PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 4 : **G06F 15/20**	**A1**	**(11) International Publication Number:** **WO 90/03614** **(43) International Publication Date:** 5 April 1990 (05.04.90)

(21) International Application Number: PCT/US89/03940

(22) International Filing Date: 14 September 1989 (14.09.89)

(30) Priority data:
248,811 23 September 1988 (23.09.88) US

(71) Applicant: DATACARD CORPORATION [US/US]; 11111 West Bren Road, Minneapolis, MI 55343 (US).

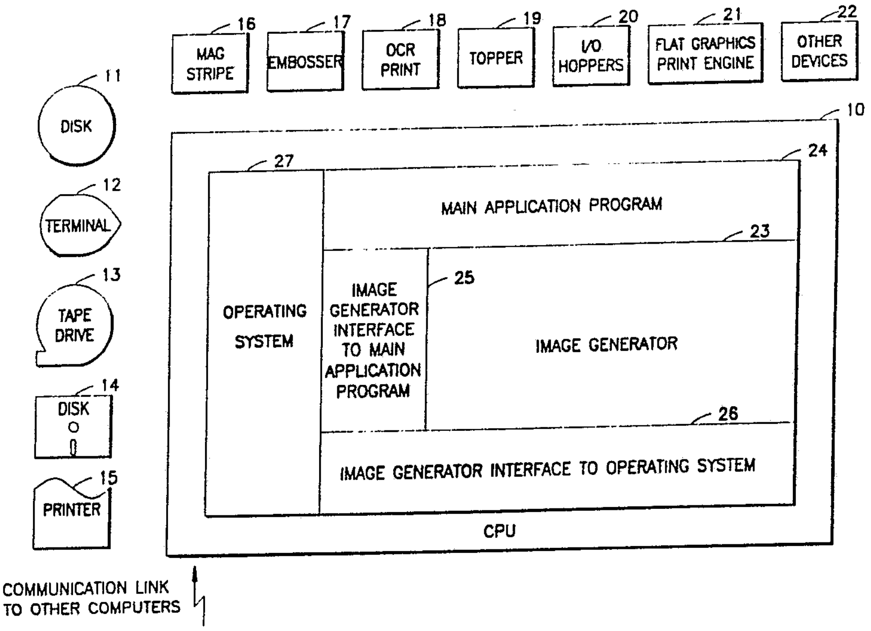
(72) Inventors: WENDT, Richard, W. ; 535 Thomas Avenue, Shakopee, MI 55379 (US). JOHNSON, Brian ; 4501 Park Glen Road, ,110, St. Louis Park, MI 55416 (US).

(74) Agent: SCHUMANN, Michael, D.; Merchant, Gould, Smith, Edell, Welter & Schmidt, 3100 Norwest Center, 90 South Seventh Street, Minneapolis, MN 55402 (US).

(81) Designated States: AT (European patent), BE (European patent), CH (European patent), DE (European patent), FR (European patent), GB (European patent), IT (European patent), JP, LU (European patent), NL (European patent), SE (European patent).

Published
· *With international search report.*

(54) Title: IMAGE GENERATOR SOFTWARE



(57) Abstract

An apparatus and method for printing graphics images on plastic cards is disclosed, comprising a computer (10), a graphics printer (15), and an image generating program (23) executing on the computer (10) and supplying graphic images to the graphics printer (15). The image generating program (23) can combine logo images, Kanji character images, English character images, and barcode images for printing on the plastic card. In addition, a method defining an input data record, and a plurality of graphics commands contained therein, is disclosed.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria
AU	Australia
BB	Barbados
BE	Belgium
BF	Burkina Fasso
BG	Bulgaria
BJ	Benin
BR	Brazil
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CM	Cameroon
DE	Germany, Federal Republic of
DK	Denmark
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
HU	Hungary
IT	Italy
JP	Japan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
LI	Liechtenstein
LK	Sri Lanka
LU	Luxembourg
MC	Monaco
MG	Madagascar
ML	Mali
MR	Mauritania
MW	Malawi
NL	Netherlands
NO	Norway
RO	Romania
SD	Sudan
SE	Sweden
SN	Senegal
SU	Soviet Union
TD	Chad
TG	Togo
US	United States of America

IMAGE GENERATOR SOFTWARE

Technical Field of the Invention

The present invention pertains generally to an apparatus and method for printing graphics images, text, and barcodes on plastic cards.

Background of the Invention

Plastic cards, such as those used for credit cards, debit cards, security cards, and the like, are pervasive within modern society. Prior art methods of manufacturing these plastic cards first require that generic cards be stamped with logos, corporate names, and other features. An inventory of these stamped cards is maintained and supplied eventually to an embossing machine for personalization. The personalization process includes, for example, embossing a customer's name on the plastic card. These prior art methods are costly and inefficient because an inventory of stamped cards must be maintained prior to the personalization process.

Summary of the Invention

The present invention replaces prior art methods with a sophisticated card personalization system. One object of the present invention is to permit the imprinting of graphic images, text, and barcodes, on generic plastic cards, thereby bypassing the need for maintaining an inventory on stamped cards.

Brief Description of the Drawings

In the drawings, where like numerals refer to like elements throughout the several views:

Figure 1 is a block diagram describing the computer system, associated peripheral devices, and the Image Generator software of the present invention;

Figure 2 is a block diagram describing the relationships between tasks of the Image Generator software;

Figure 3 is a flow chart describing the functions Grafix Executive task;

Figure 4 is a flow chart describing the functions of the Parse task;

Figure 5 is a flow chart describing the functions of the Add A Logo task;

Figure 6 is a flow chart describing the functions of the Add A Font task;

Figure 7 is a flow chart describing the functions of the Make (English or Kanji) Text task;

Figure 8 is a flow chart describing the functions of the Make A Barcode (#1,2,3) task;

Figure 9 is a flow chart describing the functions of the Build A Card task;

Figure 10 illustrates the card image buffer allocated by the Add A Logo task;

Figure 11 describes the format of the Logo Definition File;

Figure 12 describes the input data format and field structure;

Figure 13 describes the image element data structure;

Figure 14 shows three examples of graphics data and the resulting card images that are created;

Figure 15 describes the graphics command language used by the present invention; and

Figure 16 describes the cell sizes of some representative character sets.

Detailed Description of the Preferred Embodiment

In the following detailed description of the preferred embodiment, reference is made to the accompanying drawings which form a part hereof, and in which is shown by way of illustration a specific embodiment in which the invention may be practiced. It is to be understand that other embodiments may be utilized without departing from the scope of the present invention.

The present invention is a graphics Image Generator program that imprints graphic images, text, and barcodes on plastic cards. The present invention includes the specific data format for supplying input to these graphics printing functions.

Referring initially to Figure 1, the Image Generator 23 resides on a computer system that drives a number of peripheral devices associated with card personalization. Shown in Figure 1 is the Computer 10, that executes the Image Generator 23. Attached to the Computer 10 are such standard peripherals as a Hard Disk 11, a Terminal 12, a Tape Drive 13, a Floppy Disk 14, and a Printer 15. The Computer 10 also controls the operation of a number of peripheral devices associated with card personalization systems. These devices include a Magnetic Striper 16, an Embosser 17, an OCR Printer 18, a Topper 19, Input and Output Hoppers 20, a graphics Print Engine 21, and other devices 22. Within the Computer 10, the Image Generator 23 creates the graphic images printed via the Print Engine 21. The Image Generator 23 operates under the direction of a Main Application program 24. To enhance portability, the Image Generator 23 includes an Interface Package 25 for handling communications with the Main Application 24 and an Interface Package 26 that insulates the Image Generator 23 from the particular Operating System 27 running on Computer 10. This enhances portability and permits implementation of the Image Generator 23 on a plurality of different computer systems.

The Image Generator 23 permits a number of different types of images to be printed on plastic cards, including logos, English text, Kanji text, and a plurality of barcode types. The functions provided by the Image Generator 23 are illustrated in Figure 2. The primary routine, called the Grafix Executive task 25, controls the operation of the Image Generator 23. The

Executive task 25 communicates with the Main Application 24 and in turn directs the operation of a number of other tasks, labeled in Figure 4 as 26, 27, 28, 29, 30, 31, 32, 33, and 34. Although an alternative embodiment could operate in a non-multi-tasking environment, the preferred embodiment distributes functions of the Image Generator 23 amongst distinct tasks within a real-time, pre-emptive, multi-tasking environment. This provides the advantages of speed, portability, and expandability. The speed of execution of the Image Generator 23 increases because while one task is waiting, for example, on an I/O completion, other tasks can continue to execute. The modular nature of the tasks simplifies portability by breaking a complex piece of software into smaller, more manageable units. This modular approach to tasks also makes it easy for a programmer to add more features to the Image Generator 23 and to promote additional parallelism.

Figure 3 is a flow diagram describing the functions performed by the Grafix Executive task. The Executive task 25 waits to receive graphics data from the Main Application 24 (35). The Main Application 24 reads the graphics data from an input tape or from a communications line. The Executive task 25 sends the graphics data to the Parse task 26 (36) and then waits for the Parse task 26 to complete its functions (37). If an error occurs during the parsing function, the Executive task 25 sends a message to the operator (38). If the parsing function is completed successfully, a tag identifying the graphics record is sent to the Main Application 24 so that the Main Application 24 can synchronize graphics image printing within the personalization process and respond in the future with a "commit" message (39). The Executive task 25 then initiates the tasks which creates the image elements for the card image (40). The Executive task 25 waits for a "commit" command from the Main Application 24 and the completion of the Build A Card task 31 (41). If the Main Application 24 does not issue a "commit" command but instead issues an "abort" command, the Executive task 25 will discard the card image built by the Build A Card task 31 (41). Once a "commit" is received from the Main Application 24 and the Build A Card task 31 completes, the resulting card image is sent to the Main Application 24 (44), which synchronizes and initiates the operation of the Print Engine 21. Upon completion, the Executive task 25 awaits another graphics record (35).

