

M6809PLOT(D2)

EXORSET PLOT PACKAGE
REFERENCE MANUAL

The information in this document has been carefully checked and is believed to be entirely reliable. However, no responsibility is assumed for inaccuracies. Furthermore, such information does not convey to the purchaser of the product described any license under the patent rights of Motorola Inc. or others.

Motorola reserves the right to change specifications without notice.

EXORset, EXORdisk, and XDOS are trademarks of Motorola Inc.

Second Edition June 1982
Copyright 1982 by Motorola Inc.

TABLE OF CONTENTS

		Page
CHAPTER 1 -	INTRODUCTION	
1.1	Graphics Display Memory Definition	1-01
1.2	Arguments.	1-01
CHAPTER 2 -	USING PLOT WITH BASICM	
2.1	Calling PLOT from BASICM	2-01
2.1.1	Address Declaration.	2-01
2.1.2	Argument Types	2-02
CHAPTER 3 -	EXAMPLES	3-01
CHAPTER 4 -	COMMAND SUMMARY	4-01

1. INTRODUCTION

The PLOT program delivered on mini-diskette provides an easy means to drive the EXORset graphics display memory. This minimal package (520 bytes) contains a collection of very basic utilities and is both ROMable and position-independent.

The PLOT subroutines were designed to be easily called as external procedures from a BASIC-M user program, although they can be used in a different context. This document outlines the software interface to PLOT (entry parameters, arguments passing, ...etc).

1.1 GRAPHICS DISPLAY MEMORY DEFINITION

The EXORset graphics display memory is defined as a matrix of 256 rows by 320 dots, and occupies the memory space between \$4000 and \$7FFF. The dot positions (coordinates) with respect to the graphics image are as follows :

(0,255)	(319,255)
:	
Y-axis	
:	
:	
(0,0) ... X-axis ... (319,0)	

1.2 ARGUMENTS

Several PLOT functions support arguments which actually are values of coordinates in the cartesian chart defined above. The routines which operate on individual dots support two arguments (the X and Y coordinates of the dot), while those dealing with straight line generation support four arguments (the X and Y coordinates of the line extremities). The arguments must obey the following rules :

- A. The actual arguments must agree in number, order, and type with the formal arguments of the PLOT functions.
- B. The X and Y coordinates are 16-bit quantities which must be in the range defined in paragraph 2.
- C. Argument passing convention.

Upon entry in a parameterized PLOT subroutine, it is assumed that the MC6809 Y-register points to an

argument table that contains 16-bit pointers to the argument values; for instance, if one wishes to draw a straight line between (10,15) and (40,70), one could write the following :

:				
LDY	#TABLE	TABLE	FDB	X1,Y1,X2,Y2
JSR	VECTON	:		
:		X1	FDB	10
		Y1	FDB	15
		:		
		Y2	FDB	70
		:		
		X2	FDB	40

As is shown on this example, the argument addresses (pointers to the arguments values) must be contiguous in the argument table, whereas the argument values need not be adjacent.

This scheme was chosen because it conforms to the way BASIC-M handles argument passing (refer to BASIC-M User Guide - paragraph 10.3).

2. USING PLOT WITH BASIC-M

As was mentioned before, the various PLOT subroutines can be easily called from a BASIC-M user program as external procedures. However, due to the fact that the EXORset graphics memory occupies the memory space into which the BASIC-M compiler and runtime package get loaded, BASIC-M cannot be used interactively. Therefore the user is required to supply the necessary compile command options and relocation procedures so that neither the runtime package, nor the user program PSCT (user code section), nor the user program DSCT (data section), reside in the graphics memory space at execution time.

As a simple example, the commands and procedures illustrated below would build a software environment where the runtime package and program DSCT originate at \$800 and \$200, respectively. The user code originates at \$9BD2 and need not be relocated since it falls outside the graphics display memory.

```
READY
COMPILE R=$800, D=$200, M
```

```
      :
Symbol Table Printout
```

```
      :
DSCT : 0200-06AF
PSCT : 9BD2-9C13
```

```
READY
PATCH
```

```
.MV
BEG 0000 6500
END 0000 9BC0
DEST 0000 800
```

```
.9BD2;G
```

- enter the EXORbug monitor to relocate the runtime package to \$800 using the MOVE command.
- refer to the file BASCNEWS delivered on the BASICM diskette to read the runtime installation addresses.
- push the RESTART or ABORT buttons.
- Invoke program execution (in this particular example, the program section originates at \$9BD2, as is indicated at the end of the symbol table printout).

2.1 CALLING PLOT FROM BASIC-M

2.1.1 Address Declaration

The PLOT subroutines which are to be accessed from a BASIC-M program must be declared as EXTERNAL procedures and be assigned an origin via the ADDRESS declaration clause. For easy remembering, the useful entry points of PLOT have been grouped in a jump table at the very beginning of the package. The "as-delivered" base address of PLOT is \$C000 (first EROM socket available in the EXORset primary map); thanks to its position independence, the

package may be relocated elsewhere without requiring re-assembly.

