(specification)

CAD MEMO 4

PLOTTER PROGRAM FOR GENERATING MICROSTRIP ARTWORK ON HP7475A PLOTTER FROM APPLE][PLUS COMPUTER

Stowe Keller
March 7, 1986

I. INTRODUCTION

This report describes "STEP & REPEAT", an Applesoft BASIC microcomputer program intended to facilitate the generation of one or more microstrip circuit patterns or other patterns on the Hewlett-Packard 7475A plotter. The patterns are drawn on paper, mylar transparencies, or polyester film. The transparencies can be used directly as masks for photolithography, but usually the plotter output is photographically reduced in size to produce a mask on a glass plate or photographic film.

The necessary equipment includes: Apple][Plus or Apple //e microcomputer with disk drive, clock and CRT; HP7475A plotter with associated pens, paper and manuals; Apple serial (RS-232C) or HP-IB (IEEE-488) interface card and cable connected to plotter; and the STEP & REPEAT disk. The reader is expected to be familiar with operating the Apple computer, and at least some knowledge of Applesoft BASIC programming and program storage/retrieval under Apple DOS 3.3. In order to generate patterns and fully utilize the plotter, the user will have to learn the Hewlett-Packard Graphics Language (HP-GL), described in detail in the HP7475A "INTERFACING AND PROGRAMMING MANUAL".

The user is expected to place the circuit pattern description, using HP-GL commands and/or subroutine calls, into DATA statements at the end of the STEP & REPEAT program. STEP & REPEAT has subroutines for generating useful patterns, such as solid triangles and quadrilaterals, that HP-GL does not itself provide. When the program is run, the user specifies the paper size that is being used (A or B), the scaling factor, and the number and spacing of horizontal and vertical

repetitions of the circuit pattern on the page. The user is encouraged to use inexpensive paper and pens for debugging purposes, and to use polyester film, or other high-quality media, and drafting pens for the final artwork.

II. USING THE PROGRAM

With the STEP & REPEAT program, the user takes his microstrip design on paper and translates it into a sequence of HP-GL commands and stores them in DATA statements at the end of the STEP & REPEAT program. Although the HP Graphics Language provides a large set of powerful instructions, some patterns that arise in microstrip circuits are not directly available in HP-GL. Therefore, some of these are provided in the STEP & REPEAT program as invokable "subroutines" to which parameters may be passed. The circuit description can be commented by placing REMark statements at the end of each DATA statement, or on lines by themselves in between DATA lines. This ability is particularly useful when entering and debugging complex microstrips. After the circuit description has been entered into the program, the program may then be RUN. The user will be asked for the size of the paper being used (A or B), a scaling factor, and the offset horizontally and vertically from the origin at which to start the pattern output (this is how the user allocates room for a border or other graphics to be output outside the microstrip repetitions). Further questions ask how many vertical and horizontal repetitions of the pattern are required, along with the spacing (or "step") between them. If desired, the program can be saved to disk for future use.

In order to enter a microstrip pattern into the STEP & REPEAT program, the user should have the pattern already laid out on paper with all necessary measurements available. The user will need a working knowledge of the Hewlett-Packard Graphics Language (HP-GL), and at least a minimal knowledge of BASIC programming

(enough to be able to enter, modify and delete lines, produce listings, etc; the programming utility Program Line Editor is useful for modifying BASIC program lines). HP-GL is a powerful graphics language that consists of two-letter commands, most of which are followed by one or more numeric parameters, separated by commas and terminated with a semi-colon (";"). STEP & REPEAT makes use of some of these commands in order to prepare the plotter for a microstrip pattern: the plotter is initialized, the paper size is set, and the scale of the plotter, or "user units", is set up so as to treat one unit of distance as being equal to the scaling factor multiplied by 1 mil (1/1000 of an inch). For example, with a scaling factor of 10 (the most common value used for scaling), measurements that are entered in mils will correspond to hundredths of an inch on paper.

