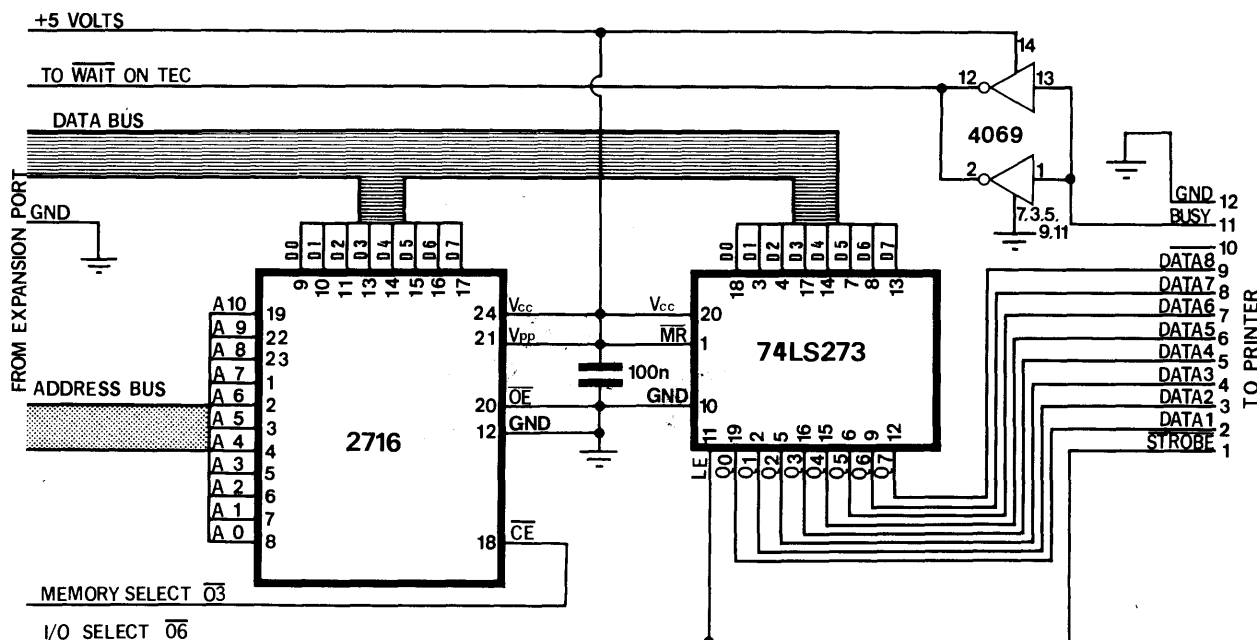


Connecting the TEC to a:

Parts: \$24.80 PC board: \$3.80

PRINTER/PLOTTER



Printer/Plotter circuit

Buy a Dick Smith VZ 200 Printer-Plotter.

This project explains how to directly access (talk to) a PRINTER/PLOTTER. We have used the most readily available printer/plotter as it is not only the cheapest, but can be obtained from a number of suppliers.

Talking to one of these clever little performers is not very involved when you know how. But without the correct information it will remain completely DEAD. When you know how to supply it with the right stuff, it will do practically anything bar talk to you.

Actually you only have to send it the necessary codes to produce the character, all the creation of the shape of the symbol is done by the chip within the printer.

Not only do these printer/plotters accept instructions to produce numbers letters and symbols, but they can also be told to rotate, plot, vary the size of the characters and move in almost any direction.

There is a two-way interaction between printer and computer. Data is sent to the printer faster than it can be executed and to save holding up the computer, it is deposited in a FIFO register in blocks of about 4 bytes (in our case). Larger computers can be instructed to go away and execute other work while the FIFO register empties.

Bursts of data are transmitted like this until the whole program is executed.

As we have used a standard printer, it is obvious that it has been designed to connect to any computer which has a normal, full-size, key-board so that each key will produce the corresponding letter on the paper.

But this luxury is not absolutely necessary as the computer merely produces a code number which is sent to the printer.

The code number (or value) is called an ASCII number or ASCII CODE and fortunately is identical for all types and models of personal computers.

The secret to getting the printer to work on the TEC is the latch chip. It holds the data long enough for the printer to read it.

PARTS LIST

- 1 - 100n mono block
- 1 - 4049
- 1 - 74LS273
- 1 - 2716 (programmed)
- 1 - 14 pin IC socket
- 1 - 20 pin IC socket
- 1 - 24 pin IC socket
- 1 - 24 pin wire-wrap socket
- 1 - 24 pin DIP HEADER
- 1 - 36 pin Centronics type plug.
- tinned copper wire
- hook-up flex
- 3 - 'quick connect' pins and sockets

PRINTER INTERFACE PC BOARD

This means all we have to do is produce the same set of ASCII numbers (or codes) and the printer will produce the correct set of shapes on the paper.

Thus we don't need a full-size computer at all.

It may be a bit slow pressing the keys on the TEC, but all the printing capabilities will be possible, and that's all we want.

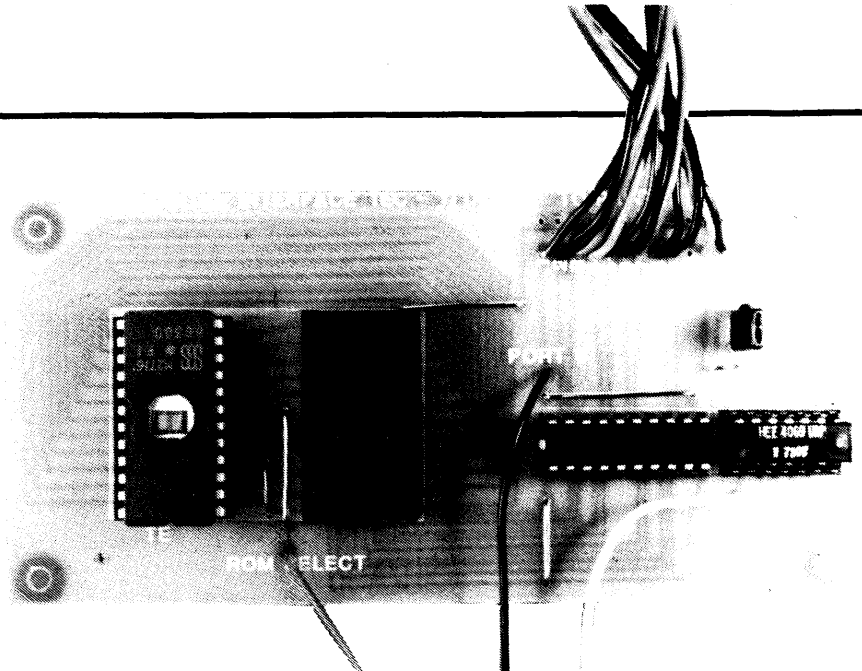
In this series of articles, we will explore the functions of the printer/plotter and create some amazing effects.

The most important aspect of this is realizing you can create a CONTROL PROGRAM with machine code listings and thus fill the minimum amount of memory for any given effect.

In this way you can produce your own system and expand it as much as you like without having to resort to buying a ready-made console. This will produce a cheaper and more compact system and will gain you much more respect from your boss or customer.

The first part of this project requires assembly of the printer interface board. This board contains a latch and EPROM (filled with a number of handy programs). This will give you a run-up program to test the interface board and provide instant transfer of data from computer to paper to reduce the amount of button-pushing.