Figure 4 is a flow chart describing the functions of the Parse task 26. The Parse task 26 waits to receive graphics data from the Executive task 25 (45). Once the data is received the Parse task 26 will allocate a parameter buffer (46). The Parse task 26 then begins parsing graphics commands from the graphics data field (47). This parsing function continues until the end of the graphics data field is encountered (48). If logo commands are parsed, the Parse task 26 will set the logo parameters in the parameter buffer (50). If font commands are encountered, the Parse task 26 will set the font parameters in the parameter buffer (51). If text commands are encountered, the Parse task 26 will set the text parameters in the parameter buffer (52). If barcode commands are encountered, the Parse task 26 will set the barcode parameters in the parameter buffer (53). Once the end of the graphics data field is encountered, the address of the parameter buffer 54 is sent back to the Executive task 25 (49), which in turn sends the parameter data to the other tasks.

Figure 5 is a flow chart describing the functions performed by the Add A Logo task 27. The Logo task 27 receives the logo parameters from the Executive task 25 (55). The Logo task 27 allocates a Card Image Buffer 60 (56), shown in Figure 10, which in the preferred embodiment is a two-dimensional array of 512 bits x 810 bits. The Logo task 27 retrieves the requested bit mapped image from the Disk 11 by matching the name passed in the logo parameters with file names residing on the Disk 11 (57). Figure 11 describes the format of a Logo Definition File 86. The Logo Bit Map 91 is positioned within the Card Image Buffer 60 according to either the logo parameters received from the Executive task 25 or the Default Horizontal Coordinate 87 and Default Vertical Coordinate 88 residing in the Logo Definition File 86. The Card Image Buffer 60 is then sent to the Build A Card task 31 (59). The Logo task 27 will await the next set of logo parameters (55).

Figure 6 is a flow chart describing the functions of the Add A Font task 30. The Font task 30 receives the font parameters from the Executive task 25 (61). The Font task 30 accesses the Disk 11 to determine the size of the requested character set (62). The Font task 30 allocates a buffer of sufficient size to store the character set (63). The Font task 30 then reads the character set from the Disk 11 (64) and loads it into memory (65). At this point, the Font task 30 transmits the character set 67 to the Make English Text task 29 (66). The Font task 30 then returns to wait for the next set of font parameters (61).

Figure 7 is a flow chart describing the functions of either the Make Kanji Text task 28 and the Make English Text task 29. The Text tasks 28 and 29 wait to receive the the text parameters from the Executive task 25 (68). The Make English Text task 29 also waits to receive the character set from the Font task 30 (68). The Text tasks 28 and 29 execute two loops, the first to process all Kanji or English text lines (69), the second to process all characters within a single text line (71).

Each character from the data parameter is used to access a specific character image (72). The Make

English Text task 29 accesses a character image residing in the character set sent from the Font task 30. The Make Kanji Text task 28 accesses a character image residing on a Kanji ROM peripheral device described in the co-pending patent application "Rom Conversion Device And Method" and assigned to the assignee of the present invention. The Make Kanji Text task 28 may also access custom Kanji character images stored in files on Disk 11 if the character image desired does not reside on the Kanji ROM peripheral device.

The character image retrieved is stored in a Image Element 111 as shown in Figure 14 (73). The Image Element 111 is then stored in a linked list pointed to by Element List Pointer 110 (74). This cycle repeats for each character in a text line (71) and for each text line specified in the text parameters (69). Once all text lines have been processed, the linked list containing the characters stored in individual Image Elements 111 is sent to the Build A Card task 31 (70). The Text tasks 28 and 29 then wait for the next set of text parameters (68).

Figure 8 is a flow chart describing the functions of the Make A Barcode tasks 32, 33, and 34. In the preferred embodiment three types of barcodes can be generated: JAN8, JAN13, and NW7. Three separate Barcode tasks, 32, 33, or 34, are used to process each of the three barcode types, in an attempt to enhance parallelism and performance. Each Barcode task waits to receive barcode parameters from the Executive task 25 (76). Once received, the Barcode task will loop through all barcode lines, of the corresponding type, specified in the barcode parameters (77). Within each line a barcode may be generated from a look-up table similar to a character set (78). Some barcodes require the generation of "check digits" before this look-up function occurs (78). Some barcode algorithms also require a specific amount of spacing between the barcode elements (78). The entire barcode is built at one time (78) and the resulting data is stored in an Image Element 111 as shown in Figure 14 (79). Each barcode is inserted into a barcode linked list pointed to by an Element List Pointer 110 (80). Upon completion, this barcode linked list 82 is sent to the Build A Card task 31 (81).

Figure 9 is a flow chart describing the functions of the Build A Card task 31. The Build task 31 waits to receive a Card Image Buffer 60 from the Logo task 27 and separate linked lists for Kanji or English text 75 and barcodes 82. Once all linked lists have been received (83), the Build task 31 will load these images into the Card Image Buffer 60, positioning each element according to the Horizontal and Vertical Coordinates 112 and 113 stored in the Image Element 111 (84). Once this is accomplished, the Card Image Buffer 60 is returned to the Executive task 25 (85), which in turn transmits the Card Image Buffer 60 to the Main Application 24.

A custom set of graphics commands are used to specify the data for each of the graphics images discussed herein. The graphics commands can be provided either from magnetic tape or on-line communications with a remote computer system. The commands are independent of media type. These commands specify what graphics images to use, where on the card the data should be placed, which character set to use, and which barcode type should be generated. The commands are specifically designed to be extendable for future enhancements.

Figure 12 describes the format of the input data. Figure 12 shows the Graphics Data field 100 and its relationships to the rest of the data in the input record. The input data stream is divided into physical groups of characters called blocks. These blocks are either transmitted individually from a remote computer system or separated by an Inter-Record Gap 94 when stored on a magnetic tape 92. Each block consists of one or more logical groups of characters called records. The preferred embodiment accepts only those data streams which have one record per block. Within each record there can be many fields. A number of data fields are shown in Figure 12: Search Code 95, Embossing Data 96, OCR Print Data 97, Magnetic Stripe Data 98, and Forms Printer Data 101. In the preferred embodiment, placement of these different fields must be structured as shown in Figure 12. The start of the Graphics Data field 100 is identified by the Graphics Module Code 99. This 8 bit code must be unique within the block of data.

The first character of the Graphics Data field 100 is the Command Language Version Code 102, which specifies the version of the command language. The intent is to provide a "hook" to identify future versions of the command language (which may not be compatible with the previous versions). The Image Generator 23 can identify which version is in use by reading this initial character. An error condition can be indicated if the Image Generator 23 does not support the version indicated. In the preferred embodiment, this character must be "G".

The second character of the Graphics Data field 100 is the Command Delimiter Code 103. This position in the Graphics Data field 100 specifies a single character code which is used throughout the remainder of the current Graphics Data field 100 as a delimiter. This character precedes every graphics command. The end of the Graphics Data field 100 is signified by two sequential occurrences of this character. Any character may be used as the Command Delimiter Code 103. In Figure 12 a "back-slash" is used.

Following the Command Delimiter Code 103, the Image Generator 23 reads the Graphic Data field 100 in a sequential manner looking for the next occurrence of the Command Delimiter Code 103. Until the Command Delimiter

Code 103 is found, all characters 104 that are read are ignored. This provides the capability of embedding comments into the Graphics Data field 100.

After the Image Generator 23 finds an occurrence of the Command Delimiter Code 103, the following field is a Graphics Command field 105. The first three characters of each Graphics Command field 105 is a Command Mnemonic 107. In general, there are two command types, those which supply data and those which specify how and where the data should be put on the plastic card. Following the three character Command Mnemonic 107 is an Operator Subfield 108 consisting of one or two equal sign characters. One equal sign indicates the value which follows is to be used for only the Graphics Data field 100 in which it appears. Such a value is called a "transient" parameter. Two equal signs indicate that the value which follows is to be used not only for the Graphics Data field 100 in which it appears, but also for all subsequent Graphics Data fields 100 until it is changed by a future occurrence of the same command. Such a value is called a "persistent" parameter. The third portion of the Graphics Command field 105 holds the Command Value 109. The format of this field and its length depends on which Command Mnemonic 107 is used.

Figure 15 describes each Command Mnemonic 107 and its usage. For all of these commands, a warning message is displayed to the operator in the event that an unrecognized command or value is used. Warnings are also generated if a command would position an image off the edge of the card or if the requested logo, character set, or barcode type does not exist. When any errors occur, card personalization is interrupted until the operator resumes production.

The "LN#=" command 129 is a transient data command that specifies the name of the logo identified by "#". This command must be used in each Graphics Data field 100 that requires a logo on the card. An example of this command is shown in Figure 14. Graphics Data field 118 associates the logo name "DATACARD" with the command "LN1". The resulting logo 120 is shown on the card 119.

The "LH#=" command 130 is a transient control command that specifies the horizontal position of the left edge of the logo identified by "#". The command value is four numeric characters representing a base 10 number of 0.001 inch units. This number is the horizontal distance from the left edge of the card to the left edge of a minimum size, imaginary, rectangle which encloses the logo. The reference point for logo positioning is defined as the lower left corner of the imaginary rectangle, called the "logo definition rectangle", which is just big enough to enclose the logo. An example is shown in Figure 14. Graphics Data field 118 contains the command "LH1" which results in the horizontal placement of image 120 on card 119. If an "LH#=" is not specified in the Graphics Data field 100, then the Default Horizontal Coordinate 87 stored in the Logo Definition File 86 is used. It is recommended that this command not be used in the Graphics Data field 100. Instead, logo positioning should be controlled by the values stored in the Logo Definition File 86, which the system operator can easily modify.

The "LV#=" command 131 is a transient control command that specifies the vertical position of the bottom edge of the logo identified by "#". This command is used within each Graphics Data field 100 that activates a logo. The command value must be four numeric characters representing a base 10 number of 0.001 inch units. This number is the vertical distance from the bottom edge of the card to the bottom edge of the logo definition rectangle which encloses the logo. An example is shown in Figure 14. Graphics Data field 118 contains the command "LV1" which results in the vertical placement of image 120 on card 119. If an "LV#=" is not specified in the Graphics Data field 100, then the Default Vertical Coordinate 88 stored in the Logo Definition File 86 is used. It is recommended that this command not be used in the Graphics Data field 100. Instead, positioning should be controlled by the values which were stored in the Logo Definition File 86, which the system operator can easily modify.