HP-GL offers commands to draw points, lines, circles, arcs, empty and solid rectangles and wedges, draw ASCII characters, change the pen velocity, and can output information (such as status or digitization) from the plotter back to the computer. (STEP & REPEAT has a subroutine to check the error condition of the plotter and notify the user if a problem has occurred; otherwise, these HP-GL output commands are inaccessible to STEP & REPEAT.) Many commands allow relative instead of absolute coordinates, making placement and movement of objects much easier during debugging. For a list of some of the HP-GL commands that can be used in STEP & REPEAT, see Table 1. Again, the reader is referred to the HP7475A "INTERFACING AND PROGRAMMING MANUAL" for a thorough discussion.

The process of translating a microstrip circuit on paper into STEP & REPEAT commands involves breaking the circuit down into simple graphics components such as lines and rectangles, which directly translate into HP-GL commands. The program treats the plotting area as the first quadrant of a Cartesian coordinate system, with the lower left corner of the pattern as the origin, i.e. (0,0), with X going

horizontally and Y going vertically. The plotting limits for size A paper are 1015 by 780, and the limits for size B are 1630 by 1015. (Note that the plotter draws on size B paper at a 90 degree angle from size A due to the mechanics of the paper transport, but the results are essentially the same.) As mentioned earlier, the user-specified scaling value controls the size of the plotter output. In most situations, distances will be entered in mils in the pattern description, and a scaling factor of ten will be entered at runtime to generate a plot that is ten times the scale of the actual circuit.

Some patterns in microstrip circuits cannot be easily described in HP-GL commands, such as thick arcs, solid triangles or solid quadrilaterals. To remedy this, STEP & REPEAT provides generalized subroutines to provide these and other capabilities. A subroutine is invoked by placing it on a DATA line by itself, in the format: "$SUBx,parameter1,...,parameterN" where x is a number 1 through 6, and the parameters are explicit numeric values (expressions and variables are not currently allowed). Some subroutines have no parameters, while one of them requires eight parameters. The subroutines, with their syntax and description, are listed in Table 2. In order for the arc, triangle and quadrilateral subroutines to work effectively, the user should specify the pen thickness with subroutine 1 before using any other subroutines. Specifying the pen thickness with the HP-GL "PT" command will not affect these subroutines.

The first few commands in a microstrip description will usually be Pen Select and Fill Type. Several HP-GL commands can be stored in a single DATA statement if they are separated by semi-colons, such as:

```
5000  DATA "SP1;FT1;PU0,0" : REM USE PEN 1,FILL TYPE 1,PEN UP AT 0,0
```

However, subroutine calls MUST occur in their own DATA statements and cannot be intermixed with HP-GL commands.

In many situations it will be desirable to output one or more graphics commands outside the loops that repeats the microstrip pattern across the page. For example, outputting a date stamp, or a border around the entire image, is something that might only occur once. STEP & REPEAT allows for this by requiring the use of the command "$LOOP" to terminate that part of the microstrip pattern which is to be repeated; any subsequent graphic statements will be output only after all microstrip repetitions have been plotted. The origin of the user's coordinates at this time will be located at the origin (i.e., lower left) of the very first pattern which was output; thus, anything to be positioned below or to the left of the microstrip patterns will require one or more negative coordinates. If no commands are needed outside the loop, then the following command should be "$END" (discussed next). (Note that the "$LOOP" command must appear in a statement by itself, and may only occur once in a pattern description.)

The last DATA statement in a microstrip description must be "$END" to inform the program that there are no more commands to be read. Any additional DATA statements will be ignored by the STEP & REPEAT program; this can be useful during debugging if the user wishes to debug the first part of the description and to temporarily "hide" the rest of the description from the program. After debugging, the "$END" can be moved to the actual end of the description. Subroutine 5 offers another way to debug major errors by fetching the plotter's error status; if an error has occurred, the program pauses and displays an error message and number in inverse on the screen and waits for a keypress before continuing. (If no error has occurred, i.e. error status = 0, then operation will continue uninterrupted.)

III. PROGRAM DESCRIPTION

STEP & REPEAT is written in Applesoft][BASIC and is designed to run under Apple DOS 3.3 on an Apple][Plus or Apple //e equipped with a clock card in slot 4 and a serial or HP-IB (IEEE-488) interface card in slot 2 connected to the plotter. Because of the way the Apple is designed to send and receive characters with interface cards, only one "device" may be active at a time. When the plotter interface is active, almost anything printed by the computer will be sent to the plotter, including error messages. To prevent this, STEP & REPEAT has an ONERR GOTO statement that will trap program errors, disconnect the plotter, generate an error message and in all but one case will terminate program execution. (The exception is "?BAD RESPONSE TO INPUT", which generates a warning message to re-enter and resumes program execution.) This prevents the plotter from entering an error condition due to an Applesoft error message, and keeps the computer from accidentally receiving garbage input from the plotter.

