

— CONTENTS —

	U/PROM
PROM FORMATTER	1
Activation of the PROM Formatter	1
PROM WRITER FORMATS	3
BNPF	3
B10F	4
HEXADECIMAL	5
BINARY	6
Performance Boards of Various Companies	7
PROM FORMATTER COMMANDS	9
File Input/Output Commands	9
Y (Yank file) Command	9
S (Save file) Command	9
CY (Yank disk) Command	10
CS (Save disk) Command	10
Formatting Commands	11
P (Punch) Command	11
R (Read) Command	11
Other Commands	12
M (Memory dump/modify) Command	12
V (Verify) Command	13
\ (FDOS) Command	13
# (Change printer mode) Command	13
& (Clear) Command	13
? (Disp free area) Command	14
! (Return) Command	14
Format Commands	14
PROM FORMATTER COMMANDS & MESSAGES	16
	U/PLOTTER
EXAMPLE OF PLOTTER CONTROL APPLICATION	1
Interface Card	1
Plotter Control Program	1
1. Conditions for linkage with a BASIC program	1
2. Linkage conditions when an error occurs	2
3. Use of external subroutines	2
4. Plotter control codes	2
5. Program outline	3
Command Table	6
Outline of the BASIC Program	7
Sample Program (Plotter Control Routines)	10
BASIC Main Program	19

FROM

FROM FORMATTER 1

Activation of the FROM Formatter 1

FROM WRITER FORMATS 2

BNPF 3

BIOP 4

HEXADECIMAL 5

BINARY 6

Performance Boards of Various Computers 7

FROM FORMATTER COMMANDS 9

File Input/Output Commands 9

Y (Yank file) Command 9

S (Save file) Command 9

CY (Yank disk) Command 10

CS (Save disk) Command 10

Formatting Commands 11

F (Punch) Command 11

R (Read) Command 11

Other Commands 12

M (Memory dump/modify) Command 12

V (Verify) Command 13

/ (FDOS) Command 13

(Change printer mode) Command 13

& (Clear) Command 13

? (Disp free space) Command 14

! (Print) Command 14

Format Commands 14

FROM FORMATTER COMMANDS & MESSAGES 16

EXAMPLE

EXAMPLE OF PLOTTER CONTROL APPLICATION 1

Interface Card 1

Plotter Control Program 1

1. Conditions for linkage with a BASIC program 1

2. Linkage conditions when an error occurs 2

3. Use of external subroutines 2

4. Plotter control codes 2

5. Program outline 3

Command Table 6

Outline of the BASIC Program 7

Sample Program (Plotter Control Routine) 10

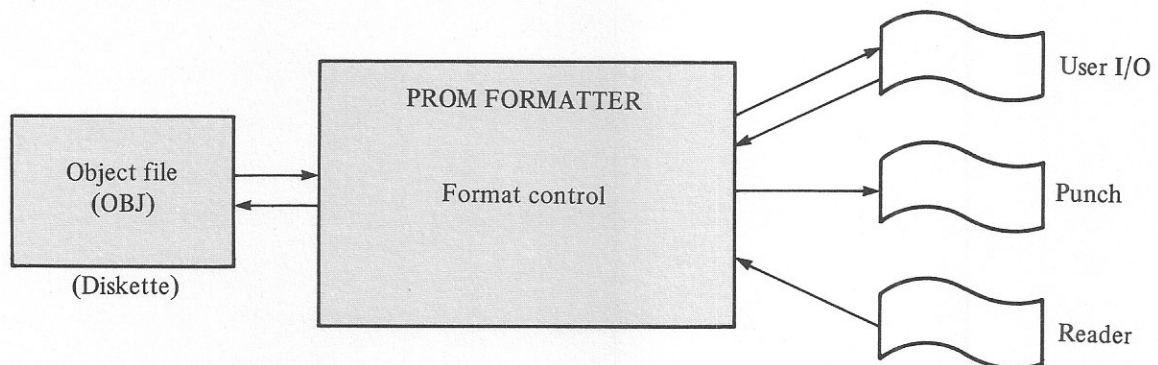
BASIC Main Program 10

PROM FORMATTER

The rapid advances in LSI technology have allowed the functions of a computer's CPU to be concentrated onto a single semiconductor chip. These microprocessors are becoming ever more sophisticated, while at the same time they are becoming less expensive. As a result, the range of fields in which microprocessors are being utilized is growing rapidly. One subject of great importance to the development of new device applications is that of developing efficient application programs; it is not too much to say that the quality of the application program determines how well a newly developed device performs.

On the other hand, developments in LSI technology have also stimulated efforts to develop low cost, large capacity memory elements (RAM and ROM). The increased availability of PROMs which are erasable with ultraviolet rays has had a particularly strong influence on the development of devices which incorporate microprocessors.

The procedure which is most suitable for efficiently developing application programs is to create an object file from a source file created through assembly programming using an assembly language, then to reassemble it after debugging. The function of the PROM formatter is to load one or more object programs created with the assembler and linker, then to output it to a paper tape punch after converting it to PROM writer format.



Functions of the PROM formatter

It also allows programs which are written in different formats to be input from a reader for storage on diskette and enables conversion of programs to the required format for output on paper tape or the like.