The command "F##=" 133 is a persistent format control command which specifies the font for the text line identified by "##". Currently the only command values that are valid are "GFx", "KJx", and "KIx". The value "GFx" specifies that the text line uses single byte EBCDIC character coding for English text and that the character image definitions are stored in the character set identified by "x". The value "KJx" specifies that the text line uses two byte JIS (Japanese Information Standard) Kanji character coding and prints the Kanji characters at the size associated with "x". The command value "KIx" specifies that the text line uses two byte IBM Kanji character coding and prints the Kanji characters at the size associated with "x". In the preferred embodiment, the size of the character cell is listed in the table in Figure 16. The cell width and cell height values 142 through 151 in Figure 16 represent the number of bits in each dimension. The various command values, or character set designators, are not tied to the specific combinations listed in Figure 16. Rather, they can be treated as logical designators. The contents associated with each character set designator can be altered by input data or operator command. These commands request the loading of a particular character set into a memory location associated with the command value. For example, assume there are only two memory locations for storing character sets, but the system has a selection of four character sets from which to choose. The system could be forced to load any of the character sets into the memory location associated with command value "GF1".

The command "D##=" 136 is a transient data command which specifies the data to be used for the text line identified by "##". This command is used within each Graphics Data field 100 that activates a text line of any type. For English text, identified by the command "F##==GF", the value field is a string of single byte EBCDIC codes which each represent one text character. For Kanji text, specified by the command "F##==KJ", the value field is a string of two byte JIS codes, each representing one Kanji text character. Each JIS code activates one Kanji character from either the 6,349 character JIS standard set or from the custom Kanji character set. For Kanji text, identified by the operator "F##==KI", the value field is a string of two byte IBM codes each representing one Kanji text character. Each IBM code activates one Kanji character from either of the IBM Kanji character set or from the custom Kanji character set. In all cases, the string of character codes must be followed the end-of-data character specified by the command mnemonic "EDC==". An example is shown in Figure 14. Graphics Data field 121 contains two "D##=" fields. The command field "D01" in Graphics Data field 121 results in the image 123 on card 122. The command field "D02" in Graphics Data field 121 results in the image 124 on card 122.

The "EDC==" command 132 is a persistent format control command that sets the end-of-data character for all text and barcode data strings. Being a persistent parameter, this code can be used once in a Graphics Data field 100 and it will remain in effect for subsequent Graphics Data fields 100 until the command mnemonic is again encountered. An example is shown in Figure 14. Graphics Data field 121 contains two "D##=" fields that are both terminated by with the EDC character "~".

The "H##==" command 134 is a persistent format control command that sets the horizontal position of the first character of the text line identified by "##". The command value must be four numeric characters representing a base 10 number of 0.001 inch units. This number is the horizontal distance from the left edge of the card to the horizontal center line of the text character. An example is shown in Figure 14. Graphics Data field 121 contains two "H##=" fields. The command field "H01" in Graphics Data field 121 results in the horizontal placement of image 123 on card 122. The command field "H02" in Graphics Data field 121 results in the horizontal placement of image 124 on card 122.

The command "V##==" 135 is a persistent control command that sets the vertical position of the text line identified by "##". The value for this command consists of four numeric characters representing a base 10 number of 0.001 inch units. This value is the vertical distance from the bottom edge of the card to the base line of the text character. An example is shown in Figure 14. Graphics Data field 121 contains two "V##=" fields. The command field "V01" in Graphics Data field 121 results in the vertical placement of image 123 on card 122. The command field "V02" in Graphics Data field 121 results in the vertical placement of image 124 on card 122.

The command "BT#==" 138 is a persistent format control command that sets the barcode type for the barcode identified by "#". This command is used within each Graphics Data field 100 that activates a barcode or, since it is a persistent parameter, it can be used only once and it will remain set until specified again. Currently, only the following barcode types are valid: JAN8, JAN13, and NW7. Other barcode types can be added easily. An example is shown in Figure 14. Graphics Data field 125 contains two "BT#=" fields. The command field "BT1" in Graphics Data field 125 results in the JAN8 barcode 128 on card 126. The command field "BT6" in Graphics Data field 125 results in the NW7 barcode 127 on card 126.

The command "BD#=" 141 is a transient data command which specifies the data to be used for the barcode identified by "#". This command is used whenever a Graphics Data field 100 activates the barcode of any type. An example is shown in Figure 14. Graphics Data field 125 contains two "BD#=" fields. The command field "BD1" in Graphics Data field 125 results in the JAN8 image 128 on card 126. The command field "BD6" in Graphics Data field 125 results in the NW7 image 127 on card 126.

The command "BH#==" 139 is a persistent control command that sets the horizontal position for the barcode's left edge. This command is used within each Graphics Data field 100 that activates a barcode image or, because it sets a persistent parameter, it may be specified only once and will remain set until specified again. The command field consists of four numeric characters 0-9 representing a base 10 number of 0.001 inch units. This value is the horizontal distance from the left edge of the card to the left edge of an imaginary rectangle which encloses the barcode. An example is shown in Figure 14. Graphics Data field 125 contains two "BH#=" fields. The command field "BH1" in Graphics Data field 125 results in the horizontal placement of image 128 on card 126. The barcode command "BH6" in Graphics Data field 125 results in the horizontal placement of image 127 on card 126.

The command "BV#==" 140 is a persistent control command that sets the vertical position of the barcode's bottom edge for the barcode identified by "#". The value for this command consists of four numeric characters representing a base 10 number of 0.001 inch units. This value is the vertical distance from the bottom edge of the card to the bottom edge of an imaginary rectangle which encloses the barcode. An example is shown in Figure 14. Graphics Data field 125 contains two "BV#=" fields. The command field "BV1" in Graphics Data field 125 results in the vertical placement of image 128 on card 126. The command field "BV6" in Graphics Data field 125 results in the vertical placement of image 127 on card 126.

The commands discussed above use 0.001 inch units to describe the image elements' positions. The actual positioning of image elements requires that any dimensions specified in inches be converted to bits. The Print Engine 21 driven by the Image Generator 23 has a printing resolution of 240 bits per inch horizontal and 240 bits per inch vertical. The entire card image, shown in Figure 10, is 810 bits wide and 512 bits high. The formula used to convert inches to bits, therefore, is:

Bits = ((0.001 inch units x 6) + 12)/25

An alternative embodiment of the present invention includes a macro definition and execution facility for the graphics command language. This embodiment provides the "EXM=" command which means "execute the macro stored under the following macro name." The Image Generator 23 would read the macro file, retrieve the Graphics Data field 100 stored therein under the macro name, and execute the specified graphics commands. Additional command variations would include means for specifying the vertical and horizontal positions of an image in alternative units, centering commands, left and right justification commands, and scaling commands. An alternative embodiment would also include means for entering bit mapped images via the input data stream.

Although a specific configuration has been illustrated and described for the preferred embodiment of the present invention set forth herein, it will be appreciated by those of ordinary skill in the art that any configuration which is calculated to achieve the same purpose may be substituted for the preferred embodiment shown. This application is intended to cover any adaptations or variations of the present invention. Therefore, it is manifestly intended that this invention be limited only by the claims and the equivalence thereof.

WHAT IS CLAIMED IS:

1. Apparatus for printing graphics images on plastic cards, comprising:

a computer;

a graphics printer operatively connected to said computer; and

an image generating program executing on said computer and supplying graphic images to said graphics printer.

2. The apparatus of claim 1, wherein the image generating software comprises:

means for controlling the operation of said image generating program during execution of a command sequence embedded within an input data record;

means for parsing and identifying commands embedded within said input data record supplied to said image generating program;

means for creating logo images to be printed on the plastic card;

means for creating Kanji character images to be printed on the plastic card;

means for creating English character images to be printed on the plastic card;

means for creating barcode images to be printed on the plastic card;

means for combining said logo images, said character images, and said barcode images into a plastic card image; and

means for supplying said card image to said graphics printer.

3. A method for storing logo images comprising:

storing a default horizontal coordinate for the logo;

storing a default vertical coordinate for the logo;

storing a width value of the logo;

storing a height value of the logo; and

storing the bit mapped logo.

4. A method defining an input data record comprising:

storing a search code;

storing an embossing data field;

storing an OCR data field;

storing a magnetic stripe data field;

storing a graphics module code;

storing a graphics data field; and

storing a forms data field.

5. The method of claim 4 wherein storing a graphics data field comprises:

storing a command language version code in a first byte of said graphics data field;

storing a command delimiter code in a second byte of said graphics data field;

storing a graphics command field subsequent to said command delimiter code and immediately preceded by a second command delimiter code; and

storing a last pair of command delimiter codes to identify an end to said graphics data field.

6. The method of claim 5, wherein the graphics command field comprises:

storing a command mnemonic identifying a particular action;

storing an operator sub-field to identify between a transient value or a persistent value for said command mnemonic; and

storing a command value for said command mnemonic.

7. The method of claim 6, wherein the command mnemonic comprises a logo name identifier, whereby a logo image can be retrieved from a memory.

8. The method of claim 6, wherein the command mnemonic further comprises a control command that specifies a horizontal position for a logo.

9. The method of claim 6, wherein the command mnemonic further comprises a control command that specifies a vertical position for a logo image.

10. The method of claim 6, wherein the command mnemonic further comprises a format control command that specifies a font for use in constructing a text line.

11. The method of claim 6, wherein the command mnemonic further comprises a data command which specifies a data string for use in constructing a text line.

12. The method of claim 6, wherein the command mnemonic further comprises a format control command that sets an end-of-data character for all text and barcode data strings.

13. The method of claim 6, wherein the command mnemonic further comprises a format control command that sets a horizontal position for a text line.

14. The method of claim 6, wherein the command mnemonic further comprises a control command that sets a vertical position for a text line.

15. The method of claim 6, wherein the command mnemonic further comprises a format control command that sets a barcode type for a bar code image.

