PROGRAMMERS' REFERENCE MANUAL

S-C 4020

Computer Recorder

Document No. 9500056
October 1964

Revised

June 1968

Stromberg Datagraphics, Inc.

CHARACTRON® is a trademark of the Stromberg-Carlson Corporation.

PROGRAMMERS' REFERENCE MANUAL

S-C 4020 Computer Recorder

Table of Contents

Section I

INTRODUCTION

The S-C 4020 Computer Recorder, designed and manufactured by Stromberg-Carlson Corporation, Data Products, San Diego, Calif., operates as peripheral equipment for high-speed digital computers. This document describes the features of the machine, gives the basic programming requirements including the coding and word format of commands used for control and recording operations, and describes a set of subroutines to assist the programmer in developing grid backgrounds, scaling data, and formatting printed material. These routines are made available by Stromberg-Carlson and have been derived from a programming system contributed largely by the programming groups of North American Aviation Company, Inc.

The obvious application of the S-C 4020 is the rapid production of labeled graphs. Results in graphical form are usually much easier to analyze than are the same results in printed tabular form.

Less obvious, but often highly desirable, is the exploitation of the S-C 4020 as a high-speed printer. For certain applications, this affords distinct advantages over the use of a standard printer. For one thing, printed records can be obtained on 35mm or 16mm film, which greatly reduces the amount of space required to store the records. A form suited to the printed information can be generated or superimposed by the S-C 4020.

A variation is the production of a frame that combines tabular data with a plot of the related curve. Still another use of S-C 4020 techniques is the creation of diagrams and line drawings.

At first glance, the list of S-C 4020 routines on the following pages seems formidable indeed, but the user should not feel baffled by the great volume of details available. Some routines were developed for special purposes with limited usage. Many are used principally as lower-level modules for more general subroutines, and will seldom be called directly by the programmer.

This Programmers' Reference Manual has been organized in major sections according to the function of the routines described therein. The major sections are: Introduction and Description of the S-C 4020, Machine Control Routines, Grid

Drawing Routines, Scaling Routines, Point-Plotting and Line-Drawing Routines, Titling and Labeling Routines, and Higher Level Printing and Graphics Routines. Information about the interpretation of the system subroutines and an operating system has been included in the Controls section.

In order to become familiar with the equipment, the programmer should read the following machine description before proceeding to the programming details.

DESCRIPTION OF MACHINE FEATURES

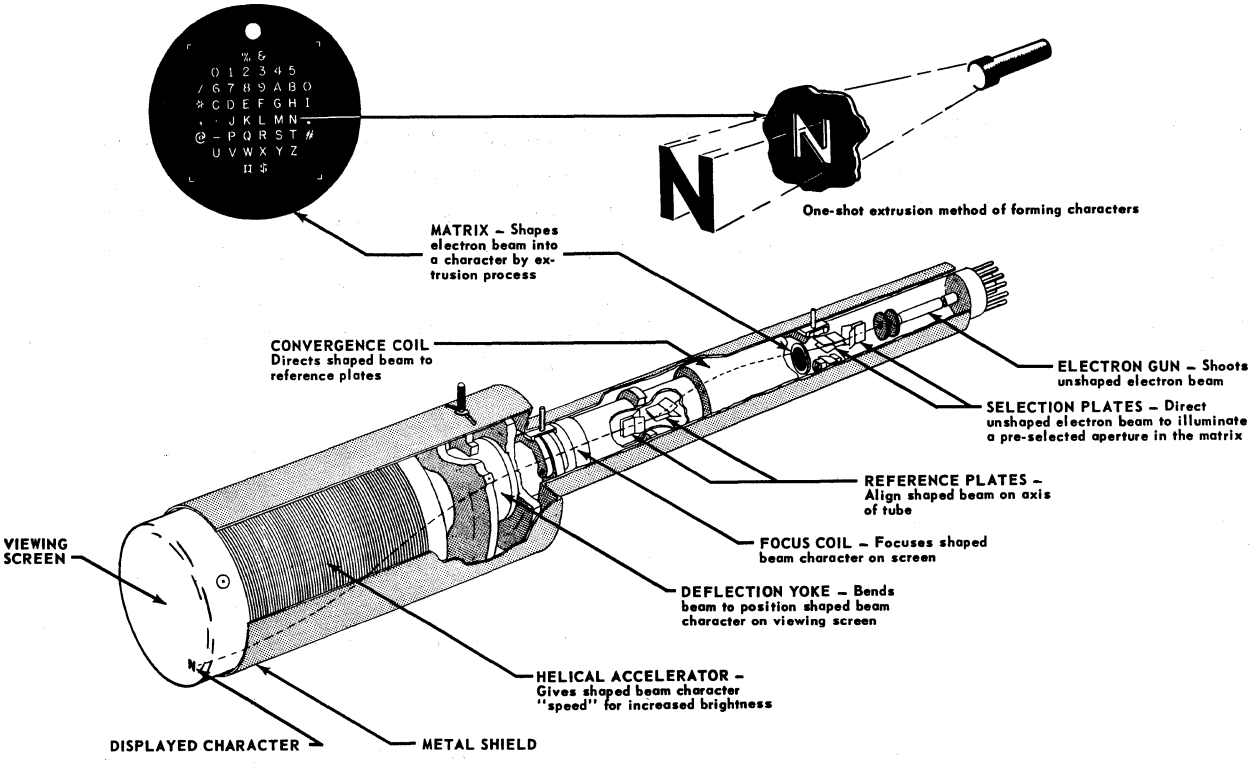
The CHARACTRON® Shaped Beam Tube, the basic device of the S-C 4020, permits generation of certain characters at a very high speed. This tube, illustrated schematically in Figure 1-1, contains a built-in character-forming section which consists of an electron gun, character-selection plates, and a stencil-like matrix.

The Matrix, a small thin disk of special alloy metal, employs 64 character-shaped openings engraved in a space less than one-fourth of an inch square. These 64 characters are arranged in an 8 x 8 array, thus giving the matrix its name.

The Electron Gun directs an electron beam toward the matrix through two sets of electrostatic character-selection plates. These plates deflect the beam so that it passes through the particular matrix aperture desired. Thus, as the beam leaves the matrix, it has been shaped in the form of the character aperture through which it has passed. As this shaped beam proceeds toward the tube face, it is accelerated (brightened) and again deflected to the proper position on the tube face. From this description, it can be seen that it takes no longer to print an entire CHARACTRON (beam-shaped) character than it does a single dot.

Each CHARACTRON Character occupies approximately 6-horizontal by 9-vertical positions on the plottable area of the tube face. This area, called the raster, contains over one million plottable positions. These CHARACTRON characters cannot be rotated nor changed in size in any way. However, if other characters which may differ in size, shape, or attitude from those provided by the matrix are desired, they may be constructed from dots or lines.

A Typewriter Simulator, so-called because it provides automatic spacing and "carriage return" capabilities, is built into the S-C 4020. Each CHARACTRON character "typed" occupies 8-horizontal and 16-vertical positions on the raster; this character space area provides the spacing necessary to separate adjacent characters, both vertically and horizontally. In this mode of operation, a maximum of 128-horizontal and 64-vertical alphanumeric characters are possible in one


% &
0 1 2 3 4 5
/ 6 7 8 9 A B O
* C D E F G H I
, · J K L M N .
@ – P Q R S T #
U V W X Y Z
¤ $
N
One-shot extrusion method of forming characters
MATRIX – Shapes electron beam into a character by extrusion process
CONVERGENCE COIL Directs shaped beam to reference plates
ELECTRON GUN – Shoots unshaped electron beam
SELECTION PLATES – Direct unshaped electron beam to illuminate a pre-selected aperture in the matrix
REFERENCE PLATES – Align shaped beam on axis of tube
VIEWING SCREEN
FOCUS COIL – Focuses shaped beam character on screen
DEFLECTION YOKE – Bends beam to position shaped beam character on viewing screen
HELICAL ACCELERATOR – Gives shaped beam character "speed" for increased brightness
DISPLAYED CHARACTER
METAL SHIELD

Charactron Shaped Beam Tube, 7-inch Type

Figure 1-1.

display. The point at which the first character is to be printed must be specified (the center of the character position), and thereafter the horizontal printing position is automatically advanced one character-space to the right. At the edge of the frame, the horizontal position is returned to the left side of the frame and the vertical position is advanced down one character-space. This continues until the machine is told to stop printing, i. e. , leave the typewriter mode of operation.

The Typewriter-like Operation takes place at an extremely rapid rate. Unlike the human typist however, the machine does not check to make sure that it ends a line at the end of a word; a carriage return must be programmed or the appropriate number of blanks inserted to insure that the lines will come out correctly. Also, if the end of the film frame is reached, "typing" will continue at the upper left-hand corner of the current film frame (on top of anything that was put there before) unless an advance-film command is given.

Heavy and Light Intensities of exposure are available for recording CHARACTRON-generated characters. A greater range of intensities may be obtained by repeated exposure at either light or heavy setting. This control does not affect the intensity of lines drawn by the axis generator or the vector generator. Overexposure due to excessive repetition results in haloing on the film and poor definition in the output. Once the intensity has been set, all plotting and typewriter-printing will be at that intensity until it is changed. For special applications, a variable intensity plot command option is available. This permits plotting at any of sixteen intensity levels without affecting the intensity mode set by a light or heavy command.

A Vector Generator permits drawing a line from a specified point to the resultant of the specified X and Y components. The length of one vector is limited; the X and/or Y components may be no greater than 63 raster counts*. A longer line may be formed by a series of vectors plotted head-to-tail. Since the extent of a vector is determined by the X and Y components, the vectors will not necessarily stop at the edge of the frame. This feature might be used to advantage in marking the location of errors.

An Axis Generator enables grid lines to be drawn horizontally or vertically across the film frame. The coordinates of the starting point of each axis must be specified. The stop point is specified as the Y coordinate or the complement of the X coordinate, whichever is appropriate.

Two Cameras are available and can be used separately or simultaneously to photograph the display on the tube face. The microfilm camera may be either 35mm (with sprocket holes) or 16mm (without sprocket holes). The image size is a 17.5-mm square on 35mm film and a 14-mm square on 16mm film. The 9.5-inch camera records the same picture as a 7.5-inch square on sensitized vellum paper.

Rotatable Tube Mount allows output to be obtained in any of four orientations on the film.

A Form Projector permits an overlay on top of the image projected from the CHARACTRON tube. This is obtained by using a glass slide, independent of the tube, to project an image in registration with the tube output. The program can control when the form is "flashed" (projected), but the S-C 4020 operator must manually change the slides.

Specified Image Size Generator allows the image to be expanded and returned to normal four inches under program control. The size of the expanded image may be adjusted by the Customer Engineer. However, machine alignment in expanded image is difficult.

*Raster count is the number of plottable positions on the raster in either the horizontal or vertical direction.

A Current Point Register in the S-C 4020 is used to hold the coordinates specified by the last operation. This register can be set by the programmer (by plotting a blank, if necessary, at the point desired), but not retrieved by him.

Input to the S-C 4020 may be directly from the computer through an I/O adapter or from a tape unit which is off-line. When operating off-line, an F-53 buffer is normally used to control the tape drive and to convert tape records to 36-bit control words for the S-C 4020. The F-53 also provides the means for accepting high-density, blocked information from the tape unit and transmitting the information to the S-C 4020 on demand. An alternative off-line configuration is to use the S-C 4020 with a small general purpose computer in place of the F-53. This has been accomplished successfully with a CDC 160-A and a GE 225.

ERROR MARKS

Certain types of errors are indicated on the frame by one of the following error marks.

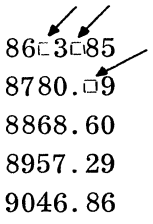
Void Mark

This is a solid circle which the S-C 4020 places in the margin when forced to proceed past an error condition. This occurs whenever a parity error in reading the tape cannot be corrected by rereading, or whenever the tape block exceeds the size of the F-53 input buffer. In either condition, the S-C 4020 will automatically try a second time to read the record. The void light comes on only when the operator "forces" the S-C 4020 to proceed after detecting an error.

Error Symbol

The CHARACTRON character □ is displayed by the S-C 4020 if a parity error occurs when printing or plotting; that is, the specifications for a symbol are garbled. It is positioned at the coordinates contained in the erroneous word. This error symbol is not displayed if the erroneous word is one which normally does not display a character; for example, an axis or vector generation word. In addition to displaying the error symbol, the tape drive is stopped when a parity error is detected. A frame with such a mark will usually have a Void Mark as well.

Error Symbol



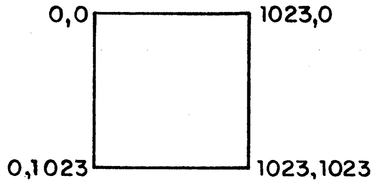
```
86□3□85
8780.□9
8868.60
8957.29
9046.86
```

Section II

BASIC S-C 4020 PROGRAMMING

The S-C 4020 plots on a grid of 1024-horizontal by 1024-vertical points. There are, therefore, over 1 million addressable positions. The center of any character may be plotted at any of these positions. However, each character in the matrix occupies an amount of space that varies from the square of 3-horizontal by 3-vertical points used by a plotting dot to the rectangle of 8-horizontal by 16-vertical points used by all characters in the typewriter mode.

The S-C 4020 grid is a reflection of the first quadrant. The origin (0, 0) is in the upper left corner, with X and Y increasing to the right and downward, respectively as shown in the figure below. This is the machine-oriented reference. (see Positioning Information regarding programmer orientation of rasters.)



The plot appears in a 4-inch by 4-inch square on the face of the CHARACTRON® shaped beam tube and is recorded on film.

Information is transmitted to the S-C 4020 through a 36-bit input register. The basic unit breaks down each 36-bit word into its component parts: the operation code and the information necessary to accomplish the specified operation. The bits, from left to right, are numbered from 0 (or S, the sign bit) through 35. Word format and operation code for each command is given in the following pages of this section.

NOTE

> The S-C 4020 requires a delay following its receipt of certain operation codes. The delay is of concern only when operating without an F-53 or other input buffer. These operations, presented in the following discussion, are marked with an asterisk.

CONTROL OPERATIONS

No information is plotted on the grid as the result of any of the following nine operations:

ADVANCE FILM*
RESET*
CARRIAGE RETURN
SELECT CAMERA 1*
SELECT CAMERA 2*
SELECT BOTH CAMERAS*
EXPAND IMAGE*
REDUCE IMAGE*
STOP TYPE

Advance Film

The ADVANCE FILM command causes the film in the camera (or cameras) selected to be advanced one frame. If the supply of film is exhausted, the command cannot be executed. An alarm is activated.

ADVANCE FILM* OPCODE 46

OP CODE	NOT USED
0 5	6 35

Reset

The RESET command performs simultaneously the functions of the ADVANCE FILM, STOP TYPE, and EXPOSE HEAVY commands. It also resets the deflection registers. In the typewriter mode, operation code 56 may replace any of the character codes shown below. Any bits following the code 56 are ignored.

RESET* (PLOTTING MODE) OPCODE 56

OP CODE	NOT USED
0 5	6 35

RESET* (TYPEWRITER MODE) OPCODE 56

CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE
0 5	6 11	12 17	18 23	24 29	30 35

Carriage Return

The CARRIAGE RETURN command is effective only in the typewriter mode. Operation code 52, replacing any of the 6-bit character codes shown below, causes an effective typing position at the left of the grid and down one line (X = 0, Y = Y last + 16) to be assumed. Spacing, therefore, may be achieved through the appearance of this code in contiguous 6-bit groups. If a CARRIAGE RETURN command is given which would specify an effective position of X = 0, Y = 1024, the position assumed is X = 0, Y = 0 of the same grid, since Y is computed modulo 2^{10}. Typing is resumed with the next legitimate character.

CARRIAGE RETURN **OPCODE 52**

CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE
0 5	6 11	12 17	18 23	24 29	30 35

Select Camera

The SELECT CAMERA commands are used to open the shutter of the camera or cameras designated and to close the shutter on the one not selected. If only one camera is mounted, it is not necessary to give a SELECT CAMERA command.

SELECT CAMERA 1* **OPCODE 41**

OP CODE	NOT USED
0 5	6 35

SELECT CAMERA 2* **OPCODE 42**

OP CODE	NOT USED
0 5	6 35

SELECT BOTH CAMERAS* **OPCODE 43**

OP CODE	NOT USED
0 5	6 35

Expand Image

The EXPAND IMAGE command increases the image size on the CHARACTRON tube face. The exact size is adjusted so that successive frames on the microfilm will abut to allow plots to extend continuously through several adjacent frames. The number of raster points remains the same and the CHARACTRON characters remain the same size; therefore, the space between characters is increased in expanded image mode. The image may be expanded uniformly in both directions in order to avoid distortion in graphs due to uneven scaling. The image remains expanded until the REDUCE IMAGE command is given.

EXPAND IMAGE* **OPCODE 44**

OP CODE	NOT USED
0 5	6 35

Reduce Image

The REDUCE IMAGE command causes the image to be reduced to four inches square and to remain at four inches until another EXPAND IMAGE command is given.

REDUCE IMAGE* **OPCODE 45**

OP CODE	NOT USED
0 5	6 35

Stop Type

The STOP TYPE command is used to return the S-C 4020 to the plotting mode from the typewriter mode of operation. Operation code 12 may replace any of the character codes shown below. Bits following the 12 are lost.

STOP TYPE **OPCODE 12**

CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE
0 5	6 11	12 17	18 23	24 29	30 35

RECORDING OPERATIONS

Information is placed on the grid by using one of the following nine commands. The length of time that characters, plotted or typed, are exposed results in their being recorded on film with heavy or light density. There is, however, no density-mode selection when drawing vectors or generating axes.

PLOT
EXPOSE LIGHT
EXPOSE HEAVY
TYPE SPECIFIED POINT
TYPE CURRENT POINT
DRAW VECTOR
GENERATE X-AXIS*
GENERATE Y-AXIS*
PROJECT FORM*

Plotting Mode

The PLOT, EXPOSE LIGHT, and EXPOSE HEAVY commands are exactly the same in that they plot the character represented by bits 18 through 23 at the point specified by the X and Y deflections. However, code 00 (PLOT) operates in the density mode to which the S-C 4020 has already been set, while 02 and 04 set the machine to the EXPOSE HEAVY and EXPOSE LIGHT modes, respectively. All plotting or typing subsequent to the recognition of an 02 or 04 operation code is done in the density mode thus established. The EXPOSE HEAVY mode can be terminated only by the reception of an EXPOSE LIGHT code. The EXPOSE LIGHT mode can be changed to the EXPOSE HEAVY mode by either an EXPOSE HEAVY code or a RESET code.

PLOT **OPCODE 00**

OP CODE	NOT USED	X DEFLECTION	CHAR CODE	NOT USED	Y DEFLECTION
0 5	6 7	8 17	18 23	24 25	26 35

EXPOSE LIGHT **OPCODE 04**

OP CODE	NOT USED	X DEFLECTION	CHAR CODE	NOT USED	Y DEFLECTION
0 5	6 7	8 17	18 23	24 25	26 35

EXPOSE HEAVY **OPCODE 02**

OP CODE	NOT USED	X DEFLECTION	CHAR CODE	NOT USED	Y DEFLECTION
0 5	6 7	8 17	18 23	24 25	26 35

Typewriter Mode

The typewriter mode is initiated by either a TYPE SPECIFIED POINT or a TYPE CURRENT POINT command. A STOP TYPE or RESET operation returns the S-C 4020 to the plotting mode.

In the typewriter mode, the grid of 1024-horizontal by 1024-vertical points may be regarded as containing 64 lines of 128 typing spaces each. Once typing has been commenced, it continues across the line until the 128th space has been filled or a CARRIAGE RETURN command has been executed. In either case, typing continues on the next lower line at the extreme left of the grid (X = 0, Y = Y last + 16).

An ADVANCE FILM command must be given after the 64th line has been typed; otherwise, typing will continue on the first line of the same grid at X = Y = 0 with resulting overlay. If vertical alignment of characters is desired, X_0 should be either 0 or divisible by eight.

Note that, in the typewriter mode, character codes 12, 52, and 56 are not legitimate in that they do not cause a character to be typed but instead cause an operation to be performed (STOP TYPE, CARRIAGE RETURN, and RESET, respectively).

TYPE SPECIFIED POINT (FIRST WORD) OPCODE 20

OP CODE	NOT USED	X DEFLECTION	CHAR CODE	NOT USED	Y DEFLECTION
0 5	6 7	8 17	18 23	24 25	26 35

TYPE CURRENT POINT (FIRST WORD) OPCODE 22

OP CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE
0 5	6 11	12 17	18 23	24 29	30 35

For both of the above commands, all words after the first have the following format:

(SUBSEQUENT WORDS)

CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE	CHAR CODE
0 5	6 11	12 17	18 23	24 29	30 35

One of the character codes of the subsequent word's format may be replaced by the operation code for CARRIAGE RETURN. Characters specified by succeeding bits are typed. The six bits following the last character to be typed must contain either the

RESET or STOP TYPE operation code. Any bits still remaining in this word are lost.

The current point is defined as the point specified by the last operation performed, i.e., either the last point plotted or typed, or the origin of the last vector or axis drawn. It is advisable, therefore, with a TYPE CURRENT POINT command to make the first two characters typed blanks.

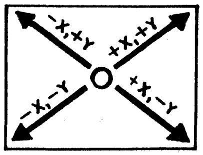
Vector Generation

The DRAW VECTOR command causes a vector to be drawn from the point specified by the X and Y deflections to the resultant of the X and Y components.

DRAW VECTOR **OPCODE 1 BIT IN POSITIONS 0 AND 1**

OP CODE	X COMPONENT	X DEFLECTION	SIGN X COMPONENT	SIGN Y COMPONENT	Y COMPONENT	Y DEFLECTION
0 1	2 7	8 17	18	19	20 25	26 35

The direction is determined by the signs of the X and Y components as shown below. A one in bit 18 indicates a plus X component; a zero means a minus X component. Bit 19 similarly indicates the sign of the Y component.



The maximum component of the vector which may be drawn is 1/16 of full scale (64 plotting positions).

Axis Generation

The GENERATE X-AXIS command causes a line to be drawn parallel to the X-axis. The line starts at the point designated by the X and Y deflections and extends to the right to the stop point. The ones complement of the X deflection of the stop point is placed in bits 6, 7, and 18 through 25 of the command word. If the stop point bits contain all zero, the axis generated will extend to the right boundary of the frame. Thus, programs prepared for early models of the S-C 4020 will generate full axes in machines equipped with the specified stop point axis generator. The stop point must be at least 64 raster counts to the right of the start point.

GENERATE X-AXIS* **OPCODE 30**

OP CODE	STOP POINT	X DEFLECTION	STOP POINT	Y DEFLECTION
0 5	6 7	8 17	18 25	26 35

The GENERATE Y-AXIS command causes a line to be drawn parallel to the Y-axis. The line starts at the point designated by the X and Y deflections and extends upward to the stop point. The stop point is not complemented for a Y-axis. The Y deflection of the stop point is placed in bits 6, 7, and 18 through 25 of the command word. The stop point must be at least 64 raster counts upward from the start point. Programs prepared for early models of the S-C 4020 will generate full length axes in machines equipped with the specified stop point axis generator.

GENERATE Y-AXIS* **OPCODE 32**

OP CODE	STOP POINT	X DEFLECTION	STOP POINT	Y DEFLECTION
0 5	6 7	8 17	18 25	26 35

Form Projection

The PROJECT FORM command causes the information on the form slide to be superimposed on the film, along with the data displayed on the tube. This command may be given only once every 100 milliseconds.

PROJECT FORM* **OPCODE 50**

OP CODE	NOT USED
0 5	6 35

S-C 4020 OPERATION CODES

The 6-bit codes for commands, written in octal notation, are given below.

Operation	Code	Operation	Code
PLOT	00	SELECT CAMERA 2*	42
EXPOSE HEAVY	02	SELECT BOTH CAMERAS*	43
EXPOSE LIGHT	04	EXPAND IMAGE*	44
STOP TYPE	12	REDUCE IMAGE*	45
TYPE SPECIFIED POINT	20	ADVANCE FILM*	46
TYPE CURRENT POINT	22	PROJECT FORM*	50
GENERATE X-AXIS*	30	CARRIAGE RETURN	52
GENERATE Y-AXIS*	32	RESET*	56
SELECT CAMERA 1*	41	DRAW VECTOR	6x or 7x

S-C 4020 STANDARD CHARACTER MATRIX

The standard scientific character set used by the S-C 4020 is illustrated in Figure 2-1.

S-C 4020 STANDARD SCIENTIFIC CHARACTER MATRIX

	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	8	9	∂	=	"	'	δ	α
2	+	A	B	C	D	E	F	G
3	H	I	π	•	)	β	±	?
4	−	J	K	L	M	N	O	P
5	Q	R	·	$	*	γ	~	d
6		/	S	T	U	V	W	X
7	Y	Z	°	,	(	∫	Σ	□

BCD Code in Octal = Number at Left and Above Character
Note: 12, 52, and 56 cannot be used in Typewriter Mode

Figure 2-1

OPTIONAL COMMANDS

Specified Intensity Plot

This command is an additional PLOT command. The intensity of the plotted character is controlled by regulating the exposure time. The sixteen levels of intensity are specified by setting bits 6, 7, 24, 25 from 0000 (lightest) to 1111 (darkest). Bit 6 is the high order bit and bit 25 is the low order bit. The operation of this command does not affect the model intensity for other plotting established by a prior EXPOSE HEAVY or EXPOSE LIGHT command. S-C 4020's not equipped with the Specified Plot Intensity option, treat this command as a PLOT with OPCODE 00.

SPECIFIED INTENSITY PLOT OPCODE 01

OP CODE	SPEC INT	X DEFLECTION	CHAR CODE	SPEC INT	Y DEFLECTION
0 5	6 7	8 17	18 23	24 25	26 35

STANDARDIZED S-C 4020 PROGRAMMING ROUTINES

The Stromberg-Carlson Corporation provides, as a service to users, programming support for the S-C 4020. In 1961 a set of routines was generated by Stromberg-Carlson and distributed to users and potential users of the S-C 4020. These routines were a tremendous boon to programmers and are still in use in some installations. Over the months of usage many users have developed subroutines of a more sophisticated nature to reduce the programming task to produce S-C 4020 output.

The various divisions of the North American Aviation Co., cooperatively developed a system of subroutines for use on their IBM 7090 computers within the framework of their operating system. Since the NAA system has been contributed to the UAIDE library, Stromberg-Carlson has adopted these routines as the base for a standardized programming system. It is intended that this system of routines will be maintained and upgraded by Stromberg-Carlson to keep pace with the continuing improvement program for the S-C 4020 and the operating systems in use by the majority of the users of the S-C 4020. As new generalized routines become available they will be added to the system. Changes to the system will be announced by systems bulletins to be issued by Stromberg-Carlson as required.

OPERATING SYSTEMS: PLOT.., (PLOT)

The programming system is presently available for use under IBSYS for the IBM 7090/7094 computers in either the FORTRAN II, Version 3 monitor or the IBJOB, Version 9 monitor.

All output from the programming system is handled by a routine called (PLOT) in the FORTRAN II version and PLOT.. in the IBJOB version. These routines store the word to be transmitted to the S-C 4020 in a core buffer. When the buffer is filled the routine causes the contents of the buffer to be written on tape. The system is double buffered to take advantage of the I/O buffering of the IBM 7090/7094. The buffer size is set by the command:

```
FORTRAN II      (SIZE)        EQU     680       (deck OUTV)
IBJOB            buffer size is set by the FILE card (deck UN16)
```

A buffer size of 680 words is maximum and can only be used on machines not using the F53 or on machines using the F53 for 4096 characters. For the small size F53, a buffer size of 170 words is maximum. The buffer size is set at assembly time and should only be changed by reassembling the source deck labeled OUTV with the appropriate change in the card defining (SIZE). IBM 7090/7094 DCS systems may further restrict the buffer size.

On the first entry into (PLOT) or PLOT.. control is transferred to an initialization routine stored in the second buffer region. This routine causes the first three commands transmitted to the S-C 4020 to be a Reset, Select Both Cameras, and Reduce Image Size. This assures that the output will start on a clean sheet of paper, the image size will be small, and that both cameras are selected. The routine then proceeds to output an ID frame, followed by another Reset.

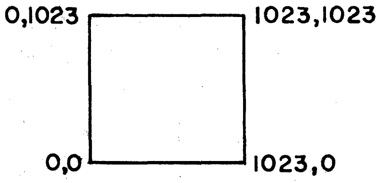
THE IDENTIFICATION FRAME

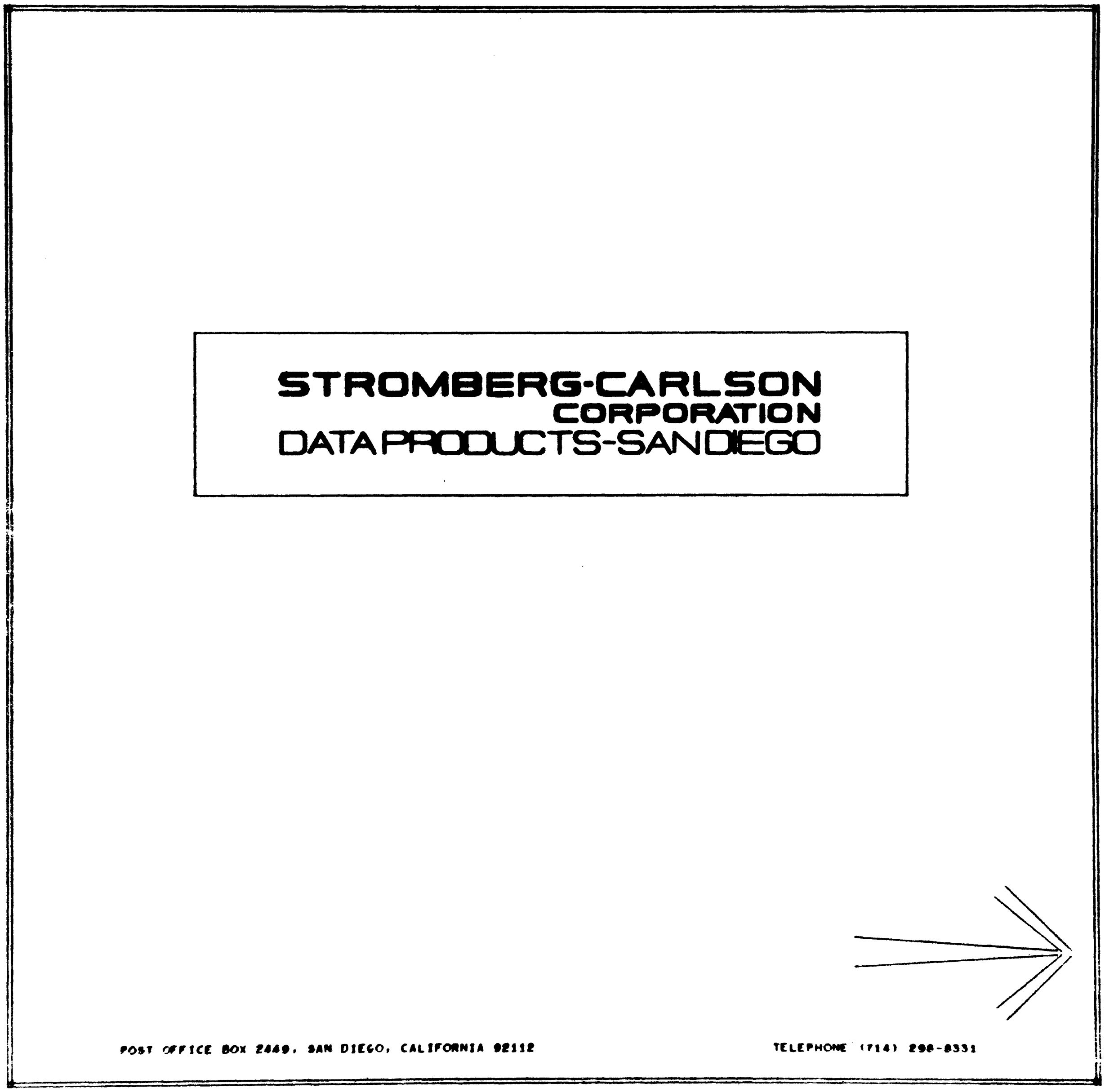
The chief purpose of the ID frame (Figure 2-2) is to enable the job dispatcher to return the developed output to the right programmer. However, since it is always generated in the same way, the frame can also serve as a clue to output quality. Several machine features are demonstrated. The information printed in small characters is an example of CHARACTRON characters, printed in the typewriter mode, using bright intensity. These characters are displayed only once; i.e., they are not replotted for greater intensity. Since the typewriter mode is used, the spacing of characters within words is determined by the equipment, not by the programmer. An example of vectors may be seen in the arrow. Since the lines that form the shaft of the arrow are longer than the maximum vector length, each line is made up of three vectors joined end-to-end. The border is formed by pairs of axis lines. On the ID Frame, the outermost axis lines lie on the outermost raster coordinates. The border, therefore, shows the maximum size of a frame (and, incidentally, shows how close the frame is to being square). The small box is also made with axes and tests the accuracy of stop points.

The coding to produce the ID frame can be easily altered to communicate with the local operating system in order to place the required ID information for job identification in the ID frame. A few moments study by a systems programmer should be all that is required.

POSITIONING INFORMATION

For all the routines discussed in the following sections, the raster is reflected about the X axis to place the origin (0, 0) in the lower left-hand corner and (1023, 1023) is in the upper right-hand corner. The subroutine does the transformation to the S-C 4020 system described in the section on basic programming. Any position in the frame can be addressed by specifying the number of raster counts from the left and from the bottom of the frame. Fractional distances cannot be addressed. The programmer raster orientation is shown in the figure below.





STROMBERG-CARLSON
CORPORATION
DATA PRODUCTS-SAN DIEGO

POST OFFICE BOX 2449, SAN DIEGO, CALIFORNIA 92112

TELEPHONE (714) 298-8331

Figure 2-2

An Identification Frame

SCALING INFORMATION

The typical S-C 4020 application involves the display of some physical phenomenon scaled to the dimensions of the available plotting area. For the most part, the programmer can do his planning on the basis of the physical model. The system provides subprograms that will accept information given in terms of the model and perform the necessary scaling and conversion to relate the information to the raster area.

SYMBOL CONVENTIONS

In the subprogram descriptions, floating point variable names have been assigned in accordance with the FORTRAN convention. If integer variables are required, names beginning with I, J, K, L, M, or N are used. The actual floating point number (or integer) may be used in the argument lists in place of a floating point (or integer) variable name, if the argument represents input to the subprogram. Constants should never be substituted for argument names that represent output from the subprogram.

Section III

S-C 4020 CONTROL FUNCTIONS

SELECTING THE CAMERA: CAMRAV

At the outset of every job, both cameras are selected before drawing the ID frame. If no other camera selection is done all output will be on both cameras. A manual override is provided so that the S-C 4020 operator can change the camera selection at will.

Camera selection is implemented by the following statement:

```
CALL CAMRAV (N)
```

N = 9	The 9-inch camera will be called and the 35mm camera will be dropped out of select.
N = 35	The 35mm camera will be called and the 9-inch camera will be dropped out of select.

N = Any other number. Both cameras will be selected.

Note that selection of one camera causes the S-C 4020 display to be photographed only by the camera selected. That camera will be used until another CALL CAMRAV statement changes the selection. If the 9-inch camera is selected, and a CALL CAMRAV (35) is encountered, usage of the 9-inch camera will be discontinued and only the 35mm camera will be employed.

Therefore, CALL CAMRAV(935) is not equivalent to the two statements:

```
CALL CAMRAV (9)
CALL CAMRAV (35)
```

ADVANCING THE FILM: FRAMEV, RESETV

Following the generation of the ID frame, a reset command is given by the program. This leaves the S-C 4020 with blank film in both cameras, set in the Expose Heavy mode, and not in the Typewriter mode. During the programming, it is the responsibility of the programmer to advance the film whenever a new page is to be started. A subroutine is provided which can be called by the statement CALL FRAMEV (N). The argument N controls the display of the job ID, a frame count, and the short vectors which indicate the corners of the frame. The effect of the argument and different values is as follows:

N	Effect
3	Film advanced, no print out.
2	Film advanced, corners drawn, no ID.
1	Film advanced, ID printed, no corners.
0	Film advanced, corners drawn, ID printed.

Another subroutine called by the statement CALL RESETV (N) also causes the film to be advanced. In addition, this command sets the exposure to heavy and assures that the S-C 4020 is not in the typewriter mode. The sense of the parameter is the same as for FRAMEV. When drawing graphs and using the subroutine GRID1V to be described later, the film advance is controlled by a parameter of GRID1V which in turn calls FRAMEV. No argument in FRAMEV or RESETV has the effect of a zero argument.

FRAME COUNT RETRIEVAL: NOFRV

NOFRV enables the programmer to obtain the current output frame counts. He can use this number as a "cut-off" point to prevent a wasteful loop. (Some programmers have used it, in conjunction with a timing routine, to determine how much computing time was needed to produce a frame, or for other accounting purposes within their programs.) The call statement is:

```
CALL NOFRV (N9, N35)
```

N9	Name of location in which current count of 9-inch camera frames will be stored.
N35	Name of location in which current count of 35mm camera frames will be stored.

Since both arguments represent output from the routine, they must be variables.

A special case of NOFRV is necessary to obtain the current frame count numbers printed (if different), instead of the current number of output frames on the S-C 4020. This occurs when the subprogram FRMNOV has been called. When a third argument is used in the calling sequence, the current camera frame counts appearing on the

S-C 4020 output are returned in the first two arguments. The call statement is:

```
        CALL          NOFRV (N9, N35, ANY)
```

The format of the 3rd argument is immaterial.

INTENSITY SELECTION: BRITEV, FAINTV, RESETV

GRID1V ensures that the bright intensity mode is on. Normally, this intensity mode should be left on, since experience has shown that it produces the best results. If the programmer wants to change this setting to the faint mode, he can use the following statement:

```
        CALL FAINTV
```

Then, to restore the bright intensity mode, he can use the statement:

```
        CALL BRITEV
```

RESETV is explained under FRAMEV and STOPTV.

IMAGE SIZE CONTROL: BIGV, SMALLV

The image size may be selected by the programmer to be 4.00 inches square or to be expanded, normally to 4.53 inches square. The program statement

```
        CALL BIGV
```

expands the image; and the statement,

```
        CALL SMALLV
```

reduces the image. The system initialization program sets the image size to small.

BYPASSING THE ID FRAME AND SPECIAL TREATMENT OF FRAME COUNTS: FRMNOV

FRMNOV serves two purposes. They are both for special applications and should not be of concern to the programmer for normal usage.

The first use of FRMNOV is to bypass the ID frame and initialization routine as discussed on page 2-12. This is done by calling FRMNOV before any other call is made that puts information on the S-C 4020 output tape. The frame normally following the ID frame will then have frame counts beginning with the arguments supplied in the CALL FRMNOV. This is normally 1 and 1 for each camera frame count. The call statement is:

CALL FRMNOV (M9, M35)

M9, M35	Normally, these are both 1. Or they may be whatever framecounts are desired for the first (which is yet to be advanced) S-C 4020 output frame.

The second use of FRMNOV is to set either or both camera frame counts at any time in the program. If only one camera frame count is set, the other is unaffected. As in the above usage, the frame count setting made will appear on the following frame, not the frame currently exposed on the tube of the S-C 4020. The call statement is:

CALL FRMNOV (M9) Set 9-inch camera frame count

or

CALL FRMNOV (±M9, M35) Set only 35mm camera frame count if M9 is negative. Set both camera frame counts if M9 is positive.

The frame count settings will apply to the next frame to be advanced.

CHANGING THE SYSTEMS ROUTINES OUTPUT TAPE: TPNUMV

TPNUMV is used for special applications and should not be of concern to the programmer in normal usage.

The system as distributed will write output on tape A7 in the FORTRAN II version and on unit LB4 inthe IBJOB version.

The output tape can be changed at assembly time in either version. In the FORTRAN II version, the decrement of the location CRTAPE in deck OUTV must be set to the logical tape number of the desired output tape unit. In the IBJOB version, an output file must be redefined.

At object time, in FORTRAN II, the output tape can be changed with the FORTRAN statement:

```
CALL TPNUMV (LOGNUM)
```

which causes all successive transfers from core to tape to be written on the tape designated by the logical number LOGNUM. The capability of changing the output tape during object time of an IBJOB run is not currently available.

EMPTYING THE OUTPUT BUFFER: PLTND

The S-C 4020 output buffer will not be dumped on tape until it is full unless the programmer forces a dump for his own purposes. The current contents of the S-C 4020 output buffer can be dumped by the FORTRAN statement:

```
CALL PLTND
```

Programmers using an S-C 4020 not equipped with an F53 or other input buffer must remember to dump the buffers and cause a tape gap following each time a control routine is called. Since the buffers are not dumped until full, the last command of any program using the S-C 4020 programming system must be a CALL PLTND followed by a statement to write an END OF FILE on the output tape.

Section IV

GRAPHS

GRAPHING DATA: KWKPLT

The purpose of this routine is to provide the programmer with a quick look at the relationship between two variables. KWKPLT will automatically provide the programmer with a series of linearly connected points on a scaled linear grid with or without identification printing.

The calling sequence without identification printing is:

```
CALL                KWKPLT              ( X,   Y, N)
```

where X = starting location of a forward stored array of floating point numbers representing the X-coordinates.

Y = starting location of a forward stored array of floating point numbers representing the Y-coordinates.