The other chip on the board provides an inverted WAIT signal to halt the Z80. This basically keeps the two units in synchronisation.



The printer/plotter interface board complete and ready for plugging into the TEC & printer.

Set out all the parts on your bench and check everything. Solder the sockets, cap and 6 jumper links to the board. Mount the wire-wrap socket through the board so that the long pins act as 'stand-offs' for the component header plug. See the RELAY DRIVER BOARD article and photos for details of how this is done.

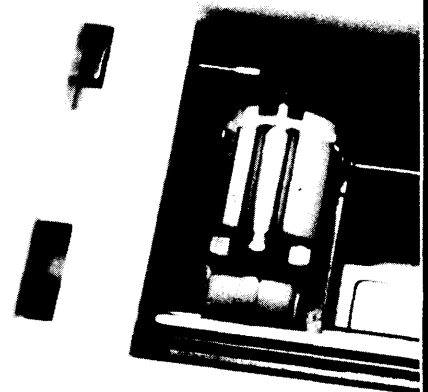
The final task involves connecting the board to the 36 pin Centronics plug.

WIRING THE PLUG

Wiring the Centronics-types plug to the printer interface is very easy. On the printer interface PC board there are 24 holes. Twelve of these are numbered. These numbers correspond to the numbers on the Centronics plug. Solder a length of

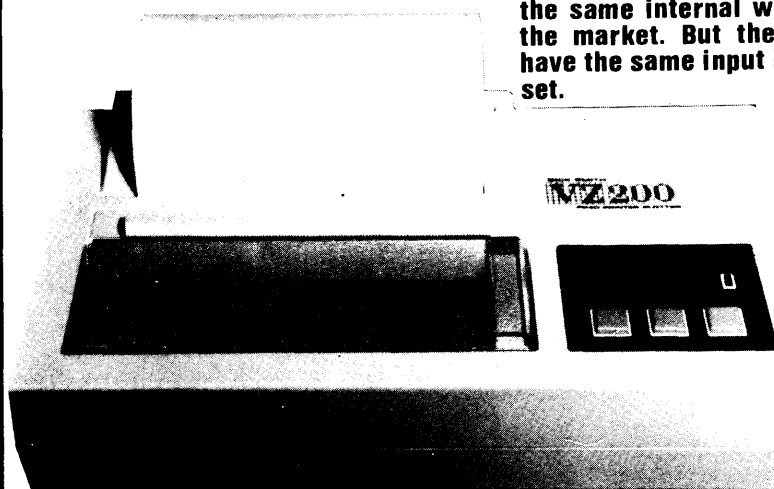
We used a VZ 200 printer/plotter but there are other units with the same internal workings on the market. But they may not have the same input instruction set.

The pens use water-based ink and tend to dry-out fairly easily. If they fail to start: open them up, add a drop of water, heat them up and fit them back into the printer.



A close-up of the 4-pen print head.

hook-up wire between each hole and a corresponding hole on the connector plug. Pin 10 is not used, so no lead is needed. It is not necessary to use special connecting flex such as twisted pairs or screened lead. Our prototype worked perfectly with ordinary hook-up flex. It's best to use different coloured flex for each line to make tracing easier. These leads can be about 50 cm long and kept together with ties or tape at regular intervals.



The only remaining wires left are the 3 control lines. These are:

**Memory select 03,
I/O select 06, and
WAIT.**

These are fitted with 'quick connect' terminals which push onto matrix pins on the main PC board. Heat-shrink tubing can be placed over the terminals to strengthen the solder joint and make them easier to handle.

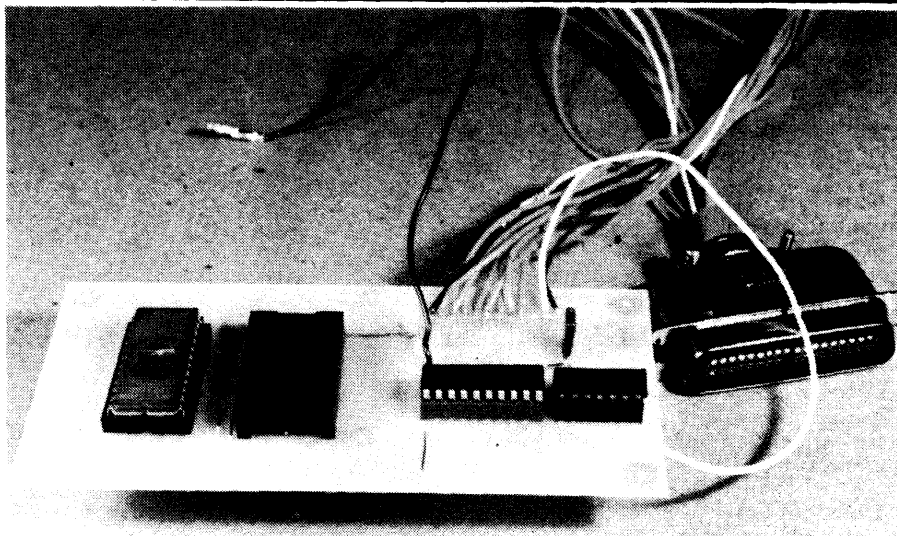
When the printer is first turned on it runs through an initial program (from its internal memory) which feeds the paper, sets the pen colours and starts the ink flowing by producing a box with each pen.

After this, there is very little else you can do via the buttons on the unit, except forward feed, change the colour of the pens and/or remove them.

All the rest of the action must come in the form of data from an outside source.

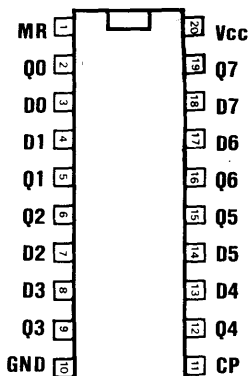
This is why we need the TEC. It supplies data at high speed to get the print-head moving.

Connect the centronics plug into the rear of the printer and fit the **PRINTER INTERFACE PC BOARD** to the computer. Connect the 3 flying leads as shown in the diagram:



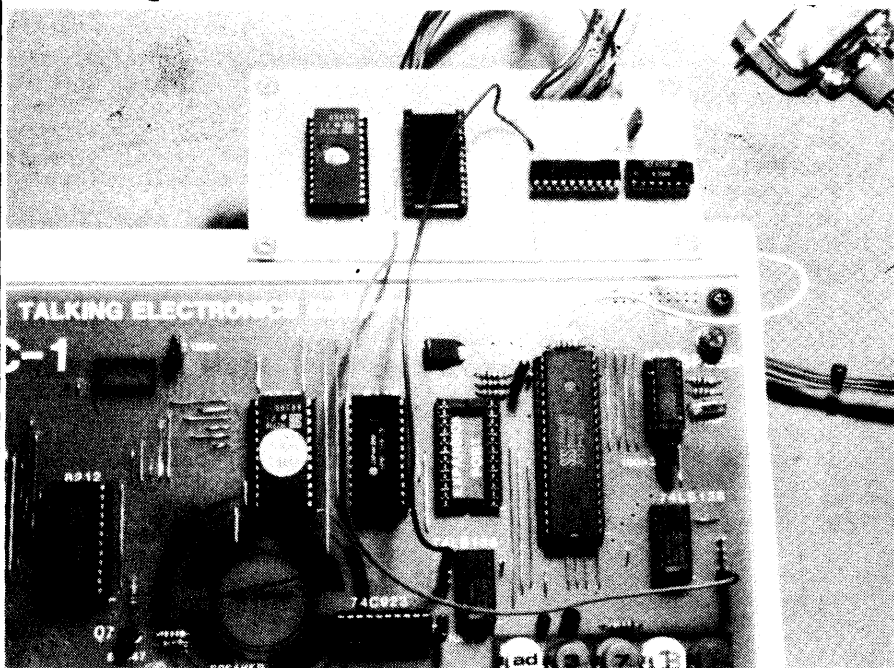
All the parts shown are included in the kit.