16. The method of claim 6, wherein the command mnemonic further comprises a data command which specifies a data string for a barcode image.

17. The method of claim 6, wherein the command mnemonic further comprises a control command that sets a horizontal position for a barcode image.

18. The method of claim 6, wherein the command mnemonic further comprises a control command that sets a vertical position for a barcode image.

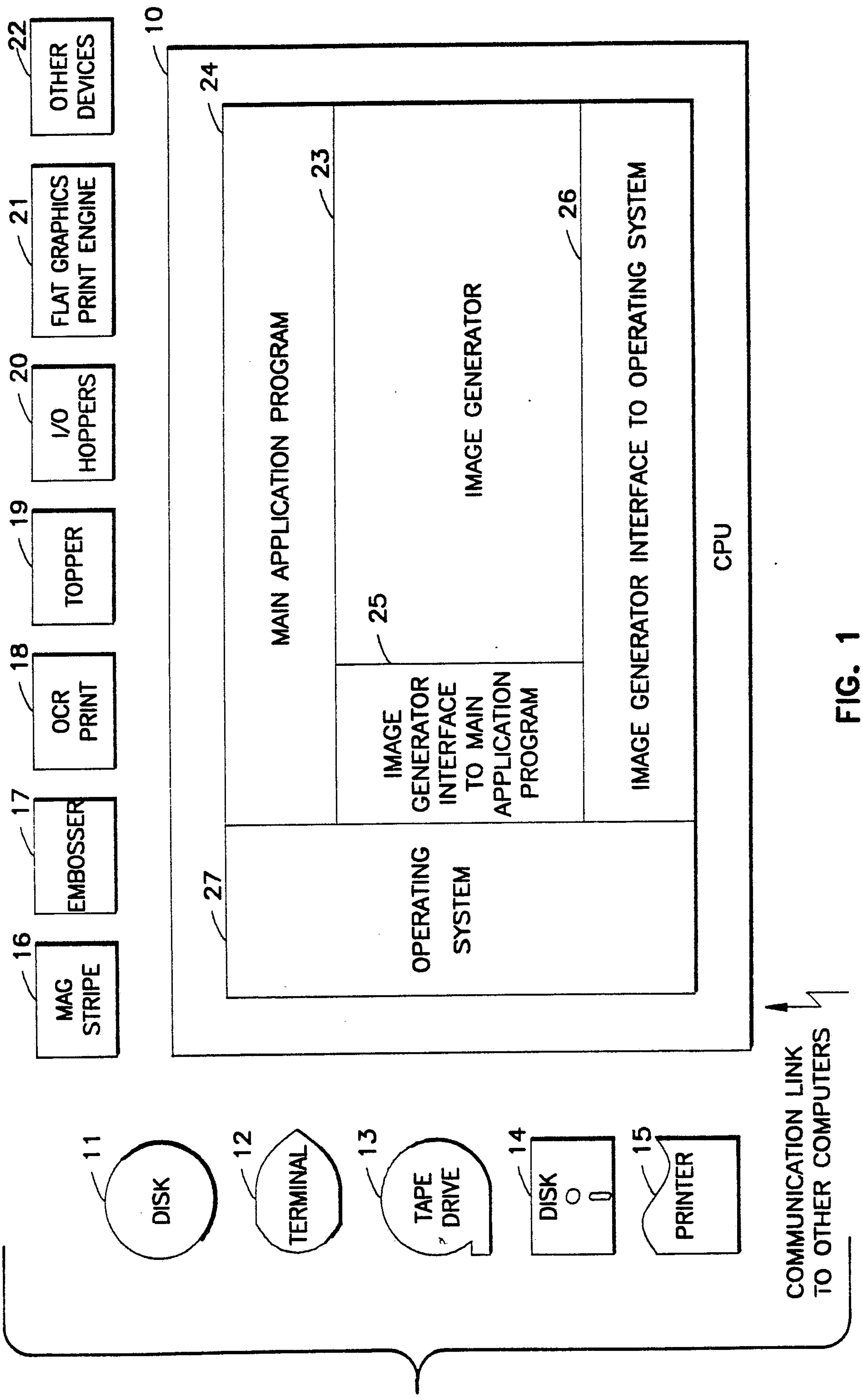

10
CPU
24
23
MAIN APPLICATION PROGRAM
IMAGE GENERATOR
25
IMAGE GENERATOR INTERFACE TO MAIN APPLICATION PROGRAM
26
IMAGE GENERATOR INTERFACE TO OPERATING SYSTEM
27
OPERATING SYSTEM
16
MAG STRIPE
17
EMBOSSER
18
OCR PRINT
19
TOPPER
20
I/O HOPPERS
21
FLAT GRAPHICS PRINT ENGINE
22
OTHER DEVICES
11
DISK
12
TERMINAL
13
TAPE DRIVE
14
DISK
15
PRINTER
COMMUNICATION LINK TO OTHER COMPUTERS

FIG. 1

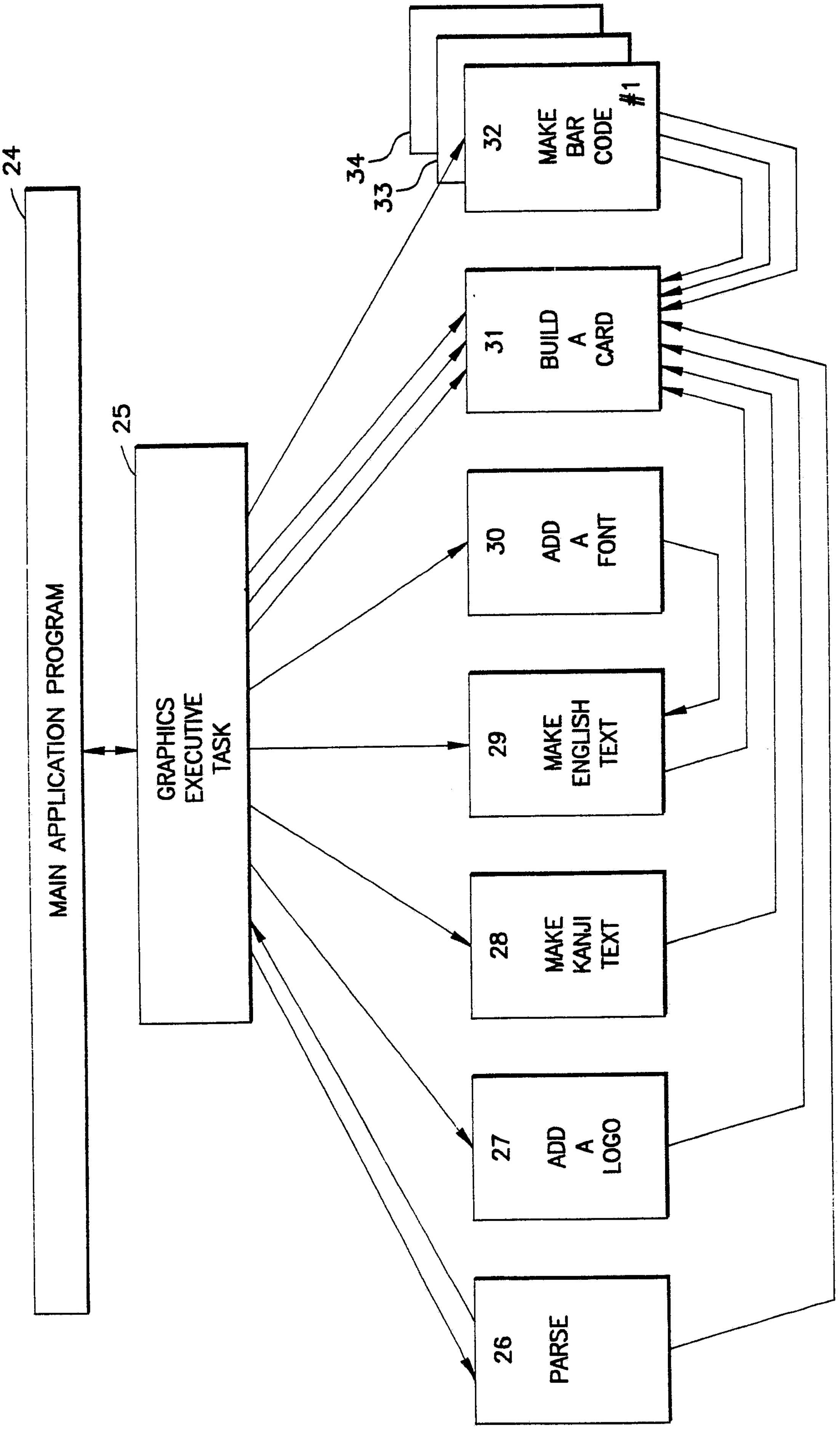

24
MAIN APPLICATION PROGRAM
25
GRAPHICS EXECUTIVE TASK
26
PARSE
27
ADD A LOGO
28
MAKE KANJI TEXT
29
MAKE ENGLISH TEXT
30
ADD A FONT
31
BUILD A CARD
32
MAKE BAR CODE
#1
33
34