N = number of points to be plotted.

It is not necessary to arrange the coordinates in an increasing or decreasing order of magnitude. If the table of X-coordinates are not in ascending order, KWKPLT will rearrange them in ascending order within the table. The Y-coordinates will be arranged accordingly.

The calling sequence with identification printing is:

```
CALL                KWKPLT         ( X,   Y, N, 18H(LH), 18H(LV))
```

where X, Y, and N are the same as given above.

(LH) = 18 character identification for the X-coordinates.

(LV) = 18 character identification for the Y-coordinates.

The printing routine assumes a full 18 characters including blanks.

Section V.

GRIDS

GENERATING A GRID: GRID1V

In many ways, plotting on the S-C 4020 is very much like plotting on a sheet of graph paper, but there are also distinct differences. For one thing, the programmer must create the grid; the film frame is completely blank to start with.

Although every line of the grid must be specified on the S-C 4020, there are advantages to this situation. A hand-plotted graph must be adapted to some preprinted form; more frequently than not, this means that some plotting area must be sacrificed in order to use the most convenient scale.

On the S-C 4020, the programmer can select a scale that will be easy to read and that will accommodate the entire range of data. The number of light grid lines, the number of emphasized grid lines, and the spacing between lines can be chosen to suit the plot. The programmer is not restricted to the use of a single form for a variety of plots. For each graph, a new grid can be tailored to the data.

The easiest way to create a grid for S-C 4020 plots is to call the GRID1V subprogram. At the outset, GRID1V makes certain that the Typewriter Mode is off, and that the Bright Intensity Mode is on.

GRID1V will produce a grid which has some lines emphasized and some lines labeled. Margin space (which may be used for titles) will be reserved at the top, left side, and bottom of the grid. Normally, the title margin spaces are 24 raster counts wide.

Upon completion of GRID1V, scale factors will have been established and made available (internally) for the conversion requirements of other subprograms; i. e., the conversion of floating point coordinates into raster coordinates.

The call statement for GRID1V appears below, with a description of the arguments.

```
CALL GRID1V (L, XL, XR, YB, YT, DX, DY, ±N, ±M, ±I, ±J,
             ±NX, ±NY)
```

L — This integer argument controls the film advance and frame identification display:

L = 0 — The film will be advanced, ID will be printed but the corner marks will be suppressed.

L = 1 — The film will be advanced. The job number and frame counts will be placed in the upper right corner of the frame.

L = 2 — The film will not be advanced and the identification information will not be displayed.

L = 3 — The film will be advanced but the identification information will be suppressed.

L = 4 — The film will be advanced and both the ID and the corner marks will be suppressed.

XL, XR — Floating point values of X for the left-most and right-most limits of the grid.

YB, YT — Floating point values for the bottom limit and the top limit of the grid.

After margin space for titles and labels has been reserved, the limits of the remaining space are assigned the data values given for XL, XR, YB, and YT. Scale factors are then computed; they will remain in effect until another GRID1V statement is made (or until other action is taken to compute new scale factors).

DX, DY — Floating point data increment at which vertical (specified by DX) and horizontal (specified by DY) grid lines will be displayed. If 0.0, no lines will be shown.

Positions are stepped off in DX increments in the positive and negative directions from X = 0, and in DY increments in the positive and negative directions from Y = 0.

N, M — Fixed point integers that cause every Nth vertical grid line and every Mth horizontal grid line to be retraced for emphasis. If N (or M) is zero, no vertical (or horizontal) lines will be emphasized.

To force the grid to be square, a negative sign should be used on N and/or M. (If either N or M is zero, the negative sign should go on both N and M.)

I, J — Fixed point integers which cause every Ith vertical line and every Jth horizontal line to be labeled. If I (or J) is zero, no vertical (or horizontal) lines will be labeled.

If I and J are positive, the line labels will lie along the X = 0 and Y = 0 lines, provided these lines are within the grid limits. If X = 0 (or Y = 0) does not fall within the grid limits, labels will be placed in a space reserved at the left (or at the bottom) of the grid.

Negative signs can be used on I and/or J to force labels outside the grid. Label space is reserved at the left if I is negative, or at the bottom if J is negative, and labels will be placed in these reserved spaces. Note that label margin space is in addition to the margin reserved for titles.

NX, NY — Fixed point integers indicating the number of characters to be displayed in the labels of vertical and horizontal lines.

+NX, +NY — The labels will be in a decimal format similar to the F-type format. In specifying +NX and +NY, a decimal point must be counted as one of the NX or NY characters, but the sign is not counted. The largest number of digits permitted is 6 (or 7 if one character is a decimal point).

-NX, -NY — The labels will be in scientific notation. (Example: $1.25\text{x}10^{+02}$.) NX indicates the number of significant figures in the labels of vertical grid lines, and NY indicates the same for the labels of horizontal lines. The sign, decimal point, and exponent will be displayed in addition to NX (or NY) characters. NX (or NY) must not be greater than 6.

Examples of GRID1V Usage

Figures 5-1 through 5-9 are examples of the effect of the various parameters in the

GRID1V call statement. These examples are reproduced from S-C 4020 output. The call statement to produce each graph is printed by the S-C 4020 on the frame with the grid.

Figure 5-1 is a simple grid with the x=0, y=0 lines crossing in the middle of the grid. The numeric labels have been placed along the x=0, y=0 lines. For simplicity in the illustration, constants were used in the parameter list. In actual usage, variable names may be substituted for any parameter.

Figure 5-2 is similar to Figure 5-1 except that the XL, XR and YB, YT have been reversed to show that the scaling routines have no difficulty handling data which decreases from left to right and bottom to top.

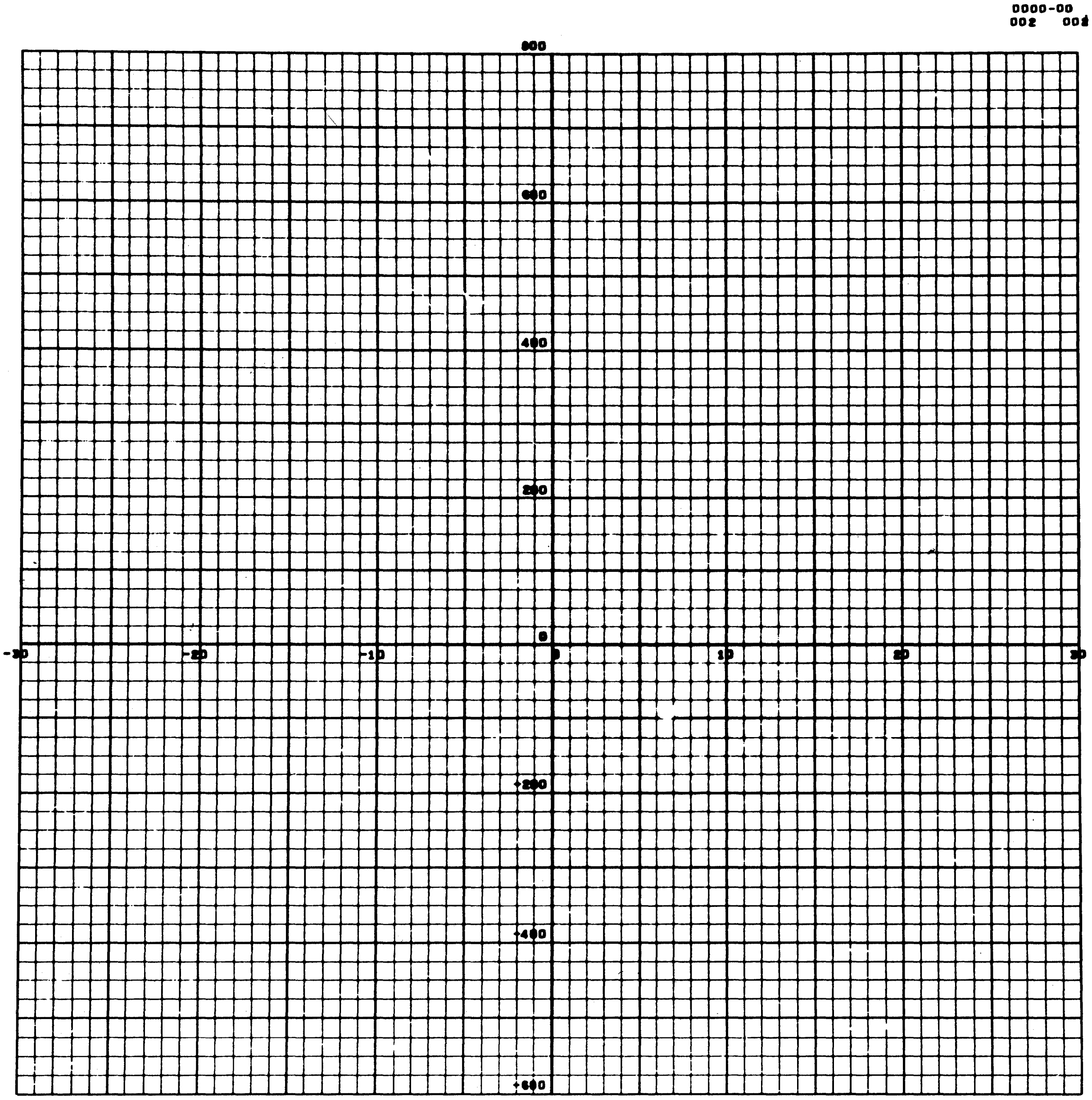
Figure 5-3 illustrates the effect of negative values for I, J in the parameter list. Note that the numeric labels are outside the grid and that the margins have been increased to accommodate the labels.

Figures 5-4 and 5-5 show the same grid with labels in integer notation and scientific notation.

Figure 5-6 has been double exposed to show the effect of negative arguments at N or M. The outer frame was produced by the first call statement with the positive argument for N and M. The grid utilizes the maximum available space in both directions and is taller than it is wide. The second call statement with negative N and M forced the frame to be shorter in the Y direction in order to be square. It is important to have a square grid when representing geometric figures such as a circle or a square. Note that the first parameter of the second call statement is a 2 which inhibits the frame advance.

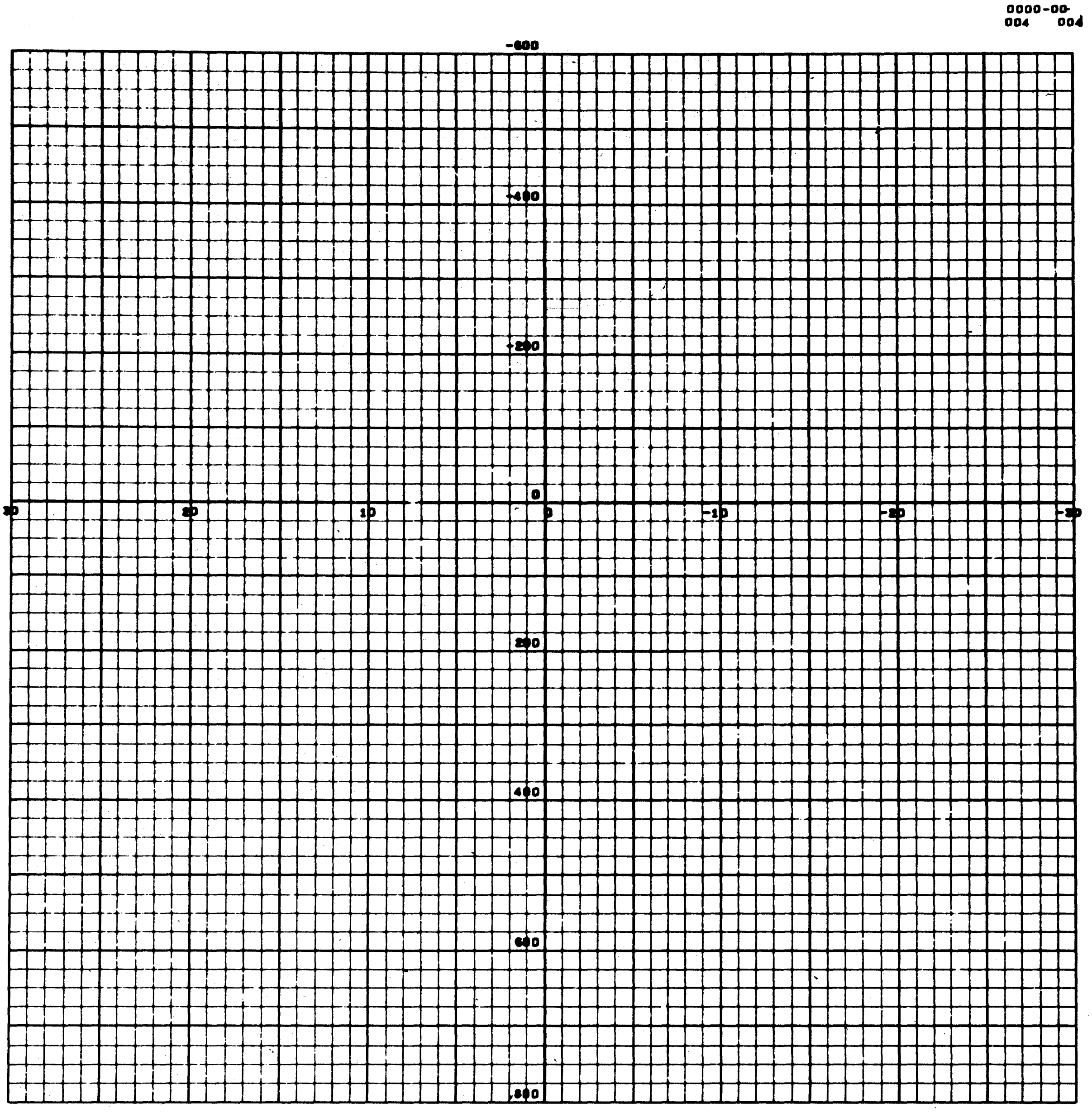
Figure 5-7 illustrates the use of the routine DXDYV to compute some of the values for the GRID1V parameter statement. DXDYV is explained on page 5-13.

Figures 5-8 and 5-9 show the influence upon the grid of the density factor used by DXDYV. For the case of Figure 5-8, a density factor of 8.0 was specified as the 8th argument of DXDYV. A larger factor, 20.0, caused DXDYV to derive values of DX and DY such that the grid in Figure 5-9 is less dense.



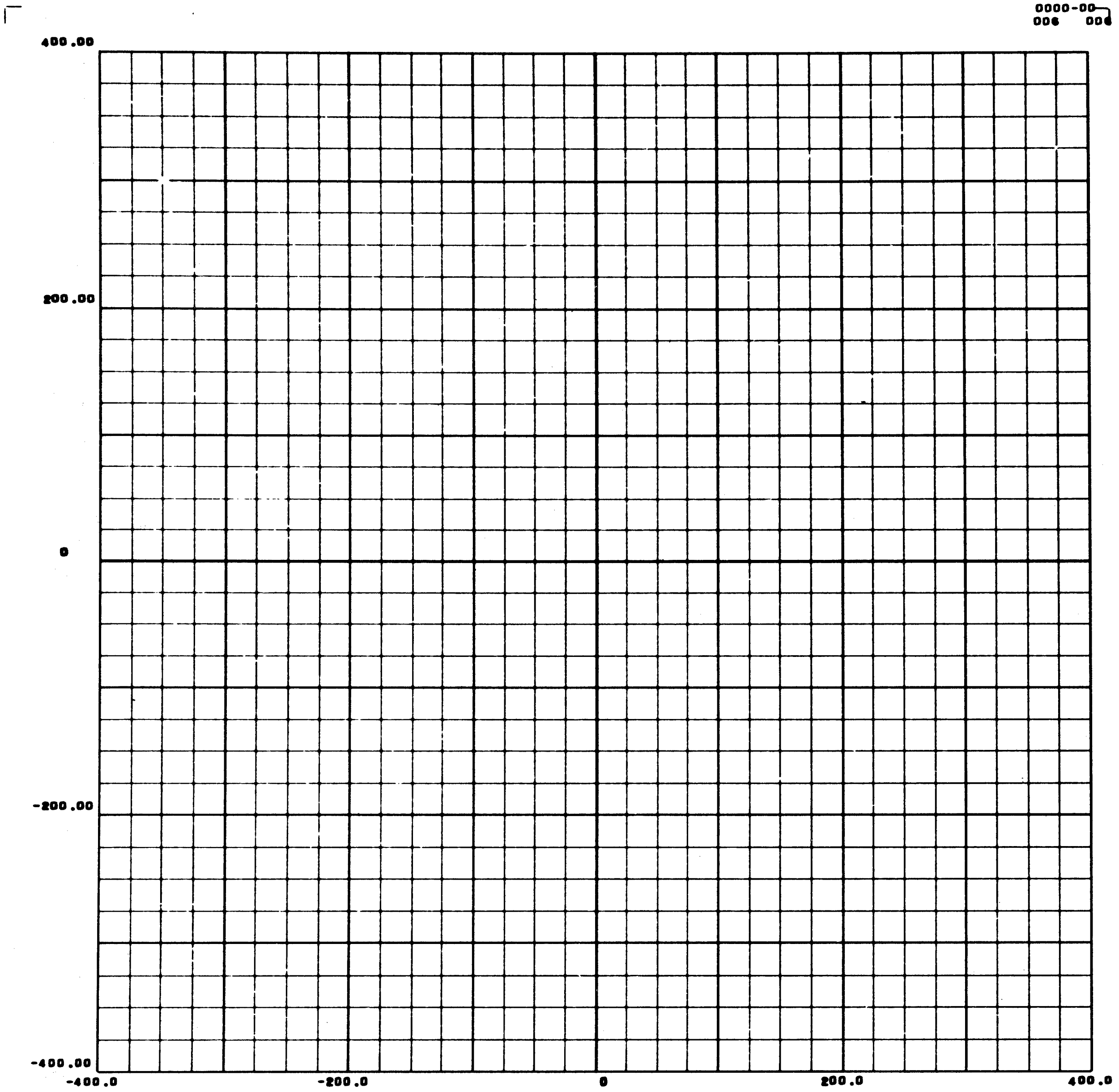
```
CALL GRID1V (1,-30.0, 30.0,-600.0, 800.0, 1.0, 25.0, 5, 4, 10, 8, 2, 3 )
```

Figure 5-1



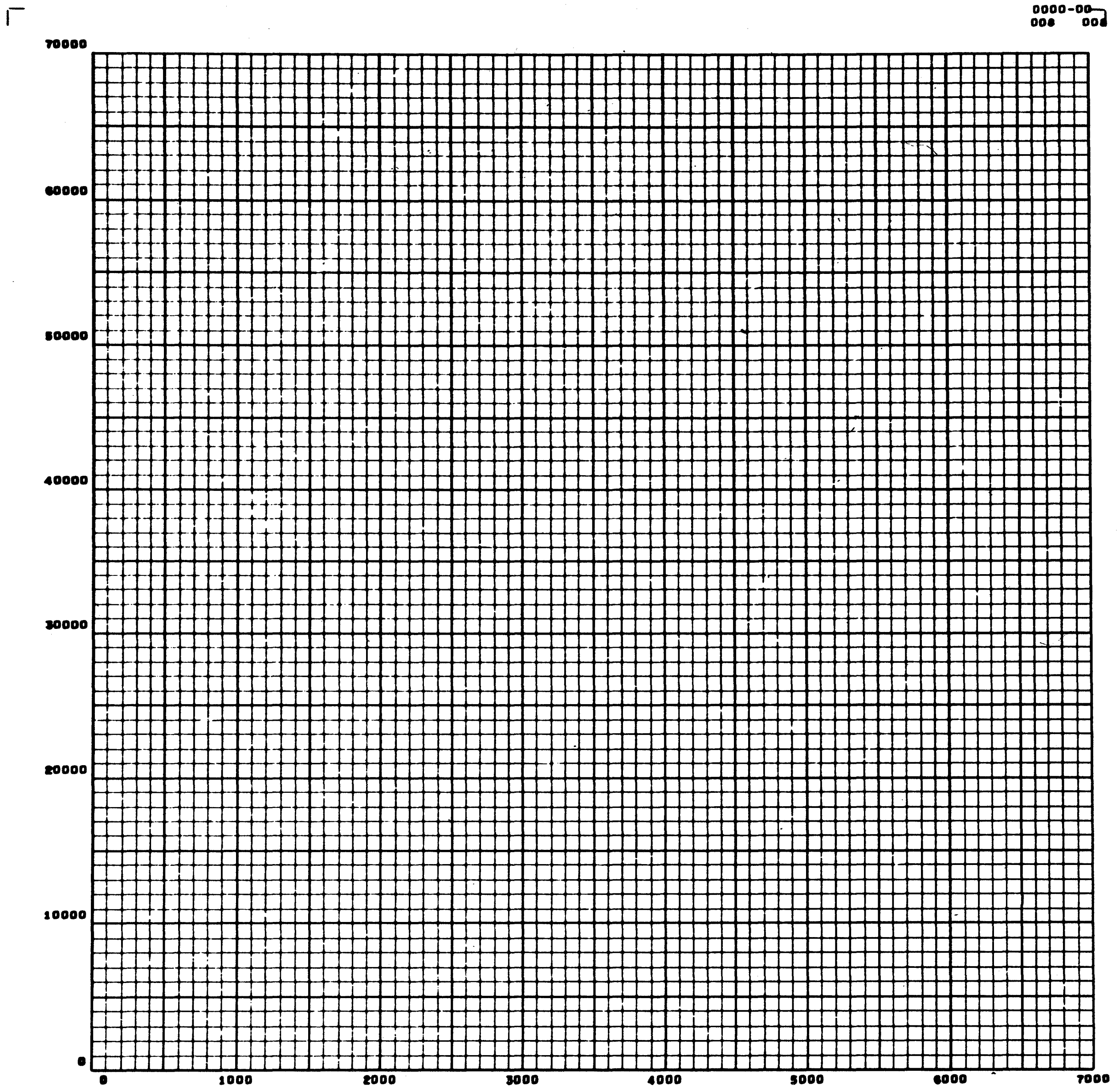
```
CALL GRID1V (1, 30.0,-30.0, 800.0,-600.0, 1.0, 25.0, 5, 4, 10, 8, 2, 3 )
```

Figure 5-2



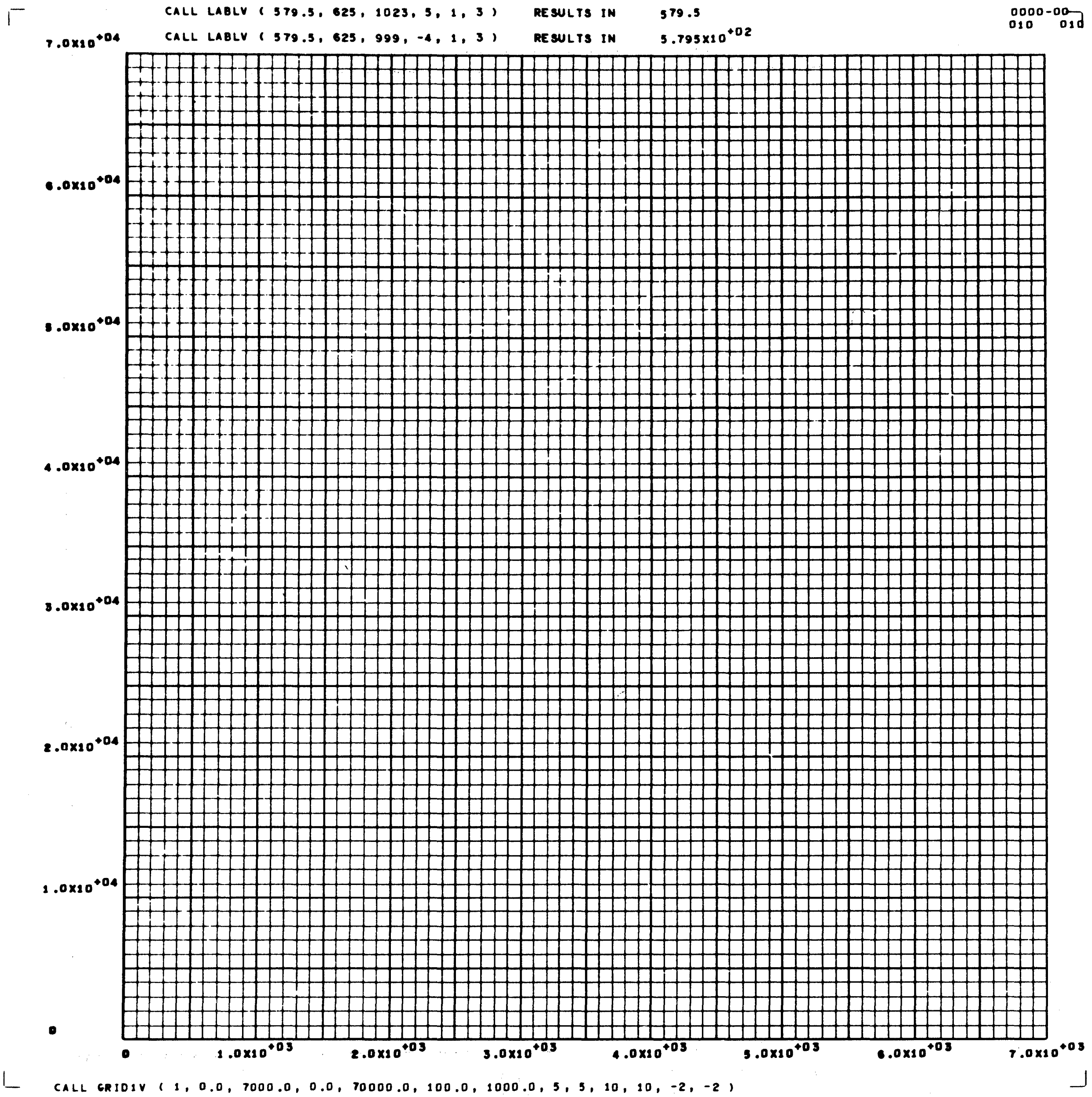
```
CALL GRIDIV (1,-400.0, 400.0,-400.0, 400.0, 25.0, 25.0, 4, 4,-8,-8, 5, 6 )
```

Figure 5-3



```
CALL GRID1V ( 1, 0.0, 7000.0, 0.0, 70000.0, 100.0, 1000.0, 5, 5, 10, 10, 4, 5 )
```

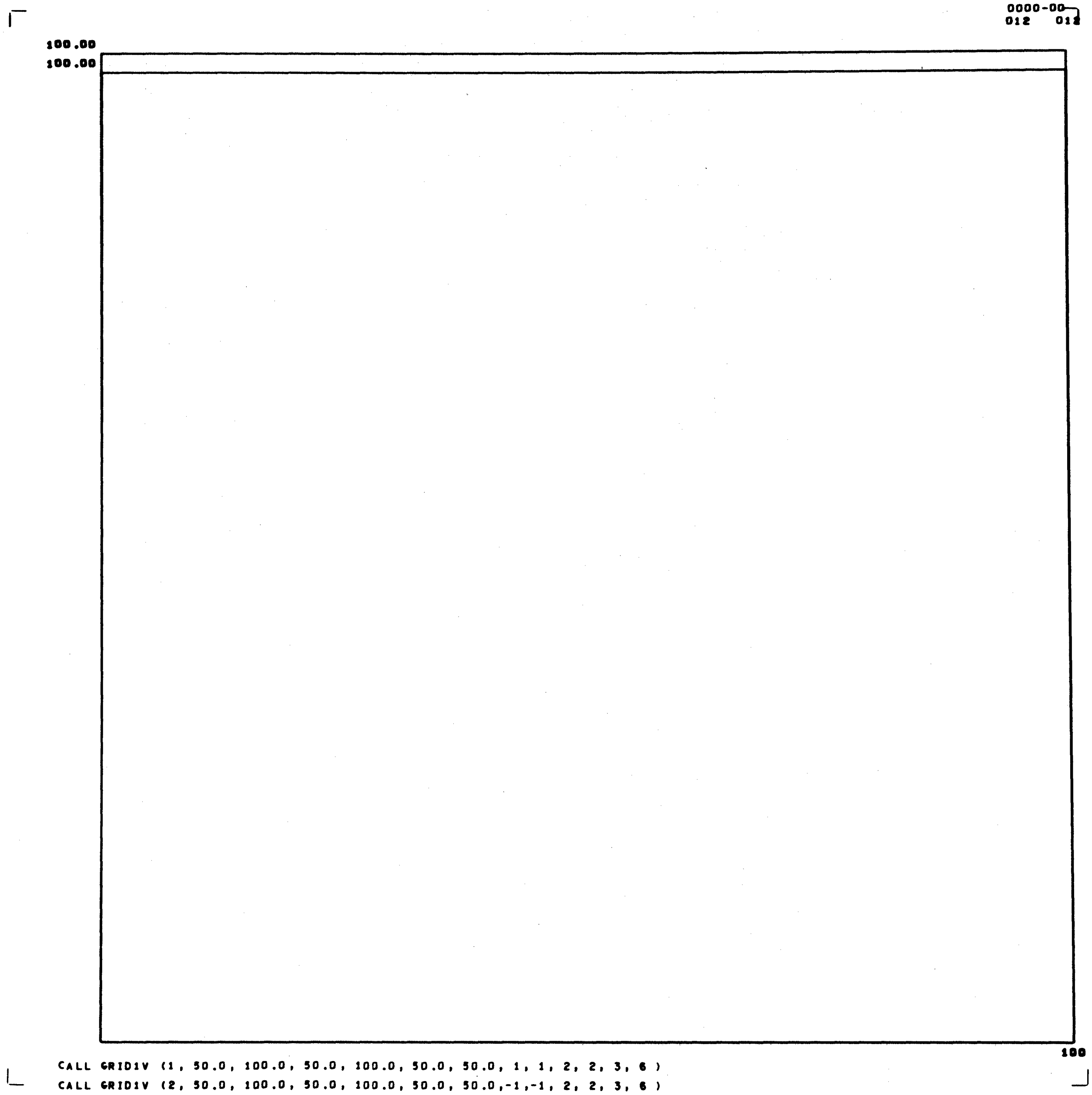
Figure 5-4



```
CALL LABLV ( 579.5, 625, 1023, 5, 1, 3 )    RESULTS IN    579.5
CALL LABLV ( 579.5, 625, 999, -4, 1, 3 )    RESULTS IN    5.795X10+02
```

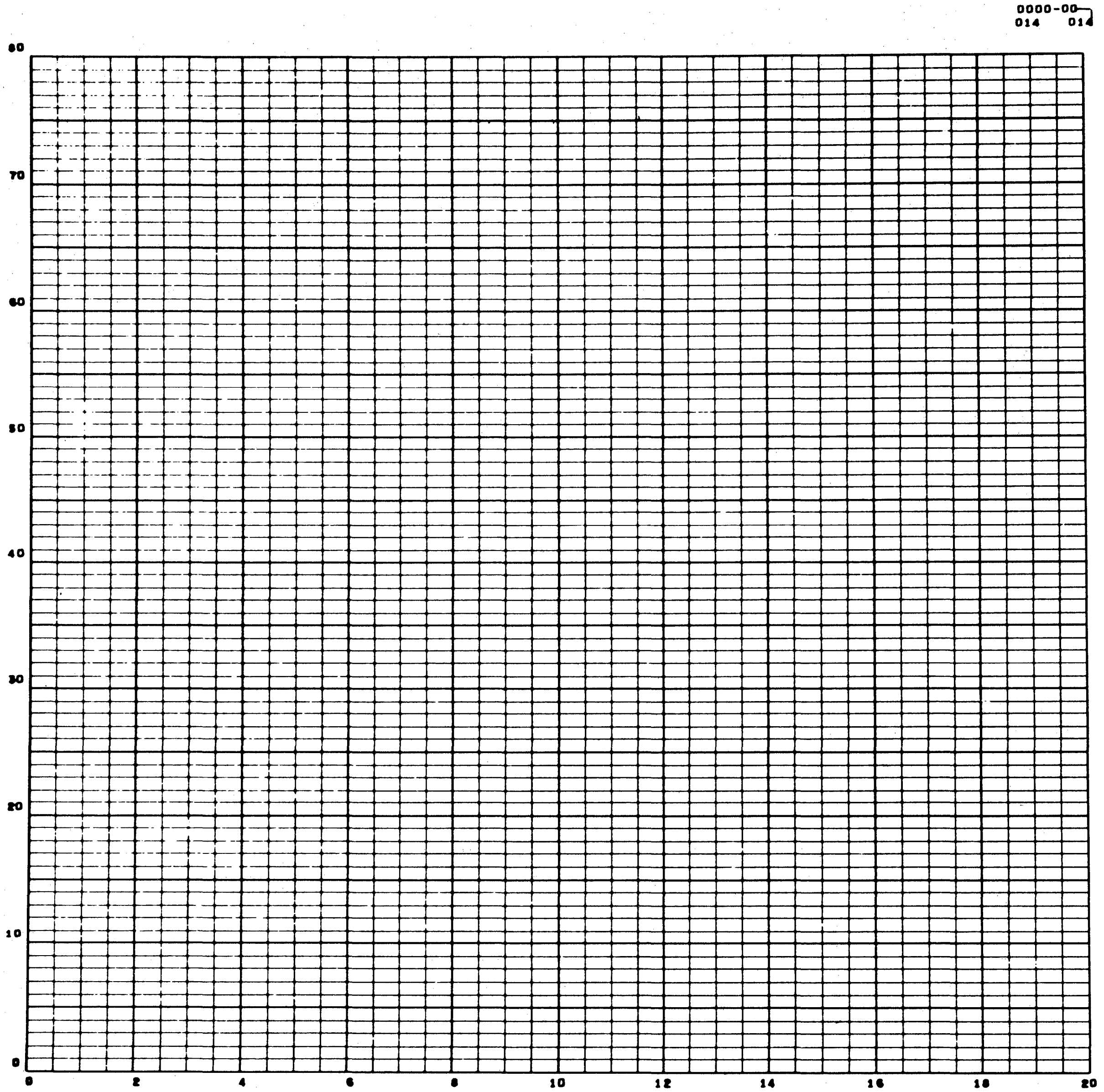
```
CALL GRIDIV ( 1, 0.0, 7000.0, 0.0, 70000.0, 100.0, 1000.0, 5, 5, 10, 10, -2, -2 )
```

Figure 5-5



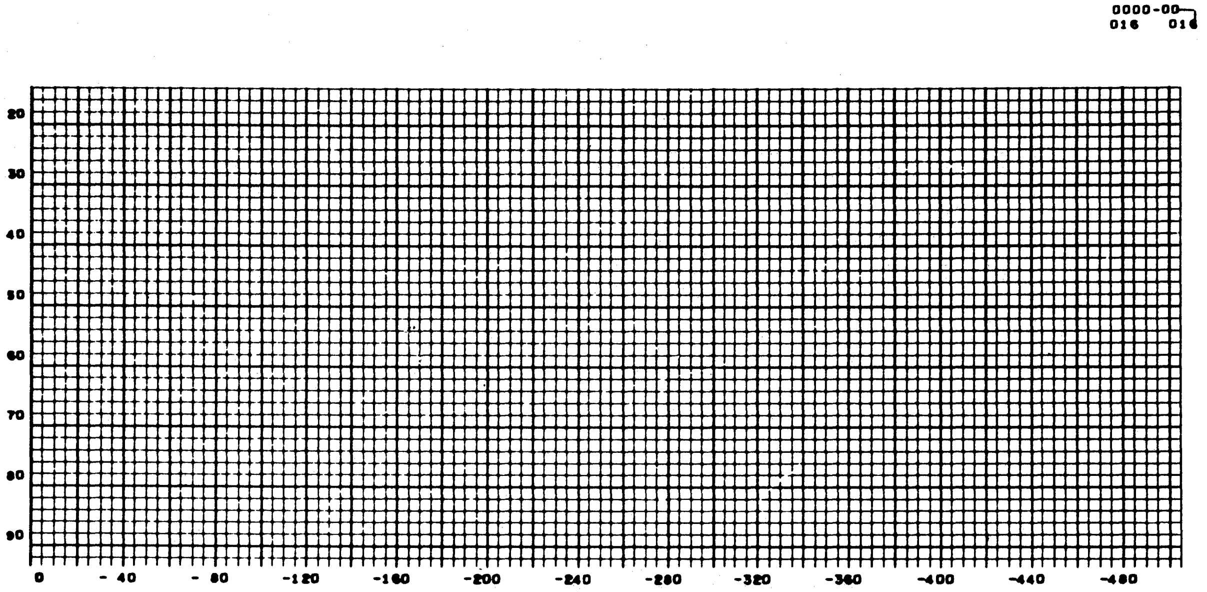
```
CALL GRID1V (1, 50.0, 100.0, 50.0, 100.0, 50.0, 50.0, 1, 1, 2, 2, 3, 6 )
CALL GRID1V (2, 50.0, 100.0, 50.0, 100.0, 50.0, 50.0,-1,-1, 2, 2, 3, 6 )
```

Figure 5-6



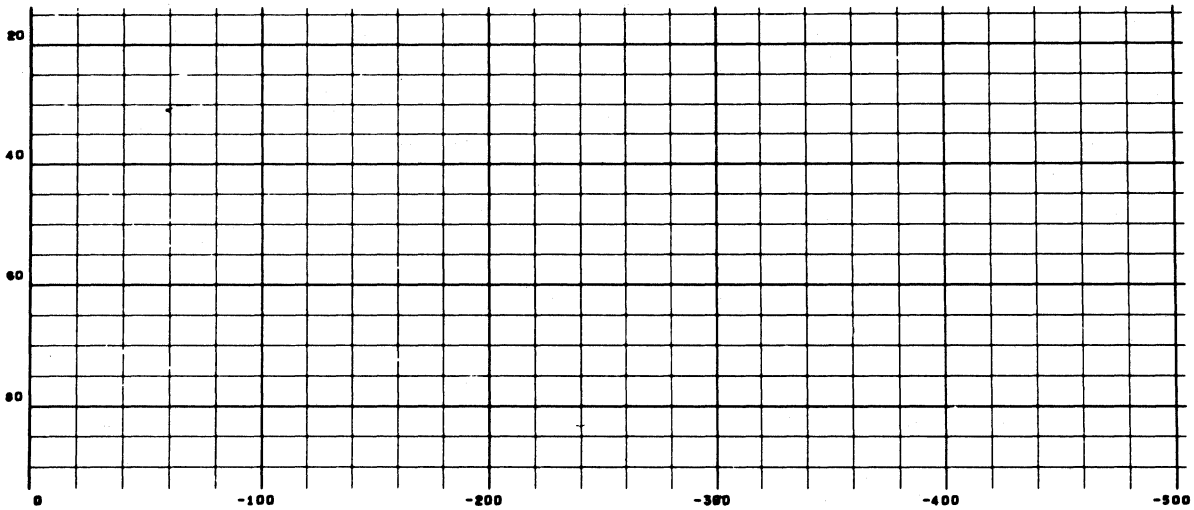
```
CALL DXDYV (1, 0.0, 20.0, DX, N, I, NX, 10.0, IERR1 )
CALL DXDYV (2, 0.0, 80.0, DY, M, J, NY, 10.0, IERR2 )
CALL GRID1V (1, 0.0, 20.0, 0.0, 80.0, DX, DY, N, M, I, J, NX, NY )
```

Figure 5-7



```
CALL DXDYV ( 1, 0.0, -505.0, DX, N, I, NX, 8.0, IERR )
CALL DXDYV ( 2, 93.5, 14.0, DY, M, J, NY, 8.0, IERR  )
CALL GRID1V ( 1, 0.0, -505.0, 93.5, 14.0, DX, DY, N, M, I, J, NX, NY )
```

Figure 5-8



```
CALL DXDYV ( 1, 0.0, -505.0, DX, N, I, NX, 20.0, IERR )
CALL DXDYV ( 2, 93.5, 14.0, DY, M, J, NY, 20.0, IERR )
CALL GRID1V ( 2, 0.0, -505.0, 93.5, 14.0, DX, DY, N, M, I, J, NX, NY )
```

Figure 5-9

COMPUTATION OF GRID1V ARGUMENTS: DXDYV

It frequently happens that the programmer does not have sufficient advance information about the range of data his program will encounter to be able to assign practical values to all arguments of GRID1V. In this case, a series of FORTRAN statements can be used to determine the upper and lower X and Y bounds. For example, the values of XL and XR for a block of data, X, can be computed as follows:

```
   XL = X(1)
   XR = X(1)
   DO 10 J = 2, NPTS        where NPTS is the number
                            of points in the X block of
                            data
   XL = MIN1F(XL, X(J))
10 XR = MAX1F(XR, X(J))
```

A similar group of statements can be used to compute YB and YT for the Y block of data.

Once XL and XR (or YB and YT) are known, the routine DXDYV is available to compute arguments for line spacing, line emphasizing, and line labeling. Two call statements are available, one for the X direction and one for the Y direction. They are:

```
CALL DXDYV(1, XL, XR, DX, N, I, NX, DC, IERR)

CALL DXDYV(2, YB, YT, DY, M, J, NY, DC, IERR)
```

On each entry to DXDYV, four arguments are furnished by the programmer:

> The first argument is a 1 or a 2, to indicate whether DXDYV is being applied in the X direction or in the Y direction.
>
> XL and XR (or YB and YT) are defined as in the summary of GRID1V arguments.
>
> DC represents a floating point quantity which limits the density of the grid. The grid lines drawn by GRID1V, using arguments furnished by DXDYV, will be no closer than DC raster positions. DC should never have a value less than 3.0; values of 8.0 to 20.0 are recommended.

