THE

REVISION a.0
8-15-83

BAA
ENGINEERING GUIDE

Being a

☆ COMPENDIUM ☆

Delineating ALL Information required to efficiently utilize the

BAA PROGRAM

* * * INCLUDING * * *

INTERESTING, EDIFYING, AND INFORMATIVE
~ SUNDRY PARTICULARS ~
Relating to its

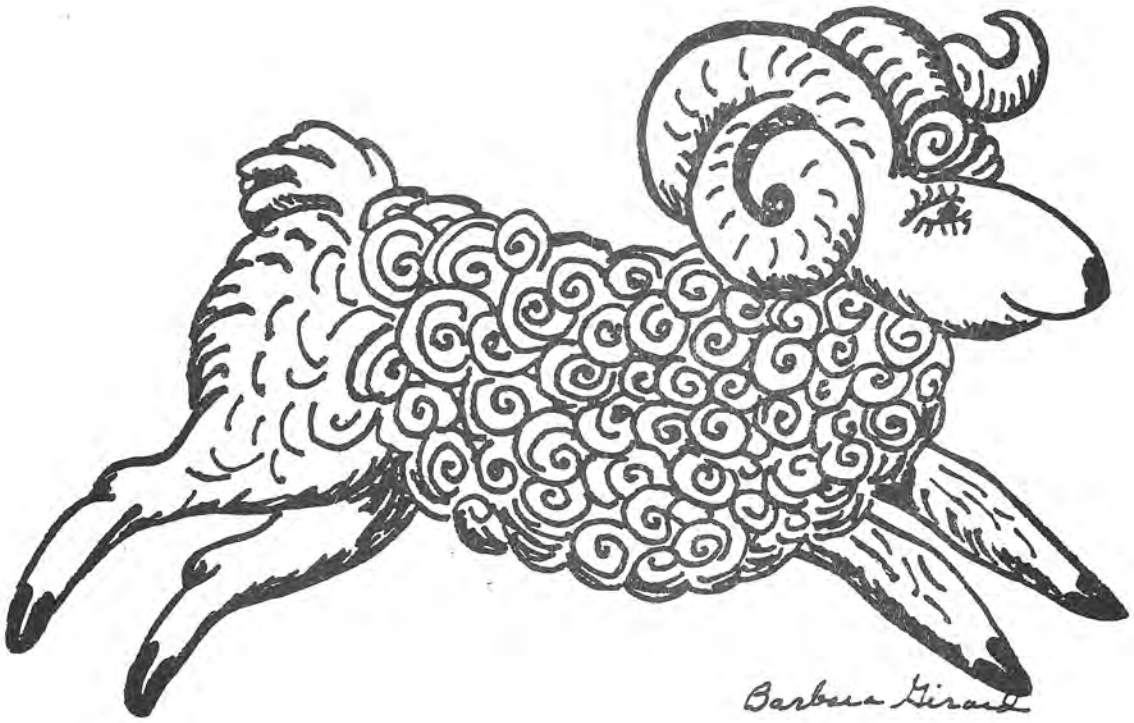
IMPLEMENTATION

SO AS TO EASILY ACHIEVE ANY DESIRED RESULT

- and containing -
Highly useful and Entertaining

FACTS, MUSINGS, AND EXAMPLES
&
→ FULLY ILLUSTRATED ←
with
Artistic, illuminating, hand drawn sketches
DRAUGHTED
PERSONALLY BY THE AUTHORS

{ THIS IS THE OFFICIAL DEPT. 7645 MANUAL. BEWARE OF SUBSTITUTES OR IMITATIONS! NIHIL OBSTAT. IMPRIMATUR. }


Barbara Girard

A NOTE TO THE READER:

THIS MANUAL MADE COMPLETELY*

BY BAA

* (I suppose, in the strict sence, I should not say entirely because, actually, not every little bit was made by using BAA, but a good deal of it was, to the extent that one might say most of it, and whew I say most I mean at the very least some of it or perhaps at least a little part, in fact I think there may be one piece on page ...., well, OK, none of this manual was made using BAA but we talked about it a lot, that is, sort of thought about it, anyway someone did mention it once, I think, although actually the thought just occurred to me and I don't feel it's a very good idea.)**

** I am truly sorry about this page and feel obligated to request that you completely ignore it as it serves no useful purpose although, completly may be too strong a word since you would have to at least glance at it somewhat in order to find this out to your own satisfaction, and, in fact, in order to do this properly you should read most of it because you might omit to notice some small bit that could have significance to you personally and, furthermore, since you would not know that you should do this until you have read this far - be sure to read this entire page.

TABLE OF CONTENTS

INTRODUCTION

BAA (BEDFORD AUTOMATED ARTWORK) IS A PROGRAM TO SIMPLIFY GERBER GENERATION OF MICROWAVE STRIPLINE OR MICROSTRIP ARTWORK. IT DOES THIS BY PROVIDING A USER ORIENTED INTERFACE BETWEEN THE DESIGNER AND RADAC. AN IMPORTANT ADVANTAGE OF BAA IS THAT IT IS MORE SYMBOLIC THAN RADAC. THIS NOT ONLY ELIMINATES MANY HOURS OF BORING, TEDIOUS, AND SLEEP INDUCING COMPUTATIONS, BUT FACILITATES CIRCUIT ALTERATIONS TO SUCH AN EXTENT THAT THE PROGRAM IS CONVENIENT TO USE EVEN IN THE INITIAL LAYOUT STAGES.

THIS MANUAL DESCRIBES AND EXPLAINS BAA. IT IS DIVIDED INTO FOUR SECTIONS CORRESPONDING FUNCTIONALLY TO HOW BAA IS USED.

SECTION I: COMPONENT LIBRARY - THIS IS A CATALOG OF COMMONLY USED CIRCUIT COMPONENTS (ALONG WITH SOME BASIC "BUILDING BLOCKS" TO FORM UNLISTED OR SPECIAL ITEMS). THE USER PICKS THE DESIRED COMPONENTS, DIMENSIONS THEM AS REQUIRED, AND PLACES THEM AT THE APPROPRIATE CIRCUIT LOCATIONS.

SECTION II: COMPONENT INTERCONNECTIONS - THIS SECTION EXPLAINS HOW TO DEFINE CIRCUIT LINES IN ORDER TO ELECTRICALLY CONNECT THE COMPONENTS SPECIFIED IN SECTION I.

SECTION III USER DEFINED ELEMENTS. THIS SECTION EXPLAINS HOW USERS CAN GENERATE THEIR OWN BAA-LIKE ELEMENTS USING CURRENT BAA COMMANDS.

SECTION II COMMANDS - THIS SECTION DESCRIBES THE ANCILLARY AND SPECIAL PURPOSE COMMANDS NECESSARY TO PRODUCE THE COMPLETE USABLE ARTWORK PROGRAM.

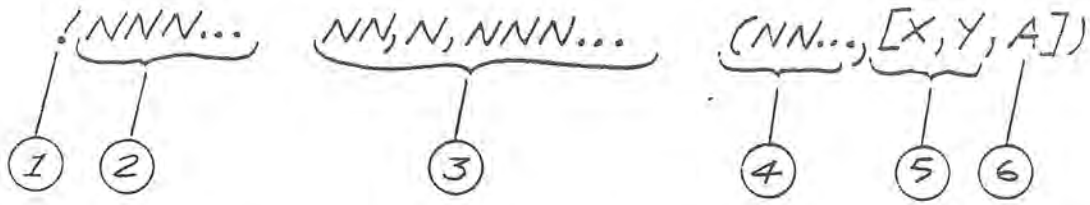
IN ADDITION, THE APPENDICES CONTAIN INFORMATION REQUIRED TO IMPLEMENT BAA ON THE CDC COMPUTER

* * * * * * * * * * * * * *

SECTION I

COMPONENT LIBRARY

THE FOLLOWING PAGES DESCRIBE ALL THE COMPONENTS AND DEVICES CURRENTLY INVENTORIED IN THE BAA LIBRARY. THE GENERAL USE FORMAT FOR THESE CATALOG ITEMS IS THE BAA STATEMENT SHOWN BELOW:



WHERE N = APPROPRIATE ALPHANUMERIC CHARACTER

1 = BAA DESIGNATOR

2 = COMPONENT GENERIC NAME – AS FOUND IN THE LIBRARY.

3 = COMPONENT PARAMETERS – USER SPECIFIED NUMERIC VALUES FOR THE DEVICE VARIABLES.

4 = COMPONENT IDENTIFIER – USER SPECIFIED NAME FOR THE SPECIFIC COMPONENT DEFINED BY THE NUMERIC VALUES OF ITEMS 3, 5, AND 6

5 = COMPONENT LOCATION – THE DESIRED CARTESIAN COORDINATE POSITION FOR THE DEVICE NAMED IN ITEM 4.

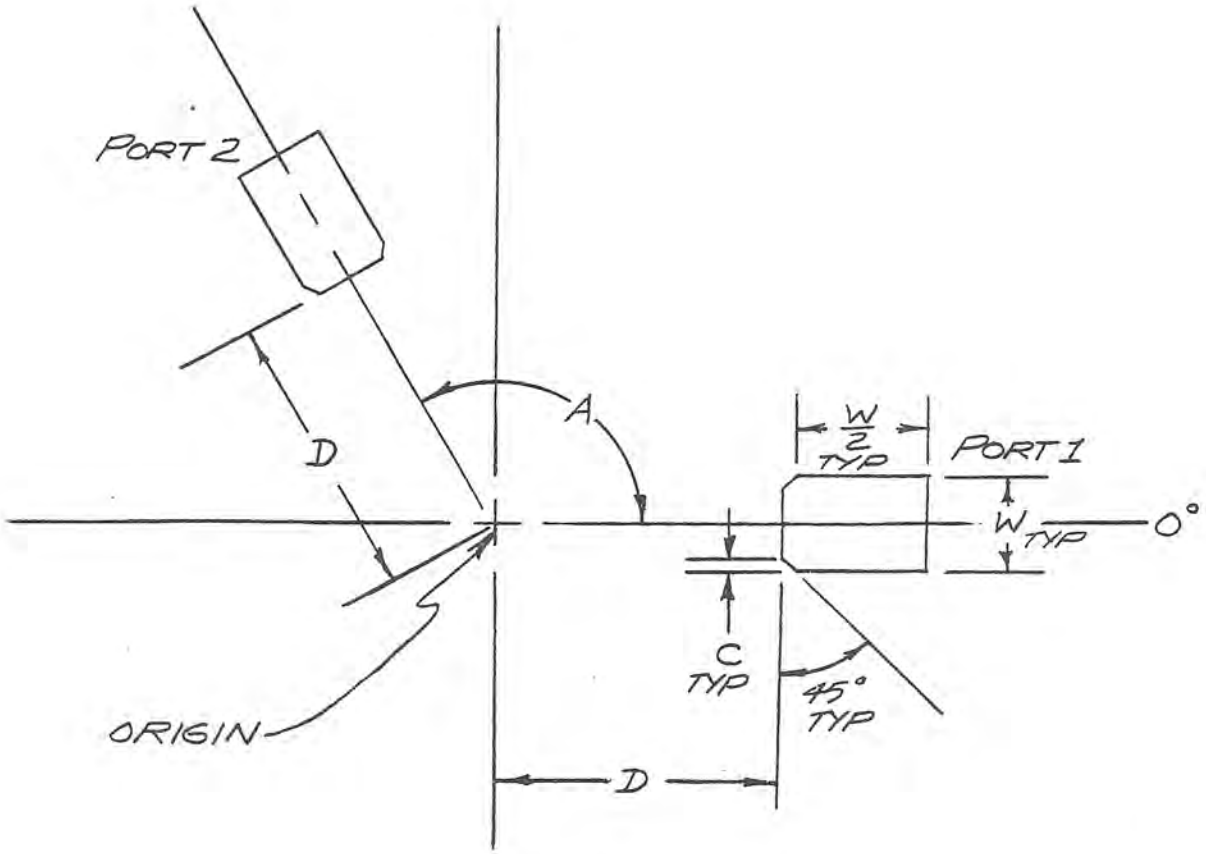
6 = COMPONENT ORIENTATION – THE DESIRED ANGULAR ORIENTATION FOR THE DEVICE NAMED IN ITEM 4.

THE CATALOG DEFINES THE ALPHANUMERIC CHARACTERS AND PARAMETERS FOR ITEMS 1 THRU 3. ITEMS 4 THRU 6 ARE NOT PRESENTED SINCE THEIR FORM AND REQUIREMENTS ARE THE SAME FOR EACH DEVICE.

PORTED COMPONENTS

BREAK 1

LINE INTERRUPTION

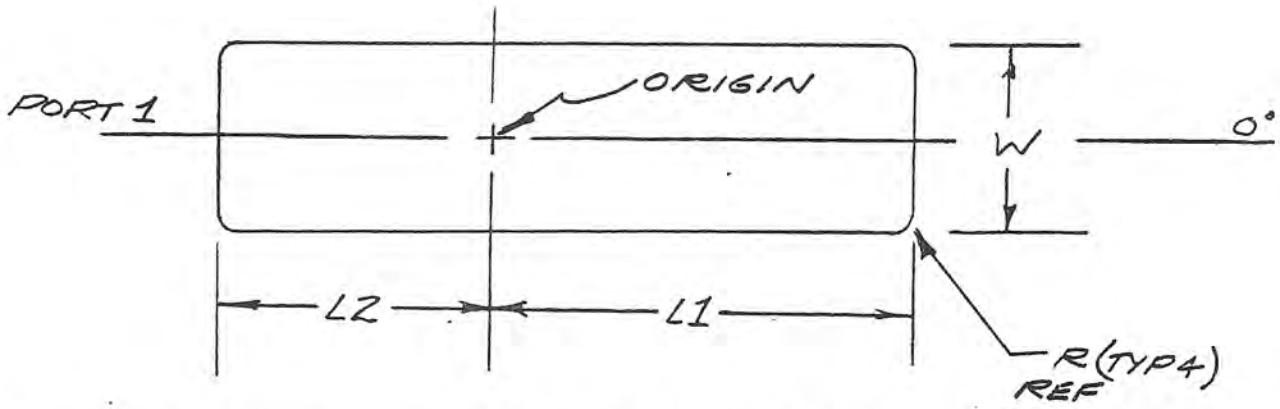


FORM: ! BREAK 1 W, C, D, A (NAME[X, Y, A])

WHERE W = LINE WIDTH
C = CHAMFER DIMENSION
D = DISTANCE FROM ORIGIN
A = ANGLE BETWEEN LINE CENTER LINES

CBOX1

BOX WITH A PORT



FORM: !CBOX1 W,L1,L2 (NAME[X,Y,A])

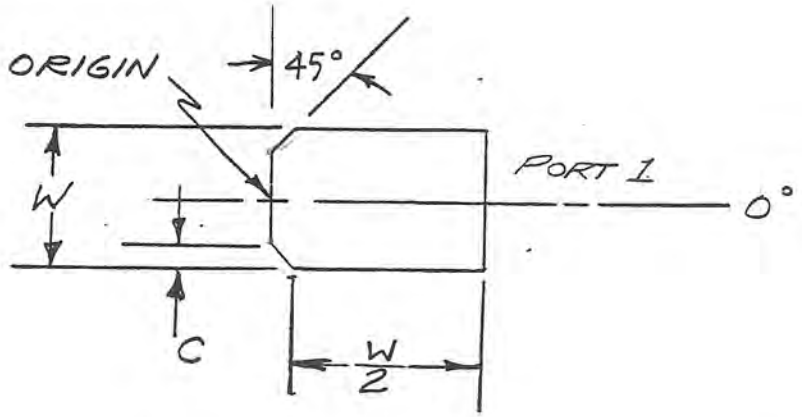
WHERE W = BOX WIDTH

L1, L2 } LENGTH OF BOX SEGEMENTS

NOTE: R PRESET TO .005. TO ALTER R USE BAPER COMMAND.

CHAMFER

CHAMFERED LINE TERMINATION

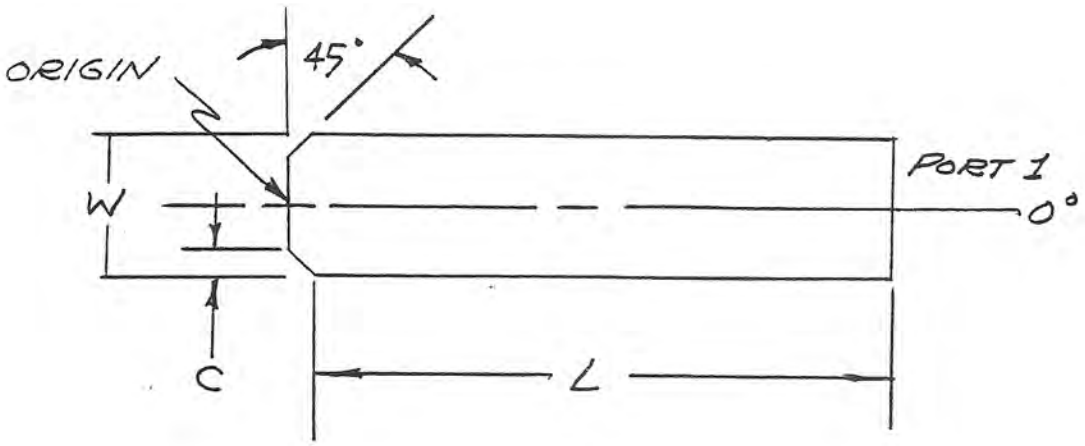


FORM: !CHAMFER W, C (NAME[X, Y, A])

WHERE W = LINE WIDTH
C = CHAMFER DIMENSION

CHAMFERL

STRAIGHT LINE WITH A CHAMFERED END

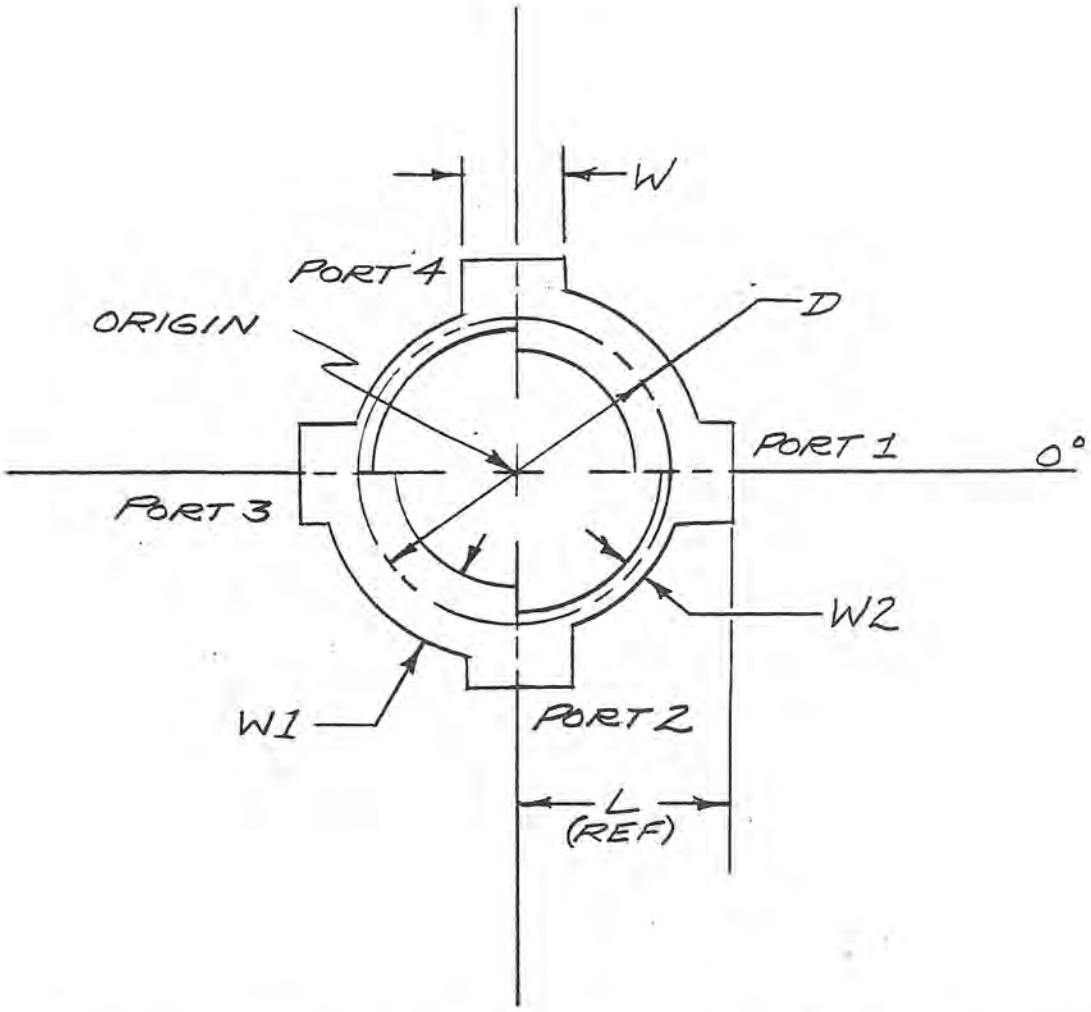


FORM: !CHAMFERL W, C, L (NAME[X, Y, A])

WHERE W = LINE WIDTH
C = CHAMFER DIMENSION
L = LINE LENGTH

CIRBLC

CIRCULAR BRANCH LINE COUPLER



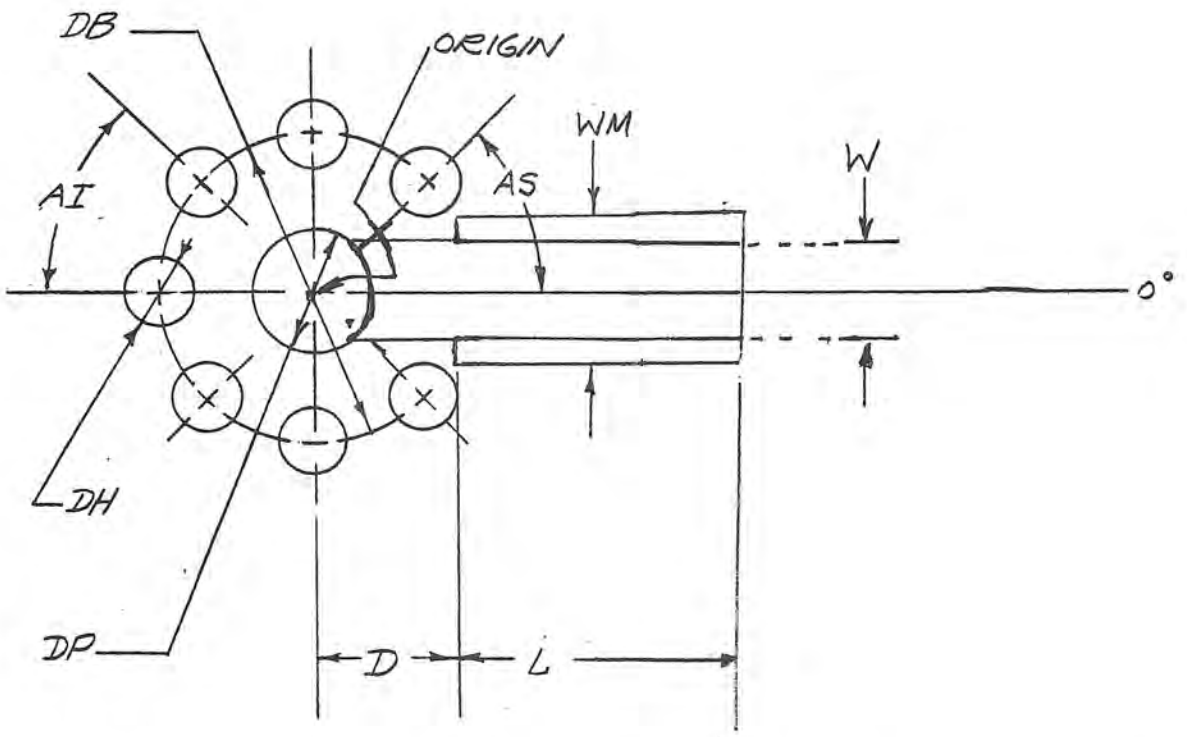
FORM: !CIRBLC W, W1, W2, D (NAME[X, Y, A])

WHERE W = LINE WIDTH
W1 = LINE WIDTH FOR INDICATED SEGMENTS
W2 = LINE WIDTH FOR INDICATED SEGMENTS
D = MEAN DIAMETER

NOTE: W1 MAY BE > OR < W2
$L = (W+D)/2 + .005 - W3/2$ (W3 = LESSER OF W1, W2)

CONN 1

LINE TERMINATION FOR SURFACE LAUNCH CONNECTOR

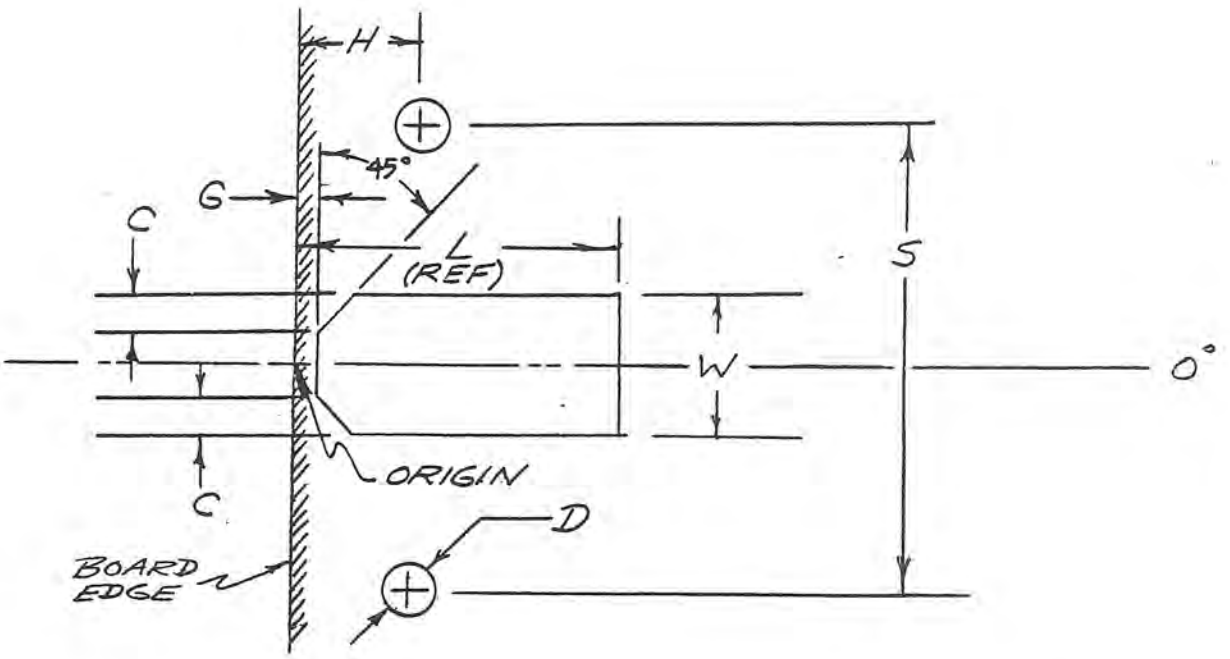


FORM: !CONN 1 W, DP, DB, DH, AS, AI, PN, D, L, WM (NAME[X, Y,

WHERE W = LINE WIDTH
DP = PAD DIAMETER
DB = BOLT HOLE DIAMETER
DH = HOLE PAD DIAMETER
AS = HOLE PATTERN START ANGLE
AI = HOLE PATTERN INCREMENT ANGLE
PN = NUMBER OF HOLES
D = DISTANCE TO EDGE OF MATCHING LINE
L = LENGTH OF MATCHING LINE
WM = WIDTH OF MATCHING LINE

CONN2

LINE TERMINATION FOR EDGE LAUNCH CONNECTOR



FORM: !CONN2 W, C, G, S, H, D (NAME[X, Y, A])

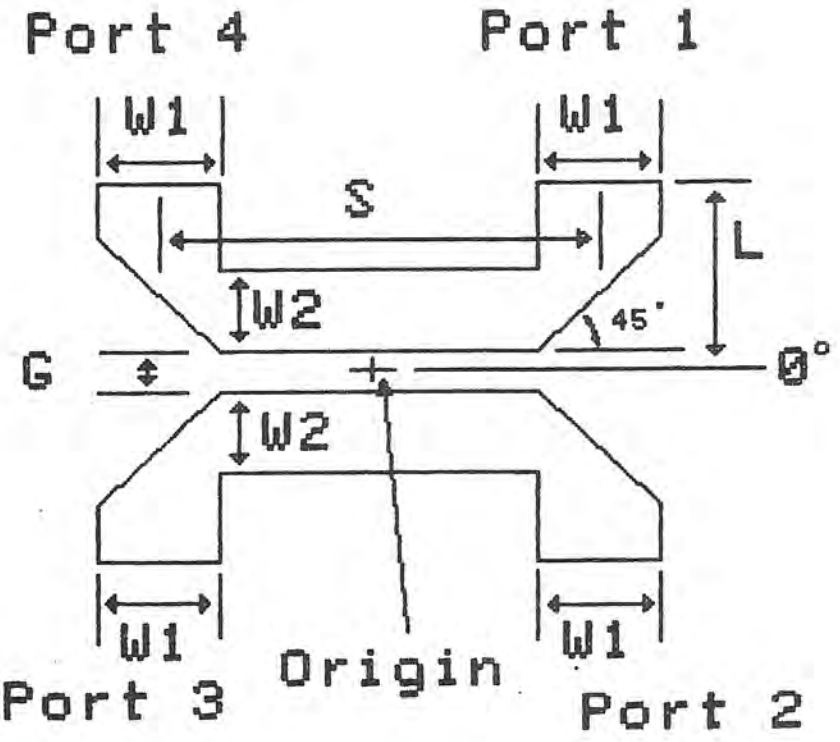
WHERE W = LINE WIDTH
C = CHAMFER DIMENSION
G = GAP BETWEEN END OF LINE AND BOARD EDGE.
S = DISTANCE BETWEEN CONNECTOR HOLES.
H = DISTANCE FROM CONNECTOR HOLE TO BOARD EDGE.
D = DIAMETER OF CONNECTOR HOLE PAD