74LS 273



Pin-out for 74LS273

This photo shows the connections to the TEC.



To get something interesting out of the printer you will need to send it a program. The first of these is:

KEN's START-UP PROGRAM:

Make sure the print-head is to the left of the printer as when the printer has been switched on.

Push **Address 18A0 GO GO.**

Watch the result.

This type of program is beyond us at the moment but you will be capable of similar effects after reading this article.

For now, the next step is to be able to get letters and characters onto the paper.

PRODUCING LETTERS etc. . .

All information is fed to the printer in ASCII code. If you want a particular character, the correct code must be sent to the printer. Even if you want to send a number to the printer, such as 150, you must send it in the form of ASCII. This means 150 translates to 31 35 30, as you will see later from the table.

A small program is required to interpret your button pushing and send it to an output port. This is similar to making a segment on the display illuminate and the program for this is contained in the **PRINTER/PLOTTER EPROM** at 1980.

To use this program:

Press **Address 1980 GO GO**

The display will go blank and the TEC will be ready for conveying your keyboard instructions directly to the printer.

Each of the letters, numbers and symbols is shown in the table below and the corresponding hex value must be used for the symbol to appear on the paper.

Try obtaining all the letters, numbers and characters by following through the table.

Any sentence you send to the printer via the keyboard can be re-presented again and again if placed into memory before-hand. It can also be corrected and adjusted (within limits). To do this, place the data at **0800** and call a program at **1880**.

PRINTER/PLOTTER ASCII VALUES:

	SPACE	20	0	30	@	40	P	50	'	60	p	70
BACK SPACE	08	21	1	31	A	41	Q	51	a	61	q	71
LINE FEED	0A	22	2	32	B	42	R	52	b	62	r	72
CR	0D	23	3	33	C	43	S	53	c	63	s	73
DC1	11	24	4	34	D	44	T	54	d	64	t	74
DC2	12	25	5	35	E	45	U	55	e	65	u	75
NEW COLOUR	1D	26	6	36	F	46	V	56	f	66	v	76
		27	7	37	G	47	W	57	g	67	w	77
		28	8	38	H	48	X	58	h	68	x	78
		29	9	39	I	49	Y	59	i	69	y	79
		2A	:	3A	J	4A	Z	5A	j	6A	z	7A
LINE BEFORE	0B	2B	;	3B	K	4B	[5B	k	6B	(7B
		2C	<	3C	L	4C	\	5C	l	6C)	7C
		2D	=	3D	M	4D]	5D	m	6D	}	7D
		2E	>	3E	N	4E	^	5E	n	6E	~	7E
		2F	?	3F	O	4F	_	5F	o	6F	⊠	7F

Try the following sequence and you will see a word appear:

49 4E 43 52 45 44 49 42 4C 45

For the hex value **49**, the letter **I** will be printed. Press each number only ONCE. The first press will appear to have no effect, but as soon as the second button is pressed, the letter **I** will be printed.

Be very careful not to press button-sequence **11** or **12** as this will cause the mode to change and everything will appear to 'lock-out'.

Try writing a sentence using the hex key pad. It's slow but eventually gets you there. A space between words is created by typing **20**.

Insert the following at **0800**:

49 4E 43 52 45 44 49 42 4C 45 20 20 48 55 4C 4B 0D 0A 1D FF.

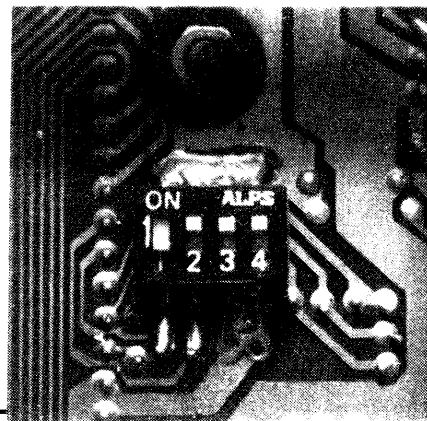
Push **ADDRESS 1880 GO GO**.

Recall it again by pressing:
ADDRESS 1880 GO GO.

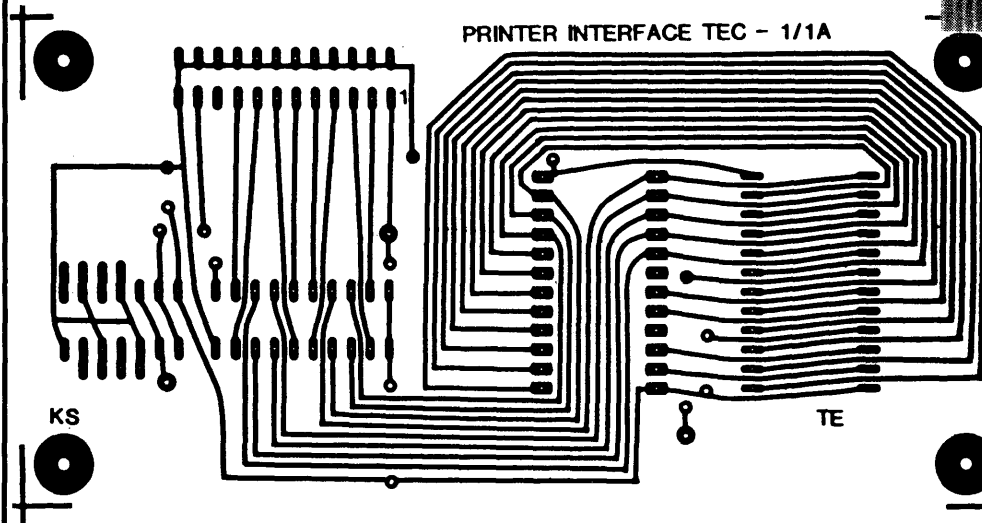
THE LIST PROGRAM

This program lists any part of the EPROM, RAM or any additional memory you add to your TEC. In fact the first thing you can do is get a print-out of your MONitor ROM. Many readers have written requesting a listing of the MONitor and now they can produce it themselves.

But before you can get a listing, you must make a modification to the operation to the printer. This involves setting the two switches under the printer:



PRINTER INTERFACE TEC - 1/1A



The PC layout for the Printer/Plotter. The overlay and parts positioning can be gained from the photo on P 31.

This is how to do it.

On the bottom of the printer is a small plate. Undo the screws and remove the plate. Inside you will find a bank of 4 switches. Switch 1 should be in the OFF position and switch 2 in the ON position. Don't worry about switch 3 and 4.