The remainder of the arguments are variables to which DXDYV will assign values. Any value previously assigned these variables will be destroyed during execution of the subroutine. NEVER USE CONSTANTS FOR THESE ARGUMENTS.

IERR is an error indicator. It is set to zero if a reasonable grid can be drawn, and to one if the parameters given would result in an impossible grid. After execution of DXDYV, IERR should always be tested before proceeding to draw the graph.

In using DXDYV, it should be noted that no provision has been made for generating labels in scientific notation. If this is desired, it is necessary to assure that there is sufficient space for the longer labels and also to change the sign of NX and NY to be negative.

GRID1V CONTROLS

Certain features of the basic linear GRID1V can be altered by subprograms that control its internal operation. The subprograms can be classified as "set" and "retrieve" routines since they permit information to be set by the programmer and retrieved during execution of GRID1V.

The routines that furnish values different from those normally employed by GRID1V are:

SETMIV, which allows the programmer to make nonstandard margin assignments. The companion routine called by GRID1V to retrieve margin values is SETMOV.

SETCIV, which makes it possible to provide extra space for grid line labels. The companion routine is SETCOV.

Routines that furnish indicators recognized by GRID1V as signals to execute alternate branches are:

HOLDIV, which assists in holding margins from graph to graph. HOLDOV is called to retrieve the indicators.

SMXYV, which enables the programmer to select a non-linear mode of operation. The companion routine is MSXYV. These two routines are described under "Log and Semilog Plotting."

Grid Margin Variation: SETMIV, SETMOV

As discussed in an earlier section, GRID1V normally reserves a strip, 24 raster counts in width, at the top, left, and bottom of the grid, for the display of titles. For the many applications which require special margin widths, the subprogram SETMIV can be called to change the basic specifications.

One obvious application of SETMIV is to provide margin space for multiple lines of printed titles and headings. In addition, and perhaps even more important, SETMIV makes it possible to display more than one graph on a frame, or to display a graph with its accompanying text.

The standard GRID1V margin specifications can be altered by the statement:

CALL SETMIV (MTL, MTR, MTB, MTT)

Each argument is an integer which specifies, in raster counts, the width of one area to be reserved for a margin.

MTL	Width of area for left margin.
MTR	Width of area for right margin.
MTB	Width of area for bottom margin.
MTT	Width of area for top margin.

GRID1V does not necessarily use these exact values for the upper and lower limits of X and Y. It guarantees that the reserved space will not be overlapped, assigning additional space if required for label margins. After the total margin space has been reserved, the remaining area will be used for the grid.

If SETMIV is never called, GRID1V will use the values 24, 0, 24, 24 as MTL, MTR, MTB, and MTT, respectively. To return to a standard grid after the margins have been altered, restore the standard margin values by

CALL SETMIV (24, 0, 24, 24)

The current values of MTL, MTR, MTB, and MTT can be retrieved by using the statement

CALL SETMOV (MTLL, MTRL, MTBL, MTTL)

where the arguments are variables (never constants) to which SETMOV is to assign the current margin values. SETMOV was designed for use by GRID1V to retrieve current margin values; the programmer will rarely have reason to call it.

Examples. Figure 5-10 shows three grids with the SETMIV and GRID1V call statements used to produce them. The grid at the bottom was the first one displayed; a 1 was used as the first argument of the first GRID1V statement executed, in order to change the film frame. The other two GRID1V statements include a 2 as the first argument, to inhibit the film advance.

Note particularly the variation in the raster locations assigned by GRID1V to XL in each of the grids. This effect is caused primarily by the differences in the specification of NY (the last argument), which gives the number of characters to be displayed in the labels of horizontal lines. In each case, NY has been assigned a value just large enough to satisfy the needs of the grid. For the bottom grid, NY = 1; for the middle grid, NY = 3; and for the top grid, NY = 5. Since margin space was reserved for labels of different lengths, the positions of the left limits, and of the corresponding values of X, vary noticeably. Such a nonalignment is often of no importance, but if it does matter, the programmer may have to make special provisions to force alignment.

PROVIDING FOR SPECIAL LABEL CHARACTERS: SETCIV, SETCOV

GRID1V computes the starting location of each label, taking into consideration the size of the characters used. If the labels are to be placed outside the grid, GRID1V assigns space for them, again taking the character size into consideration. Normally, labeling is done in CHARACTRON characters (via LABLV). If the programmer substitutes a non-system labeling routine for LABLV, it may be necessary to furnish adjusted character dimensions to GRID1V.

To state the dimensions of nonstandard label characters, use

CALL SETCIV (IW, IH)

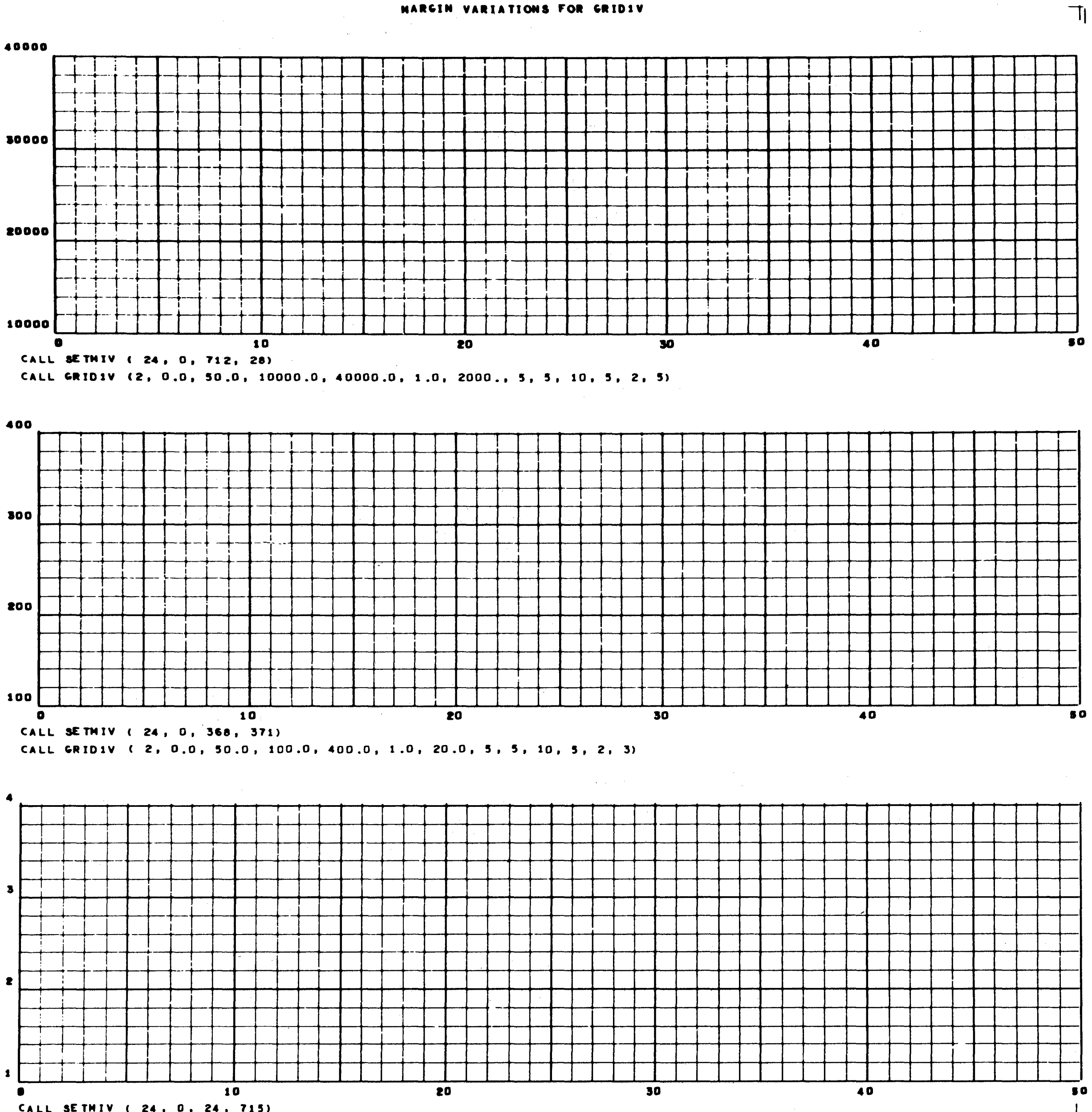
IW — An integer which specifies, in raster counts, the allowance needed for the width of each label character.

IH — An integer which specifies, in raster counts, the allowance for the height of a label character.

If SETCIV is never called, the indicator table contains IW = 8 and IH = 10. Obviously, if it is called, the arguments must be compatible with the size of the characters employed by the LABLV subprogram used.

GRID1V retrieves the values of the indicators by using:

CALL SETCOV (IWL, IHL)



```
CALL SETMIV ( 24, 0, 712, 28)
CALL GRID1V (2, 0.0, 50.0, 10000.0, 40000.0, 1.0, 2000., 5, 5, 10, 5, 2, 5)
```

```
CALL SETMIV ( 24, 0, 368, 371)
CALL GRID1V ( 2, 0.0, 50.0, 100.0, 400.0, 1.0, 20.0, 5, 5, 10, 5, 2, 3)
```

```
CALL SETMIV ( 24, 0, 24, 715)
CALL GRID1V ( 1, 0.0, 50.0, 1.0, 4.0, 1.0, 0.2, 5, 5, 10, 5, 2, 1)
```

Figure 5-10

The width will be retrieved from the table and stored in the fixed point variable location IWL, and the height will be similarly stored in IHL. (The arguments must not be constants.) Note that GRID1V uses this information to control the space that will be reserved for labels; it does not control the size of the label characters themselves in any way.

HOLDING MARGINS FROM GRAPH TO GRAPH: HOLDIV, HOLDOV

For a large graph, it may be necessary for the programmer to display segments of the graph in separate frames, and join the segments "tile fashion" to form the complete plot. If the graphs are to have the same scale, certain equalities should exist.

1. The range of X in each segment should be equal, and the range of Y should be equal. In other words, the quantity (XR - XL) should be the same in each segment, and, similarly, the quantity (YT - YB).
2. The dimensions of the scaled area should be the same from segment to segment; that is, the dimensions of the space between margins should be equal.

The programmer can easily provide for equality in the ranges of X and Y, but he cannot so readily ensure equality in the scaled areas. Since GRID1V computes label margins (and, therefore, total margins) to suit the needs of each graph, the dimensions of the scaled area may vary.

One method that will usually give equality of scaled areas is to specify the option that forces labels to be placed outside the grid, and to always request the same number of label characters (NX, NY) for each segment. If this is not practical, a "holding" feature is provided.

GRID1V can be instructed to hold the label margin spaces used for the preceding grid and use them in computing total margins for the next grid. The statement to be used is

```
CALL HOLDIV (NH)
```

If NH $\neq$ 0, the label margins from the preceding grid will be used again. If NH = 0 (as is the case if HOLDIV is never called), label margins will be computed in the normal manner.

The status of this indicator is tested in GRID1V by using

CALL HOLDOV (NHL)

The value of the indicator will be retrieved and stored in the location named in the argument.

The "hold" may be released by executing the statement

CALL HOLDIV (0)

Figure 5-11 is similar to Figure 5-10 except that NY = 5 on all three grids, permitting the left limits to be in line. If XL, NX, and NY are equal from grid to grid, the desired alignment will usually be achieved.

Figure 5-12 shows four graphs on a single frame. The SETMIV statements used to produce the margins for each grid are shown. The programmer must remember to set the first argument of the GRID1V statement to 2 so as not to advance the film.

Figures 5-13, 5-14, and 5-15 show additional examples of special effects which can be obtained with GRID1V when the routine HOLDIV is used to retain grid margins from one grid to another. The labeling is self explanatory.

OPERATIONAL DETAILS OF GRID1V

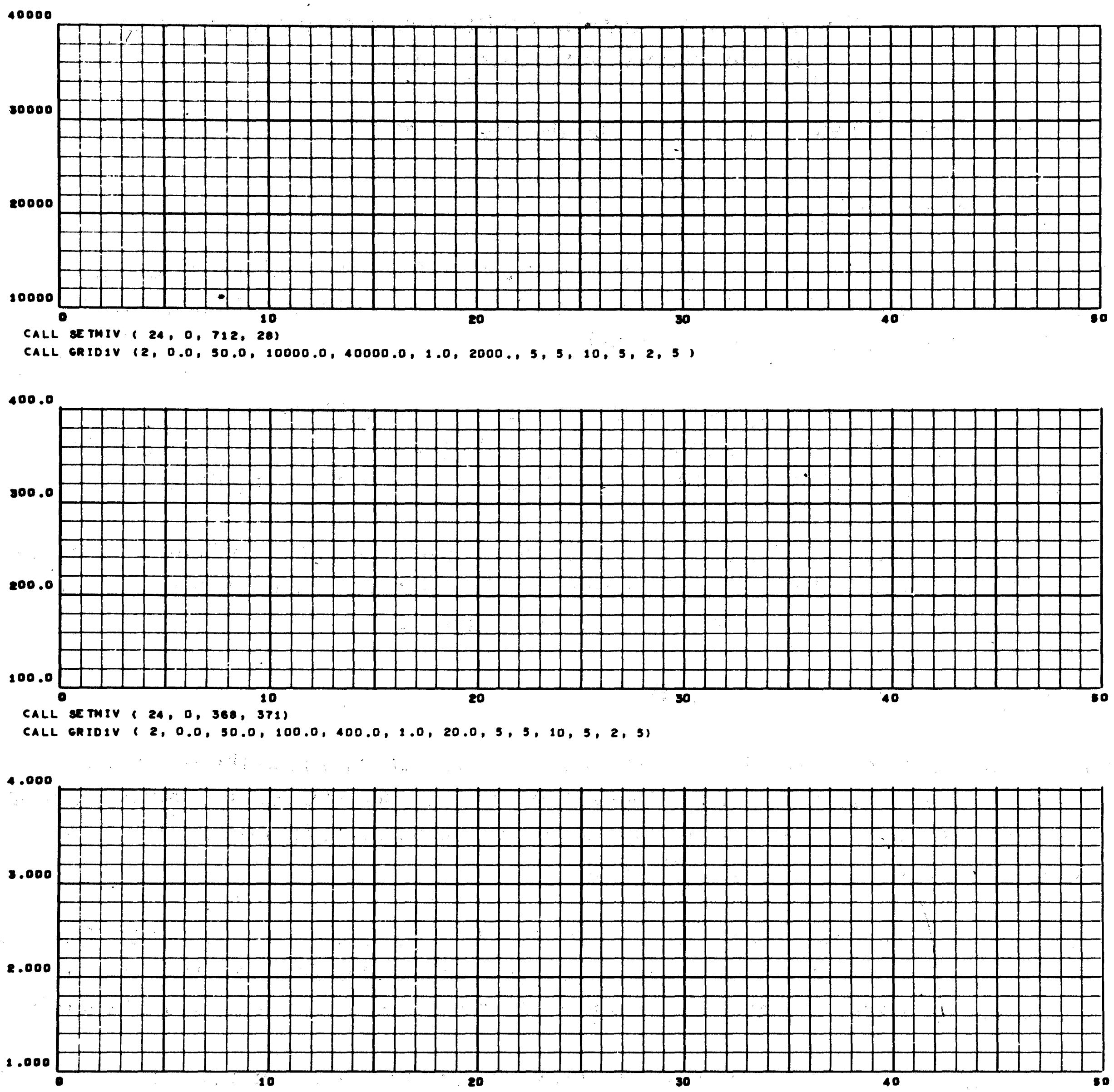
GRID1V is, in many respects, an executive routine. It examines the information furnished by the argument list and by certain external subprograms, makes decisions based on this information, and then calls other subprograms to advance the film, compute scale factors, generate the grid, etc.

Initially, GRID1V uses STOPTV to ensure that the typewriter mode is "off," BRITEV to ensure that the bright intensity mode is "on," and FRAMEV to advance the film and affix the job number and frame count (unless an option is used that inhibits this).

GRID1V then checks certain internal locations to obtain basic information, by calling the following subprograms:

SETMOV — Retrieves margin assignments. The programmer may have changed the standard specifications by a CALL SETMIV.

SETCOV — Retrieves character size specifications that programmer may have changed by a CALL SETCIV.



```
CALL SETMIV ( 24, 0, 712, 28)
CALL GRIDIV (2, 0.0, 50.0, 10000.0, 40000.0, 1.0, 2000., 5, 5, 10, 5, 2, 5 )
```

```
CALL SETMIV ( 24, 0, 368, 371)
CALL GRIDIV ( 2, 0.0, 50.0, 100.0, 400.0, 1.0, 20.0, 5, 5, 10, 5, 2, 5)
```

```
CALL SETMIV ( 24, 0, 24, 715)
CALL GRIDIV ( 1, 0.0, 50.0, 1.0, 4.0, 1.0, 0.2, 5, 5, 10, 5, 2, 5)
```

Figure 5-11

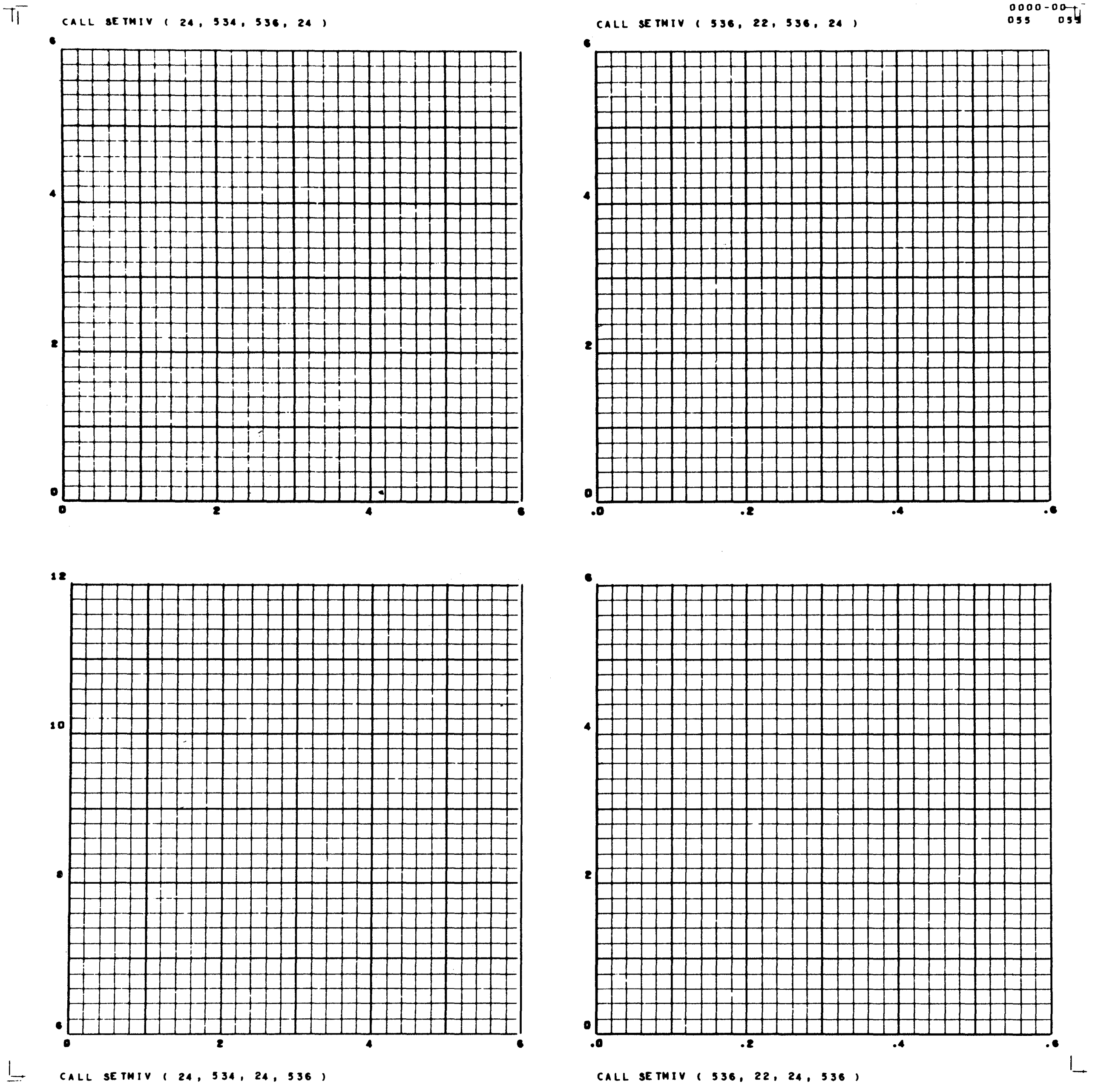
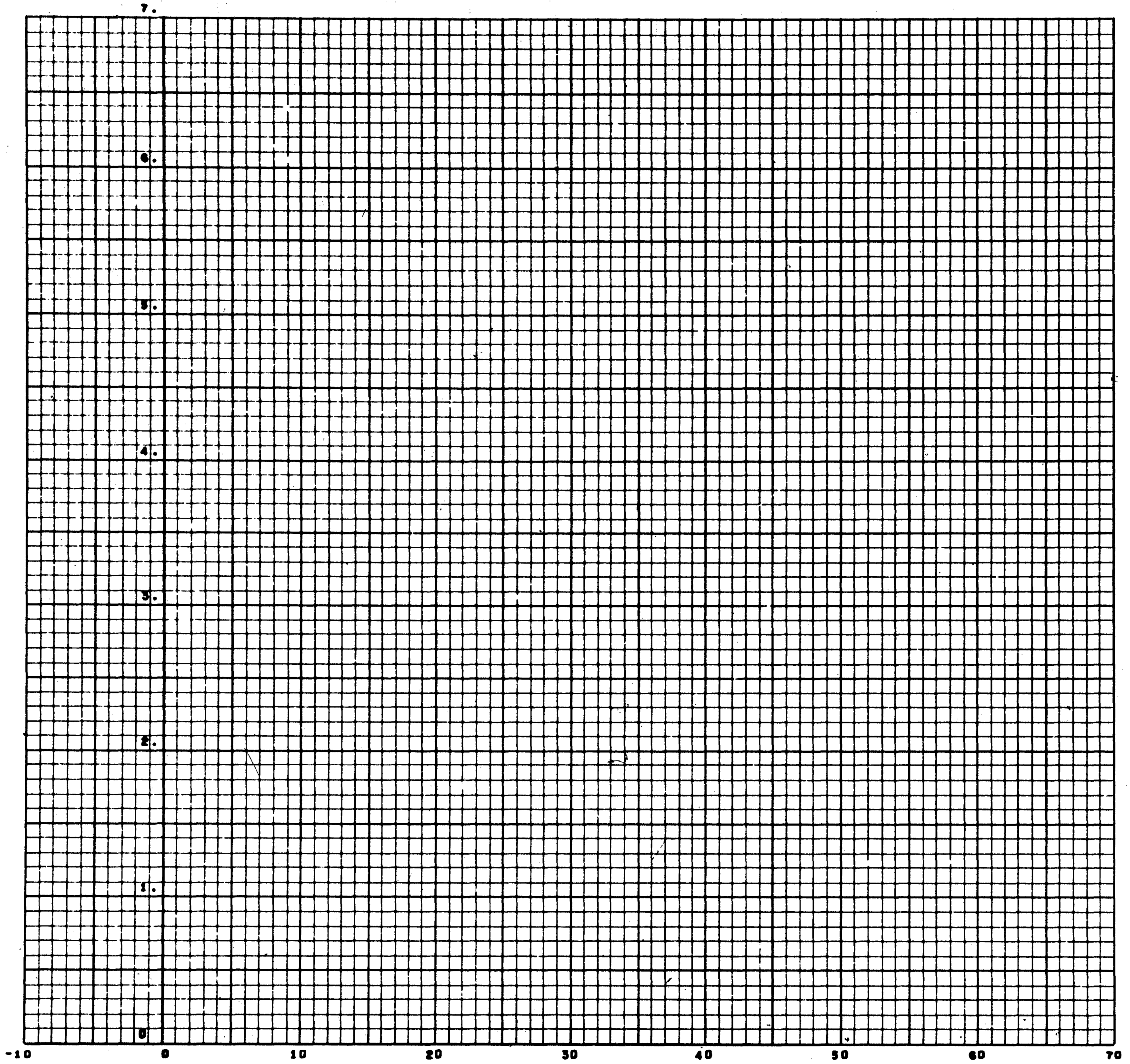

0000-00
055 055
CALL SETMIV (24, 534, 536, 24)
6
4
2
0
0
2
4
6
CALL SETMIV (536, 22, 536, 24)
6
4
2
0
.0
.2
.4
.6
12
10
8
6
0
2
4
6
CALL SETMIV (24, 534, 24, 536)
6
4
2
0
.0
.2
.4
.6
CALL SETMIV (536, 22, 24, 536)

Figure 5-12

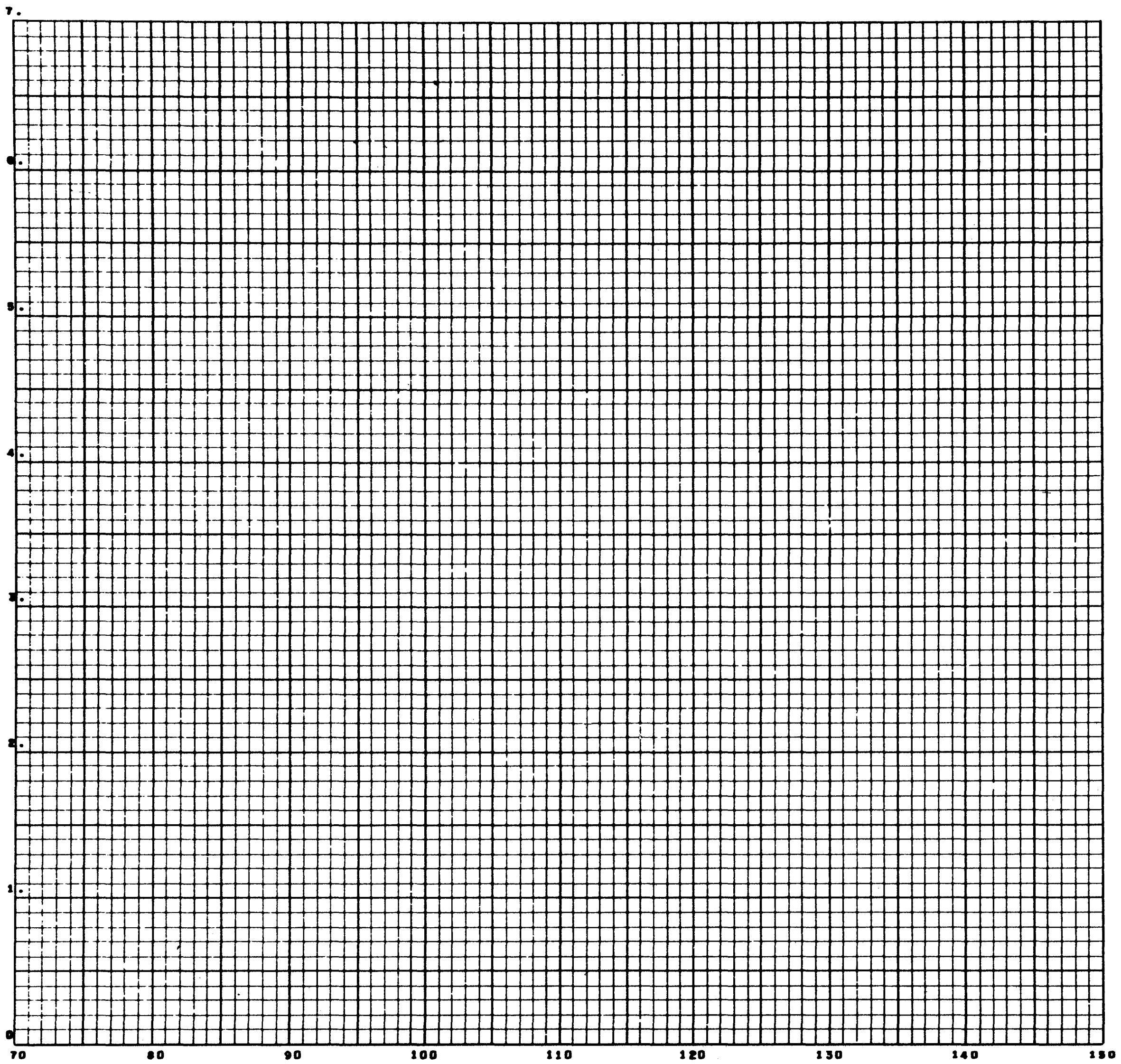
SPECIAL GRIDIV EXAMPLES

HOLDIV AS AN AID IN MAINTAINING THE SAME SCALE FROM ONE GRID TO ANOTHER



```
CALL GRIDIV ( 1, -10.0, 70.0, 0.0, 7.0, 1.0, 0.1, 5, 5, 10, 10, 2, 2 )
CALL HOLDIV(1)
                  THE HOLD STATEMENT WILL CAUSE LABEL MARGINS TO BE HELD FOR SUBSEQUENT GRIDS
                  UNTIL CALL HOLDIV(0) RELEASES THE HOLD
```

Figure 5-13



```
CALL GRID1V ( 1, 70.0, 150.0, 0.0, 7.0, 1.0, 0.1, 5, 5, 10, 10, 3, 2 )
NOTE...NOT ONLY HAS HOLDIV BEEN USED BUT ALSO XL,XR,YB,YT HAVE BEEN SPECIFIED SUCH THAT
XR-XL AND YT-YB OF ONE GRID EQUAL THEIR COUNTERPART IN THE OTHER GRID
```

Figure 5-14

SPECIAL GRIDIV EXAMPLES

0000-00
063 063

USE OF HOLDIV WITHIN A FRAME

```
CALL SETMIV ( 24,599, 50, 573 )
CALL GRIDIV ( 1, -10.0, 90.0, 0.0, 9.0, 5.0, 1.0, 4, 2, -4, -2, 2, 1 )
CALL HOLDIV (1)
CALL SETMIV ( 524, 99, 50, 573 )
CALL GRIDIV ( 2, 90.0, 190.0, 0.0, 9.0, 5.0, 1.0, 4, 2, -4, -2, 3,1 )
```

```
     NOTE... NOT ONLY MUST XR-XL AND YB-YT OF ONE GRID EQUAL THE CORRESPONDING QUANTITY IN THE
             OTHER GRID, BUT ALSO THE ARGUMENTS OF SETMIV MUST HAVE A SPECIFIC
             RELATIONSHIP........SAY THE ARGUMENT LIST IS (MTL,MTR,MTB,MTT) ,THEN THE
             VALUES 1023-(MTL+MTR) AND 1023-(MTB+MTT) OF ONE GRID MUST EQUAL THE
             CORRESPONDING QUANTITY IN THE OTHER GRID
```

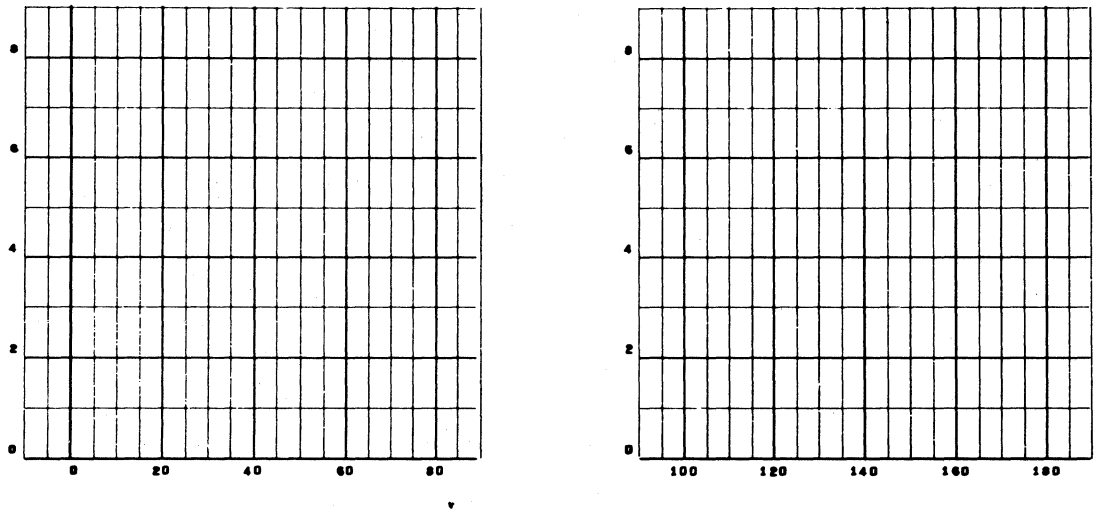


Figure 5-15

MSXYV Determines linear or nonlinear mode. Indicators may have been set to nonlinear by a CALL SMXYV.

HOLDOV Determines whether margin specifications were held over from preceding graph. Indicator may have been specified by a CALL HOLDIV.

Computations are made to determine the raster positions to be used as grid boundaries. ERRLNV and/or ERRNLV are called to check the boundaries and the data limits to see if a grid can be produced from this information. Finally, XSCALV and YSCALV are called to compute scale factors, and LINRV (in the linear mode) and/or NONLNV (in the logarithmic mode) are called to generate the grid.

DETERMINING GRID BOUNDARY POSITIONS

GRID1V sets aside the margin space specified by a prior call to SETMIV (or the standard margin space if SETMIV has not been called). With the exception of the small area in the upper right corner used for frame identification, GRID1V does no writing in these basic margins.

Then GRID1V tests to see if labels are to be placed (or may extend) outside the grid area. Space required for such labels is computed, taking into consideration the type of label (fixed point or scientific), the number of label characters specified in GRID1V arguments plus space for a sign, and the height and width of the label characters.

If any space is needed for labels at the left, right, bottom, or top of the grid, it is added to the basic margins specified by SETMIV table, to produce the total margins. (HOLDIV can alter this procedure by causing label spaces "held" from a previous grid to be added when computing the total margins).

If the option for a square grid is specified, the right, or top, total margin will be adjusted so the remaining area will be square.

If the total margins and the grid limits (XL, XR, YB, and YT) meet certain error tests, they are used as arguments of XSCALV and YSCALV in computing the scale factors required for generating the grid and plotting on it.

Since so many items influence the margin assignments, the grid boundaries will rarely fall at the exact raster positions that the programmer might have estimated. If the precise raster positions of the boundaries are required in a program, they should be derived by converting the limits into raster counts after GRID1V has been called. For example:

IXL = NXV (XL)

IXR = NXV (XR)   (The right total margin is 1023 - IXR)

IYB = NYV (YB)

IYT = NYV (YT)   (The top total margin is 1023 - IYT)

GRID1V ERROR PROCEDURE: ERMRKV

Even if bad input data is used, GRID1V will attempt to produce a grid. The philosophy is that some useful information may be revealed, even if the grid is inaccurate.

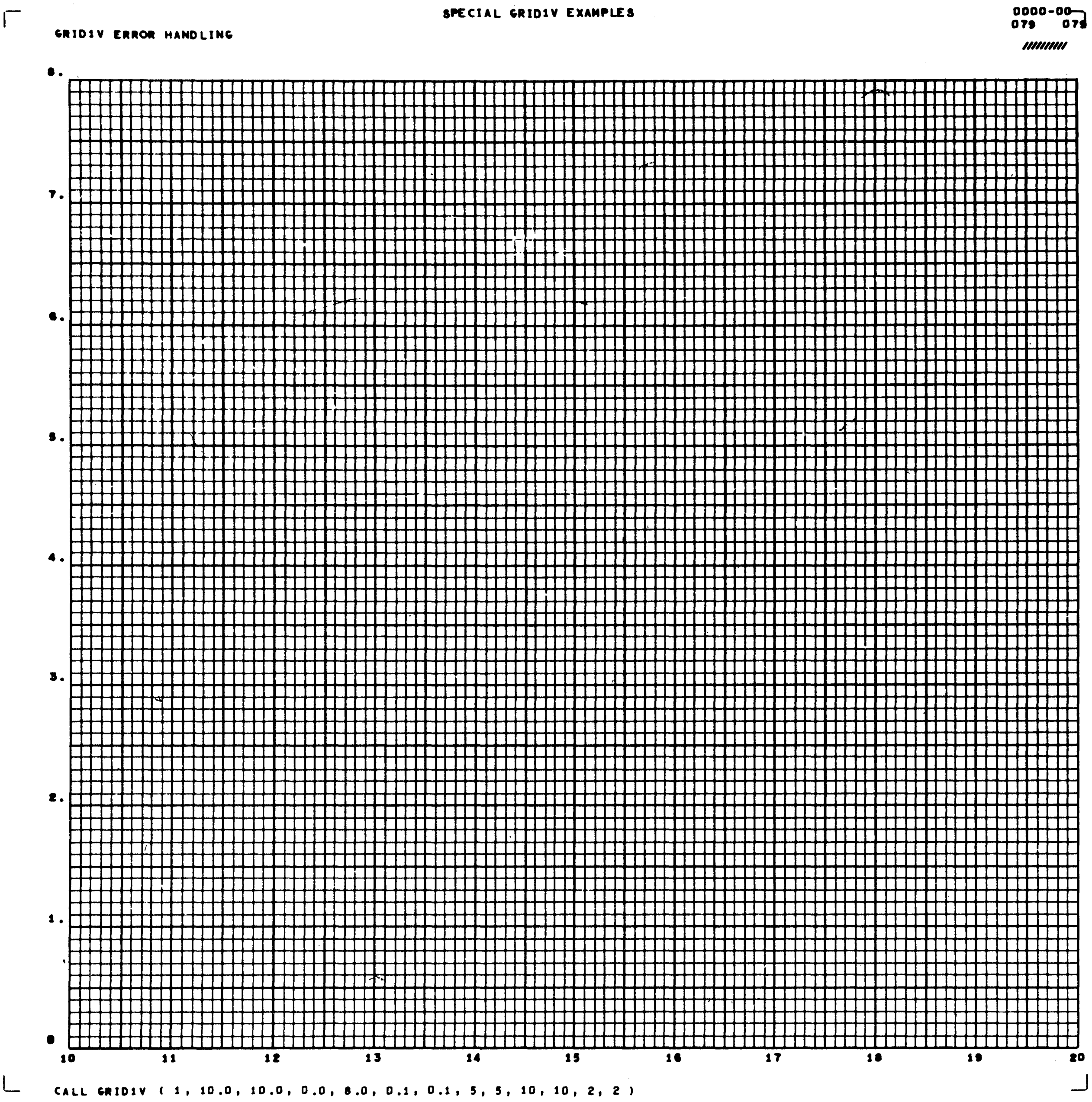
The grid limits and total margins are tested by lower-level subprograms, ERRLNV (for linear mode) and/or ERRNLV (for log mode). If these tests show that a grid cannot be produced from the given information, some data values are manufactured so that the program can continue. The manufactured quantities are used only internally; data values in the main program will not be affected. (Since ERRLNV and ERRNLV are meant to be used only by GRID1V, the call statements are not given.)

When artificial quantities are used, an error mark (//////) is placed in the upper right corner of the frame by ERMRKV. Although ERMRKV was designed for use by GRID1V, the programmer can place this mark on the frame if he uses the statement

```
CALL ERMRKV
```

An error mark from GRID1V may indicate that one or more of the following errors has been found.

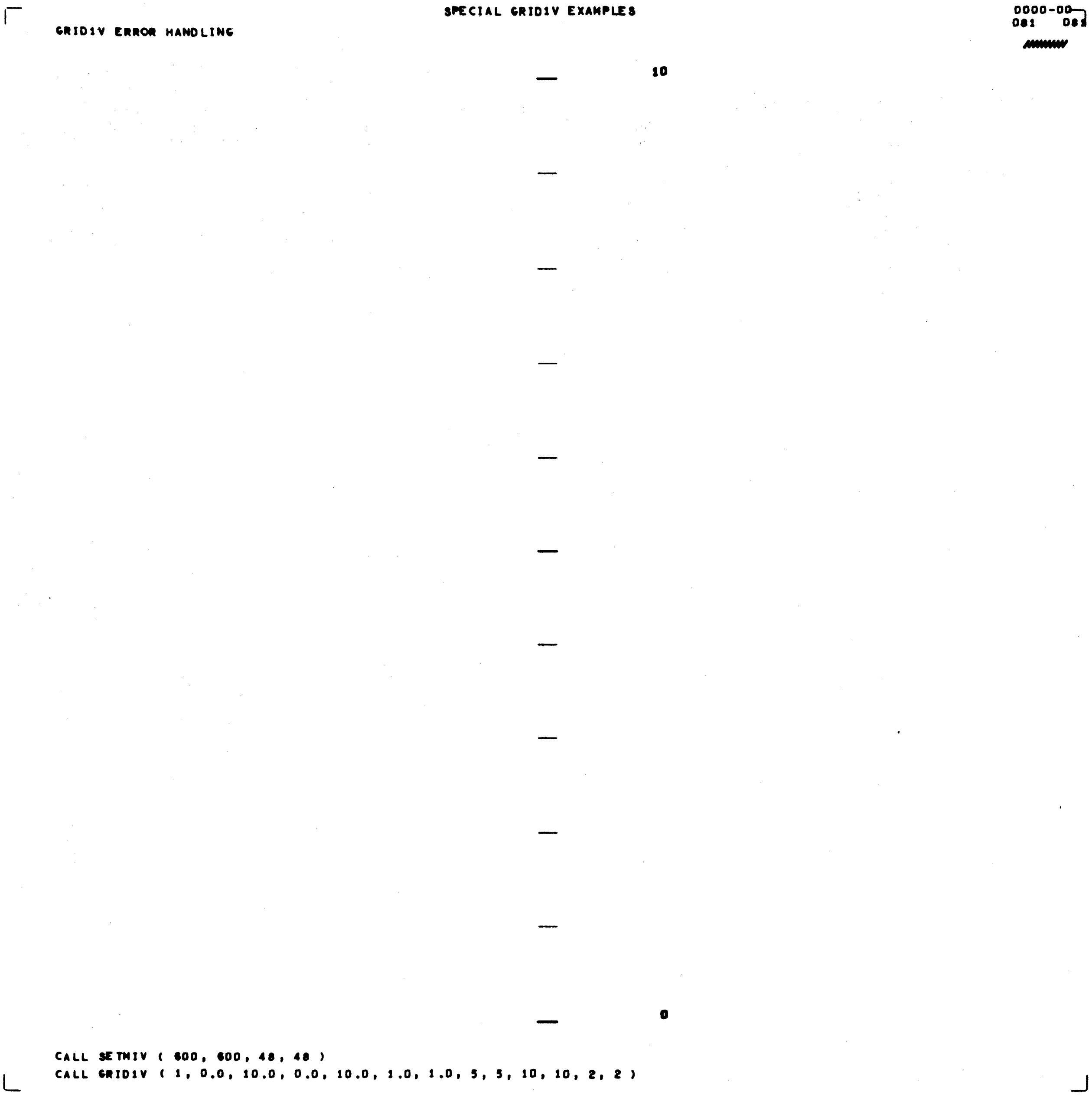
1. Equal values have been specified for grid limits; that is, XL = XR and/or YB = YT. See Figure 5-16.
2. The specified margins are too wide. In other words, MR + ML - 1023, and/or MB + MT - 1023. See Figure 5-17.
3. In a linear grid, the specified values of DX and/or DY would result in grid lines spaced closer than 3 raster counts.
4. In a log grid the value of one or more of the limits is either zero or negative.
5. In the log mode, there are more than 10 cycles in either the X or Y direction.


