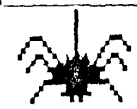

Screen Output using RST 10.

3.5

The ROM calls for screen output are all in the form of restart 10 calls. Following each of these calls is data which tells the ROM routine what to do. At first this is a little confusing as data is stored in the path of the program, but is in fact remarkable easy to use.

Writing ASCII characters to the Screen.

The following RST 10 routine passes the ascii representations of registers B and C to the screen.

```
START:  LD BC,"TM"
        RST 10
        DB 192
        RET
```

A less trivial example would be the following:

```
START:  LD E,8
        LD HL,DATA
LOOP:   LD A,(HL)
        LD B,0
        LD C,A
        RST 10
        DB 192 ; write BC token.
        DEC E
        JR NZ,LOOP
        RET
DATA:   DB " MEMOTECH"
```

Try changing the line that loads B with zero to load B with the space character and see what happens.

Sending Messages to the Screen.

To avoid the complication of the above routine we can send a complete string in the following way:

```
        RST 10
        DB £8D,"MEMOTECH LTD"
        RET
```

The byte following the RST 10 is made up in the following way:

RST 10 Control byte - Bit Format

7	6	5	4	3	2	1	0
1	0	C	<--------n------->				

Where bit 5 indicates that the routine should continue to interpret data after this instruction. n is the number of bytes in the string.

Virtual Screens and RST 10

The format for the virtual screen RST 10 instruction byte is:

RST 10 Control byte - Bit Format

7	6	5	4	3	2	1	0
0	1	c	*	cls	<---n--->		

Where c is the continuation bit, n the virtual screen to be selected and cls gives the option to clear the screen.

* = don't care. (It can asssume any value and has no operational effect when used in this mode).

One Byte Screen Write.

This RST 10 call allows the transfer of single bytes to the screen.
Its format is:

RST 10 Control byte - Bit Format

7	6	5	4	3	2	1	0
0	0	c	*	*	*	*	*

Where c is the continuation bit.

* = don't care. (It can asssume any value and has no operational effect when used in this mode).

The following is an example using RST 10 and CALL £79, keyboard input.

```
START:  CALL £79
        JR Z,START
        LD C,A
        LD B,0
        RST 10
        DB 192
        CP 13
        JR NZ,START
        RET
```

The routine reads the keyboard and echos the typed characters on the screen.

Control Codes and RST 10.

In the ASCII character set there are 32 invisible characters before the first printable character (space). These invisible characters are called control characters. For example pressing both the control key and the "G" key at the same time generates the bell code, character 7. These codes are extremly powerfull in the MTX when used with RST 10.

Try the following example:

```
START:  LD B,160
LOOP:   RST 10
        DB £86," *    "
        DJNZ LOOP
LOOP1:  RST 10
        DB £8B,15,0,"*",£10,£10,£38
        DB £54,£10,£10,£10,£38
```

```
          CALL PAUSE
          RST 10
          DB £8B,15,0,"*",0,0,£10,£54
          DB £38,£10,£7C,£44
          CALL PAUSE
          JP LOOP1
          RET
PAUSE:    LD B,50
PAUSE1:   HALT
          DJNZ PAUSE1
          RET
```

The program works by first printing up a series of "*" characters and then redefining them to give an animation effect.

The following is a list of commands avilable through RST 10:

ASCII code	Function
1	plot x,y
2	line x1,y1,x2,y2
3	cursor x,y
7	bell
10	line feed,cursor down
12	cls and home
11	vertical tab
13	carage return
14	ctlspr p,x
15	genpat p,n,d0,d1,d2,d3, d4,d5,d6,d7
16	colour p,n
17	adjspr p,n,v
18	sprite n,p,xp,yp,xs,ys,col
19	movspr p,n,d
20	view dir,dis
21	insert key
22	delete key
23	back tab
25	tab key
26	home key
27,65	attr p,state
27,89	crvs n,t,x,y,w,h,s
27,90	vs n
27,67	gr$ x,y,b (result in work space)

Printer Output.

All screen output can be redirected to either the RS232 or the centronics interface and hence the printer. To do this from basic type:

POKE 64143,DEV - Where DEV is 0 for screen
1 for Centronics