—Activation of the PROM Formatter—

Entering PROM CR while in the FDOS command entry mode activates the PROM formatter. Commands may be entered as soon as activation is completed.

The following formats are provided for in the PROM formatter:

1. BNPF

- Britronics
- Intel
- Takeda Riken

2. B10F

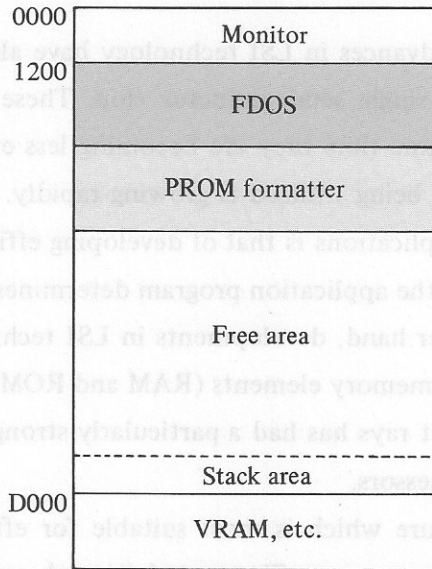
- Takeda Riken

3. HEXADECIMAL

- Britronics
- Takeda Riken
- Minato Electronics

4. BINARY

- Britronics



PROM formatter memory map



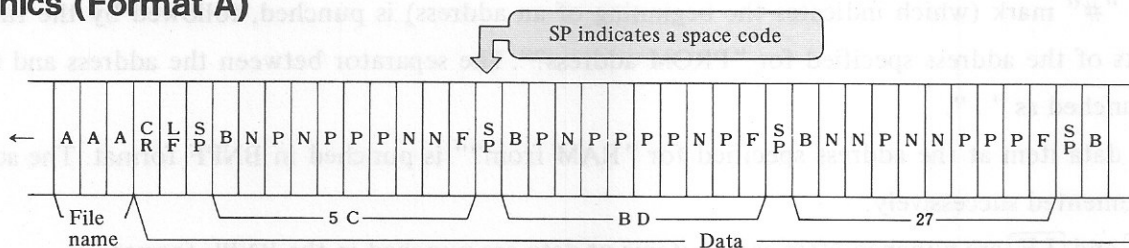
PROM WRITER FORMATS

PROM writers are provided in many formats by different companies. This section discusses forms which are converted by the PROM formatter; refer to the individual PROM writer manuals for details.

The examples in the figures include the file name "AAA", the address "0000", and the data "5C", "BD" and "27". The leader section for the start of punched output and the trailer section for the end of punched output are created automatically.

—BNPF—

Britronics (Format A)

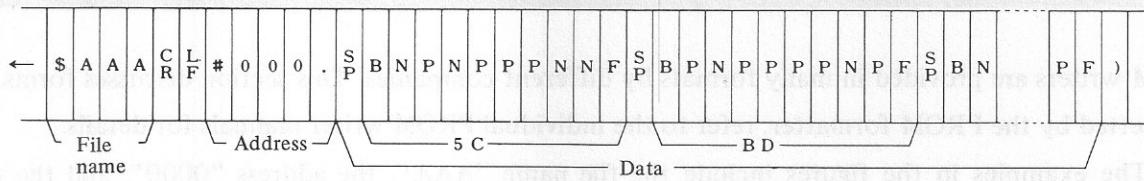


- The file name is punched in ASCII code (if one is specified). (Using the character "B" as a file name will result in incorrect identification of the beginning of data.)
- **CR** and **LF** are punched in ASCII code.
- The space code (20H) and the byte of data at the address specified for "RAM from?" are punched in BNPF format. The address is incremented successively.
- **CR** and **LF** are punched after each 6 items of data are punched in the BNPF format.
- Punching is performed in BNPF format up to the address specified for "to?".

Intel (Format D)

- This is the same as the Britronics format. The BNPF format is one which has a relatively high degree of standardization; thus, the PROM formatter can also be used with devices other than those which are discussed in this manual.

Takeda Riken (Format E)

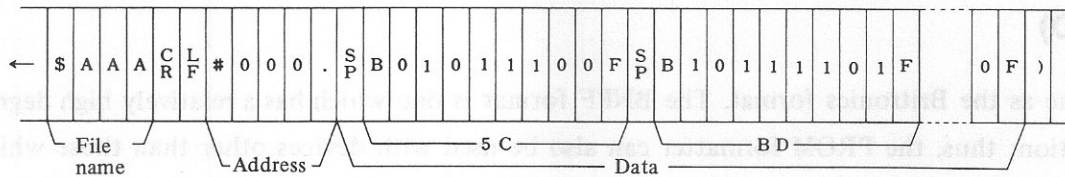


- The "\$" mark, which denotes the file name, is punched in ASCII code.
- The file name is punched in ASCII code (if one is specified).
- **CR** and **LF** are punched. The "\$" mark is regarded as denoting the beginning of a comment statement; the end of a comment statement is denoted with an **LF** code.
- The "#" mark (which indicates the beginning of an address) is punched, followed by the first three digits of the address specified for "PROM address?". The separator between the address and the data is punched as ".".
- The data item at the address specified for "RAM from?" is punched in BNPF format. The address is incremented successively.
- **CR** and **LF** are punched after each 6 items of data are punched in the BNPF format.
- A tape leader stop mark ")" is punched after the data has been punched up to the address specified for "to?".