FIG. 2

FIG. 3

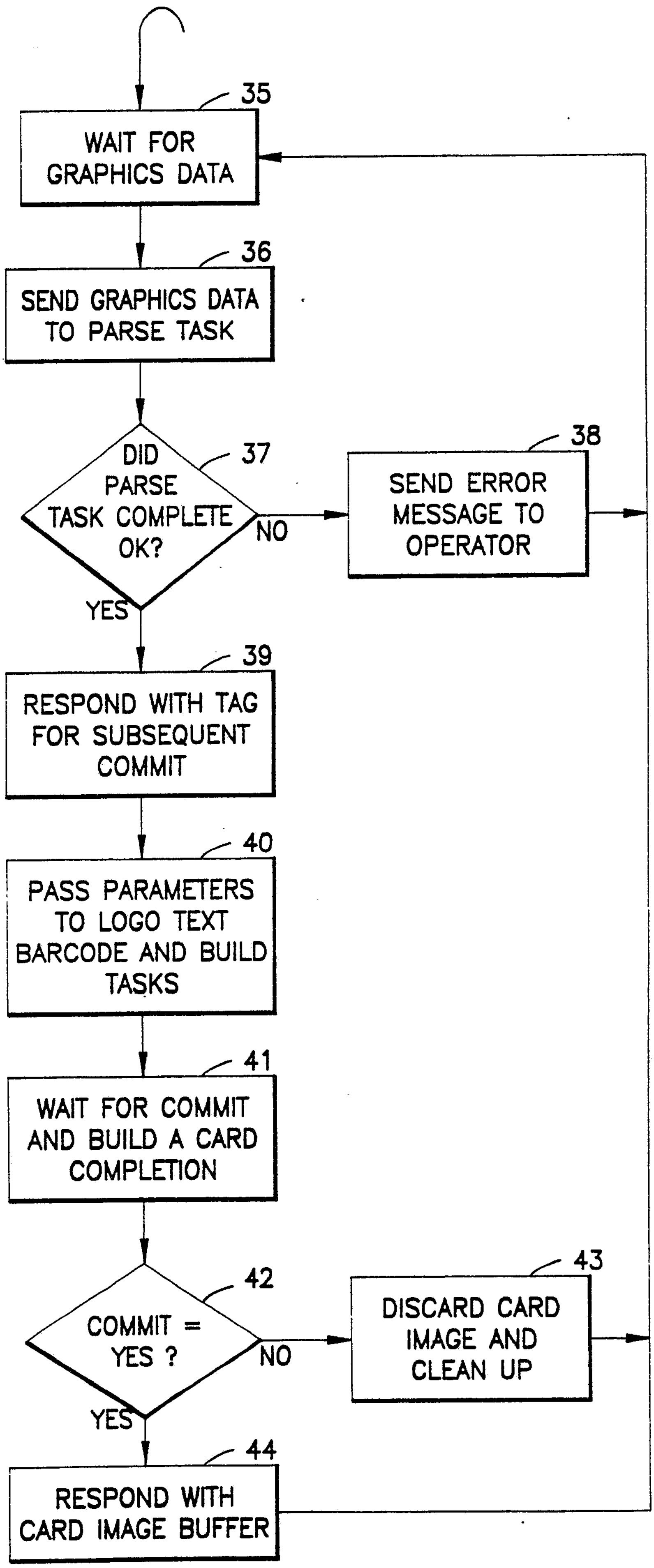
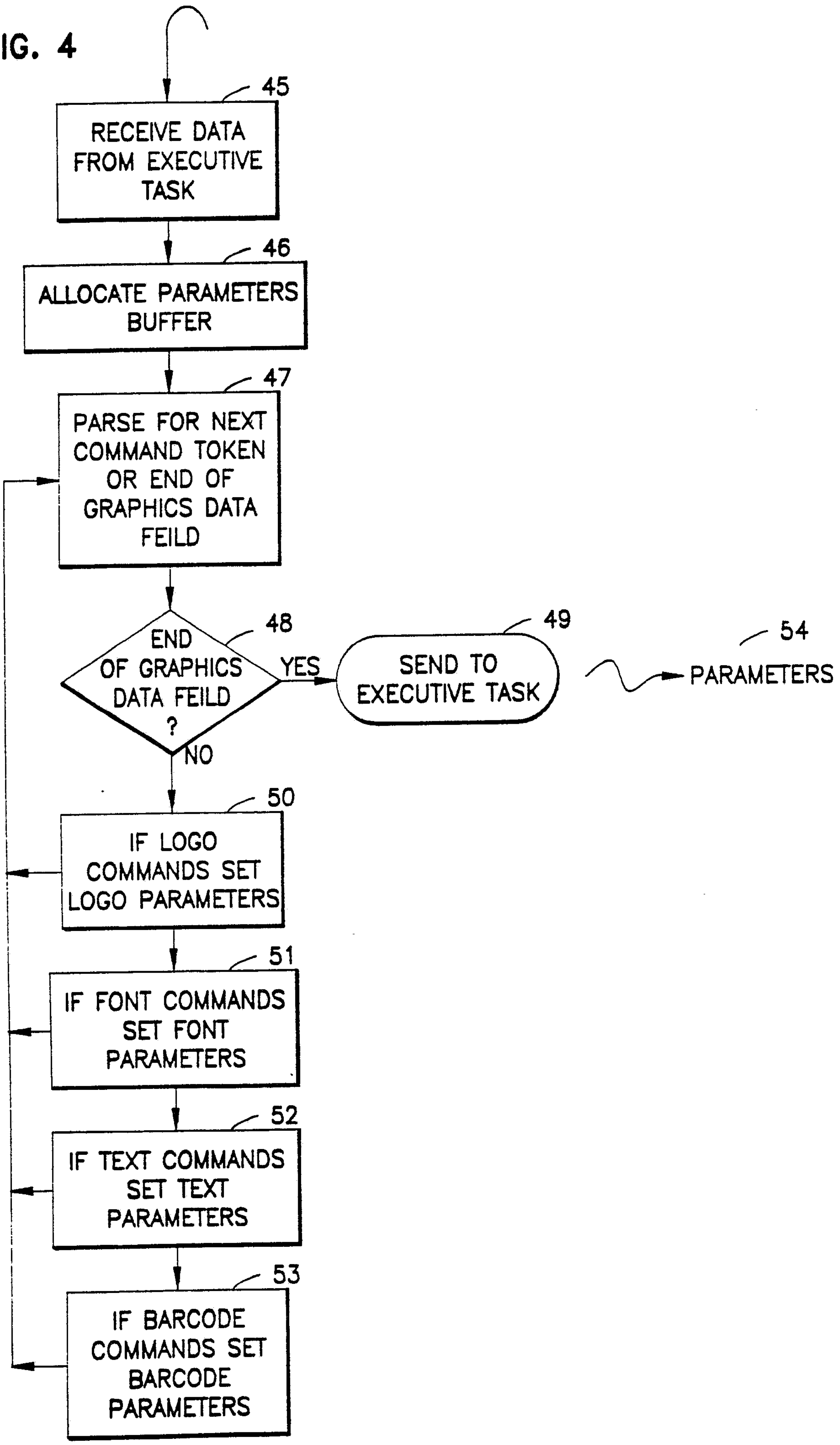

35
WAIT FOR GRAPHICS DATA
36
SEND GRAPHICS DATA TO PARSE TASK
37
DID PARSE TASK COMPLETE OK?
NO
38
SEND ERROR MESSAGE TO OPERATOR
YES
39
RESPOND WITH TAG FOR SUBSEQUENT COMMIT
40
PASS PARAMETERS TO LOGO TEXT BARCODE AND BUILD TASKS
41
WAIT FOR COMMIT AND BUILD A CARD COMPLETION
42
COMMIT = YES ?
NO
43
DISCARD CARD IMAGE AND CLEAN UP
YES
44
RESPOND WITH CARD IMAGE BUFFER

FIG. 4


45
RECEIVE DATA FROM EXECUTIVE TASK
46
ALLOCATE PARAMETERS BUFFER
47
PARSE FOR NEXT COMMAND TOKEN OR END OF GRAPHICS DATA FEILD
48
END OF GRAPHICS DATA FEILD ?
YES
NO
49
SEND TO EXECUTIVE TASK
54
PARAMETERS
50
IF LOGO COMMANDS SET LOGO PARAMETERS
51
IF FONT COMMANDS SET FONT PARAMETERS
52
IF TEXT COMMANDS SET TEXT PARAMETERS
53
IF BARCODE COMMANDS SET BARCODE PARAMETERS

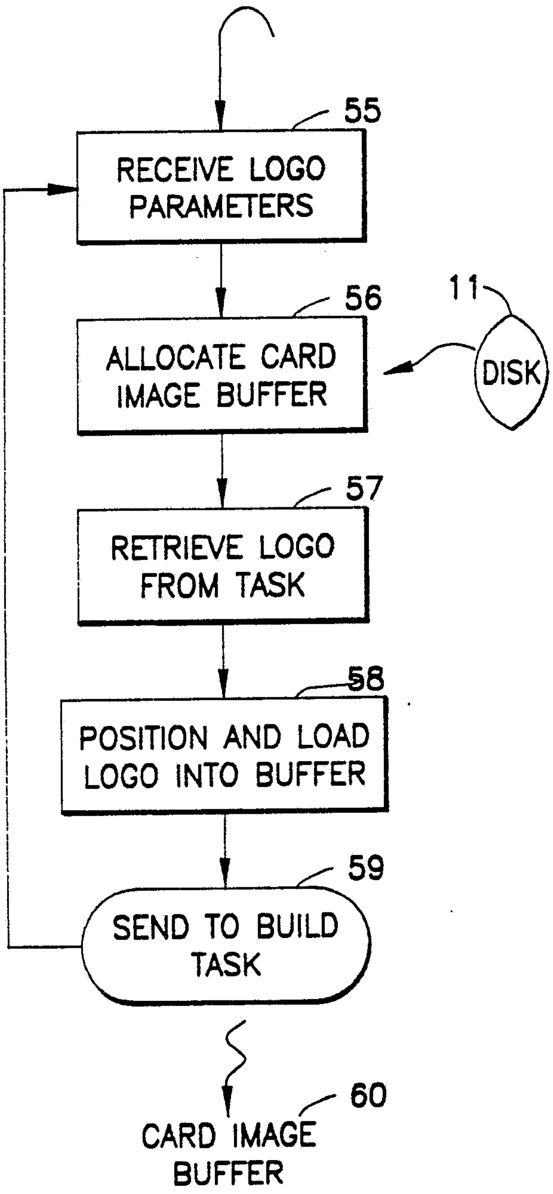

55
RECEIVE LOGO PARAMETERS
56
ALLOCATE CARD IMAGE BUFFER
11
DISK
57
RETRIEVE LOGO FROM TASK
58
POSITION AND LOAD LOGO INTO BUFFER
59
SEND TO BUILD TASK
60
CARD IMAGE BUFFER