When the switches are set like this, CR (carriage Return) will set the print-head to the left of the paper without feeding the paper forward. The paper can then be fed forward by using LF (Line Feed). The switches should be set like this because the program in ROM automatically line feeds after each carriage return. If the switches are not set like this, the typing will be double line spaced.

Enter the following into the TEC:
Address 1800 GO GO:

The display will go blank and the printer will CR and LF. Now enter **0000** and the printer will start printing out characters in pairs. This is a listing of the contents of your monitor ROM.

If you want a listing of any of the programs you have typed into memory, start at **0800** or where your program starts, and enter a 4-digit number into the keyboard. It must be 4 digits, so don't forget the leading 0.

The text mode is not very interesting. After all, we have seen electric/electronic typewriters for years. But for a print-head to produce **GRAPHICS!** That's different!

GRAPHICS MODE

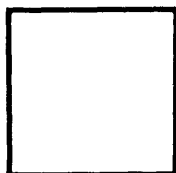
The program at **1880** can also be used to generate graphics on the printer.

Remember, all information must be programmed into the printer in ASCII.

Type the program below into the TEC's memory at **0800**. An **FF** is placed after the last piece of data to signify the end of a program. Now run the program at **1880** by pressing **Address 1880 GO GO**.

at 0800: 0A 0D 12 49 2C 44
38 30 2C 30 2C 38 30 2C 2D
38 30 2C 30 2C 2D 38 30 2C
30 2C 30 0D FF.

The printer will draw a square.



Look at the listing. It may look complex but can be easily decoded using the table. It will decode to this:

OA = LF = Line Feed.
OD = CR = Carriage Return
12 = DC2 = Graphic Mode
49 = I = sets the pen's location as co-ordinates 0,0.
2C = , = Separates I from D
44 = D = draw from present location to the co-ordinate given by the next byte(s) of data.

38 = 8

30 = 0

2C = ,

30 = 0

2C = ,

38 = 8

30 = 0

2C = ,

2D = -

38 = 8

30 = 0

2C = ,

30 = 0

2C = ,

2D = -

38 = 8

30 = 0

2C = ,

30 = 0

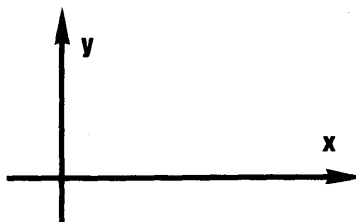
2C = ,

30 = 0

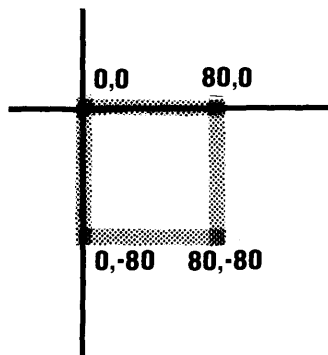
OD = CR = carriage return

FF = signifies end of program.

The printer uses a co-ordinate system exactly like the x,y axis used to draw graphs. The origin is 0,0 (or 00,00) and the positive direction of x and y is shown on the diagram.



The co-ordinates of the corners of the box are shown in this diagram. This clearly shows how the values are obtained.



The program can be separated into 4 sections, each drawing one side of the box. This will show how the program goes together.

The following program produces the top of the square:

at 0800 type: 0A 0D 12 49 2C 44 38
30 0D FF.

Address 1880 GO GO.

The result will be:

Let us produce a line the full width of the paper. For this you will need a 3-digit value. The printer is capable of accepting a value as high as 999 (also -999) but this will be too high for our width of paper. Try 300.

The ASCII value is 33 30 30.

at 0800:

0A 0D 12 49 2C 44 33 30 30 2C 30
0D FF.

Press **Address 1880 GO GO.**

The final **0D** is important to get the printer to execute the graphics command.

The value **300** will not quite reach the far side of the paper. Try **450**. This will be about the longest line possible and don't forget to use the ASCII values in the program.

Shorten the side of the box to 80 and continue with the experiment.

The second side of the box will be produced at an angle other than 90° by inserting the following co-ordinates: 50, -80

at 0800:

0A 0D 12 49 2C 44 38 30 2C 30 2C 35
30 2C 2D 38 30 0D FF.

Run the program. Does it produce two sides of an irregular figure?

The next side will be produced as follows:

0A 0D 12 49 2C 44 38 30 2C 30 2C
35 30 2C 2D 38 30 2C 31 35 30 2C
2D 38 30 0D FF.

Run the program and see the result.

Finally:

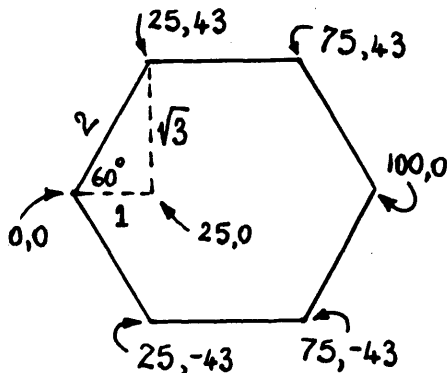
0A 0D 12 49 2C 44 38 30 2C 30 2C
35 30 2C 2D 38 30 2C 31 35 30
2C 2D 38 30 2C 0A 0D 11 1D 0D FF.

Produce other shapes and you will understand how to plot co-ordinates.

HEX

The second shape we will investigate is a HEXAGON.

To produce this shape you need to know the value of the internal angle and produce a 30° 60° 90° triangle as shown. This will give you the length of the sides of the triangle and from this the first set of co-ordinates can be obtained (25,43) These values are 1/4 of 100, 173, which are the lengths of the sides of the triangle.

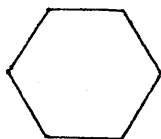


The second co-ordinate, 75,43 is found by adding 50 to the value 25. Continue around the hex shape until the figure is closed.

This is the listing for the printer:

at 0800:

```
12 49 0D 44 32 35 2C 34 33 2C 37
35 2C 34 33 2C 31 30 30 2C 30 2C
37 35 2C 2D 34 33 2C 32 35 2C 2D
34 33 2C 30 2C 30 0D FF.
```



o's and X's

The new instruction with this shape is the MOVE command.— 4 D

This instructs the pen to lift from the page and move to a specified location without drawing on the paper.

Here is the listing and the shape which will be drawn:

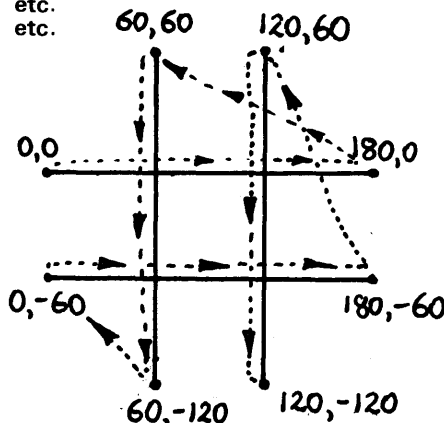
```
0800 0A 0D 12 49 0D 44 31 38
0808 30 2C 30 0D 4D 36 30 2C
0810 36 30 0D 44 36 30 2C 2D
0818 31 32 30 0D 4D 30 2C 2D
0820 36 30 0D 44 31 38 30 2C
0828 2D 36 30 0D 4D 31 32 30
0830 2C 36 30 0D 44 31 32 30
0838 2C 2D 31 32 30 0D FF
```

This is a decoding of the first part of the listing. This will be sufficient to understand how the program is written.