NOTES:

$L = G + C + W/2$

CUPLR1

EDGE LINE COUPLER



FORM : !CUPLR1 W1,W2,S,G(NAME[X,Y,A])

WHERE W1 = LINE WIDTH OF INPUT AND OUTPUT PORTS
W2 = LINE WIDTH OF COUPLING REGION
S = SPACING BETWEEN ARMS
G = COUPLING GAP

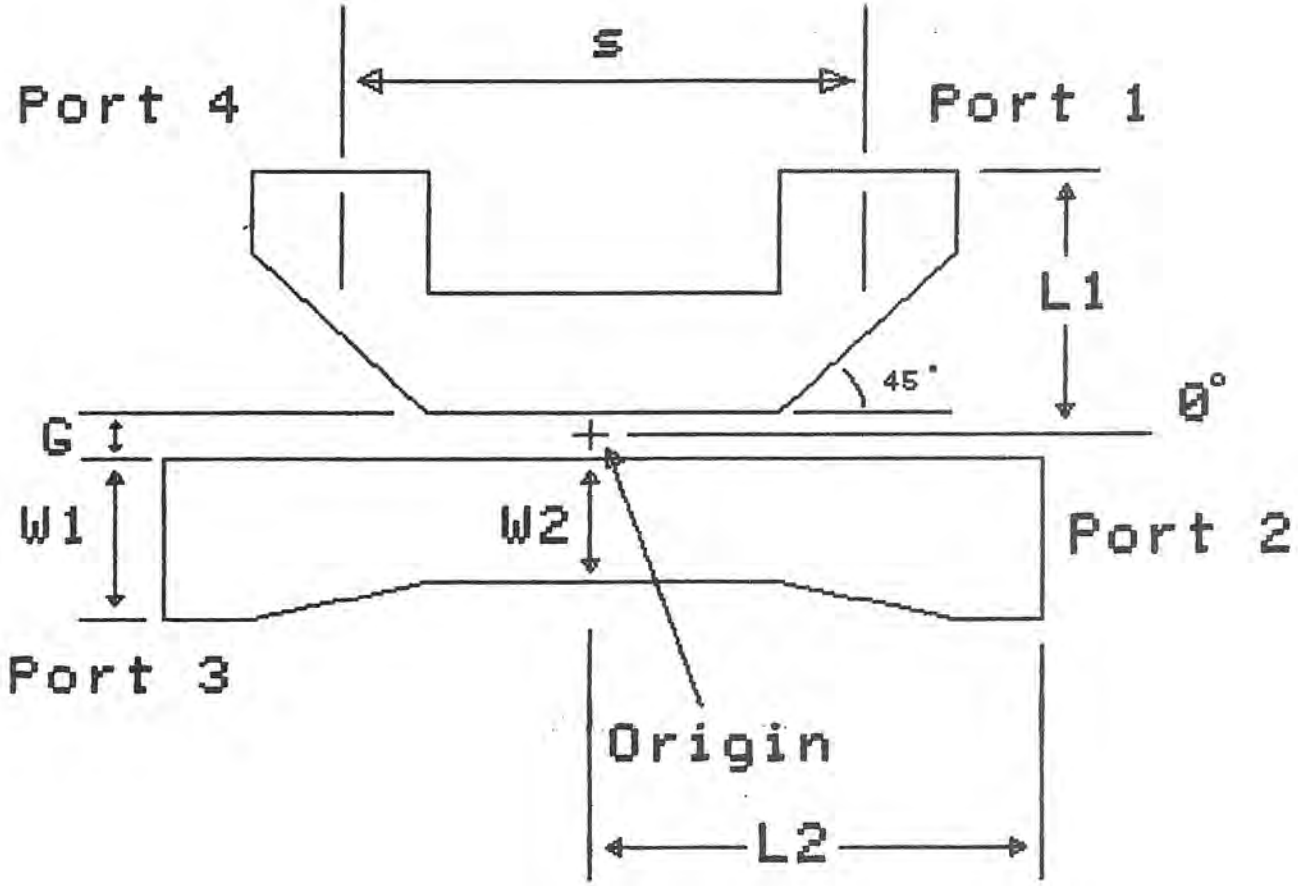
NOTE : L = G/2+1.5W1

EXAMPLE : THE ILLUSTRATION ABOVE WAS DRAWN BY THE FOLLOWING PROGRAM:

```
!NAME CUPLR1
!TARGET .01[-3,-3,0][-3,3,0][3,-3,0][3,3,0]
!CUPLR1 .7,.5,2.5,.25(NAME[0,0,0])
!END
!STOP
```

CUPLR2

EDGE LINE COUPLER



FORM : !CUPLR2 W2,W2,S,G(NAME[X,Y,A])

WHERE W1 = LINE WIDTH OF INPUT AND OUTPUT PORTS
W2 = LINE WIDTH OF COUPLING REGION
S = SPACING BETWEEN ARMS
G = COUPLING GAP

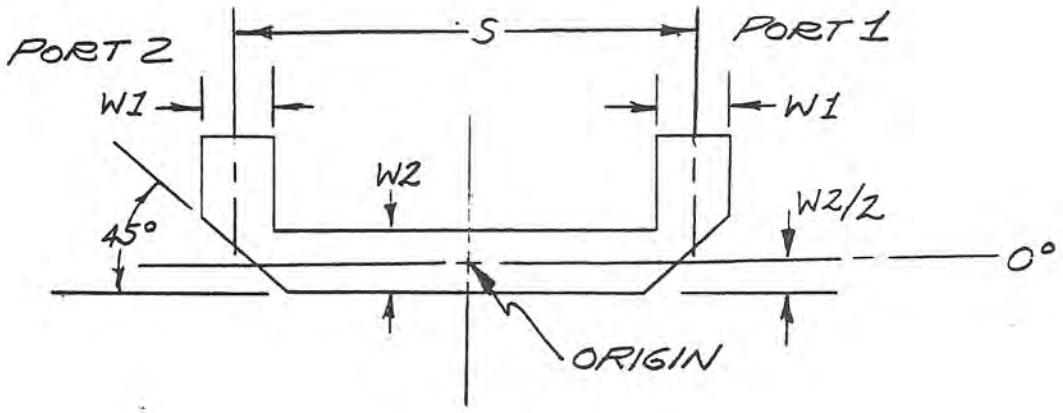
NOTE : L1 = G/2+1.5W1
L2 = S/2+W1

EXAMPLE : THE ILLUSTRATION ABOVE WAS DRAWN BY THE FOLLOWING PROGRAM:

```
!NAME CUPLR2
!TARGET .01[-3,-3,0][-3,3,0][3,-3,0][3,3,0]
!CUPLR2 .7,.5,2.5,.25(NAME[0,0,0])
!END
!STOP
```

CUPLRHALF

ONE HALF OF EDGE COUPLER



FORM: !CUPLRHALF W1,W2,S (NAME[X,Y,A])

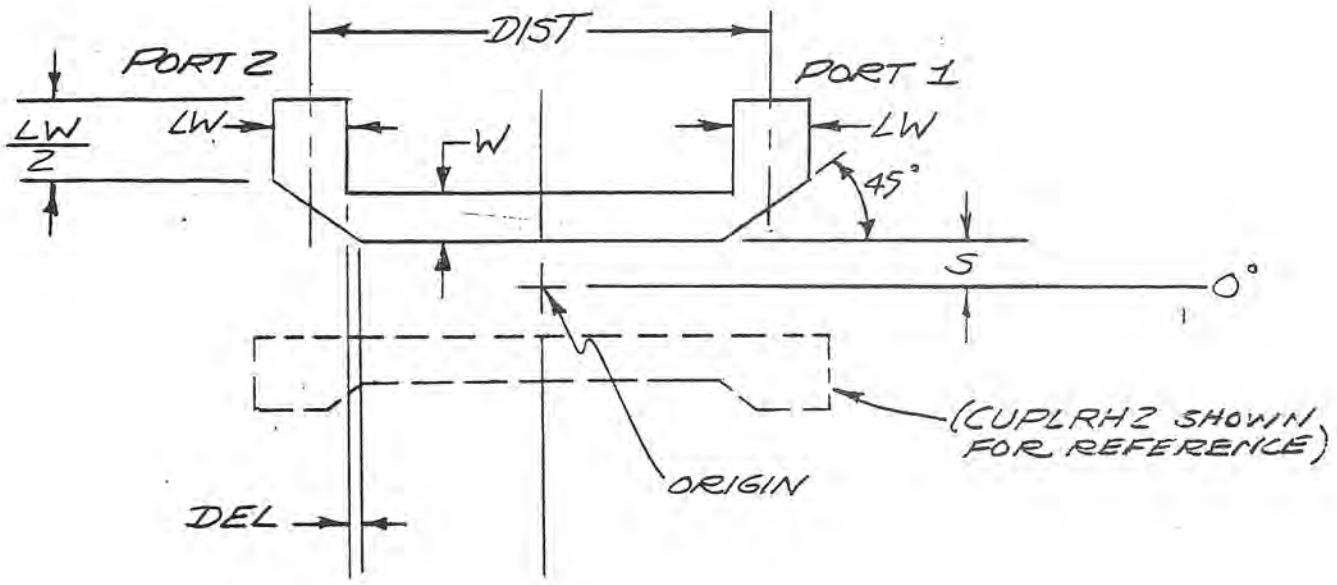
WHERE W1 = LINE WIDTH OF INPUT AND OUTPUT PORTS

W2 = LINE WIDTH OF COUPLING REGION

S = SPACING BETWEEN ARMS

CUPLRH1

ONE HALF OF OVERLAY COUPLER

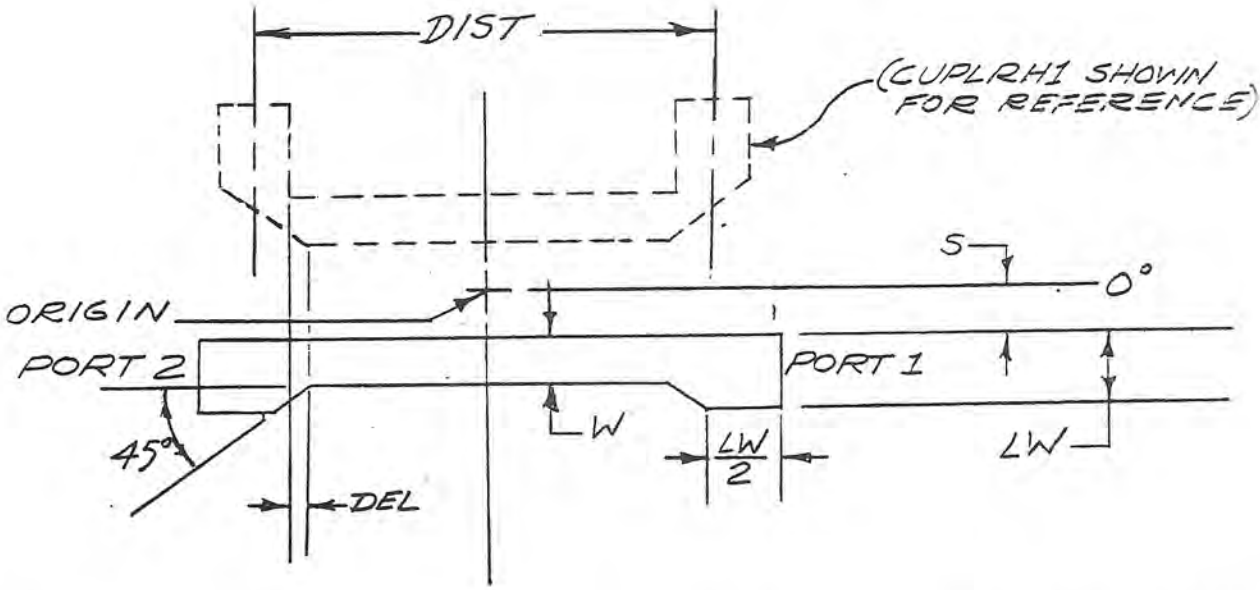


FORM: !CUPLRH1 LW, W, DIST, S, DEL (NAME[X,Y,A],

WHERE LW = LINE WIDTH OF INPUT & OUTPUT ARMS
W = LINE WIDTH OF COUPLING SECTION
DIST = ARM SPACING (℄ TO ℄)
S = ONE HALF COUPLER GAP. MAY BE POSITIVE (AS SHOWN IN FIGURE) OR NEGATIVE. A NEGATIVE S PRODUCES AN OVERLAP.
DEL = MITERED CORNER DIMENSION. MAY BE POSITIVE (AS SHOWN) OR NEGATIVE. A TYPICAL VALUE IS +0.14W.

CUPLRH2

ONE HALF OF OVERLAY COUPLER

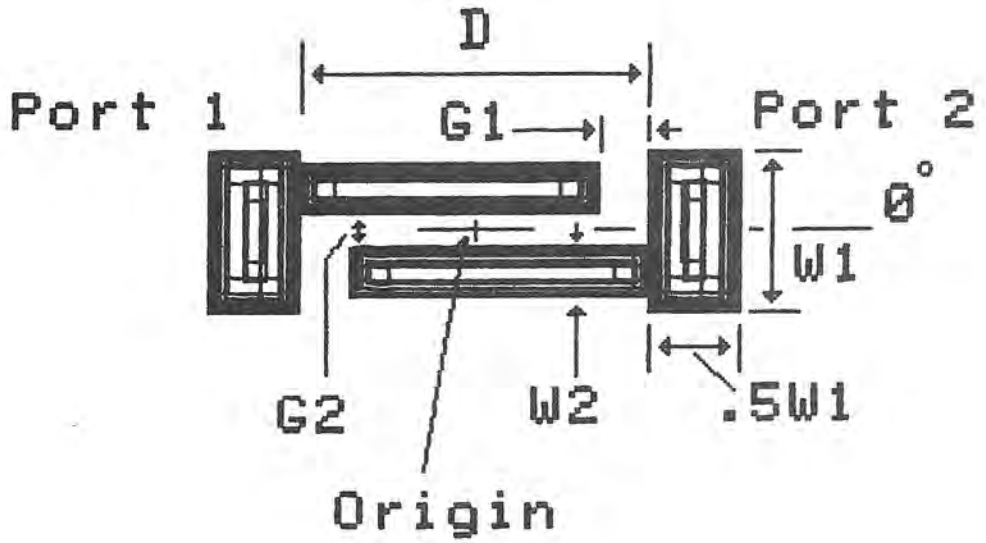


FORM: ! CUPLRH2 LW, W, DIST, S, DEL (NAME[X,Y,A

WHERE LW = LINE WIDTH OF INPUT & OUTPUT ARMS
W = LINE WIDTH OF COUPLING SECTION
DIST = ARM SPACING (℄ TO ℄)
S = ONE HALF OF COUPLER GAP. MAY BE POSITIVE (AS SHOWN IN FIGURE) OR NEGATIVE. A NEGATIVE S PRODUCES AN OVERLAP.
DEL = MITERED CORNER DIMENSION. MAY BE POSITIVE (AS SHOWN) OR NEGATIVE. A TYPICAL VALUE IS +0.14W.

DCBLOCK

DC BLOCK - NO CHAMFER



FORM : !DCBLOCK W1 W2 G2 G1 D(NAME[X Y A])

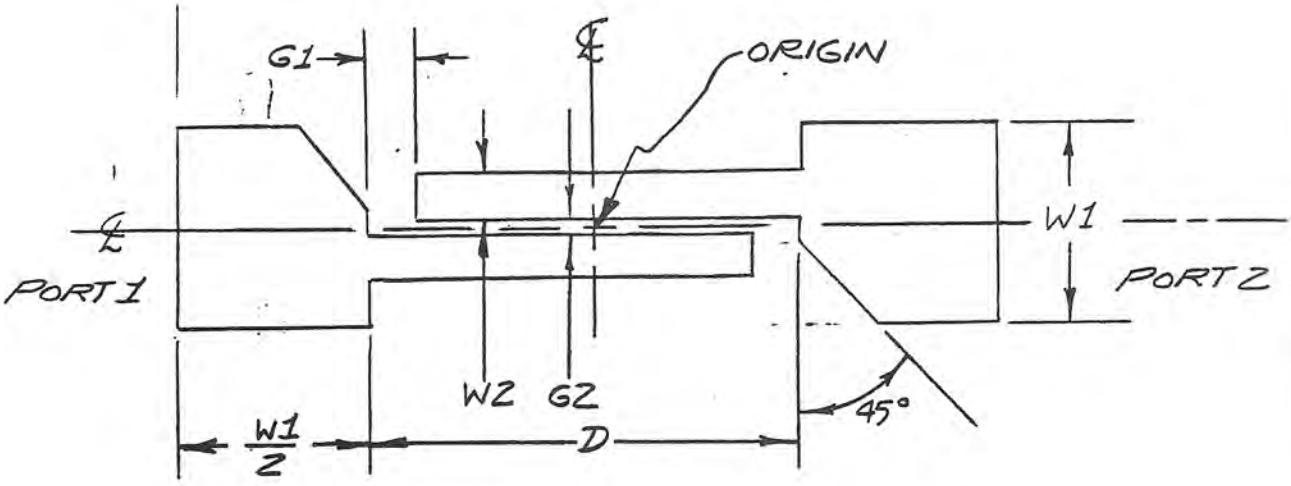
WHERE
- W1 = INPUT AND OUTPUT LINE WIDTH
- W2 = FINGER LINE WIDTH
- G2 = GAP BETWEEN FINGERS
- G1 = GAP BETWEEN FINGER AND LINE
- D = DISTANCE BETWEEN LINES

EXAMPLE : THE ILLUSTRATION ABOVE WAS DRAWN BY THE FOLLOWING PROGRAM:

```
!NAME DCBLOCK
!TARGET .01[-3,-3,0][-3,3,0][3,-3,0][3,3,0]
!DCBLOCK 1.0 .3 .22 .3 2.0(NAME[0 0 0])
!END
!STOP
```

DCBLK1

DC BLOCK WITH CHAMFER

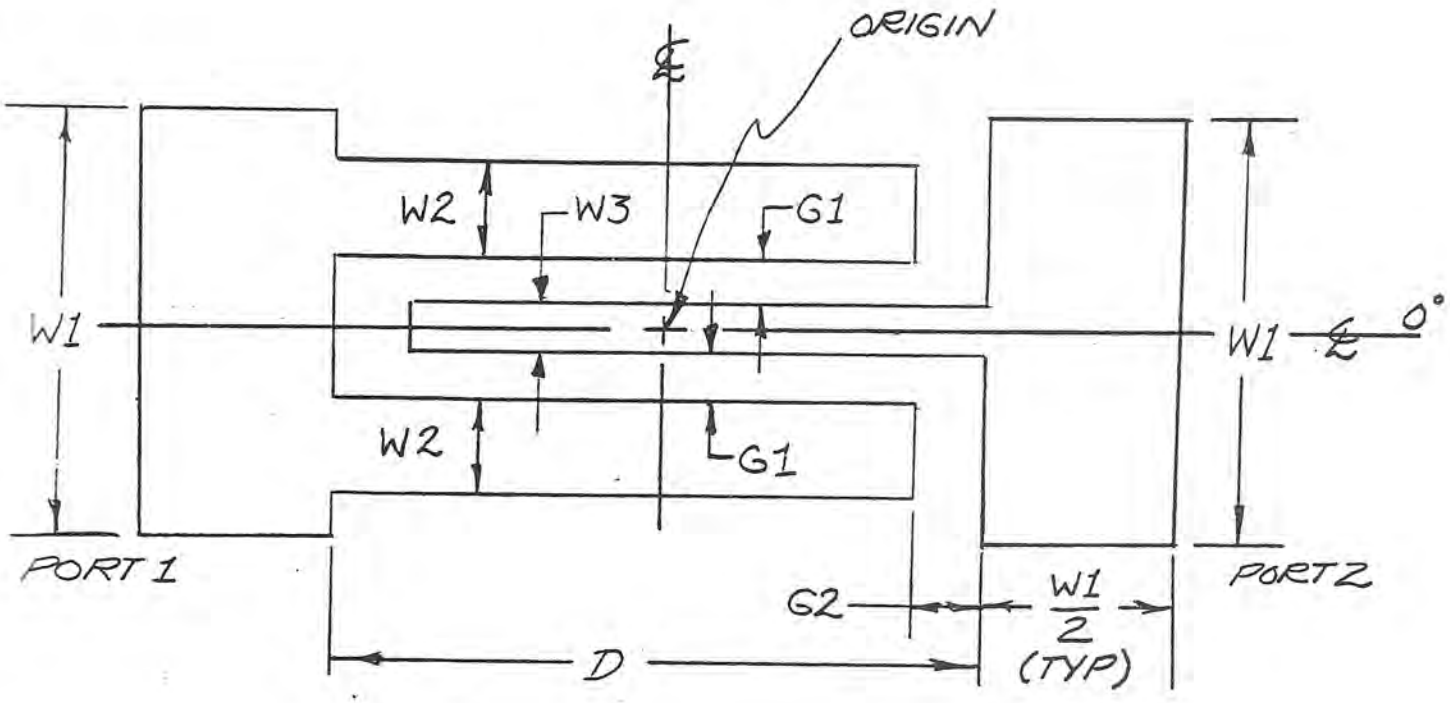


FORM: !DCBLK1 W1, W2, G2, G1, D (NAME[X,Y,A]

WHERE W1 = INPUT AND OUTPUT LINE WIDTH
W2 = FINGER LINE WIDTH
G2 = GAP BETWEEN FINGERS
G1 = GAP BETWEEN FINGER & LINE
D = DISTANCE BETWEEN LINES

DCBLK2

DC BLOCK - 3 FINGER CONFIGURATION

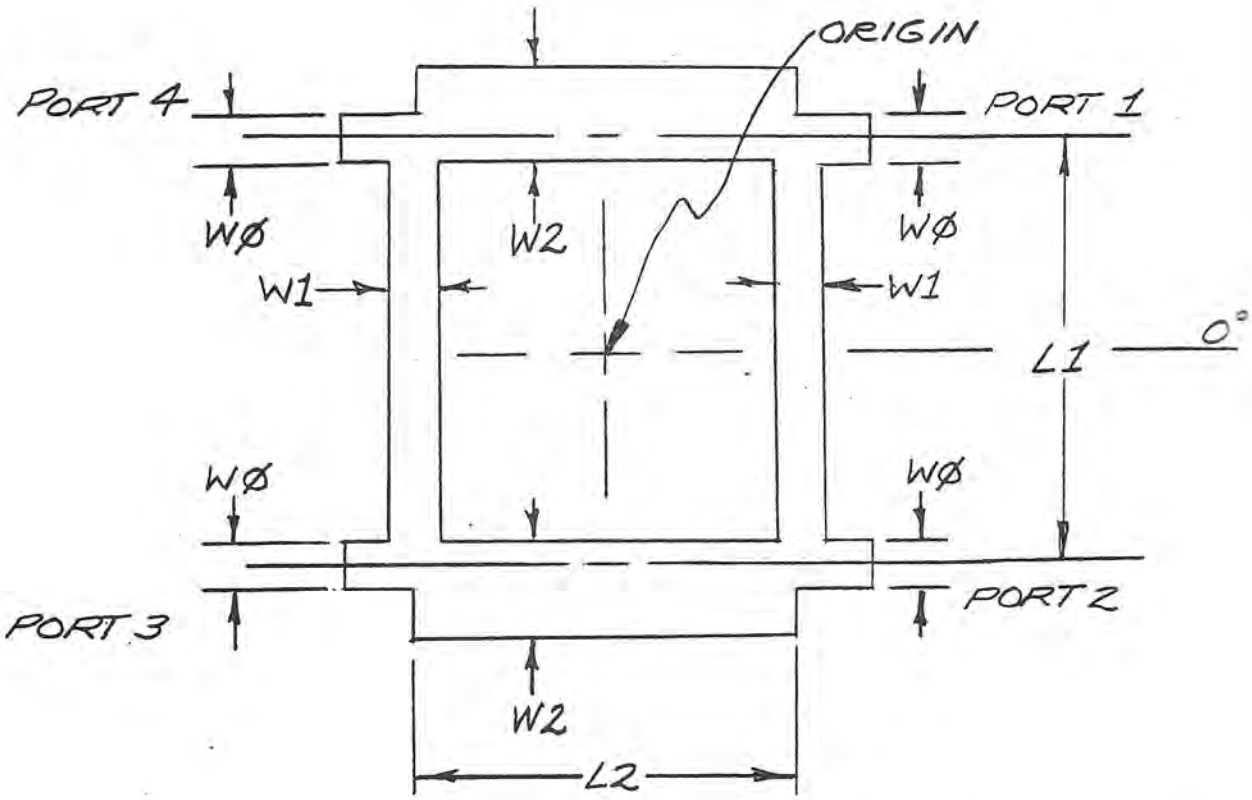


FORM: !DCBLC2 W1, W2, W3, G1, G2, D (NAME[X,Y,A])

WHERE W1 = INPUT LINE WIDTH
W2 = OUTER FINGER WIDTH
W3 = INNER FINGER WIDTH
G1 = GAP BETWEEN FINGERS
G2 = GAP BETWEEN FINGER & LINE
D = DISTANCE BETWEEN INPUT LINES

HYB

90 DEGREE HYBRID (BRANCH LINE COUPLER)

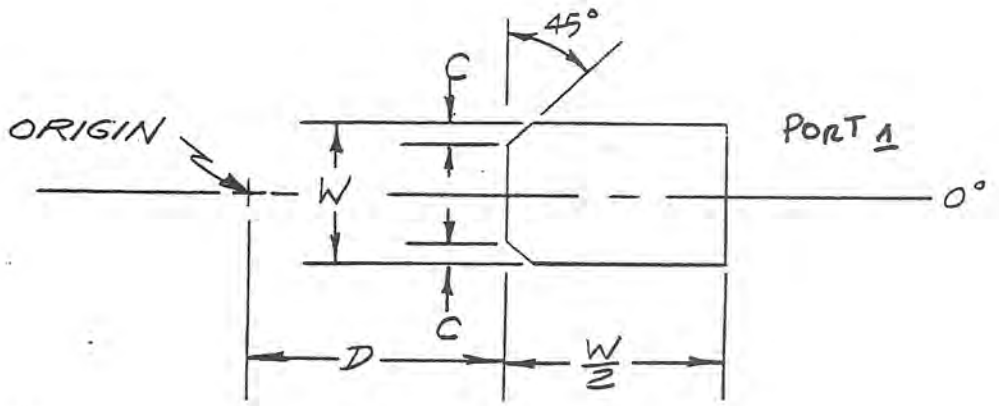


FORM: !HYB WØ, W1, W2, L1, L2 (NAME [X, Y, A])

WHERE WØ = INPUT LINE WIDTH
W1 = HYBRID BRANCH LINE WIDTH
W2 = HYBRID THRU LINE WIDTH
L1 = DISTANCE BETWEEN PORT CENTER LINES
L2 = LENGTH OF THRU LINE

LOAD1

LINE TERMINATION FOR DROP IN ELEMENT

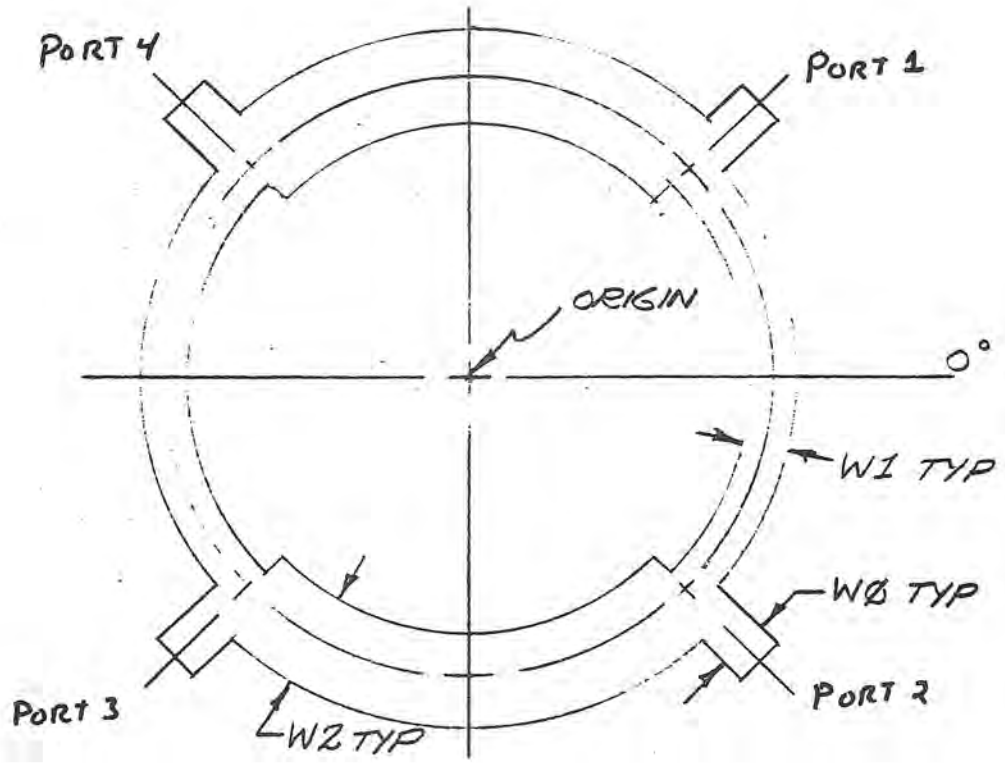


FORM !LOAD1 W, C, D (NAME[X, Y, A])