Note: Care must be taken to ensure that characters which act as control characters (B, :, \$, #, etc.) are not used when a file name is specified. (Otherwise, incorrect operation will result.)

—B10F—

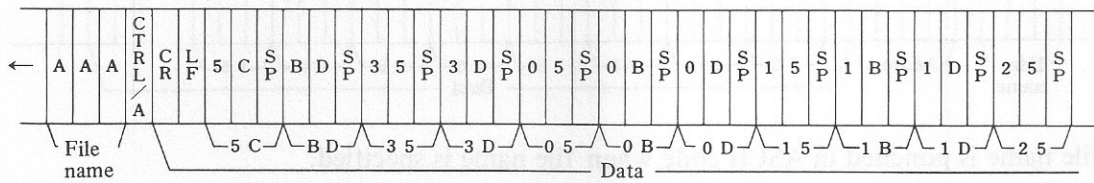
Takeda Riken (Format F)



- Except for the NP section, this is the same as Takeda Riken's BNPF format.
- The B10F format corresponds to the BNPF format in that 1 = P and 0 = N.

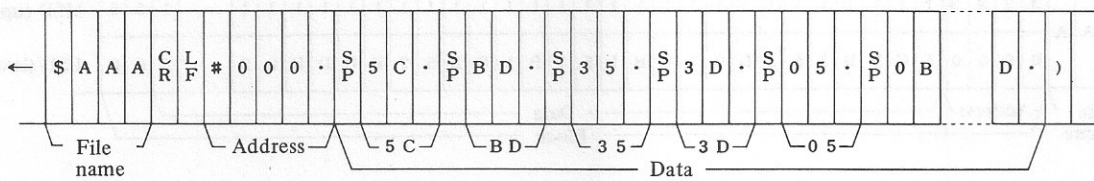
—HEXADECIMAL—

Britronics (Format B)



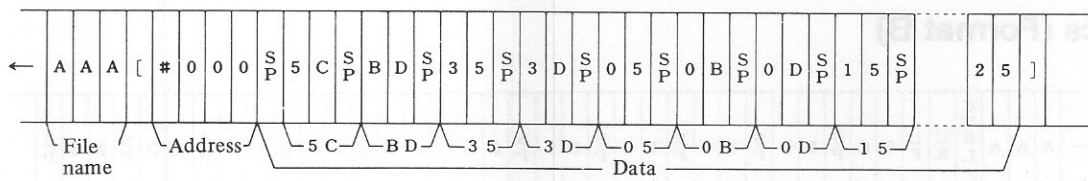
- The file name is punched in ASCII code (if one is specified).
- The "CTRL/A" mark (01H) indicating the beginning of data is punched.
- **CR** and **LF** are punched.
- The data item at the address specified for "RAM from?" is punched as a 2-digit ASCII code, then a space code is punched.
- **CR** and **LF** are punched after 16 bytes of data have been punched.
- Data is punched up to the address specified for "to?".

Takeda Riken (Format G)



- The "\$" mark, which denotes the file name, is punched in ASCII code.
- The file name is punched in ASCII code (if one is specified).
- After **CR** and **LF** are punched (followed by the address specification mark "#" and 3 digits of the address specified for "PROM address?"). The separator "." is punched.
- The space code is punched, followed by the 2-digit ASCII code for the data at the address specified for "RAM from?". The separator "." is punched after the data item.
- **CR** and **LF** are punched after 16 bytes of data have been punched.
- Data is punched up to the address specified for "to?", at which point the tape leader stop mark ") " is punched.

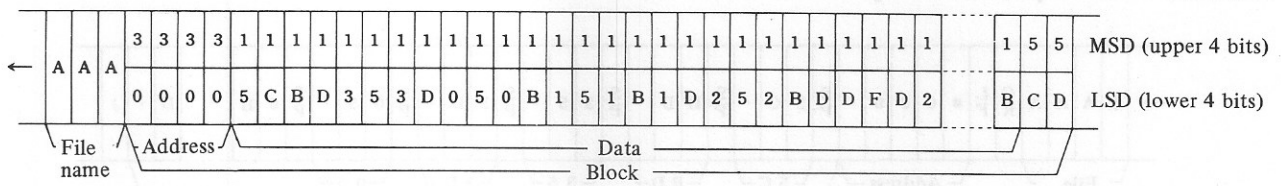
Minato Electronics (Format H)



- The file name is punched in ASCII code when file name is specified.
- The start-of-data mark "[" is punched.
- The address designation mark "# " is punched, followed by a 3-digit ASCII code for the address specified for "PROM address?".
- A space code is punched, then the data at the address specified for "RAM from?" is converted to a 2-digit ASCII code and punched.
- 16 combinations of space codes and data items are punched, then **CR** and **LF** are punched.
- The end of data mark "]" " is punched after data has been punched up to the address specified for "to?".