Example : the following program declares three particular utilities in PLOT to switch on the EXORset graphics memory, to erase it, and to light on a dot at coordinates X and Y, respectively.

```

10  EXTERNAL GON ADDR $C003, ERASE ADDR $C000
20  EXT DOTON ADDRESS $C00F
    :
60  GON
70  ERASE
    :
95  IF X>30 AND X<50 THEN DOTON(X,Y)
    :
```

2.1.2 Arguments type

The PLOT subroutines which support arguments assume that these latter are all of the integer type; the user is therefore responsible for declaring explicitly the argument variables as such, and/or for insuring that the arithmetic expressions used as arguments yield an integer result (refer to BASIC-M User Guide - paragraph 4.4).

Invalid examples :

```

10  INTEGER X
    :
40  DOTON (X+2,$30)
```

\$30 is a valid integer constant, but ...
X+2 is an arithmetic expression that yields a real result.
Line 40 must be written : 40 DOTON(X+\$2,\$30)

```

10  INTEGER X,Y
    :
50  DOTON (X*SQR(Y),Y)
```

SQR is a real function. Program should be written :

10 INTEGER X,Y		10 INTEGER X,Y,Z
:	or	:
50 DOTON (X*FIX(SQR(Y)),Y)		40 Z=SQR(Y)
		50 DOTON (X*Z,Y)

3. EXAMPLES

Valid examples : Draw 8 concentric squares centered on (160,128).

```

100 EXT ERASE ADDR $C000, GON ADDR $C003, LINKON ADDR $C024
110 INTEGER X1,X2,Y1,Y2
120 X0 = 160           \ coordinates of center
130 Y0 = 128           \ of each square.
140 GON                \ switch on graphics memory.
150 ERASE              \ erase it.
160 FOR J=1 TO 8       \ loop until 8 squares drawn.
170 DIST=10*J          \ square number determines side length.
180 X1=X0-DIST
190 Y1=Y0+DIST
200 X2=X0+DIST
210 Y2=Y0-DIST
220 LINKON(X1,Y1,X2,Y1,X2,Y2,X1,Y2,X1,Y1) \ draw 4 sides
230 NEXT J

```

Plot the function $y = K * \sin(x)$ for
 $x = 0..4*PI$, $K = \{0.5, 0.75, 1\}$

```

100 EXT ERASE ADDR $C000, GON ADDR $C003
110 EXT AXES ADDR $C02D, DOTON ADDR $C00F
120 INTEGER X,Y
130 PI=3.14159265
140 FOUR_PI=4*PI
150 DELTA_TETA=PI/100
160 GON
170 ERASE
180 AXES(FIX(0),FIX(128)) \ Draw axes
190 FOR K=.5 TO 1 STEP .25
200 FOR TETA=0 TO FOUR_PI STEP DELTA_TETA
210 Y=128+K*127*SIN(TETA)
220 X=319*TETA/FOUR_PI
230 DOTON(X,Y)
240 NEXT TETA
250 NEXT K                \ Draw next curve

```

Draw all the straight lines that connect
 10 pairs of randomly defined coordinates.

```

100 EXT ERASE ADDR $C000, GON ADDR $C003
110 EXT VECTON ADDR $C018
120 INTEGER C(10,2) \ matrix of coordinates
130 FOR I=1 TO 10 \ generate random coordinates
140 C(I,1)=319*RND \ in the range 0-319 for x,
150 C(I,2)=255*RND \ 0-255 for y.
160 NEXT I
170 GON
180 ERASE
190 FOR I=1 TO 9

```

```
200  FOR J=I+1 TO 10
210  VECTON(C(I,1),C(I,2),C(J,1),C(J,2))
220  NEXT J
230  NEXT I
240  FOR I=1 TO 1000 \ delay
250  NEXT I
260  GOTO 130          \ loop for ever
```