0A = LF = Line Feed
 0D = CR = Carriage Return
 12 = DC2 = Graphics Mode
 49 = I = initialize the co-ords 0,0
 0D = CR = signifies the end of the previous command. It does not cause the carriage to return but enables the previous command to be carried out.

44 = D = Draw
 31 = 1
 38 = 8
 30 = 0
 2C = ,
 30 = 0
 0D = CR = end of draw statement.
 4D = M = Move. The pen is instructed to move without drawing.

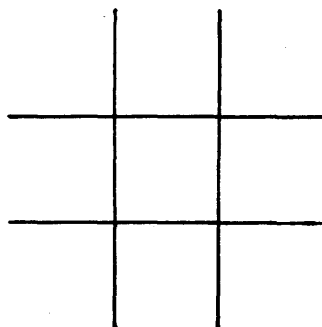
36 = 6
 30 = 0
 2C = ,
 36 = 6
 30 = 0
 0D = CR End of Move statement.
 44 = D = Draw
 36 = 6
 30 = 0
 etc.
 etc.



This diagram shows the value of the co-ordinates required to draw the shape.

Copy out the complete listing and decode it to prove that the path taken by the print-head is as shown in the diagram.

Address 1880 to use.



WAR GAMER'S DELIGHT

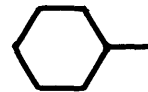
The full impact of this effect is shown on the next page.

The first thing you notice about the program is a set of values at the beginning which the printer does not recognise. This means they must be Machine Code values for an 'operations' program for the Z80. And they are.

The program produces a honey-comb pattern.

Anyone into war games will soon recognise the possibilities of the honey-comb as a playing board. The reason is each block has 6 borders, increasing the possible moves and thus the strategy, over a regular field of squares.

This shape is created using a picture element of a hexagon attached to a straight line thus:



This pattern is repeated 4 times across the paper and then a move to a new starting point -450, -86 down the paper.

The co-ordinates of the new starting point can be explained as follows:

After each picture element is drawn, the printer is initialized. This means that the present co-ordinate of the pen is taken as 00,00.

This gives us a value of 450,00 for the commencement of the 4th picture element with reference to the origin.

The next row of hexagons commence at the left-hand edge, which is -450 with reference to the above X co-ordinate and a y value of -86, with reference to the y value above.

The only way to understand how the honey-comb has been produced is to decode the listing. It contains two loops, one to draw the picture element and the other to count-to-4 across the screen.

Write each of the ASCII codes in a single file and alongside it place the printer value it represents.

You can experiment further by making the hexagons smaller. This will use a 2-digit ASCII value for the length of the sides. In the program, the original 3-digit ASCII values have been converted to 2-digit by using 00 for the 3rd value.

WAR GAMER'S DELIGHT

The first 18H bytes (31 bytes is the MAIN program and this contains the instructions to fetch one byte of data from the printer program and send it to the printer.

Data for the printer is stored in the form of a BYTE TABLE and starts at 0820.

The main program is divided into two separate parts. 0800 - 080F is a loop which loads the printer program and runs it 4 times.

811 - 81D loads the data for the MOVE commands and each piece of data is sent to the printer until FF is detected.

The count-to-4 operation is performed by DJNZ (at 80F) which automatically decrements register B by ONE on each pass of the loop until it becomes zero.

The program then advances to loading HL register-pair with the contents of memory location 0860 and this instruct the print-head to move to the left-hand edge and down the paper to a new starting point. The main program then jumps to the start (0800) via instruction JR Z E7 (at 817).

```

800 06 LD B,04
801 04
802 21 LD HL,0820
803 20
804 08
805 7E LD A,(HL)
806 FE CP FF
807 FF
808 28 JR Z 05
809 05
80A D3 OUT (06),A
80B 06
80C 23 INC HL
80D 18 JR F6 (to 805)
80E F6
80F 10 DJNZ F1 (to 802)
810 F1
811 21 LD HL 0860
812 60
813 08
814 7E LD A,(HL)
815 FE CP FF
816 FF
817 28 JR Z E7 (to start)
818 E7
819 D3 OUT (06),A
81A 06
81B 23 INC HL
81C 18 JR F6 (to 814)
81D F6
81E FF NOT USED
81F FF NOT USED
    
```

DATA FOR PRINTER:

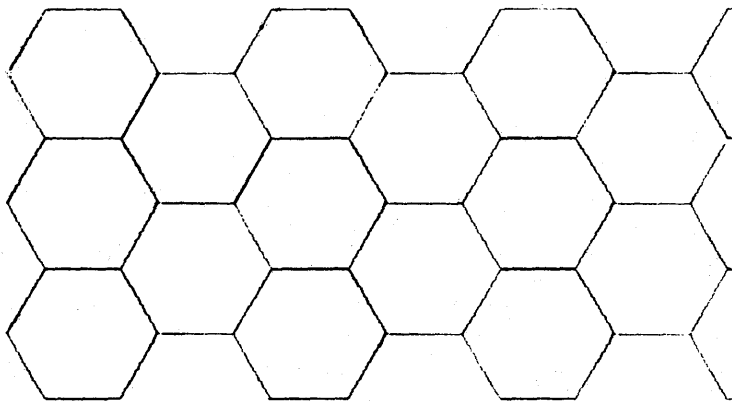
```

820 12 Graphics Mode
821 49 Initialize
822 0D CR
823 44 Draw
824 32 2
825 35 5
826 2C ,
827 34 4
828 33 3
829 2C ,
82A 37 7
82B 35 5
82C 2C ,
82D 34 4
82E 33 3
82F 2C ,
830 31 1
831 30 0
832 30 0
833 2C ,
834 30 0
835 2C ,
836 37 7
837 35 5
838 2C ,
839 2D -
83A 34 4
83B 33 3
83C 2C ,
83D 32 2
83E 35 5
83F 2C ,
840 2D -
841 34 4
842 33 3
843 2C ,
844 30 0
845 2C ,
846 30 0
847 0D CR
848 4D Move
849 31 1
84A 30 0
84B 30 0
84C 2C ,
84D 30 0
84E 0D CR
84F 44 Draw
850 31 1
851 35 5
852 30 0
853 2C ,
854 30 0
855 0D CR
856 FF End
    
```

Data for MOVE COMMANDS:

```

860 4D Move
861 2D -
862 34 4
863 35 5
864 30 0
865 2C ,
866 2D -
867 38 8
868 36 6
869 0D CR
86A FF End
    
```