WHERE W = LINE WIDTH
C = CHAMFER DIMENSION
D = DISTANCE FROM ORIGIN TO EDGE OF LINE

MICBLC

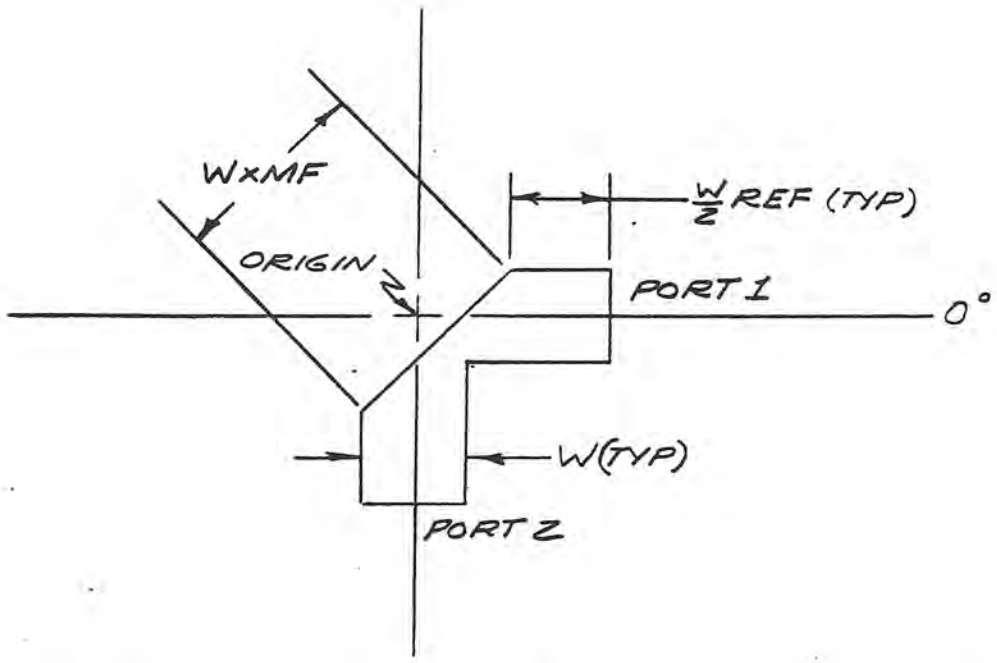
MICROSTRIP BRANCH LINE COUPLER



FORM: !MICBLC WØ, W1, W2, L1, L2 (NAME [X, Y, A])

WHERE WØ = INPUT LINE WIDTH
W1 = LINE WIDTH AS SHOWN
W2 = LINE WIDTH AS SHOWN
L1 = FULL ELECTRICAL WAVELENGTH FOR LINE W1
L2 = FULL ELECTRICAL WAVELENGH FOR LINE W2

MITRE90

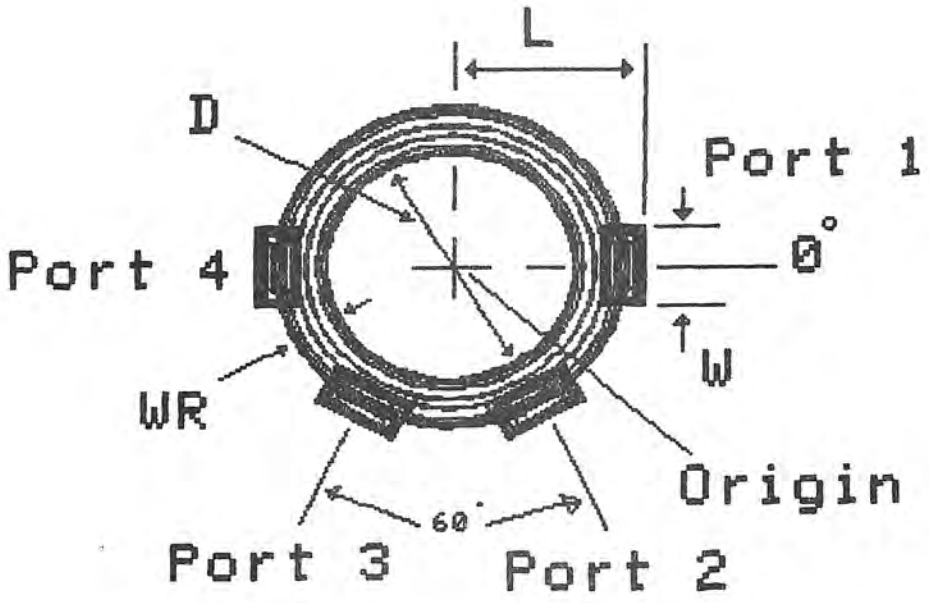


FORM: !MITRE90 W, MF (NAME [X, Y, A])

WHERE: W = LINE WIDTH
MF = MITER FACTOR (TYPICALLY 1.61)

RATEQ

HYBRID RING - EQUAL POWER SPLIT



FORM : !RATEQ W WR D(NAME[X Y A])

WHERE W = LINE WIDTH
WR = RING LINE WIDTH
D = RING DIAMETER

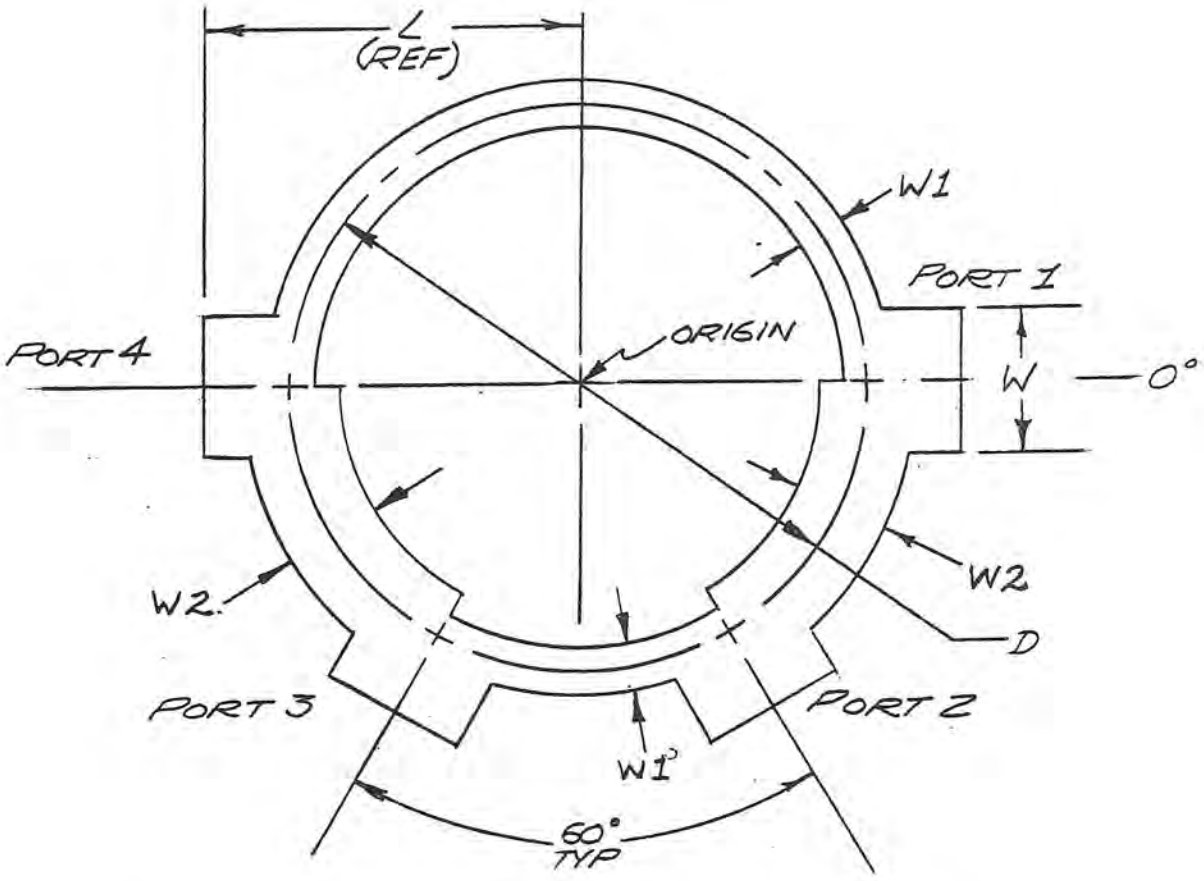
NOTE : L = (W+D)/2+.005-WR/2

EXAMPLE : THE ILLUSTRATION ABOVE WAS DRAWN BY THE FOLLOWING PROGRAM:

```
!NAME RATEQ
!TARGET .01[-3,-3,0][-3,3,0][3,-3,0][3,3,0]
!RATEQ .5 .3 1.7(NAME[0 0 0])
!END
!STOP
```

RATUNEQ

HYBRID RING - UNEQUAL POWER DIVISION



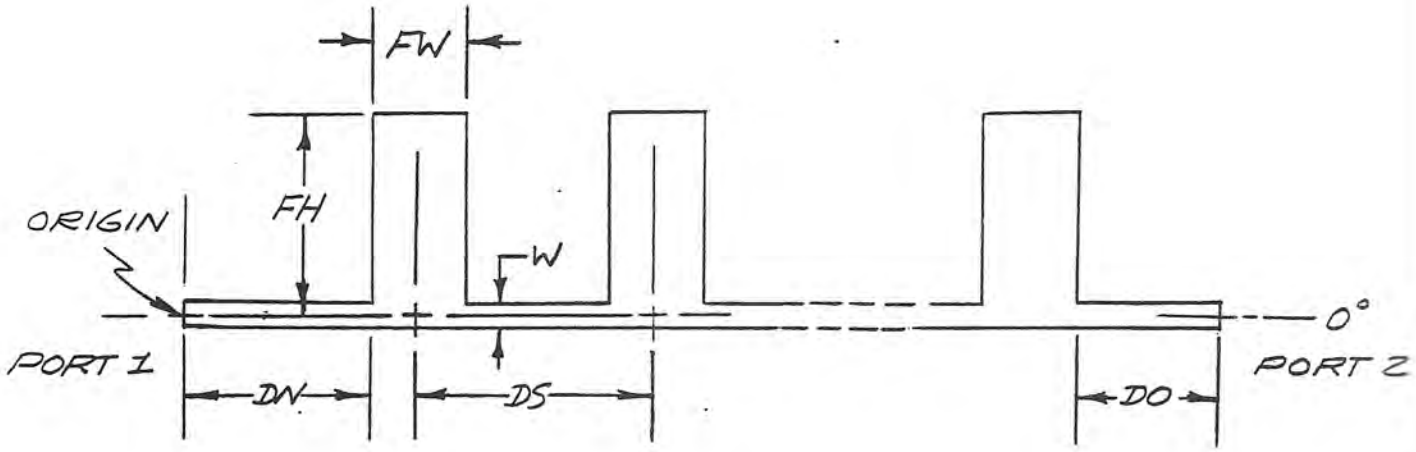
FORM: !RATUNEQ W, W1, W2, D (NAME[x, y, A])

WHERE W = LINE WIDTH
W1 = LINE WIDTH FOR INDICATED SEGMENTS
W2 = LINE WIDTH FOR INDICATED SEGMENTS
D = RING DIAMETER

NOTE: W1 MAY BE > OR < W2
$L = (W+D)/2 + .005 - W1/2$

RF CHOKE

R.F. CHOKE STRUCTURE

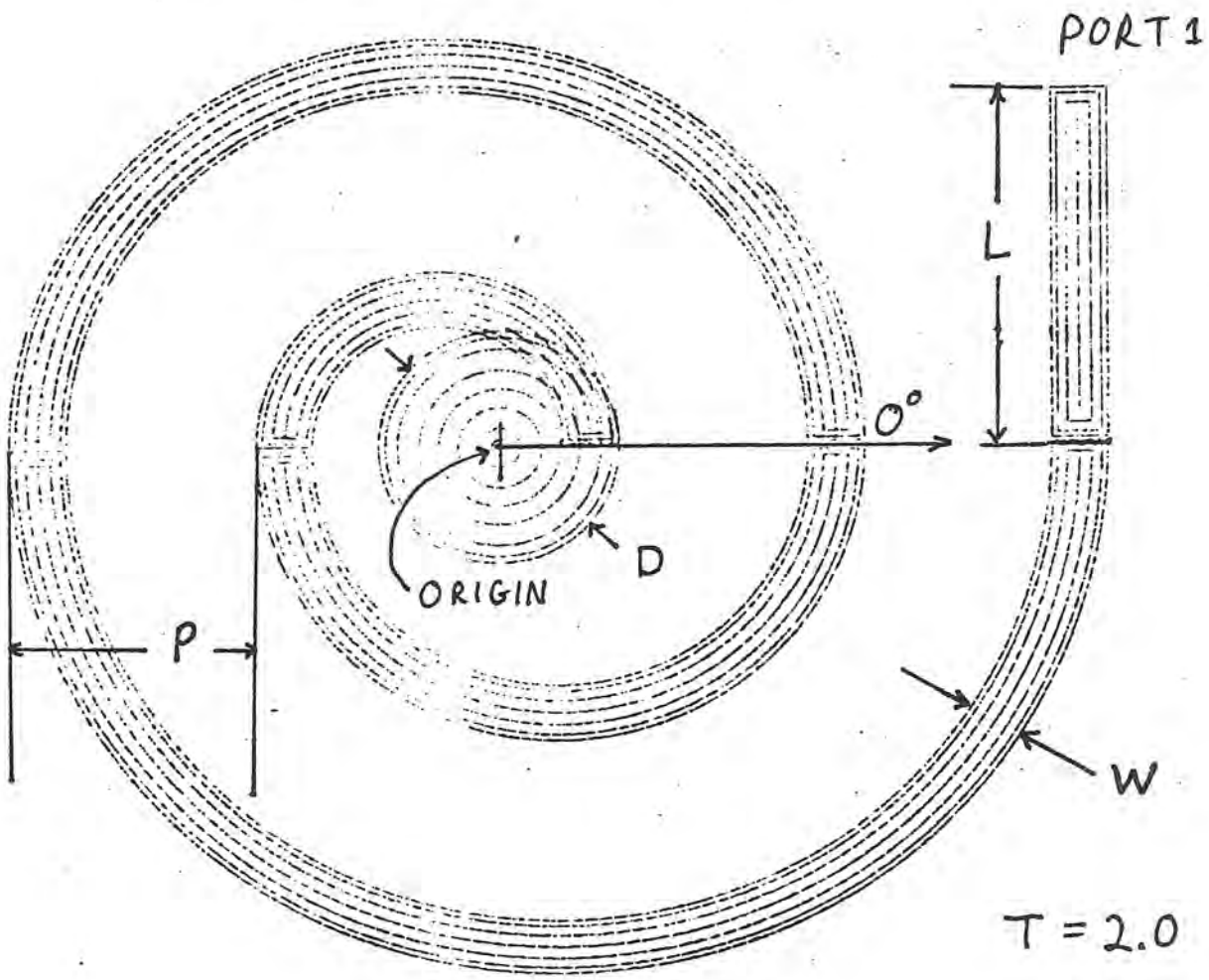


FORM: !RFCHOKE W, FW, FH, DN, DO, DS, N (NAME[X,Y,F

WHERE W = LINE WIDTH
FW = WIDTH OF FLAG
FH = FLAG HEIGHT
DN = DISTANCE FROM INPUT TO EDGE OF FIRST FLAG
DO = DISTANCE FROM OUTPUT TO EDGE OF LAST FLAG
DS = CENTERLINE DISTANCE BETWEEN FLAGS
N = NUMBER OF FLAGS

SPIRAL

SPIRAL INDUCTOR WITH 1 PORT AND A TAIL

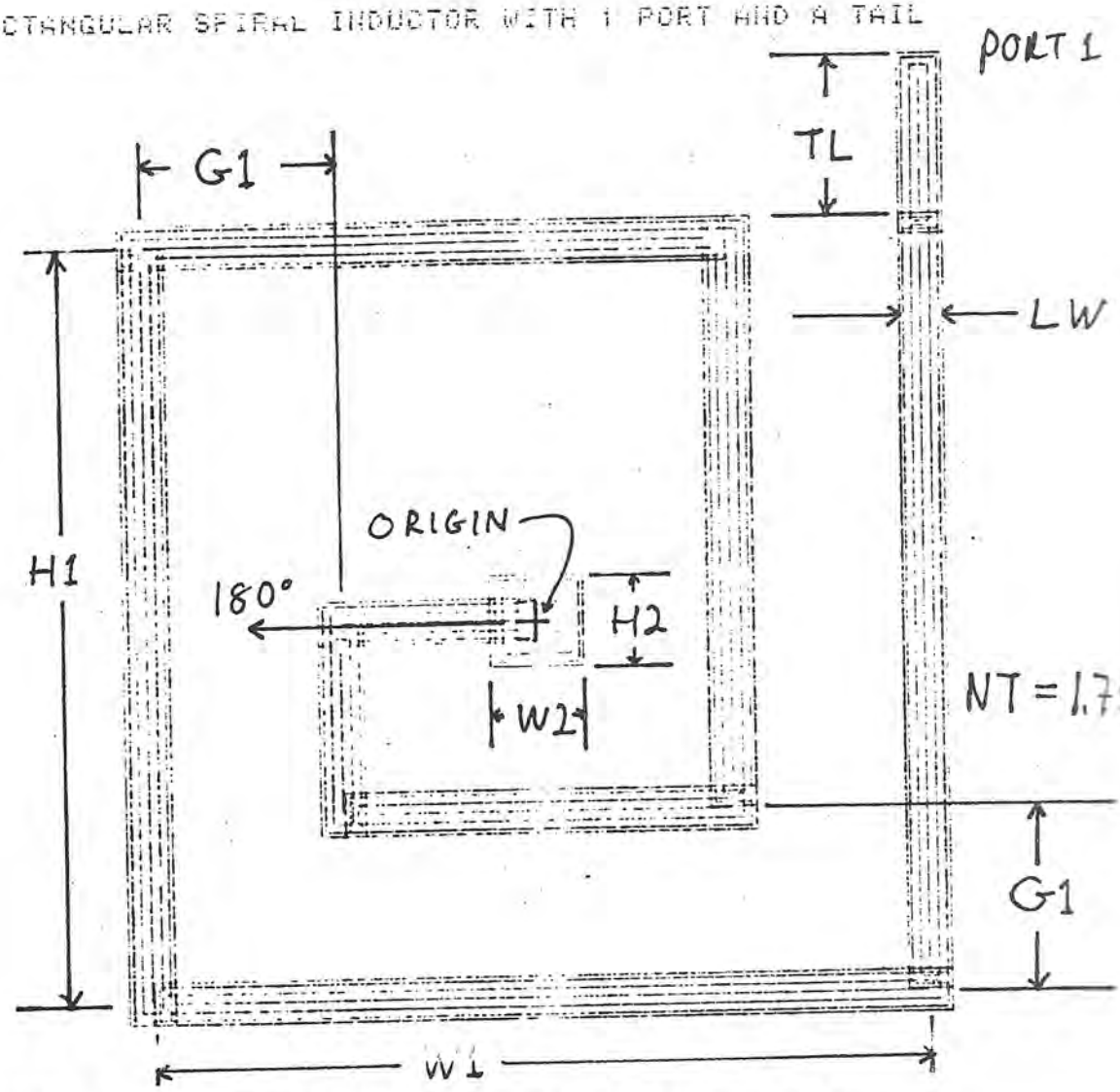


FORM : !SPIRAL D P W T L (NAME[X,Y,A])

WHERE D = CENTER PAD DIAMETER
P = PITCH IN INCHES/TURN ALONG 0 AXIS
W = WIDTH OF SPIRAL LINE
T = NUMBER OF TURNS
L = LENGTH OF STRAIGHT SECTION AT OUTER END OF SPIRAL

| SPIRAL1 |

RECTANGULAR SPIRAL INDUCTOR WITH 1 PORT AND A TAIL



FORM : !SPIRAL1 W1 H1 LW G1 NT W2 H2 TL (NAME[X,Y,A])

WHERE W1 = CENTERLINE WIDTH OF INDUCTOR
H1 = CENTERLINE HEIGHT OF INDUCTOR
LW = WIDTH OF SPIRAL LINE
G1 = LINE CENTER SPACING
NT = NUMBER OF TURNS (4*N MUST BE AN INTEGER!)
W2 = WIDTH OF CENTER PAD
H2 = HEIGHT OF CENTER PAD
TL = LENGTH OF TAIL

NOTES:

THE SPIRAL WORKS INWARDS FROM THE W1 AND H1 DIMENSIONS. IT IS UP TO THE USER TO MAKE SURE THAT THE GEOMETRY WORKS!

PHANTOM COMPONENT WITH A NAME AND n PORTS

FORM : !XPORTS NAME **N** [X1,Y1,A1] [X2,Y2,A2]...[Xn,Yn,An]

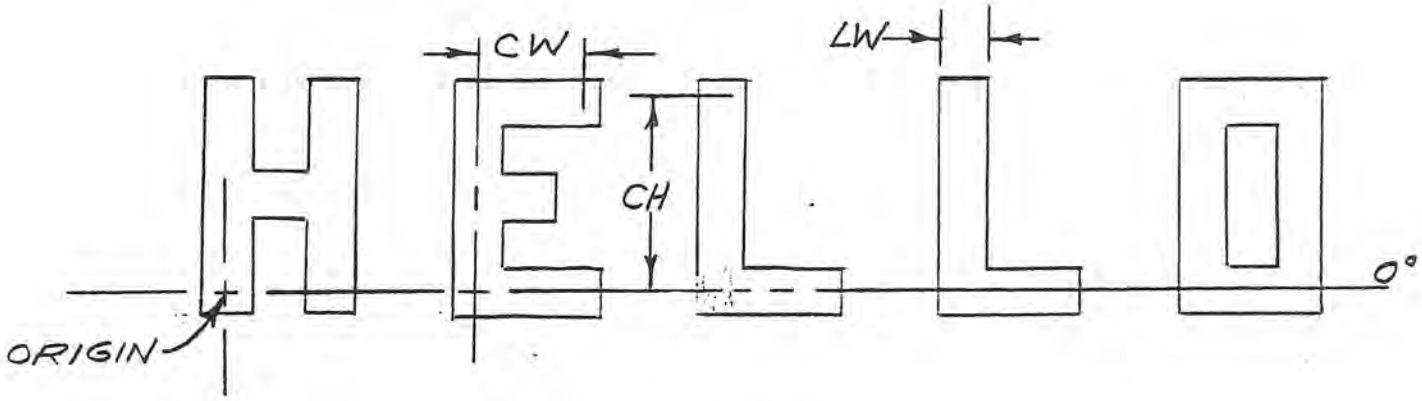
WHERE NAME = PORT SET NAME
N = NUMBER OF PORTS TO BE DEFINED
Xi = X COORDINATES OF PORTS
Yi = Y COORDINATES OF PORTS
Ai = ANGLES OF PORTS

NOTES: THE !XPORTS COMMAND IS USED TO DEFINE N PORTS TO BE ASSOCIATED WITH A NAME. THE PORTS MAY THEN BE USED IN !CONNECT OR !CLINE STATEMENTS IN THE NORMAL WAY. THE NUMBER OF PORTS N MUST BE EQUAL TO THE NUMBER OF [X,Y,A] SETS PROVIDED. THE PORTS CREATED BY !XPORTS MAY BE PLACED ANYWHERE, SO THAT THE CREATION OF CUSTOM COMPONENTS IS NOW MADE EASY.

NON-PORTED COMPONENTS

ALF

ALPHANUMERIC CHARACTER GENERATION



FORM: !ALF "XXX...X" CH, CW, LW [X, Y, A]

WHERE "XXX...X" = MESSAGE TO BE WRITTEN (60 CHARACTERS MAXIMUM)
CH = CHARACTER HEIGHT
CW = CHARACTER WIDTH
LW = LINE THICKNESS USED TO DRAW CHARACTER.
X,Y = LOCATION OF MESSAGE ORIGIN
A = MESSAGE ORIENTATION ANGLE.

NOTE: IN SOME CASES THE CHARACTER PRINTED WILL NOT CORRESPOND WITH THE KEYBOARD CHARACTER. ALSO, SOME KEYS WILL NOT PRINT ANYTHING. UNLESS SHOWN BELOW, THE KEYBOARD AND PRINTED CHARACTERS WILL BE IDENTICAL. (SORRY ABOUT THIS).

KEYBOARD CHARACTER:	*	'	(	)	_	\	^
CHARACTER PRINTED:	$	)	[	±	%	)	°

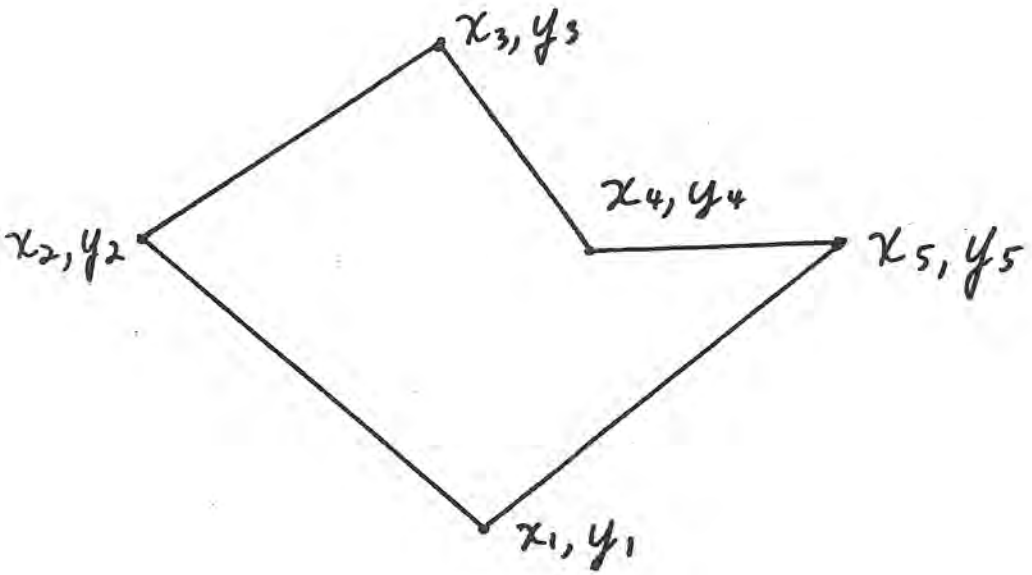
NON PRINTING CHARACTERS: ! & @ [< > ?

ALPHANUMERIC CHARACTERS
AVAILABLE FROM TALF

```
%  (+°]
ZXCVBNM▫▫/
ASDFGHJKL
QWERTYUIOP
$ )[+| = *
1234567890▫▫
```

BOUNDARY

IRREGULAR POLYGON DEFINED BY VERTICES



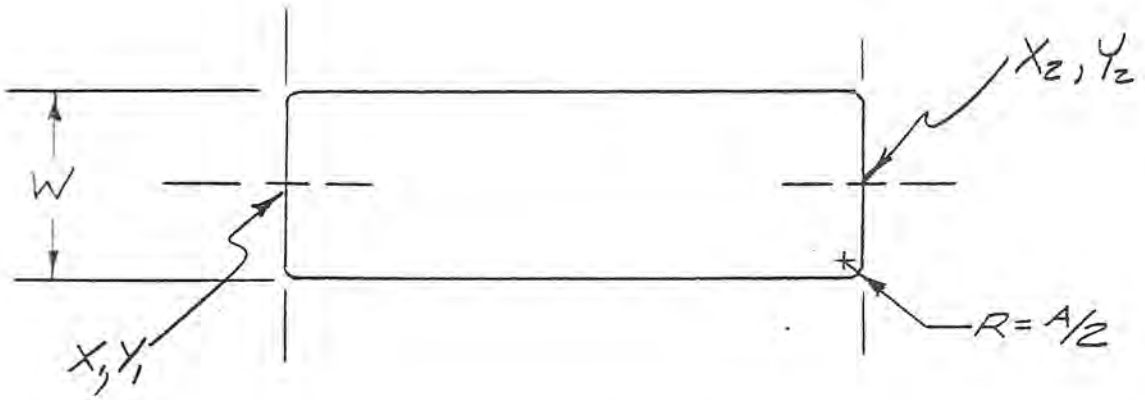
FORM : !BD LW [X1,Y1] [X2,Y2]...[Xn,Yn]

WHERE LW = LINE WIDTH USED TO PAINT INSIDE AREA OF POLYGON
Xi = X COORDINATES OF POINTS
Yi = Y COORDINATES OF POINTS
(n must be less than 401)

NOTES :

1. DO NOT CLOSE THE POLYGON PERIMETER BY REPEATING THE STARTING POINT AT THE END!
2. NO TWO POINTS SHOULD COINCIDE.
3. NO THREE ADJACENT POINTS SHOULD BE COLINEAR.
4. THE POINTS WILL BE CONNECTED SEQUENTIALLY IN THE ORDER GIVEN.
5. IN ORDER FOR THE ETCH FACTOR COMMAND TO WORK PROPERLY THE INTERIOR OF THE POLYGON MUST BE TO THE RIGHT AS THE FORM IS TRACED. THAT IS, YOU MUST DESCRIBE THE POLYGON BY GOING AROUND IT CLOCKWISE. IF YOU GO THE WRONG WAY, THE ETCHFACTOR WILL BE SUBTRACTED INSTEAD OF ADDED!
6. IF YOU WISH YOU MAY TURN OFF THE ETCHFACTOR BY USING AN !ETCHFACTOR 0 BEFORE THE !BD AND THEN RESETTING THE !ETCHFACTOR AFTERWARDS.