SPECIAL GRID1V EXAMPLES

GRID1V ERROR HANDLING

```
CALL GRID1V ( 1, 10.0, 10.0, 0.0, 8.0, 0.1, 0.1, 5, 5, 10, 10, 2, 2 )
```

Figure 5-16



SPECIAL GRID1V EXAMPLES

GRID1V ERROR HANDLING

```
CALL SETMIV ( 600, 600, 48, 48 )
CALL GRID1V ( 1, 0.0, 10.0, 0.0, 10.0, 1.0, 1.0, 5, 5, 10, 10, 2, 2 )
```

Figure 5-17

Examples of GRID1V Characteristics

Certain special characteristics of the grid generated by GRID1V are shown in the following examples.

Example 1. During construction of a linear grid, vertical lines are displayed at intervals "stepped off" from X = 0 in DX increments until the maximum limit of the grid is exceeded. Then they are "stepped off" from X = 0 in the negative direction until the minimum limit of the grid is passed. Similarly, horizontal lines are "stepped off" from Y = 0 in DY increments. Because of this method of construction, lines will be generated at the grid limits only if XL and XR are integer multiples of DX, and if YB and YT are integer multiples of DY. Figures 5-18 and 5-19 illustrate the effect of changing DX and DY. Figures 5-20 and 5-21 illustrate a similar effect in the negative range of values.

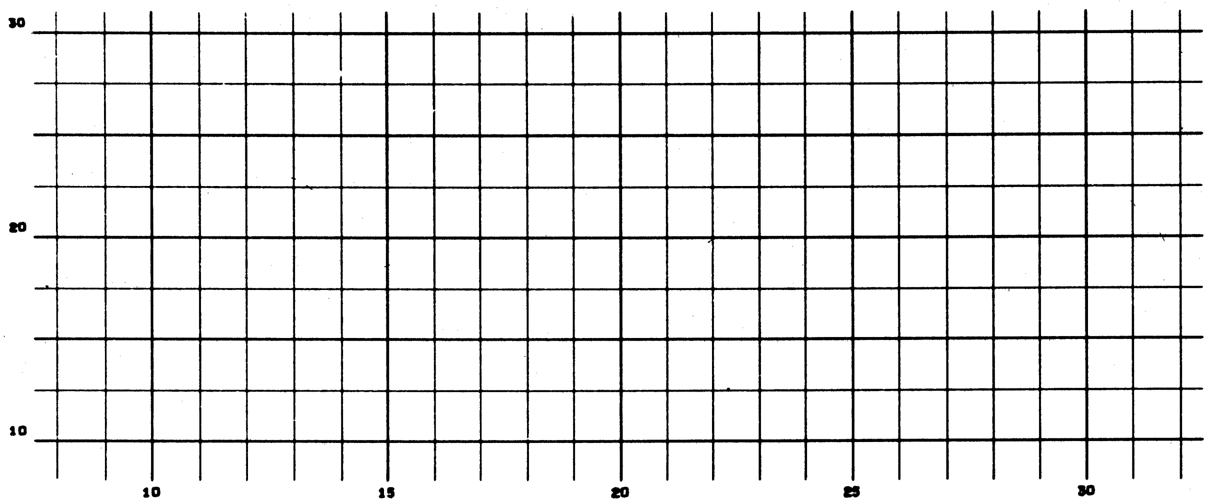
Example 2. Vertical lines to be emphasized are "stepped off" from X = 0 by increments of N·DX, and vertical lines to be labeled are "stepped off" by increments of I·DX. Similarly, horizontal lines to be emphasized are "stepped off" from Y = 0 by increments of M·DY, and those to be labeled by increments of J·DY. This procedure means that labels and/or emphasized lines will not necessarily occur at the grid limits. Lines at XL and XR will be emphasized only if XL and XR are integer multiples of N·DX, and labeled only if XL and XR are integer multiples of I·DX. In the same way, lines at YB and YT are integer multiples of M·DY, and labeled only if they are integer multiples of J·DY.

This method of determining the positions of line labels is responsible for the absence of labels on the limit lines in Figures 5-19 and 5-21. Notice in the example that the labels for vertical grid lines are adequate; there is no real need to provide labels on the lines at XL and XR. However, the labels for the horizontal lines in this illustration are not adequate; they demonstrate another factor to be considered in planning for line labels. To be meaningful, labels should appear on at least two lines. In the example, the use of J = 5, instead of J = 10, would cause labeling of the lines of Y = 25.0, 30.0, and 35.0, giving a scale that can be read easily.

SPECIAL GRIDIV EXAMPLES

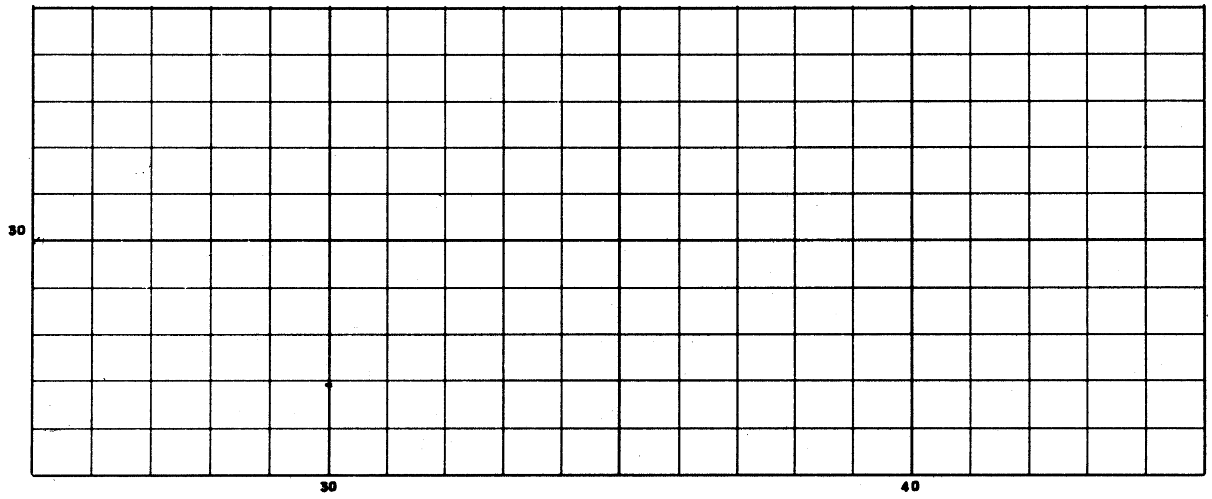
0000-00 083 083

EFFECTS OF STEPPING OFF DELTA INCREMENTS FROM ZERO



```
CALL GRIDIV ( 1, 7.5, 32.5, 8.0, 31.0, 1.0, 2.5, 5, 2, 5, 4, 2, 2 )
```

Figure 5-18



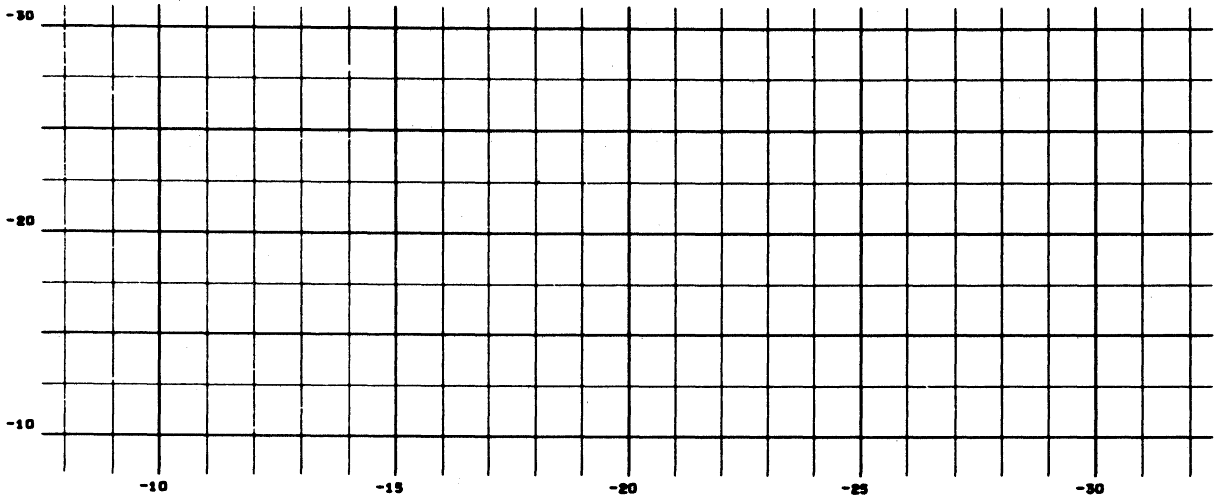
```
CALL GRIDIV ( 2, 25.0, 45.0, 25.0, 35.0, 1.0, 1.0, 5, 5, 10, 10, 2, 2 )
```

Figure 5-19

SPECIAL GRID1V EXAMPLES

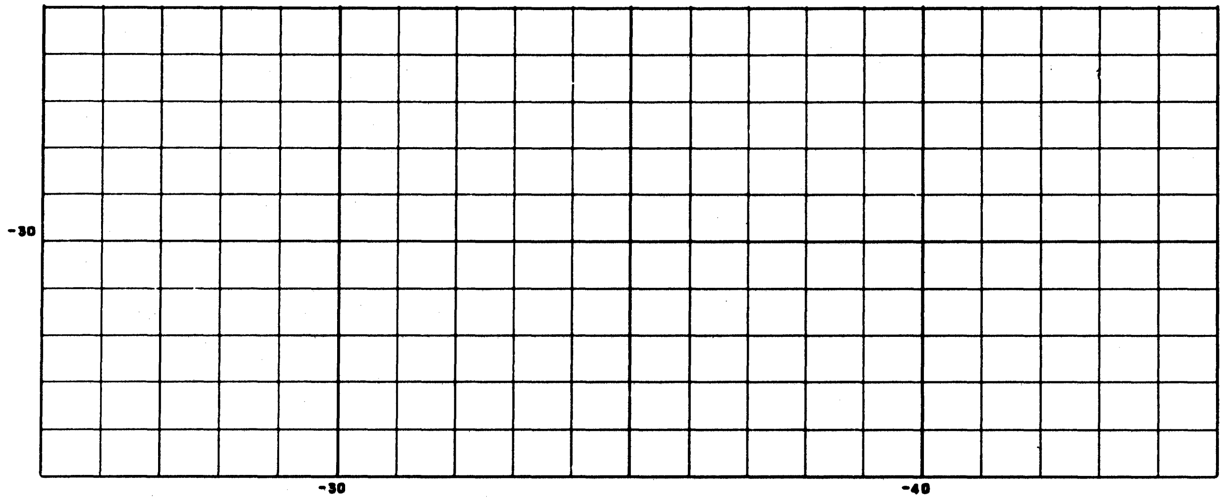
0000-00
085 085

THE NEGATIVE RANGE OF VALUES IS HANDLED IN A SIMILAR WAY



```
CALL GRID1V ( 1, -7.5, -32.5, -8.0, -31.0, 1.0, 2.5, 5, 2, 5, 4, 2, 2 )
```

Figure 5-20



```
CALL GRID1V ( 2, -25.0, -45.0, -25.0, -35.0, 1.0, 1.0, 5, 5, 10, 10, 2, 2 )
```

Figure 5-21

Example 3. Under certain conditions, GRID1V purposely omits labels. Figure 5-22 shows how this can happen. The limits of X used for the grid were -99.9999 and 99.9999, and the limits of Y were 0.0 and 9.9999. During scaling, these limits were rounded, causing them to be treated as -100.0, 100.0 and 0.0, 10.0.

This process caused the decimal scales of the grid limits to be larger than the initial limits, i.e., the values of NX and NY specified by the programmer were not compatible with the new limits. To avoid erroneous labeling, label values will not be displayed if the rounding process causes the decimal scale to be larger than the initial value.

The system does not adequately provide for all problems of this type. Such complications can be avoided if the programmer considers the rounded values of XL and XR, and YB and YT, and when specifying NX and NY.

Example 4. When X = 0 (and/or Y = 0) lies within the grid limits, and I (and/or J) is positive, GRID1V places labels along the X = 0 (or Y = 0) line. If there isn't enough space for the label between X = 0 and the left limit of the grid (or between Y = 0 and the bottom of the grid), GRID1V will provide space outside the grid area for labels, just as if negative values of I (and/or J) had been specified. An illustration is shown in Figure 5-23.

Example 5. The number of label characters, NX (or NY), specified for fixed point labels should satisfy the largest and smallest label values to be displayed. Normally, the quantity should be the sum of: (1) the decimal scale of the largest value of X (or Y), (2) the number of fractional positions required in the smallest value, and (3) one more position to provide for the decimal point, if required. The total quantity may not exceed 6 (or 7 if the decimal point is included).

Figure 5-24 illustrates how trouble can occur when NX and/or NY are not specified properly. Note that the values of NX = 1 and NY = 1 are adequate for the largest values of X and Y, but do not allow for the fractional values required in some of the labels. In this example, 3 should have been used for the values of NX and NY, in the top graph.

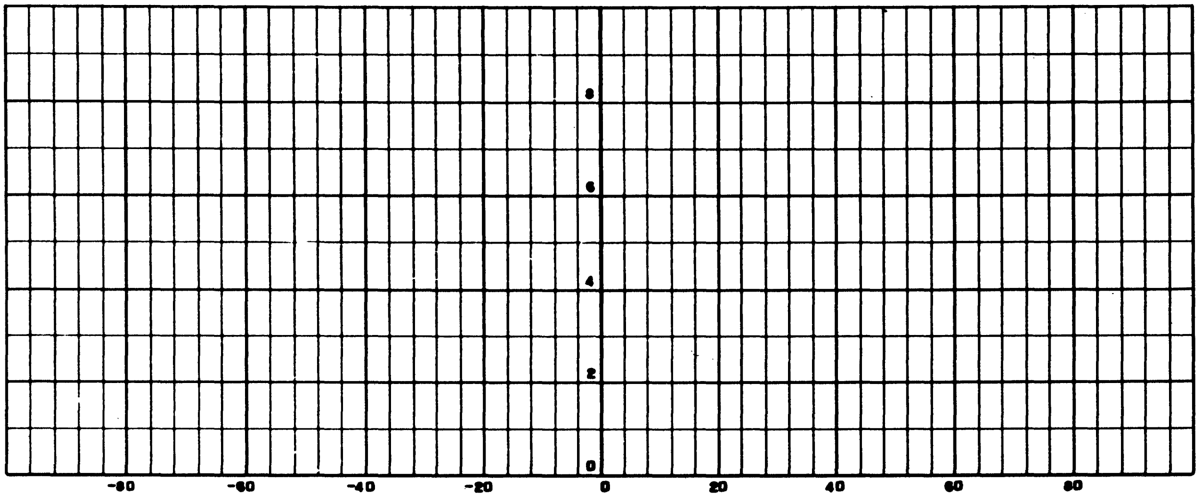
For some applications, it may be advantageous to specify values of NX and/or NY less than the decimal scale of the largest label value. In the middle grid of Figure 5-24, the two low-order positions in the labels have been dropped intentionally, by using values of NX and NY that are less than the decimal scales of the largest values of X and Y. In this case, the resulting graph has been effectively rescaled.

This procedure must be used with care. In the bottom grid, Figure 5-24, the values of NX and NY are so small that the increment between labeled lines is not reflected in the labels that are displayed.

SPECIAL GRID1V EXAMPLES

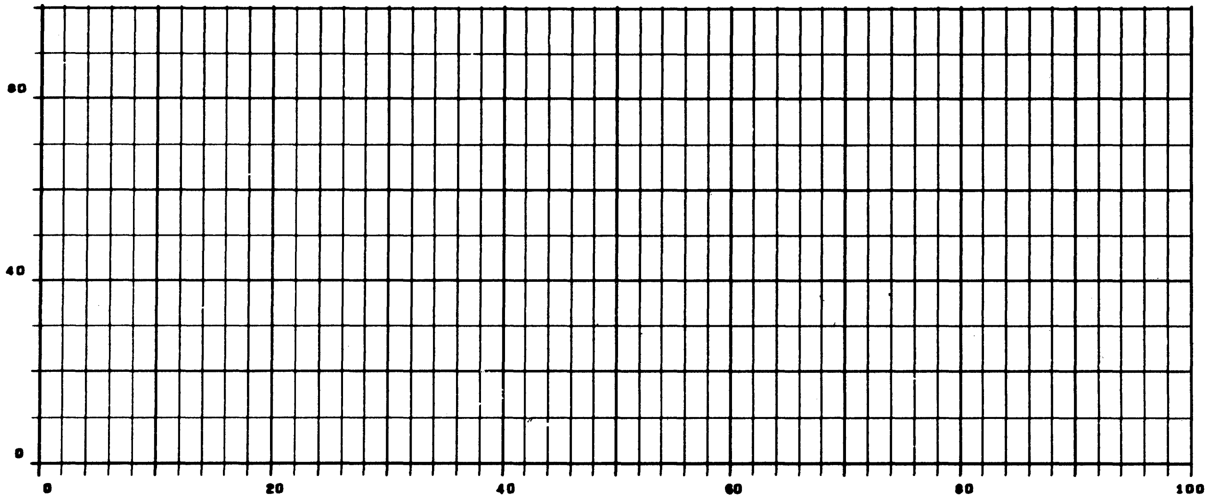
0000-00
087 087

SPECIAL CASES AFFECTING GRID LABELS



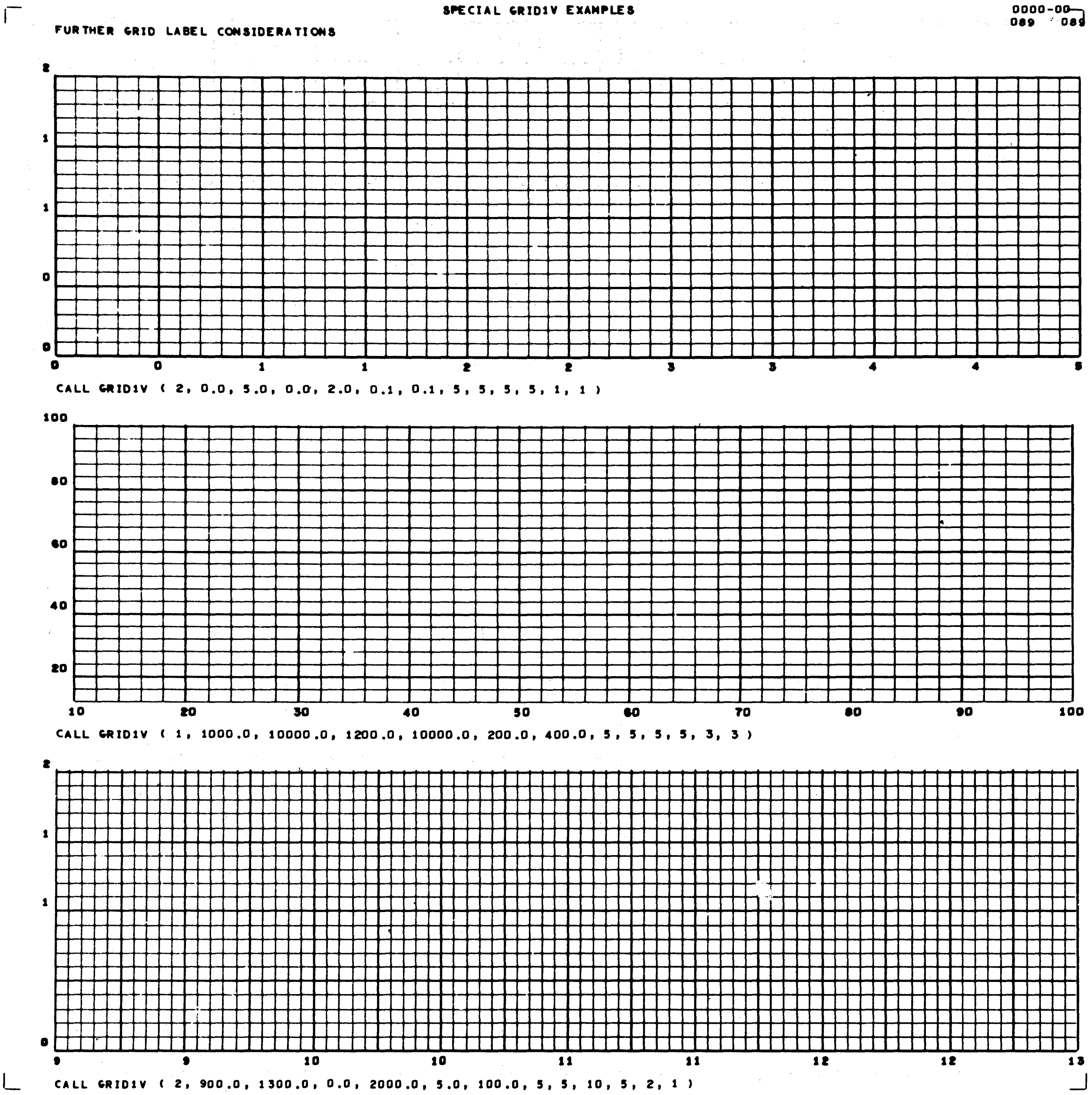
```
CALL GRID1V ( 1, -99.9999 , 99.9999 , 0.0, 9.9999, 4.0, 1.0, 5, 2, 5, 2, 2, 1 )
```

Figure 5-22



```
CALL GRID1V ( 2, -0.7, 100.0, -3.0, 100.0, 2.0, 10.0, 5, 2, 10, 4, 3, 3 )
```

Figure 5-23



```
CALL GRID1V ( 2, 0.0, 5.0, 0.0, 2.0, 0.1, 0.1, 5, 5, 5, 5, 1, 1 )
```

```
CALL GRID1V ( 1, 1000.0, 10000.0, 1200.0, 10000.0, 200.0, 400.0, 5, 5, 5, 5, 3, 3 )
```

```
CALL GRID1V ( 2, 900.0, 1300.0, 0.0, 2000.0, 5.0, 100.0, 5, 5, 10, 5, 2, 1 )
```

Figure 5-24

LOG AND SEMI-LOG GRIDS: SMXYV, MSXYV

The earlier examples used only linear scaling and conversion. The modal subroutine SMXYV can be used to alter the scaling mode such that scaling and conversions will be made in the logarithmic mode.

The call statement for establishing the logarithmic mode is:

```
        CALL SMXYV (MX, MY)
```

where MX and MY are scale mode indicators that designate whether the logarithmic or linear mode is to be used:

If $MX \neq 0$, $MY \neq 0$ Log in X, log in Y

$MX \neq 0$, $MY = 0$ Log in X, linear in Y

$MX = 0$, $MY \neq 0$ Linear in X, log in Y

$MX = 0$, $MY = 0$ Linear in both X and Y
(to restore linear mode)

At the beginning of each job, MX and MY are zero, so that linear scaling and conversion result if SMXYV is never called.

If the programmer wants to generate a log or semi-log plot, he must call SMXYV to set the logarithmic mode before using any S-C 4020 subroutines that involve scaling or conversion.

Once the log-log or semi-log mode has been set by SMXYV and the scale factors have been established, the function statements NXV and NYV can be used to convert data coordinates into raster coordinates, in the same manner as shown for linear scaling.

For chain jobs, note that the values of MX and MY do not carry over from link to link. If a scale mode other than linear-linear is desired, it will be necessary to restate the SMXYV statement in each chain link.

FAP subprograms use two other references in connection with these indicators, causing the names (XXXX) and (YYYY) to appear on the load map.

The contents of the scale mode indicators may be retrieved by using the following statement:

```
CALL MSXYV (MXL, MYL)
```

The indicator for the X scale mode will be stored in MXL, and for the Y scale mode in MYL.

GRID1V, IXV, IYV, NXV, NYV, XSCALV, and YSCALV all use this statement to determine what scale mode has been selected by the programmer.

RESTRICTIONS ON LOGARITHMIC MODE

In general, once SMXYV has been called, the programmer can use any of the routines to generate a logarithmic display. (One exception: DXDYV should not be used to generate arguments for GRID1V in the direction in which logarithmic scaling is being used.)

However, some of the GRID1V arguments are restricted in the logarithmic mode. Following is a list of the 13 arguments, with notations as to the arguments affected if the mode is logarithmic in the direction affected by each.

```
CALL GRID1V (L, XL, XR, YB, YT, DX, DY,
             ±N, ±M, ±I, ±J, ±NX, ±NY)
```

L Controls film advance.

XL, XR Left and right limits of the grid. May not be negative or zero.

YB, YT Bottom and top limits of the grid. May not be negative or zero.

DX, DY Should be set to 1.0. If less, only the cycle lines will be displayed. If greater, no lines will be drawn.

N, M Will be ignored; however, an argument must be present. A negative sign on N or M will force the grid to the square.

I, J Will be ignored; however, an argument must be present.

NX, NY Number of characters to be displayed in the labels.

Generally, line labels will be placed only at the cycle lines. If, however, the grid spans less than one complete cycle, each grid line will be labeled.

No more than 10 log cycles are permitted in the GRID1V system.

Examples. Figure 5-25 illustrates a plot that is logarithmic in both the X and Y directions. Note the arguments for SMXYV and GRID1V in the coding.

Figure 5-26 illustrates a semi-log grid, with plotting of the data used in Figure 5-25.

BUILDING SPECIAL GRIDS: LINRV, NONLNV

Two of the lower-level modules employed by GRID1V are useful as building blocks for building special grids. LINRV may be used to generate only the vertical portion or only the horizontal portion of a linear grid. NONLNV can generate only the vertical or only the horizontal portion of a log grid. See examples in Figure 5-27 and Figure 5-28.

In addition to generating the vertical and horizontal portions of a grid in separate operations, these modules offer other special capabilities:

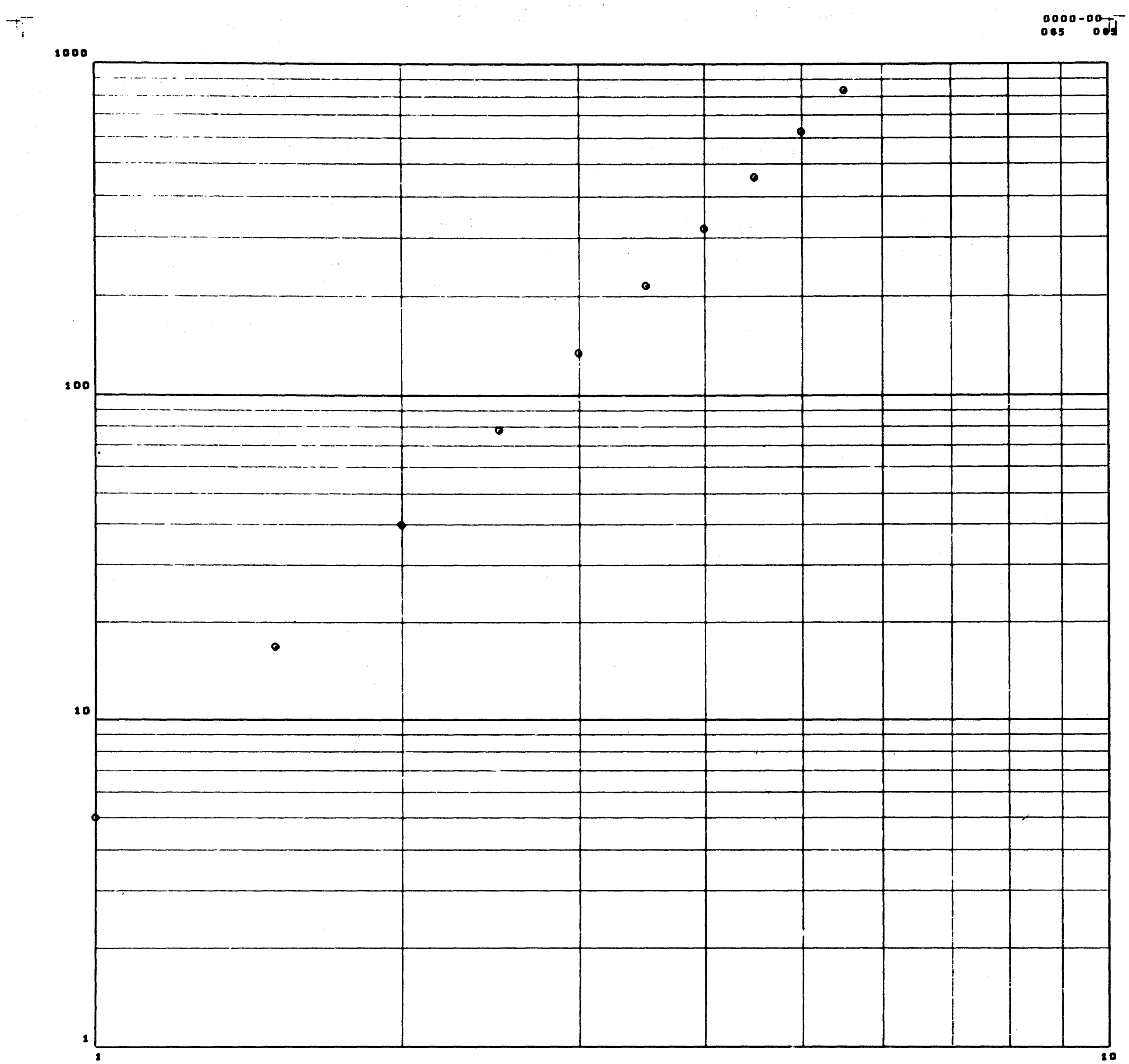
1. The programmer can control the length of the grid lines. For example, the grid lines can be as short as "time tics."

2. The programmer can exert greater control over the position of line labels by specifying a label reference location.

3. Selected grid lines can be emphasized and/or labeled in the same way as with GRID1V.

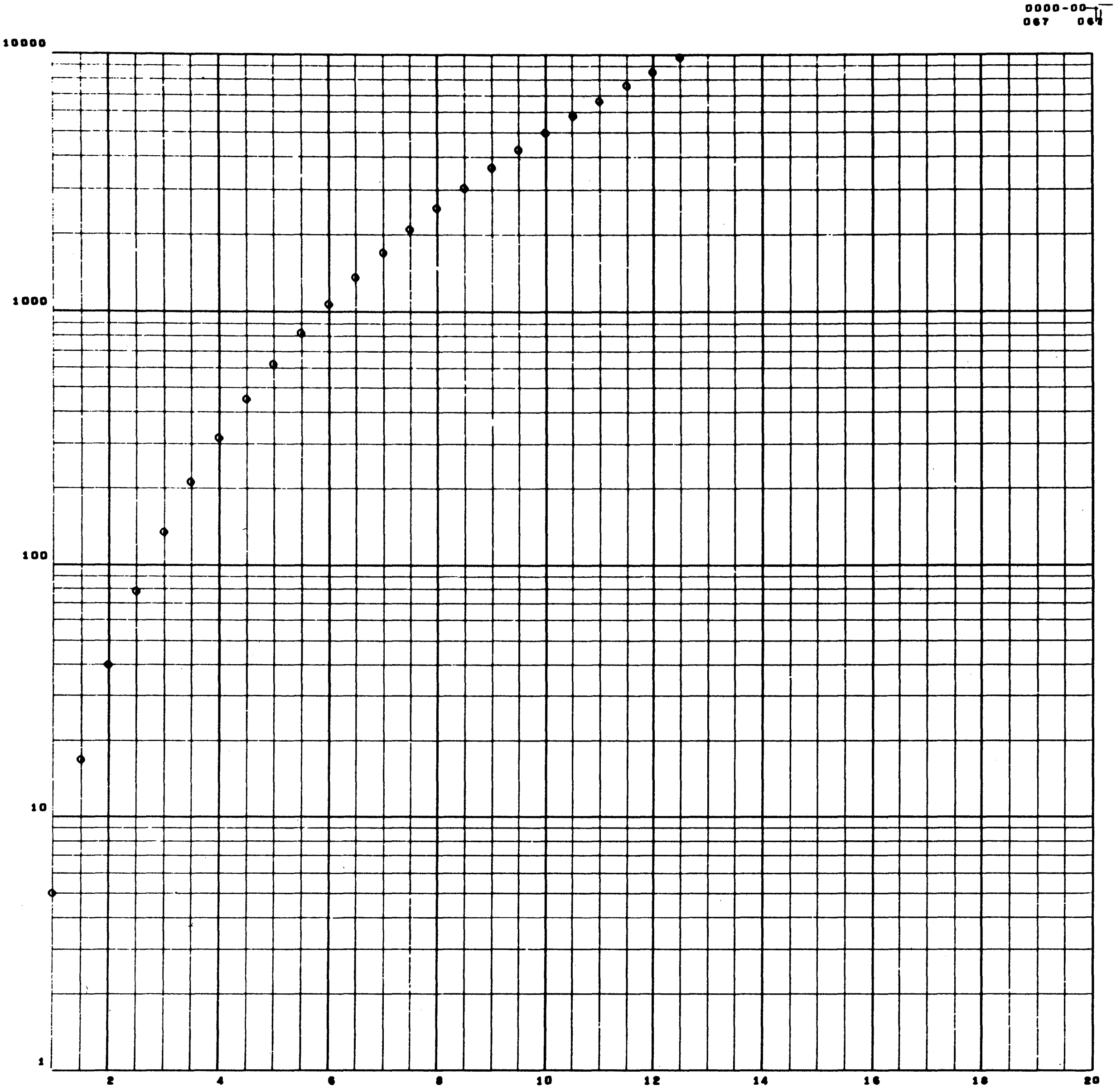
4. DXDYV can be used with LINRV in much the same way it is used with GRID1V.

Certain subprograms must be executed prior to the use of LINRV or NONLNV. The frame must have been advanced, and XSCALV and/or YSCALV must have established scale factors. If a change in scale mode is required, SMXYV must have been called.



```
CALL SMXYV ( 1, 1 )
CALL GRID1V ( 1, 1.0, 10.0, 1.0, 1000.0, 1.0, 1.0, 1, 1, 1, 1, 2, 4 )
         POINTS PLOTTED BY USING CALL POINTV ( X,Y, 1  )  IN A LOOP
```

Figure 5-25



```
CALL SMXYV ( 0, 1 )
CALL DXDYV ( 1, 1.0, 20.0, DX, N, I, NX, 12.0, IERR )
CALL GRID1V ( 1, 1.0, 20.0, 1.0, 10000.0, DX, 1.0, N, 1, I, 1, NX, 5 )
```

Figure 5-26

The call statement for using LINRV to generate a vertical grid is

```
    CALL LINRV (1, LYREFR, IYMIN, IYMAX, XL, XR,
                DX, ±N, ±I, ±NX, IW)
```

For a horizontal grid, the statement is

```
    CALL LINRV (2, LXREFR, IXMIN, IXMAX, YB, YT,
                DY, ±M, ±J, ±NY, IH)
```

LYREFR
LXREFR
Label reference locations.
For a vertical grid, LYREFR is the Y raster coordinate that will be used to position the labels of the vertical grid lines. The X raster coordinate will vary, and will be computed by the subprogram.

For a horizontal grid, LXREFR is the X raster coordinate that will be used to position the labels of the horizontal grid lines. The Y raster coordinates will vary, and will be computed by the subprogram.

The character dimensions, IW and IH (below), should be considered in assigning LYREFR and LXREFR. For example, LYREFR should be chosen to allow at least one character height (IH) below the origin of vertical grid lines (IYMIN). LXREFR should be a raster position that is at least IW* (NY + 1) raster counts to the left of the origin of the horizontal grid lines (IXMIN). In both instances, some additional space should be added to prevent overlapping.

If scientific labels are selected (-NX and/or -NY), provision must be made for 7 additional label characters in the computation of LXREFR. About 5 raster counts extra must be added to LYREFR to allow for the raised exponent.

NOTE: IW* (NY + 1) allows space for a sign μ.

IYMIN, IYMAX
IXMIN, IXMAX
Raster positions that determine the origin and end of each line. For a vertical grid, these integers are raster counts in the Y direction. For a horizontal grid, these integers are raster counts in the X direction.

IW, IH — The dimension to be allowed for each label character: IW for the width, and IH for the height. Since LINRV uses LABLV to display the labels, and since the system LABLV uses Charactron characters, the dimensions should be IW = 8 and IH = 10. (If the standard values have not been altered, these values can be obtained by calling SETCOV).

The remainder of the arguments have the same definitions given for these terms in the discussion of GRID1V.

The vertical portion of a log grid can be generated by the statement

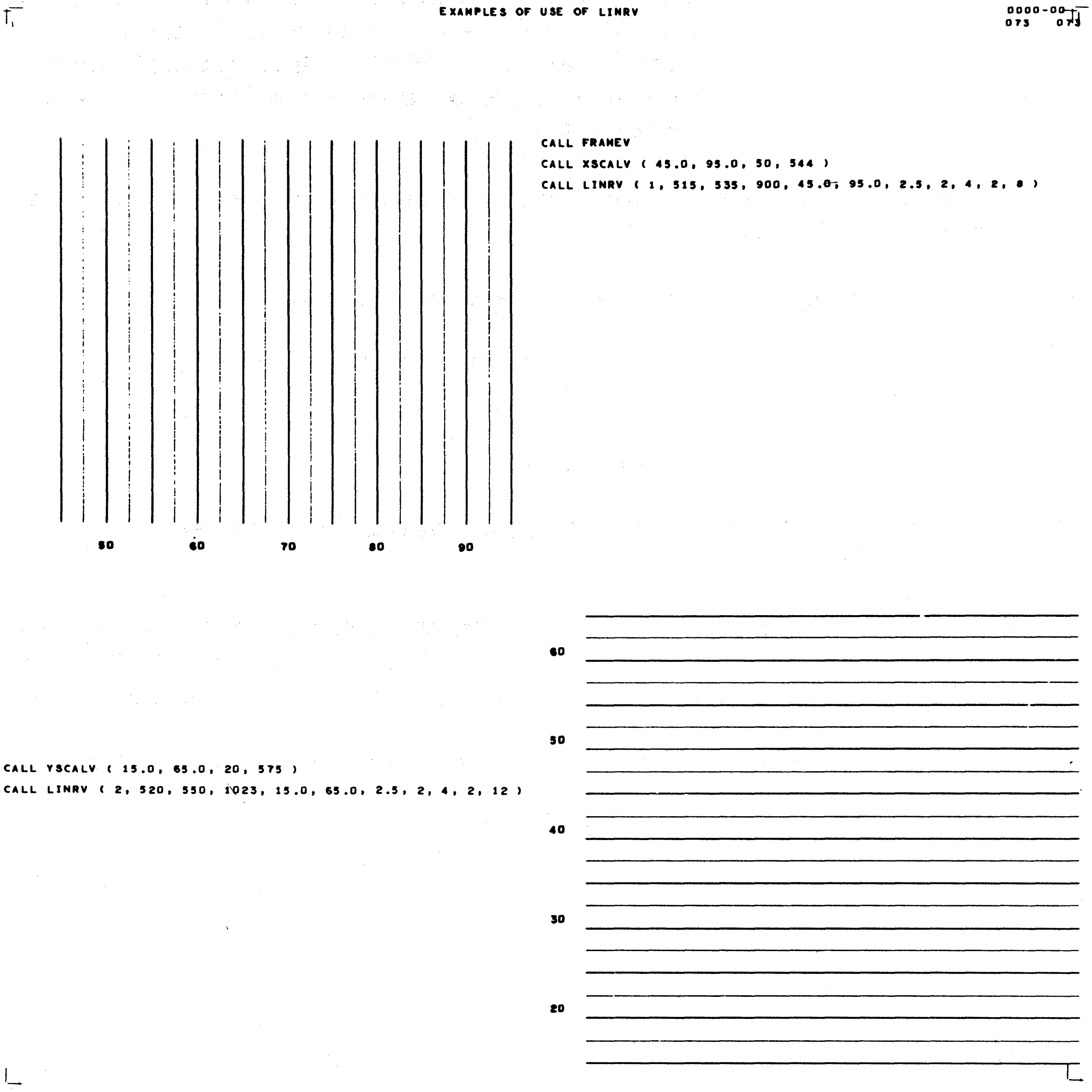
```
CALL NONLNV (1, LYREFR, IYMIN, IYMAX, XL, XR, DX,
             ±N, ±I, ±NX, IW)
```

For a horizontal log grid, the statement is

```
CALL NONLNV (2, LXREFR, IXMIN, IXMAX, YB, YT, DY,
             ±M, ±J, ±NY, IH)
```

The arguments XL, XR, DX, N, I, NX, and YB, YT, DY, M, J, NY have the same definitions (and restrictions) given for these terms in the discussion of GRID1V for the log mode. The remaining arguments have the definitions given for LINRV.