—BINARY—

Britronics (Format C)



- In the binary format, the 4-bit mark section and the 4-bit data section are expressed together as one character (8 bits). The mark section is punched as the upper 4 bits of the paper tape, while the data section is punched as the lower 4 bits.
- The file name is punched in ASCII code (if one is specified). Specifications which result in a "3" in the upper 4 bits of the ASCII code file name are not permitted. Such specifications will result in incorrect operation, since incorrect determination that the lower 4 bits of the file name are an address will result.
- Three binary digits for the address specified for "PROM address?" are punched in the lower 4 bits. The address designation mark ("3") is punched in the upper 4 bits.
- A data mark ("1") is punched in the upper 4 bits and data at the address specified for "RAM from?" is punched in the lower 4 bits.
- Data is punched 4 bits at a time (with the upper and lower 4 bits punched in alternation) up to the address specified for "to?".
- Check sum marks ("5") are punched in the upper 4 bits, followed in alternation by check sum data in the lower 4 bits.

—Performance Boards of Various Companies— (Note: Consult the various manufacturers for details.)

a) Intel

2716
 2732
 8748/8741
 3621, 3602, 3622, 3602A, 3622A, 3604, 3624, 3604A, 3624A, 3605, 3625, 3605A, 3625A, 3628, 3608,
 3604AL-6, 3604AL
 8702A/1702A
 8708/8704/2708/2704
 8755A

b) Britronics

Company	Element
Intel	3602 A / 22 A, 3604 A / 24 A, 3604 A L / 24 L, 3605 / 25, 3608 / 28
Intersil	5600 / 10, 5603 A / 23, 5604 / 24, 5605 / 25
Fujitsu	7055, 7051, 7052, 7058, 7053, 7059, 7054, 7057
Monolithic Memory	5330 / 6330, 5331 / 6331, 5300 / 6300, 5335 / 6335, 5336 / 6336, 5308 / 6308, 5309 / 6309, 53134 / 63134, 53135 / 63135, 5305 / 6305, 5306 / 6306, 53137 / 63137, 53141 / 63141, 5340 / 6340, 5341 / 6341, 5348 / 6348, 5349 / 6349, 5350 / 6350, 5351 / 6351, 5352 / 6352, 5353 / 6353, 5380 / 6380, 5381 / 6381, 5384 / 6384, 5385 / 6385, 5386 / 6386, 5387 / 6387
Harris	7602 / 03, 7610 A / 11 A, 7620 A / 21 A, 7640 A / 41 A, 7640 AR / 41 AR, 7642 / 43, 7644, 7646 R / 47 R, 7648 / 49, 7608, 7680 / 81, 7680 R / 81 R, 7680 P / 81 P, 7680 RP / 81 RP, 7683, 7684 / 85, 7684 P / 85 P, 7686 / 87, 7686 R / 87 R, 7686 P / 87 P, 7686 RP / 87 RP
Fairchild	93417 / 27, 93436 / 36, 93438 / 48, 93452 / 52
National Semiconductor	54 / 74 S 387, 54 / 74 S 287, 54 / 74 S 470, 54 / 74 S 471, 54 / 74 S 570, 54 / 74 S 571, 77 / 87 S 295, 77 / 87 S 296, 54 / 74 S 473, 54 / 74 S 472, 54 / 74 S 572, 54 / 74 S 573
NEC	403 D, 406 D
Raytheon	29660 / 61, 29600 / 01, 29612 / 13
Signetics	82 S 114 / 115, 82 S 126 / 127, 82 S 130 / 131, 82 S 140 / 141, 82 S 136 / 137, 82 S 180 / 181, 82 S 2708, 82 S 184 / 185, 82 S 190 / 191
Texas Instruments	54 / 7488A, 54 / 74 S / 88, 54 / 74 S 288, 54 / 74 S 470, 54 / 74 S 71, 54 / 74 S 73, 54 / 74 S 72, 54 / 74 S 75

c) Minato Electronics

Adaptable to all PROMs.

d) Takeda Riken

MOS Type

Element	Bit configuration, capacity (words x bits = capacity)	Maker name											
		Okai Denki	Toshiba	NEC	Hitachi	Fujitsu	Mitsubishi	Intersil	Intel	Texas Instruments	Fairchild	AMD	SIGNE
MOS	256 x 8 = 2048			μPD454D⑩	HN351702A⑦	MB8503 MB8513	M5L1702⑦			1602A ⑦ 1702A		AM1702A⑦	1702A ⑦
	512 x 4 = 2048		TMM121C TMM121C-1										
	512 x 8 = 4096												
CMOS	1024 x 8 = 8192	MSM3758⑥	*TMM322C③	μPD458D⑩	HN462708③	MB8518③	M5L2708③			2704 ④ 2708 ④ 2758 ④		AM2708 ④	2708 ④
	2048 x 8 = 16384		TMM323C④	μPD2716D⑤	HN462716⑤	MB8516⑤	M5L2716⑤			2716 ⑤ 2732	TMS2708④ TMS2716④ TMS2516④ TMS2532		
	4096 x 8 = 32768												
	512 x 8 = 4096												
Compound MOS	1024 x 4 = 4096												
	1024 x 8 = 8192						M58460 S			8755			