Because of the difference between serial and HP-IB interface protocols, the program must know which one is currently installed. The first program line in STEP & REPEAT is an assignment statement that specifies the interface: if the variable HP is assigned the value 0, then a serial interface card is assumed; a non-zero value, corresponding to the plotter's device number, will specify the HP-IB interface. The program's title message will automatically state which interface has been assigned.

At runtime, the program is designed to ask the user for the paper size to be plotted on, a horizontal and vertical offset from the plotter's physical plotting origin, and the number of times the pattern is to be output in both horizontal and vertical directions. If a pattern is to be output more than once in a given direction, the program asks for the spacing (or "step") from the origin of one pattern to the next. The program does a simple check to see if the repetitions

will fit on the paper size that was specified; if not, an error message will inform the user and the number of repetitions and spacing for that direction must be re-entered. The program then reads the date from the clock card in slot 4 into the variable DT$; this is the string used by the date stamp subroutine (#6).

The interface is then enabled and an interface initialization string is sent to the plotter, as described in the HP 7475A "OPERATION AND INTERCONNECTION MANUAL". The plotter is sent commands to initialize the plotter itself, set the paper size as specified by the user, program the scale of the user units given the specified scaling factor, and set the character size to .15cm wide and .2cm high. The program then enters two nested FOR-NEXT loops: the first is for every pattern to be output vertically, and the inner loop is for every pattern to be output horizontally. The program then sends an "IP" command to position the plotting area on the paper based on the origin offset, step, and repetition number for the current pattern. (This is the mechanism by which the microstrip pattern can be entered once into the program but may be repeated across the page without having to change any coordinates in the pattern description.) The BASIC DATA statement pointer is initialized with the RESTORE statement, and another loop, formed by IF-THEN and GOTO, is entered. This innermost loop reads DATA statements into the variable S$, examines each string and takes one of three possible actions: If S$ = "$LOOP", then this loop is exited and the program advances to the next output of the pattern (if any). If S$ starts with the characters "$SUB", then it is treated as a special STEP & REPEAT subroutine call (discussed in detail later). If S$ does not match either of the above two conditions, it is sent unmodified to the plotter, followed by a ";" as a statement terminator. A GOTO statement causes the loop to resume at the statement which reads S$.

Exiting this innermost, DATA reading loop causes execution of the "NEXT" statements that terminate the horizontal and vertical repetition loops. After these loops are finished, a new loop is entered to process any commands that might occur outside the repeated pattern description, such as a date stamp or border draw. As soon as the "$END" command has been found, this loop terminates and the program instructs the plotter to park the pen in the pen stall, disconnect the interface to the plotter and END the program. (Lack of an "$END" will cause a fatal error when the program tries to read from non-existent DATA statements.)

When the loop that reads the DATA statements encounters a command starting with "$SUB", that string (stored in S$) is sent to a parameter parsing subroutine to determine which subroutine number is desired, as well as to extract any parameters that may have been specified. (Parameters are separated by commas.) Upon exiting the parsing subroutine, the specified subroutine number is stored in the variable P1, and any additional parameters are stored in P2,P3...P8,P9. P1 is then used as the index for an ON-GOSUB statement to transfer control to the correct subroutine. Each subroutine can then use the P2...P9 parameters as necessary for its algorithm.

The Applesoft BASIC error handler occurs next; as described earlier, it is designed to prevent error messages from accidentally being sent to the plotter. The interface to the plotter is immediately disconnected in the error handler, and an error message in inverse text is sent to the screen. If the error was due to an improper response to an INPUT statement, which is not a fatal error, a "?REENTER" message is displayed and the program tries again with the RESUME statement. Otherwise, program execution terminates to allow debugging the program.

The first STEP & REPEAT subroutine takes one parameter (in millimeters), converts it to real-world values given the scaling factor, and stores it in global variable PT. This value is used by the arc, triangle and quadrilateral routines for accurate area filling.

