ARMADILLO GRAPHICS

Preliminary

Consumer Group
Mail Station 5890
2301 N. University
Lubbock, Texas 79414

TEXAS INSTRUMENTS
INCORPORATED

Date: April 22, 1983
Version 1.0

TABLE of CONTENTS

Paragraph Title

SECTION 1 ARMADILLO GRAPHICS

SECTION 1

ARMADILLO GRAPHICS

1.1 Introduction

This section attempts to describe all the additions and changes between Armadillo BASIC and TI Extended BASIC graphic capabilities. Armadillo BASIC has added several graphics modes. These new modes affect the results of certain subprograms, such as: CALL CHAR, CALL COLOR, CALL HCHAR, CALL SCREEN, and CALL VCHAR. Four new graphics subprograms have also been added. These are: CALL DCOLOR, CALL DRAW, CALL DRAWTO, and CALL FILL.

1.2 Graphics Modes

Armadillo BASIC has six graphics modes. The format for selecting the desired mode is:

CALL GRAPHICS(mode)

The following modes are valid:

Mode	Description
1	Graphics I - A 32 column by 24 row grid of pattern positions.
2	Text - A 40 column by 24 row alphanumeric grid.
3	Split I - Split screen where the top 1/3 of the screen displays text, and the bottom 2/3's of the screen displays high resolution graphics.
4	Split II - Split screen where the top 2/3's of the screen displays high resolution graphics, and the bottom 1/3 of the screen displays text.
5	Graphics II - High resolution color graphics (256 x 192).

6 Graphics III - Low resolution color graphics (64 x 48 blocks).

The default graphics mode is mode 1, Graphics I, with left and right margins of 2, and top and bottom margins of 0. Whenever the graphics mode changes, the following occurs:

* The requested mode is invoked.
* The screen is cleared.
* The standard character set is assigned.
* The standard colors are assigned.
* All sprites are turned off.
* The screen margins are set to zero
* The current position for graphics subprograms is initialized to (1,1).
* The sound is turned off.

Changing graphics modes does not clear the memory or reset any values.

1.2.1 Graphics I Mode.

This is the default mode and is completely compatible with TI-99/4A Extended BASIC. Any program that ran under TI Extended BASIC should run in Graphics I Mode.

Each position in the 32 x 24 grid is one 8 x 8 pixel block. Alphanumeric and special characters can be used in this mode. Color can be assigned by character-set (one color per 8 characters) or by sprite number. None of the graphics subprograms can be used. There is a maximum of 32 sprites available.

1.2.2 Text Mode.

Text Mode uses the same character set as the Graphics I Mode, but each character is represented in a 6 x 8 pixel block rather than an 8 x 8 pixel block. This allows the screen to have 40 columns instead of 32.

Only two colors (foreground and background) can be used at a time. None of the graphics subprograms and no sprites can be used.

1.2.3 Split I Mode.

This mode has a split screen in which the top one-third (32 x 8 grid) displays text and the bottom two-thirds (256 x 128 grid) displays high resolution color graphics. The text portion is a column and row grid where each position is one character. The graphics portion is a dot-column and dot-row grid where each position is one pixel.

Color can be assigned by character-code to each character in the text portion of the screen, and to each 8 pixels in the graphics portion of the screen. All the graphics subprograms work in this mode, but the results from the ACCEPT, DISPLAY, INPUT, LINPUT, and PRINT statements are placed in the text portion of the screen and so are never visible in the graphics portion of the screen. Also, results from the HCHAR and VCHAR subprograms are always placed in the graphics portion of the screen. There is a maximum of 32 sprites available.

1.2.4 Split II Mode.

Split II Mode is the same as Split I Mode, except the top two-thirds of the screen displays high resolution color graphics and the bottom one-third displays text.

1.2.5 Graphics II Mode.

The entire screen (256 x 192 grid) displays high resolution color graphics. The screen is a dot-column and dot-row grid where each position is one pixel. Color can be assigned to each 8 pixels. All the graphics subprograms work in this mode, but the results from the ACCEPT, DISPLAY, INPUT, LINPUT, and PRINT statements are not visible. The HCHAR and VCHAR subprograms will place characters on the screen. There is a maximum of 32 sprites available.

1.2.6 Graphics III Mode.

The entire screen (64 x 48 blocks) displays low resolution color graphics. A block is a 4 x 4 pixel block. Each block is individually assigned a color. None of the graphics subprograms work in this mode, and the results from the ACCEPT, DISPLAY,

INPUT, LINPUT, and PRINT statements are not visible. Also, the results of the HCHAR and VCHAR subprograms are not visible on the screen. There is a maximum of 32 sprites available.

1.3 CALL MARGINS Subprogram

There is a new subprogram, CALL MARGINS, which defines the screen margins. The format is:

```
CALL MARGINS(left,right,top,bottom)
```

where left, right, top, and bottom are integers which represent the number of columns or rows to indent the screen margins. This means that all 32 columns and 24 rows can be used, or a smaller "window" can be defined as desired. There can only be one "window" defined at a time. TI Extended BASIC has only one "window" which can be defined as:

```
CALL MARGINS(2,2,0,0)
```

in Armadillo BASIC.