Bipolar Type

Element	Bit configuration, capacity (words x bits = capacity)	Maker name														
		Nihon Denki	Hitachi	Fujitsu	Mitsubishi	Intersil	Intel	Texas Instruments	Harris	Fairchild	Raytheon	MMI	AMD	SIGNE		
Bipolar	32 x 8 = 256			MB7051 MB7056	M54730	1M5600 1M5610				HM7602 HM7610A HM7611A HM7603				6330-1 6331-1	AM27S18 AM27S19	82S23 82S123
	256 x 4 = 1024	μPB403D②		MB7052 MB7057②	M54700	1M5603A 1M5623				HM7610A HM7611A HM7625R HM7629	93417 93427	29662 29663 29660 29661		6300-1 6301-1	AM27S20 AM27S21	82S126 82S129
	256 x 8 = 2048															
Bipolar	512 x 4 = 2048			MB7053 MB7058		1M5604 1M5624				HM7620 HM7621 HM7621A	93436 93446	29612 29613		6305-1 6306-1	AM27S12 AM27S13	82S130 82S131
	512 x 8 = 4096	μPB405D μPB425D		MB7054 MB7059		1M5605 1M5625				HM7640 HM7641 HM7640A HM7641A HM7647R HM7648 HM7649	93438 93448		6341-1 6340-1	AM27S15	82S115 82S140 82S141	
	1024 x 4 = 4096	μPB406D⑤ μPB426D				1M5606 1M5626				HM7642 HM7643 HM7644 HM7642P HM7643P HM7680 HM7681 HM7680R HM7681R HM7680RP HM7681RP	93452 93453		6350-1 6352-1 6353-1	AM27S32 AM27S33	82S136 82S137	
	1024 x 8 = 8192	μPB417D⑥ μPB427D		MB7055 MB7060		3608A 3628A				HM7682 HM7683 HM7684 HM7685 HM7686 HM7687 HM7686P HM7687R HM7688R HM7689RP HM7690RP	93450 93451		6380-1 6381-1 6384-1 6385-1 6386-1 6387-1 63133-1		82S180 82S181 82S2708	
	2048 x 4 = 8192															
Compound MOS	2048 x 8 = 16384									HM7616 HM76160 HM76161						82S190 82S191

Elements annotated with the same figures in circles can be used with the same performance boards.

PROM FORMATTER COMMANDS

—File Input/Output Commands—

Y (Yank file) Command

Reads the object (OBJ) file specified by the file name into the free area.

* YCHARAGEN

RAM from? 8000 to 87FF

Reads in CHARAGEN.OBJ

Specifies read-in from address 8000
(read up to address 87FF)

- File name can be specified by entering a Y (Yank file command) when "*" appears to indicate that command entry is awaited.
- Specify the starting address of the file to be read in as a 4-digit hexadecimal number. (Reading will start at the address specified regardless of the actual data address of the object file.)
- The last address read is displayed when file read-in is completed.

Caution: The address specified for read-in must be in the free area.

S (Save file) Command

Writes the specified program (or data) in the free area onto a diskette.

* STEST# 2

RAM from? 8400 to? 87E7

exec? 1300 data? 1300

Output program (data) to TEST# 2.OBJ

Output program (data) from address 8400 to 87E7

Execute address 1300, data address 1300

- Another file name can be specified by entering an S (Save file) command when "*" appears to indicate that command entry is awaited.
- The addresses of the memory block in the free area which is to be output are specified with 4-digit hexadecimal numbers.
- The execute address and data address of the object (OBJ) file created are specified with 4-digit hexadecimal numbers. The data address is the address to which the program (data) is to be reloaded into memory by a later RUN or LOAD command. The execute address is the address from which a program reloaded into memory is to be executed. Specify 0000 when either of these addresses is not necessary.

CY (Yank disk) Command

This command loads data in units of 256 bytes from the specified sector(s) of the specified track on the floppy disk in the specified drive into RAM.

```
* CY  CR
drive? 1
track, sect? 0301
byte size? 0100 adrs? 7000
```

- Enter the CY command when "*" appears to indicate that command entry is awaited.
- Specify the number of disk drive containing the floppy disk storing the data to be loaded.
- Enter the track number and the first of the sector numbers in which the data to be loaded is stored as 4 continuous hexadecimal digits.
- Enter the 4 hexadecimal digits which indicate the number of 256 byte units of data, then enter the starting address (4 hexadecimal digits) of the RAM area into which the data is to be loaded.
- The 4-digit hexadecimal number which indicates the amount of data is constructed as shown below.

XX XX

— No numbers other than 0 should be entered in these two places (otherwise, the number of data units specified in first two places will be affected).

— Indicate the number of 256-byte units of data to be loaded.

Ex.) 512 bytes of data are loaded when 0200 is entered (0101 is also interpreted as 0200).

CS (Save disk) Command

This command saves data in 256-byte units from RAM memory in the specified sector(s) of the specified track of the floppy disk in the specified disk drive.

```
* CS  CR
drive? 2
track, sect? 3002
byte size? 0100 adrs? 7000
```

- Enter the CS command when "*" appears to indicate that command entry is awaited.
- Specify the number of the disk drive containing the floppy disk on which the data is to be saved.
- Enter the track number and the first of the sector numbers in which the data is to be saved as a 4-digit hexadecimal number.
- Enter the 4-digit hexadecimal number which indicates the number of 256 byte units of data, then enter the starting address (4 hexadecimal digits) of the RAM area from which the data is to be saved.
- The 4-digit hexadecimal number which indicates the amount of data is constructed in the same manner as for the CY command.

For example, 0101 is interpreted as 0200.