BOX



FORM: !BOX W, A [X1, Y1][X2, Y2] ··· [XN-1, YN-1][XN, YN]

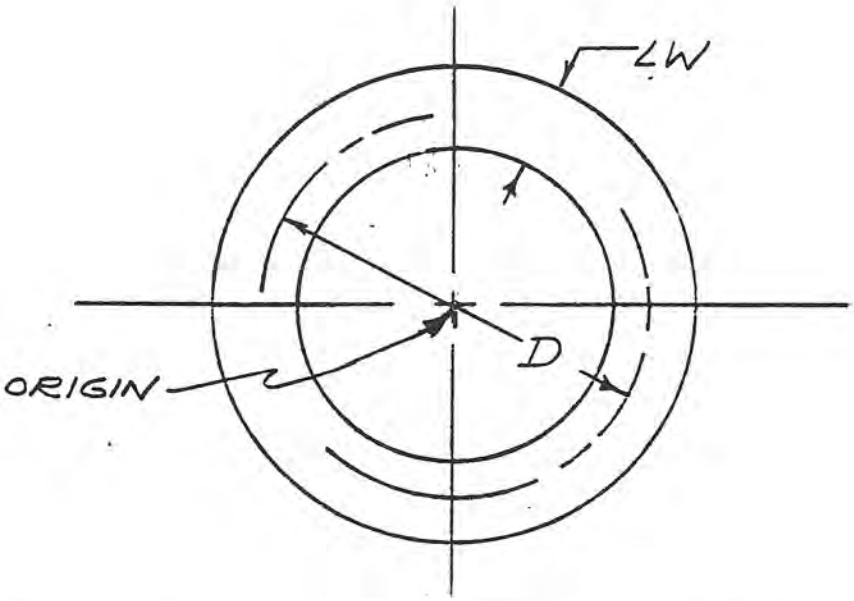
WHERE W = WIDTH
A = TWO TIMES THE CORNER RADIUS
(MINIMUM A=.010)
[X1, Y1][X2, Y2] = COORDINATE PAIR DEFINING THE BOX LENGTH AND ORIENTATION

NOTE: "A" SPECIFIES THE GERBER APERTURE TO BE USED IN DRAWING THE BOX PERIMETER. IF THE DESIRED RADIUS IS NOT CONSISTANT WITH THE GERBER APERTURES AVAILABLE, THE NEAREST, SMALLER APERTURE WILL AUTOMATICALLY BE SELECTED.

IF MORE THAN ONE BOX HAVING THE SAME "W" AND "A" IS REQUIRED, ADD THE APPROPRIATE COORDINATE PAIRS TO THE INITIAL STATEMENT.

NO PORT NUMBERS ARE ASSOCIATED WITH !BOX. CONNECTIONS ARE MADE TO THE APPROPRIATE X, Y LOCATIONS

CIRCLE

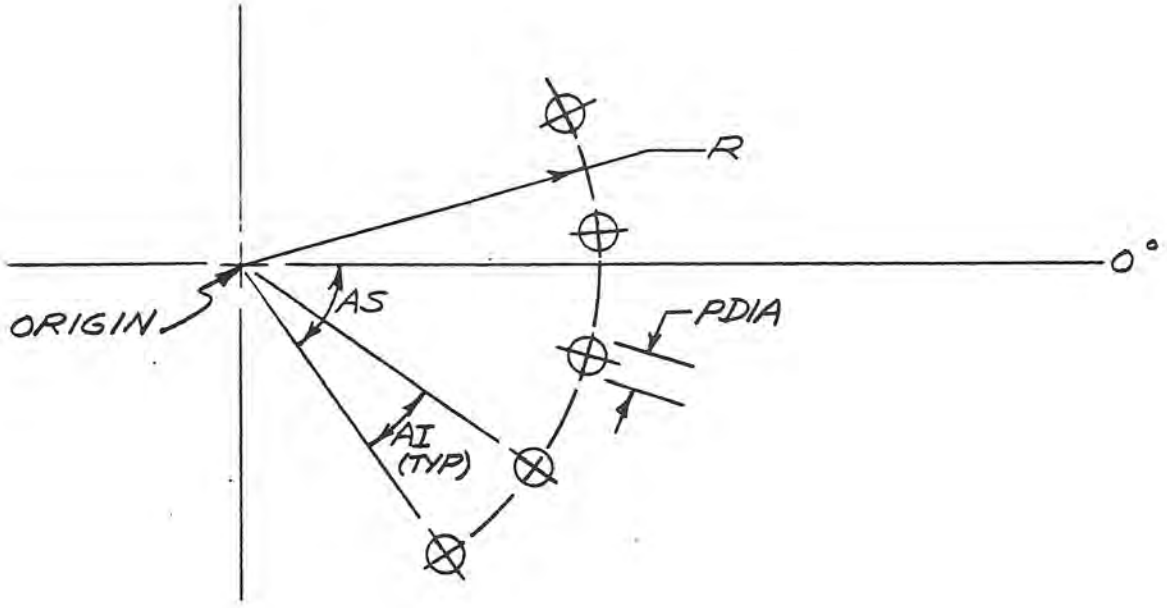


FORM: !CIRCLE LW, D [X, Y]

WHERE LW = LINE WIDTH
D = DIAMETER

EPDR

EXACT PADS ON A RADIUS



FORM: !EPDR PDIA, R, AS, AI, PN [X, Y]

WHERE: PDIA = PAD DIAMETER
R = RADIUS
AS = START ANGLE
AI = INCREMENT ANGLE
PN = NUMBER OF PADS

LINE

DRAWS A (NON-EXACT) LINE

FORM: !LI A $[X_1, Y_1]$ $[X_2, Y_2]$ .... $[X_N, Y_N]$

WHERE: A = LINE WIDTH (.010" MIN)
X_1, Y_1 = POINT WHERE LINE STARTS
X_2, Y_2 = INTERMEDIATE POINT
X_N, Y_N = POINT WHERE LINE ENDS

NOTES: "A" SELECTS THE GERBER APERTURE USED TO DRAW THE LINE. IF THE WIDTH SELECTED DOES NOT CORRESPOND TO A GERBER APERTURE CURRENTLY AVAILABLE, THE NEXT SMALLEST WILL BE USED.

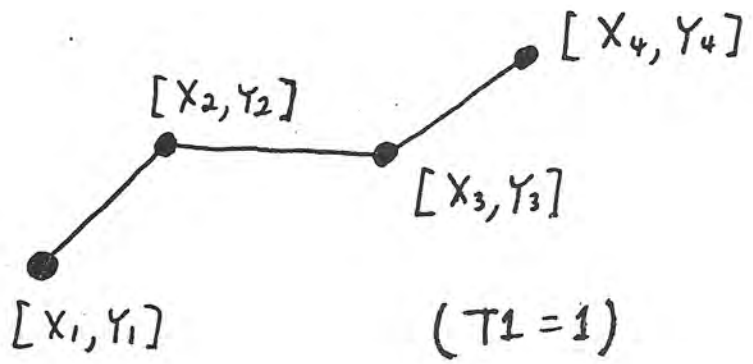
THE LINE WILL HAVE RADII AT THE ENDS AND INTERMEDIATE POINTS CONSISTANT WITH THE APERTURE USED.

LINE WIDTHS AVAILABLE ARE:

.010	.013	.015	.016
.020	.025	.030	.050
.060	.100	.120	

LIPD

BROKEN LINE WITH PADS AT ENDS AND AT ANGLES



FORM : !LIPD W1 D1 T1 [X1,Y1],[X2,Y2],[X3,Y3].....

WHERE W1 = LINE WIDTH (INEXACT--USES NEXT SMALLER APERATURE)
D1 = DIAMETER OF PADS (INEXACT)
T1 = TYPE (SEE NOTE)
[Xi,Yi] = POINTS TO CONNECT

NOTES:

THE BROKEN LINE IS DRAW THROUGH THE POINTS [Xi,Yi] AS LISTED
IF T1 = 0, PADS ARE DRAWN AT THE FIRST AND LAST POINTS ONLY
IF T1 = 1, PADS ARE DRAWN AT ALL LISTED POINTS

MATRIX

FORM !MATRIX NAME, CS, RS, NC, NR [X,Y,A] [X_2,Y_2,A_2] ... [X_n,Y_n,A_n]

WHERE NAME = NAME OF THE MATRIX (UP TO 10 CHARACTERS)
CS = SPACING BETWEEN THE COLUMNS
RS = SPACING BETWEEN THE ROWS
NC = NUMBER OF COLUMNS
NR = NUMBER OF ROWS
X,Y,A = X,Y COORDINATES AND ANGLE OF ROTATION OF THE INITIAL COMPONENT OF THE MATRIX

FUNCTION: TO GENERATE A MATRIX OF USER-DEFINED ELEMENTS

NOTES: THE COLUMN SPACING AND THE ROW SPACING WILL REMAIN CONSTANT REGARDLESS OF ANGLE OF ROTATION OR OFFSET.

SEE THE USER-DEFINED ELEMENT SECTION FOR A FURTHER DESCRIPTION OF THE MATRIX COMMAND

PAD

(NON-EXACT) CIRCULAR PAD

FORM: !PD A [X_1, Y_1] . . . . [X_N, Y_N]

WHERE: A = DIAMETER OF PAD (.010 MIN)
X_1, Y_1 } PAD LOCATIONS
X_N, Y_N }

NOTES: "A" SELECTS THE APERTURE USED TO FLASH THE PAD. IF THE DIAMETER SELECTED DOES NOT CORRESPOND TO A CURRENTLY AVAILABLE GERBER APERTURE, THE NEXT SMALLEST WILL BE USED.

IF A = 999 A MOIRE PATTERN WILL BE DRAWN.

AVAILABLE PAD DIAMETERS

.036 .055 .063 .068 .075 .086

.093 .100 .112 .125 .150

PDEXACT

EXACT PAD

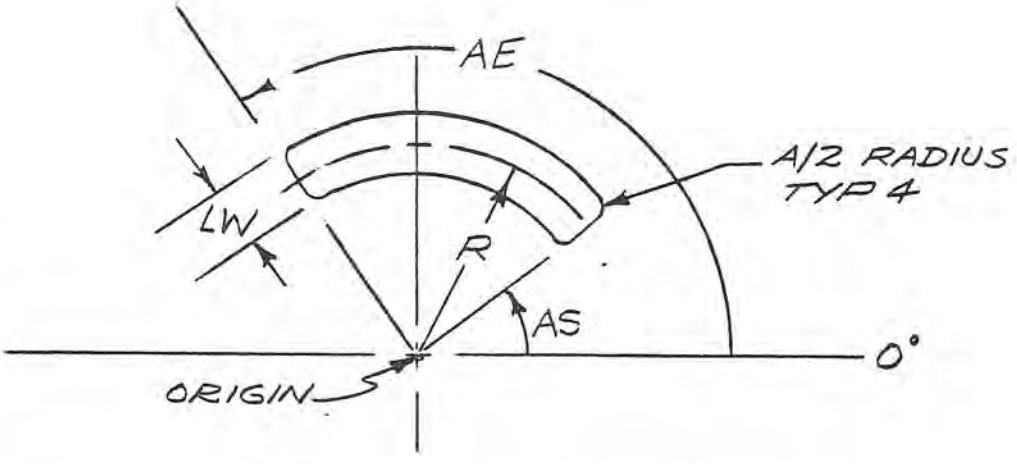
FORM: ! PDEXACT PDIA [X_1,Y_1] [X_2,Y_2] .... [X_n,Y_n]

WHERE PDIA = DIAMETER OF PAD
X,Y = LOCATION OF PAD

NOTE: THE PAD DIAMETER, PDIA WILL BE EXACTLY DRAWN UNLIKE PD WHICH SELECTS THE CLOSEST APERTURE

SARC

ARC SEGMENT



FORM: !SARC LW, R, A, AS, AE [X, Y]

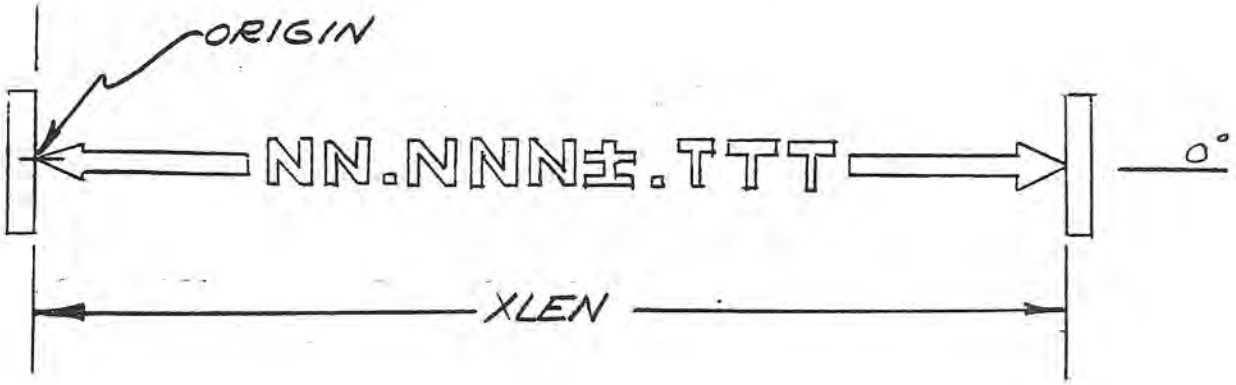
WHERE: LW = LINE WIDTH
R = RADIUS
A = CORNER DIAMETER
AS = START ANGLE
AE = END ANGLE

NOTES: "A" SPECIFIES THE GERBER APERTURE TO BE USED IN DRAWING THE ARC PERIMETER. IF THE DESIRED RADIUS IS NOT CONSISTANT WITH THE GERBER APERTURES AVAILABLE, THE NEAREST SMALLER APERTURE WILL USED (UNLESS "A" SPECIFIED AS ≤ .010, IN WHICH CASE A = .010 WILL BE SELECTED).

"AS" AND "AE" ARE MEASURED CCW FROM 0° REFERENCE AXIS.

SZMSG

SIZE MESSAGE (MESSAGE INDICATING DIMENSION)

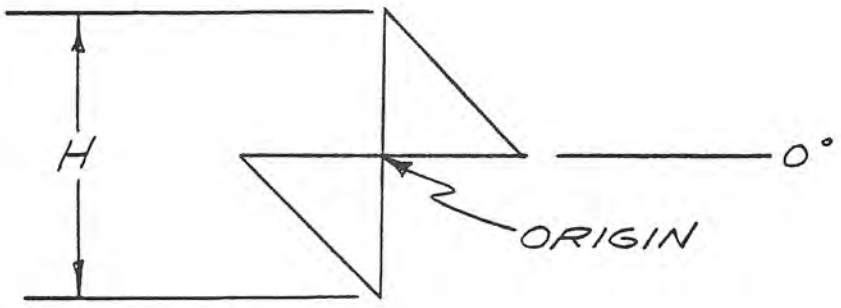


FORM: !SZMSG XLEN, XERR, 0.0 [X, Y, A]

WHERE: XLEN = LENGTH BETWEEN LINES, ALSO PRINTED DIMENSION NN.NNN
XERR = PRINTED TOLERANCE, .TTT

NOTE: IF XLEN ≤ 1.4, THE MESSAGE WILL BE SHIFTED .2 * SCALE FACTOR " BENEATH THE ORIGIN

TARGET



FORM: !TARGET H [X, Y, A]

WHERE H = TARGET HEIGHT

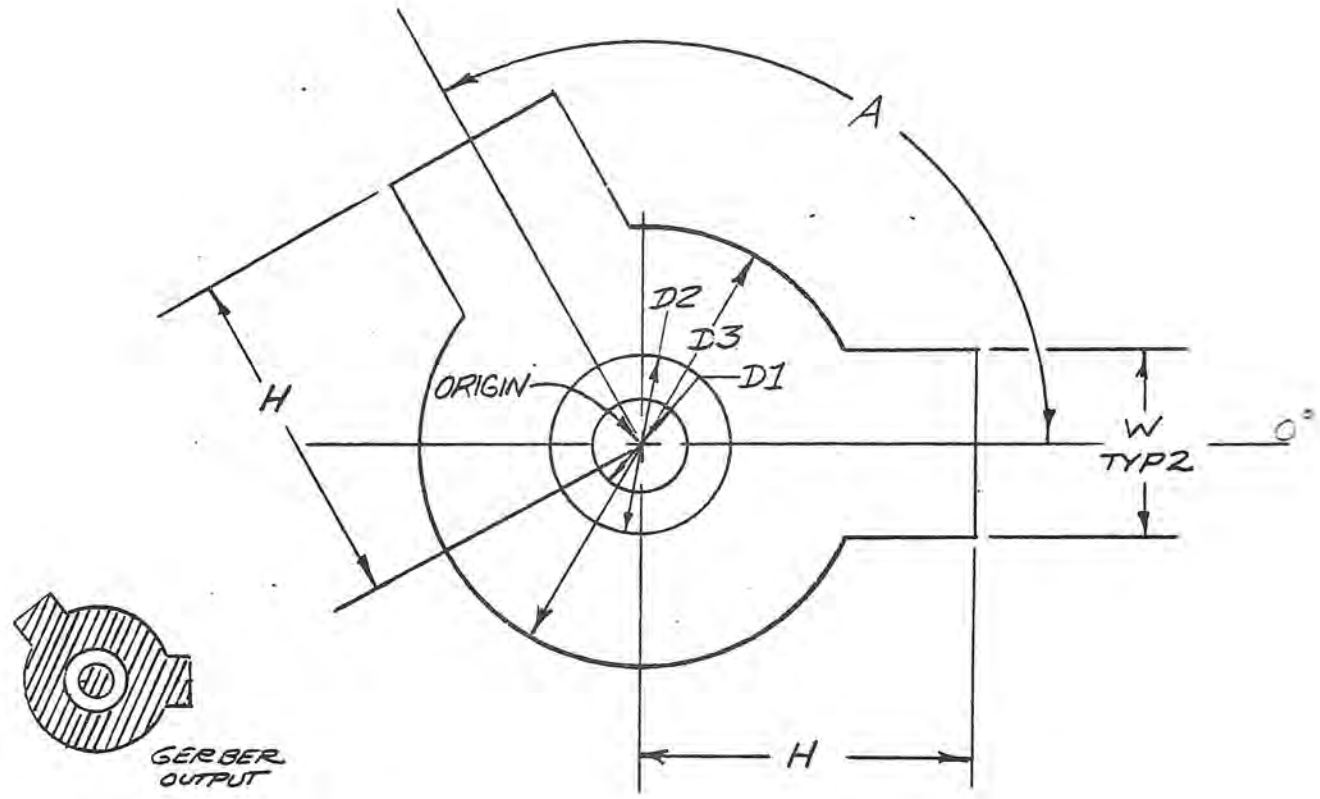
GROUND PLANE COMPONENTS

COMPONENT	PG.	DESCRIPTION
BREAKCVR	50	CUT-OUT, CIRCULAR DROP-IN ELEMENT
CONN1CVR	51	PATTERN FOR SURFACE LAUNCHER
CONN2CVR	52	PATTERN FOR EDGE LAUNCHER
CVR1	53	CUT-OUT, RECTANGULAR DROP-IN
DRILLPD	54	DRILL PAD

THESE ELEMENTS PROVIDE GROUND PLANE CONFIGURATIONS TO BE USED WHEN CERTAIN COMPONENTS ARE IMPLEMENTED IN STRIPLINE. THE GERBER OUTPUTS INDICATED SHOW THE LIGHT AND DARK AREAS PRODUCED ON THE GERBER FILM. TO REVERSE THESE LIGHT AND DARK REGIONS MAKE A CONTACT PRINT OF THE GERBER FILM.

BREAKCVR

GROUND PLANE PATTERN FOR CIRCULAR DROP IN COMPONENT



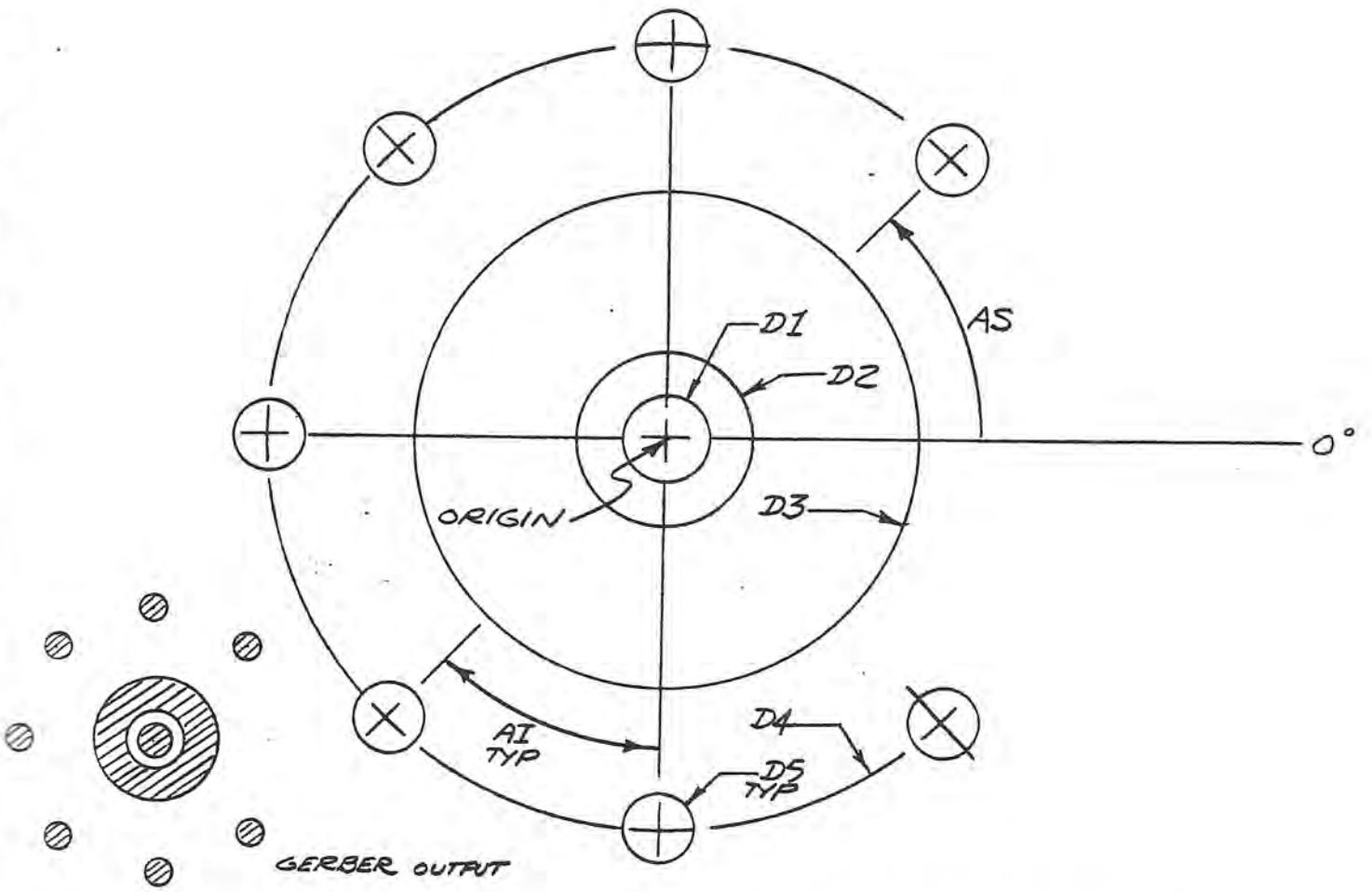
FORM:!BREAKCVR W,D1,D2,D3,H,A [X,Y,A]

WHERE: W = TAB WIDTH
D1 } DRILL PAD
D2 } DIAMETERS
D3 = OUTER DIAMETER
H = TAB LENGTH
A = ANGLE BETWEEN TAB CENTERLINES

NOTES: SETTING H=0 REMOVES TABS
SETTING A=0 PRODUCES ONE TAB

CONN1CVR

GROUND PLANE PATTERN FOR SURFACE LAUNCH CONNECTOR

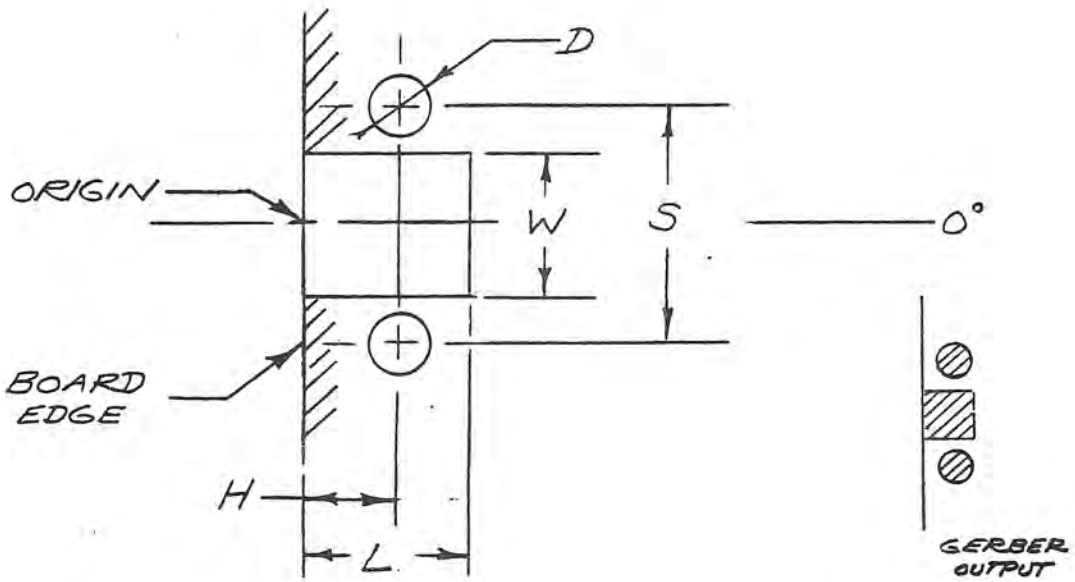


FORM: !CONN1CVR D1, D2, D3, D4, D5, AS, AI, PN [X, Y, A]

WHERE: D1 } FORM DRILL PAD FOR CENTER
D2 } CONDUCTOR PIN HOLE
D3 = CLEARANCE ETCH IN GROUND PLANE
D4 = ESTABLISHES BC FOR CONNECTOR SCREWS
D5 = DIAMETER OF SCREW HOLE DRILL PAD
AS = START ANGLE
AI = INCREMENT ANGLE
PN = NUMBER OF SCREW HOLE DRILL PADS

CONN2CVR

GROUND PLANE HOLE PATTERN FOR STRIPLINE EDGE LAUNCHER (CONN2)



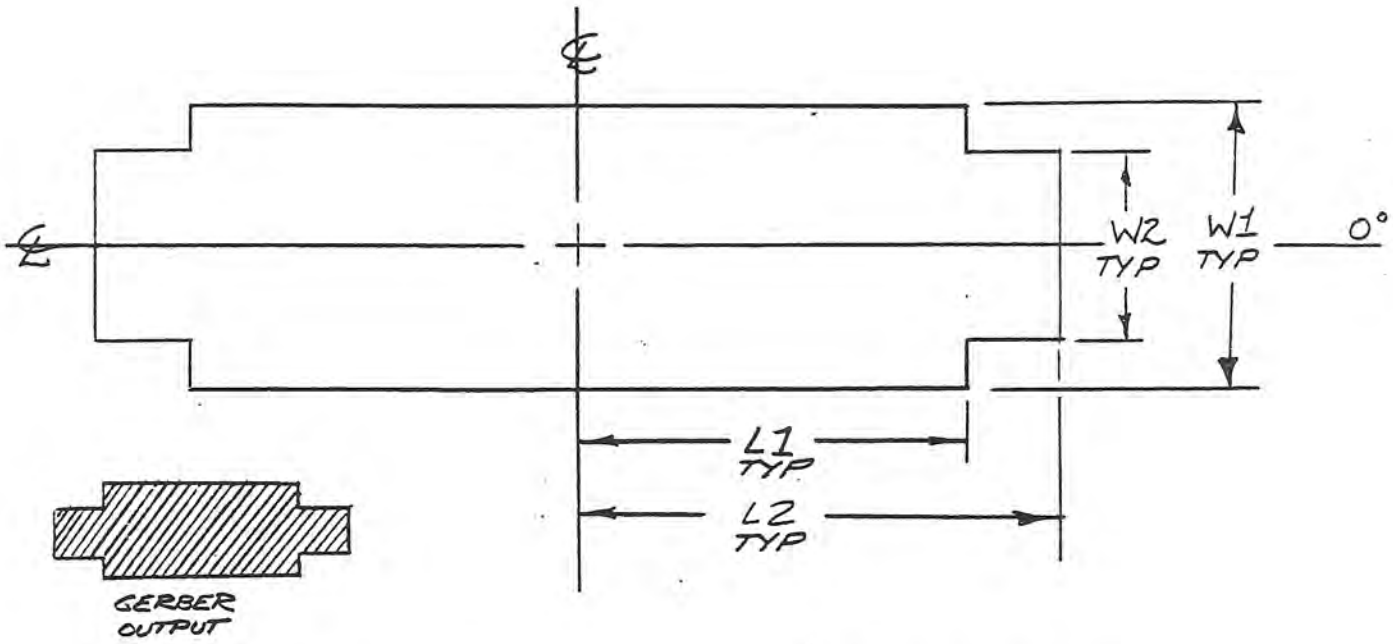
FORM: !CONN2CVR S, H, D, W, L [X, Y, A]

WHERE: S = DISTANCE BETWEEN HOLES
H = DISTANCE FROM CONNECTOR HOLE TO BOARD EDGE
D = DIAMETER OF CONNECTOR HOLE PAD
W = WIDTH OF PLUG CUT OUT
L = LENGTH OF PLUG CUT OUT