At power up time, mode 1 and margins of (2,2,0,0) are the defaults. The rest of the time the default margins are (0,0,0,0), unless a program executing in mode 5 or mode 6 ends or aborts. This causes the system to revert to mode 1 with margins of (2,2,0,0).

This subprogram affects the screen results of the following:

* Scrolling - Scrolls only the current window.
* CLEAR subprogram - Clears only the portion of the screen defined by the current margins.
* ACCEPT statement - The position (1,1) is the upper left-hand corner of the current window.
* DISPLAY statement - The position (1,1) is the upper left-hand corner of the current window.

This subprogram does not affect the screen results of the following:

* HCHAR subprogram
* VCHAR subprogram

* GCHAR subprogram

* Modes 3 and 4 - The graphics portion of the screen is not affected.

* Modes 5 and 6 - These modes are not affected at all.

1.4 Graphics Subprograms

There are four graphics subprograms which can only be used in modes 3, 4, and 5. The subprograms are: CALL DCOLOR, CALL DRAW, CALL DRAWTO, and CALL FILL. The following paragraphs describe the graphics subprograms.

1.4.1 CALL DCOLOR Subprogram.

The CALL DCOLOR subprogram specifies the color to use in the DRAW, FILL, HCHAR, and VCHAR subprograms. The format is:

```
CALL DCOLOR(foreground-color,background-color)
```

where the colors can be any of the 16 colors described in Extended BASIC. The default color is black on transparent.

1.4.2 CALL DRAW Subprogram.

The CALL DRAW subprogram draws a line connecting the two given points. The type of line to draw is specified in the CALL DRAW subprogram. The format is:

```
CALL DRAW(type,y1,x1,y2,x2[,y3,x3,y4,x4][,...])
```

where type is:
1 - For draw a line (using the colors specified in DCOLOR)
0 - For erase a line
-1 - For reverse .... more explanation ...
and where y is a dot-row position and x is a dot-column position.

1.4.3 CALL DRAWTO Subprogram.

The CALL DRAWTO subprogram draws a line from the current position to the point specified. The type of line to draw is specified in the CALL DRAWTO subprogram. The format is:

```
CALL DRAWTO(type,y1,x1[,y2,x2][,...])
```

where type is:
1 - For draw a line (using the colors specified in DCOLOR)
0 - For erase a line
-1 - For reverse .... more explanation ...
and where y is a dot-row position and x is a dot-column position.

If the current position has not been assigned (by previously calling the DRAW subprogram), the default position is (1,1).

1.4.4 CALL FILL Subprogram.

The CALL FILL subprogram fills all pixels surrounding and including the pixel defined by the specified dot-row and dot-column. The format is:

CALL FILL(dot-row,dot-column)

This subprogram fills the screen until a pixel of the foreground color or the screen edge is encountered. Filling may also stop if the figure is too complicated.

1.5 CALL COLOR Subprogram

The results and the format of the CALL COLOR subprogram is dependent on the current graphics mode. The format:

CALL COLOR(#sprite-number,foreground[,...])

is valid for all modes, except mode 2. Additional formats are:

Mode	Format
1	CALL COLOR(character-set,foreground,background[,...])
2	CALL COLOR results in an error.
3	CALL COLOR(character-code,foreground,background[,...])
4	CALL COLOR(character-code,foreground,background[,...])
5	CALL COLOR results in an error, unless for a sprite.
6	CALL COLOR(row,column,color[,...]) where row is 1 to 48 and column is 1 to 64.

Notice the distinction between character-set and character-code in modes 1, 3, and 4.

The Color Codes have remained the same as in TI Extended BASIC. The Color Codes are:

Color Code	Color
1	Transparent
2	Black
3	Medium Green
4	Light Green
5	Dark Blue
6	Light Blue
7	Dark Red
8	Cyan
9	Medium Red
10	Light Red
11	Dark Yellow
12	Light Yellow
13	Dark Green
14	Magenta
15	Gray
16	White

The Character Sets for mode 1 have been modified. All Extended BASIC characters still have the same Character Code, but there are now more characters available. The new Character Sets are:

Set Number		Character Codes	
29		0-7	
30		8-15	
31		16-23	
0		24-31	
1		32-39	
2		40-47	
3		48-55	
4		56-63	
5		64-71	
6		72-79	
7	Extended BASIC	80-87	
8		88-95	
9		96-103	99/4A Console BASIC
10		104-111	
11		112-119	
12		120-127	
13		128-135	
14		136-143	
15		144-151	
16		152-159	
17		160-167	
18		168-175	
19		176-183	
20		184-191	

21	192-199
22	200-207
23	208-215
24	216-223
25	224-231
26	232-239
27	240-247
28	248-255

1.6 CALL SCREEN Subprogram

The CALL SCREEN subprogram has been modified. The format is:

```
CALL SCREEN(background-color[,foreground-color])
```

This subprogram works in all six modes, but the foreground color is only effective in mode 2 (Text Mode). If the foreground color is not specified, the foreground color is not affected. The default color foreground color for mode 2 is black.