNZ	R2-$		
1A	CD	3E	00			CALL	CHIN		
1D	6F					LD	L,	A	
1E	CD	3E	00			CALL	CHIN		Get start
21	67					LD	H,	A	address block
22	CD	3E	00			CALL	CHIN		number and
25	5F					LD	E	A	length.
26	CD	3E	00			CALL	CHIN		
29	57					LD	D,	A	
2A	0E	00				LD	C,	0	
2C	CD	5C	04			CALL	TXI		Display leader
									values and
2F	CD	3E	00			CALL	CHIN		check against
32	89					CP	C		checksum.
33	20	12				JR	NZ,	R6-$	
35	43					LD	B,	E	Set B to length of data.

See description of the R command.

PROGRAM LISTING Sheet 2 of 2

PROGRAM: MONITOR
ROUTINE: READ

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
07	36	0E	00				LD	C,	0	
	38	CD	3E	00		R4	CALL	CHIN		
	3B	77					LD	(HL),	A	
	3C	81					ADD	A,	C	Load the data.
	3D	4F					LD	C,	A	
	3E	23					INC	HL		
	3F	10	F7				DJNZ	R4-$		
	41	CD	3E	00			CALL	CHIN		Check against
	44	B9					CP	C		checksum.
	45	28	05				JR	Z,	R7-$	
	47	EF				R6	RST	28H		Error message.
	48	45	52	52	00		'ERR'	NOP		
	4C	CD	40	02		R7	CALL	CRLF		Put onto next
	4F	AF					XOR	A		line, check for
	50	BA					CP	D		end.
	51	CA	51	00			JP	Z,	MOTFLP	
	54	18	B9				JR	R1-$		Next block.

See description of the R command.

PROGRAM LISTING Sheet 1 of 1

OGRAM: MONITOR
UTINE: EXTENDED REGISTER DISPLAY

.C.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
56	CD	D9	06		EREG	CALL	ERI		)Call routine for)additional output
59	CD	40	02			CALL	CRLF		) Output CRLF at) end of line.
5C	00					NOP			
5D	C3	59	03			JP	STRTØ		) Return to first) EPROM to
									) continue) as normal.

his extension to the first EPROM provides a call to routine ERI, which displays dditional information about register contents.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: EXTENSION TO PARSE FOR ADDITIONAL COMMANDS

P.C.		MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
07	60	79				EPARSE	LD	A,	C	Command entered.
	61	21	6F	07			LD	HL,	ECTAB	Start of table.
	64	CD	66	04			CALL	TABLE		)Look up address)in table.
	67	B7					OR	A		) If table value) was 0, then
	68	CA	86	02			JP	Z,	PARSE	) search failed.)
	6B	EB					EX	DE,	HL	) Put address into) DE.
	6C	C3	CA	02			JP	PEND5		) Push address of) PARSE onto stack
										) and jump to) address.
										Command-Routine
07	6F	57	00	04		ECTAB				W - WRITE
	72	52	0C	07						R - READ
	75	49	14	05						I - ICOPY
	78	41	27	05						A - ARITH
	7B	48	9A	07						H - H
	7E	4E	A1	07						N - N
	81	47	A3	06						G - G
	84	00	00	00						Delimiter
	87									

This is the main extension to the first EPROM. The TABLE routine is used to search the additional command table.

PROGRAM LISTING Sheet 1 of 1

GRAM: MONITOR
TINE: H, N

.	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
9H	21	A6	07		H	LD	HL,	KEX	) Adjust to hex
9D	22	4E	0C		H1	LD	(0C4E)	,HL	) keyboard.
A0	C9					RET			)
A1	21	69	00		N	LD	HL,	0069H	) Return to
A4	18	F7				JR	H1-$		) normal KBD.

descriptions of the H and N commands. These commands modify the address of the board reflection.

PROGRAM LISTING Sheet 1 of 1

PROGRAM: MONITOR
ROUTINE: KEX

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
07	A6	CD	69	00		KEX	CALL	KBD		) Call normal
										) keyboard routine
	A9	D0					RET	NC		) No key pressed,
										) so return.
	AA	E5					PUSH	HL		Save HL.
	AB	21	08	0C			LD	HL,	0C08H	) Point to table
										) for space key.
	AE	FE	20				CP	20H		) If space, then
	B0	37					SCF			) normal end.
	B1	28	0B				JR	Z,	KX3-$	)
	B3	CB	66				BIT	4,	(HL)	) If space not
										) held down, then
	B5	28	07				JR	Z,	KX3-$	) normal end.
	B7	CD	44	02			CALL	B2HEX		) Output hex value
										) and a space to
	BA	CD	3C	02			CALL	SPACE		) the display.
										) Return no value
	BD	B7					OR	A		) from routine.
	BE	E1				KX3	POP	HL		) Restore HL,
	BF	C9					RET			) return.

See description of the H command. This routine is substituted for the normal KBD routine.