Subroutine 2 takes six parameters and uses them to draw a thick arc. The algorithm converts angles in degrees to radians, and determines how many arcs must be drawn, given the pen thickness, in order to achieve the desired arc thickness. The arcs are drawn in alternating directions so as to keep the pen down during the entire plot; trigonometry is used to position the pen at the ends of the arc. The inner and outer radius of the arc is calculated from the specified radius by adding or subtracting the desired arc thickness, with some adjustment for the pen thickness. The pen is placed in the up position before exiting the subroutine.

The third subroutine takes six parameters as specifying three coordinate pairs for the vertices of a triangle to be filled. The current algorithm resequences the points in such a way so as to have the two longest sides first, and divides those two sides up into an equal number of points. Lines are drawn between corresponding points of the two sides, bidirectionally, from the shortest side of the triangle to the point where the two longest sides intersect. This has proved to be a mediocre algorithm, and is intended to be reworked in the future. (Compensating for pen thickness is a remarkably difficult problem with triangles.)

Subroutine 4 takes eight parameters as the 4 points of a quadrilateral to be filled. The algorithm simply takes the first 3 points and invokes the triangle algorithm, and the last 3 points and invokes the triangle algorithm again. As a result of this, it is necessary to specify the points in a way that defines a convex polygon in order for this subroutine to correctly draw a quadrilateral.

Subroutine 5 is unique in that it is the only part of STEP & REPEAT that is designed to accept data from the plotter. It instructs the plotter to output its error status (with the "OE" instruction) and uses an INPUT statement to receive a one-character number. If the number is zero, the subroutine exits with no action

taken; if non-zero, it means an error has occurred and the interface is temporarily disconnected while the user is informed of the error. The message appears with the error number in inverse text, and waits for a keypress from the user. (See Table 2 for a list of plotter error messages.) After being acknowledged, the interface is reconnected, the subroutine is exited and the program continues. (Note: the current version of STEP & REPEAT does not support this routine with the HP-IB interface.)

The sixth subroutine is the date stamp subroutine, which takes the date, as was read into DT$ at the beginning of the program from the clock card, and sends it to the plotter via the label ("LB") command. The date stamp will start at the current pen location, and the character size is normally .15 cm wide and .2 cm high, unless modified by the user.

Subroutines 7 through 9 are reserved for future expansion. If the user wishes, he may incorporate subroutines of his own, starting with 10, provided that the ON-GOSUB statement holds the correct line numbers of the new subroutines. To retrieve parameters, the user should use the parameter parsing routine discussed above, and the variables P2 through P9 will contain whatever values were specified in the subroutine call.

The very last part of the program contains the DATA statements for storing the microstrip circuit pattern description. Double quotes should be used to enclose the graphics commands in the DATA statements; REMark statements can and should be used to document the design as well as to describe which commands pertain to what parts of the circuit. The first three HP-GL commands are usually Select Pen ("SP"), Fill Type ("FT"), and Velocity Select ("VS"); immediately following the pattern description should be the command "$LOOP", followed by any graphics

commands that are to be output only once. The last DATA statement must be "$END" to inform STEP & REPEAT that there are no more graphics commands to be processed.

IV. SUMMARY

The STEP & REPEAT program for the HP7475A plotter described in this paper was written to facilitate the design and output of microstrip circuit patterns, with the ability to to repeat the pattern at specified intervals on a single page. This eliminates the need for cutting and pasting artwork should the user wish to create several printed circuits at one time. The program offers several features to aid in specifying and debugging the circuit pattern layout, and scales the output under user control. Multiple pattern output is easily controlled at runtime, and the pattern and program can be saved to disk for future use.

V. ACKNOWLEDGMENTS

The author would like to thank Dr. Larry D'Addario and Dr. Sandy Weinreb for their supervision and advice during this project.

VI. REFERENCES

1. HP7475A Graphics Plotter Interfacing and Programming Manual, Hewlett-Packard, 1983.
2. HP7475A Operation and Interconnection Manual, Hewlett-Packard, 1983.
3. HP Plotter Notes, "Hewlett-Packard Short Body Drafting Pens", H-P Application Note 229-7, January 1985.