NOTE: L=0 REMOVES PLUG CUT OUT

CVR1

GROUND PLANE PATTERN FOR RECTANGULAR DROP IN COMPONENT



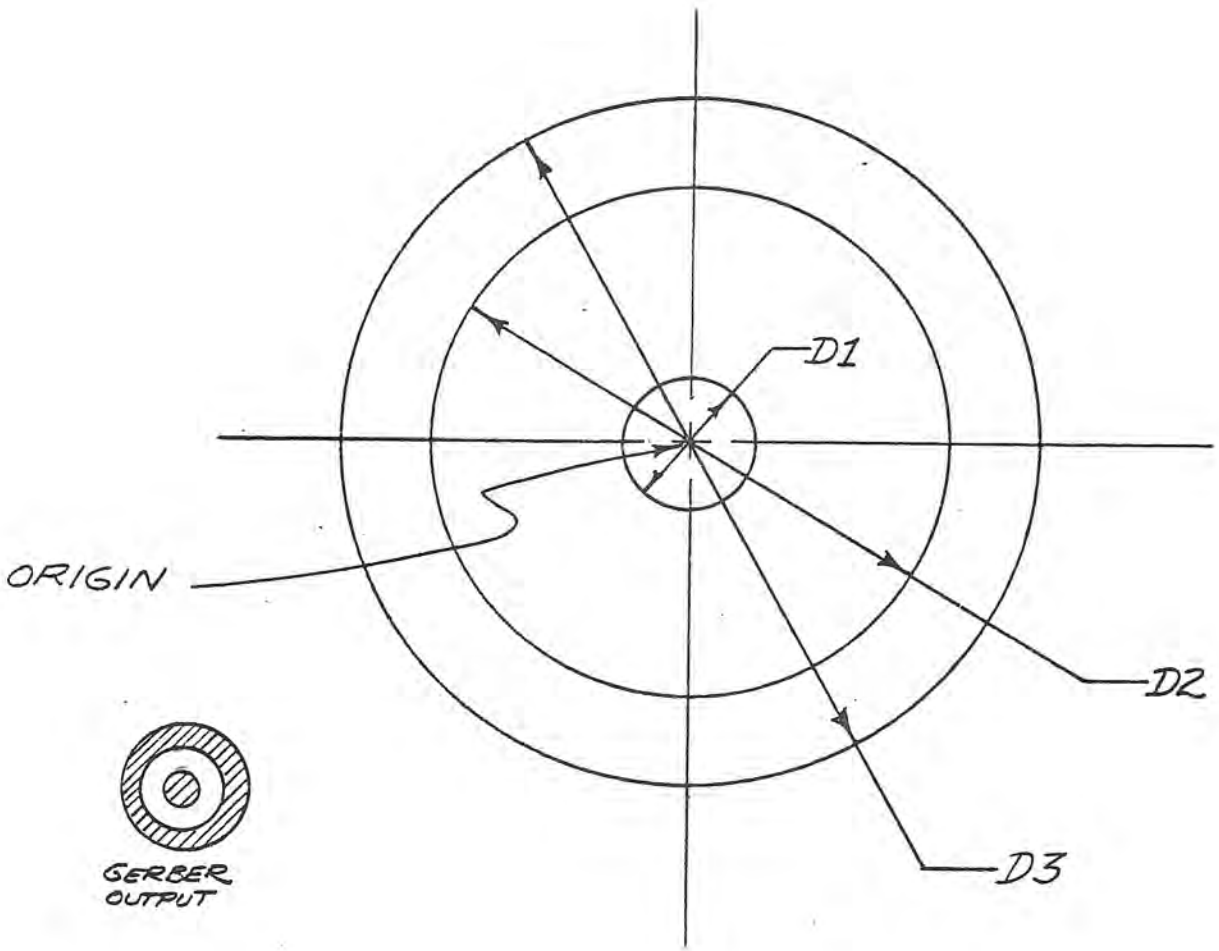
FORM: !CVR1 W1, W2, L1, L2 [X, Y, A]

WHERE: W1 = WIDTH
W2 = TAB WIDTH
L1 = LENGTH
L2 = TAB LENGTH

NOTE: L2=0 REMOVES END TABS

DRILLPAD

GROUND PLANE DRILL PAD



FORM: !DRILLPD D1,D2,D3 [X,Y]

WHERE: D1 = INNER CIRCLE DIAMETER
D2 = OUTER DIAMETER OF (BLANK) CIRCLE
D3 = OUTER DIAMETER OF OUTSIDE CIRCLE

SECTION II

COMPONENT INTERCONNECTION

CONNECTING COMPONENTS TOGETHER

! CONNECT

THE CONNECT COMMAND IS USED TO GENERATE A LINE TO CONNECT ANY TWO POINTS. THE ONLY RESTRICTION IS THAT THE RESULTING LINE MUST BE STRAIGHT OR HAVE ONLY ONE BEND. THE FORM OF THE CONNECT STATEMENT IS:

! CONNECT W,R [POINT A][POINT B]

WHERE W = LINE WIDTH
R = BEND RADIUS
POINT A, POINT B } = START AND END LOCATIONS

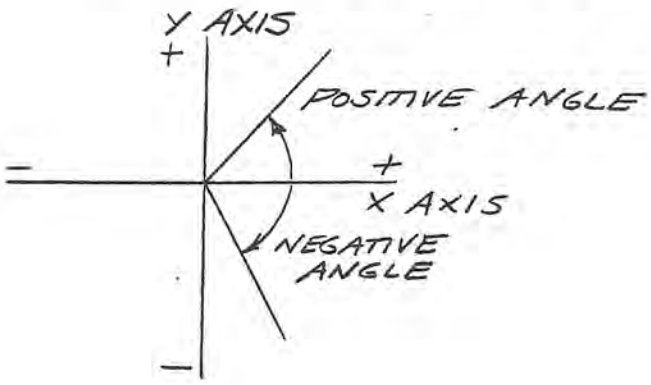
THE LINE WIDTH MAY BE ANY VALUE, HOWEVER, A SCALE FACTOR AND PHOTOGRAPHIC REDUCTION MUST BE USED TO OBTAIN WIDTHS LESS THAN 0.010". THE PROGRAM WILL AUTOMATICALLY DETERMINE AND SELECT THE APPROPRIATE GERBER APERTURES TO BE USED.

IF THE CONNECTION RESULTS IN A STRAIGHT LINE, A RADIUS MUST STILL BE CALLED OUT TO COMPLETE THE COMMAND. FOR 90 DEGREE BENDS, MITERED CORNERS MAY BE USED INSTEAD OF RADII. SEE EXAMPLE #3.

POINTS MAY BE DEFINED IN X,Y COORDINATES OR CALLED OUT AS SPECIFIC COMPONENT PORTS. IF X,Y COORDINATES ARE USED, AN ANGLE DEFINING THE LINE ORIENTATION MUST ALSO BE SPECIFIED. THE FORMAT FOR COORDINATE POINTS IS:

[X, Y, ANGLE]

THE ANGLE, IN DEGREES, IS MEASURED FROM THE POSITIVE X AXIS WITH CCW ROTATION CONSIDERED POSITIVE. ANGLES MAY BE SPECIFIED AS EITHER + OR −. IF NO ANGLE IS SPECIFIED A ZERO DEGREE ORIENTATION IS ASSUMED. HOWEVER, IF THE ANGLE IS OMITTED ON BOTH POINTS, A STRAIGHT LINE CONNECT IS PRODUCED.



POINTS DEFINED AS COMPONENT PORTS HAVE THE FOLLOWING FORM:

COMPONENT NAME (PORT NUMBER)

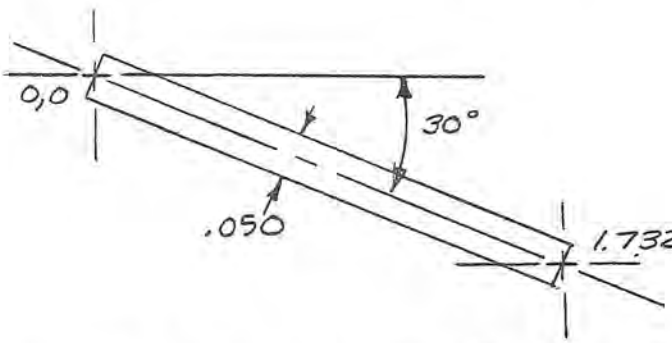
NOTE THAT BRACKETS ARE NOT NECESSARY.
IF THE SPECIFIED COMPONENT HAS ONLY ONE PORT, NO PORT NUMBER IS REQUIRED.

THE COMPONENT NAME CAN BE ANY ALPHANUMERIC COMBINATION OF UP TO 10 CHARACTERS. THE ONLY RESTRICTION IS THAT THE FIRST CHARACTER MUST BE A LETTER.

A SPECIAL FORM ? COMPONENT NAME WILL GIVE THE ORIGIN LOCATION OF THAT COMPONENT

EXAMPLES OF !CONNECT

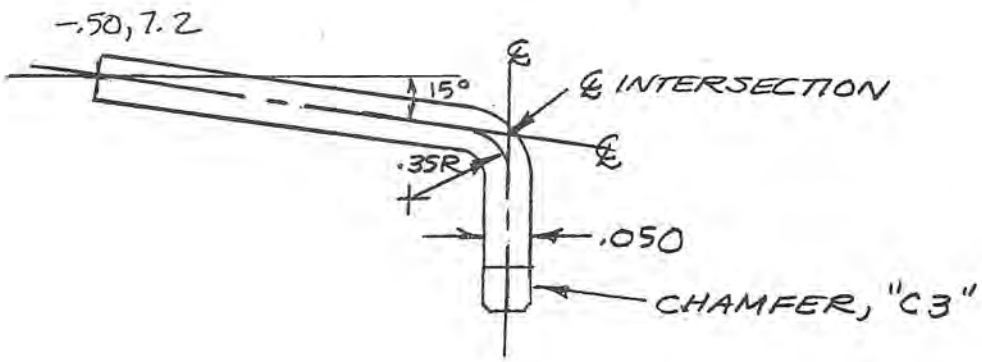
EXAMPLE #1 A STRAIGHT LINE



!CONNECT .050, 1.0 [0, 0, -30] [1.732, -1.0, 150]

NOTE THAT A RADIUS HAS BEEN SPECIFIED EVEN THOUGH THE LINE IS STRAIGHT.

EXAMPLE #2 A BENT LINE

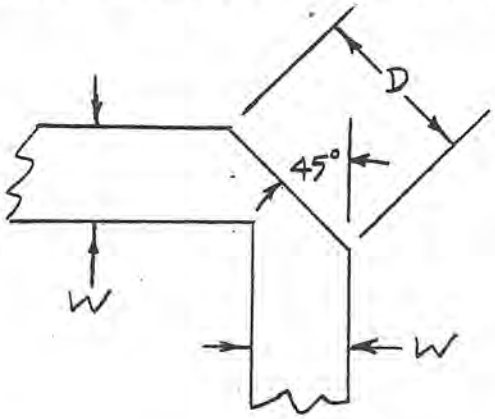


!CONNECT .050, .35 [-.50, 7.2, -15] C3

THE COMMAND DETERMINES THE POINT OF INTERSECTION OF THE PROJECTED CENTER LINES, THEN BLENDS THE LINES TOGETHER AT THIS POINT USING THE SPECIFIED RADIUS. (SINCE THE CHAMFER HAS ONLY ONE PORT NO PORT NUMBER NEED BE SPECIFIED).

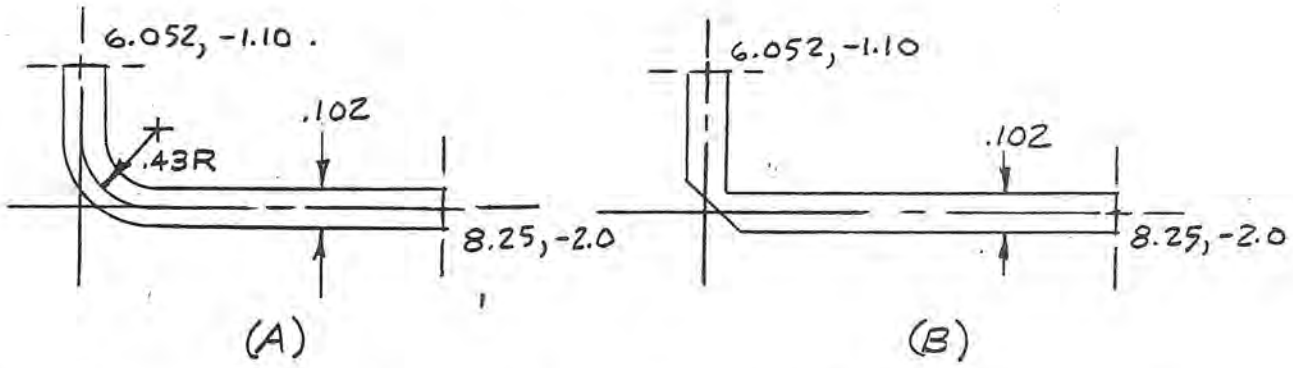
EXAMPLE 3 MITER BENDS

THE PROGRAM OFFERS THE OPTION OF USING A 90 DEGREE MITERED CORNER IN PLACE OF ANY 90 DEGREE BEND. TO DO THIS, A MITER FACTOR IS SPECIFIED INSTEAD OF A RADIUS. THE MITER FACTOR IS A USER SPECIFIED CONSTANT THAT DETERMINES THE DIAGONAL LENGTH OF THE CORNER AS SHOWN BELOW. (DO NOT USE THIS OPTION ON ANYTHING BUT A 90 DEGREE BEND!)



D = (MITER FACTOR) X W

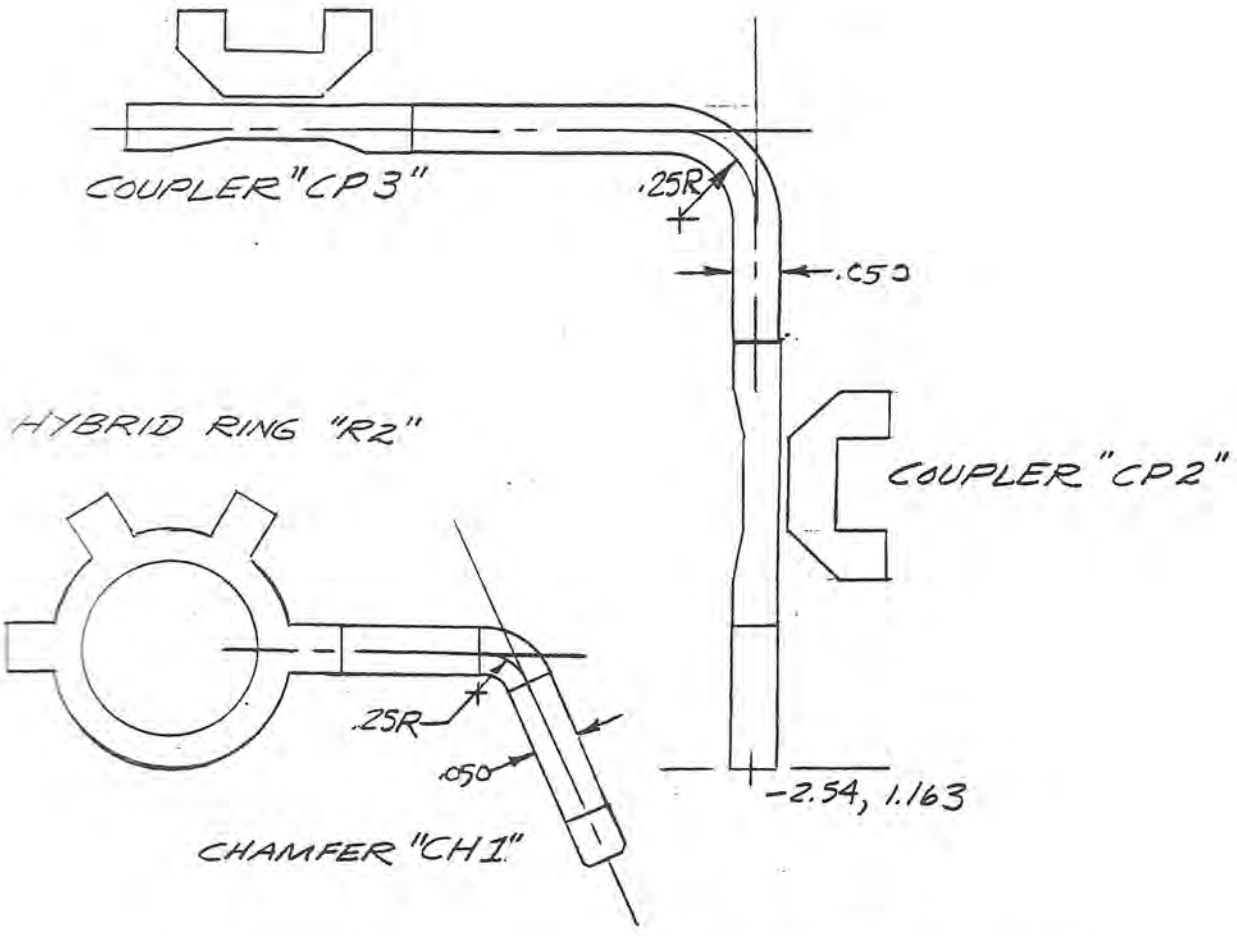
TYPICALLY THE MITER FACTOR IS 1.61. TO DIFFERENTIATE BETWEEN MITER FACTORS AND RADII A MINUS SIGN IS USED (I.E. MITER FACTORS ARE CALLED OUT AS NEGATIVE RADII).



(A) ! CONNECT .102, .43 [6.052, -1.10, 270] [8.25, -2.0, 180]

(B) ! CONNECT .102, -1.61 [6.052, -1.10, 270] [8.25, -2.0, 180]

EXAMPLE 4 MULTIPLE CONNECT STATEMENTS




```
! CONNECT  .05, .25  CP3(2)  CP2(3)
! CONNECT  .05, .25  R2(4)  CH1
! CONNECT  .05, .25  CP2(2)  [-2.54, 1.163, 90]
```

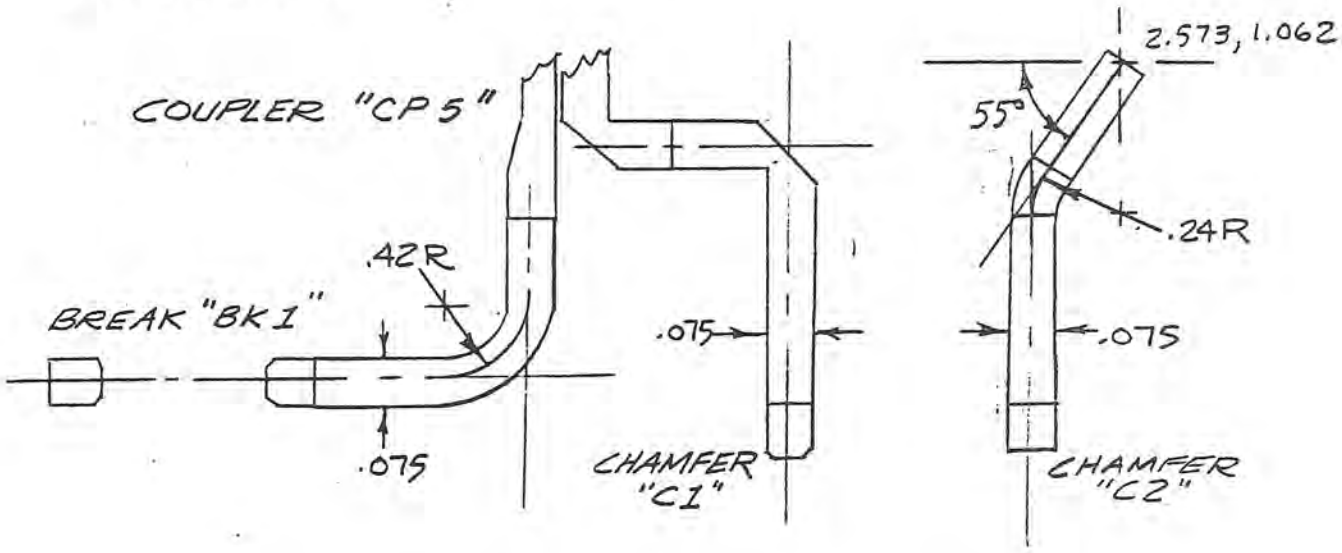
WHEN ! CONNECT COMMANDS CALL OUT THE SAME LINE WIDTHS THEY MAY BE COMBINED INTO A SINGLE STATEMENT. THE THREE ! CONNECTS SHOWN ABOVE MAY BE WRITTEN AS:

```
! CONNECT .05, .25 CP3(2) CP2(3) R2(4) CH1 CP2(2) [-2.54, 1.163, 90]
```

THE PROGRAM WILL AUTOMATICALLY GROUP THE SPECIFIED POINTS IN PAIRS AND MAKE EACH CONNECTION.

EXAMPLE 5 CHANGING RADII AND/OR MITERS

IN MULTIPLE CONNECT STATEMENTS, ALL THE CONNECTIONS MUST HAVE THE SAME LINE WIDTHS. HOWEVER, USING THE AMPERSAND (&) AND PARENTHESIS PERMITS REDEFINING THE RADIUS AND/OR MITER CONSTANTS.



SEPERATELY, THE ABOVE THREE CONNECTS ARE:

(A) !CONNECT .075, .42 BK1(1) CP5(2)
(B) !CONNECT.075, -1.61 CP5(1) C1
(C) !CONNECT.075, .24 C2 [2.573, 1.062, -125]

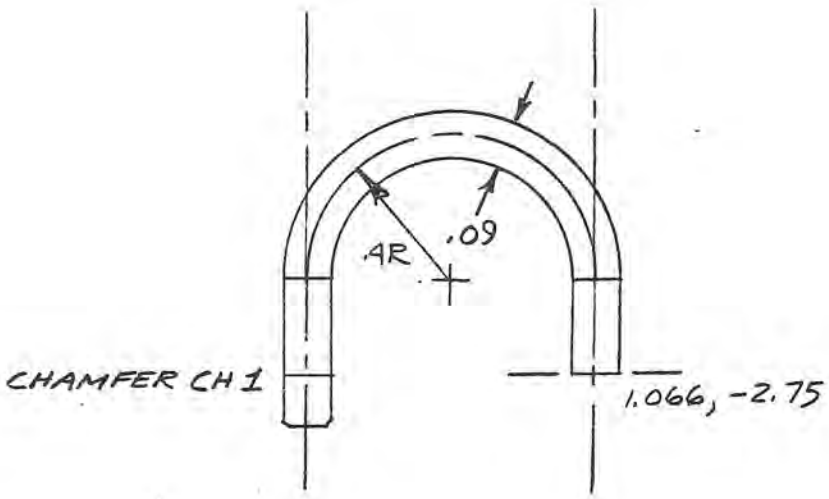
REWRITTEN AS A SINGLE STATEMENT:

!CONNECT .075, .42 BK1(1) CP5(2) & (-1.61) CP5(1) C1

& (.24) C2 [2.573, 1.062, -125]

EXAMPLE 6 SPECIAL CASE - CANNOT BE DONE WITH !CONNECT

THE FOLLOWING CONFIGURATION, EVEN THOUGH IT HAS ONLY ONE BEND, CANNOT BE MADE USING THE !CONNECT, SINCE THE PROJECTED CONNECTION POINT CENTERLINES DO NOT INTERSECT (CF. EXAMPLE 2)



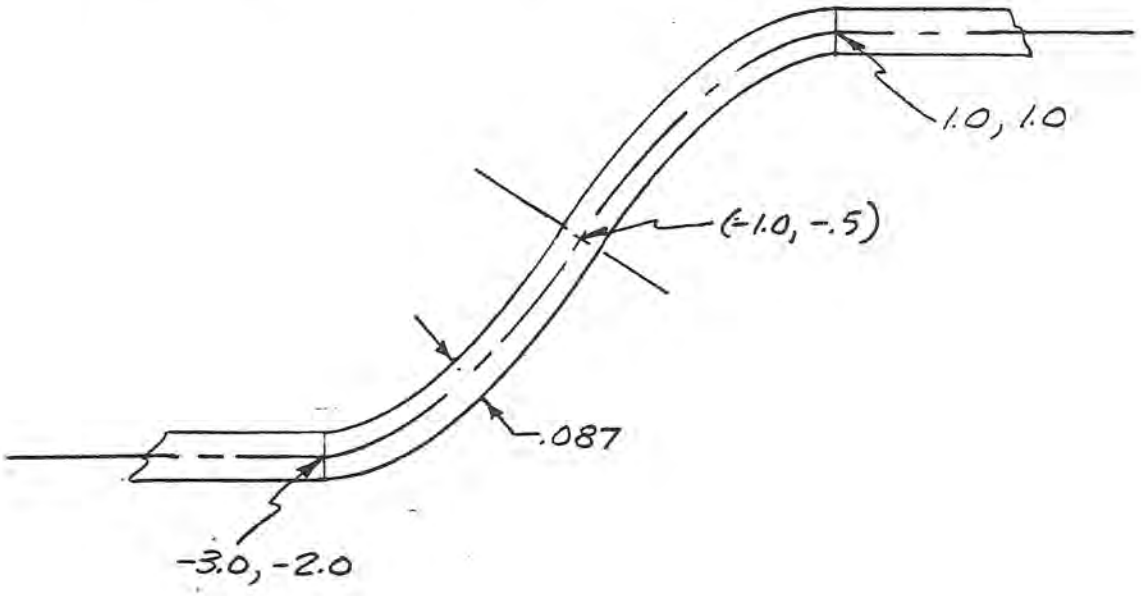
!CONNECT .09, .4 CH1 [1.066, -2.75, 90] WILL NOT WORK.

RULE:

FOR ANY CASE WHERE THE PROJECTED POINT CENTER LINES DO NOT INTERSECT, THE !CLINE COMMAND MUST BE USED. NATURALLY, THERE IS A POSSIBLE EXCEPTION TO THIS RULE. SEE EXAMPLE 7.

EXAMPLE 7 SPECIAL CASE - TWO BEND CONNECT

IN EXAMPLE #6 BOTH LINES WERE "RUNNING" IN THE SAME DIRECTION (I.E. +90°). IF THE LINES "RUN" IN OPPOSITE DIRECTIONS, ANOTHER SPECIAL CASE EVOLVES.



EVEN THOUGH THE PROJECTED CENTER LINES DO NOT INTERSECT, !CONNECT WILL JOIN THE TWO POINTS USING A MINIMUM DISCONTINUITY OGEE CURVE.

```
!CONNECT .087,.5 [-3.0,-2.0][1.0,1.0,180]
```

NOTE THAT A RADIUS, WHILE SPECIFIED, IS NOT USED IN THE CONNECT CONSTRUCTION. THIS IS THE ONLY CASE WHERE THE RADIUS IS IGNORED IN A CURVED LINE.

IF THIS CONNECTION SCHEME IS NOT SATISFACTORY FOR YOUR PURPOSES, USE !CLINE INSTEAD.

!CLINE

!CLINE IS THE GENERAL CONNECTION COMMAND TO BE USED WHENEVER !CONNECT IS NOT ADEQUATE. THEREFORE, USE !CLINE IF THE CONNECTION HAS MORE THAN ONE BEND OR IS REQUIRED IN A SPECIAL CASE (SEE EXAMPLES *6 AND *7). THE FORM OF !CLINE IS:

!CLINE W,R [POINT A][POINT B][POINT C] .... [POINT N]

WHERE W = LINE WIDTH
R = BEND RADIUS
[POINT A] = STARTING POINT
[POINT B]
[POINT C]
⋮
} = INTERMEDIATE CONNECTION POINTS
[POINT N] = ENDING POINT
N = NUMBER OF BENDS + 2

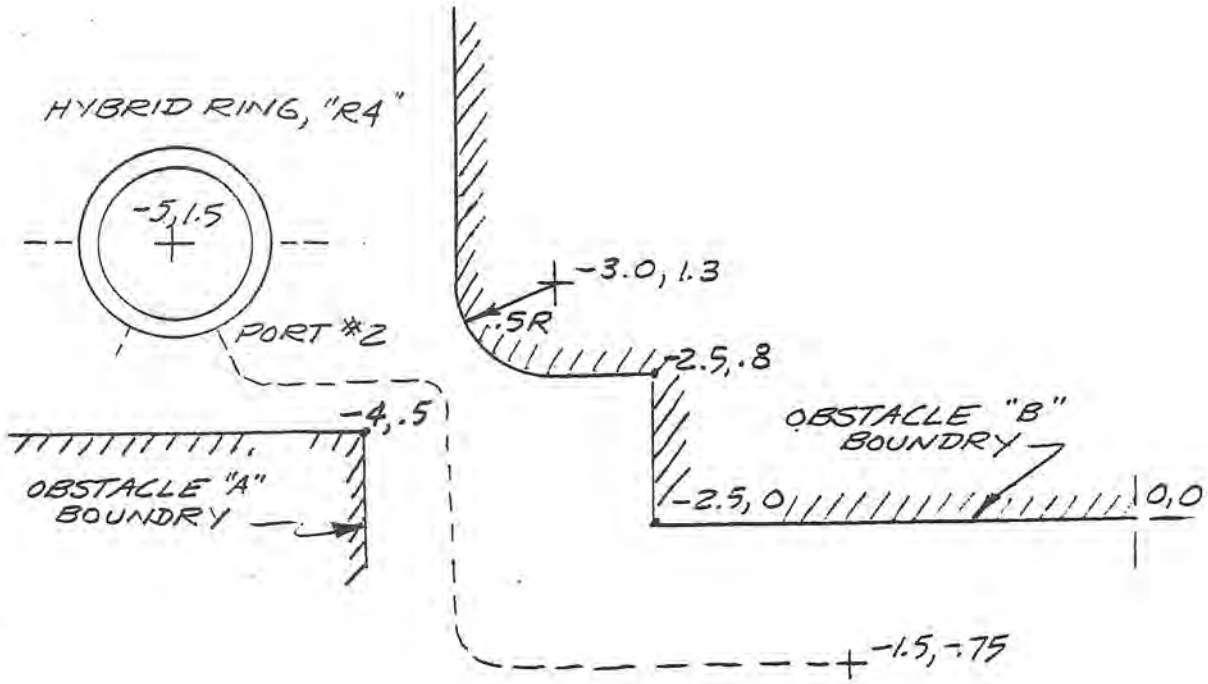
NOTE: 1 - COMPONENT PORTS DO NOT REQUIRE BRACKETS.
2 - INTERMEDIATE POINTS REQUIRE X,Y COORDINATES ONLY
3 - N = NUMBER OF BENDS + 2 IS A SACRED RULE. DO NOT VIOLATE IT.