Figures 5-27 and 5-28 show examples of portions of grids created by LINRV and NONLNV, respectively.



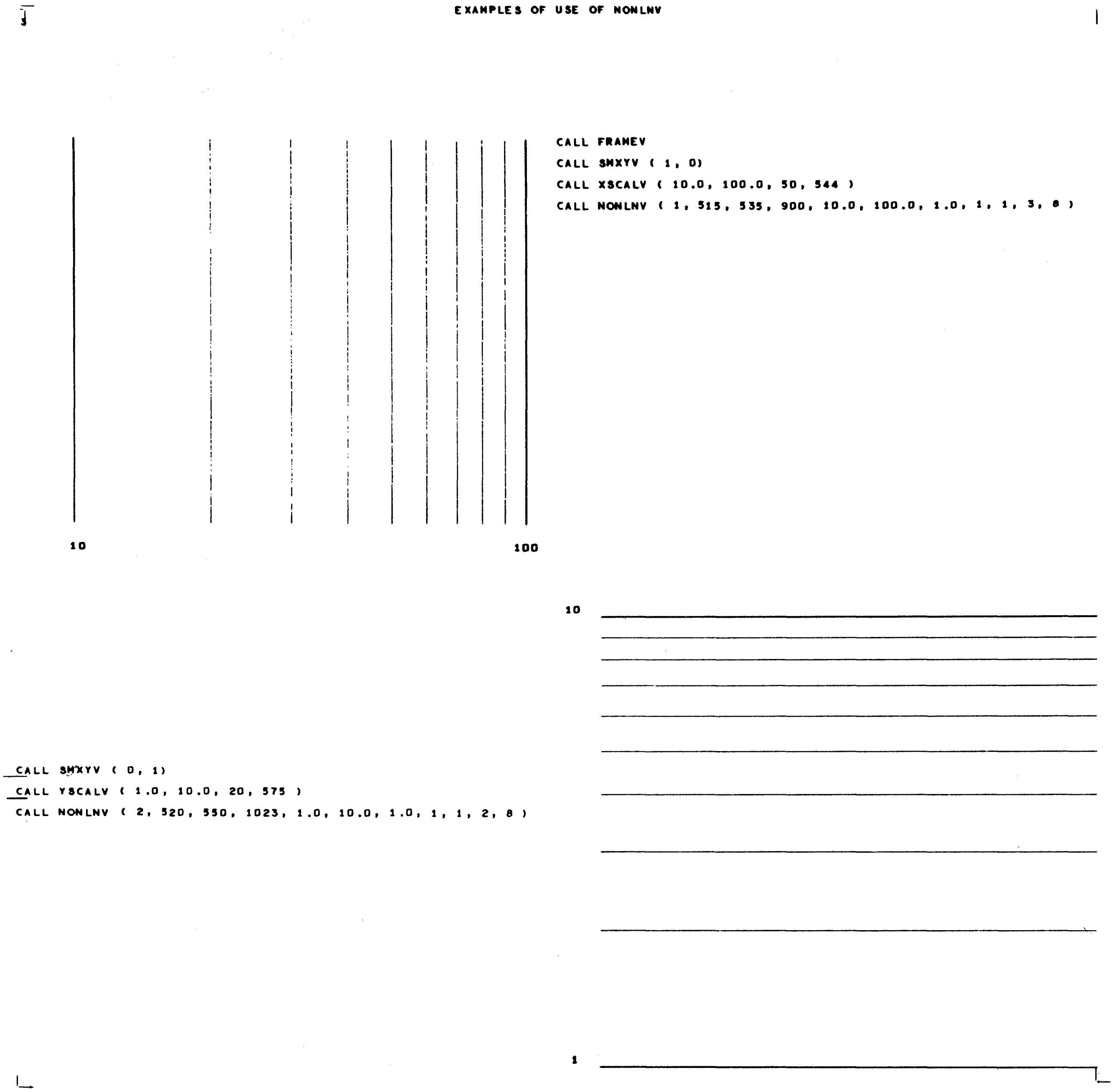
EXAMPLES OF USE OF LINRV

0000-00
073 073

```
CALL FRAMEV
CALL XSCALV ( 45.0, 95.0, 50, 544 )
CALL LINRV ( 1, 515, 535, 900, 45.0, 95.0, 2.5, 2, 4, 2, 8 )
```

```
CALL YSCALV ( 15.0, 65.0, 20, 575 )
CALL LINRV ( 2, 520, 550, 1023, 15.0, 65.0, 2.5, 2, 4, 2, 12 )
```

Figure 5-27



EXAMPLES OF USE OF NONLNV

```
CALL FRAMEV
CALL SMXYV ( 1, 0)
CALL XSCALV ( 10.0, 100.0, 50, 544 )
CALL NONLNV ( 1, 515, 535, 900, 10.0, 100.0, 1.0, 1, 1, 3, 8 )
```

```
CALL SMXYV ( 0, 1)
CALL YSCALV ( 1.0, 10.0, 20, 575 )
CALL NONLNV ( 2, 520, 550, 1023, 1.0, 10.0, 1.0, 1, 1, 2, 8 )
```

Figure 5-28

AXIS LINES: XAXISV, YAXISV

Axis lines XAXISV, YAXISV enable the programmer to use the axis line feature of the S-C 4020 to generate horizontal or vertical lines. Horizontal axis lines started at a specified raster position will be swept to the right to the specified X stop point in raster count. Vertical axis lines started at a specified raster position will be swept upwards to the specified Y stop point in raster count. The axis generator will not sweep lines down or to the left; therefore, the program automatically interchanges the coordinates if it is required in order not to stop the S-C 4020. If no stop point is given in the parameter list, the axis will be swept to the right edge of the frame or the top edge of the frame.

The call statement for sweeping a horizontal line is:

```
    CALL XAXISV (IX, IY)

or  CALL XAXISV (IX, IY, NSTPT)
```

IX, IY The fixed point raster coordinates of the origin of the line. IX may have a value from 0 to 959, while IY may have a value from 0 to 959.

NSTPT The fixed point X coordinate of the stop point.

For a vertical line, the statement is:

```
    CALL YAXISV (IX, IY)

or  CALL YAXISV (IX, IY, NSTPT)
```

IX, IY The fixed point raster coordinates of the origin of the line. IX may have a value from 0 to 959, while IY may have a value from 0 to 959.

NSTPT The fixed point Y coordinate of the stop point.

Section VI

SCALING AND CONVERSION

In earlier sections, scaling and conversion problems have been left to the routine GRID1V. However, much of the actual computation is done in lower-level modules. This section describes these and some associated modules that provide additional tools for some scaling and conversion problems.

The descriptions may also be useful in clarifying the operation of higher-level routines. For example, GRID1V uses XSCALV, YSCALV as a lower-level routine to do scaling. Consequently, the comments in this section concerning scaling and conversion equations, scale factors, and retrieving and resetting scale factors also apply when the programmer uses GRID1V to control scaling.

Methods of operation in both the linear and nonlinear modes are discussed. The nonlinear mode built into the system is the logarithmic mode, but the possibility of substitution of other nonlinear modes is mentioned.

BASIC SCALING SUBPROGRAMS: XSCALV, YSCALV

XSCALV, YSCALV will compute the scale factors for a specified display and store them in an internal table for later use by those functions which convert data. The calling statements are:

```
CALL XSCALV (XL, XR, ML, MR)

CALL YSCALV (YB, YT, MB, MT)
```

XL, XR Floating point values of X for the leftmost and rightmost limits of the scaled plotting area.

ML, MR The amount of margin space to be reserved to the left and right of the scaled area, expressed in raster counts (fixed point integers).

YB, YT Floating point values of Y for the bottom and top limits of the scaled plotting area.

MB, MT The amount of margin space to be reserved below and above the scaled area, expressed in raster counts (fixed point integers).

XSCALV, YSCALV contains a test for nonlinear mode. If this mode is indicated, XL, XR and/or YB, YT will be transformed before the scale factors are computed by the basic scaling equations.

Example

Figure 6-1 illustrates the relationship of the arguments. The margin specifications are: ML = 170, MR = 192, MB = 340, MT = 128.

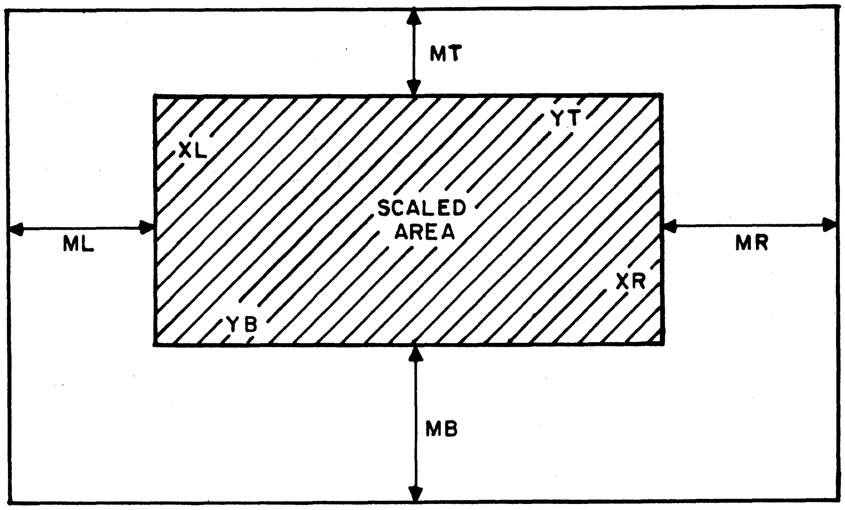


Figure 6-1

XSCALV will assign XL to raster location IX = 170, and XR to raster location IX = 831 (i.e., 1023 - 192). YSCALV will assign YB to raster location IY = 340, and YT to raster location IY = 895 (i.e., 1023 - 128). The scaled area will then be the rectangle from IX = 170 to IX = 831, and from IY = 340 to IY = 895.

BASIC SCALING EQUATIONS

In the following equations, "A" and "B" represent the scale factors computed and stored by XSCALV, and "C" and "D" are the factors computed and stored by YSCALV. (Since the computation is done in floating point arithmetic, the floating point variable names FML, FMR, FMB, and FMT are used to represent the floating point equivalents of the margin values ML, MR, MB, and MT.)

$$A = \frac{(1023.- \text{FMR}) - \text{FML}}{\text{XR} - \text{XL}} \qquad \text{I}$$

$$B = \text{FML} - \text{A*XL} \qquad \text{II}$$

$$C = \frac{(1023. - \text{FMT}) - \text{FMB}}{\text{YT} - \text{YB}} \qquad \text{III}$$

$$D = \text{FMB} - \text{C*YB} \qquad \text{IV}$$

CONVERSION OF DATA: NXV, NYV, IXV, IYV

Four function subprograms, NXV, NYV, IXV, and IYV, are provided to convert data coordinates into raster coordinates. The argument for each of the functions must be a floating point quantity; the result will be an integer quantity.

The following FORTRAN statements show how these functions may be used to convert data coordinates X (or Y) into raster coordinates IX (or IY):

```
IX = NXV(X)

IY = NYV(Y)

IX = IXV(X)

IY = IYV(Y)
```

These four functions are similar in that they all convert data by means of the basic equations for data conversion discussed below. They are dissimilar in the way they handle off-scale data (that is, data which falls outside the limits XL, XR or YB, YT.)

The functions NXV and NYV check for off-scale data values. The result IX (or IY) will be set to zero if the argument X (or Y) is outside the limits that were used to establish the scale. In addition, an error indication is set, as discussed under Off-Scale Error Detection.

The functions IXV and IYV do not test for off-scale data values. The resulting position can be outside the plotting area, or even outside the frame, but the value will be properly scaled relative to the plotting area. (However, no test is made for the possibility that the result is greater than 131,071; integer bits above the 17th will be lost.)

Figure 6-2 illustrates how error testing of the results of NXV, NYV can be used to by-pass plotting of points that are off-scale. NXV and NYV were used to convert the points along the curve into raster positions, and LINEV was employed to connect the points. Since the results of NXV and NYV were tested for zeros, and plotting was by-passed whenever a point was off-scale, the curve stopped at the top and right limits of the scaled area (outlined).

In Figure 6-3 two sine curves are shown, one plotted after using NXV, NYV to do the conversion, and the other after IXV, IYV were used. As in Figure 6-2, LINEV was used to connect the points. NO ERROR TESTS WERE MADE.

The lower curve shows the line going to zero when off-scale values were encountered by NXV, NYV. Note that this curve drops to the bottom of the frame (IY = 0) when values that were off-scale in Y were encountered. Also note that the off-scale initial and last values of X caused the curve to start and end at the left edge of the frame (IX = 0).

The higher curve was drawn after IXV, IYV were used to convert the points. The curve continued past the boundaries of the scaled area when off-scale values were encountered. (When IXV, IYV are used, the programmer must decide what action should be taken when the result of IXV and/or IYV is $<$ 0 or $>$ 1023.)

All four functions test for nonlinear mode. If indicated, X (or Y) will be transformed before it is converted by one of the basic conversion equations.

BASIC CONVERSION EQUATIONS

The following equations show how the conversion functions convert data coordinates X and Y into raster coordinates IX and IY:

$$IX = A*X + B$$

$$IY = C*Y + D$$

The scale factors A, B, C, and D are those derived from the equations I, II, III, and IV.

Generally speaking, the programmer should use the conversion functions rather than writing statements of his own containing these equations. The functions offer the

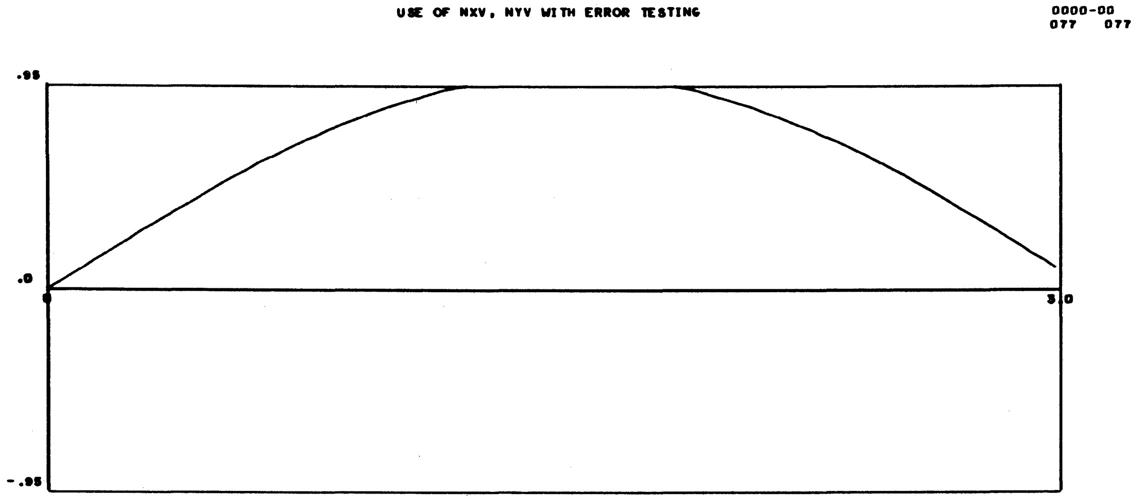


Figure 6-2

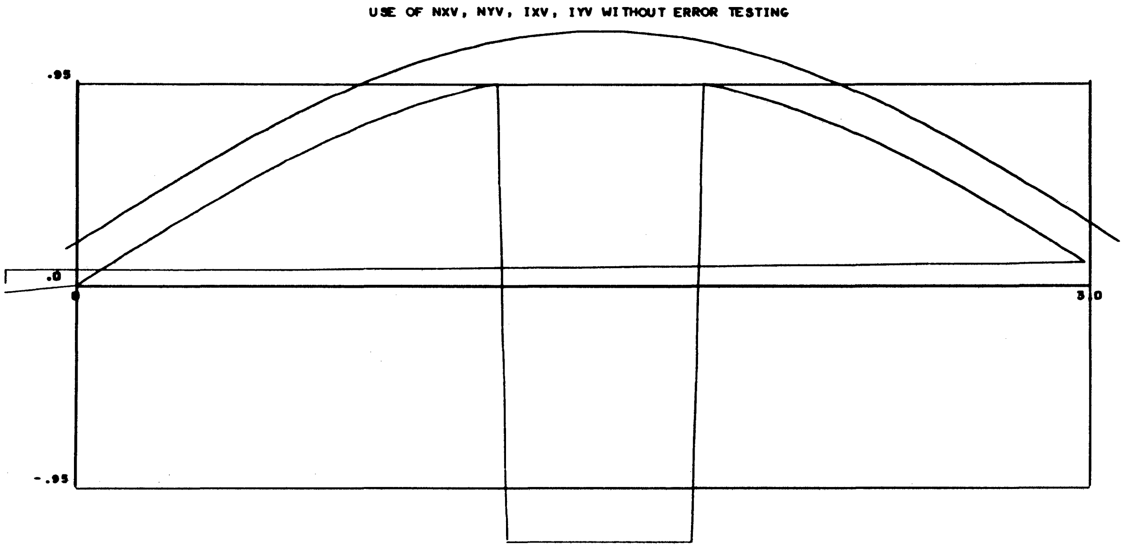


Figure 6-3

following advantages:

a. They have direct access to the internal table in which the scale factors A, B, C, and D are stored.

b. They check the scale mode, and use a nonlinear conversion if that mode is indicated.

c. NXV, NYV contain a test for off-scale points.

If it is ever necessary to use these equations directly, the programmer can retrieve the scale factors (A, B, C, D) by employing the routine SCLSAV.

INVERSE CONVERSION UXV, UYV

The functions UXV, UYV allow the programmer to obtain the coordinates of a specified raster location in terms of his data. The following statements show how these functions may be used:

```
X = UXV (IX)

Y = UYV (IY)
```

Although UXV, UYV represent the inverse of IX = IXV (X) and IY = IYV (Y), the results are approximate because truncation occurs in the IXV and IYV functions).

NOTE

UXV, UYV CANNOT BE USED
IN THE NONLINEAR MODE.

The equations used by UXV, UYV are the inverse of the equations for data conversion:

$$X = \frac{IX - B}{A}$$

$$Y = \frac{IY - D}{C}$$

(The computation is performed in floating point arithmetic.)

RETRIEVAL OF SCALING INFORMATION: SCLSAV

The subroutine SCLSAV will retrieve scale factors and other scaling information from an internal table and store them in an array named by the programmer. Although SCLSAV makes scaling information available for special-purpose conversions, certain limit tests, etc., its principal value is that it permits saving scaling information from one program link to another. When a new link is entered, another routine RESCLV, can be called to restore the scaling information in the internal table, where it is accessible to NXV, NYV, IXV, or IYV.

The calling statement to retrieve the scaling information is:

```
CALL SCLSAV (R)
```

R The name of a ten-cell array, named and dimensioned by the programmer.

The storage locations in the block of cells, R, are assigned as follows:

R(10)	Minimum IY	
R(9)	Minimum IX	
R(8)	Maximum IY	
R(7)	Maximum IX	
R(6)	D	
R(5)	B	Scale factors
R(4)	C	
R(3)	A	
R(2)	Scale mode indicator for Y	
R(1)	Scale mode indicator for X	

RESETTING SCALING INFORMATION: RESCLV

If the scaling information has been stored in COMMON by SCLSAV, it can be reset into the internal table when a new chain link is entered. The statement is:

```
CALL RESCLV(R)
```

where R is the ten-cell array described under SCLSAV.

NONLINEAR SCALING AND CONVERSION

The only nonlinear capability built into the system is logarithmic scale mode, used in connection with log grids. However, the system design allows the programmer to incorporate some special nonlinear scale mode, by the substitution of one module of his own. For this reason, the more inclusive term, "nonlinear," is used in the following discussion, instead of "logarithmic." The linear and nonlinear scale mode indicators are discussed on page 5-35.

NONLINEAR TRANSFORMATION: XMODV, YMODV

The scaling and conversion equations shown under number conversion apply not only to the linear mode, but also to the nonlinear mode if transformed arguments are used. The functions XMODV, YMODV are provided to perform such transformation; they can be used in statements of the following type:

```
XPRIME = XMODV(X)

YPRIME = YMODV(Y)
```

The scaling and conversion subprograms (XSCALV, YSCALV, IXV, IYV, NXV, NYV) test the scale mode indicators to see if they have been set to nonlinear mode by a call SMXYV. If nonlinear mode is indicated, each of these subprograms will use XMODV (or YMODV) to perform a nonlinear transformation on X (orY) before the remainder of the scaling or conversion takes place. Since the XMODV, YMODV functions in the system compute the log of X or Y, the system nonlinear mode is synonymous with log mode.

For a problem requiring special nonlinear transformation, the programmer can substitute subprograms of his own named XMODV and YMODV, with one argument and one result. Since the system XMODV and YMODV are physically contained in one subprogram, both must be replaced if a substitution is made for either. Obviously, such substitute functions must meet the nonlinear scaling and conversion requirements of the entire program (or chain link), since they will replace the system functions.

System subprograms that use XMODV and YMODV (either directly or indirectly) are: GRID1V, IXV, IYV, LINRV, NONLNV, NXV, NYV, XSCALV, YSCALV, APLOTV, POINTV.

> WARNING: GRID1V WAS DESIGNED ONLY FOR THE LINEAR, LOG, AND SEMI-LOG OPTIONS. SUBSTITUTION OF A SPECIAL TRANSFORMATION FUNCTION MAY CAUSE UNEXPECTED DIFFICULTIES IF USED BY GRID1V.

OFF-SCALE ERROR DETECTION

Whenever there is a possibility that off-scale data points might be encountered, error tests should be made by the programmer. For example, APLOTV sets an error indicator which should be tested. A zero result from NXV or NYV nearly always indicates an error; a test should be made for this condition.

There are additional situations which demand special error detection procedures. For one thing, if no left and/or bottom margin space is reserved, the conversion of XL and/ or YB can produce a legitimate zero result from NXV, NYV. More important, the programmer may be using NXV or NYV indirectly, via other modules, and thus be unable to test the results. For these reasons, additional subprograms are provided for detailed analysis of conversion errors resulting from off-scale points.

Keep in mind that the special procedures which follow are designed for unusual situations, in which normal error testing does not suffice.

SET CONVERSION ERROR INDICATORS: SCERRV

Two internal cells are used by NXV and NYV to store indications of successful or unsuccessful data conversion. The subroutine SCERRV allows the programmer to assign two cells which NXV, NYV will use in place of the internal error cells. In this way, the programmer can name error indicator locations that are accessible to his program. The call statement is:

```
CALL SCERRV (KX, KY)
```

When NXV converts a quantity successfully, it will place a "0" in KX; when unsuccessful, it will store a "1" in KX. NYV will use the cell KY in the same way. The cells named will be used in each subsequent execution of NXV, NYV (including execution via other subprograms) until new cells are named by another call to SCERRV (or the internal cells are reset at the beginning of a new link of a chain job). If possible, tests of these error cells should be made as soon after execution of NXV, NYV as possible, to avoid any possibility that they might be altered by subsequent executions of NXV, NYV.

The following practical examples show how SCERRV can provide error indicator cells in connection with POINTV. Since POINTV uses NXV and NYV only once, the named error cells will still contain indications of off-scale errors when control is returned from POINTV to the calling program. (The examples assume that the scale factors have already been established.)

Example 1

```
          .

          .

          .

          CALL SCERRV (KX, KY)

          DO 300 I = 1, N

          CALL POINTV (X, Y, NS)

          IF (KX*KY) 700, 300, 700       Test for non-zero KX and/or KY

      300 CONTINUE

          .

          .

          .

      700 CALL DUMP                      Dump when an off-scale point is
                                         encountered
```

Example 2

```
          .

          .

          .

          CALL SCERRV (KX, KY)

          M1 = 0

          M2 = 0

          DO 300 I = 1, N

          CALL POINTV (X, Y, NS)

          M1 = M1 + KX

    300   M2 = M2 + KY

          .

          .

          .
```

In Example 2, M1 and M2 will contain the total number of points that are off-scale. The contents of these locations may be printed out at the end of the job or after all points have been plotted on each frame.

SAVING AND RESETTING ERROR INDICATOR CELLS: SERSAV, SERREV

As has been pointed out, the locations assigned by SCERRV will be used as error indicator cells until new ones are named (or a new link of a chain job is loaded). As a result, subsequent executions of NXV, NYV, whether on the same level or on a lower or higher one, will alter the contents of the error cells (setting them to 1 for an unsuccessful conversion, to 0 for a successful conversion). The only two levels involved are NXV and NYV themselves, and POINTV.

Two problems can occur. The obvious one is that indications of error may be masked by a subsequent execution of NXV, NYV. This can usually be avoided if the programmer completes error tests before there is any possibility that the contents of the error cells might be changed.

The less obvious problem is that the error cells named in one level of a program will not be accessible for error testing in other levels unless special action is taken. One way to solve this problem is to carry the error cell names in the call statements of the subprograms, not always a desirable solution.

The modules SERSAV and SERREV, used in conjunction with SCERRV, offer a convenient means for avoiding these problems in multi-level jobs. The call statements are:

```
CALL SERSAV (LOCX, LOCY)
```

> This subprogram saves, in LOCX and LOCY, the locations of the cells that are currently being used for off-scale indicators.

```
CALL SERREV (LOCX, LOCY)
```

> This subroutine resets the off-scale error cell locations that were saved by SERSAV.

The arguments, LOCX and LOCY, must be variable names (either fixed or floating point) which are not used for any other purpose between the execution of SERSAV and of SERREV.

Section VII

PLOTTING AND LINE GENERATION

PLOTTING DATA

The 64 CHARACTRON characters are shown in Figure 7-1 along with the selection code for each. Note that, for FORTRAN characters, the selection codes are the same as the BCD codes recognized by IBM equipment.

Reduced to fundamentals, printing or plotting of one CHARACTRON character involves the selection and display of that character at a specified position on the raster. (A basic subprogram, PLOTV, can be used to select and display one character at a time.) The higher-level subprograms, however, contain features that make each one suitable for a specialized purpose: plotting, printing, or labeling. These specialized features of the higher-level routines make them appear to be distinctly different from each other.

POINT PLOTTING SUBPROGRAMS

Since point plotting usually involves the scaled representation of a physical phenomenon, most point plotting subprograms accept physical data coordinates for position information. During execution of these subprograms, the data coordinates are converted into raster coordinates. Scale factors must have been established for the plotting routines to use in making these conversions; this requirement can be satisfied by a prior entry to GRID1V.

Scale factors are not normally saved from link to link in a chain job. Consequently, plotting should be done within the link in which scale factors are computed.

PLOTTING AN ARRAY: APLOTV

APLOTV was designed for situations in which a large number of X values are stored in one array and the corresponding Y values are stored in another array. It is possible to plot the entire set of data with one entry to APLOTV. If desired, only a portion of the data can be plotted.

S-C 4020 STANDARD SCIENTIFIC CHARACTERS

NUMERIC

CHAR	DECIMAL	OCTAL
0	0	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	10
9	9	11

ALPHABETIC

CHAR	DECIMAL	OCTAL
A	17	21
B	18	22
C	19	23
D	20	24
E	21	25
F	22	26
G	23	27
H	24	30
I	25	31
J	33	41
K	34	42
L	35	43

ALPHABETIC

CHAR	DECIMAL	OCTAL
M	36	44
N	37	45
O	38	46
P	39	47
Q	40	50
R	41	51
S	50	62
T	51	63
U	52	64
V	53	65
W	54	66
X	55	67
Y	56	70
Z	57	71

OTHER FORTRAN

CHAR	DECIMAL	OCTAL
=	11	13
+	16	20
−	32	40
*	44	54
/	49	61
.	27	33
,	59	73
(	60	74

OTHER FORTRAN

CHAR	DECIMAL	OCTAL
)	28	34
\$	43	53
BLANK	48	60

NON-FORTRAN

CHAR	DECIMAL	OCTAL
∂	10	12
"	12	14
˙	13	15
δ	14	16
α	15	17
?	31	37
β	29	35
±	30	36
π	26	32
•	42	52
γ	45	55
~	46	56
ɗ	47	57
°	58	72
∫	61	75
Σ	62	76
□	63	77

•, ∂, ~ NOT AVAILABLE IN TYPEWRITER MODE

SUGGESTED PLOTTING CHARACTERS AND COMBINATIONS

•	42	52	+	16	20	C	19	23	0	0	0	⊗	38,55	46,67
O	38	46	X	55	67	H	24	30	Ø	38,49	46,61	⊠	63,55	77,67
□	63	77	*	44	54	U	52	64	⧄	63,49	77,61			

Figure 7-1

It is also possible to use different CHARACTRON characters as symbols to identify curves. The programmer furnishes a table (of one or more CHARACTRON characters) that will be selected cyclically.

APLOTV also keeps a tally of the number of off-scale points encountered.

The calling statement is:

CALL APLOTV (N, XARRAY, YARRAY, JX, JY, ±NC, MARKPT, IERR)

N — N is the number of points in the arrays from which the data points are to be taken (as if all the data in the arrays are to be plotted in succession). N is normally the dimension of the arrays. The value of N may be computed by letting K in the following formulas equal the number of points to be plotted:

N = K*JX or N = K*JY, whichever is larger.

XARRAY
YARRAY — Normally, the names of the arrays of floating point data to be plotted. Since these arguments must name the locations of the data coordinates of the first point to be plotted, subscripts may be necessary. Both arrays must be in the same order of storage.

JX, JY — Fixed point positive integers giving the increments to be added to the subscripts of XARRAY and YARRAY as each point is plotted.

NC — The number of characters in the array MRKPT to be used as plotting symbols (usually the size of the array MRKPT). Positive NC indicates that MRKPT is a normal FORTRAN array. Negative NC shows that MRKPT is stored in increasing locations in core.

MRKPT The array that contains the selection codes of the plotting character(s) to be used. These will be used cyclically, with the first one being used again after the NCth one has been used.

To set up this information when only one character is to be used, the argument can be one of the following:

a. An integer selection code, as "42" for the plotting dot, "38" for the alphabetic letter O, etc. (see Figure 7-1)

b. A Hollerith argument for one FORTRAN character, as "1HO."

c. The name of a location containing an integer, as "MRKPT."

If more than one character is desired, the seventh argument may be one of the following:

a. An array of integers read in by the I format, or generated by a series of arithmetic statements: MRKPT(1) = 38; MRKPT(2) = 55, etc. Such an array must have a fixed point variable name.

b. An array read in by the A format, one character to a location. Such an array is not restricted to integer-type names; it could be called PTMRK, for example, if the contents are not used in a computation.

IERR The name of an error location supplied by the programmer. Any point that falls outside the grid drawn by GRID1V will not be plotted. Instead, a count of such points will be stored in IERR.

APLOTV plots only one character per point. If a center dot is desired, APLOTV may be repeated, using the plotting dot as the symbol. The use of APLOTV is illustrated by the Figures 7-2 and 7-3.

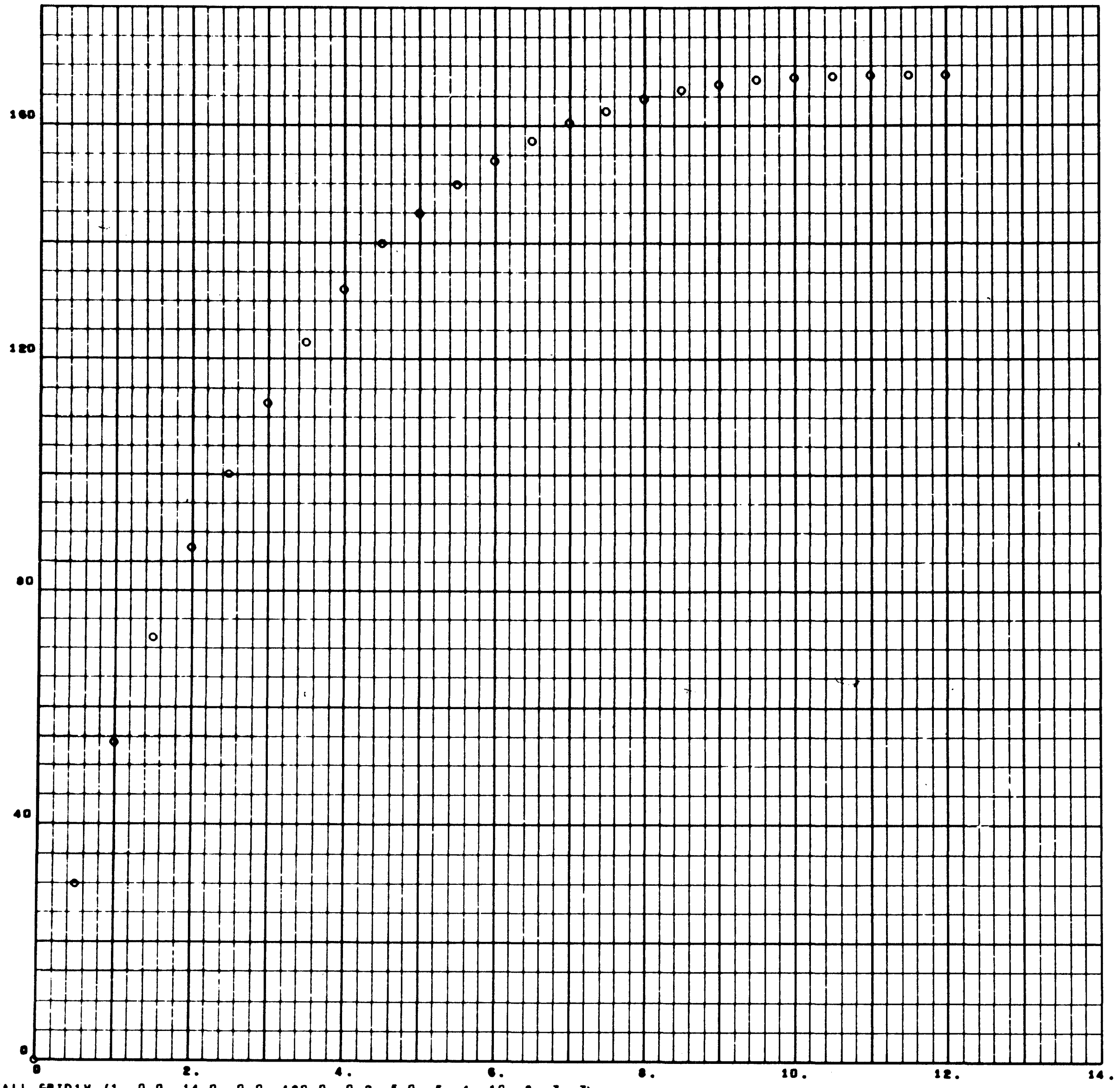
In Figure 7-2, it is assumed that the data are stored in two arrays: X(1), X(2), ..., X(25), and Y(1), Y(2), ..., Y(25). A prior entry was made to GRID1V, using the statement:

```
CALL GRID1V (1, 0.0, 14.0, 0.0, 180.0, 0.2, 5.0, 5, 4, 10, 8, 3, 3)
```

Note that the X and Y limits cover the range of the data to be plotted. Also, since GRID1V was entered first, the necessary scale factors have been established for APLOTV. The calling statement of APLOTV for this example is the following:

DATA IS STORED X1,X2,...X25 AND Y1,Y2,...Y25

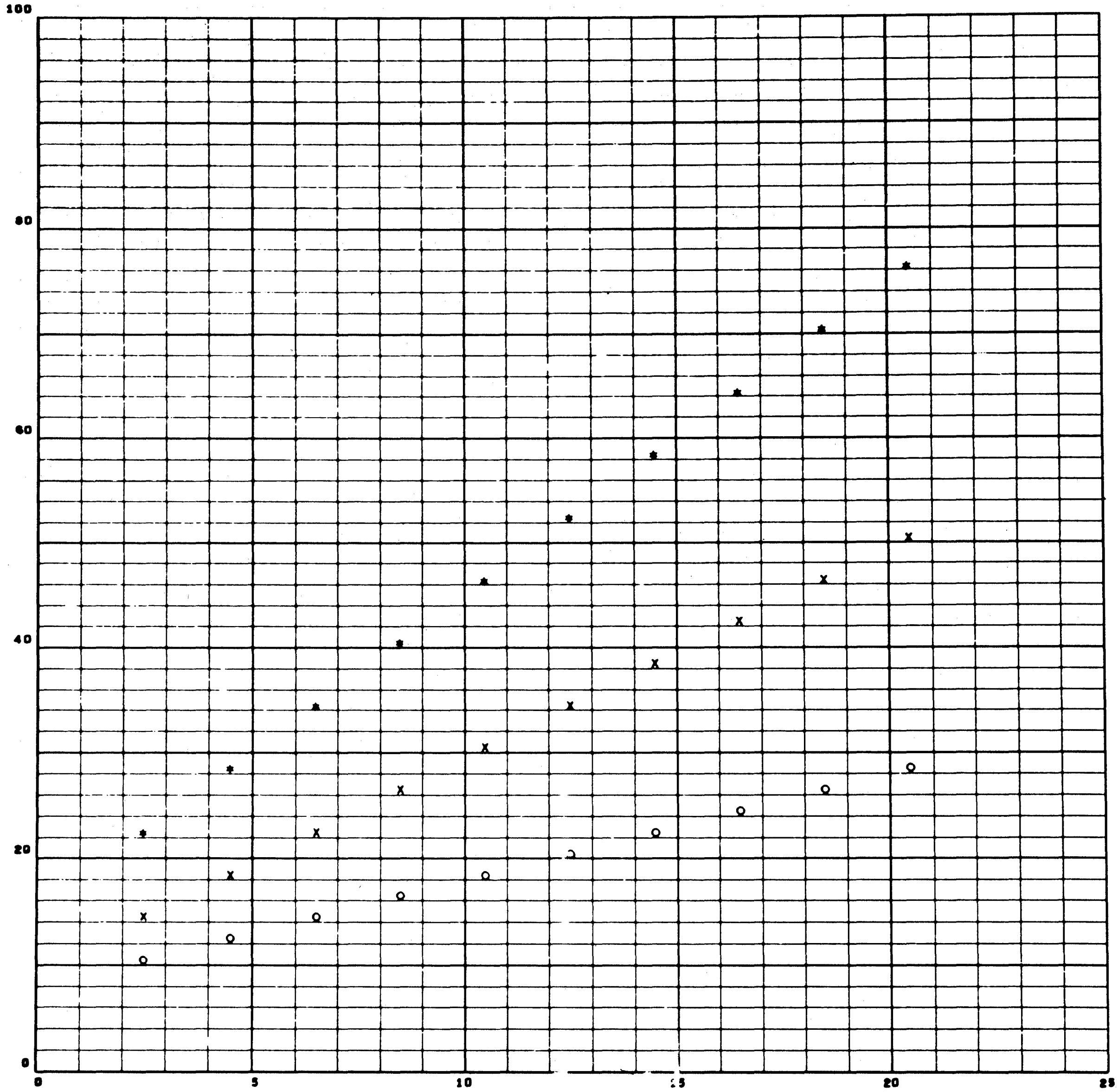
0000-00
018 018



```
CALL GRID1V (1, 0.0, 14.0, 0.0, 180.0, 0.2, 5.0, 5, 4, 10, 8, 3, 3)
CALL APLOTV (25, X, Y, 1, 1, 1, 38, IERR )   OR   CALL APLOTV (25, X, Y, 1, 1, 1, 1HO, IERR )
```

Figure 7-2

```
XA ARRAY IS STORED X1,X2,...X10   YA ARRAY IS STORED A1,A2,...A10 B1,B2,...B10 C1,C2,...C10          0000-00
PTMRK CONTAINS O,X,AND * LOADED UNDER CONTROL OF FORMAT (3A1)                                           020    020
```



```
     DO 1240 I=1,10
1240 CALL APLOTV ( 30, XA(I), YA(I), 0, 10, 3, PTMRK, IERR)
```

Figure 7-3

```
   CALL APLOTV (25, X, Y, 1, 1, 1, 38, IERR)
or CALL APLOTV (25, X, Y, 1, 1, 1, 1HO, IERR)
```

N = 25 The array size, 25 in this case, is equivalent to the number of points to be plotted. This can only be true when all points in each array are to be plotted.

X, Y Names of the X and Y arrays (properly dimensioned).

JX = 1
JY = 1 Since every X versus Y is to be plotted, the fourth and fifth arguments are set to 1.

NC = 1 Only one plotting symbol is to be used; hence the sixth argument is set to 1.

MARKPT In the two calling sequences shown, one specifies the plotting symbol by an integer; the other uses a Hollerith argument.