TABLE 1. Some Useful Hewlett-Packard Graphics Language Commands

MNEMONIC	NAME	DESCRIPTION
AA	Arc Absolute	Draw arc using absolute coordinates
AR	Arc Relative	Draw arc using relative coordinates
CA	Character set/ Alternate	Specify the alternate character set
CI	Circle	Draw circle
CP	Character Plot	Move pen specified number of spaces & lines
CS	Character set/ Standard	Specify the standard character set.
DF	Default	Return plotter to default condition
DI	Direction	Set absolute direction of labels
DR	Direction Relative	Set relative direction of labels
DT	Define Terminator	Defines the label terminator character
EA	Edge rectangle/ Absolute	Draw edge of rectangle using absolute coordinates
ER	Edge rectangle/ Relative	Draw edge of rectangle using relative coordinates
EW	Edge Wedge	Draw edge of wedge
FT	Fill Type	Specify type of area fill for rectangles and wedges
IM	Input Mask	Set masks for filtering out error conditions
IN	Initialize	Sets defaults, clears errors, etc. This is normally the first instruction sent.

IP	Input P1 & P2	Sets scaling points. Used by STEP & REPEAT to move and pattern across page.
IW	Input Window	Specify plotting window
LB	Label	Draws subsequent string of ASCII characters up until terimator character.
LT	Line Type	Specify pattern to use while drawing lines.

(HP-GL has several Output and digitization commands but they are not accessible from the current version of STEP & REPEAT.)

PA	Plot Absolute	Maintains pen up/down status while plotting to specified absolute X,Y points
PD	Pen Down	Set pen down
PR	Plot Relative	Maintains pen up/down status while plotting to specified relative X,Y points
PS	Paper Size	Specify paper size (eg, A or B)
PT	Pen Thickness	Specify pen thickness for use in area fill
PU	Pen Up	Set pen up
RA	shade Rectangle Absolute	Draw solid rectangle at absolute coordinates
RO	Rotate coordinates	Rotates coordinate system 90 degrees
RR	shade Rectangle Relative	Draw solid rectangle at relative coordinates
SA	Select Alternate character set	Select alternate character set as set by CA
SC	Scale	Scales plotting area into user units

SI	absolute character Size	Sets character width and height in centimeters
SL	character Slant	Specify slant for labeled charactera
SM	Symbol Mode	Specify character to draw at plotted points
SP	Select Pen	Specify which pen to use
SR	character Size/ Relative	Specify relative character width and height
SS	Select Standard character set	Select standard character set designated by CS
TL	Tick Length	Specify tick length for XT and YT
UC	User defined Character set	Draw user defined symbols
VS	Velocity Select	Specify pen velocity
WG	shade Wedge	Draw solid wedge
XT	X-Tick	Draw vertical tick mark
YT	Y-Tick	Draw horizontal tick mark

TABLE 2. Special Subroutines Available in STEP & REPEAT:

SUBRTN	SYNTAX	DESCRIPTION
1	$SUB1	Specify pen thickness in mm for use in other subroutines
2	$SUB2,x,y,radius,thickness,start angle,sweep angle	Draw a thick arc using the specified parameters: x,y : center of circle from which arc is to be taken radius : radius of circle; thickness expands inward and outward from radius thickness : thickness of arc start angle : angle in degrees at which to start drawing arc (0=east,90=north) sweep angle : angle in degrees of the sweep of the arc (positive=clockwise, negative=counterc/w)
3	$SUB3,x1,y1,x2,y2,x3,y3	Draw a solid triangle at the location specified by the three points
4	$SUB4,x1,y1,x2,y2,x3,y3,x4,y4	Draw a solid quadrilateral at the location specified by the four points. Points must be in clockwise or counter clockwise direction so as to form a convex polygon or algorithm will not work properly.
5	$SUB5	Fetches error condition from plotter and notifies user with on-screen message if error has occurred. Possible errors: 1= Bad instruction 2= Wrong # of parameters 3= Parameter out of range [4 is unused] 5=Unknown character set 6=Position overflow [7 is unused] 8=Pinch wheels not down
6	$SUB6	Outputs date stamp on plot at current pen position. Date and time taken from internal clock card.