!CLINE WILL CONNECT TOGETHER ALL THE SPECIFIED POINTS USING THE DESIGNATED LINE WIDTH AND RADIUS. THE START AND END POINTS ARE DEFINED AS IN !CONNECT, WITH EITHER COMPONENT PORTS OR X, Y COORDINATES AND ORIENTATION ANGLE. IF EITHER OF THESE START OR ENDING ANGLES IS DESIGNATED AS 999, BAA WILL COMPUTE

THE ANGLE FROM THAT POINT TO THE NEXT/PRECEEDING POINT FOR YOU. INTERMEDIATE POINTS DO NOT REQUIRE ORIENTATION ANGLES.

BAA USES THE SPECIFIED POINTS TO GENERATE A NETWORK OF CENTERLINES. THEN IT USES THE CENTERLINE INTERSECTION POINTS TO CONFIGURE THE DESIRED LINE. UNLESS THE LINE MUST FOLLOW AN EXACT, SPECIFIC PREDETERMINED PATH, WE SUGGEST YOU NOT BOTHER TO COMPUTE THE APPROPRIATE INTERSECTION POINTS. INSTED, ESTIMATE (OR GUESS) AND LET THE COMPUTER DO THE TRIGONOMETRY AND LINE LAYOUT FOR YOU. THEN, OBSERVE THE RESULTANT LINE ON THE TERMINAL SCREEN OR XYNETIC PLOT, AND ASSESS ITS SUITABILITY. IF IT'S NOT SATISFACTORY, THIS LAYOUT CAN BE USED TO DETERMINE THE POINT MODIFICATIONS NECESSARY. SEVERAL ITERATIONS CAN BE QUICKLY CHECKED IN LESS TIME THAN IT WOULD TAKE TO PERFORM THE COMPUTATIONS MANUALLY. THIS IS PARTICULARLY TRUE FOR LINES THAT MUST BE OF A PREDETERMINED LENGTH (SUCH AS FOR PHASE BALANCING) WHERE THE !LENGTH AND !TRACE COMMANDS CAN BE USED TO QUICKLY SEE THE ACTUAL LINE LENGTH.

AN EXAMPLE OF !CLINE



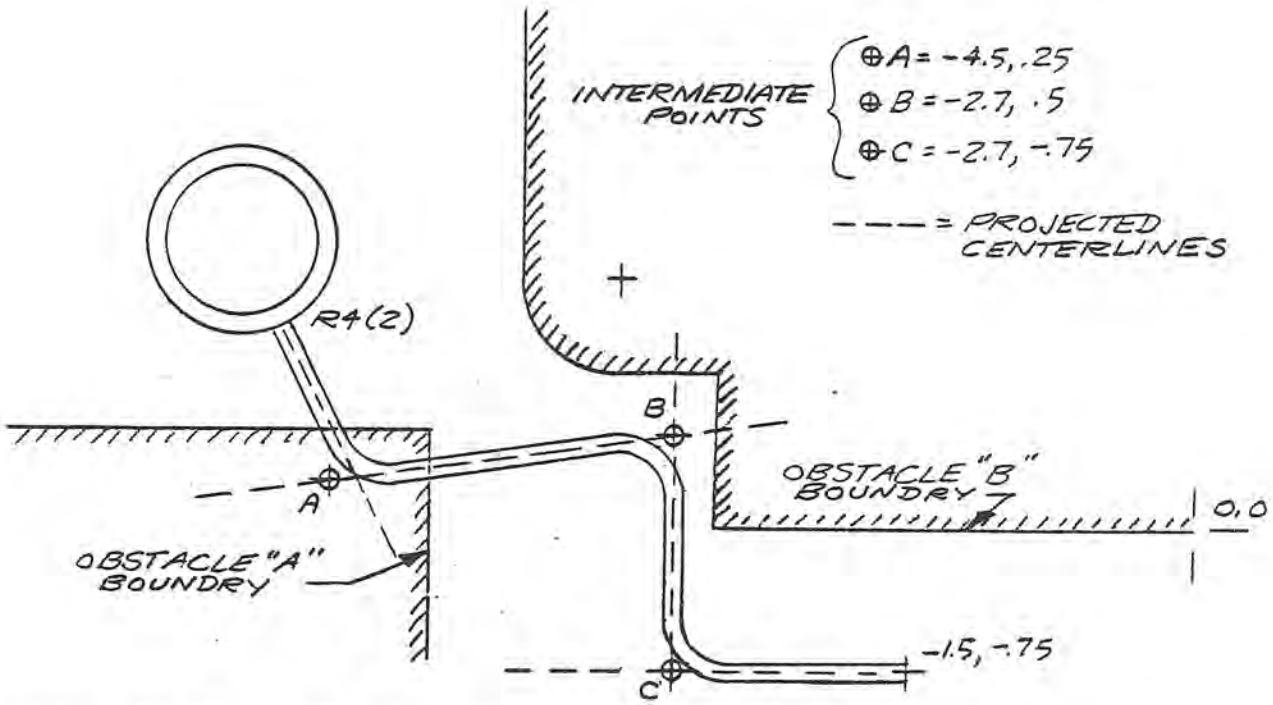
WE WISH TO CONNECT "R4, PORT 2" TO THE POINT "-1.5, -.75". WE ALSO WANT TO AVOID THE OBSTACLES SHOWN, AND APPROACH THE SPECIFIED POINT AT AN ANGLE OF 180 DEGREES.

WE DECIDE TO CONFIGURE THE CONNECTION AS SHOWN BY THE DASHED LINE. THERE ARE 3 BENDS SO 5 POINTS MUST BE SPECIFIED, BUT, SINCE WE KNOW THE BEGINNING AND END POINTS, ONLY THE 3 INTERMEDIATE POINTS MUST BE DETERMINED. CONNECTING THE 5 POINTS WILL DEFINE CENTERLINES UPON WHICH THE STRAIGHT SEGEMENTS OF THE CONNECTION WILL BE PLACED. BENDS WILL BE LOCATED AT THE INTERSECTIONS OF THESE CENTERLINES. WE GUESS A RADIUS OF .300, ESTIMATE THE 3 INTERMEDIATE POINTS AND TRY THE FOLLOWING CASE:

(1ST ESTIMATE)

```
!CLINE .100,.300 R4(2)[-4.5,.25][-2.7,.5][-2.7,-.75][-1.5,-.75,180]
```

AN EXAMPLE OF !CLINE (CON'T)



CONNECTION PRODUCED BY:

!CLINE .100, .300 R4(2) [-4.5, .25][-2.7, .5][-2.7, -.75][-1.5, -.75, 180]

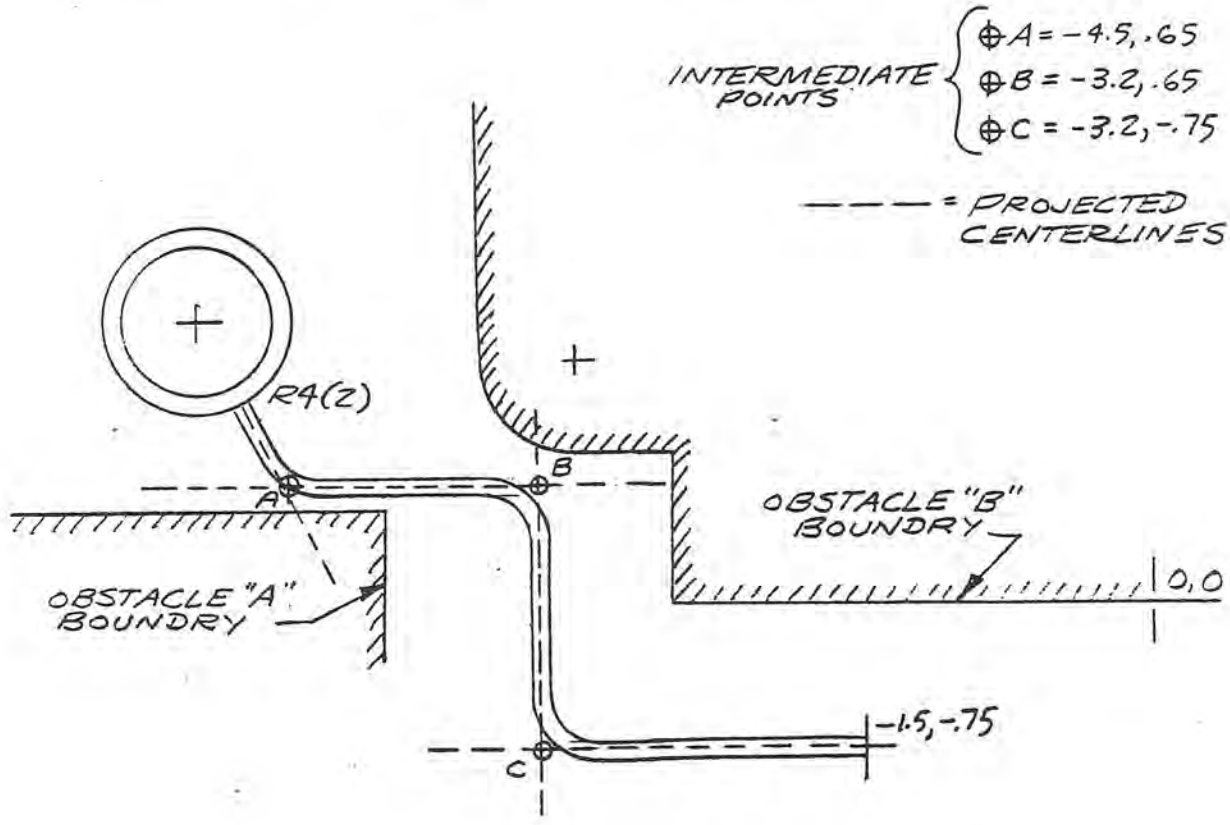
(THE NOTATIONS, CENTERLINES, AND INTERMEDIATE POINTS WILL NOT APPEAR ON THE TERMINAL SCREEN OR XYNETIC PLOT. THEY ARE SHOWN ABOVE ONLY TO INDICATE HOW !CLINE CONSTRUCTS THE INTERCONNECTION. FOR EXAMPLE, NOTICE THAT THE BENDS ARE LOCATED AT THE PROJECTED CENTERLINE INTERSECTIONS, THAT THE STATEMENT POINTS DEFINE THESE CENTERLINES, AND THAT POINT C COULD BE CHANGED FROM [-2.7, -.75] TO [-2.7, 0.0] WITHOUT ALTERING THE INTERCONNECTION SHOWN IN ANY WAY.)

OUR FIRST ESTIMATE PRODUCED A LAYOUT THAT HAS TWO DEFECTS: THE RESULTING LINE PENETRATES OBSTACLE "A", AND THE SEGEMENT "B-C" MAY BE TOO CLOSE TO OBSTACLE "B". WE DECIDE TO MOVE POINTS A AND B UP, AND TO BRING POINTS B AND C TO THE LEFT BY TRYING THE FOLLOWING:

(2ND ESTIMATE)

!CLINE .100, .300 R4(2) [-4.5, .65][-3.2, .65][-3.2, -.75][-1.5, -.75, 180]

AN EXAMPLE OF !CLINE (CON'T)



CONNECTION PRODUCED BY:

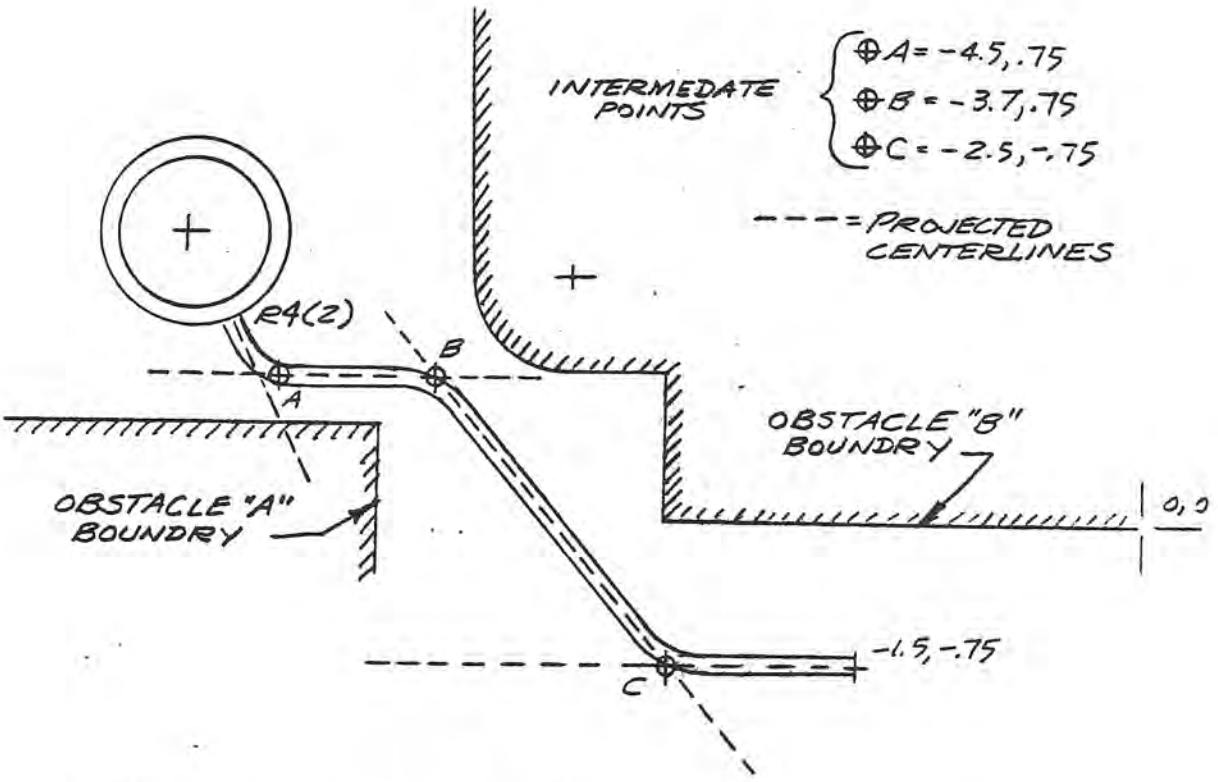
!CLINE .100,.300 R4(Z) [-4.5,.65][-3.2,.65][-3.2,-.75][-1.5,-.75,180]

THIS STATEMENT HAS PRODUCED A LINE THAT COMES TOO CLOSE TO OBSTACLE "A", HOWEVER, IF WE MOVE POINTS A AND B MUCH HIGHER, THE LINE GETS TO CLOSE TO OBSTACLE "B". WE DECIDE TO MOVE THIS LINE SEGEMENT UP (TO CLEAR OBSTACLE "A") AND REPOSITION POINT B TO THE LEFT (CLEARING OBSTACLE "B"). IN DOING THIS, WE GUESS THAT THE LINE WILL "LOOK" BETTER IF WE MOVE POINT C TO THE RIGHT AND INCREASE THE RADII OF THE SECOND AND THIRD BENDS. WE TRY:

(3RD ESTIMATE)

!CLINE .100,.300 R4(Z) [-4.5,.75] & (.45) [-3.7,.75][-2.5,-.75][-1.5,-.75,180]

AN EXAMPLE OF !CLINE (CON'T)



CONNECTION PRODUCED BY:

!CLINE .100,.300 R4(2) [-4.5,.75] & (.45) [-3.7,.75] [-2.5,-.75] [-1.5,-.75,180]

WE FEEL THAT THIS LINE IS SATISFACTORY, SO THIS CONNECTION IS FINISHED. THE !TRACE ROUTINE CAN BE USED TO FIND THE ARC CENTERS AND LENGTHS AS WELL AS THE TOTAL LENGTH OF THE LINE.

USER DEFINED ELEMENTS

USER DEFINED ELEMENTS WERE CREATED TO ALLOW THE USER TO GENERATE THEIR OWN BAA-LIKE ELEMENTS USING BAA COMMANDS. THE ELEMENTS, REFERRED TO AS UDE'S WILL HAVE ALL THE PROPERTIES OF REGULAR PORTED BAA ELEMENTS. THIS MEANS THAT THEY CAN BE ROTATED, OFFSET, AND MIRRORED, HAVE NAMES ASSOCIATED WITH EACH, AND HAVE PORTS ASSIGNED TO THEM.

DEFINE

FORM: !DEFINE NNN (TURNS FUNCTION ON)

WHERE NNN.... = THE NAME (UP TO 10 CHARACTERS) SELECTED BY THE USER.

FUNCTION: PERMITS THE USER TO CONFIGURE A "SUB-CIRCUIT" FROM THE STATEMENTS INCLUDED BETWEEN !DEF NNN... AND !ENDDEF NNN.... THE RESULTING "SUB-CIRCUIT" IS TREATED AS A SINGLE CIRCUIT ELEMENT BY USING ITS NAME AND THE !GET COMMAND.

NOTES: !DEFINE ROUTINES MAY NOT BE NESTED, HOWEVER A !GET MAY BE USED WITHIN A !DEFINE.

!DEFINE RESETS THE GLOBAL OFFSET, ROTATE AND MIRROR VALUES TO ZERO FOR THE DURATION OF THE DEFINITION.

ENDDEF

FORM: !ENDDEF NNN NP [X,Y,A] [X_2,Y_2 A_2] ... [X_n,Y_n,A_n]

WHERE: NNN = NAME OF SUBCIRCUIT OPENED BY THE DEFINE CARD

NP = NUMBER OF PORTS TO BE ASSOCIATED WITH SUBCIRCUIT

[X,Y,A] = EITHER AN X,Y,A LOCATION OR A COMPONENT PORT NAME

FUNCTION: USED TO CLOSE A SUBCIRCUIT DEFINITION AND ASSIGN PORTS TO THE SUBCIRCUIT

EXAMPLE: !ENDDEF SCKT 3 [4.,3.5,-45] R1(3) ?C3

SUBCIRCUIT NAME

OF PORTS FOR SUBCIRCUIT

PORT 1 OF THIS SUBCIRCUIT IS THIS POINT

PORT 2 OF THIS SUBCIRCUIT IS PORT 3 OF DEVICE R1

PORT 3 OF SUBCIRCUIT IS THE ORIGIN LOCATION OF DEVICE C3

WHEN THE SUBCIRCUIT IS RETRIEVED (SEE !GET COMMAND) THE PORTS ARE RELOCATED AND ENTERED INTO THE PORT TABLE.

NOTE:

1) THE !ENDDEF CARD WILL REMOVE ALL PORTS WHICH WERE CREATED DURING THE SUBCIRCUIT DEFINITION. IN OTHER WORDS, THE PORT TABLE AFTER EXECUTION OF THE !ENDDEF CARD WILL LOOK LIKE IT DID BEFORE THE !DEFINE CARD WAS EXECUTED.

2) THE !ENDDEF CARD ALSO RESTORES THE GLOBAL ROTATE, OFFSET, AND MIRROR VALUES BACK TO WHAT THEY WERE BEFORE THE !DEFINE CARD SET THEM TO ZERO.

GET

FORM: !GET NNN... (NAME [X_1, Y_1, A_1]).... (NAME [X_n, Y_n, A_n])

WHERE: NNN... = THE NAME OF A CIRCUIT CONFIGURATION DESCRIBED IN A DEFINE (!DEF) ROUTINE

NAME = NAME TO BE ASSOCIATED WITH RETRIEVED CIRCUIT

X = X LOCATION COORDINATE WHERE DEFINED CIRCUIT IS TO BE PLACED.

Y = Y LOCATION COORDINATE WHERE DEFINED CIRCUIT IS TO BE PLACED.

A = ANGULAR ORIENTATION OF DEFINED CIRCUIT

FUNCTION: !GET ALLOWS THE RETRIEVAL AND IMPLEMENTATION (AT X, Y, A) OF A "SUB CIRCUIT" NAMED AND DELINEATED BY A !DEFINE COMMAND.

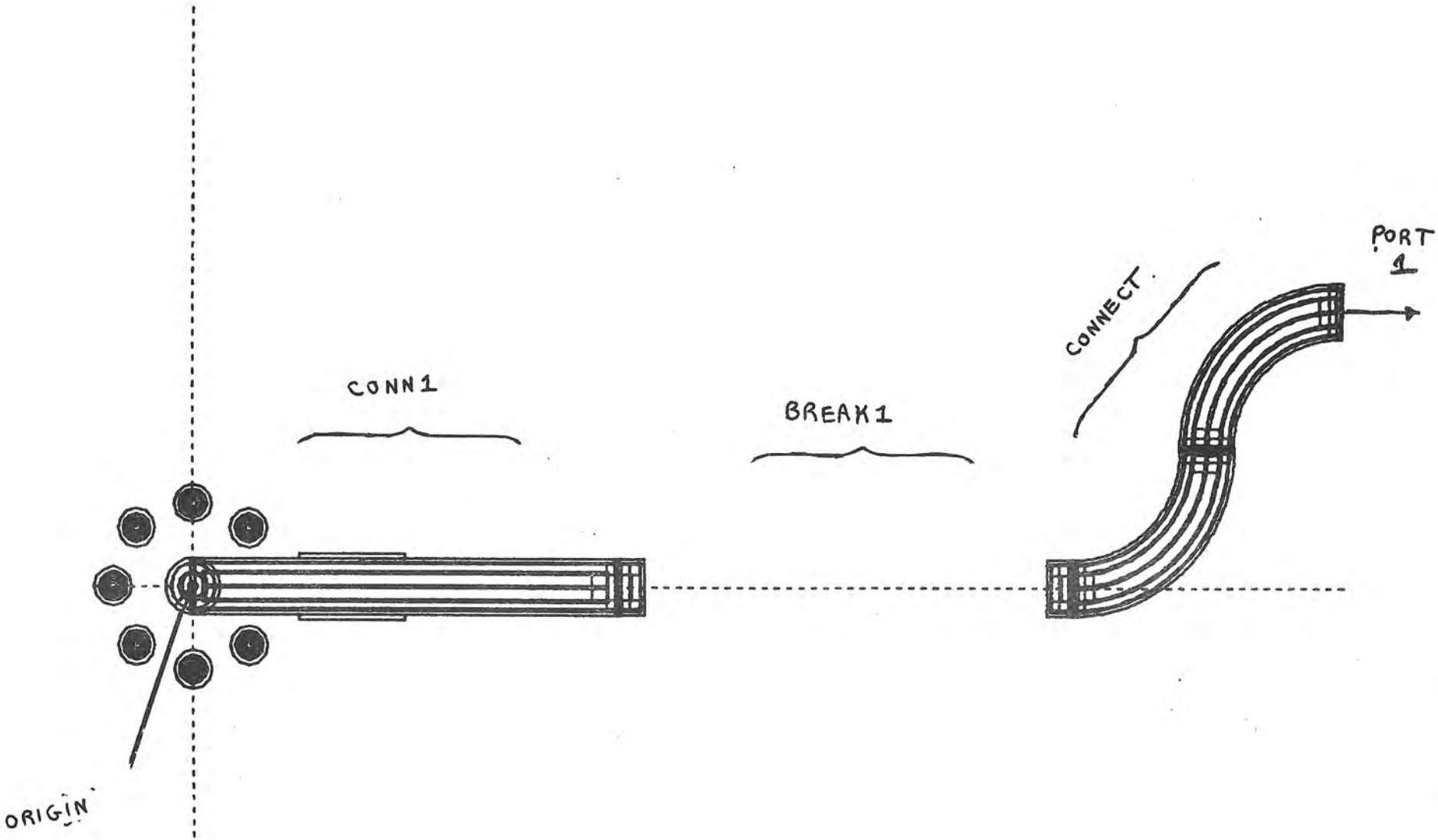
```
!NAME CKT
*TEST CIRCUIT FOR BAA 2.0
!LET W=.1
!LET R=3*W
!DEFINE ARM
!CONN1 W,.102,.3,.07,45,45,7,.2,.2,.120
       (C1[0,0,0])
!BREAK1 W,0,.375,180 (B1[1.225,0,0])
!CONNECT W,R C1 B1(2)
            B1(1) [2.15,.5,180]
!DUMP
!ENDDEF ARM 1 [2.15,.5,0]
!DEFINE HALF
!RATEQ W,.056,.371,(R1[0,0,30])
!GET ARM (A1[.275,2.45,-90])
!MIRROR Y
!GET ARM (A2[.275,2.45,-90])
!MIRROR Y
!CONNECT W,R R1(2) A1
            R1(4) A2
!DUMP
!ENDDEF HALF 2 R1(3) R1(1)
!RATEQ W,.056,.371 (R1[0,0,-30])
!GET HALF (H1[.65,.6625,0])
         (H2[-.65,.6625,0])
!CONNECT W R R1(1) H1(1)
            R1(3) H2(1)
!DUMP
!END
!STOP

READY.
```

ASSIGN VALUES TO "W" AND "R"

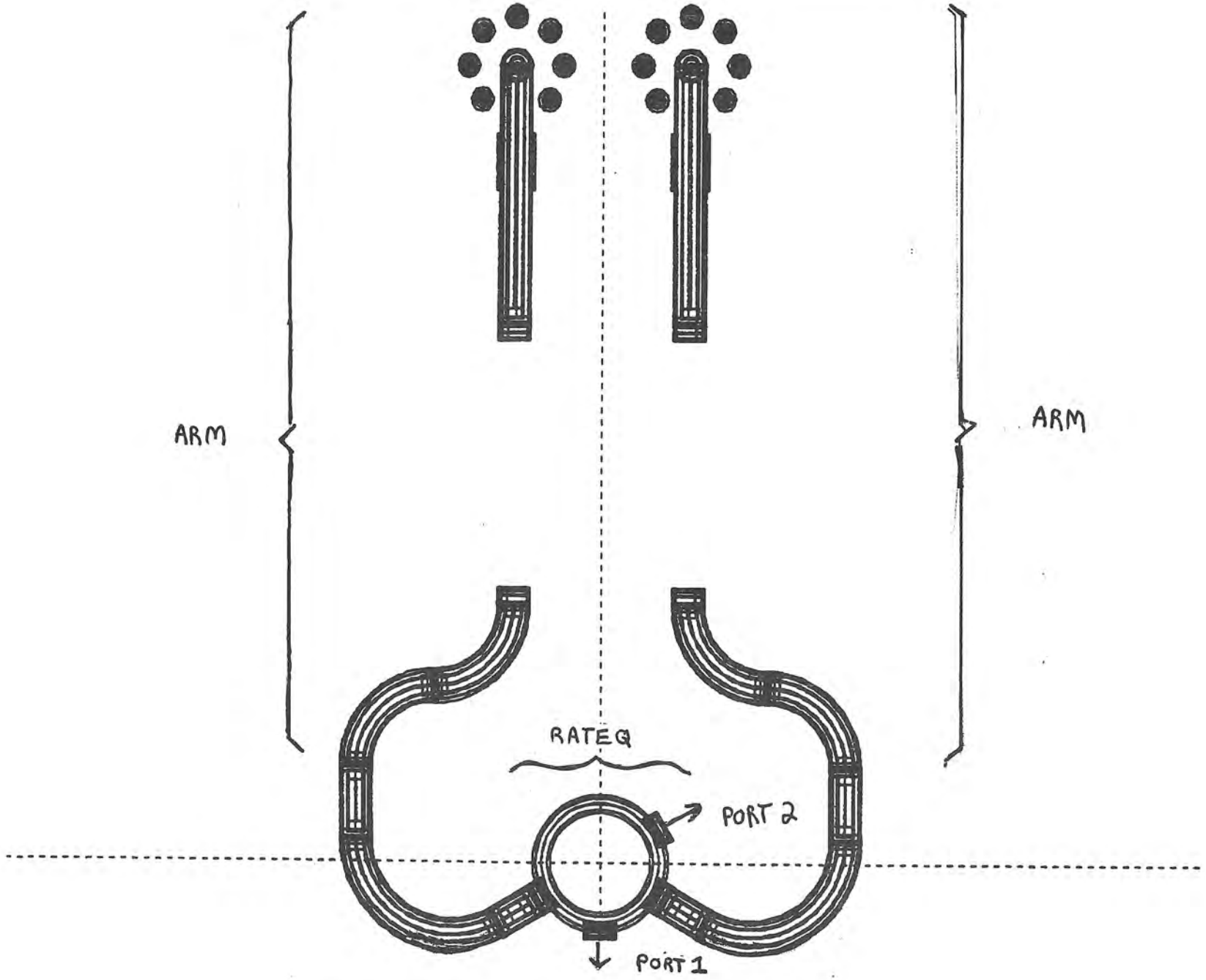
CREATE DEFINITION CALLED "ARM"
ARM HAS ONE PORT
AT [2.15, .5, 180]

CREATE DEFINITION OF "HALF" USING "ARM"
"HALF" HAS TWO PORTS
ONE AT R1(1) AND THE OTHER AT R1(1)

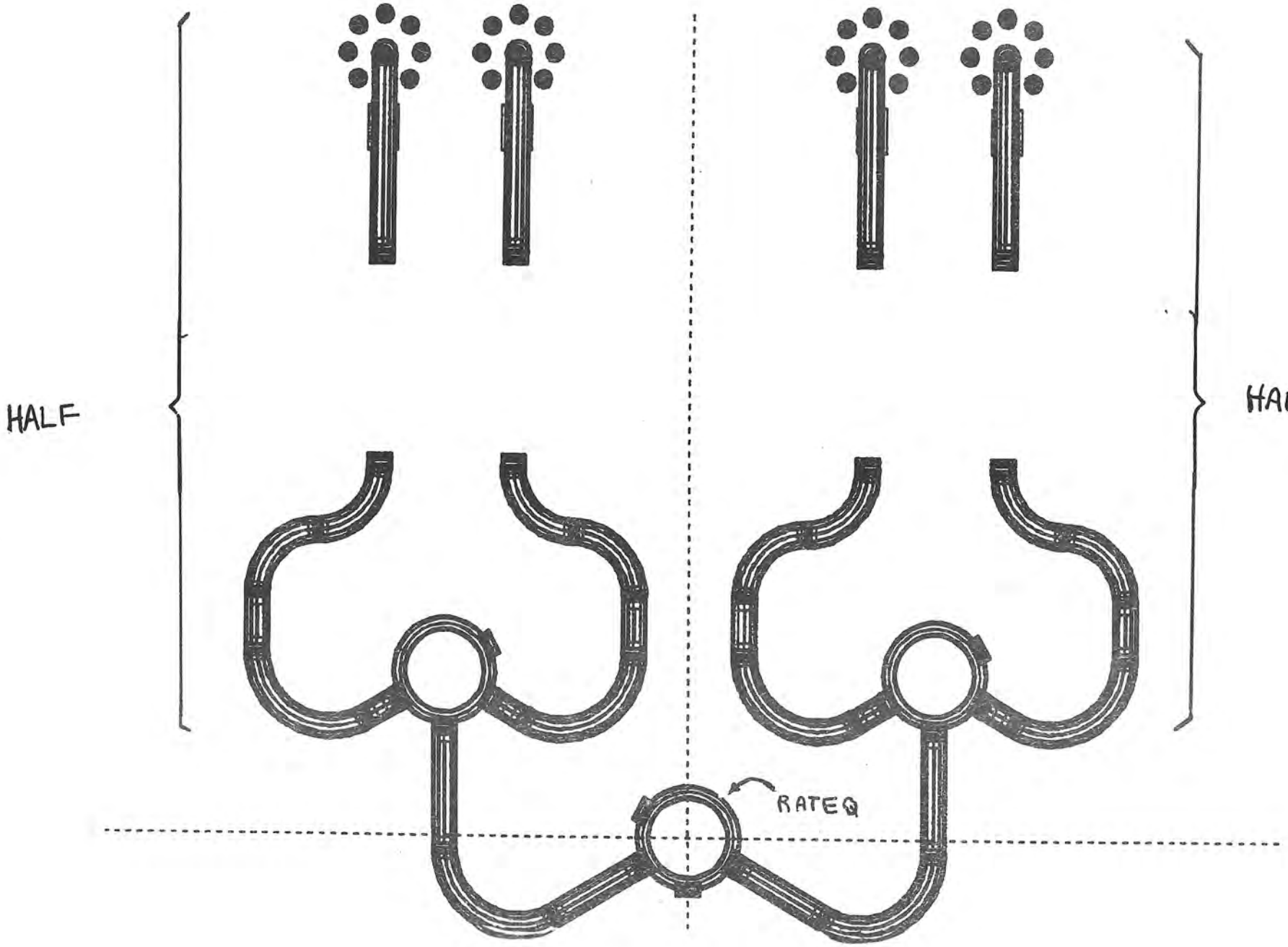
COMPLETE CIRCUIT BY USING "HALF"



USER DEFINED ELEMENT "ARM"



USER DEFINED ELEMENT "HALF"



FINAL CIRCUIT

page 78

MATRIX

FORM !MATRIX NAME, CS, RS, NC, NR [X,Y,A] [X_2,Y_2,A_2] ... [X_n,Y_n,A_n]

WHERE NAME = NAME OF THE MATRIX (UP TO 10 CHARACTERS)
CS = SPACING BETWEEN THE COLUMNS
RS = SPACING BETWEEN THE ROWS
NC = NUMBER OF COLUMNS
NR = NUMBER OF ROWS
X,Y,A = X, Y COORDINATES AND ANGLE OF ROTATION OF THE INITIAL COMPONENT OF THE MATRIX

FUNCTION: TO GENERATE A MATRIX OF USER-DEFINED ELEMENTS

NOTES: THE COLUMN SPACING AND THE ROW SPACING WILL REMAIN CONSTANT REGARDLESS OF ANGLE OF ROTATION OR OFFSET.