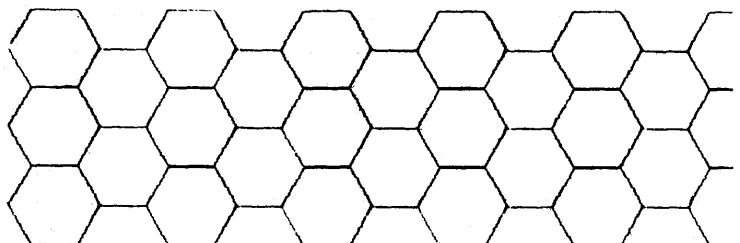


```

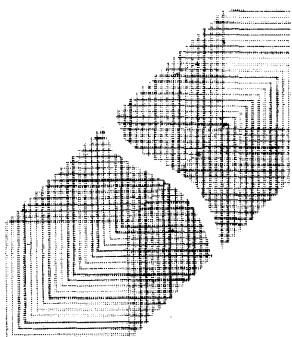
0800 06 04 21 20 08 7E FE FF
0808 28 05 D3 06 23 18 F6 10
0810 F1 21 60 08 7E FE FF 28
0818 E7 D3 06 23 18 F6 FF FF
0820 12 49 0D 44 32 35 2C 34
0828 33 2C 37 35 2C 34 33 2C
0830 31 30 30 2C 30 2C 37 35
0838 2C 2D 34 33 2C 32 35 2C
0840 2D 34 33 2C 30 2C 30 0D
0848 4D 31 30 30 2C 30 0D 44
0850 31 35 30 2C 30 0D FF FF
0858 FF FF FF FF FF FF FF FF
0860 4D 2D 34 35 30 2C 2D 38
0868 36 0D FF
    
```

```

3800 06 06 21 20 08 7E FE FF
3808 28 05 D3 06 23 18 F6 10
0810 F1 21 60 08 7E FE FF 28
0818 E7 D3 06 23 18 F6 FF FF
0820 12 49 0D 44 32 35 2C 34
0828 36 2C 34 35 2C 32 36 2C
0830 00 36 30 2C 30 2C 34 35
0838 2C 2D 32 36 2C 31 35 2C
0840 2D 32 36 2C 30 2C 30 0D
0848 4D 00 36 30 2C 30 0D 44
0850 39 30 00 2C 30 0D FF FF
0858 FF FF FF FF FF FF FF FF
0860 4D 2D 34 35 30 2C 2D 35
0868 32 0D FF
    
```



COMPUTER GRAPHICS



Being able to draw some of the basic shapes (as we have shown), opens up a whole new world of computer graphics.

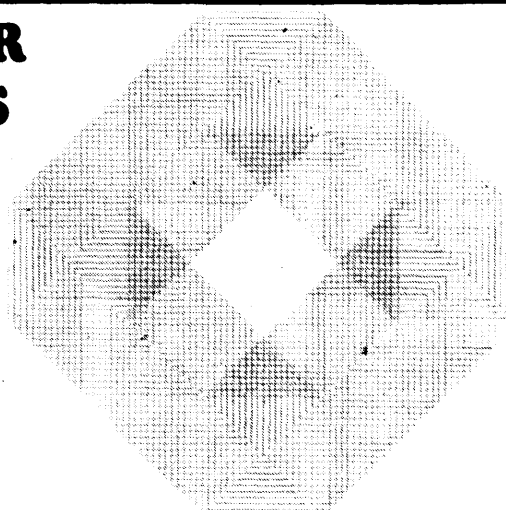
If we take the box-shape, we can produce a very effective pattern simply by re-defining the start co-ordinates and repeating the shape many times. The result can be anything from a 'check-tie' to an irregular octogon.

The colourful patterns which can be obtained (of which we can only see the result in black and white) is produced by a combination of drawing, shifting and colour-changing. The first of these to be investigated will be an irregular octogon or DIAMOND.

We have already outlined the structure of the program and briefly it is a set of instructions which are loops. Each sets a particular condition and then decrements on each pass.

For the diamond shape, a square is generated at the origin, 00,00 via the program at 0880. The lengths of the sides of the square are 80. When the 4 sides have been drawn, the pen lifts off the paper and moves to a new origin with co-ordinates 04,04. The program is now up to location 083C. It then jumps to 0842. The contents of the accumulator (which is the value at location 0860 i.e. 00) is loaded into 089A. Register pair DE is incremented and now looks at location 0861. The value 34 is loaded into the accumulator. At 0847 the contents of the accumulator is loaded into location 089B.

So far, the program at 0880 has not been altered but the next two sets of



load instructions will change the data at 89D, 89E from 00 34 to 2D 34 and this will create the movement in the negative direction.

800	06	06	3E	0A	D3	06	10	FC
808	3E	12	D3	06	11	60	08	06
810	08	C5	06	04	21	80	08	7E
818	FE	FF	CA	23	08	D3	06	23
820	C3	17	08	10	EF	3E	11	D3
828	06	3E	1D	D3	06	AF	D3	06
830	3E	12	D3	06	C1	10	DA	1A
838	FE	FF	C2	42	08	3E	11	D3
840	06	C7	32	9A	08	13	1A	32
848	9B	08	13	1A	32	9D	08	13
850	1A	32	9E	08	13	C3	0F	08
858	00	FF	00	FF	00	FF	00	FF
860	00	34	2D	34	2D	34	2D	34
868	2D	34	00	34	FF	FF	FF	FF
870	FF	FF	FF	FF	FF	FF	FF	FF
878	FF	FF	FF	FF	FF	FF	FF	FF
880	49	2C	44	38	30	2C	30	2C
888	38	30	2C	2D	38	30	2C	30
890	2C	2D	38	30	2C	30	2C	30
898	0D	4D	00	34	2C	00	34	0D
8A0	FF	FF	FF	FF	FF	FF	FF	FF

The Machine Code listing required to produce the DIAMOND.

Complete decoding of the above listing, with explanations.

800	06	06	LD B,06	Load B with 6
802	3E	0A	LD A,0A	Load A with the Forward Feed instruction
804	D3	06	OUT (06),A	OUT to the printer port
806	10	FC	DJNZ 0804	create 6 loops of forward feed
808	3E	12	LD A,12	Select the Graphics Mode
80A	D3	06	OUT (06),A	OUT to the printer
80C	11	60 08	LD DE,0860	Load DE with start of Direction Change TABLE
80F	06	08	LD B,08	Sets number of colour changes before a direction change
811	C5		PUSH BC	Save B. B must be paired with C to be saved
812	06	04	LD B,04	Sets number of squares for each colour
814	21	80 08	LD HL,0880	Load HL with start of DRAWING TABLE
817	7E		LD A,(HL)	Load the data at 880 into A
818	FE	FF	CP FF	detects end of TABLE
81A	CA	23 08	JP Z,0823	At end of table, jump to 823
81D	D3	06	OUT (06),A	OUT data value at 880 to printer
81F	23		INC HL	Increment to 881, 882 etc
820	C3	17 08	JP 0817	Jump to 817 to increment through Drawing Table
823	10	EF	DJNZ 0814	Loop DRAWING TABLE 4 times
825	3E	11	LD A,11	Change to TEXT MODE
827	D3	06	OUT (06),A	OUT to printer
829	3E	1D	LD A,1D	NEXT COLOUR
82B	D3	06	OUT (06),A	OUT to printer
82D	AF		XOR A	Clear A
82E	D3	06	OUT (06),A	OUT to printer
830	3E	12	LD A,12	Select GRAPHICS MODE
832	D3	06	OUT (06),A	OUT to printer
834	C1		POP BC	Get B from STACK. Actually BC.
835	10	DA	DJNZ 0811	Decrement B and jump to 811 for 6 loops
837	1A		LD A,(DE)	Load A with data at 860 etc
838	FE	FF	CP FF	Detects end of DIRECTION CHANGE program
83A	C2	42 08	JP NZ,0842	If not zero, jumps to 842
83D	3E	11	LD A,11	If zero, change to TEXT MODE
83F	D3	06	OUT (06),A	OUT to printer
841	C7		RST 0	END OF PROGRAM. ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
842	32	9A 08	LD (089A),A	Load first byte of Direction Change table into location 089A
845	13		INC DE	Increment DIRECTION CHANGE table
846	1A		LD A,(DE)	Load next byte of Direction Change table into A
847	32	9B	LD (089B),A	Load this byte into location 089B
84A	13		INC DE	Increment the DIRECTION CHANGE table
84B	1A		LD A,(DE)	Load the third byte into the accumulator
84C	32	9D 08	LD (089D),A	Load this third byte into location 089D
84D	13		INC DE	Increment the DIRECTION CHANGE TABLE
84E	1A		LD A,(DE)	Load the fourth byte of the direction change table into A
84F	32	9E 08	LD (089E),A	Load this fourth byte into location 089E
850	13		INC DE	Increment the DIRECTION CHANGE table ready for next
851	C3	0F 08	JP 080F	Jump to 80F to commence the next direction