FIG. 5

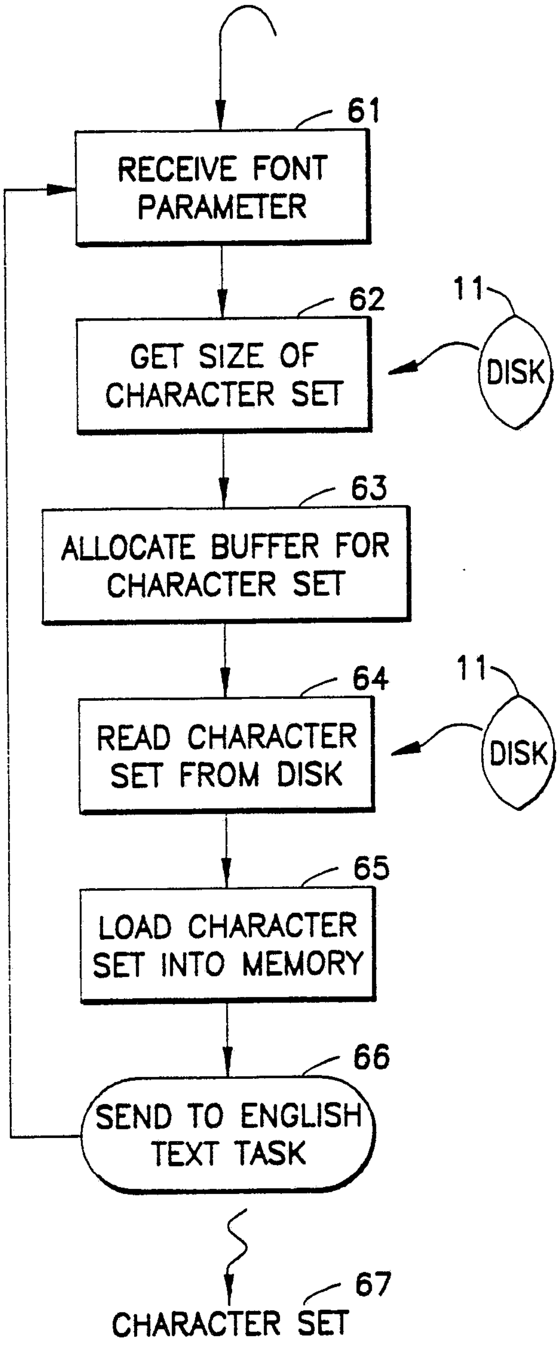
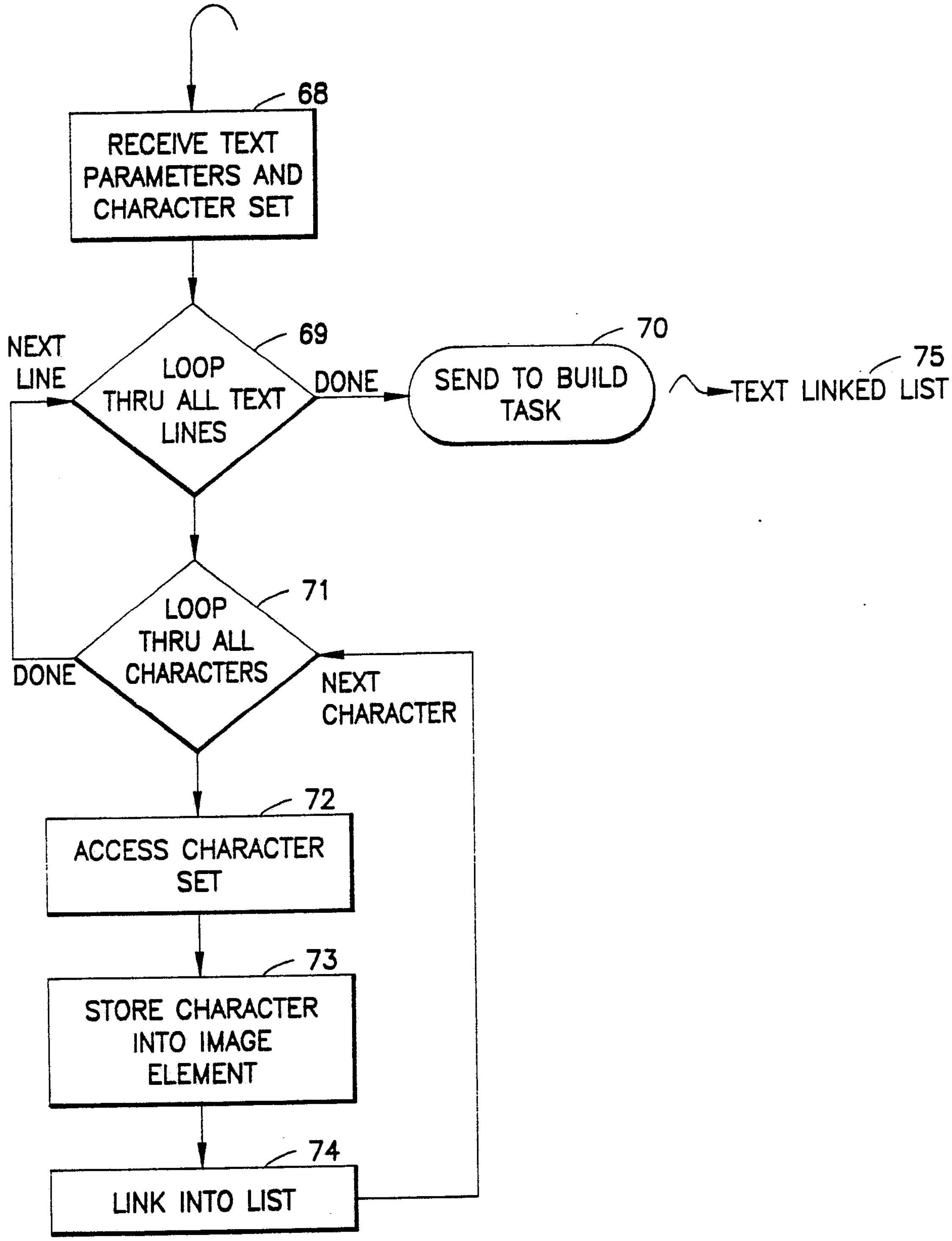

61
RECEIVE FONT PARAMETER
62
GET SIZE OF CHARACTER SET
11
DISK
63
ALLOCATE BUFFER FOR CHARACTER SET
64
READ CHARACTER SET FROM DISK
11
DISK
65
LOAD CHARACTER SET INTO MEMORY
66
SEND TO ENGLISH TEXT TASK
67
CHARACTER SET

FIG. 6

FIG. 7


68
RECEIVE TEXT PARAMETERS AND CHARACTER SET
69
LOOP THRU ALL TEXT LINES
NEXT LINE
DONE
70
SEND TO BUILD TASK
75
TEXT LINKED LIST
71
LOOP THRU ALL CHARACTERS
DONE
NEXT CHARACTER
72
ACCESS CHARACTER SET
73
STORE CHARACTER INTO IMAGE ELEMENT
74
LINK INTO LIST

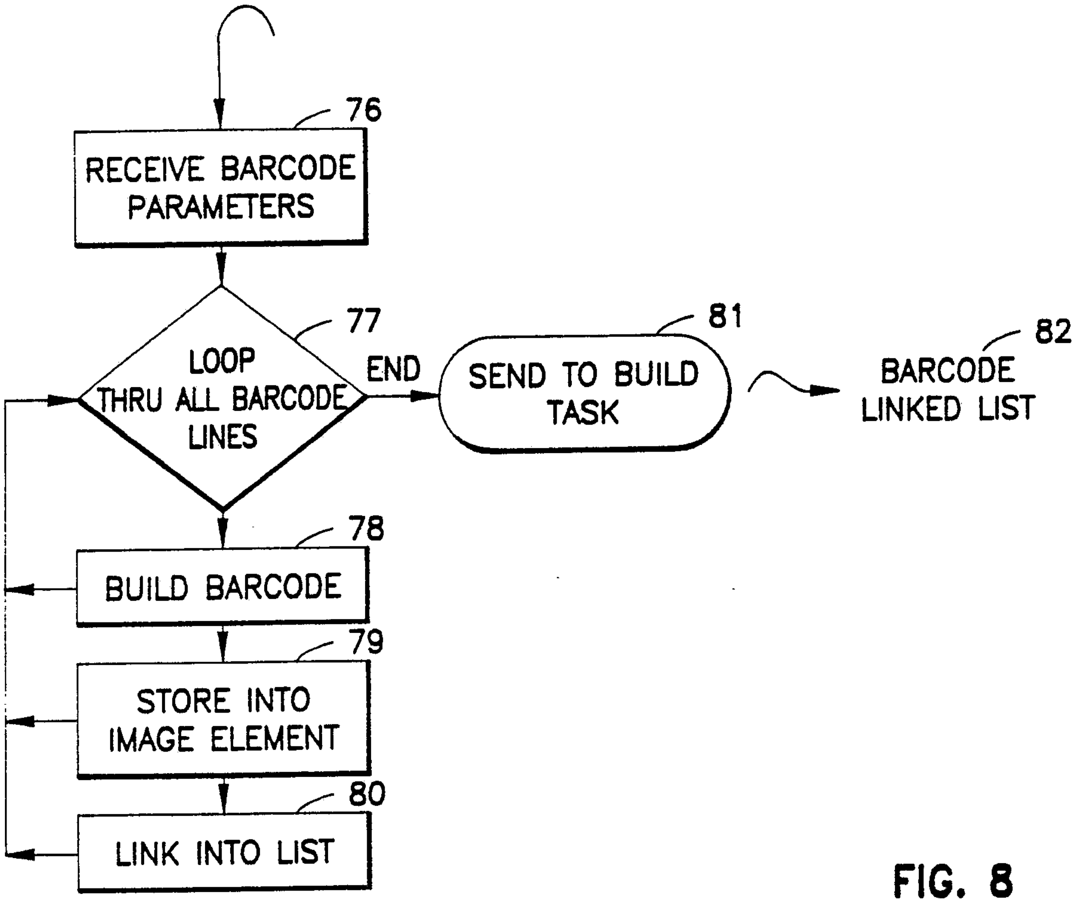

76
RECEIVE BARCODE PARAMETERS
77
LOOP THRU ALL BARCODE LINES
END
81
SEND TO BUILD TASK
82
BARCODE LINKED LIST
78
BUILD BARCODE
79
STORE INTO IMAGE ELEMENT
80
LINK INTO LIST