Figure 7-3 shows one way that APLOTV might be used to plot a family of curves. The example assumes that the XA and YA arrays are properly dimensioned. The data is stored XA(1), XA(2), . . . , XA(10); corresponding Y's for the three curves are stored in the array YA in the order A(1), A(2), . . . , A(10), B(1), B(2), . . . , B(10), C(1), C(2), . . . , C(10). PTMRK is the name of a three-word array which was loaded as the BCD equivalents (read in by the A format) of the characters O, X, and *.

In the example, A(I), B(I), and C(I) were plotted versus X(I) each time APLOTV was entered. A DO loop was used to proceed to the next value of X, so that a total of 10 entries were made to APLOTV. The coding was:

```
          DO 1240 I = 1, 10

1240      CALL APLOTV (30, XA(I), YA(I), 0, 10, 3, PTMRK, IERR)
```

Note that the plotting symbols are used cyclically, returning to the first one when the array PTMRK is exhausted. If desired, a center plotting dot can be superimposed upon the plotting symbol by repeating the entries to APLOTV with the plotting dot used for PTMRK.

PLOTTING INDIVIDUAL DATA POINTS: POINTV

For each entry to POINTV, one symbol is plotted (with or without a center dot). The coordinates may be specified as floating point data, which POINTV will convert into raster coordinates. Scale factors must have been established; this can be accomplished by a prior entry to GRID1V.

POINTV relieves the programmer of the burden of setting up a symbol table. The most suitable plotting symbols derived from CHARACTRON characters are arranged in a table, with the best choices near the beginning of the group. The symbols are selected by using integers from 1 to 48 as shown below. The calling statement is:

CALL POINTV (X, Y, ±NS)

X, Y	Coordinates of the point to be plotted, stated as floating point data values.
±NS	Integer which selects a plotting symbol. If NS is minus, there will be no center dot plotted. If NS is positive, a center dot will appear.

Points outside the scaled area will not be plotted. Until the programmer learns how to detect off-scale points (by a method explained in the advanced material), he should be sure that the data coordinates will fall within the limits of the scaled area.

An alternate version of this subprogram allows the programmer to specify position information in raster coordinates. This is particularly useful when the programmer wants to construct a legend in the margin, showing the symbols used and their meaning. The alternate call statement is:

CALL POINTV (IX, IY, ±NS, ANY)

IX, IY	Raster coordinates of the point to be plotted; fixed point.
±NS	Integer to select a plotting symbol, as stated above.
ANY	The inclusion of a fourth argument signals the subroutine that the position information is specified as raster coordinates. The form of this argument is immaterial; it may be any fixed point or floating point variable or constant.

The available symbols are shown in Figure 7-4. An illustration of the use of POINTV appears in Figure 7-5.

NS	SYMBOL	NS	SYMBOL	NS	SYMBOL	NS	SYMBOL
0	•	13	ʃ	26	G	39	E
1	O	14	Z	27	S	40	F
2	X	15	⊠	28	W	41	M
3	□	16	⧖	29	Q	42	P
4	Y	17	⊕	30	(	43	R
5	+	18	⊞	31	~	44	ɣ
6	*	19	Ø	32	)	45	2
7	L	20	⍁	33	N	46	3
8	U	21	⌿	34	T	47	7
9	0	22	⊟	35	=	48	8
10	H	23	Ø	36	D		
11	C	24	Ӿ	37	A		
12	V	25	I	38	B		

Figure 7-4

BASIC PLOT-PRINT SUBPROGRAM: PLOTV

Any one of the 64 CHARACTRON characters can be displayed at a specified raster position by using PLOTV, the basic subprogram used as a lower-level module of other routines. However, the programmer may find it useful when other plotting or printing subprograms are not suitable. The call statement is:

CALL PLOTV (IX, IY, NS)

IX, IY — Fixed point raster coordinates at which the character will be displayed.

NS — An integer which selects the character. The selection code (decimal equivalent) of the desired character should be used. (See Figure 7-1).

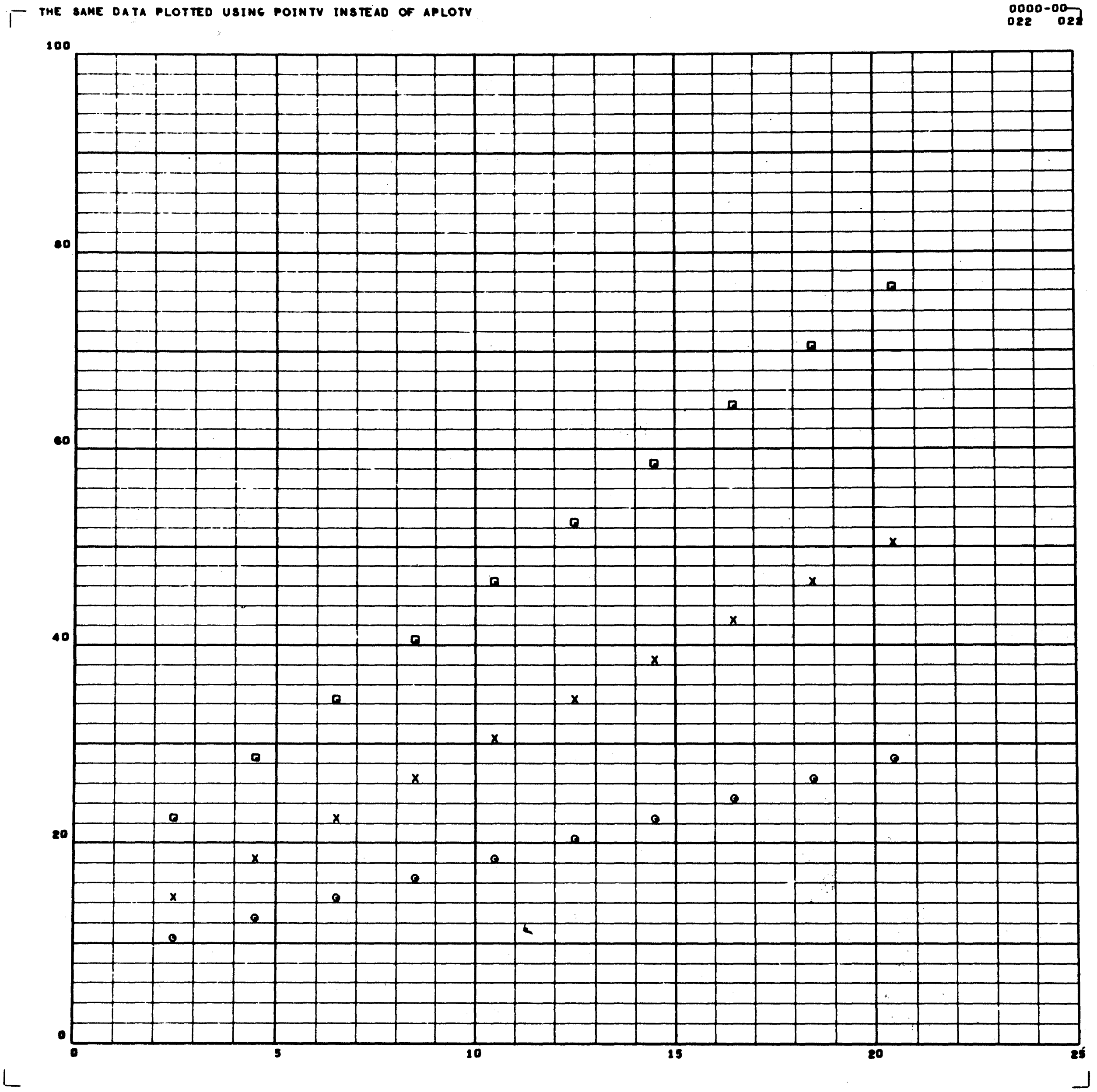

THE SAME DATA PLOTTED USING POINTV INSTEAD OF APLOTV
0000-00
022 022
100
80
60
40
20
0
0
5
10
15
20
25

Figure 7-5

LINE GENERATION: LINEV, LINE2V

LINEV connects two points by a straight line composed of vectors, joined end-to-end. The arguments for LINEV, which specify the points to be connected, must be given in raster counts. As described above, the programmer may connect two data points by a line if he first uses the functions NXV and NYV to convert the data coordinates into raster coordinates. (If there is a possibility that the data points being converted may be off-scale, the conversion results should be tested for errors before LINEV is executed.) The calling statement is:

```
CALL LINEV (IX1, IY1, IX2, IY2)
```

IX1, IY1	Raster coordinates of one end point.
IX2, IY2	Raster coordinates of the other end point.

Figure 7-6 contains an illustration of the use of LINEV.

LINE2V is used to draw a line from a fixed point in some direction specified by DX and DY. The calling sequence is:

```
CALL LINE2V (IX1, IY1, ±IDX, ±IDY)
```

IX1, IY1	Raster coordinates of starting point.
±IDX, ±IDY	Number of raster points that the line is to be extended in the X and Y directions.

In either LINEV or LINE2V, a floating point data value may be utilized,if scaling has been established,by utilizing the function subprograms NXV, NYV as follows:

```
CALL LINEV (NXV(X1), NYV(Y1), NXV(X2), NYV(Y2))
```

Figure 7-7 is an example of the use of LINE2V. Each line in the figure is produced by incrementing IX1 (IY1) by ±IDX (±IDY).

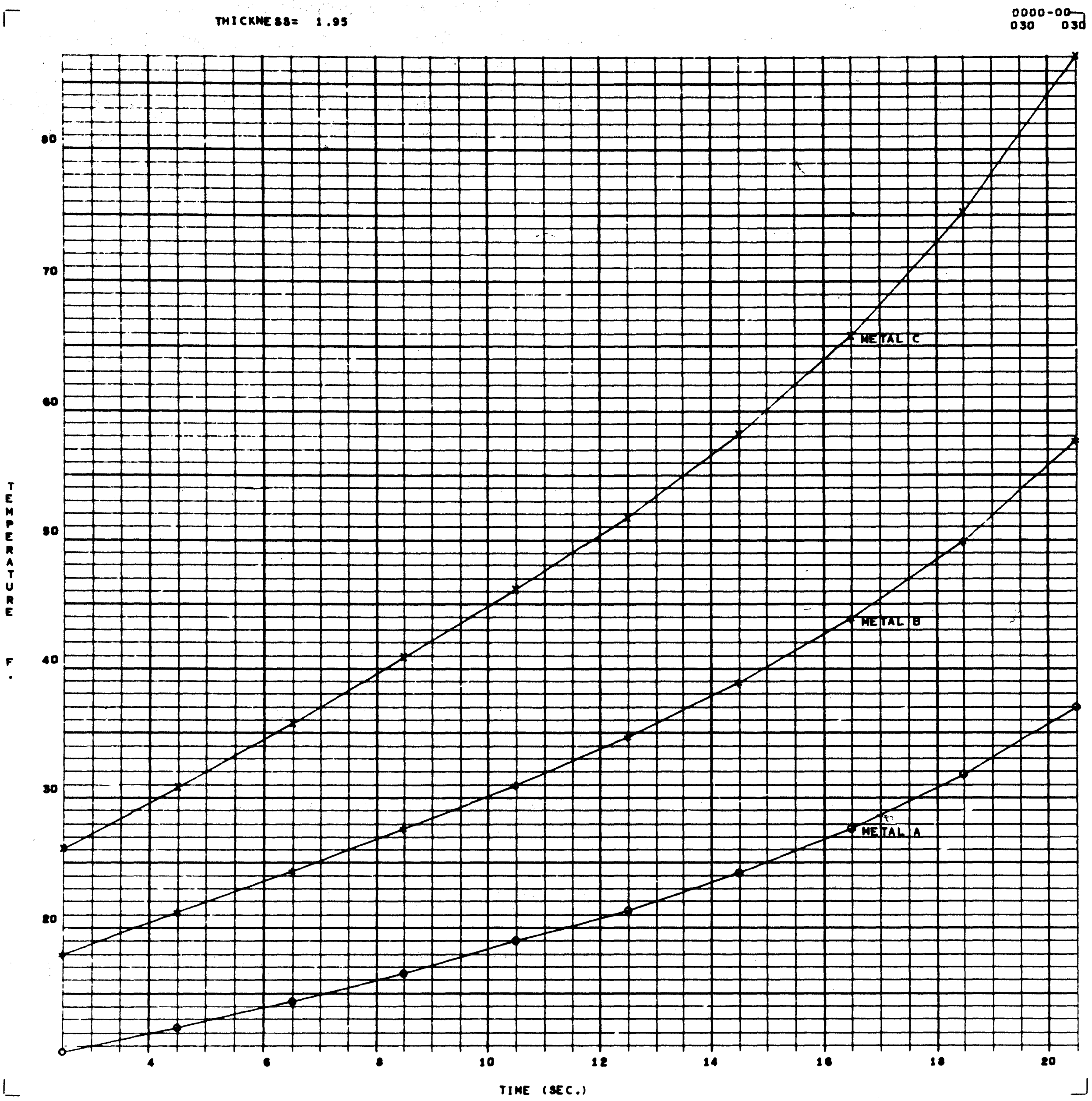

THICKNESS= 1.95
0000-00
030 030
80
70
60
50
40
30
20
TEMPERATURE F.
METAL C
METAL B
METAL A
4
6
8
10
12
14
16
18
20
TIME (SEC.)

Figure 7-6

```
   CALL XSCALV (-511.0, 512.0, 0, 0)
   CALL YSCALV (-511.0, 512.0, 0, 0)
   Z0 = 0.0
   Z1 = 4.0
   Z2 = 4.0
   CALL POINTV (Z0, Z0, -16)
   DO 5 I = 1, 63
   INC = I
   JNC = -INC
   CALL LINE2V (NXV (Z1), NYV (Z0), 0, INC)
   CALL LINE2V (NXV (Z2), NYV (Z0), 0, JNC)
   CALL LINE2V (NXV (Z0), NYV (Z1), JNC, 0)
   CALL LINE2V (NXV (Z0), NYV (Z2), INC, 0)
   Z1 = Z1 + 3.0
5  Z2 = Z2 - 3.0
```

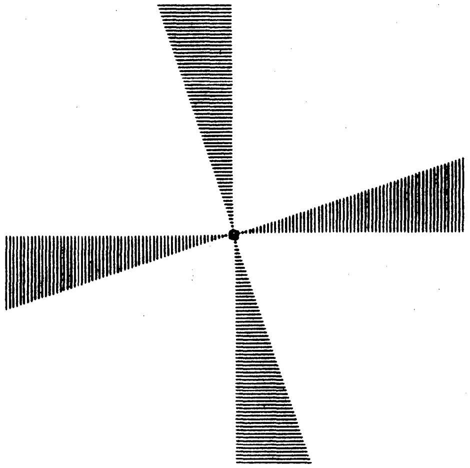


Figure 7-7

DASHED LINE SUBPROGRAMS: DOTLNV, INCRV

Calling Sequence:

```
CALL DOTLNV (IX1, IY1, IX2, IY2)
```

where: IX1, IY1 = start coordinates (integer values)
IX2, IY2 = stop coordinates (integer values)

The routine INCRV is used to supply the line and space size. This is set normally at 8 for the line and 4 for the space. These values may be altered, however, by calling INCRV prior to DOTLNV.

Calling Sequence:

```
CALL INCRV (IL, IS)
```

where: IL = desired length of line in raster units(integer value)
IS = desired size of space in raster units(integer value)

If IL and IS are such that the stop coordinates (IX2, IY2) terminate in the space portion of a dash segment, the last line portion extends to the stop coordinates. Illegal values of IL or IS are ignored by INCRV. DOTLNV will then use the last used values of IL and IS.

Section VIII

TITLING AND LABELING

TITLING AND LABELING SUBPROGRAMS

The printing and labeling subprograms enable the programmer to affix titles and other identifying information to a picture. Three subprograms of this type will be introduced: PRINTV, APRNTV, and LABLV. Other means for printing and labeling will be given in the section on printing.

For many applications, the positions of titles and labels must be independent of the scale. Therefore, printing and labeling subprograms accept position information in raster coordinates. This contrasts with the plotting routines, which includes facilities for the conversion of data into raster counts. Titles or labels may be positioned relative to data. The conversion functions discussed later can be employed to find raster coordinates from floating point location data.

TYPEWRITER PRINTING: PRINTV

The S-C 4020 has a built-in Typewriter Simulator, which prints a series of CHARACTRON characters very rapidly. The typewriter mode allows eight raster counts for the width of a character space. If "typewriting" exceeds 128 characters in any row and carries over to the next row, the next printing will start at the left edge of the frame, 16 raster counts below the preceding row. Writing takes place only in horizontal rows, and the characters are always upright. If the vertical advance causes the raster count to exceed 1023, printing will continue at the top of the frame, but no frame advance will occur.

In the Typewriter Mode, the machine recognizes octal codes 56 and 12 as special instructions. If these codes are accidentally used, imperfect pictures will result.

The PRINTV calling statement provides for printing characters read in by the A format, or characters specified in the call statement itself in the nH------ form.

CALL PRINTV (N, BCDTXT, IX, IY)

or CALL PRINTV (-N, nH------, IX, IY)

N	The number of CHARACTRON characters to be printed. A negative sign on N signals that the second item is a Hollerith argument. The typewriter mode is used by PRINTV until N characters have been printed.
BCDTXT	An array containing the BCD (A- or O-type format) text to be printed.
nH------	A Hollerith argument containing the text to be printed.
IX, IY	The raster coordinates for the center of the first character. IX and IY may be any number from 0 to 1023.

VERTICAL TITLES: APRNTV

This subprogram can be used to display vertical titles composed of CHARACTRON characters. Each individual character will be upright. The call statement provides for printing characters in BCD form (read in by the A format) or characters stored as a Hollerith argument. Since APRNTV does not use the Typewriter Mode, the spacing of the characters is controlled by arguments specified by the programmer. See Figure 8-1.

The call statement for APRNTV is:

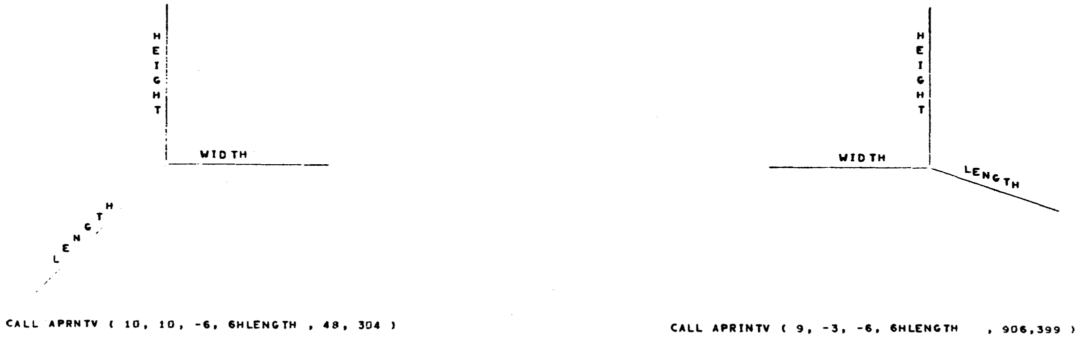
CALL APRNTV (INCRX, INCRY, N, BCDTXT, IX, IY)

or CALL APRNTV (INCRX, INCRY, -N, nH------, IX, IY)

INCRX INCRY	Increments used to space the characters in the X or Y direction, given in raster counts. For vertical titles, INCRX will be zero and INCRY should have a negative value. (It is suggested that INCRY fall in the range between -12 and -18 for vertical titles in most applications.) Figure 8-2 shows a vertical title, with INCRY equal to -14. Using a slightly higher value for the Y increment would have spaced the letters farther apart.

The remaining arguments are as specified under PRINTV.

APRNTV was introduced earlier and used to write vertical titles. The example in Figure 8-1 shows another application of APRNTV. This frame shows that APRNTV may be used to write along certain slopes.



```
CALL APRNTV ( 10, 10, -6, 6HLENGTH , 48, 304 )
CALL APRINTV ( 9, -3, -6, 6HLENGTH   , 906,399 )
```

Figure 8-1

DISPLAY OF NON-FORTRAN CHARACTERS

The characters in the last column of the table of CHARACTRON characters (Figure 7-1) require special consideration. IBM equipment does not provide for key-punching these characters, so they cannot appear in a Hollerith argument nor on an A-type format card. However, the octal equivalents of the selection codes can be read into storage by the O-type format, and APRNTV or PRINTV can be employed to display them.

A single non-FORTRAN character can easily be displayed by the basic plot-print subprogram, PLOTV, or by APLOTV, using a decimal integer as the selection code. If a character is being inserted into information otherwise printed by PRINTV, the fact that PRINTV allots 8 raster counts to each character must be taken into consideration.

FIXED POINT LABELS: LABLV

LABLV was developed for GRID1V to employ in labeling grid lines, but will prove helpful when the value of computed quantities must be printed. The routine performs two chief tasks: it converts a floating point quantity into a number in BCD format, with the decimal point in the proper position, and it displays the BCD number at the raster coordinates specified.

Before the BCD label is displayed, it is right adjusted in accordance with the specified decimal scale. (The number of positions to the left of the decimal point is called the "decimal scale.") During the right adjustment, BCD blanks are moved into the high order positions that are vacated. A specified number of characters counting leading blanks and decimal point will then be displayed.

The result of this procedure is demonstrated by the column of numbers: 150.5, 85.7, 0.3 in Figure 8-2. Notice that the three quantities have been truncated to the same number of decimal places, and that the decimal points have been aligned.

The calling statement of LABLV is:

```
CALL LABLV (D, IX, IY, NCHAR, NT, NDMAX)
```

D The floating point quantity to be printed.

IX, IY The raster coordinates which will position the first character of the label. Note that this first character may be a leading blank. If the quantity to be displayed is negative, the minus sign will be displayed one character space to the left of IX, IY.

NCHAR Number of characters to be displayed, including leading blanks and the decimal point, if any. NCHAR is limited to 6 (or 7 if one of the characters is the decimal point).

NT The number of times each character is to be displayed (number of over-strikes). Normally this should be 1, but 2 or more may be chosen if a darker label is desired.

NDMAX Maximum decimal scale; i.e., maximum number of characters to be displayed to the left of the decimal point.

An integer quantity may be displayed by first changing it to floating point form and then using LABLV.

An alternate form of LABLV may be used to display labels in scientific notation. The call statement shown is used with the following changes:

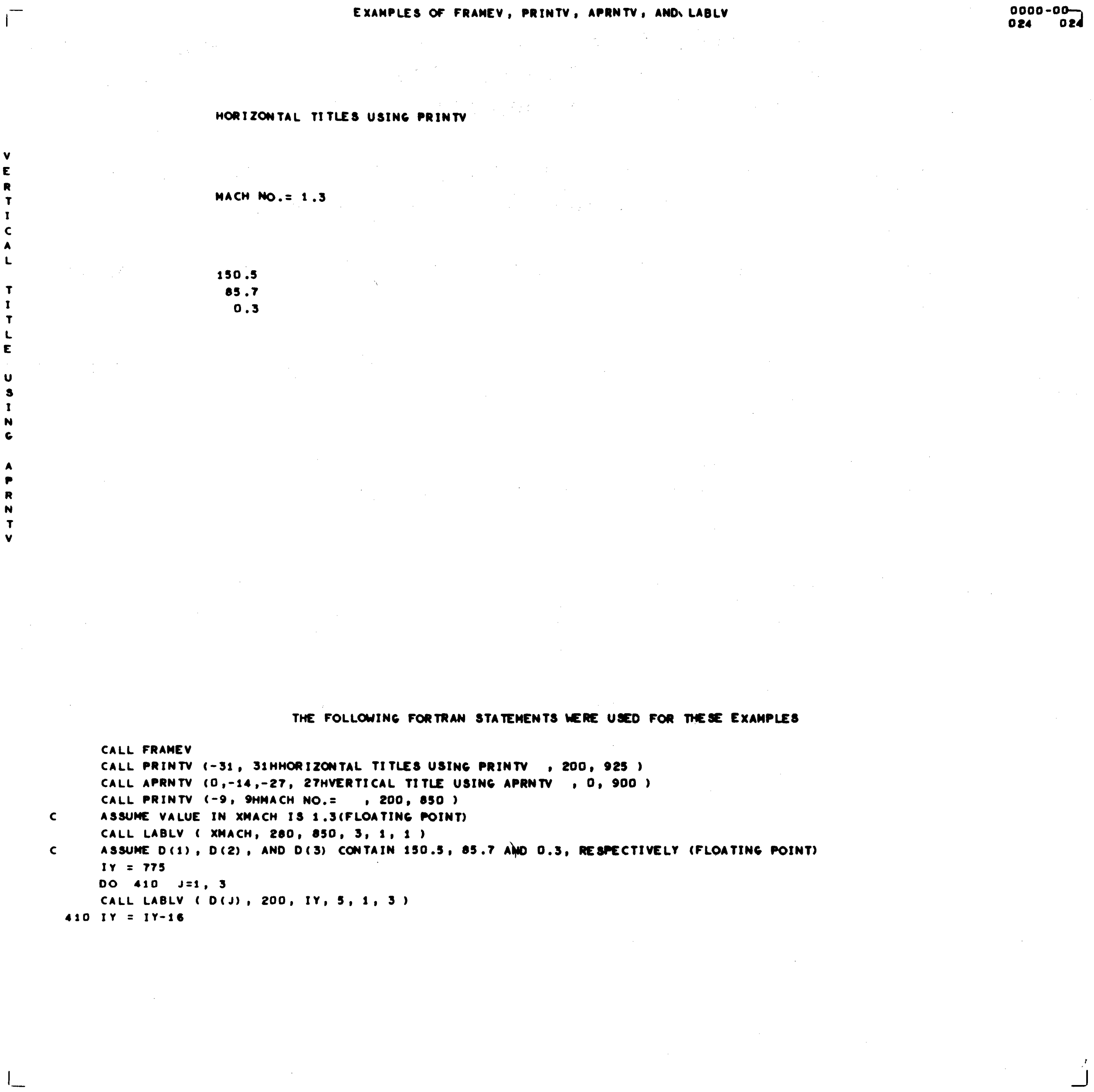
-NCHAR Number of significant figures to be displayed. NCHAR may be less than or equal to 6. The negative sign will result in the use of scientific notation.

NDMAX May be any fixed point quantity. Since the right adjustment of these labels will not be necessary, the value of NDMAX will be ignored.

NCHAR will affect the format in the following ways:

NCHAR	FORMAT
1	Y.x10±YY
2	Y.Yx10±YY
3	Y.YYx10±YY
4	Y.YYYx10±YY
5	Y.YYYYx10±YY
6	Y.YYYYYx10±YY

Since the space required for these labels will be greater than that required for the fixed point format, the programmer should allow NCHAR + 7 CHARACTRON character spaces in width and 1-1/2 spaces in height as a minimum; it may be necessary to allow even more to avoid overlapping other images.



EXAMPLES OF FRAMEV, PRINTV, APRNTV, AND LABLV

0000-00
024 024

HORIZONTAL TITLES USING PRINTV

VERTICAL TITLE USING APRNTV

MACH NO.= 1.3

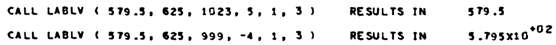
```
150.5
 85.7
  0.3
```

THE FOLLOWING FORTRAN STATEMENTS WERE USED FOR THESE EXAMPLES

```
      CALL FRAMEV
      CALL PRINTV (-31, 31HHORIZONTAL TITLES USING PRINTV  , 200, 925 )
      CALL APRNTV (0,-14,-27, 27HVERTICAL TITLE USING APRNTV  , 0, 900 )
      CALL PRINTV (-9, 9HMACH NO.=   , 200, 850 )
C     ASSUME VALUE IN XMACH IS 1.3(FLOATING POINT)
      CALL LABLV ( XMACH, 280, 850, 3, 1, 1 )
C     ASSUME D(1), D(2), AND D(3) CONTAIN 150.5, 85.7 AND 0.3, RESPECTIVELY (FLOATING POINT)
      IY = 775
      DO  410  J=1, 3
      CALL LABLV ( D(J), 200, IY, 5, 1, 3 )
  410 IY = IY-16
```

Figure 8-2

The following illustrates a scientific and a fixed point label.



CALL LABLV (579.5, 625, 1023, 5, 1, 3)   RESULTS IN   579.5

CALL LABLV (579.5, 625, 999, -4, 1, 3)   RESULTS IN   5.795X10^{+02}

NUMBER CONVERSION: NXV, NYV

For many applications, the placement of printing and labeling is independent of the values of the data being plotted. For this reason, the printing and labeling subprograms have been designed to accept position information specified directly in raster coordinates. In addition, many other S-C 4020 subprograms, particularly the most basic ones, are designed to accept position information specified in raster coordinates. For subprograms of this type, it is possible to position a display relative to data values by making use of the two conversion functions, NXV and NYV, described in the section on scaling.

Section IX

PRINTING

S-C 4020 PRINTING

Printing can be accomplished by the S-C 4020 in any of three ways:(1) simulated typewriting, (2) plotting single characters, (3) forming characters from vectors. The obvious application of placing titles and labels on graphs has already been introduced using the routines PRINTV, APRNTV, and LABLV. PRINTV is the basic routine for using the typewriter simulator. This is the fastest means of getting information out of the computer. In this, mode information is stored as a string of characters, six to the word, and can be output at a maximum rate of 17, 000 characters/second or approximately 7, 000 lines/minute. Of course, this rate is never accomplished due to the necessity for frame advances every sixty-four lines. In APRNTV, the information is output one character per word transferred to the S-C 4020. This is considerably slower than the simulated typewriter mode but allows printing other than in straight horizontal lines.

This section of the manual presents the details of operation of the S-C 4020 typewriter simulator, introduces some additional control routines for use in printing applications, and routines forming characters from vectors. The vector method of forming characters is very flexible and provides for a variety of type fonts in selected sizes and aspect ratios as well as in four rotations. This is the only means at the programmer's disposal for producing rotated characters.

In addition to the basic routines, the routines SCOUTV and VCPS (the latter a separate package from the SCORS-UAIDE 006 system), will be described to aid the programmer in page formatting. The routine SCOUTV is used to output normal FORTRAN printed output on the S-C 4020. This routine utilizes the output routines implemented for the IBM 7090/7094 FORTRAN systems. The VCPS routine provides the capability of preparing printed pages with justified margins and proportional letter spacing. It also provides for mixing fonts at will to achieve special effects. These routines are described in sections X and XI.

SCOUTV is a separate routine distributed with the SCORS system. VCPS is a routine distributed upon request.

TYPEWRITER MODE

The typewriter mode, a "built-in" feature of the S-C 4020, has the following characteristics:

1. CHARACTRON characters are displayed.

2. Once positioned and started in the typewriter mode, the S-C 4020 prints character after character, automatically spacing 8 raster counts to the right as each character is printed.

3. The commands for the typewriter mode (except for those that start and stop typing) are simply the 6-character BCD words to be typed, transmitted without alteration. This mode, therefore, is very economical of computer time and the length of the tape used for words in the typewriter mode is held to a minimum. Since the S-C 4020 can display 6 characters per command, this mode is also economical of S-C 4020 time.

4. When a line of typing reaches the right edge of the frame, (128 characters) an automatic carriage return takes place. Typing halts at this point and resumes at the left edge of the frame, 16 raster counts below the preceding line.

5. If typing reaches the bottom of the frame, a return to the top of the frame occurs automatically. No automatic film advance takes place, so it is possible for typing to overlay previously displayed information.

6. Three of the 64 CHARACTRON characters serve special purposes when they are encountered in the words of the text being typed.

   42 (Decimal)
   52 (Octal) — A carriage return,described in item 4, will occur when this character is encountered.

   10 (Decimal)
   12 (Octal) — "Stop type." Typing mode is stopped, any characters remaining in the word will be ignored. The next word on tape will be interpreted by the S-C 4020 in the graphic mode.

46 (Decimal) "Reset" is executed. This means
56 (Octal) that typing is stopped, the film is advanced, and the bright intensity mode is turned on. Any characters remaining in the word will be ignored. The next word on tape will be interpreted in the graphic mode.

Because 42, 10, and 46 are used for these special purposes, the characters • (plotting dot), ∂ , and ∼ are unavailable in the typewriter mode.

7. Typing may be started at either a specified point or at the "current point". The "current point" is defined as the point specified by the last operation performed; i.e., either the last point plotted or typed, or the origin of the last vector or axis drawn.

SYSTEM SUBPROGRAMS AND THE TYPEWRITER MODE

The subprograms PRINTV and LABLV, and the SCOUTV system, make use of the typewriter mode. Therefore, the output of these subprograms will be influenced by the characteristics of this mode.

Since it was not practical to require the programmer to insert a "stop type" code, other methods of stopping typing are used. PRINTV and LABLV insert the "stop type" code after the specified N characters. The SCOUTV system inserts the code in accordance with the FORMAT statement used.

It should be apparent that an incompatibility would occur if a "stop type" or "reset" code were encountered within the text being typed by PRINTV, LABLV, or SCOUTV. Any information following the special code would not be interpreted as typewriter output.

If PRINTV, LABLV, or SCOUTV printing encounters a "carriage return" character or the right edge of the frame, a carriage return will be made. This would interfere with the normal line advance provided in the SCOUTV system. It can be used to advantage, however, with PRINTV.

Except for the method used to stop typing, PRINTV can be considered a basic typewriter mode routine. It is a suitable lower-level module for other CHARACTRON character typing routines; in fact, it is used in this way by LABLV.

"STOP TYPE" ROUTINES: STOPTV, RESETV

It is remotely possible that some unusual use of the subroutines or of the S-C 4020 could cause the typewriter mode to be in effect when the graphic mode is needed. Two subroutines are provided which ensure that the typewriter mode is no longer on.

CALL STOPTV

Turns off the typewriter mode. This has no effect on the mode if the typewriter mode is already off. However, since this statement writes a word on tape, it should be used with discretion.

GRID1V uses STOPTV to ensure that the plotting mode is in effect at the start of a new grid display.

CALL RESETV or CALL RESETV (NF)

Turns off the typewriter mode, advances the film and establishes the bright intensity mode. The argument agrees with FRAMEV.

SETTING THE "CURRENT POINT": PRINTV, BRITEV, FAINTV

The current point feature of the typewriter mode can be utilized if a special form of PRINTV is employed. Typing may be started at the last point plotted or printed, or at the origin of the last vector or axis line drawn, by giving one of the following statements:

CALL PRINTV (N, BCDTXT)

CALL PRINTV (-N, nH-------)

Except for the use of the current point instead of specified coordinates, this alternate form of PRINTV functions in the usual manner.

If the programmer wants to set a new current point, he may plot or print a blank at a specified position. Another method, which includes both setting the current point and establishing the "Brite" or "Faint" mode, is by one of the following statements:

CALL BRITEV (NX, NY)

CALL FAINTV (NX, NY)

NX, NY The raster coordinates which will be set as the current point. Only a blank will be plotted at NX, NY by either of these statements.

USE OF THE FORM SLIDE: FORMV

In using the S-C 4020 for printing applications, it is desirable to provide the equivalent of using preprinted forms. For this purpose, the S-C 4020 is equipped with a FORM FLASH unit to expose the film with a preprepared slide which can be precisely positioned. The CHARACTRON tube can be aligned to make the image fit the form to be used. This is a manual operation and FORM slides cannot be changed under program control.

The preparation of good FORM slides is rather expensive and therefore should only be considered for well established forms to be used hundreds of times. The FORM slide can occasionally be used for short runs if a special effect is required which is difficult or impossible to achieve by programming the S-C 4020. It must be noted here that a FORM slide should not be considered for use as a grid background for plotting since slight distortions in the image are not harmful if they affect the data and the grid at the same time but would ruin the precision of a graph drawn against a projected grid.

The FORM is projected by the statement:

```
CALL FORMV
```

SPECIAL PRINTING AND LABELING AIDS: HOLLV, BNBCDV

HOLLV and BNBCDV were designed as lower-level modules for the system printing and labeling subprograms. However, they can be called directly, and can sometimes be useful building blocks for the programmer's character printing and labeling subprograms.

ISOLATING ONE CHARACTER FROM A BCD ARRAY: HOLLV

HOLLV selects the Nth BCD character from BCD words that are either contained in a one-dimensional FORTRAN array or given as a Hollerith argument, and stores the numerical equivalent of the selected character in a specified location. PLOTV or VCHARV can then be used to display the character. The call statement is one of the following:

```
CALL HOLLV (N, BCDTXT, NS)

CALL HOLLV (-N, nH------, NS)
```

±N — The position of the desired character within the block of BCD characters, counting from 1. If the block of characters is a

FORTRAN array, N should be positive. If the BCD characters are in a Hollerith argument, the sign of N should be negative.

BCDTXT	A one-dimensional FORTRAN array that contains the BCD characters.
nH----	A Hollerith argument.
NS	The name of a location in which the numerical equivalent of the selected character will be stored. For example, if the Nth BCD character is R, the stored result will be the same as if the statement "NS = 41" had been used.

A practical example of HOLLV usage is illustrated by the way APRNTV uses it. The latter subprogram loops through a sequence of BCD characters, using HOLLV to select a character and PLOTV to print it. APRNTV then advances the print position and returns to the beginning of the loop for the next character.

OBTAINING BCD EQUIVALENT OF A FLOATING POINT NUMBER: BNBCDV

BNBCDV is used by LABLV as a lower-level module to obtain the BCD equivalent of the floating point number displayed in the label. Since it converts a floating point number so as to yield the six most significant digits and the decimal scale, it can be useful also in programs that do not involve the S-C 4020. The call statement is:

CALL BNBCDV (D, BCDWD, NDS)

D	Floating point number to be converted into BCD.
BCDWD	Location in which the 6-character (rounded) BCD equivalent will be stored.
NDS	Location in which the decimal scale (number of digits to the left of the decimal point) will be stored.

The sign of D is ignored, and must be handled separately by the programmer.

Examples

D = 12.345678

The BCD equivalent of 123457 will be placed in BCDWD and the decimal scale of 2 will be stored in NDS.

D = -.01234567

The BCD equivalent of 123457 will be placed in BCDWD, and the decimal scale of -1 in NDS.

PROGRAM-GENERATED CHARACTERS: RITE2V, VCHARV, CHSIZV, RITSTV

THE VECTOR CHARACTER WRITING SUBROUTINE: RITE2V

The RITE2V subprogram provides a means for printing variable-size characters in one of four different orientations. It uses, on a lower level, the vector character-generating subprogram, VCHARV.

The call statement can be either of the following:

```
CALL RITE2V (IX, IY, LIMIT, K, INT, NTOTAL, NTH, BCDTXT, NLAST)
CALL RITE2V (IX, IY, LIMIT, K, INT, NTOTAL, -NTH, nH------, NLAST)
```

IX, IY — The raster coordinates used for positioning the center of the first character to be printed.

LIMIT — A raster position which will determine the right margin. When this limiting position is reached, writing is halted and a new line is started below the starting position of the previous line.

K — A code number which selects the direction of writing and the orientation of characters within the written text.

K = 90 — Characters will be upright, as A

Writing will be horizontal, left to right, as AND

K = 180 — Characters will be rotated left, as A

Writing will be vertical, bottom to top as AND

K = 270 — Characters will be upside down, as Ɐ

Writing will be horizontal, right to left, as ᗡNⱯ

K = 0 (or 360) — Characters will be rotated right, as A

Writing will be vertical, top to bottom, as AND

INT — An integer which controls intensity of the characters by controlling the number of times each vector is displayed. If "overstriking" for a darker character is not required, INT should be 1.

NTOTAL An integer which specifies the number of characters to be printed.

±NTH A character position count that specifies the location within a BCDTXT where writing is to begin. (Usually this is 1, but this argument allows printing to begin with some character inthe BCDTXT other than the first.)

The sign indicates the type of storage used for the next argument. NTH should be positive if the 8th argument is a FORTRAN array. A negative value should be used if it is a Hollerith argument.

BCDTXT The name of an array of BCD information to be written.

nH----- A Hollerith argument. If used, a negative sign must be used with NTH to signal the subroutine that the information is in this form.

NLAST The name of an error location (fixed point). NLAST will be set to zero if all the NTOTAL characters have been displayed. If writing must be stopped because the end of the frame has been reached, NLAST will be set to the character count where writing was stopped.

The standard character size is approximately 12 raster counts in width by 18 counts in height. The standard spacing allowance for adjacent characters is 18 raster counts, and for adjacent rows is 26 raster counts.

The programmer can alter the size of characters and spacing by means of other subroutines, discussed later in this section. He will not need to use them if the standard specifications are satisfactory for his purposes. Figure 9-1 illustrates the use of RITE2V to title a graph on any margin.

VARIATION IN SIZE OF VECTOR CHARACTERS: CHSIZV

The width and height of vector characters can be altered by the subroutine CHSIZV ("Change Size V"). The call statement is:

```
CALL CHSIZV (LVW, LVH)
```

LVW An integer, from 1 to 15, that controls the width of characters.

LVH An integer, from 1 to 9, that controls the height of characters.

Obviously, LVW and LVH need not be equal, but any great disparity will result in the formation of characters that are elongated in one direction.