PROGRAM LISTING Sheet 1 of 2

AM: MONITOR
NE: CURSOR MOVEMENT ROUTINES : MCR, MCL, MCD, MCU

	MACHINE CODE				LABEL	MN.	Op.1	Op.2	COMMENTS
0	D5				MCR	PUSH	DE		) Move cursor
1	23					INC	HL		) right.
2	0C					INC	C		
3	79					LD	A,	C	
4	FE	31				CP	31H		) Less than 49
6	38	33				JR	C,	ECM-$	) is OK.
8	0E	01				LD	C,	1	) Set to
A	11	00	FF			LD	DE,	0FFD0H	) column 1.
D	18	2B				JR	ECMA-$		)
F	D5				MCL	PUSH	DE		Move cursor left.
0	2B					DEC	HL		
1	0D					DEC	C		
2	20	27				JR	NZ,	ECM-$	
4	0E	30				LD	C,	30H	) Set to
6	11	30	00			LD	DE,	30H	) column 48.
9	18	1F				JR	ECMA-$		)
B	D5				MCD	PUSH	DE		Move cursor down.
C	11	40	00			LD	DE,	40H	) Add 64.
F	19					ADD	HL,	DE	)
0	04					INC	B		
1	78					LD	A,	B	
2	FE	10				CP	10H		) Less than 16
4	38	15				JR	C,	ECM-$	) is OK.
6	06	01				LD	B,	1	Set to line 1.

routines allow the cursor to be moved to any position, quickly. The cursor is

PROGRAM LISTING Sheet 2 of 2

PROGRAM: MONITOR
ROUTINE: CURSOR MOVEMENT ROUTINES : MCR, MCL, MCD, MCU

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
07	EB	11	40	FC			LD	DE,	0FC40H	(-03C0H)
	EB	18	0D				JR	ECMA-$		
	ED	D5				MCU	PUSH	DE		Move cursor up.
	EE	11	C0	FF			LD	DE,	0FFC0H	) Subtract 64.
	F1	19					ADD	HL,	DE	)
	F2	05					DEC	B		
	F3	20	06				JR	NZ,	ECM-$	
	F5	06	0F				LD	B,	0FH	) Jump to line 15.
	F7	11	C0	03			LD	DE,	03C0H	)
	FA	19				ECMA	ADD	HL,	DE	Add in jump.
	FB	7E				ECM	LD	A,	(HL)	) Set A to value
	FC	D1					POP	DE		) on display.
	FD	CB	47				BIT	0,	A	) Condition flag
	FF	C9					RET			) for bit 0.

HL = Current cursor address) Updated by the routines.
B = Row (1-15)(Scrolled area)) The values of B and C supplied must correspond to
C = Column (1-48)) the display address HL.

PROGRAM LISTING Sheet 1 of 2

PROGRAM: ORIGINAL MONITOR
ROUTINE: LIST OF MODIFICATIONS

P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
00	45	30	F7				JR	NC,	CHIN-$	) Tape load fix.
	47	DB	01				IN	A,	(1)	)
	49	C9					RET			)
01	09	A9	(8A)	A1	(99)					Key position fix.
00	B1	(3A)	09	(0C)			LD	A,	(KMAP)	Shift key fix.
00	20	E3					EX	(SP),	HL	) ALLOW NMI
	21	2B					DEC	HL		) Decrement PC,
	22	E3					EX	(SP),	HL	) on stack.
	23	(C3)	05	(03)			JP	TRAP		)
03	3A	00					NOP			) Do not decrement) DE, already done
03	05	F5				TRAP	PUSH	AF		) Save registers.
	06	E5					PUSH	HL		)
	07	3A	00	0C			LD	A,	(PORTØ)	) Reset NMI flag.
	0A	D3	(00)				OUT	(0),	A	)
	0C	3A	1A	0C			LD	A,	(CONFLG)	) If CONFLG is 0,) then not an E
	0F	B7					OR	A		) command (so S,) NMI, or
	10	28	13				JR	Z,	TR1-$	) breakpoint).
	12	2A	15	0C			LD	HL,	(BRKADR)	) Set breakpoint.
	15	7E					LD	A,	(HL)	)
	16	32	17	0C			LD	(BRKVAL),	A	)
	19	36	E7				LD	(HL),	0E7H	)

PROGRAM LISTING Sheet 2 of 2

PROGRAM: ORIGINAL MONITOR
ROUTINE: LIST OF MODIFICATIONS

	P.C.	MACHINE CODE					LABEL	MN.	Op.1	Op.2	COMMENTS
	03	1B	AF						A		) Set CONFLG to 0.
		1C	32	1A	0C			LD	(CONFLG)	A	)
		1F	00	00				NOP	NOP		Spare
		21	E1					POP	HL		) Restore
		22	F1					POP	AF		) registers.
		23	(ED)	(45)				RETN			
		25	(D5)				TR1	PUSH	DE		
5.	02	3A	00	(00)				NOP	NOP		) Remove stupid) code.
6.	02	BA	(7E)				PEND1	LD	A,	(HL)	) If command
		BB	(B7)					OR	A		) letter not in
		BC	CA	60	07			JP	Z,	0706H	) table, jump to
		BF	23					INC	HL		) second EPROM.
		C0	B9					CP	C		)
		C1	28	04				JR	Z,	PEND4-$	)
7.	00	8A	C3	8A	04			JP	048AH		) Jump to extended) keyboard routine
			00					NOP			) in second EPROM.
8.	03	FE	76	76				HALT	HALT		Two halts added.
9.	03	56	C3	56	07			JP	0756H		) Jump to Extended) Register Display
10.	02	2B	F5					PUSH	AF		) Use stack) instead of D to
	02	2E	F1					POP	AF		) hold A, in) TBCD2.

INSTALLATION INSTRUCTIONS

B-Bug is supplied protected from static charges. When installing take care to avoid any danger of static charges in the vicinity and treat B-Bug as if it were a delicate CMOS device.

Make sure that the power supply has been off for at least thirty seconds.

Plug B-Bug 2 into the empty socket next to the 'Nasbug' 2708, being sure not to bend any of the pins either outwards or under the chip. B-Bug 2 should be orientated the same way round as Nasbug. Now unplug Nasbug and replace it with B-Bug 1, as above.

Keep Nasbug safe because it can be reprogrammed.

Forthcoming Nascom compatible software: Tiny Basic
Assembler