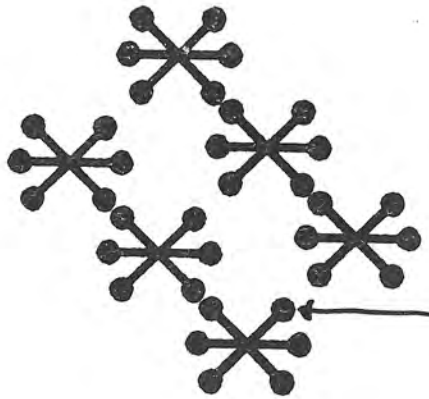
THE FOLLOWING IS AN EXAMPLE OF THE USE OF THE MATRIX COMMAND

```
!NAME STORM
!DEFINE BAR
!CIRCLE .1,.1 [-1.,0.]
!CIRCLE .1,.1 [0.,0.]
!CONNECT .05,.1 [-1.,0.] [0.,0.]
!ENDDEF BAR 1 [0.,0.,0]
!DEFINE FLAKE
!GET BAR (A1[0.,0.,0])
!ROTATE 90
!OFFSET -.5,.5
!GET BAR (A2[0.,0.,0])
!DUMP
!ROTATE 45
!OFFSET -.83495,.3495
!GET BAR (A3[0.,0.,0])
!DUMP
!ENDDEF FLAKE 1 [0.,0.,0]
!MATRIX FLAKE 1.5,1.25,2,3,[-3,-3,0] [8,6,45]
!END
!STOP
END OF FILE
??
```

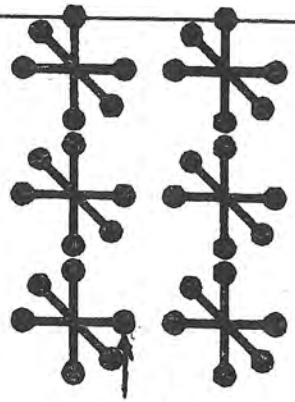
USER DEFINED ELEMENT

FLAKE

MATRIX EXAMPLE



(8,6)



(-3,-3)

MATRIX EXAMPLE

SECTION IV

COMMANDS

BAPER

FORM:!BAPER XXX..

WHERE XXX... = APERTURE SIZE

FUNCTION: PERMITS THE USER TO SPECIFY THE GERBER APERTURE SIZE USED TO DRAW COMPONENT PERIMETERS.

NOTES: BAPER HAS BEEN PRESET TO .000 SO THAT NORMALLY THE SMALLEST APERTURE SIZE (.010) IS ALWAYS USED. IF THIS DOES NOT SUIT YOUR REQUIREMENTS, ANY OF THE FOLLOWING APERTURE SIZES MAY BE USED:

.010	.013	.015	.016
.020	.025	.030	.050
.060	.100	.120	

IF YOU SPECIFY AN APERTURE SIZE THAT IS NOT SHOWN ABOVE, BAA WILL SELECT THE NEXT SMALLEST SIZE.

BAPER IS AFFECTED BY THE SCALE COMMAND, THEREFORE, TAKE THIS INTO ACCOUNT WHEN SPECIFYING AN APERTURE.

CLEAR

FORM: !CLEAR NAME

FUNCTION: THIS COMMANDS CLEARS THE TABLES INDICATED BY "NAME"

NOTES: CLEARING THE APPROPRIATE TABLE IS IMPORTANT IF THE SAME PORT, VARIABLE OR DEFINE NAMES ARE USED IN TWO DIFFERENT CIRCUIT DESCRIPTIONS (FILES). USING CLEAR WILL ELIMINATE PROBLEMS ARISING FROM USING SIMILAR PORT, VARIABLE, OR DEFINE NAMES WHEN MULTIPLE FILES ARE RUN.

WE SUGGEST THAT IT IS GOOD PRACTICE TO USE A !CLEAR COMMAND IMMEDIATELY AFTER THE !NAME FOR EACH FILE GENERATED.

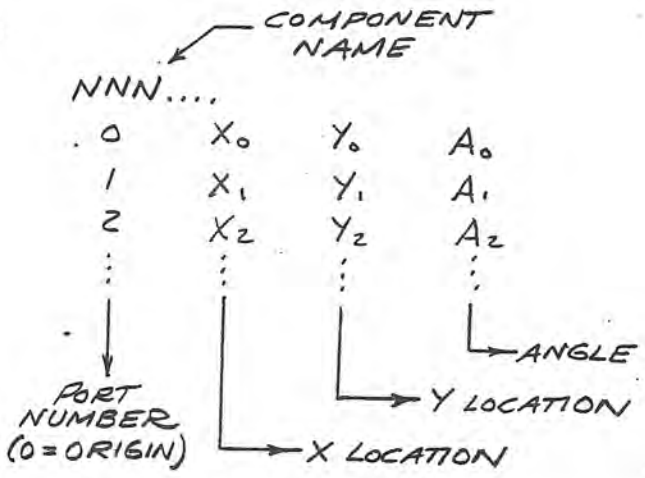
NAME = PORTS — PORT TABLE
VARIABLES - VARIABLE SYMBOL TABLE
DEFINES - USER DEFINED ELEMENT TABLE
ALL - ALL OF THE TABLES

DUMP

FORM: !DUMP

FUNCTION: PRINTS THE LOCATION OF THE ORIGIN AND PORTS FOR EACH NAMED COMPONENT IN THE STATEMENTS UP TO THE POINT WHERE DUMP IS USED.

NOTES: THE DUMP MODE IS ACTIVATED AFTER EACH STATEMENT THAT CONTAINS NAMED COMPONENTS. THE FORMAT USED IS:



END

FORM: !END

FUNCTION: TERMINATES A CIRCUIT DESCRIPTION, BUT NOT A BAA "RUN" (E.G. END WOULD BE USED TO SEPARATE THE BAA DISCRIPTIONS OF LAYER 3 FROM LAYER 4 IF THEY WERE PROGRAMMED SEQUENTIALLY).

ETCHFACTOR

FORM: !ETCHFACTOR XXX...

WHERE: XXX... = ETCH FACTOR

FUNCTION: PERMITS THE USER TO AUTOMATICALLY ADJUST ALL EXISTING COMPONENT AND LINE WIDTHS TO ACCOMODATE FOR SELECTED ETCH FACTOR. ETCH FACTOR, AS DEFINED AND USED HERE IS THE TOTAL ETCH FACTOR (I.E. ONE-HALF THE ETCH FACTOR IS ADDED TO EACH SIDE OF A CIRCUIT LINE).

EXTEND

FORM: !EXTEND XXX..

WHERE XXX... = EXTENSION DIMENSION

FUNCTION: PERMITS THE USER TO SPECIFY THE EXTENSION DIMENSION [Valve preset to zero (Ø)]

NOTES: WHAT IS THE EXTENSION DIMENSION? WELL, IN EARLY BAA COMPONENT PORTS AND LINE ENDS WERE EXTENDED SLIGHTLY (.005) TO INSURE COMPONENTS AND LINE SEGMENTS "KNITTED" TOGETHER PROPERLY. IT TURNED OUT THAT, NOT ONLY WAS THIS EXTENSION UNNECESSARY, IT CAUSED HEADACHES AT LARGE SCALE FACTORS. HENCE, IT IS NOW PRESET TO ZERO, ALTHOUGH YOU STILL HAVE THE OPTION OF CHANGING IT IF THIS APPEARS TO BE DESIRABLE FOR A SPECIAL CASE.

EXTEND IS AFFECTED BY THE SCALE COMMAND.

KILL

FORM: !KILL NNN....

WHERE NNN... = NAME (OPTIONAL)

FUNCTION: SEE NOTE

NOTE: !KILL DOES NOT EXIST IN BAA AND HENCE IS AN

ILLEGAL COMMAND!

NEVERTHELESS, THE USER IS FREE TO USE IT, OR ANY OTHER OF HIS/HER LIKING, IF IT IS FELT IT WILL RELIEVE TENSION AND/OR FRUSTRATION. ACTUALLY, IT WILL PROBABLY MAKE YOU LOOK SILLY IN THE SIGHT OF BOTH PEERS AND UNDERLINGS. IN FACT, YOU ARE SILLY TO HAVE READ THIS.

LENGTH

FORM: !LENGTH 1 (TURNS FUNCTION ON)
!LENGTH Ø (TURNS FUNCTION OFF)

FUNCTION: PRINTS THE RESULTANT TOTAL CENTERLINE LENGTHS FOR EACH CLINE AND CONNECT STATEMENT INCLUDED BETWEEN !LENGTH 1 AND !LENGTH Ø.

NOTES: EACH CLINE STATEMENT IS FOLLOWED BY ITS LENGTH. MULTIPLE CONNECT STATEMENTS ARE FOLLOWED BY THE LENGTHS FORMED BY THE SPECIFIED COORDINATE PAIRS. THESE LENGTHS ARE NUMBERED AND LISTED SEQUENTIALLY.

THE LENGTH FUNCTION IS NOT SUPPRESSED BY PRINT Ø

THE INITIAL STATE OF LENGTH IS OFF.

```
|--------|
|  LET   |
|--------|
```

VARIABLE DEFINITION AND CALCULATION

FORM : !LET NAME = EXPRESSION

WHERE NAME = A SYMBOL TO TAKE ON THE VALUE OF THE EXPRESSION

EXPRESSION = ANY NUMBER, DEFINED NAME, OR VALID COMBINATION OF THE ABOVE WITH THE OPERATORS + - * / ^ ( )

FUNCTION : TO ALLOW DEFINITION OF SYMBOLIC ARGUMENTS FOR LATER USE AS ARGUMENTS OF COMPONENT DEFINITIONS

TO ALLOW CALCULATION OF VALUES AS FUNCTIONS OF PRE-DEFINED VALUES

NOTES : THE NAME MAY BE ANY NAME BEGINNING WITH A LETTER AND INCLUDING LETTERS AND NUMBERS BUT NO IMBEDDED BLANKS, WITH A MAXIMUM LENGTH OF 10 CHARACTERS. NO NAME MAY BE THE SAME AS A COMMAND.

THE EXPRESSION MAY BE ANY VALID MATHEMATICAL EXPRESSION USING THE OPERATORS
- + ADDITION
- - SUBTRACTION
- * MULTIPLICATION
- / DIVISION
- ^ EXPONENTIATION (RAISE TO POWER)
- ( ) PARENTHESES MAY BE USED TO GROUP OPERATIONS

EXPRESSIONS ARE EVALUATED FROM LEFT TO RIGHT UNTIL A BLANK IS REACHED. NORMAL OPERATOR PRECEDENCE IS MAINTAINED. THAT IS, UNLESS PARENTHESES ARE USED, EXPONENTIATION PRECEEDS MULTIPLICATION AND DIVISION, AND THESE PRECEED ADDITION AND SUBTRACTION.

IF SEVERAL LET STATEMENTS DEFINE THE SAME NAME, THE LATEST LET WILL ALWAYS BE IN EFFECT. THIS MEANS THAT VALUES MAY BE RE-DEFINED AS THE PROGRAM PROGRESSES, BUT ONCE A COMPONENT IS "DRAWN" ITS DIMENSIONS WILL NOT BE AFFECTED BY CHANGING ANY VALUES.

EXAMPLES : LEGAL NAMES :

LENGTH WIDTH W1 W2 W3 N412 NPORTS

ILLEGAL NAMES:

```
CLINE          (a BAA command)
%4DV37         (contains the illegal character %)
17WAYS         (begins with a number)
THISISTOOLONG  (more than 10 characters)
```

LET COMMAND...CONTINUED

EXAMPLES : LEGAL EXPRESSIONS:

```
1    -1    (-1)    1.05   (-1.09)   WIDTH   -N412   W2^2
(WIDTH-LENGTH)^2/(17-W1)*NPORTS
```

ILLEGAL EXPRESSIONS:

```
((2+3*5)        ((unmatched parentheses)
CLINE-2         (use of an undefinded variable name)
W2**2           (2 operators together--this is not fortran!)
```

TRICKY PROBLEMS WITH EXPRESSIONS:

```
2 + 2           ( =2 because the first blank stops evaluation)
2+2             (ok; = 4)
12/3/2          ( =2)
12/(3/2)        ( =8)
12/3*2          ( =8)
```

EXAMPLE OF USE OF LET STATEMENTS IN A BAA PROGRAM:
(REFER TO THE PAGE ABOUT !DCBLK2)

```
!LET LAMDA = .75
!LET LINEW1 = .25
!LET OFW = LINEW1*.3333
!LET IFW = LINEW1*.16667
!LET FINGERGAP = LINEW1*.05
!LET GAPFTOLINE = LINEW1*.07
!LET DIST = LAMDA/4
.....
!DCBLK2 LINEW1 OFW IFW FINGERGAP GAPFTOLINE DIST (DC1[1.,2.,0.])
```

NOTE THAT ALL THE PARAMETERS OF THE DC BLOCK ARE SPECIFIED AS FUNCTIONS OF LAMDA AND OF THE MAIN LINE WIDTH LINEW1. THIS MEANS THAT IF THE DESIGNER WISHES TO CHANGE THE LINEWIDTH AND MAINTAIN A CONSTANT "SHAPE" IN HIS COMPONENT, ALL HE HAS TO DO IS CHANGE THE VALUE OF LINEW1 BY CHANGING THE SINGLE !LET STATEMENT IN WHICH IT IS DEFINED.

MIRROR

FORM: !MIRROR X
!MIRROR Y

FUNCTION: TO PRODUCE A MIRROR IMAGE OF COMPONENTS ABOUT THE DESIRED AXIS

!MIRROR X will mirror components about the X-axis
!MIRROR Y will mirror components about the Y-axis

Note. This implementation of mirror is as a toggle. Therefore you need only specify which mirror is to be affected and that mirror state will be toggled.

```
!NAME TESTD4
!CHAMFER .1,.03 (C1[-1.5,.5,0]) (C2[-.5,.5,180]) (C3[-1.5,1.,0])
!CONNECT .05,.1 C1 C2
!CONNECT .05,.3 [-1.,.5,90] C3
!DUMP
!MIRROR Y
!CHAMFER .1,.03 (C4[-1.5,.5,0]) (C5[-.5,.5,180]) (C6[-1.5,1.,0])
!CONNECT .05,.1 C4 C5
!CONNECT .05,.3 [-1.,.5,90] C6
!DUMP
!MIRROR Y
!MIRROR X
!CHAMFER .1,.03 (B1[-1.5,.5,0]) (B2[-.5,.5,180]) (B3[-1.5,1.,0])
!CONNECT .05,.1 B1 B2
!CONNECT .05,.3 [-1.,.5,90] B3
!DUMP
!END
!STOP
END OF FILE
??
```

Element 1

toggle Y MIRROR

Element 2

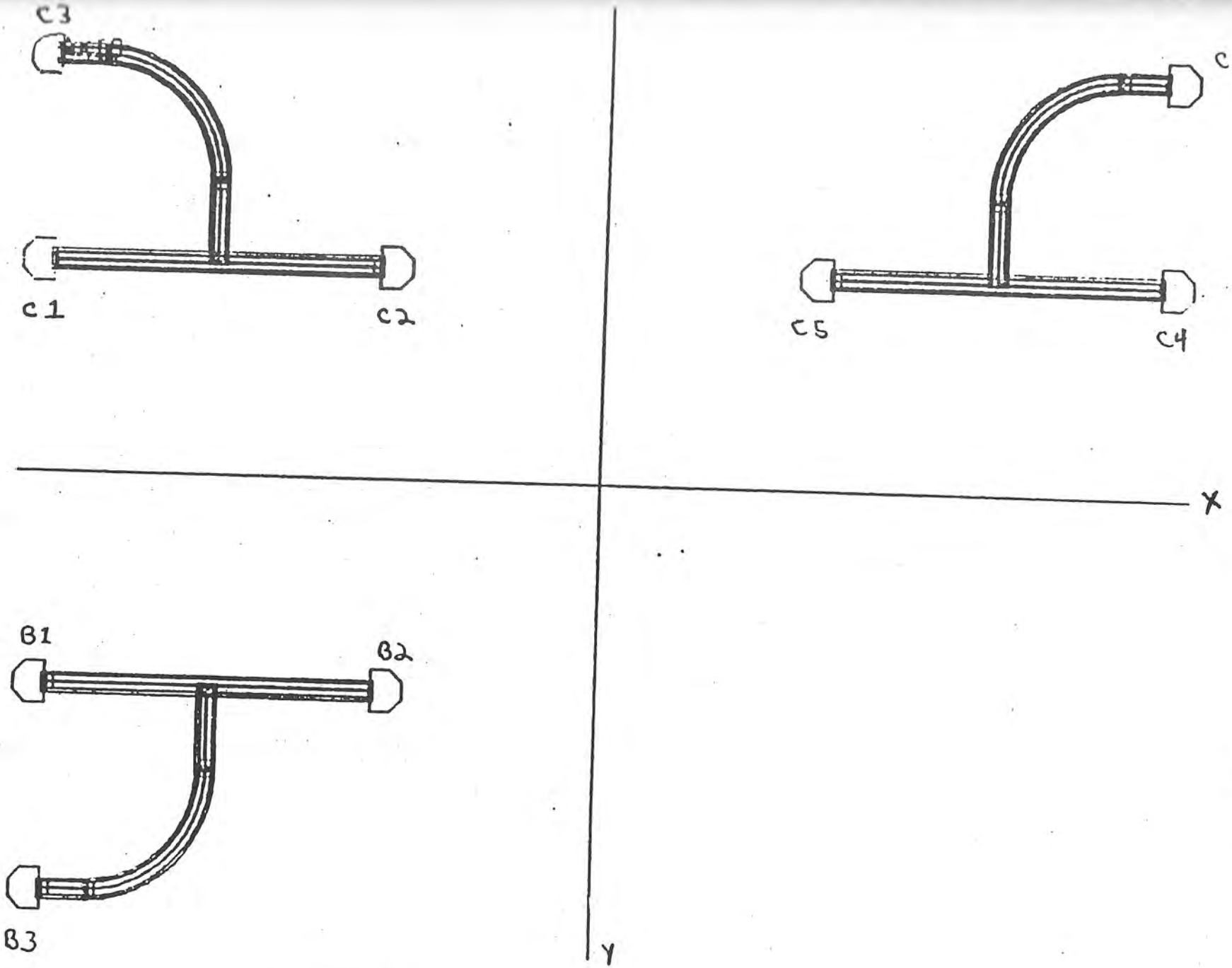
toggle Y MIRROR

toggle X MIRROR

Element 3

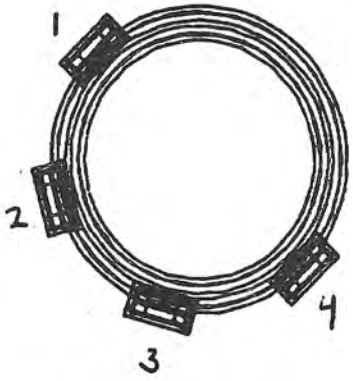
PAGE 94

- FILE TESTD4/UN=R767829

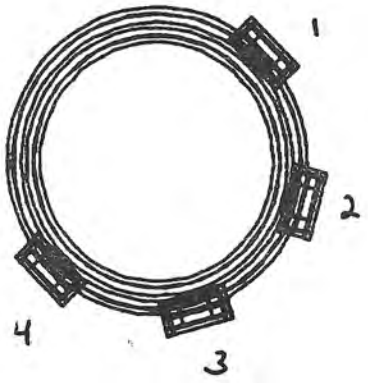


- MIRROR COMMAND

?



R2



R1

EXAMPLE OF MIRROR Y

```
!NAME TESTD7
!RATEQ .1,.05,.4 (R1[1.,2.,45])
!DUMP
!MIRROR Y
!RATEQ .1,.05,.4 (R2[1.,2.,45])
!DUMP
!END
!STOP
 EOI ENCOUNTERED.
/
```

EXAMPLE OF MIRROR Y

```
1       !NAME TESTD7
2       !RATEQ .1,.05,.4 (R1[1.,2.,45])
3       !DUMP
    R1
      0     1.0000     2.0000    45.0000
      1     1.1768     2.1768    45.0000
      2     1.2415     1.9353   345.0000
      3     1.0647     1.7585   285.0000
      4      .8232     1.8232   225.0000
4       !MIRROR Y
5       !RATEQ .1,.05,.4 (R2[1.,2.,45])
6       !DUMP
    R1
      0     1.0000     2.0000    45.0000
      1     1.1768     2.1768    45.0000
      2     1.2415     1.9353   345.0000
      3     1.0647     1.7585   285.0000
      4      .8232     1.8232   225.0000
    R2
      0    -1.0000     2.0000    45.0000
      1    -1.1768     2.1768   135.0000
      2    -1.2415     1.9353   195.0000
      3    -1.0647     1.7585   255.0000
      4     -.8232     1.8232   315.0000
7       !END
8       !STOP
/EOI ENCOUNTERED.
```

EXAMPLE OF MIRROR y LIST FILE

pg 98

NAME

FORM: !NAME NNN...

WHERE: NNN...CIRCUIT NAME SELECTED BY USER.

FUNCTION: INITIALIZES FILES TO BEGIN A NEW CIRCUIT DESCRIPTION.

NOTE: THE CIRCUIT NAME CAN BE ANYTHING, BUT IT MUST BEGIN WITH A LETTER.

OFFSET

CHANGE IN ARTWORK ORIGIN

FORM: !OFFSET X,Y

WHERE X,Y DEFINE THE NEW ARTWORK ORIGIN WITH RESPECT TO THE INITIAL ORIGIN. OFFSET COMMANDS ARE NOT CUMULATIVE.

PRINT

FORM: !PRINT 1 (TURNS FUNCTION ON)
!PRINT Ø (TURNS FUNCTION OFF)

FUNCTION: PRINTS ALL BAA STATEMENTS AND COMMANDS INCLUDED BETWEEN !PRINT 1 AND !PRINT Ø

NOTE: THE INITIAL STATE OF PRINT IS ON.

ROTATE

FORM: !ROTATE A

WHERE A = ANGLE OF ROTATION SPECIFIED IN DEGREES

NOTE: ROTATE COMMANDS ARE CUMULATIVE
A !ROTATE 90 FOLLOWED BY A !ROTATE 45
LATER IN THE CIRCUIT DESCRIPTION WILL YIELD
A ROTATION OF 135°. POSITIVE ROTATIONS ARE
IN A CCW DIRECTION AND NEGATIVE ROTATIONS ARE
IN A CW DIRECTION.

: !ROTATE 0 will reset the GLOBAL ROTATIONAL OFFSET
to 0°

SCALE

FORM: !SCLXXX...

WHERE: XXX... = SCALE FACTOR (MAY BE > OR < 1)

FUNCTION: MULTIPLIES ALL DIMENSIONS BY THE SCALE FACTOR. SOMETIMES, WHEN USING LARGE SCALE FACTORS, CERTAIN PRESET BAA PROGRAM CONSTANTS MAY NOT BE APPROPRIATE. BE AWARE OF THE BAPER AND EXTEND COMMANDS.

STOP

FORM: !STOP

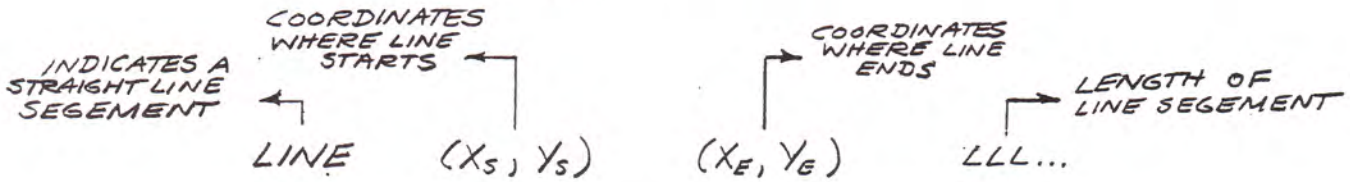
FUNCTION: TERMINATES AN ARTWORK GENERATION RUN.

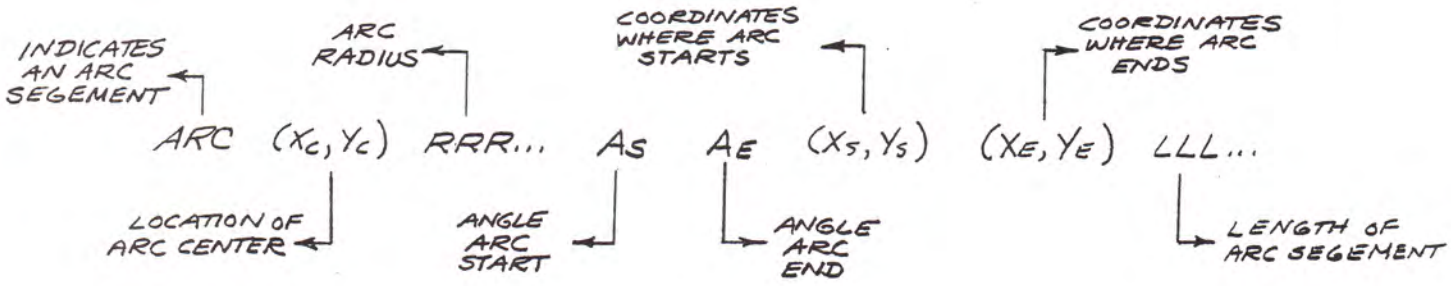
TRACE

FORM: !TRACE 1 (TURNS FUNCTION ON)
!TRACE Ø (TURNS FUNCTION OFF)

FUNCTION: PRINTS INFORMATION PERTAINING TO THE INDIVIDUAL LINES AND ARCS GENERATED BY THE CLINE AND CONNECT STATEMENTS INCLUDED BETWEEN !TRACE 1 AND !TRACE Ø

NOTES: THE TRACE MODE IS ACTIVATED AFTER EACH CLINE OR CONNECT STATEMENT. THE FOLLOWING INFORMATION IS SUPPLIED IN THE FORMAT SHOWN BELOW:





NOTE: INITIAL STATE OF TRACE IS OFF

! TRADEMARK !

FORM: !TRADEMARK W [X,Y,A]

WHERE W = OVERALL WIDTH OF TRADEMARK

X,Y,A = X,Y COORDINATES AND ANGLE OF ROTATION OF TRADEMARK

FUNCTION: DRAWS RAYTHEON TRADEMARK INCLUDING BAA LOGO

EXAMPLE: THE ILLUSTRATION ABOVE WAS DRAWN BY THE FOLLOWING PROGRAM

```
!NAME TRADMK
!TRADEMARK 1. [0,0,0]
!END
!STOP
```

NOTE: FOR SPEED PURPOSES, SEEIT SIMULATES ALPHANUMERIC CHARACTER GENERATION BY DRAWING A RECTANGLE IN THE CORRECT SIZE, ORIENTATION, AND JUSTIFICATION OF THE CHARACTERS. THE FINAL NEGATIVE WILL HAVE THE CHARACTERS IN PLACE.