- The track number must be 3 or greater. Great care must be taken not to destroy existing data on the floppy disk.

—Formatting Commands—

P (Punch) Command

Punches data in the free area in the specified format.

* P <input type="text" value="CR"/>	Punch command
filename? CHARAGEN <input type="text" value="CR"/>	File name assigned to the paper tape to be punched.
format? C <input type="text" value="CR"/>	Format C
RAM from? 8000 to? 87FF	Addresses 8000 to 87FF in the free area
PROM address? 0000	PROM write address 0000

- Enter the P (Punch) command when "*" appears to indicate that command entry is awaited.
- Next, the file name is specified. This is not the file name which is included on the diskette, but the name which is to be punched at the beginning of the tape. Refer to the explanations of the various formats for details. When no file name is needed, enter only .
- Next, specify the conversion format (A~H) and enter .
- Specify the starting and ending addresses of the memory block in the free area which is to be output with 4-digit hexadecimal numbers.
- Finally, specify the PROM write address. (This step may not be required, depending on the format.)

The P command described above outputs formatted data to a PTP device. (More precisely, \$PTP/LF is used as the output device.)

The PROM FORMATTER can also output converted format data to devices other than PTP (including user I/O and diskette).

(Ex. 1)	*P\$USR1 <input type="text" value="CR"/>	Outputs to user I/O
(Ex. 2)	*PXYZ <input type="text" value="CR"/>	Outputs file name XYZ.ASC to the diskette.
(Ex. 3)	*P\$PTP/PE/LF <input type="text" value="CR"/>	Adds even parity to PTP, affixes <input type="text" value="LF"/> after <input type="text" value="CR"/> and outputs the file.

As an application, data may be sent directly to the PROM writer by creating hardware and user routines for its online interface.

R (Read) Command

This command reads in data formatted in the BNPF, HEXADECIMAL or other format from a paper tape reader.

* R <input type="text" value="CR"/>	Read command
format? C <input type="text" value="CR"/>	Format C
RAM from? 8000 to 83FF	Addresses in the free area into which data is to be read.
filename PROM#2	

- Enter the R (Read) command when "*" appears to indicate that command entry is awaited.
- Next, specify the format of the data to be read.
- Finally, specify the starting address of the free area into which the data is to be read with a 4-digit hexadecimal number.
- The last data address and the file name are displayed after the read is completed and entry of the next command is awaited.
- With this PROM writer format, it may not be possible to read tapes punched using other programs because of the need to maintain a certain minimum degree of redundancy.

The R command described above reads in formatted data from a PTR device, (More precisely, \$PTR is used as the input device.)

The PROM FORMATTER can also read in converted format data from devices other than PTR (including user I/O and diskette).

- | | | | |
|---------|------------|---------------------------------|--|
| (Ex. 1) | *R\$USR2 | <input type="text" value="CR"/> | Inputs from user I/O |
| (Ex. 2) | *RXYZ | <input type="text" value="CR"/> | Inputs from XYZ.ASC on a diskette |
| (Ex. 3) | *R\$PTR/PE | <input type="text" value="CR"/> | Inputs data with even parity affixed from PTR. |

—Other Commands—

M (Memory dump/modify) Command

This command is used to display and modify the contents of the free area.

* M	<input type="text" value="CR"/>	M command
RAM from?	7000 to?	7014
7000	ED 73 C8 5F 01 C6 5F 09	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
7008	60 22 D8 CD 3E 28 CA 27	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
7010	28 22 DE 26 CD	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

- Enter the M (Memory dump/modify) command when "*" appears to indicate that command entry is awaited.
 - Next, specify the starting and ending addresses in the free area of the data to be displayed with 4-digit hexadecimal numbers.
 - The PROM formatter, divides data in the specified addresses into 8-byte segments and displays the address (4 hexadecimal digits), the 8 bytes of data (as groups of 2 hexadecimal digits) and the 8 corresponding ASCII characters in that order.
- However, when the corresponding ASCII character cannot be displayed, a "." is displayed in its place. Further, data is printed in 16 byte segments when the printer is used with the "#" command.
- Execution of the M command can be suspended or resumed by pressing . A switch can be made to the command entry mode by pressing .
 - If no change is required in data displayed using the M command, just press . When a change is required, move the cursor to the position where the change is to be made and press after entering the 2 new hexadecimal digits. (The change is made when is pressed.) After data modification is completed, move the cursor to an empty line and press to return to the command wait state.

— Data can also be changed using the cursor when display is halted with `SPACE`. In this case, display is resumed when the cursor is moved to an empty line and `CR` is pressed.

Caution: Data is only printed when the printer is used with the "# " command; modification of data is not possible in this case.

V (Verify) Command

Reads data formatted in BNPF, HEXADECIMAL and so forth from the paper tape reader and compares it with the contents of the RAM free area.

* V <code>CR</code>	Verify command
format? C <code>CR</code>	Format C
RAM from? 8000 to 8615	The start and end addresses of the area containing
filename ABC	the data to be compared.
Verify OK.	

- Enter the V (Verify) command when "*" appears to indicate that command entry is awaited.
- Specify the format of data to be read.
- Specify the start address (4-digit hexadecimal) of the area containing the data to be compared.
- The end address of the data read is displayed and "Verify OK." is displayed when the data read matches the data compared. If not, the end address is not displayed and "Verify error." is displayed.