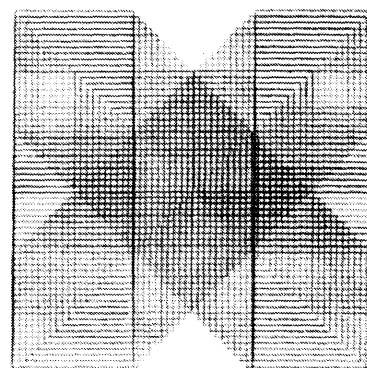
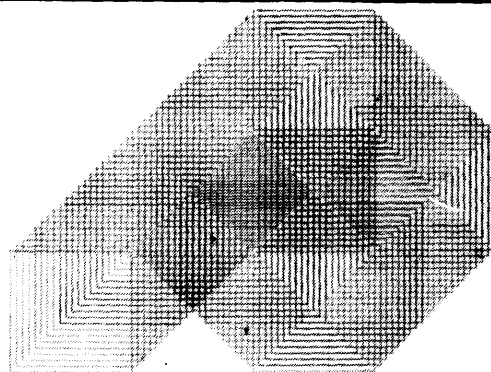
The program will then jump to **80F** and draw the second side of the diamond.

On the next pass, the register pair DE will be looking at locations **0864**, **0865**, **0866**, and **0867**. This will change locations **089A**, **089B**, **089D** and **089E** to **2D 34** and thus the third side of the diamond will be drawn.

Via the same reasoning, the 4th side of the diamond will be completed.

Try experimenting and changing this program to produce other patterns. We have included two examples on the right and will be continuing with these and more 'add-ons' in the next issue.

The aim of **COMPUTER GRAPHICS** is to be able to produce 'forms' and ruled work for invoices etc. and place information in correct locations. It also gives you an understanding of **ROBOT** movement, an area we all would like to investigate.



860 00 0	880 49 = I	88C 38 = 8	898 0D = CR
861 34 4	881 2C = ,	88D 30 = 0	899 4D = Move
862 2D -	882 44 = Draw	88E 2C = ,	89A 00 = 0
863 34 4	883 38 = 8	88F 30 = 0	89B 34 = 4
864 2D -	884 30 = 0	890 2C = ,	89C 2C = ,
865 34 4	885 2C = ,	891 2D = -	89D 00 = 0
866 2D -	886 30 = 0	892 38 = 8	89E 34 = 4
867 34 4	887 2C = ,	893 30 = 0	89F 0D = CR
868 2D -	888 38 = 8	894 2C = ,	8A0 FF = end
869 34 4	889 30 = 0	895 30 = 0	
86A 00 0	88A 2C = ,x	896 2C = ,	
86B 34 4	88B 2D = -	897 30 = 0	

The Direction Change table and Drawing table with decoded values.

MON-1B

LOOKING AT THE REGISTERS

The **MONITOR ROM** for the **TEC-1A** (it can also be fitted to the **TEC-1**) is a **MON-1B**. This ROM has the facility for looking at the registers.

This ROM is the result of a number of requests from readers who needed to look at the contents of the various registers during the running of a program.

If you would like one of these updated ROMs, send your **MON-1** or **MON-1A** plus \$3.00 postage and we will re-burn your EPROM to include the additional instructions.

There is a limit to when and where you can use the register facility but it can help enormously with debugging programs.

For instance, it can let you know the progress of a program or delay routine simply by interrupting it part way through.

The way this facility works is as follows:

If you reset the computer while it is executing a program, by pressing the reset button **ONCE**, the contents of each of the registers is pushed onto a stack.

This stack starts at **0FF0** and increases downwards to **0FD8**.

To look at any of the registers, press reset once and key the address of the register you want to look at.

The following list identifies the location of each register:

Mem ADD:	Reg:
0FF0	
0FEF	A
0FEE	F
0FED	B
0FEC	C
0FEB	D
0FEA	E
0FE9	H
0FE8	L
0FE7	IX MSB
0FE6	IX LSB
0FE5	IY MSB
0FE4	IY LSB
0FE3	A'
0FE2	F'
0FE1	B'
0FE0	C'
0FDF	D'
0FDE	E'
0FDD	H'
0FDC	L'
0FDB	I
0FDA	-
0FD9	Stack MSB
0FD8	Pointer LSB

Note: Reset clears the **I** register and thus it will always equal **00**.

Use **JP 0000** if you wish to look at **I**.

An alternate method of saving the registers is to insert a **JP 00 00** instruction in the program at the position you wish to investigate. This will cause a **JUMP** to the beginning of the **MONITOR ROM** where it will find a jump to the register-save routine.

This will enable you to exit a program at a pre-determined point and look at the registers. The contents will be shown in the data displays.

Pushing Reset twice will destroy the information.

This is the program at **05F0** which performs the 'REGISTER-SAVE' operation. Don't forget the Monitor ROM has an instruction at **0000** to Jump to **05F0**.

05F0	ED	LD (0FD8),SP
05F4	31	LD SP,0FF0
05F7	F5	PUSH AF
05F8	C5	PUSH BC
05F9	D5	PUSH DE
05FA	E5	PUSH HL
05FB	DD	PUSH IX
05FD	FD	PUSH IY
05FF	08	EX AF,AF'
0600	D9	EXX
0601	F5	PUSH AF
0602	C5	PUSH BC
0603	D5	PUSH DE
0604	E5	PUSH HL
0605	ED	LD A,I
0607	F5	PUSH AF
0608	C3	JP 0580
060B	FF	RST 38H

```

0760 0F      RRCA
0761 0F      RRCA
0762 0F      RRCA
0763 E6 0F   AND 0F
0765 21 B0 07 LD HL,07B0
0768 85      ADD A,L
0769 6F      LD L,A
076A 7E      LD A,(HL)
076B 02      LD (BC),A
076C 03      INC BC
076D C9      RET
076E 21 F0 0F LD HL,0FF0
0771 06 06   LD B,06
0773 0E 01   LD C,01
0775 7E      LD A,(HL)
0776 D3 02   OUT (02),A
0778 79      LD A,C
0779 D3 01   OUT (01),A
077B 0E 40   LD C,40
077D 0D      DEC C
077E 20 FD   JR NZ,077D
0780 07      RLCA
0781 4F      LD C,A
0782 AF      XOR A
0783 D3 01   OUT (01),A
0785 23      INC HL
0786 10 ED   DJNZ,0775
0788 C9      RET
0789 FF      RST 38
078A FF      RST 38
078B FF      RST 38
078C FF      RST 38
078D FF      RST 38
078E FF      RST 38
078F FF      RST 38
0790 D5      PUSH DE
0791 C5      PUSH BC
0792 CB 9A   RES 3,D
0794 1A      LD A,(DE)
0795 FE FF   CP FF
0797 20 09   JR NZ,07A2
0799 13      INC DE
079A 0B      DEC BC
079B 78      LD A,B
079C B1      OR C
079D 20 F5   JR NZ,0794
079F C1      POP BC
07A0 D1      POP DE
07A1 C9      RET
07A2 3E 47   LD A,47
07A4 D3 02   OUT (02),A
07A6 3E 01   LD A,01
07A8 D3 01   OUT (01),A
07AA 76      HALT
07AB 18 F2   JR 079F