APPENDIX

RUNNING BAA

1) BAA CIRCUIT DESCRIPTION FILE "FNAME" IS CREATED BY USING XEDIT. THE FILE MUST CONFORM TO THE FOLLOWING RULES:

   a) NO LINE NUMBERS
   b) ALL STATEMENTS MUST BEGIN IN COLUMN 1
   c) ALL CONTINUATION STATEMENTS MUST HAVE A BLANK IN COLUMN 1.

2) THE BAA SYSTEM IS ACCESSED BY:

   GET, BAA / UN=R764519

3) THE CIRCUIT DESCRIPTION IS THEN COMPILED AND VIEWED BY:

   BEGIN, SEEIT, BAA, "FNAME"

   WHERE "FNAME" IS THE NAME OF THE DISK FILE FROM 1)

4) IF CIRCUIT COMPILATION COMPLETES WITHOUT SERIOUS ERRORS THE USER WILL THEN ENTER THE GRAPHIC INTERFACE

5) THE GRAPHICS INTERFACE WILL LIST ALL VALID COMMANDS AND ALLOW THE USER TO EXAMINE THE FILE

6) WHEN THE CIRCUIT IS SATISFACTORY, A GERBER TAPE CAN BE MADE BY:

BEGIN, TAPEIT, BAA

NOTE: IT IS NOT NECESSARY TO RUN "SEEIT" BEFORE "TAPEIT"

THE TAPEIT FILE WILL ASK THE FOLLOWING QUESTIONS:

a) USER #, PASSWORD?

b) DSO?

c) LONG OR SHORT TAPE REPORT? (A LONG REPORT IS THE SAME AS A SHORT EXCEPT THAT A RADAC COMPILATION IS INCLUDED.)

d) DO YOU WANT OUTPUT ROUTED TO CENTRAL SITE? (CENTRAL SITE = I/O ROOM IN BEDFORD)

IF YOU ANSWER "NO" AN ALTERNATE SITE WILL BE REQUESTED

e) NAME OF DISK FILE WITH BAA CODE?

AFTER ANSWERING THE QUESTIONS, A JOB WILL BE SUBMITTED AND A TAPE MADE.

```
 BEGIN,TAPEIT,BAA
TAPE-IT PROCEDURE FILE

ENTER USER NO.,PASSWORD.
? R764500,SHEEP
ENTER DSO.
? 710555121a
DO YOU WANT A LONG OR SHORT TAPE REPORT?
? LONG
DO YOU WANT OUTPUT ROUTED TO CENTRAL SITE? Y/N
? Y
ENTER NAME OF BAA FILE TO RUN.
? TESTD4


UN=R764500  LOG OFF  14.00.22.
JSN=AKFL      SRU=S        4.837
IAF      CONNECT TIME 00.05.42.
LOGGED OUT.

HOST DISCONNECTED   CONTROL CHARACTER=(ESC)
ENTER INPUT TO CONNECT TO HOST
```

A TYPICAL TAPEIT SESSION

THE SEEIT USERS MANUAL

COMMAND SUMMARY FOR THE BAA GRAPHICS INTERFACE.

SEEIT USER'S GUIDE

TABLE OF CONTENTS

SEEIT USER'S GUIDE

Introduction

SEEIT is an interactive viewing program designed to read BAA output instructions and simulate the resulting photoplot on any Tektronix (PLOT 10) compatible plotting device (see Figure 1, System Usage Diagram). This system is specifically designed for viewing microwave circuits and gives the user an alternative to the previous RADAC-PREVU method of viewing a BAA drawing.

SEEIT is a circuit design viewing program only. It cannot alter the BAA output instructions that it is reading.

System Features

(1.) The time period between BAA execution and circuit viewing is much shorter for SEEIT than for PREVU, because PREVU requires RADAC preprocessing while SEEIT can read BAA output data directly.

(2.) SEEIT can process the largest BAA output files without running into overflow problems,

(3.) Conductor paths are drawn showing their width.

SEEIT draws a series of narrow lines that represent the boundaries of the conductor path.

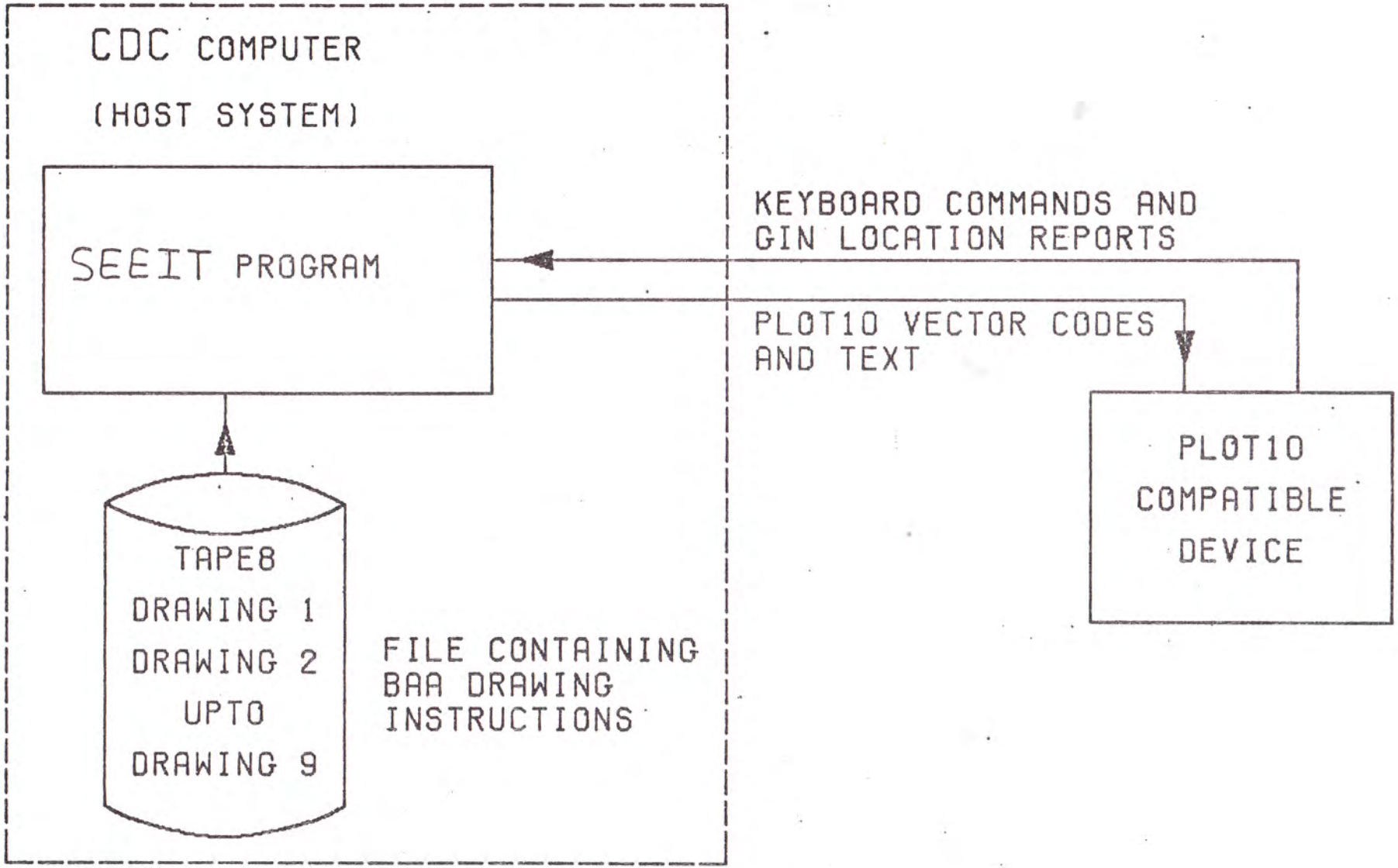


FIGURE 1 - SYSTEM USAGE DIAGRAM

(4.) Zooming (and panning) is done by specifying two diagonally opposite corner points of the zoom window.

(5.) The resolution used in drawing circles and arcs is adjusted for the relative size of the circle or arc to the size of the drawing in view. This means that a particular circular object might look different when it is "zoomed in" because the program is using more lines to draw smoother curves.

(6.) SEEIT allows the user to specify three different line types; solid, dotted, and dashed in order to distinguish an original drawing from an overlayed (superimposed) drawing. Unfortunately, both dotted and dashed lines draw more slowly than do solid lines.

(7.) SEEIT allows the user to access drawings by entering the number of the drawing or by entering the name assigned to the drawing. For example, the user can enter the command DRAW, 1 to draw the first drawing in the BAA output data file, or the user can enter the command DRAW,BACKCKT2 to draw a drawing named BACKCKT2 in the BAA output data file.

(8.) For speed purposes, SEEIT simulates alphanumeric character generation by drawing a rectangle in the correct size, orientation, and justification of the characters.

(9.) SEEIT draws the outline of a boundary area only.

(10.)The WHERE and ZOOM commands are functional only on Tektronix (PLOT 10) compatible terminals that support graphic input mode.

System Hardware Configuration

SEEIT can be used on the least sophisticated PLOT 10 compatible terminals. Plot 10 is a terminal control FORTRAN software package for Tektronix 4010 series graphic displays. There are several non Tektronix graphic terminals available that support "Tektronix PLOT 10 emulation". In order to use the WHERE and the ZOOM commands, the terminal must have graphic input capability.

In order to make hardcopies from the terminal, the graphic workstation must include either a Tektronix 4631 or 4632 hardcopy unit.

User Interface

The SEEIT program is controlled by user keyboard input. When the program is ready to process a command from the user, the command prompt prints in the upper left corner of the screen. The command prompt is the word COMMAND followed by a question mark (?) printed on the line below. The user types the command followed by an argument, if an argument is necessary, and then a carriage return to enter the command. The argument is separated from the command by one space or one comma. If there is an error in the command or if the command returns a report, then the error message or report prints below the appropriate command.

The next command prompt prints below the previous command prompt, except when the command function erases the screen. In that case, the command prompt prints in the upper left corner of the screen. When the HELP command is entered, the command prompt prints below the end of the help message. See Figure 2. Each time the command prompt appears, the terminal's "bell" is activated.

In SEEIT terminology, the unit of data referred to as a RADAC file is called a drawing. There can be several RADAC files in one CDC file such as TAPE8. These RADAC files are separated from each other by file terminating instructions. There can also be a file naming instruction at the beginning of the file. BAA writes an up to ten character drawing name with the name instruction. SEEIT reads the drawing name of each file/drawing, then uses these names to identify a drawing if the user specifies a drawing name.

Error Messages

A syntax error message is generated when the command is misspelled. If the program cannot find a match between the first three letters of the user command and the list of possible commands, then "SYNTAX?" is returned as an error message. The syntax error can also be generated if the command arguments contain inappropriate characters, e.g., a real number argument containing a non-numeric character other than an initial negative sign or decimal point.

Other error messages are described under the appropriate commands.

SEEIT Commands

The following is an alphabetized listing with descriptions and error messages of the currently available SEEIT commands.

AXIS - Draw X-Y Axis Command

This command draws the line X=0.0 and the Y=0.0. This gives the user the position of the X-Y axes relative to the drawing. Note: Axis lines are drawn using the current line type.

Command Sequence:

AXIS

No Arguments

Error Messages:

AXIS NOT ON SCREEN - The X=0.0 and the Y=0.0 lines are not within the current viewing window.

COORDINATE SYSTEM UNDEFINED - The viewing window has not been established. The user must enter a DRAW command before the axes can be drawn.

COPY - Hard Copy Command

This command activates the Tektronix hard copy system so that a hardcopy of the current screen contents are saved. Note 1: Hardcopy unit must be available in order to produce hard copies of screen pictures. See Section on System Hardware. Note 2: User can also produce hardcopies by pressing the hard-copy button available on most Tektronix terminals.

Command Sequence:

COPY

No Arguments

No Error Messages

DASH - Set Line Type to Dashed Command

This command changes the current line type to the dashed line type. Note 1: Line type commands are modal, i.e., the program uses the the last specified line type for all lines drawn until a new line type command is entered. Note 2: Dashed and dotted line types are drawn more slowly than solid lines.

Command Sequence:

DASH

No Arguments

No Error Messages

DOT - Set Line Type To Dotted Command

This command changes the current line type to the dotted line type. Note 1: Line type commands are modal, i.e., the program uses the last specified line type for all lines drawn until a new line command is entered. Note 2: Dotted and dashed line types are drawn more slowly than solid lines. Note 3: The dotted line type actually appears as a short dash on the Tektronix screen.

Command Sequence:

DOT

No Arguments

No Error Messages

DRAW - Draw a Specified Drawing Command

This command erases the screen and draws the specified drawing using the current line type.

Command Sequence:

DRAW,drawing NAME or NUMBER

Arguments:

NAME — the name associated with a particular drawing.

NUMBER — the number indicating the physical position of a drawing in a BAA output file.

Error Messages:

DRAWING NUMBER NOT FOUND or DRAWING NAME NOT FOUND - The user specified a drawing number or drawing name that was not on the list of available drawings.

END - Terminate SEEIT Command

This command terminates program control (same as QUIT).

Command Sequence:

END

No Arguments

No Error Messages

FIND - Find a Point On the Screen Command

This command draws a small cross on the screen centered at the point X,Y. Note 1: X and Y are expressed in a real number value of inches. Note 2: The small cross is drawn using the current line type.

Command Sequence:

FIND,X,Y

Arguments:

X,Y — the coordinates of the center of the cross.

Error Messages:

COORDINATE SYSTEM UNDEFINED - The viewing window has not been established. The user must enter a DRAW command before the FIND command can be entered.

NOT ON SCREEN - The point the user has entered is not on the screen.

NOT REAL # - At least one of the arguments is not a real number.

GRID - Grid Drawing Command

This command draws a location grid on the current picture with the separation between grid lines equal to the grid dimension in inches. Note 1: The grid dimension is expressed in a real number value of inches. Note 2: Grid lines are drawn using the current line type.

Command Sequence:

GRID,grid DIMension

Arguments:

DIM — a real number specifying the distance in inches between grid lines.

Error Messages:

COORDINATE SYSTEM UNDEFINED - The viewing window has not been established. The user must enter a DRAW command before the GRID command can be entered.

NOT REAL # - Grid dimension was not specified as a real number.

TOO SMALL or TOO BIG - The grid dimension is inappropriate for the current size of the viewing window. A grid dimension that is too small would require a very long time to complete. A grid dimension that is too large would not appear on the screen.

HELP - Display User's Guide Command

This command displays a short form of the user's guide.

Command Sequence:

HELP

No Arguments

No Error Messages

LINE - Set Line Type to Solid Command

This command changes the current line type to solid.

Command Sequence:

LINE

No Arguments

No Error Messages

OVERLAY - Overlay Drawing Command

This command overlays the specified drawing on the current drawing using the current line type and window limits. Note: The specified drawing becomes the current drawing. The current drawing is the only drawing that is subject to the ZOOM command. See ZOOM command.

Command Sequence:

OVERLAY,drawing NAME or drawing NUMBER

Arguments:

NAME — the character name associated with a particular drawing.

NUMBER — the number indicating the physical position of a drawing in a BAA output file.

Error Messages:

DRAWING NUMBER NOT FOUND or DRAWING NAME NOT FOUND - The user has specified a drawing number or a drawing name that is not on the list of available drawings.

QUIT - Terminate SEEIT Command

Terminate program control (same as END).

Command Sequence:

QUIT

No Arguments

No Error Messages

WHERE - Get the Position of a Point Command

This command gets and reports the X-Y coordinates of a user specified point on the screen. The graphic cursor appears on the screen. The user can manipulate the terminal's thumb wheels to position the graphic cursor at any point on the screen. When the user types any printing key, the location of the graphic cursor is printed just below the command.

Note 1: The user must wait until the graphic cursor bell rings before typing the key to select a point. Note 2: The WHERE command can be used only with terminals that support graphic input.

Calling Sequence:

WHERE

Values Returned:

X,Y — X-Y coordinates of the graphic cursor.

Error Message:

NO DRAWING TO LOCATE - The viewing window has not been established. The user must enter a DRAW command before the WHERE command.

ZOOM - Zoom and Pan Command

This command allows the user to, in effect, pan to any visible area of the current drawing then "zoom in" to see more detail. The graphic cursor appears on the screen. The user can manipulate the terminal's thumb wheels to position the graphic cursor at any point on the screen. The user must enter two points to specify a new virtual window. This window is mapped onto the full screen after the second point is entered. The window points are any two diagonally opposite corner points of the zoom window. To enter the points, the user moves the graphic cursor to the desired location then types any printing key.

Note 1: "Magnified" drawings may appear different from the original drawing due to the fact that the program increases the resolution, i.e., the number of vectors drawn for circles and arcs when the size of the virtual window decreases. Note 2: The ZOOM command can be used only with terminals that support graphic input.

Calling Sequence:

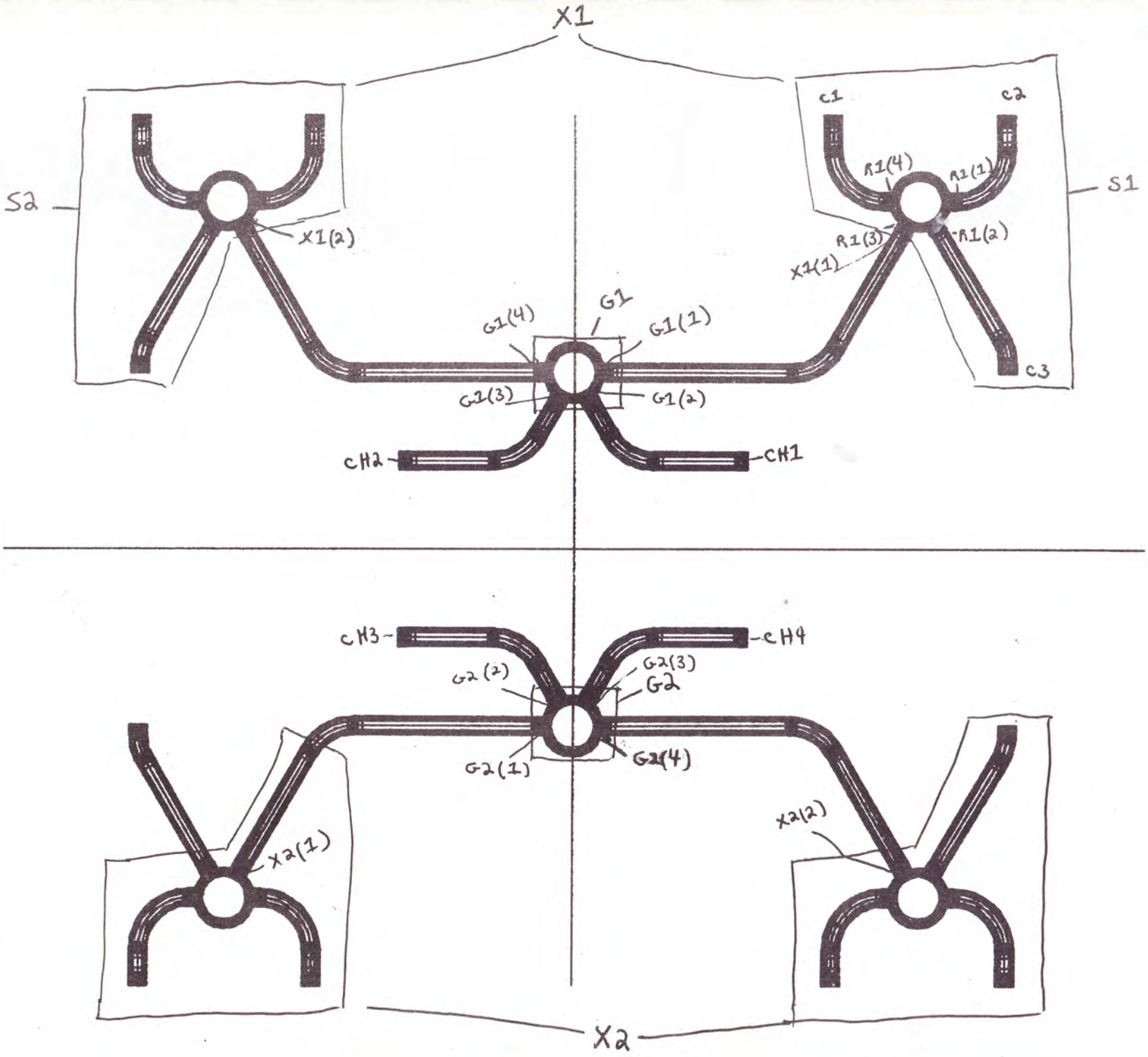
ZOOM

No Arguments

Error Message:

NO DRAWING TO ZOOM - The viewing window has not been established. The user must enter a DRAW command before ZOOM.

TWO EXAMPLES USING VARIOUS FETURES UNIQUE TO BAA 2.0


X1
S2
S1
C1
C2
C3
X1(2)
R1(4)
R1(1)
R1(3)
R1(2)
X1(1)
G1
G1(4)
G1(1)
G1(3)
G1(2)
CH2
CH1
CH3
CH4
G2(2)
G2(3)
G2
G2(1)
G2(4)
X2(1)
X2(2)
X2

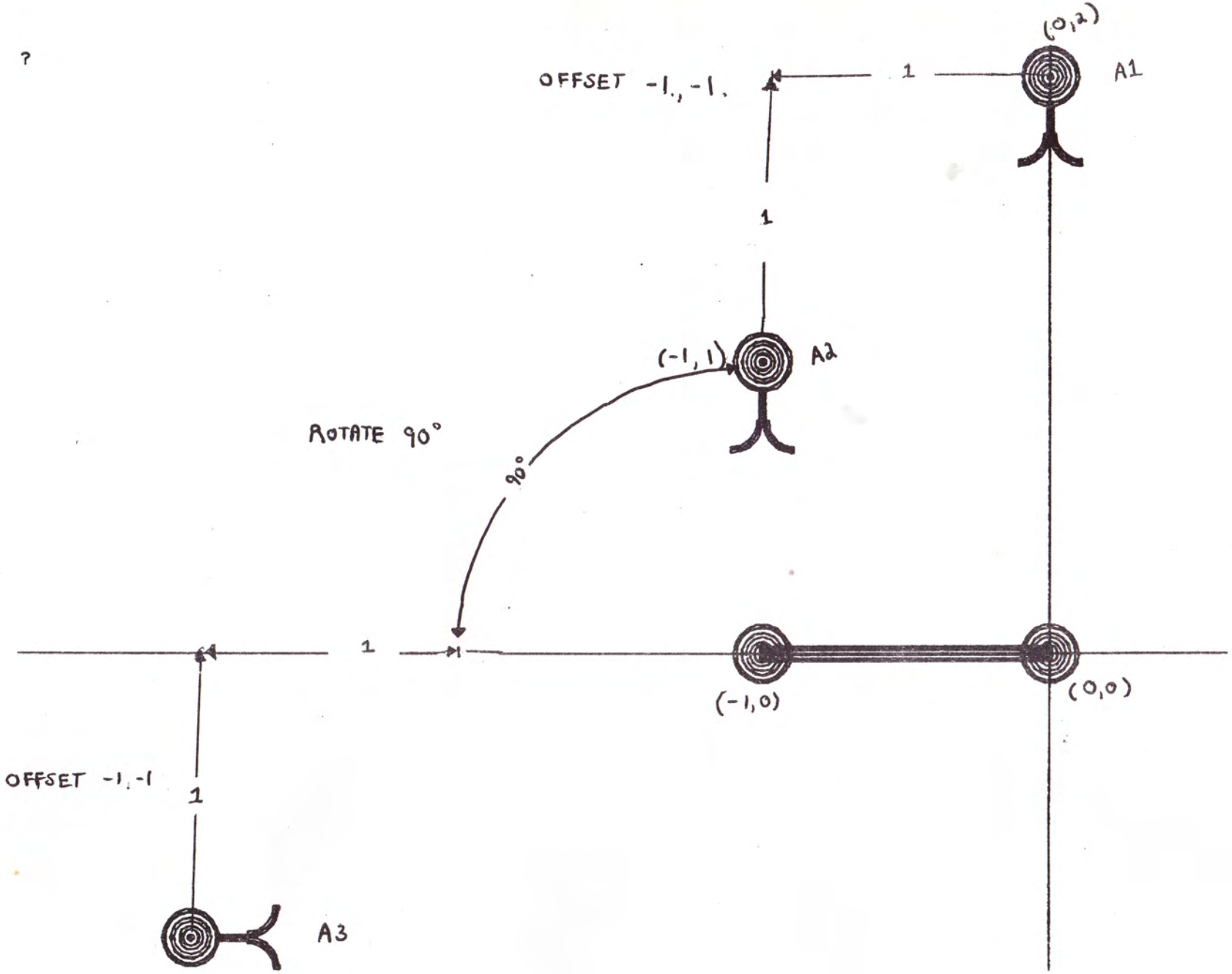
```
     1      !NAME TEST1
     2      * ACID TEST OF B4A 2.0
     3      !DEFINE SUB1
     4      !RATEQ .1,.05,.3 (R1[0,0,0])
     5      !CHAMFER .1,0 (C1[-.5,.5,-90])
     6                   (C2[.5,.5,-90])
     7                   (C3[.5,-1.,90])
     8      !CONNECT .1,.3 C1 R1(4)
     9                    C2 R1(1)
    10                    C3 R1(2)
    11      !ENDDEF SUB1 1 R1(3)
   -.1000    -.1732 240.0000
    12      !DEFINE SUB2
    13      !GET SUB1 (S1[2,0,0])
    14      !MIRROR Y
    15      !GET SUB1 (S2[2,0,0])
    16      !ENDDEF SUB2 2 S1 S2
   1.9000    -.1732 240.0000
  -1.9000    -.1732 300.0000
    17      !GET SUB2 (X1[0,2,0])
    18             (X2[0,-2,180])
    19      !RATEQ .1,.05,.3 (G1[0,1,0])
    20                          (G2[0,-1,180])
    21      !CHAMFER .1,0 (CH1[1,.5,180])
    22                   (CH2[-1,.5,0])
    23                   (CH3[-1,-.5,0])
    24                   (CH4[1,-.5,180])
    25      !CONNECT .1,.3 X1(2) G1(4)
    26                     X1(1) G1(1)
    27                    CH2 G1(3)
    28                    CH1 G1(2)
    29                    X2(2) G2(4)
    30                    X2(1) G2(1)
    31                    CH3 G2(2)
    32                    CH4 G2(3)
    33      !END
    34      !STOP
EOI ENCOUNTERED.
/
```

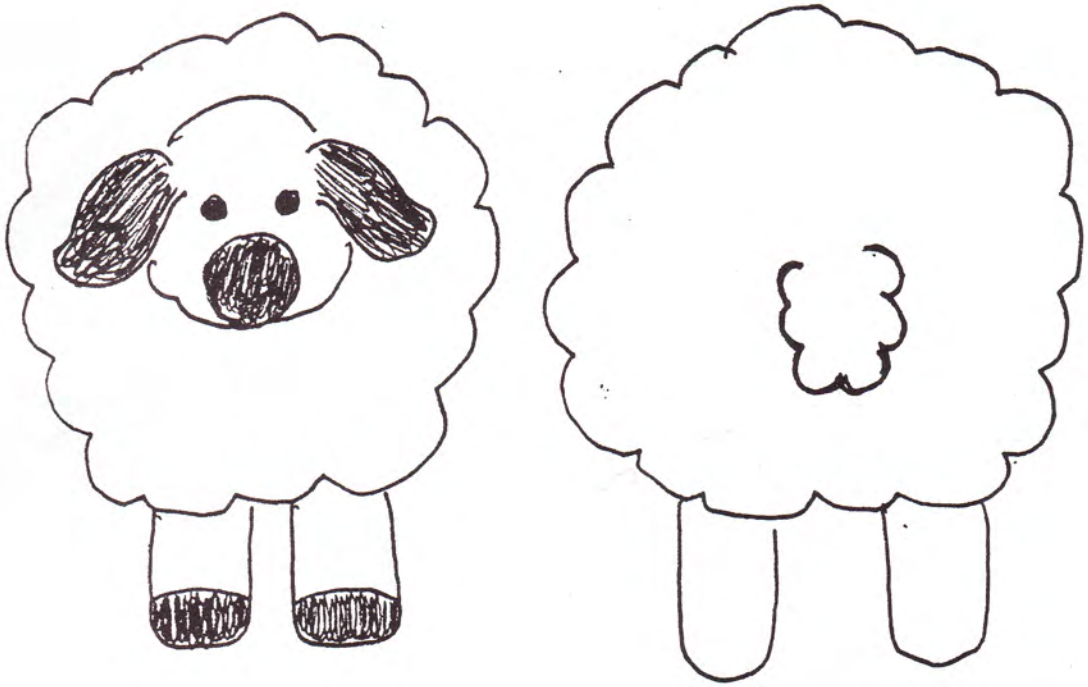
LIST FILE

```
!NAME TESTD3
!CIRCLE .1,.1 [-1.,0.]
!CIRCLE .1,.1 [0.,0.]
!CONNECT .05,.1 [-1.,0.] [0.,0.]
!DEFINE SHAPE
!CIRCLE .1,.1 [0.,2.]
!PWRDV1 .02,.02,.1,.1 (B1[0.,1.9,180])
!ENDDEF SHAPE 1 [0.,0.,0]
!GET SHAPE (A1[0.,0.,0])
!OFFSET -1.,-1.
!GET SHAPE (A2[0.,0.,0])
!ROTATE 90
!GET SHAPE (A3[0.,0.,0])
!END
!STOP
EOI ENCOUNTERED.
/
```

USER DEFINED ELEMENT

?


OFFSET -1, -1
1
(0,2)
A1
1
(-1,1)
A2
ROTATE 90°
90°
1
(-1,0)
(0,0)
OFFSET -1, -1
1
A3



THE END