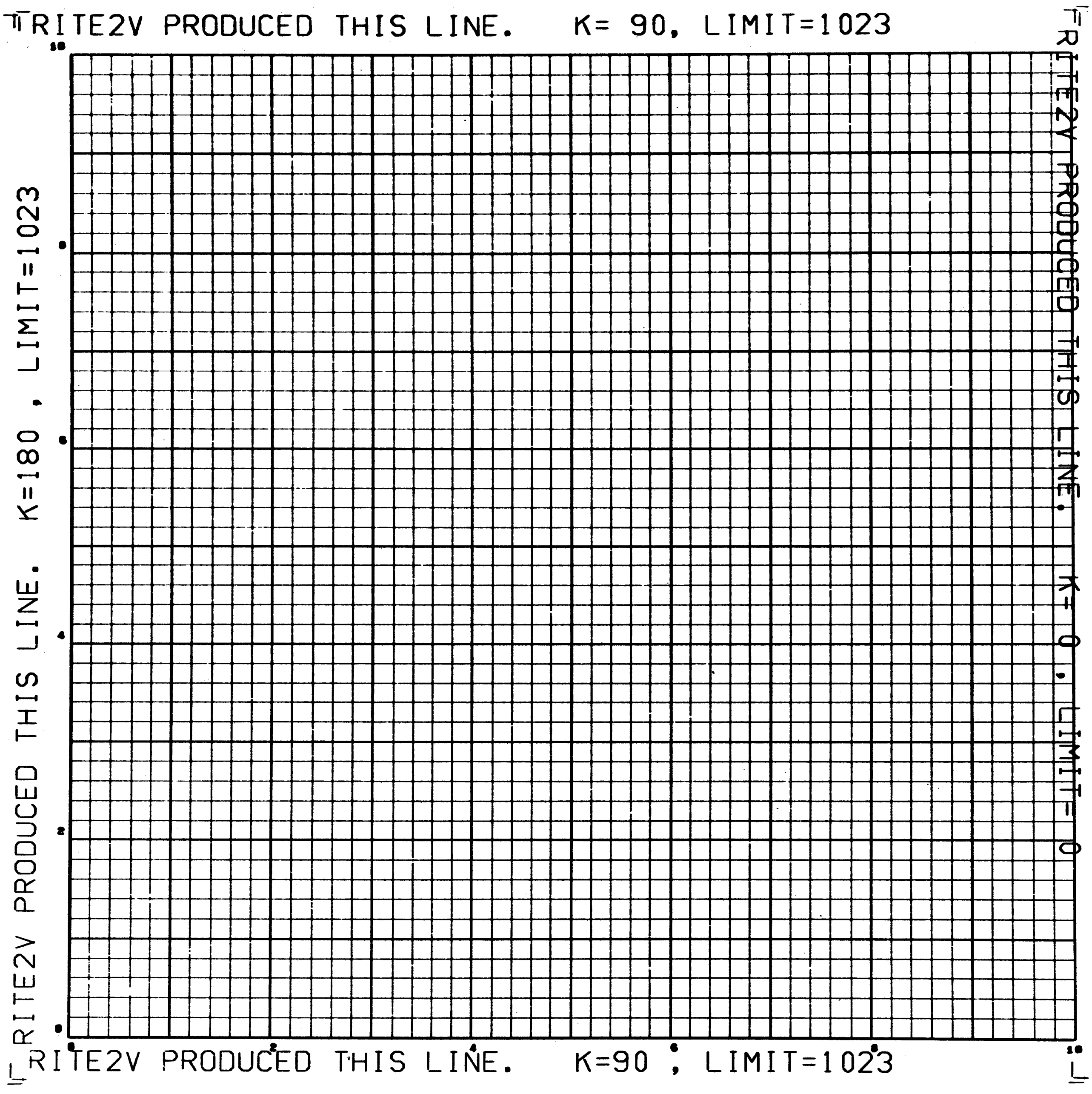

RITE2V PRODUCED THIS LINE. K= 90, LIMIT=1023
RITE2V PRODUCED THIS LINE. K=180 , LIMIT=1023
RITE2V PRODUCED THIS LINE. K= 0 , LIMIT= 0
RITE2V PRODUCED THIS LINE. K=90 , LIMIT=1023
0
2
4
6
8
10

Figure 9-1

For the usual applications of RITE2V, the character size can be considered to be 4*LVW in width, and 6*LVH in height. These dimensions do not include the space required to prevent overlapping of adjacent characters. Examples of different proportions are shown in Figure 9-2. Notice that minimum size characters are not easily read. Adjustment of vector intensity is very critical.

If CHSIZV is never called, the standard values of LVW = 3 and LVH = 3 will be used, resulting in a character size of 12 x 18 raster counts.

When CHSIZV is used to alter the standard character size, it will probably also be necessary to adjust the character spacing. The subroutine RITSTV was designed for this purpose.

CHARACTER SELECTION AND SPACING: RITSTV

RITSTV can be used to alter spacing used by RITE2V for characters and rows. In addition, RITSTV can be used to select an alternate character set to be displayed by RITE2V. The call statement is:

```
CALL RITSTV (ISPACE, IROW, TABL_V)*
```

ISPACE — An integer that specifies the number of raster counts allowed for adjacent characters. This should be somewhat larger than the width of each character, to allow a gap between them.

IROW — An integer that specifies, in raster counts, the depth of adjacent rows. This should be somewhat larger than the height of a character.

TABL_V* — The name of a table of character patterns to be used. The most frequently used table, containing the alphanumeric characters, is named TABL1V. The available tables are described in the section on VECTOR CHARACTER TABLES.

WHEN RITSTV IS USED, THE TABLE NAME GIVEN AS THE THIRD ARGUMENT MUST ALSO BE GIVEN ON AN "F" CARD IN FORTRAN II or "EXTERNAL" IN FORTRAN IV.

NOTE

TABLE names larger than 9V are designated TAB__V instead of TABL_V. References to TAB__V in this section of the manual are marked with an asterisk as a reminder of this fact.

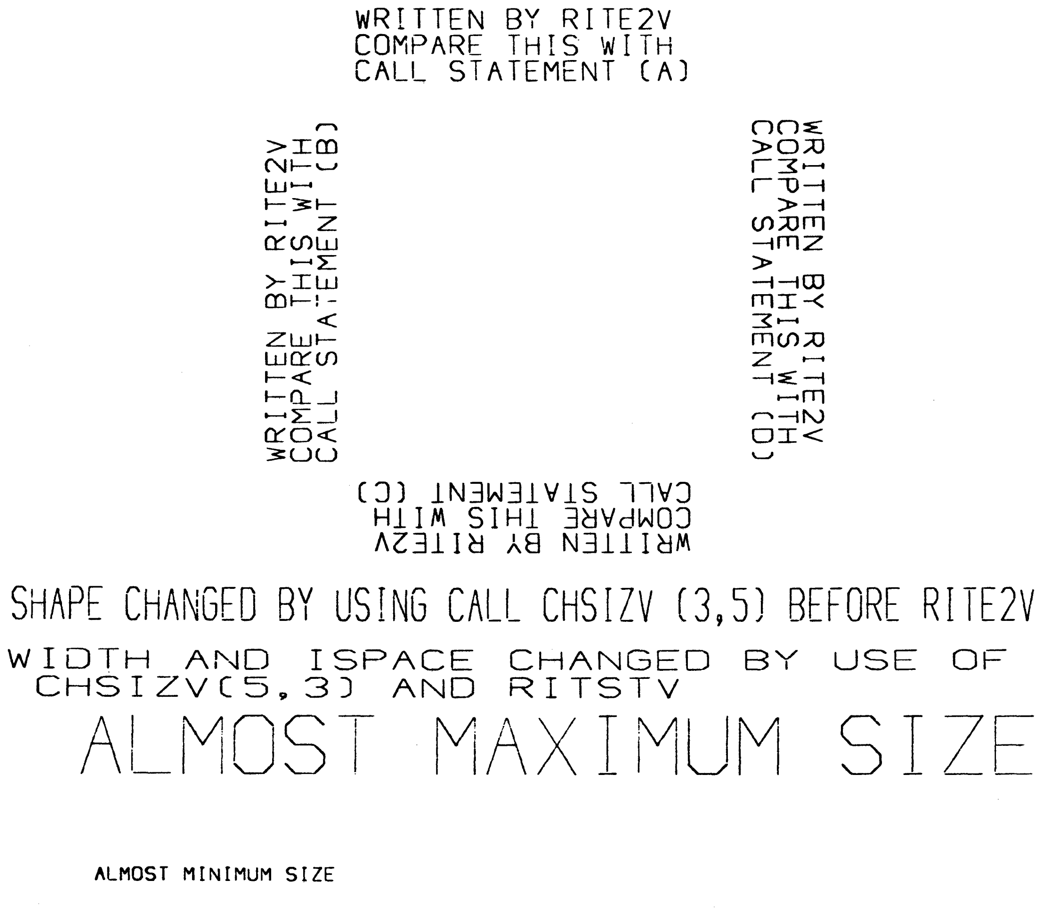

WRITTEN BY RITE2V
COMPARE THIS WITH
CALL STATEMENT (A)
WRITTEN BY RITE2V
COMPARE THIS WITH
CALL STATEMENT (B)
WRITTEN BY RITE2V
COMPARE THIS WITH
CALL STATEMENT (D)
WRITTEN BY RITE2V
COMPARE THIS WITH
CALL STATEMENT (C)
SHAPE CHANGED BY USING CALL CHSIZV (3,5) BEFORE RITE2V
WIDTH AND ISPACE CHANGED BY USE OF
CHSIZV(5,3) AND RITSTV
ALMOST MAXIMUM SIZE
ALMOST MINIMUM SIZE

Figure 9-2

If RITSTV is called, all three arguments must be specifically stated, even though not all of them are changed, and the table name must appear on an "F" card or an "EXTERNAL" definition. The vector character tables are illustrated in Figures 9-3 and 9-4.

If RITSTV is never called, RITE2V will use the standard values of ISPACE and IROW (18 and 26, respectively), and the standard table (TABL1V).

The following formulas are a general guide for the assignment of minimum values of ISPACE and IROW.

ISPACE $\approx$ 5*LVW + 3

IROW $\approx$ 7*LVH + 5

RETRIEVAL OF TERMINAL COORDINATES: RITXYV

At the time RITE2V terminates, the coordinates for the next character are set up. RITXYV retrieves these coordinates so that they can be used to determine the correct position of a special character to be inserted into a text written by RITE2V. The call statement is:

CALL RITXYV (LX, LY)

LX, LY Names of the locations in which RITXYV will store the raster coordinates (IX, IY) that RITE2V has computed as the location of the next character.

DISPLAY AN INDIVIDUAL VECTOR CHARACTER: VCHARV

VCHARV is used as a lower-level subroutine by RITE2V to generate each vector character. When a single character is to be displayed, a direct call to VCHARV by the programmer will often be more convenient than a call to RITE2V, especially if the character to be displayed is a non-Hollerith or a special character.

The call statement is:

CALL VCHARV (K, INT, ILX, ILY, NS, TABL_V)*

K Code number which selects the orientation of the character:

K=90 Character will be upright, as A

K=180 Character will be rotated left, as ∢

TABL1V,TABL2V,TABL3V --NOTE USE OF VCHARV TO DISPLAY THE CHARACTERS

0000-00
047 04

TABL1V

	0	1	2	3	4	5	6	7	8	9
00	0	1	2	3	4	5	6	7	8	9
10	∂	=	"	·	δ	α	+	A	B	C
20	D	E	F	G	H	I	π	.	)	β
30	±	?	–	J	K	L	M	N	O	P
40	Q	R	•	$	*	γ	~	∂		/
50	S	T	U	V	W	X	Y	Z	°	,
60	(	∫	Σ	□	&	%	¢	[	]	{
70	}	>	<	≥	≤	∞				

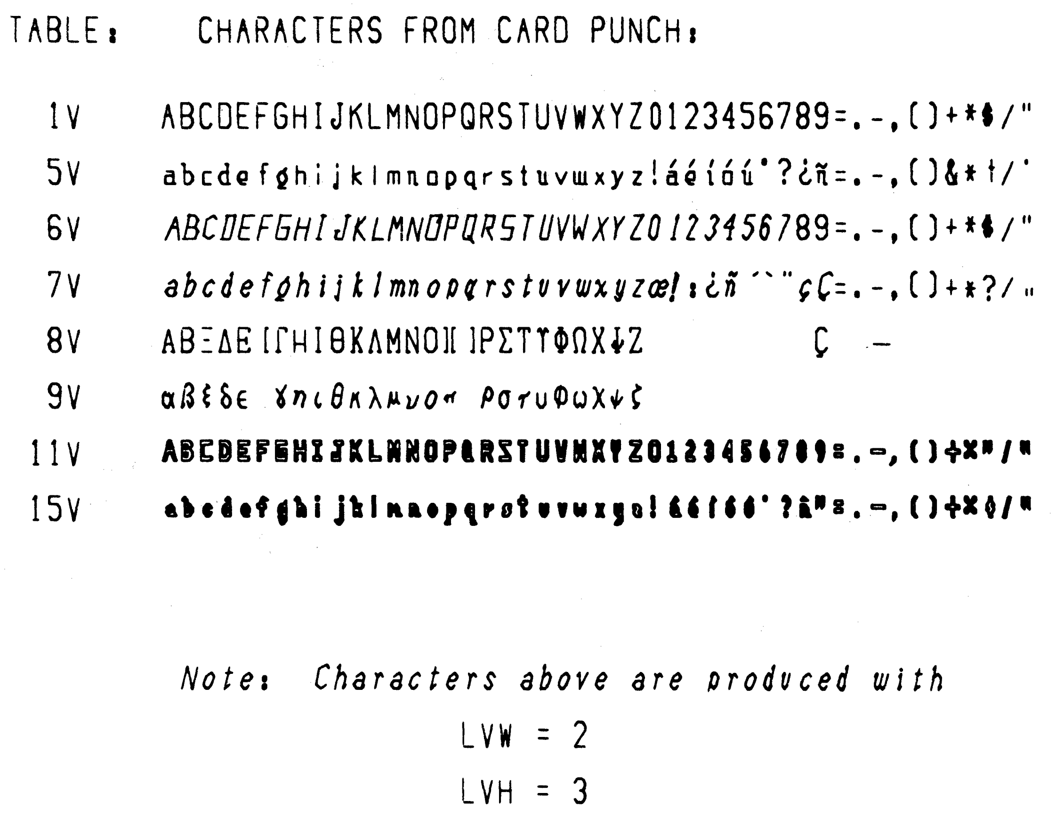
TABL2V

	0	1	2	3	4	5	6	7	8	9
00	α	β	γ	δ	ϵ	ζ	η	θ	ι	κ
10	λ	μ	ν	ξ	ο	π	ρ	σ	τ	ύ
20	φ	χ	ψ	ω						

TABL3V

	0	1	2	3	4	5	6	7	8	9
00	A	B	Γ	Δ	E	Z	H	Θ	I	K
10	Λ	M	N	Ξ	O	Π	P	Σ	T	Υ
20	Φ	X	Ψ	Ω						

Figure 9-3



```
TABLE:    CHARACTERS FROM CARD PUNCH:

  1V      ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789=.-,()+*$/"
  5V      abcdefghijklmnopqrstuvwxyz!áéíóú˙?¿ñ=.-,()&*†/'
  6V      ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789=.-,()+*$/"
  7V      abcdefghijklmnopqrstuvwxyzæ!:¿ñ ´`"çÇ=.-,()+*?/„
  8V      ABΞΔE[ΓHIΘKΛMNOΠ]PΣTΥΦΩXΨZ           Ç  –
  9V      αβξδε γηιθκλμνοπ ρστυφωχψζ
 11V      ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789=.-,()÷×"/"
 15V      abcdefghi jklmnopqrstuvwxyz! áéíóú˙?â"=.-,()÷×◊/"

          Note:  Characters above are produced with
                       LVW = 2
                       LVH = 3
```

Figure 9-4

K=270 Character will be upside down, as ∀

K = 0 (or 360) Character will be rotated right, as ⊳

INT An integer which controls the intensity of the character by controlling the number of times each vector is repeated. For normal intensity this should be 1, but may be greater if a darker character is desired.

ILX, ILY Integers which specify the raster coordinates to be used to position the character. This position will be used as the lower left corner of the character pattern. (Note that this is different from the center position specified for RITE2V.)

NS An integer which selects the character to be generated from a table of patterns. Selected by the choice of TABL_V.*

TABL_V* The name of a table of character patterns to be used. The name of the table containing the most frequently used characters is TABL1V.

WHEN VCHARV IS CALLED DIRECTLY, THE TABLE NAME GIVEN MUST ALSO APPEAR ON AN "F" CARD OR ON "EXTERNAL" DEFINITION.

RITE2V uses the specified coordinates as the center of the first character to be displayed, but VCHARV uses the specified coordinates as the lower left corner of the character. Thus, it may be necessary to translate the coordinates when direct calls to VCHARV are used to insert characters into a line of information displayed by RITE2V. Without going into great detail on the structure of vector characters at this point, the formulas given below show the relation between the position arguments for VCHARV and for RITE2V. In these formulas, LVW and LVH are the arguments used in CHSIZV. If CHSIZV has not been used to alter these quantities, LVW = LVH = 3.

FOR K = 90	ILX = IX - 2*LVW
	ILY = IY - 3*LVH
K = 180	ILX = IX + 3*LVH
	ILY = IY - 2*LVW
K = 270	ILX = IX + 2*LVW
	ILY = IY + 3*LVH

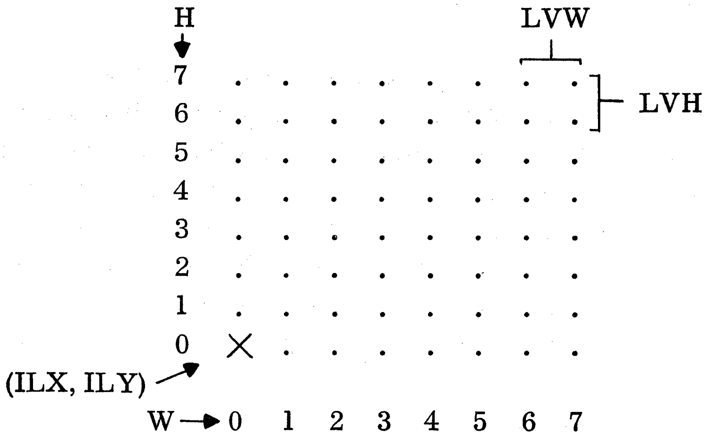
K = 0	ILX = IX - 3*LVH
(or 360)	ILY = IY + 2*LVW

VECTOR CHARACTER CONSTRUCTION

In a preceding section, the tables of vector characters which are available in the system were presented. This section explains how vector characters are generated and how the programmer can create additional characters to be displayed by VCHARV and RITE2V.

Vector Character Formation

Given the proper specifications, VCHARV will generate any design that can be constructed by joining points on an 8 x 8 lattice with straight lines. A coordinate system in width (W) and height (H) is used to refer to positions on the lattice. Patterns are specified by giving coordinates of the positions on the lattice that are to be connected by vectors. As shown on the diagram, the W and H coordinates each can have values from 0 to 7.



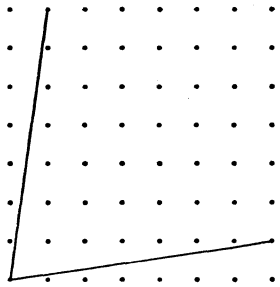
ILX, ILY are the coordinates used by VCHARV to position the character. LVW, LVH represent the distances (expressed in raster counts) from one lattice position to an adjacent position.

CHARACTER SIZE

The size of a character (or other design) depends upon the amount of space between adjacent positions on the lattice. The number of raster counts between positions may be specified by calling the subprogram CHSIZV. The arguments LVW and LVH indicate the desired number of raster counts between positions in width and height, respectively. If CHSIZV is never called, the values LVW = LVH = 3 raster counts will be used.

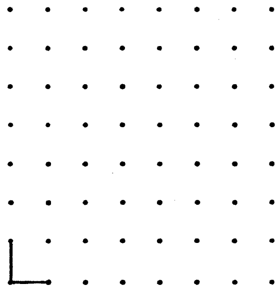
Although it is possible to assign a value as small as 1 to LVW and/or LVH, this is not practical for the majority of characters. The upper limit depends upon the particular character, and upon the restriction that no vectors in the pattern may exceed the resultant of 63 counts in the direction W, and 63 counts in the direction H. For example, if a particular pattern requires joining a position on the bottom row of the lattice with one on the top row, the maximum permissible value of LVH is 9, since 9 (raster counts) multiplied by 7 (lattice intervals) = 63 counts.

To draw these two vectors:



The maximum permissible LVW, LVH = 9 counts

To draw these two vectors:



The maximum permissible LVW, LVH = 63 counts

CHARACTER ORIENTATION AND PLACEMENT: VCHARV

VCHARV permits a selection of four possible orientations. It rotates a character into the selected orientation by the way in which it applies the increments LVW and LVH.

The following table shows the direction of application of LVW and LVH for the four orientations.

Orientation Selection	Directions in Which Increments are Applied	
K	LVW	LVH
90 (Upright)	Positive in X	Positive in Y
180	Positive in Y	Negative in X
270	Negative in X	Negative in Y
0 (or 360)	Negative in Y	Positive in X

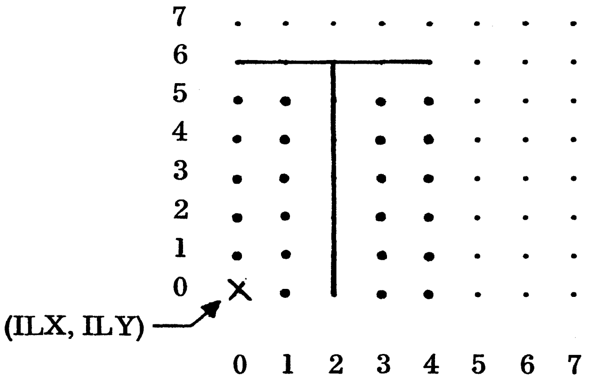
To prevent characters from overlapping the edge of the frame, the orientation and the specifications of the position coordinates (ILX, ILY) must be considered. The following table shows the restrictions upon ILX, ILY for each of the four orientations. In the table, N is one less than the number of lattice positions in width used by the character, and M is one less than the number, in height.

K	ILX	ILY
90 (Upright)	$0 \le ILX \le 1023 - N*LVW$	$0 \le ILY \le 1023 - M*LVH$
180	$M*LVH \le ILX \le 1023$	$0 \le ILY \le 1023 - N*LVW$
270	$N*LVW \le ILX \le 1023$	$M*LVH \le ILY \le 1023$
0 (or 360)	$0 \le ILX \le 1023 - M*LVH$	$N*LVW \le ILY \le 1023$

PATTERN SPECIFICATION

A pattern specifies the points on the lattice to be joined by vectors. The pattern consists of a sequence of lattice coordinates arranged in the order W1, W2, H1, H2, W3, W4, H3, H4, . . ., etc., where (W1, H1) and (W2, H2) are the coordinates of the end points of the first vector in the character, (W3, H3), and (W4, H4) are coordinates of the end points of the second vector, etc. The end of a pattern is signaled by the presence of at least 12 consecutive zero bits.

Since values of W and H will not be larger than 7, the values can be packed into computer words, alloting 3 bits per value. As many words as are needed to include all coordinates may be used. The character T appears on the lattice as follows:

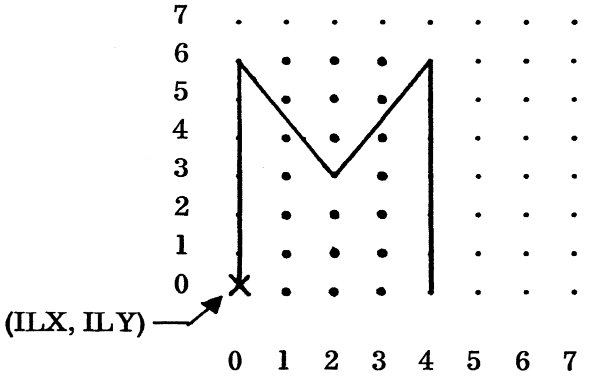


The instructions for this pattern could be read in as the single octal word:

PATRN1

0	4	6	6	2	2	0	6	0	0	0	0
W1	W2	H1	H2	W3	W4	H3	H4	(End of pattern)			

The pattern for M requires more than one word. It could be read in as an octal array, with a dimension of 2, in order to appear in storage in proper FORTRAN array order.



The instructions for this pattern, in octal are:

PATRN2(1)

0	0	0	6	0	2	6	3	2	4	3	6
W1	W2	H1	H2	W3	W4	H3	H4	W5	W6	H5	H6

PATRN2(2)

4	4	0	6	0	0	0	0	0	0	0	0
W7	W8	H7	H8	(End of Pattern)							

SINGLE PATTERN FOR VCHARV

A special form of VCHARV allows a single pattern to be furnished as a FORTRAN array. The following call statement may be used:

```
    CALL VCHARV (K, NT, ILX, ILY, 0, A, 0)
```

A — A one-dimensional array in which the pattern must be stored. The fifth and seventh arguments must be zero. The remainder of the arguments are as described earlier for VCHARV.

TABLE STRUCTURE

If the programmer has need for a number of special characters not in the system tables, he can build a table, in the FAP language, similar to the tables in Figure 9-4. As with these tables, he must use an "F" card carrying the table name.

Since the number of words in each pattern varies, a list of entries is employed in the search for a specific character pattern. The location TABL_V + 1 must contain, in its address, the location of the list of entries. Each table is compiled as if it were a FAP subprogram for FORTRAN use. Storage is FORTRAN-array type, i.e., descending storage for the table and for each individual pattern within it. Note that each individual character ends with 12 zeros. For the statement "VFD N/M, N/M, etc, N is the number of bits allotted to M, which in turn is the quantity to be used for W (or H). (The patterns illustrated are the numeric characters 0 through 5.)

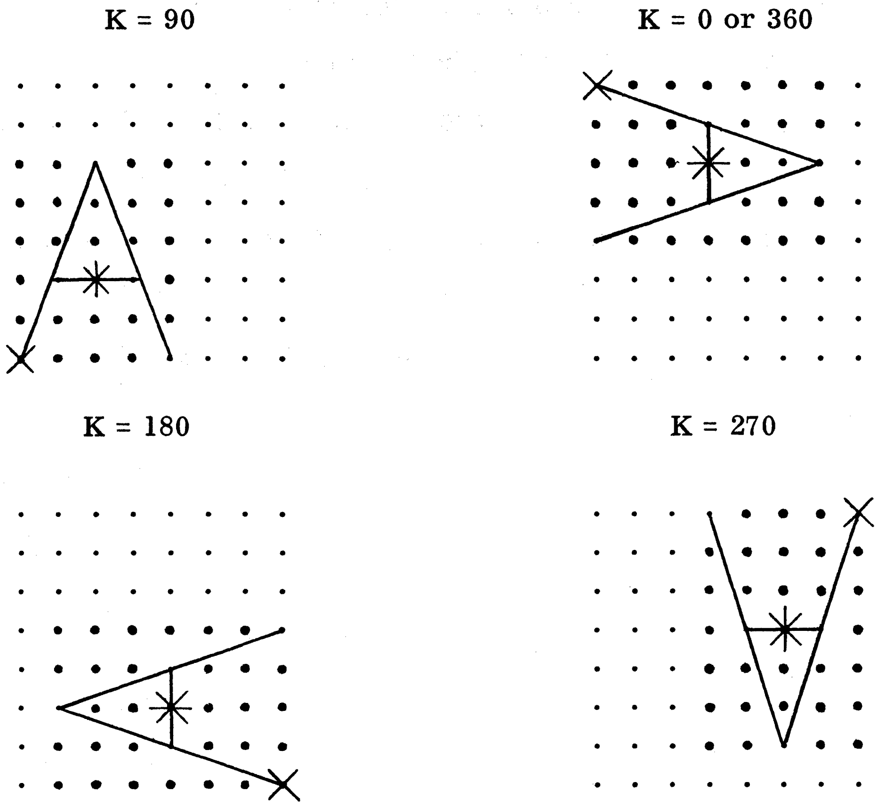
```
*           FAP
            COUNT     25
            ENTRY     TABL9V
            VFD       9/12,9/14,18/0
ENTR9V      VFD       9/0,9/3,9/5,9/8
            VFD       3/0,3/0,3/4,3/6,3/0,3/4,3/6,3/6,12/0
            VFD       3/4,3/4,3/1,3/3,3/4,3/3,3/3,3/4,3/3,3/0,3/4,3/4
```

```
N5        VFD     3/0,3/1,3/1,3/0,3/1,3/3,6/0,3/3,3/4,3/0,3/1
          VFD     3/0,3/4,3/2,3/2,24/0
N4        VFD     3/2,3/4,6/0,3/3,3/3,3/0,3/6,3/3,3/0,3/6,3/2
          VFD     3/3,3/1,3/6,3/6,3/1,3/0,3/6,3/5,12/0
          VFD     3/3,3/4,3/3,3/4,3/4,3/4,3/4,3/5,3/4,3/3,3/5,3/6
          VFD     3/4,3/4,3/1,3/2,3/4,3/3,3/2,3/3,3/3,3/2,3/3,3/3
N3        VFD     3/0,3/1,3/1,3/0,3/1,3/3,6/0,3/3,3/4,3/0,3/1
          PZE     0
          VFD     3/4,3/4,3/5,3/4,3/4,3/0,3/4,3/0,3/0,3/4,6/0
N2        VFD     3/0,3/1,3/5,3/6,3/1,3/3,3/6,3/6,3/3,3/4,3/6,3/5
          PZE     0
N1        VFD     3/1,3/3,6/0,3/2,3/2,3/0,3/6,3/1,3/2,3/5,3/6
          VFD     3/3,3/2,3/0,3/0,3/2,3/1,3/0,3/1,12/0
          VFD     3/3,3/4,3/6,3/5,3/4,3/4,3/5,3/1,3/4,3/3,3/1,3/0
TABL9V    VFD     3/1,3/1,3/1,3/5,3/1,3/2,3/5,3/6,3/2,3/3,3/6,3/6
          PZE     ENTR9V
          END
```

CHARACTER ORIENTATION AND PLACEMENT: RITE2V

If only one character is to be displayed, VCHARV is suitable, but for several characters, RITE2V can be employed more easily. Since RITE2V uses VCHARV to generate characters, much of the preceding discussion applies. However, unlike VCHARV, which uses the lower left corner of the lattice as its position reference point, RITE2V uses the center of the 5 x 7 portion of the 8 x 8 lattice.

The following diagrams show the relation between the character lattice as it is used in RITE2V and in VCHARV. The 5 x 7 lattice is shown superimposed upon the maximum 8 x 8 lattice. The asterisk indicates the reference point (IX, IY) as used by RITE2V, while the X indicates the reference point (ILX, ILY) as used by VCHARV.



K = 90

K = 0 or 360

K = 180

K = 270

Since the center is used as the character reference point by RITE2V, and since a 5 x 7 lattice is normally used, the restrictions upon the position coordinates differ slightly from those shown in the table for VCHARV. The restrictions are:

K	IX	IY
90	2*LVW ≤ IX ≤ 1023 - 2*LVW	3*LVH ≤ IY ≤ 1023 - 3*LVH
180	3*LVH ≤ IX ≤ 1023 - 3*LVH	2*LVW ≤ IY ≤ 1023 - 2*LVW
270	2*LVW ≤ IX ≤ 1023 - 2*LVW	3*LVH ≤ IY ≤ 1023 - 3*LVH
0 (or 360)	3*LVH ≤ IX ≤ 1023 - 3*LVH	2*LVW ≤ IY ≤ 1023 - 2*LVW

(One more LVH increment must be considered for the $ sign.)

The character space size and row size to be used by RITE2V depend upon the character size and the space required to prevent overlapping of characters. Since the characters in the tables normally used by RITE2V are based on a 5 x 7 lattice, the character size is, usually, 4*LVW by 6*LVH. (For the $ sign, it is 4*LVW by 7*LVH.) Also, since LVW = LVH = 3, the character size is 12 by 18, unless changed by CHSIZV. The standard quantities ISPACE and IROW used by RITE2V are 18 and 24, respectively, allowing sufficient space to prevent overlapping of characters.

The description of RITSTV shows how the programmer may redefine ISPACE, IROW, and also may name a table other than TABL1V which is normally used. The table named may be one of the alternate ones provided by the system or one of the programmer's own design.

SINGLE PATTERN FOR RITE2V

An alternate form of RITSTV allows the programmer to furnish a single character pattern in an array. The call statement is:

```
CALL RITSTV (ISPACE, IROW, A, 0)
```

A   A FORTRAN array in which the programmer has stored one character pattern. "A" must not be named on an "F" card.

The last argument must be zero, to indicate to the subprogram that A is not a table. ISPACE and IROW are as defined as before.

VCHARV can also be employed to generate a single character from a pattern supplied as an array. Although VCHARV is more economical of machine time, the programmer may find it less awkward to use RITSTV to insert a character into printed information being written by RITE2V.

Section X

SPECIAL PRINTING

LINE PRINTER SIMULATION ON THE S-C 4020: SCOUTV

Line printing can be simulated on the S-C 4020 by means of the SCOUTV system. Whenever the statement

CALL SCOUTV

appears in the program, any WRITE OUTPUT TAPE N statement, where N is the currently designated S-C 4020 output tape, will be recognized as a command to produce printed output on the S-C 4020. Any other number for N will produce standard output for the regular system. The output will be determined by the list of the WRITE OUTPUT TAPE statement and the associated FORMAT statement; S-C 4020 output will be returned instead of output from the printer.

One natural application of the SCOUTV mode is the production of printed output on film for long-term storage, making it unnecessary to keep large stacks of printed paper. The use of SCOUTV as a printer simulator is discussed on the following pages. Another application is the combination of plotting and printing on one film frame.

BASIC SPECIFICATIONS

The following list explains most of the differences and likenesses that prevail between ordinary printing and SCOUTV printing, when there have been no programmed alterations to change the normal SCOUTV output.

1. The camera must have been selected by a CALL CAMRAV (N) statement.

2. A CALL SCOUTV statement is necessary to make printer simulation effective. The SCOUTV mode will not carry over from one Chain link to the next; hence a new CALL SCOUTV statement is necessary in each link.

3. The first character of each BCD unit record is recognized as a carriage control character, as follows:

+	No space before printing.
Blank	Single space before printing.
0	Double space before printing.
-	Triple space before printing.
1	Frame advance (equivalent to sheet eject).
4	Four spaces before printing. (This is not an exact equivalent of standard printer output.)
8	Skip to the bottom line on the frame.

An incorrect carriage control character is much more critical on the S-C 4020 than on a standard printer. The S-C 4020 operator cannot override the action of these characters. Non-standard characters, or loops involving "1" or "8," can cause great waste of film.

4. The position of print line 1 will be on the Y raster coordinate that is 23 counts from the top of the frame.

5. In the basic SCOUTV mode, it is usually desirable to advance the film as you would eject a sheet of paper, by a carriage control character "1." The first information to be printed on the new frame will then start on print line 1. This is also true if the frame is changed as a result of the line count being exceeded.

6. If the frame is changed by a CALL FRAMEV, the first carriage control character of the first FORMAT statement will affect the position of the information to be displayed in one of the following ways:

   A blank will cause printing to start on print line 1.

   A "0" will cause printing to start on print line 2 (46 counts from the top).

   A "-" will cause printing to start on print line 3. Similarly, a "4" or "8" will cause printing to start on the fourth line or at the bottom of the page, respectively.

   A "1" will cause the frame to eject, leaving a wasted frame.

   Usually "+" will cause printing on line 0, right at the top of the page. This feature is not recommended.

7. Although the S-C 4020 printing will correspond line-for-line and page-for-page with printer output, the proportions of length-to-width will not be the same. This is illustrated by the SCOUTV examples on subsequent pages.

8. The spacing between lines is not governed by the typewriter mode; each new line will be started 23 raster counts below the previous line, which is more generous than the spacing provided by the typewriter mode.

9. There are 44 lines permitted per page. An attempt to write beyond the 44th line will cause the film to be advanced. Printing will continue on the new frame; the first line will be 23 raster counts from the top. The carriage control character for that line will have no effect, just as in printed output.

Figures 10-1 and 10-2 show an exercise in the use of SCOUTV printer simulation, demonstrating some of the carriage controls, Hollerith character printing, integer, fixed, and floating point output, etc. Part of the exercise is a line count, including a count of lines skipped. Note that when a line count of 44 was exceeded, a new frame was begun.

SCOUTV VARIATIONS

The basic SCOUTV system treats a frame as if it were segmented into 44 print lines, starting on the 23rd raster count from the top of the frame and ending on the 1012th raster count from the top. Each print line is 23 raster counts deep. Printing begins on the first print line, and each line starts on the 24th raster count from the left of the frame.

The following discussion explains how these basic specifications can be altered in a number of ways.

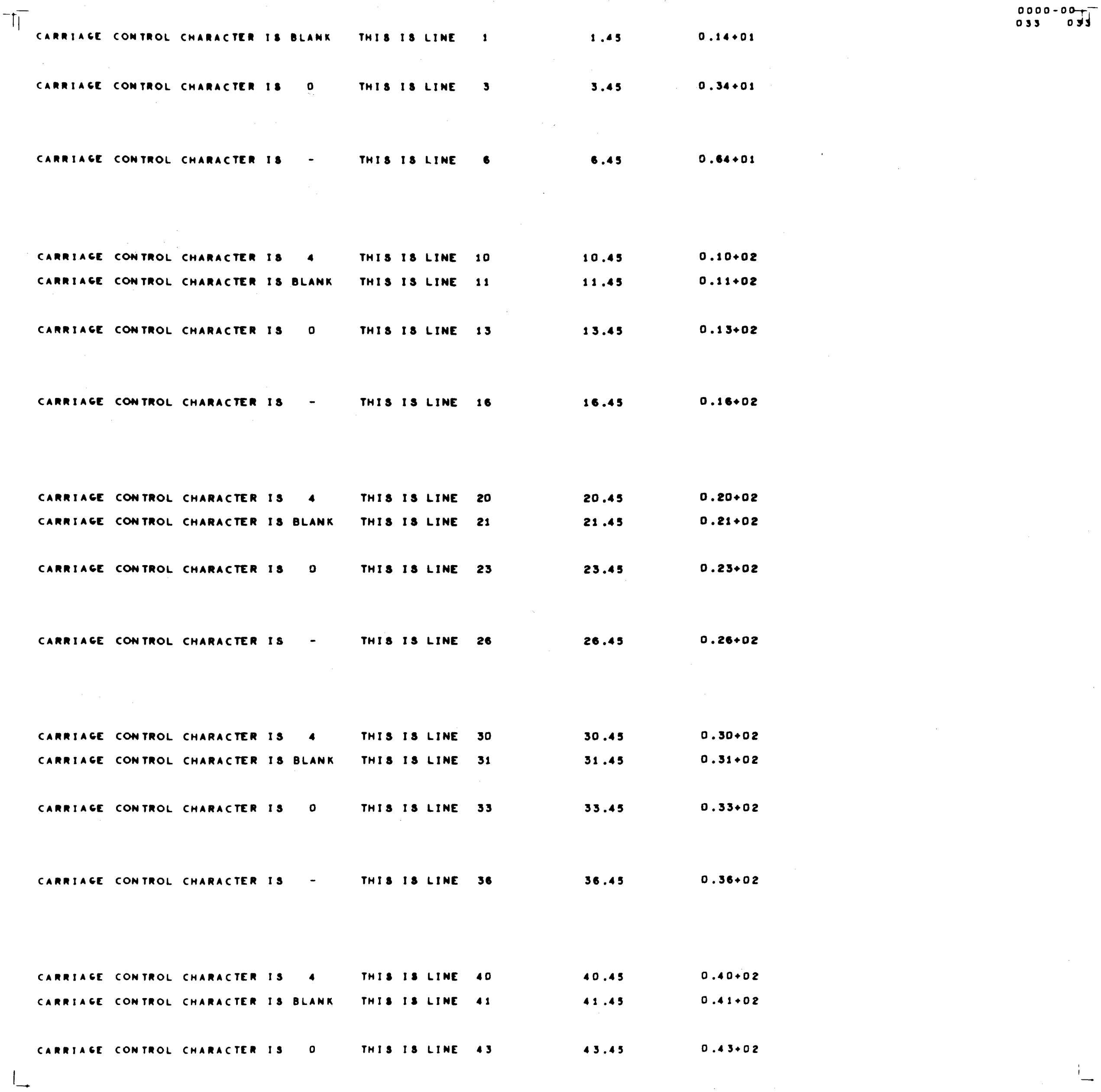
Altering the Starting Line

Printing may be started on a print line other than the first by using the statement:

```
CALL SCOUTV (LINE1)
```

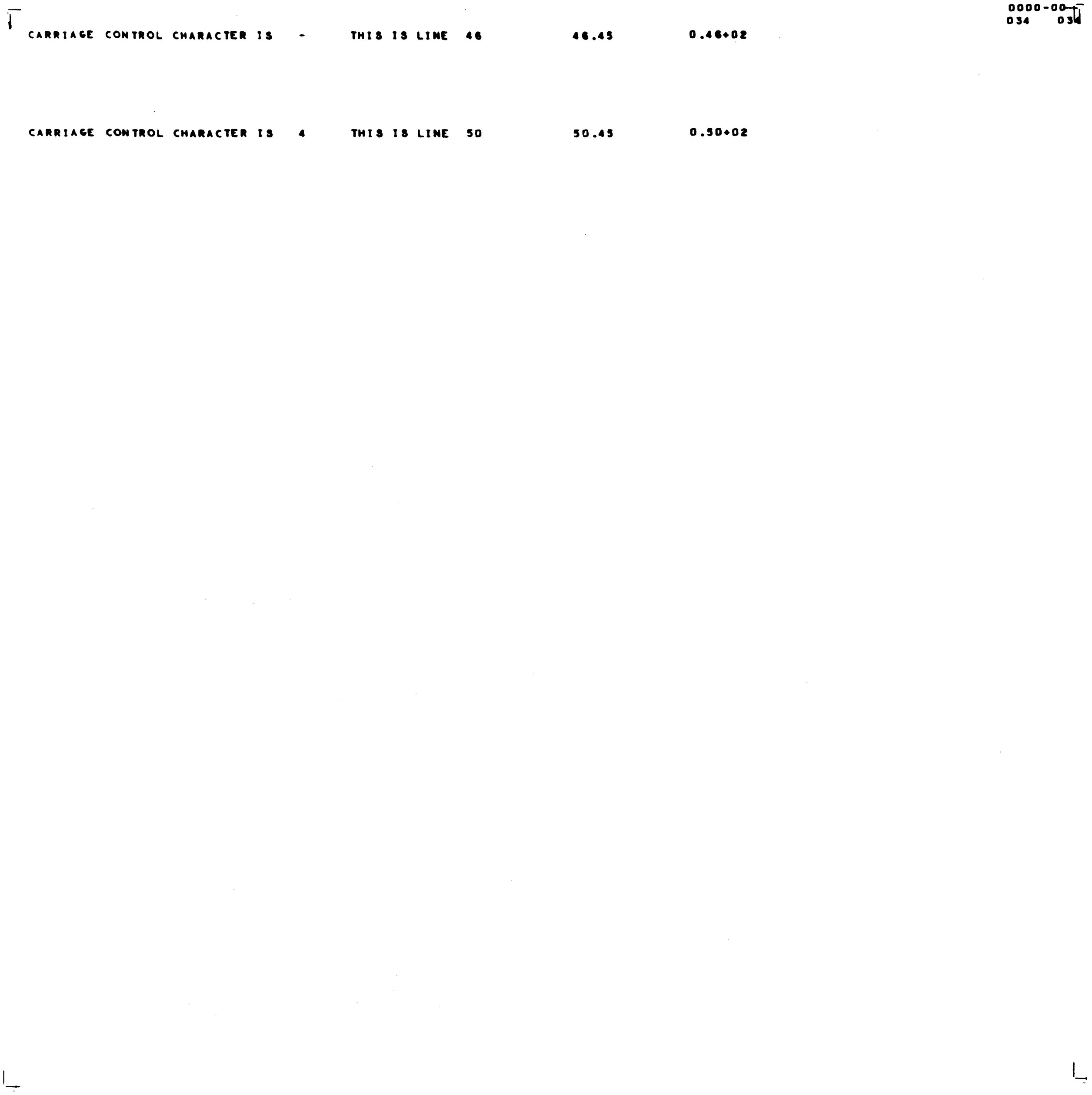
LINE1 An integer which specifies the print line on which the next printing in the SCOUTV mode will begin.