\ (FDOS) Command

This command invokes the specified built-in FDOS command. Command entry is awaited after execution of the FDOS command.

```
* \ DIR CR
```

- Enter the \ command when "*" appears to indicate that command entry is awaited.
- Next, specify the built-in FDOS command and press the `CR` key.
- The PROM formatter executes the built-in FDOS command, then awaits entry of the next PROM formatter command.
- The XFER and EXEC commands cannot be executed. The RUN command cannot be executed if the program executed by the RUN command is too long.

(Change printer mode) Command

This command starts and stops output to the printer. Printer output is OFF when the PROM formatter is activated, and is changed from ON to OFF to ON each time the "# " command is executed.

When printer output is ON, data is printed almost as it appears on the display screen.

& (Clear) Command

Buries the entire free area in hexadecimal code FFH.

? (Disp free area) Command

Displays the free area.

! (Return) Command

Terminates the PROM formatter and returns to FDOS.

—Format Commands—

Format commands are commands entered when "format?" is displayed during execution of the P, R or V commands. Selecting one of these commands during execution of the P command determines whether data is to be punched in BNPF, HEXADECIMAL or other format. Failure to specify the correct format command during execution of the R command will result in failure to correctly read the program into the free area.

A Command

— Used to specify the Britronics BNPF format. The control character "B" may not be used when the file name is specified.

B Command

— Used to specify the Britronics HEXADECIMAL format.

C Command

— Used to specify the Britronics BINARY format. Numerals and the codes (:; < = > ?) may not be used when the file name is specified.

— The message "PROM address?" is displayed during execution of the P command to request specification of the PROM loading address; specify it as a 4-digit number.

— Check sums are written following the data (with the P command).

— Data from the address specification to the check sums constitutes one block; if data is to be loaded into an address which has been skipped, the operation must be divided into two or more parts. This also applies when two or more blocks are read in with the R command.

D Command

— This command is used to specify the Intel BNPF format. The character "B" cannot be used in the file name.

E Command

— This command is used to specify the Takeda Riken BNPF format. The character "B" may be used in the file name.

— A file is a block which begins with "\$" and ends with ")".

— The message "PROM address?" is displayed during execution of the P command to request specification of the PROM loading address; specify it as a 4-digit number.

— If two or more blocks are to be read out or written in, the operation must be divided into two or more parts.

F Command

- This command is used to specify the Takeda Riken B10F format. The character "B" may be used in the file name.
- A file is a block which begins with "\$" and ends with ")".
- The message "PROM address?" is displayed during execution of the P command to request specification of the PROM loading address; specify it as a 4-digit number.
- If two or more blocks are to be read out or written in, the operation must be divided into two or more parts.

G Command

- This command is used to specify the Takeda Riken HEXADECIMAL format.
- A file is block which begins with "\$" and ends with ")".
- The message "PROM address?" is displayed during execution of the P command to request specification of the PROM loading address; specify it as a 4-digit number.
- If two or more blocks are to be read out or written in, the operation must be divided into two or more parts.

H Command

- This command is used to specify the Minato Electronics HEXADECIMAL format.
- The start-of-data symbol "[" may not be used in the file name.
- The message "PROM address?" is displayed during execution of the R command to request specification of the PROM loading address; specify it as a 4-digit number.
- Denote the end of data with the symbol "]"

PROM FORMATTER COMMANDS & MESSAGES

COMMAND		OPERATION
File Input/ Output commands	Y (Yank)	Loads a program (data) from the diskette into the free area.
	S (Save)	Saves the program (data) in the free area on diskette.
	CY (Yank disk)	Loads data in 256-byte units from the specified sector(s) of the specified track on the diskette into RAM.
	CS (Save disk)	Saves data in 256-byte units from RAM memory in the specified sector(s) of the specified track of the diskette.
Format commands	P (Punch) R (Read)	Punches the specified contents of the free area in the specified format. Reads in a paper tape punched in the format specified.
Other commands	M (Memory) V (Verify) \ (FDOS) # & (Clear) ? ! (Return)	Displays and modifies data in the free area. Reads data from the paper tape reader and compares it with the contents of the RAM free area. Executes the specified built-in FDOS command. Switches the list mode for listing on a printer. Buries all data in the free area in hexadecimal code FFH. Displays the starting and ending addresses of the free area. Returns control to FDOS.

Error message	Error content	Related command
memory protection il command il data check sum \$ LPT : not ready \$ PTP : not ready \$ PTR : not ready	An address outside of the free area was specified. The command was not entered correctly. The format specified does not match the format read. Check sum error. The printer is not ready. The paper tape punch is not ready. The paper tape reader is not ready.	Y, S, P, R, M, V R, V R, V # P R, V

See the "System Error Messages" in System Command for other error messages.

Caution:

Entry of characters other than S, Y, CS, CY, P, R, M, V, \, &, #, ? or ! will cause a return to the command wait state after the command table is displayed.

If a character other than A~H is input while "format?" is displayed and format entry awaited, the format table will be displayed, after which the format entry wait state will be reentered. A return can be made to the command wait state at this time by pressing **BREAK**.