FIG. 8

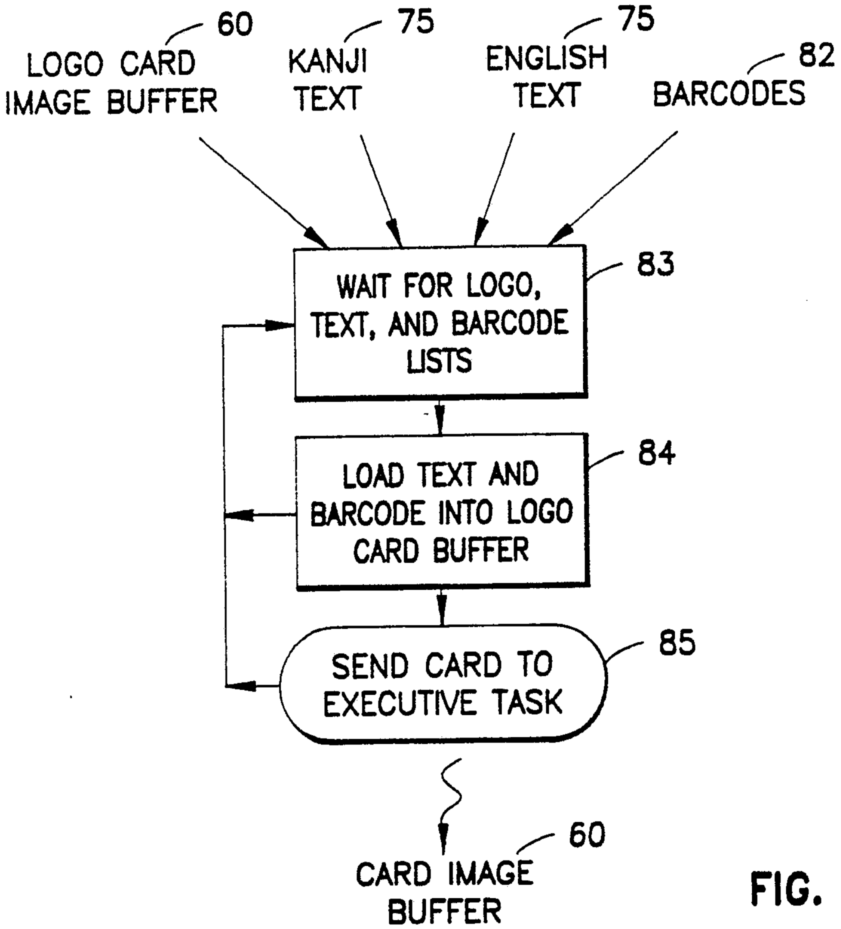

60
LOGO CARD IMAGE BUFFER
75
KANJI TEXT
75
ENGLISH TEXT
82
BARCODES
83
WAIT FOR LOGO, TEXT, AND BARCODE LISTS
84
LOAD TEXT AND BARCODE INTO LOGO CARD BUFFER
85
SEND CARD TO EXECUTIVE TASK
60
CARD IMAGE BUFFER

FIG. 9

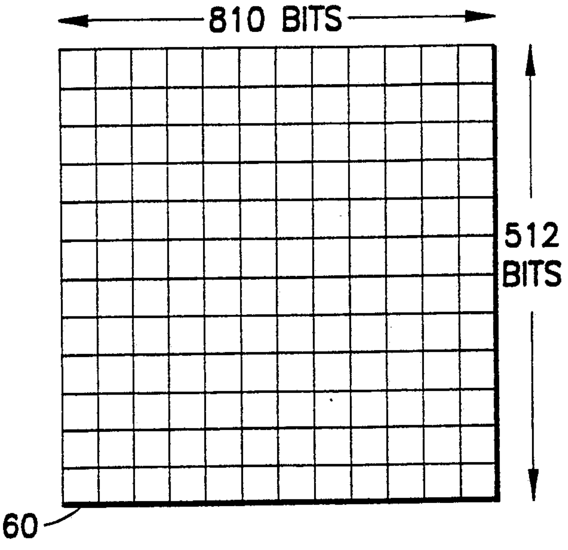


FIG. 10

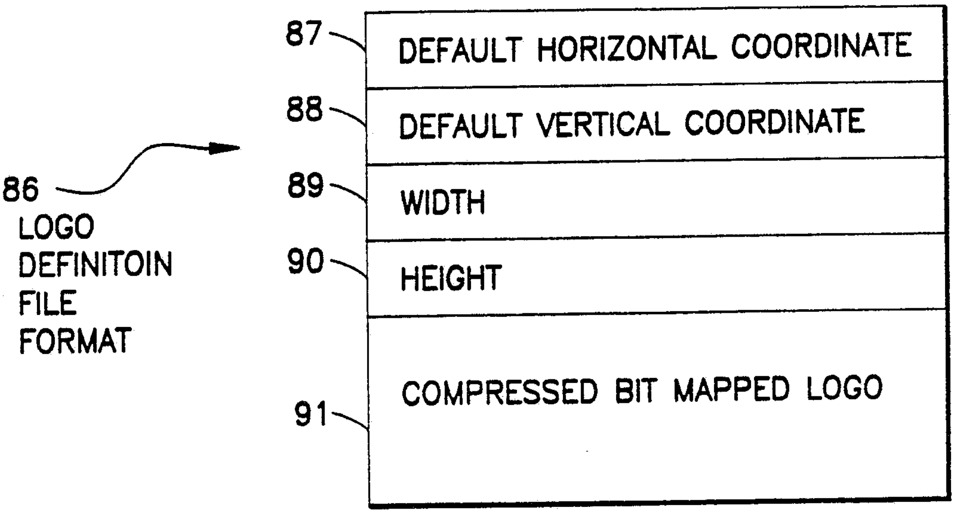

87
DEFAULT HORIZONTAL COORDINATE
88
DEFAULT VERTICAL COORDINATE
86
LOGO
DEFINITOIN
FILE
FORMAT
89
WIDTH
90
HEIGHT
91
COMPRESSED BIT MAPPED LOGO