```

```

07B0 EB 28 CD AD 2E A7 E7 29
EF 2F 6F E6 C3 EC C7 47

```

PRINT-2 AND PRINT-3 SOFTWARE

With the changes to the keyboard handler routines in both MON-2 and JMON, an up-dated printer ROM has been produced.

The new software is burnt into the same ROM at higher locations. When MON-2 was released, an up-dated ROM called print-2 was included in the printer interface kits. This gave you the same routines with an altered keyboard section. It was also a little more fancy as it showed the start address on the LED display as you typed it in. Unfortunately, Print-2 did not include a "dump string at 0900" routine to replace the dump from 0800 which is now unusable as MON-2 uses 0800 for its variable storage.

With the advent of JMON, the same arrangement has been used. The JMON printer routines are located higher again, so in the one ROM you have the printer software for all three MONitors. The list routine for JMON is an improvement on both earlier software packages, as JMON's routine uses the perimeter handler to allow you to enter both a START and END address. Print-3 includes a "dump from 0900" routine which can be used with MON-2.

The ROM with the JMON routines in it is called PRINT-3 and is supplied with the printer interface as standard.

JMON's hex dump routine is at 1A20, the typing routine at 1AA0 and the "dump string at 0900" routine is at 1AC0.

Below is a dump of PRINT-3. Burn the additional section(s) in PRINT-1/2 ROM.

The graphic demonstration routines in PRINT-1 will work with all MONitors.

```

1800 3E 0D D3 06 3E 0A D3 06
      76 ED 57 17 17 17 17 57
1810 CD 5D 18 76 ED 57 82 57
      CD 61 18 76 ED 57 17 17
1820 17 17 5F CD 5D 18 76 ED
      57 83 5F CD 61 18 C3 49
1830 18 3E 0D D3 06 3E 0A D3
      06 7A CD 5D 18 7A CD 61
1840 18 7B CD 5D 18 7B CD 61
      18 06 08 3E 20 D3 06 1A
1850 CD 5D 18 1A CD 61 18 13
      10 F1 C3 31 18 1F 1F 1F
1860 1F 21 6C 18 E6 0F 85 6F
      7E D3 06 C9 30 31 32 33
1870 34 35 36 37 38 39 41 42
      43 44 45 46 FF FF FF FF
1880 21 00 08 7E FE FF 20 05
      3E 11 D3 06 C7 D3 06 23
1890 18 F1 FF FF FF FF FF FF
      FF FF FF FF FF FF FF FF
18A0 21 C3 18 7E FE FF 28 05
      D3 06 23 18 F6 06 0A 21
18B0 CF 18 7E FE FF 28 05 D3
      06 23 18 F6 10 F1 3E 11
18C0 D3 06 C7 0D 0A 0A 0A 0A
      0A 0A 12 43 30 0D FF 49
18D0 2C 44 33 32 30 2C 30 0D
      4D 31 32 30 2C 30 0D 44
18E0 38 30 2C 2D 31 36 30 0D
      4D 32 32 30 2C 2D 38 30
18F0 0D 44 31 36 30 2C 2D 38
      30 2C 31 34 30 2C 2D 31

```

```

1900 36 30 2C 32 30 30 2C 2D
      31 36 30 0D 4D 31 35 30
1910 2C 2D 31 32 30 0D 44 32
      30 30 2C 2D 31 32 30 0D
1920 4D 33 32 30 2C 2D 38 30
      0D 44 32 36 30 2C 2D 38
1930 30 2C 32 34 30 2C 2D 31
      36 30 2C 33 30 30 2C 2D
1940 31 36 30 0D 4D 33 36 30
      2C 2D 31 32 30 0D 44 34
1950 30 30 2C 2D 31 32 30 0D
      4D 34 36 30 2C 2D 38 30
1960 0D 44 34 34 30 2C 2D 31
      36 30 0D 4D 32 2C 2D 32
1970 0D 43 33 0D FF FF FF FF
      FF FF FF FF FF FF FF FF
1980 76 ED 57 E6 0F 17 17 17
      17 57 76 ED 57 E6 0F 82
1990 D3 06 18 EC FF FF FF FF
      FF FF FF FF FF FF FF FF

```

The next block is the PRINT-2 additions:

```

19A0 76 3A E0 08 E6 0F 17 17
      17 17 57 76 3A E0 08 E6
19B0 0F 82 D3 06 18 EA FF FF
      FF FF FF FF FF FF FF FF
19C0 3E 0D D3 06 3E 0A D3 06
      3E 29 21 D8 08 06 06 77
19D0 23 10 FC CD 00 1A 32 D8
      08 CD 00 1A 32 D9 08 CD
19E0 00 1A 32 DA 08 CD 00 1A
      32 DB 08 CD D8 01 CD 89
19F0 02 50 59 C3 31 18 FF FF
      FF FF FF FF FF FF FF FF
1A00 3E FF 32 E0 08 CD A0 02
      3A E0 08 FE FF 28 F6 E6
1A10 0F C6 FF CD 70 01 D6 01
      C9 FF FF FF FF FF FF FF

```

Below is PRINT-3 additions:

```

1A20 21 34 1A 11 80 08 01 0A
      00 ED B0 21 00 00 22 9C
1A30 08 C3 44 00 00 00 3E 1A
      99 08 00 01 50 1A 04 A7
1A40 04 C7 04 EB FF FF FF FF
      FF FF FF FF FF FF FF FF
1A50 3E 0D D3 06 2A 98 08 7C
      CD 82 1A 7D CD 82 1A 06
1A60 08 C5 3E 20 D3 06 7E CD
      82 1A 23 C1 10 F3 3E 0D
1A70 D3 06 3E 0A D3 06 ED 5B
      9A 08 E5 B7 ED 52 E1 38
1A80 D6 C9 F5 0F FF 0F 0F CD
      8B 1A F1 E6 0F C6 90 27
1A90 CE 40 27 D3 06 C9 FF FF
      FF FF FF FF FF FF FF FF
1AA0 CF E6 0F 07 07 07 07 57
      CF E6 0F 82 D3 06 18 F0
1AB0 D3 06 18 EC FF FF FF FF
      FF FF FF FF FF FF FF FF
1AC0 21 00 09 7E FE FF 20 05
      3E 11 D3 06 C7 D3 06 23
1AD0 18 F1 FF FF FF FF FF FF
      FF FF FF FF FF FF FF FF

```

TE