For example, if the statement CALL SCOUTV (5) is used, the next WRITE OUTPUT TAPE statement will place printing on the 5th print line (the raster position that is 5*23 counts from the top of the frame). The carriage control character (even the "1" for "eject") will be ignored for that line.



```
                                                                                                                     0000-00
                                                                                                                     033   033
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE  1          1.45          0.14+01

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE  3          3.45          0.34+01


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE  6          6.45          0.64+01



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE 10         10.45          0.10+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE 11         11.45          0.11+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE 13         13.45          0.13+02


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE 16         16.45          0.16+02



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE 20         20.45          0.20+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE 21         21.45          0.21+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE 23         23.45          0.23+02


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE 26         26.45          0.26+02



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE 30         30.45          0.30+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE 31         31.45          0.31+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE 33         33.45          0.33+02


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE 36         36.45          0.36+02



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE 40         40.45          0.40+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE 41         41.45          0.41+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE 43         43.45          0.43+02
```

Figure 10-1



```
                                                                                                                      0000-00
                                                                                                                      034   034
CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE  46          46.45          0.46+02


CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE  50          50.45          0.50+02
```

Figure 10-2

For LINE1 to be in force, this statement must be executed in advance of the associated WRITE OUTPUT TAPE statement.

If no other alterations are made, LINE1 may have any value from 1 to 44, and the maximum number of lines of printing will be (45 - LINE1).

Since LINE1 will be effective only for one frame, CALL SCOUTV (LINE1) should be restated if printing is to start on the named LINE1 on the next frame.

Altering the Starting Line and Line Spacing Increments

Further alteration of the basic SCOUTV specifications may be made by using the statement:

```
CALL SCOUTV (LINE1, INC)
```

LINE1 — An integer which specifies the print line on which the next printing in the SCOUTV mode will begin. The actual position of the first line on the frame will depend upon the value of the second argument, INC.

INC — An increment of line spacing. The frame will be considered to be segmented into print lines that are INC raster counts deep. The maximum number of print lines will be 1012/INC, since the basic limit for the bottom line is 1012 raster counts below the top of the frame.

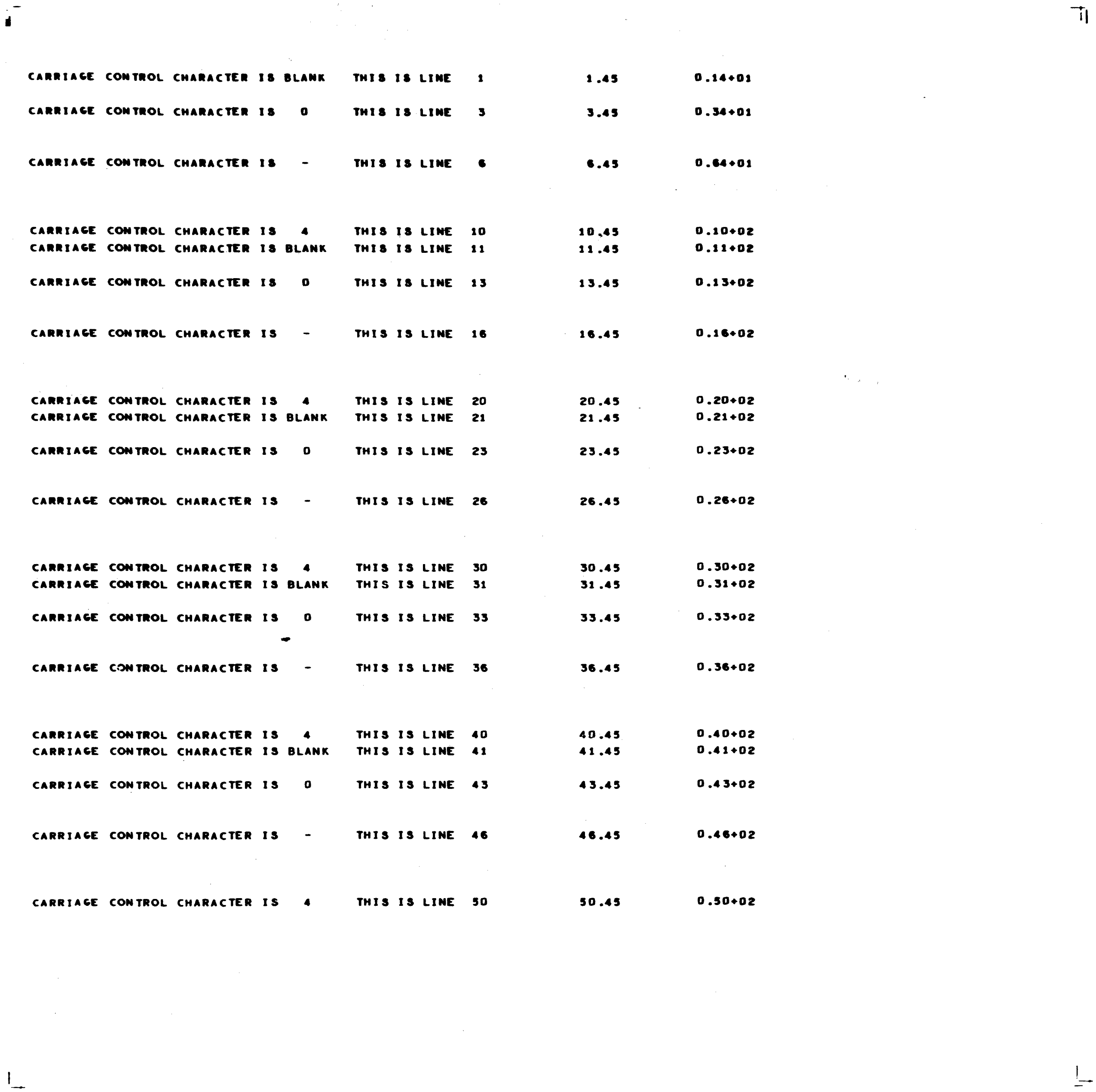
The standard LINE1 and INC for SCOUTV are reset by a CALL SCOUTV (1, 23).

The carriage control character for the first line of information to be printed following this call statement will be ignored. (Note that this also applies to the character for "eject.") Therefore, printing will start on the raster coordinate that is LINE1*INC raster counts from the top of the frame.

INC will remain in effect for subsequent SCOUTV mode printing in the program until altered by another CALL SCOUTV (LINE1, INC).

For LINE1 or INC to be effective, the CALL SCOUTV (LINE1, INC), statement must be executed before the associated WRITE OUTPUT TAPE statement.

Figure 10-3 illustrates the use of SCOUTV with arguments that alter the basic specifications. Note the use of INC = 16, which caused the lines to be closer than normal, enabling the same 50 lines shown on Figures 10-1 and 10-2 to be printed on one frame. Since LINE1 = 4 and INC = 16, the first line displayed is actually 4*16 raster counts below the top of the frame.



```
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE  1          1.45         0.14+01

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE  3          3.45         0.34+01


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE  6          6.45         0.64+01



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE  10        10.45         0.10+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE  11        11.45         0.11+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE  13        13.45         0.13+02


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE  16        16.45         0.16+02



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE  20        20.45         0.20+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE  21        21.45         0.21+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE  23        23.45         0.23+02


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE  26        26.45         0.26+02



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE  30        30.45         0.30+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE  31        31.45         0.31+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE  33        33.45         0.33+02


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE  36        36.45         0.36+02



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE  40        40.45         0.40+02
CARRIAGE CONTROL CHARACTER IS BLANK   THIS IS LINE  41        41.45         0.41+02

CARRIAGE CONTROL CHARACTER IS   0     THIS IS LINE  43        43.45         0.43+02


CARRIAGE CONTROL CHARACTER IS   -     THIS IS LINE  46        46.45         0.46+02



CARRIAGE CONTROL CHARACTER IS   4     THIS IS LINE  50        50.45         0.50+02
```

Figure 10-3

PRECISE CONTROL OF RASTER POSITIONS FOR SCOUTV PRINTING: LOCSTV

In the preceding discussion, SCOUTV was presented as a means for simulating printer output on the S-C 4020. For such simulation, vertical positioning is controlled by a line count and the increment between lines. Lateral positioning is dependent upon the built-in left margin and the number of character spaces provided by the associated FORMAT statement.

The range of SCOUTV applications has been greatly extended by means of routines which permit precise control of raster positions, with the printing still under FORMAT control. Two routines, LOCSTV and LOCSAV, which permit setting and retrieval of raster positions, add the following capabilities to the SCOUTV system:

1. Printing can be done below IY = 11. (The basic SCOUTV does not permit writing below the 1012th raster position, counted from the top of the frame.)

2. Printing can be done to the left of IX = 24. (This area is inaccessible to the basic SCOUTV system, which uses a fixed left margin of IX = 24.)

3. Printing can be started at a point which is independent of a line count, line increment, or number of character spaces. (For the basic system, which simulates line printing, printing can only be positioned in units of character widths and of print line depth.)

The LOCSTV provides these facilities by means of the call statement:

CALL LOCSTV (IX, IY, IB)

IX — The X-raster position to be used as the starting point of the next line of printing. Stated in another way, IX is the new left margin, which will be effective for subsequent lines and frames until another CALL LOCSTV is encountered.

The programmer must ensure that information to be printed on subsequent lines can be accommodated between IX and the right edge of the frame. IX should be specified such that

$1023 - IX \geq 8*$ (number of characters in the line to be printed).

IY — A Y-raster position counted from the BOTTOM of the frame, to be used as a reference for subsequent lines of printing. Since the position IX, IY will be referenced as if it were the starting point of the previous line of printing, the next line will start at IX, IY only if the carriage control character + (to suppress spacing) is used.

The position of subsequent lines of SCOUTV mode printing relative to IY will depend upon the carriage control characters used and upon the value of the current line increment, INC.

IY is effective only for subsequent lines on the <u>same frame.</u> It will be changed (reset to the top of the frame) if the frame is advanced, or if one of the forms of CALL SCOUTV intervenes between a CALL LOCSTV and the affected WRITE OUTPUT TAPE statement.

IB Limit below which printing will not take place, expressed in raster counts from the BOTTOM of the frame. It will be effective for subsequent lines and frames until another CALL LOCSTV is encountered.

To return these indicators to the basic SCOUTV specifications, use CALL LOCSTV (24, 1000, 11).

Note particularly that, unlike IX and IB, the IY reference point will not carry over from frame to frame, nor will it survive execution of a CALL SCOUTV. If the IY reference is to be used on a new frame, the CALL LOCSTV statement should be repeated.

One use of LOCSTV is illustrated in Figure 10-4.

An illustration of how SCOUTV mode printing can be combined with a graph is shown in Figure 10-5. That example also illustrates another way to print tabular data using SCOUTV and LOCSTV.

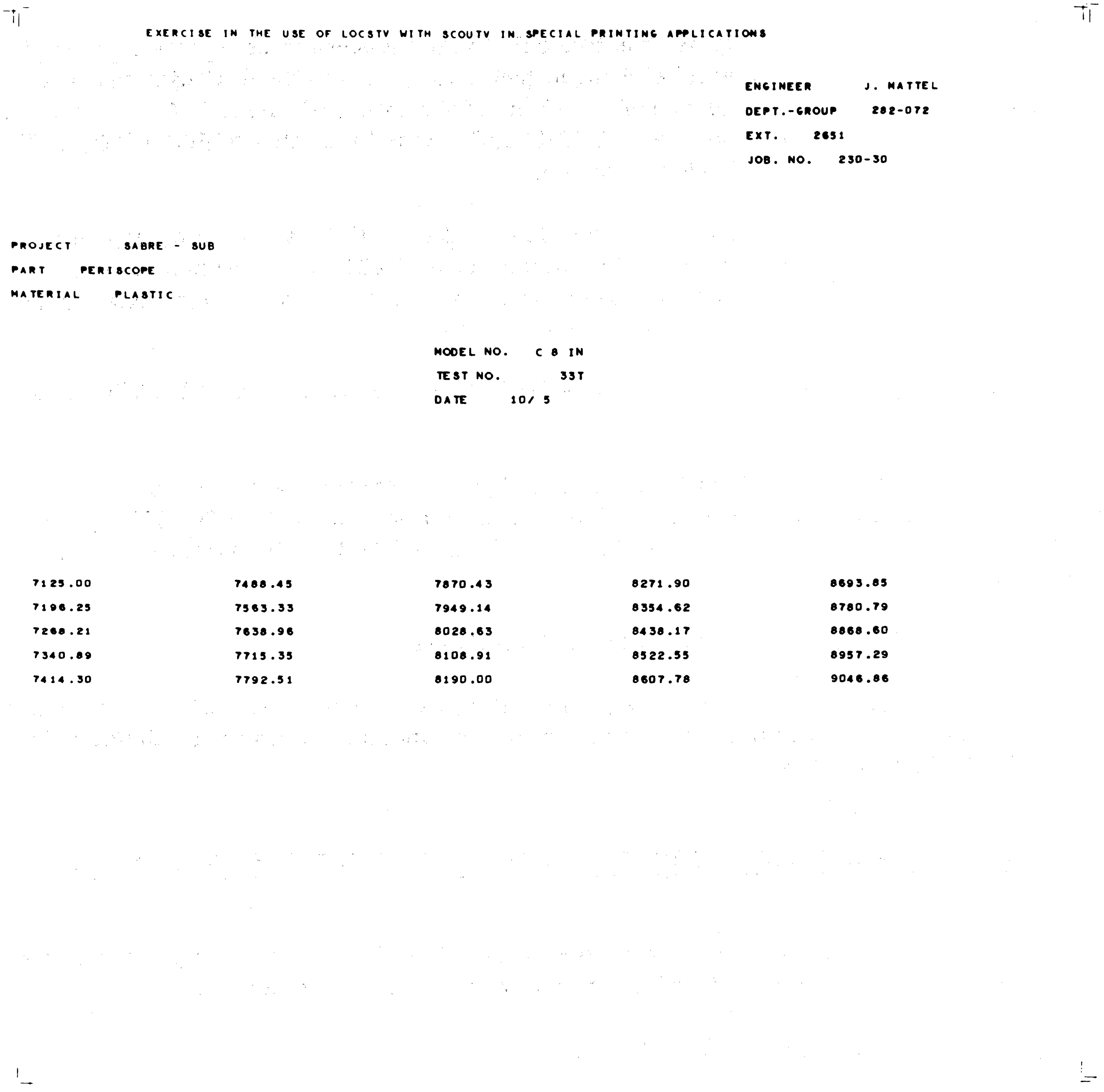
PRESERVATION (OR RETRIEVAL) OF RASTER POSITIONS USED BY SCOUTV: LOCSAV

The raster coordinates of the starting point of the line most recently printed in the SCOUTV mode can be stored in ISX, ISY by using the statement:

CALL LOCSAV (ISX, ISY)

The programmer can use this information to relate other types of S-C 4020 output to the location of the last line of print.

LOCSAV will also aid in a special situation that occurs with interrupted printing. If SCOUTV printing is interrupted on one camera in order to display one or more frames on the alternate camera, the SCOUTV printing and film advance for the alternate



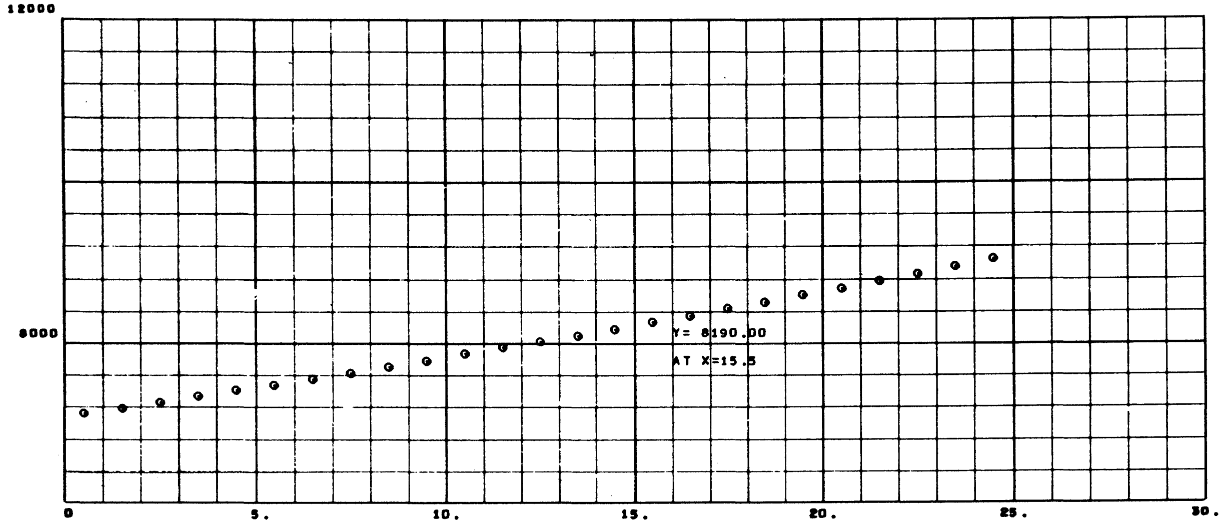
```
            EXERCISE IN THE USE OF LOCSTV WITH SCOUTV IN SPECIAL PRINTING APPLICATIONS

                                                                                ENGINEER      J. MATTEL
                                                                                DEPT.-GROUP     282-072
                                                                                EXT.    2651
                                                                                JOB. NO.   230-30


PROJECT      SABRE - SUB
PART    PERISCOPE
MATERIAL    PLASTIC

                                                   MODEL NO.   C 8 IN
                                                   TEST NO.       33T
                                                   DATE     10/ 5




  7125.00               7488.45               7870.43               8271.90               8693.85
  7196.25               7563.33               7949.14               8354.62               8780.79
  7268.21               7638.96               8028.63               8438.17               8868.60
  7340.89               7715.35               8108.91               8522.55               8957.29
  7414.30               7792.51               8190.00               8607.78               9046.86
```

Figure 10-4

GRAPH AND PRINTED INFORMATION COMBINED ON ONE FRAME

NOTE..THIS EXERCISE SHOWS FURTHER WAYS IN WHICH SCOUTV PRINTING AND LOCSTV MAY BE USED

0000-00
057 057



```
7125.00        7488.45        7870.43        8271.90        8693.85
7196.25        7563.33        7949.14        8354.62        8780.79
7268.21        7638.96        8028.63        8438.17        8868.60
7340.89        7715.35        8108.91        8522.55        8957.29
7414.30        7792.51        8190.00        8607.78        9046.86
```

Figure 10-5

camera will alter the line reference information. On return to the original frame on the first camera, the programmer will be unable to resume SCOUTV printing at the next line unless he uses LOCSAV and LOCSTV. The following sequence shows the procedure that might be utilized.

```
.
.
.
CALL SCOUTV
.
.
.
CALL CAMRAV (35)

CALL FRAMEV
.
.
.
    Write a portion of this 35mm frame, using the appropriate WRITE OUTPUT
    TAPE and FORMAT statements.
.
.
.
CALL LOCSAV (M1X, M1Y)

CALL CAMRAV (9)

CALL FRAMEV
.
.
.
    Write one or more frames on the 9-inch camera, using the appropriate
    WRITE OUTPUT TAPE and FORMAT statements.
.
.
.
CALL LOCSTV (M1X, M1Y, 11)

CALL CAMRAV (35)
.
.
.
    Continue writing on 35mm film as if no interruption had taken place.
.
.
.
```

SUBROUTINE VCPS

Subroutine VCPS produces tape for printing Matrix Symbols, Vector Characters, and Vector Lines. Clues in Key Punch Data may provide for Automatic Line Justification, Column Printing, and Page Advances, all done automatically when column or page ends occur. Column graphs may be drawn. Data may provide also for precise positioning of lines and characters, without the Automatic Justification features.

VCPS calls particular subroutines of the S-C 4020 package. Enter with the statement CALL VCPS.

Maximum use of automatic features occurs when a blank card heads Data. S in Col. 71 of this card causes leftward shift for lower case characters in some instances, to produce more attractive print - (see below) . Also, the S sets up the condition for search for usable Format information when improper format cards are encountered (described below). Page numbers print automatically at lower right. The type of card which must occur next is called FORMAT CARD.

<u>FORMAT CARD</u> may have information in Col. 1-9. (Balance not read.)

Col. 1-3 show number of characters, spaces included, to a line. Maximum allowed value is 155 , but the useful limit is 128. For more than 128 characters use a FRAME LIMITS CARD, without the leading blank card. (Frame Limits Card format is described elsewhere in this write-up.) Minimum character spacing set automatically is 8 raster points.

Col. 4-5 choose Matrix or Vector Characters, set character and line spacing, and set character size for Vector Characters. Blank Col. 4 sets Matrix Print. Value in Col. 4 sets IW for CHSIZV. Increment for spacing will be (4 X IW) + 3, but no less than 8.

Col. 5 sets IH for CHSIZV, and line spacing for both Matrix and Vector Characters. Blank Col. 5 sets lines 20 raster points apart. When Col. 5 is not blank or zero, formula is (6XIH) + (4XIH) for increments, with lower and upper margins at 36 and 24 raster points. The margins will increase for large values in Col. 5, leaving fewer lines.

> Note : When Col. 5 value is 5 or greater, characters are called from TAB11V (CAPS) AND TAB15V (lower case) . These are Block letters and are not suitable for use in smallest width.

Col. 6 Sets number of columns. Blank, zero, or 1 gives a page with 1 centered column. Multiple columns are centered, with a 3-character distance between columns. There will be a return from VCPS if character size and number are excessive. The comment NXV ERROR - TOO NEAR FRAME EDGE will print. As many as 9 columns may be attempted.

TO SUPPRESS RETURN ON THIS TYPE ERROR, punch S In Col. 71 of a blank card used before a format card to induce fullest use of automatic features. VCPS will then try to use the format by reducing character size, and then number of columns. When these efforts fail, there is a return to the calling program.

The condition set by the S also produces character adjustment. Symbols occuring after f, i, j, l, and t will be adjusted leftward to produce more attractive print. Characters used this way are in TAB05V, and TAB07V. Adjustment occurs only for lower case characters.

When last word of a line is clued for all caps, an asterisk should be entered following last character.

TO PRODUCE UNEQUAL LEFT AND RIGHT MARGINS, col. 13-18 and col. 19-24 of the leading, otherwise blank, card may be used. This card is actually a frame limits card. Integer values must be used for raster values of margins. For the left margin, enter raster count in col. 13-18 and for right margin use col. 19-24.

Col. 7 may provide to suppress data listing. Enter 2 in Col. 7 to suppress data lists.

Col. 8 may carry a SKIP OR HOLD character. For blank Col. 8, the equal sign is set up. (SEE COLUMN TEXT CARD DESCRIPTION.)

Col. 9 Sets character intensity. For blank Col. 9 , intensity will be 1 . Intensity is always COL. 9 VALUE + 1. Thus, for 2 'strikes', enter 1 in Col. 9. (VCPS WILL NOT USE AN INTENSITY GREATER THAN 3).

COLUMN TEXT CARDS

COLUMN TEXT CARDS carry text for printing. The number of columns read will be one greater than indicated in Col. 1-3 of the Format Card. VCPS reads Col. 1-72 of text cards until the Format Card requirement is met. For a character count of 72 , two cards will be read, 72 columns from one card, and 1 column on a succeeding card. In lines of 144 characters 3 cards would be read for each line of print. The additional column read each time is called "KEY" column and should never have any information other than one of the KEY COLUMN CLUE NUMBERS.

Key Column Clue Numbers are 1, 3, 4, 5, 6 and 7.

KEY 1 directs VCPS to expect Column Vector Line Cards.

KEY 3 gives frame advance, and Column Text Card must follow.

KEY 4 provides for reading a Frame Limits Card as following card. By using KEY 4, sections of VCPS other than Automatic Printer Section may be entered, without frame advance.

KEY 5 gives return from VCPS.

KEY 6 provides for reading a new Format Card. Any format parameters may be changed. Printing will continue on the same frame if there is room for at least one print line, or if not, frame advances.

KEY 7 operates as KEY 6 , but frame always advances, with printing start at top of the new page.

SKIP OR HOLD MARK may be any character in COL. 8 of a Format Card. When the selected character occurs singly in text, subsequent characters in the line do not print, and the remaining text will justify. Thus, when a word near line-end is begun and does not effectively hyphenate, placing a Skip Mark ahead will delete it. Another use is to save time by preventing execution of blanks near a line-end.

Blank cards are used to represent line skips. They must occur in pairs (or multiple) when more than one card is used per line. Cards for line skips should have skip mark in Column 1 to shorten execution time, and a digit in Column 2 will cause skipping of that many lines.

Note : When Col. 1 has a Skip Mark, ANY character in Col. 2 is

interpreted as DECIMAL CODE count for Line Skipping.

The line above did not justify to right. It has a proper look as last line of a paragraph. The period was followed by two 'hold' marks on the data card.

When Col. 8 on a Format Card is blank, the equal mark is set as ' SKIP or HOLD ' mark, and does not print. A character must be chosen which does not occur in text. A new character may be selected at any time by using KEY 6 on a Column Text Card to afford reading a new Format Card.

CONVENTIONS

Characters used for CLUES when doing Column Printing are :

, * +) (/ and SKIP or HOLD mark, as indicated by Col. 8 of the effective Format Card.
S in Col. 71 of a leading card causes use of TAB05V OR TAB07V, tables constructed for use when the 'character squeeze' justification feature is operative.

Clues are effective only within one print line.

When printing Vector Characters, TABL5V, (a table of lower case characters) is used unless a CAPITALIZING or *Italicizing* Clue occurs.

A Capital prints when a symbol occurs after 2 or more spaces, or directly after a comma, or when an asterisk has occurred next to a symbol. Numerals, the plus, and the slash must be treated as capitals.

When a character has 3 or more spaces before it and yet must be printed lower case, enter a period immediately before the character to prevent capitalization. The period thus used is treated as a clue and will not print.

As general practice it is best to use blanks on both sides of a , * +) (or / when it must be printed. The comma is an exception. *The comma prints when it occurs just before another comma, or just* before an asterisk (*) , and always prints when in Column 2.
When capitals have been 'turned on' by the asterisk, commas cease to be 'clues' and will print.

The Asterisk used as a Clue will induce exclusive use of TABL1V, (or TABL6V in *Italics*) until another Clue asterisk occurs.

Two asterisks occuring between letters or numerals cause shift into or out of capitals. Spaces of the asterisks are used to print the balance of the symbols in the word so that no gap is left.

The Plus (+) as a clue at first encounter *Induces use of Italics.* Characters will be from TABL6V or TABL7V depending on Capitalization Clues, until another plus occurs.

Two plus signs between letters or numerals cause shift into or from *Italics* . Spaces occupied by the plus signs are closed so that no gap will appear in the print.

Any number of characters in one line will be underlined from any symbol with a right parenthesis just before it to a symbol having a left parenthesis immediately after it. Greek Symbols are set by a Left

Parenthesis placed immediately after a character when an underline has not been begun. Greek symbols will be 'turned off' at the next occurrence of the condition.

A slash used as a clue will increase or decrease character intensity. Intensity increases by 1 , and another clue / decreases intensity by 1. VCPS will not use intensity greater than 3.

FOR MATRIX PRINT ONLY 3 CLUE DEVICES ARE OPERATIVE. underlining may be done with 'inverted parentheses' as described above, and SKIP or HOLD marks are valid. The asterisk (*) will produce brighter intensity when used as a Clue. All other " CLUES " print!

Clue Marks become spaces when used as clues. Successive clue marks may produce capitals, since a symbol appearing after two spaces is capitalized. Double asterisks or plus signs within words are exceptions for this rule. See use of double asterisks and plus signs above.

INDENTATION

Text will justify to left when first print character is in Col. 1, 2, or 3. Indentation is held when first print symbol is beyond Col. 3.

COLUMN GRAPHS

When ! is found in Key Column of a Column Text Card, VCPS expects data for Vector Line drawing within a column width. Following data card must have integer values thus :

Col 1-4 Character count at vector line start.
Col 5-8 Line count from previous Print line to vector start.
Col 9-12 Character count at vector line end.
Col 13-16 Line count from previous Print line to vector end.
Col 17 Any integer directs VCPS to expect column text card next.

Printing resumes at line next below last Text Line, to provide for labels on the Vector lines. To resume printing below the drawn vectors, use a Line Skip Card (a card having Skip Mark in Col 1.) See information on use of Col 2 of a skip card for producing multiple skipping. The information is found in section on use of Clue Marks.

When VCPS is supplied data as described, margins, line spacing, and increments between characters are computed by the Subroutine. For more precise control of margins and spacing a FRAME LIMITS CARD may precede data., and the combination of blank card plus Format Card is not used. When column text cards follow a Frame Limits Card having applicable columns punched as described below, automatic column and page advance features operate but framing and spacing will be as set on the Frame Limits Card.

FRAME LIMITS CARD

A Frame Limits Card precedes any data for execution by VCPS . When such a card is blank, the subroutine expects a Format Card for automatic frame determination. A card for precise frame control will have information that controls spacing absolutely. Vector and Matrix characters will print 1 unit apart in X, and justification may increase spacing between words. It is helpful to set the frame geometry so that 1023 is an integral multiple of a frame unit in Raster Points.

Frame Limits Card Format

Column :	Information :
1-6	Floating X Value at Left FrameEdge
7-12	Floating X Value, Right Frame Edge
13-18	Integer, Raster count, Left Margin
19-24	Integer, Raster count, Right Margin
25-30	Floating Y Value, Lower Frame Edge
31-36	Floating Y Value, Top Frame Edge
37-42	Integer, Raster count, Lower Margin
43-48	Integer, Raster count, Top Margin
49-54	Floating X Value, Column One Start
55-60	Floating Y Value, First Line Start
61-63	Integer is character count per line
64	Integer, Sets Character Width, IW
65	Integer, Sets Character Height, IH
66-67	Integer, Sets Table Of Characters 1 for TABL1V, 5 for TABL5V , Etc.
68	Integer, Sets Character Intensity Subroutine ADDS 1 To Value Entered
69	NOT READ
70	Key integer controls Reading and

Processing Of Following Cards

Blank or 0 when Text card follows

Note : Col. 70 Is not operative for Integer Values when col. 61-63 are NON-BLANK.

When COLUMN TEXT CARDS are NOT next, Col. 61-63 MUST be blank, and Key Numbers are effective in Col. 70.

Blank or 0 if TEXT CARD is next

1 when VECTOR LINE CARD is next

2 when DECIMAL CODE CARD follows

71 KEY PUNCH CHARACTER will become SKIP or HOLD MARK. Blank Col. 71 results in use of equals as SKIP or HOLD mark.

72 NOT OPERATIVE FOR PRINTING FROM COLUMN TEXT CARDS.

When TEXT CARDS follow, Integer controls Orientation of VECTOR CHARACTERS and PRINT LINES.

Blank or 0 for upright print.

1 to rotate print 90 DEG. CCW

2 to rotate print upside down.

3 to rotate print 90 DEG. CW.

TEXT CARD FORMAT

A Text Card may carry information for matrix or Vector Character printer. Control of position is precise. Automatic Columns and page advances do not happen, and lines do not justify. Each line will print exactly along the Y value on the card. This type of card is expected after a Frame Limits Card which contains NO INFORMATION BEYOND COL. 48. NO CLUE MARKS (AS DESCRIBED FOR COLUMN TEXT CARDS) HAVE ANY "CLUE" EFFECTS WHEN FOUND IN A TEXT CARD.

Column :	Information :
1- 3	Integer sets character width, INCRW Value must not exceed 7 Blanks, or zeroes, give Matrix Print
4- 6	Integer sets character height, INCRH Value must not exceed 7 FOR " SLANTED " MATRIX PRINT, VALUE IS RASTER x DISTANCE BETWEEN SYMBOLS Note : Vector character size varies slightly with table used. Maximum width or Height is (8 X INCRW) , or 56 Raster Points.
7- 8	NOT READ
9-11	Integer selects Character Table. 1, 0, or Blanks select TABL1V. 2, 3, 5, 6, 7, 8, 9 for TABL2V, etc. FOR SLANTED MATRIX PRINT, VALUE IS RASTER Y DISTANCE BETWEEN SYMBOLS.
12	NOT READ
13-18	Floating X value at center of first Text Character. Value should be a Whole Number. Decimal places are not retained in conversion to rasters. First text symbol is character in Col. 25, even if Col. 25 is blank. Blanks, or 0 's give start at raster 32 + left margin, or at 1.0 on frame, (The greater value is used.) X is given as for normal print when

rotation is done (see Col. 72)

19-24 Floating Y value is used to establish centerline thru a line of text.
Digits to right of decimal are used To 2 places, but truncation occurs if Col. 72 is 1 or 3.
Always give Y as for normal frame.

25-66 42 Characters of TEXT. More than 42 characters require additional cards.
Starting X value on a continuing card must be 42. greater than that given on the previous card.
82 is practical limit for number of Vector Characters. 123 is limit for Matrix Symbols USING TEXT CARDS.

67-68 Integer value is number of times each character is struck - 1.
Blanks, or 0 's for once.
1 for 1 additional strike, etc.

69 NOT READ

70 Integer is clue for processing next card :

Blank or 0 when same type card follows

1 when a VECTOR LINE CARD is next.

2 when a DECIMAL CODE CARD follows

3 to advance frame. A FRAME LIMITS CARD must follow.

4 to change Frame Geometry without Frame Advance. Frame Limits Card MUST FOLLOW.

71 Skip or Hold mark. Blank Sets Equals as Mark. SELECTED CHARACTER DOES NOT PRINT.

72 Integer is clue for Rotation.
Blank or 0 for normal upright print.
1 to rotate Print and Frame CCW 90 °
2 to rotate Print and Frame CCW 180 °
3 to rotate Print and Frame CCW 270 °

Vector Line Cards

Column :	Information :
1- 6	Floating X value at Vector Start
7-12	Floating Y value at Vector Start
13-18	Floating X value at Vector End
19-24	Floating Y value at Vector End
25-66	NOT READ
67-68	Intensity Integer. Blank, 0, or 1 for drawing vector once, 2 for Twice Etc.
69	NOT READ
70	AS DESCRIBED FOR TEXT CARDS.
71	NOT READ
72	AS DESCRIBED FOR TEXT CARDS.

Decimal Code Cards
Produce a single character at a Specified Point.

Column :	Information :
1-24	As for Text Cards But Columns 1-3 must not be blanks or 0's.
25-26	Decimal Code integer for Character Wanted. (See S-C 4020 Manual)
27-29	Degree Integer for Character Orientation. (The symbol rotates, frame does not.) Blanks or 90 For upright character. 180 to rotate 90 DEG. CCW, 270 for an Upside Down symbol, 360 for 90 DEG. CW rotation.
30-66	NOT READ
67-68	Intensity integer as for Vector Cards
69	NOT READ
70	As for Text and Vector Cards
71-72	NOT READ

KEYPUNCH CHARACTERS FOR VCPS

TABLE:	CHARACTERS FROM CARD PUNCH:
1V	ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789=.-,()+*$/"
5V	abcdefghijklmnopqrstuvwxyzœ! i¿ñ´`¨ç^=.-,()&*?†'
6V	*ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789=.-,()+*$/"*
7V	*abcdefghijklmnopqrstuvwxyzœ!:¿ñ´`¨çÇ=.-,()+*?/„*
8V	ABΞΔE IΓHIΘKΛMNOΠ]PΣTΥΦΩX↓Z Ç —
9V	αβξδε γηιθκλμνοπ ρστυΦωχψζ
11V	**ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789=.-,()÷×"/"**
15V	**abcdefghi jklmnopqrstuvwxyz! áéíóó' ?â"=.-,()÷×◊/"**

Note: Characters above are produced with

LVW = 2

LVH = 3

```
                        PRINTING WITH VCPS SUBROUTINE
                   (SAMPLE DATA CARDS- OUTPUT ON NEXT PAGE)
  CAPITALIZING,CLUES ARE,2 OR MORE,SPACES,,THE,COMMA, AND,THE,ASTERISK.==

,DOUBLE,ASTERISKS WITHIN A WORD CAUSE CHANGE TO OR FROM,CAPITALS.,FOR TWO
EXAMPLES.*JA**CKSON, NON-**E**UCLIDEAN.  .==


,THE,PLUS,SIGN IS THE+,ITALICIZING+CLUE.==

,TWO,PLUS,SIGNS WITHIN A WORD CAUSE CHANGE TO OR FROM,ITALICS, AND SPACES
CLOSE AS WITH DOUBLE ASTERISKS.,FOR EXAMPLE, NON-++COOPERATIVE, UN++TIED.
                                       .==

  )UNDERLINING(MAY BE DONE WITH INVERTED PARENTHESES.==

,A,LEFT,PARENTHESIS WHICH IMMEDIATELY FOLLOWS A CHARACTER IS THE CLUE
FOR CHANGING TO OR FROM,GREEK,CHARACTERS WHEN AN*UNDERLINE HAS NOT BEGUN.*
,FOR,EXAMPLE,,NOW DO A,GREEK,ALPHABET( ABCDEFGHIJKLMNOPQRSTUVWXYZ( ABCDEF.



,THE,EQUAL,SIGN IS A SKIP MARK, UNLESS ANOTHER SYMBOL IS SELECTED.,A LONE
,SKIP,MARK FOLLOWED BY A,BLANK CAUSES A SKIP TO NEXT LINE.,ANY SUBSEQUENT
CHARACTERS DO NOT PRINT, AND THE LINE WILL BE JUSTIFIED.,TWO,SKIP,MARKS IN
SUCCESSION HOLD A LINE-END BY PREVENTING JUSTIFICATION OF THE RIGHT EDGE.
,FOR EXAMPLES,==
,DELETE CHARACTERS AFTER THE,SKIP,MARK. = THESE SYMBOLS WILL BE DELETED.
,DELETE CHARACTERS AFTER,TWO,SKIP,MARKS. == THESE SYMBOLS WILL BE DELETED.
```

Capitalizing Clues are 2 or more spaces, The Comma, and The Asterisk.

Double Asterisks within a word cause change to or from Capitals. For two **examples, JAckson, non-Euclidean.**

The Plus Sign is the *Italicizing* clue.

Two Plus Signs within a word cause change to or from Italics, and spaces close as with double asterisks. For example, non- *cooperative.*

<u>Underlining</u> may be done with inverted parentheses.

A Left Parenthesis which immediately follows a character is the clue for changing to or from Greek Characters when an UNDERLINE HAS NOT BEGUN. For Example, Now do a Greek Alphabet αβφδε γηιθκλμνυ˜ ͻυ΄ιφυ/ψζ Abcdef.

The Equal Sign is a skip mark, unless another symbol is selected. A lone Skip Mark followed by a Blank causes a skip to next line. Any subsequent characters do not print, and the line will be justified. Two Skip marks in succession hold a line-end by preventing justification of the right edge. For examples,
Delete characters after the Skip Mark.
Delete characters after Two Skip Marks.

INDEX OF SUBROUTINES

SUBROUTINE PAGES

Underscored page denotes primary reference.