4. COMMAND SUMMARY

SUBROUTINE	ADDRESS (*)	DESCRIPTION
ERASE	..00	Clear Graphics memory
GON	..03	Enable Graphics display
GOFF	..06	Disable Graphics display
AON	..09	Enable Alphanumeric display
AOFF	..0C	Disable Alphanumeric display
DOTON (X,Y)	..0F	Light on dot (X,Y)
DOTOFF (X,Y)	..12	Light off dot (X,Y)
DOTCOM (X,Y)	..15	Complement dot (X,Y)
VECTON (X1,Y1,X2,Y2)	..18	Light on vector (X1,Y1)-(X2,Y2)
VECOFF (X1,Y1,X2,Y2)	..1B	Erase vector (X1,Y1)-(X2,Y2)
VECCOM (X1,Y1,X2,Y2)	..1E	Complement vector (X1,Y1)-(X2,Y2)
CHCK (X,Y)	..21	Read state of dot (X,Y), 0 if cleared, 1 if set. Must be called as a function !!!
LINKON (X1,Y1,...,Xn,Yn)	..24	Light on segments (X1,Y1)-(X2,Y2), (X2,Y2)-(X3,Y3), ...
LINKOF (X1,Y1,...,Xn,Yn)	..27	Same as LINKON but segments are erased.
LINKCM (X1,Y1,...,Xn,Yn)	..2A	Same as LINKON but segments are complemented.
AXES (X,Y)	..2D	Draw horizontal axis (0,Y)-(319,Y), and vertical axis (X,0)-(X,255).
FILL (X,Y,DX,DY,PAT)	..30	Fill with pattern PAT the rectangular area based at X0, and Y. X0 is the closest multiple of 8 which is less than or equal to X. The horizontal and vertical sides of the rectangle are 8*DX dots, and DY dots, respectively.

(*) ".." denotes the most significant byte of PLOT base address
(base address defaults to \$C000).

PAGE 001 PLOT .SA:1 PLOT *** PLOT PACKAGE FOR EXORSET ***

```
00001          NAM      PLOT
00002          TTL      *** PLOT PACKAGE FOR EXORSET ***
00003          OPT      NOW,LLEN=120
00004
00005          * VERSION : 1.00
00006          * DATE   : APRIL 1, 1980
00007
00008
00009          *****
00010          *          THIS PACKAGE IS ROMABLE AND POSITION INDEPENDENT          *
00011          *
00012          * THE OBJECT DEFAULTS TO ORIGIN $C000. SHOULD YOU WISH TO          *
00013          * RELOCATE IT ELSEWHERE, USE THE XDOS DUMP COMMAND AS SHOWN          *
00014          * BELOW :
00015          *
00016          * =DUMP PLOT.LO
00017          * : R FFFF
00018          * : 78/MN,OP,MN,OP/      (M,N,O,P ARE HEX DIGITS)
00019          * : W
00020          * : Q
00021          * = (PLOT NOW STARTS AT ADDRESS MNOP)
00022          *****
00023
00024          4000      A SCREEN EQU      $4000      GRAPHICS MEMORY BASE ADDRESS
00025          F018      A OUTCH EQU      $F018      EXORBUG CONSOLE OUTPUT
00026
00027A C000          ORG      $C000
00028
00029          *=====
00030          *          PLOT PACKAGE JUMP TABLE          *
00031          *=====
00032A C000 16      0030 C033 ERASE LBRA .ERASE ERASE GRAPHICS MEMORY
00033A C003 16      003A C040 GON LBRA .GON ENABLE GRAPHICS DISPLAY
00034A C006 16      003A C043 GOFF LBRA .GOFF DISABLE GRAPHICS DISPLAY
00035A C009 16      003A C046 AON LBRA .AON ENABLE ALPHANUMERIC DISPLAY
00036A C00C 16      003A C049 AOFF LBRA .AOFF DISABLE ALPHANUMERIC DISPLAY
00037A C00F 16      0059 C06B DOTON LBRA .DOTON LIGHT ON A SPECIFIC DOT
00038A C012 16      005F C074 DOTOFF LBRA .DOTOF LIGHT OFF A SPECIFIC DOT
00039A C015 16      0066 C07E DOTCOM LBRA .DOTCM COMPLEMENT A SPECIFIC DOT
00040A C018 16      00A2 C0BD VECTON LBRA .VECON TRACE VECTOR
00041A C01B 16      00A4 C0C2 VECTOF LBRA .VECOF ERASE VECTOR
00042A C01E 16      00A6 C0C7 VECCOM LBRA .VECCM COMPLEMENT VECTOR
00043A C021 16      0032 C056 CHCK LBRA .CHCK TEST DOT STATE (REAL FUNCTION)
00044A C024 16      01B4 C1DB LINKON LBRA .LKON TRACE SET OF CONTIGUOUS VECTORS
00045A C027 16      01B6 C1E0 LINKOF LBRA .LKOF ERASE SET OF CONTIGUOUS VECTORS
00046A C02A 16      01B8 C1E5 LINKCM LBRA .LKCM COMPLEMENT SET OF CONTIGUOUS VECTORS
00047A C02D 16      0137 C167 AXES LBRA .AXES DRAW AXES
00048A C030 16      016F C1A2 FILL LBRA .FILL FILL RECTANGULAR AREA
00049          *=====
00050          *          ARGUMENTS, WHEN REQUIRED, MUST ALL BE INTEGERS !!!          *
00051          *=====
00052          *          U-STACK AND S-STACK MUST BE 30 BYTES DEEP EACH          *
00053          *=====
00054
```