FIG. 11

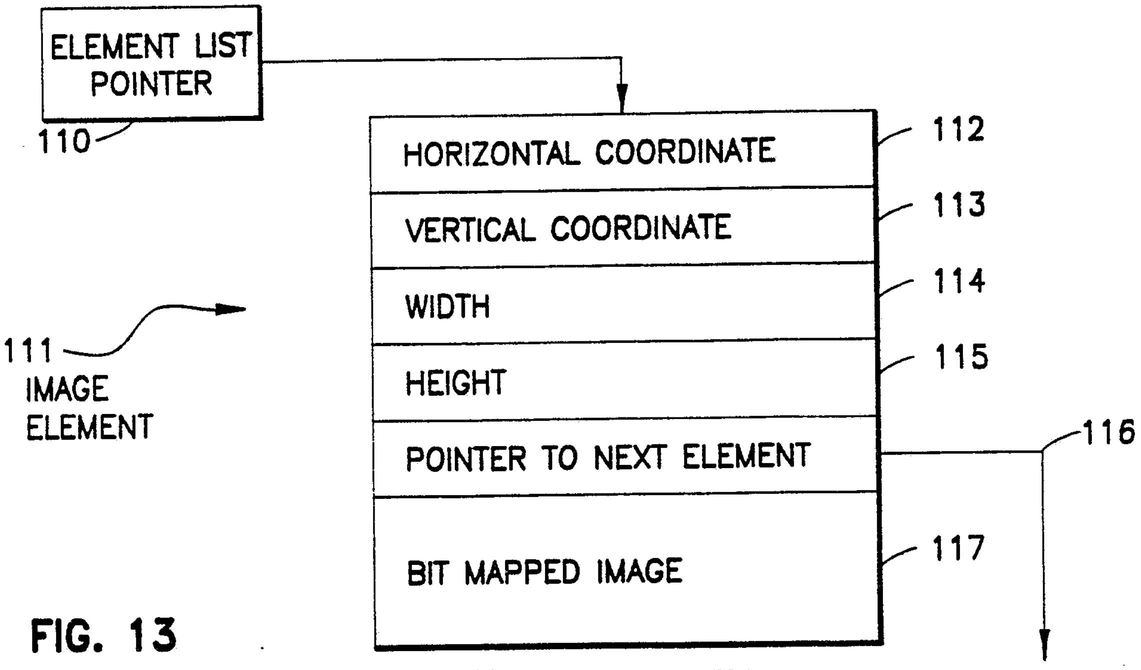


FIG. 13

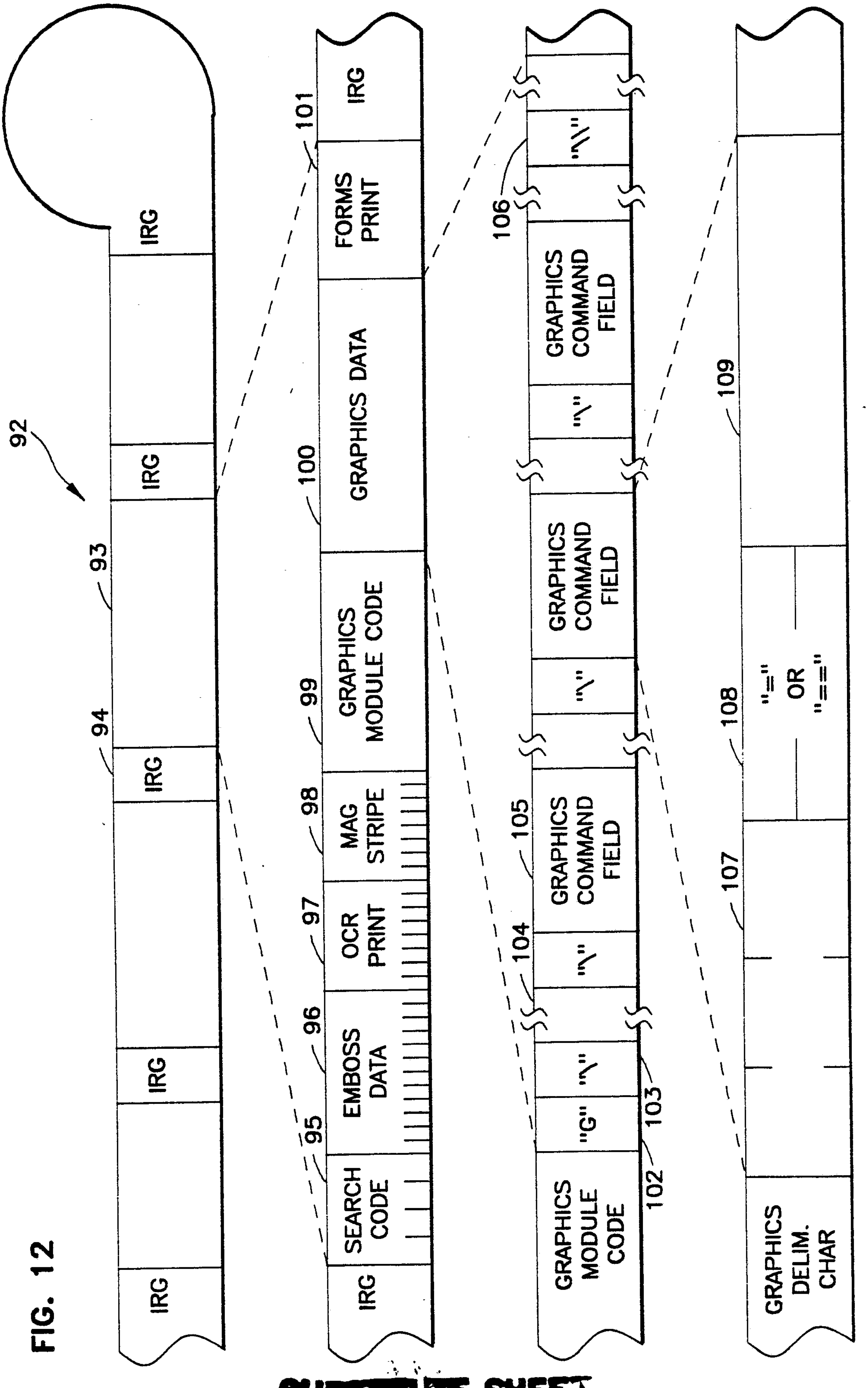
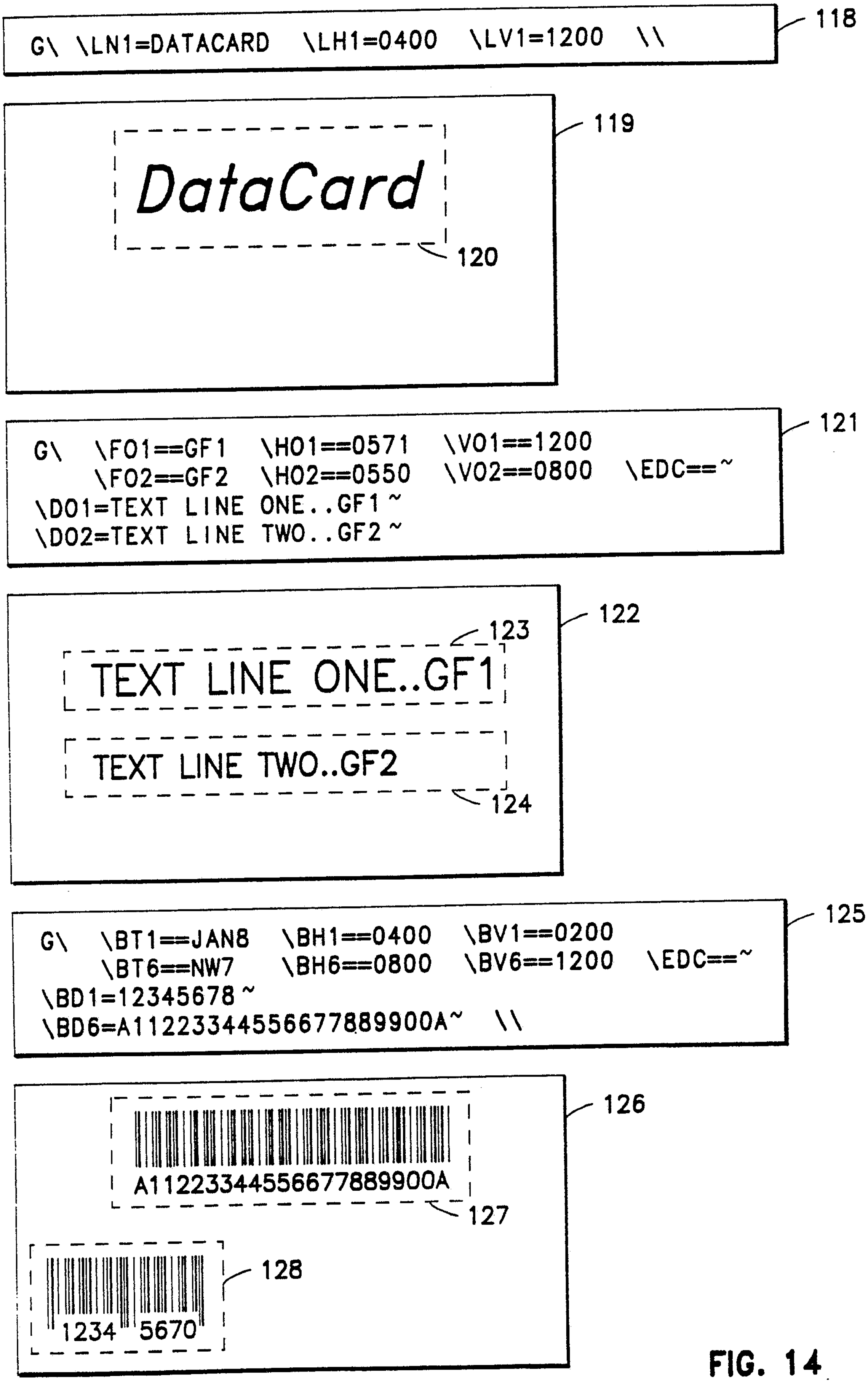


FIG. 12

SUBSTITUTE SHEET



```
G\ \LN1=DATACARD  \LH1=0400  \LV1=1200  \\
```

```
G\  \F01==GF1  \H01==0571  \V01==1200
    \F02==GF2  \H02==0550  \V02==0800  \EDC==~
\D01=TEXT LINE ONE..GF1~
\D02=TEXT LINE TWO..GF2~
```

```
G\  \BT1==JAN8  \BH1==0400  \BV1==0200
    \BT6==NW7   \BH6==0800  \BV6==1200  \EDC==~
\BD1=12345678~
\BD6=A11223344556677889900A~  \\
```

FIG. 14

LOGOS

129 - LN#=	(THIS CARD'S LOGO [#] NAME)
130 - LH#=	(THIS CARD'S LOGO [#] HORIZONTAL POSITION)
131 - LV#=	(THIS CARD'S LOGO [#] VERTICAL POSITION)

ENGLISH AND KANJI TEXT

132 - EDC==	(SET END-OF-DATA CODE FOR TEXT AND BARCODES)
133 - F##==	(SET TEXT LINE [##]'S TYPE AND FONT)
134 - H##==	(SET TEXT LINE [##]'S HORIZONTAL POSITION)
135 - V##==	(SET TEXT LINE [##]'S VERTICAL POSITION)
136 - D##=	(THIS CARD'S TEXT LINE [##] DATA)

BARCODES

137 - EDC#==	(SET END-OF-DATA CODE FOR TEXT AND BARCODES)
138 - BT#==	(SET BARCODE [#]'S TYPE)
139 - BH#==	(SET BARCODE [#]'S HORIZONTAL POSITION)
140 - BV#==	(SET BARCODE [#]'S VERTICAL POSITION)
141 - BD#=	(THIS CARD'S BARCODE [#] DATA)

FIG. 15

CHARACTER SET	CELL WIDTH	CELL HEIGHT
142 - GF1	34	48
143 - GF2	24	32
144 - KJ1	26	24
145 - KJ2	34	32
146 - KJ3	50	48
147 - KJ4	66	64
148 - KI1	26	24
149 - KI2	34	32
150 - KI3	50	48
151 - KI4	66	64

FIG. 16

INTERNATIONAL SEARCH REPORT

International Application No. PCT/US89/03940

I. CLASSIFICATION OF SUBJECT MATTER (if several classification symbols apply, indicate all) [6]

According to International Patent Classification (IPC) or to both National Classification and IPC

IPC(4): G06F 15/20
U.S.Cl. 364/518,519; 340/721

II. FIELDS SEARCHED

Minimum Documentation Searched [7]

Classification System	Classification Symbols
U.S.Cl.	364/518-521 340/721,723,724,729,750

Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched [8]

III. DOCUMENTS CONSIDERED TO BE RELEVANT [9]

Category *	Citation of Document, [11] with indication, where appropriate, of the relevant passages [12]	Relevant to Claim No. [13]
A	US,A, 4,656,602 (Berkland et al.) 07 April 1987, see the entire document.	1-18
A	US,A, 4,677,427 (Komatsu et al.) 30 June 1987, see the entire document.	1-18
A	US,A, 4,745,560 (Decker et al.) 17 May 1988 see the entire document.	1-18
A	US,A, 4,769,648 (Kishino et al.) 06 September 1988, see the entire document.	1-18
Y,P	US,A, 4,796,201 (Wake) 03 January 1989 see the entire document.	1-18

* Special categories of cited documents: [10]

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

IV. CERTIFICATION

Date of the Actual Completion of the International Search 07 NOVEMBER 1989	Date of Mailing of this International Search Report 29 NOV 1989
International Searching Authority ISA/US	Signature of Authorized Officer Heather R Herndon Heather R. Herndon