EXAMPLE OF PLOTTER CONTROL APPLICATION

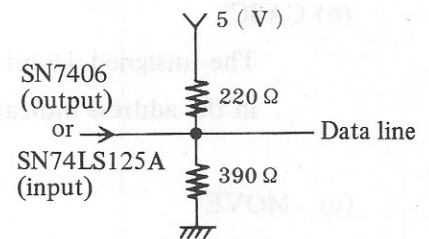
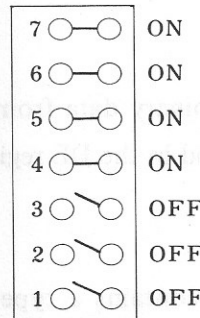
Use of the FDOS editor, assembler and BASIC compiler allows the familiar BASIC language to be used to write main programs without loss in processing speed, as long as control programs are created for control of the various devices. Another benefit is that commands developed by the user can be used as BASIC commands.

—Interface Card—

Universal interface card MZ-80IO2 is used for the interface between the MZ-80A computer and the MILOT WX4671 plotter.

The connection conditions are as shown in Figure 1.

Output port (OFH)		Input port (0EH)	
I/O card	Plotter side	I/O card	Plotter side
O 27	STROBE	I 17	Grounded on the universal interface card
O 26	DB 6	I 16	
O 25	DB 5	I 15	
O 24	DB 4	I 14	
O 23	DB 3	I 13	
O 22	DB 2	I 12	
O 21	DB 1	I 11	ERROR
O 20	DB 0	I 10	BUSY



a) Connection conditions for all input/output terminals b) Address switch settings c) Data line termination conditions

Fig. 1 Connection conditions

SN7404 is included as the data driver for the universal interface card, but ICs 16 and 17 only are changed to SN7406. All data line and status input terminations are made as shown in Figure 1-c). A 1.5 m cable can be used for this purpose.

See the universal interface card instructions for details.

—Plotter Control Program—

This section may skip if you are not interested in assembly subroutines.

1. Conditions for linkage with a BASIC program

Conditions for linkage with a BASIC program

- Command names must be externally declared with the ENT statementSIZE : ENT
- The number of parameters must be specified DEFB 1 (1 parameter)
- The parameter type must be specified. DEFB 0 (real number)
- Buffers must be specified for parameters. SE : DEFS 2 (2 bytes reserved)
- The RET instruction must be included at the end of all control routines.

The above are the linkage conditions; the processing program is written between items (d) and (e).

2. Linkage conditions when an error occurs

- (a) A subroutine is used from FDOS library RELO.LIB CALL BEERR
- (b) The error number is written. DEFB 80
- (c) The error message is written. DEFM 'PLOTTER ERROR'
- (d) The terminator is written. DEFB 0DH

This causes * ER 80:PLOTTER ERROR to be output on the display screen when a plotter error occurs.

3. Use of external subroutines

This control program uses 4 routines out of the subroutines included in FDOS library RELO.LIB. One of these is BEERR, which was shown above; the remaining three are as follows:

(a) .. INTO

16 bit binary data is set in the HL register with a sign attached when an address with a parameter is loaded in the HL register and called. All registers except the AF register are protected in the event of an overflow if the carry flag is set.

(b) CASC'

The unsigned 16-bit binary data from the HL register is converted to ASCII code and stored in the address indicated in the DE register, then 0DH is set.

(c) . MOVE'

When the parameter contains a type 1 string and an address with data is loaded in the HL register and called, a type 2 string is set in the address indicated by the DE register and 0DH is set.

See the "LIBRARY/PACKAGE" instructions for details on all subroutines.

4. Plotter control codes

All data for the MILOT WX4671 currently used is in 7-bit ASCII code. Input statuses include the BUSY signal and the ERROR signal. Data output is possible when the BUSY signal goes low, and the data is taken in on the plotter side when the STROBE signal is output.

" , " (that is, 2CH) is used as the data delimiter and 03H~0DH are valid as data terminators. However, only 0AH will be accepted when an error occurs; an error condition is not cleared by any data terminator other than 0AH.

At the port on the MZ-80 side, 0EH is used for the status (that is, as the input for the BUSY and ERROR signals) and 0FH is used for the data and STROBE signal output.

5. Program outline

Linkage conditions for a BASIC program have already been indicated; however, string type becomes applicable in the parameter type specification with the 80H. Moreover, parameter types and parameter buffers must be added depending on the number of parameters; the steps described in subparagraphs (c) and (d) of that section are not required if the number of parameters is zero. See routines CTYPE, SIZE, PLOT, HOME and so forth of the assembly listing for this.

This illustrated using the PLOT routine as a representative example. The flowchart is as shown in Figure 2 (pages 4 and 5).

(a) Data output

Although subroutines COUT, PLOT1 and DOUT are used, DOUT is the one which actually outputs the data. With DOUT, the data set in the accumulator is output to the plotter with the STROBE signal if the BUSY signal of the plotter is LOW. If BUSY is HIGH the routine repeats a loop.

With COUT and PLOT1, the data at the address indicated in the HL register is loaded in the accumulator and then DOUT is executed; this is repeated until 0DH is output. After 0DH is output, a check is made for plotter errors and a jump is made to the error routine if any are found. Note that continuation of program execution is possible if ON ERROR processing is provided on the BASIC side.

(b) Data conversion routine

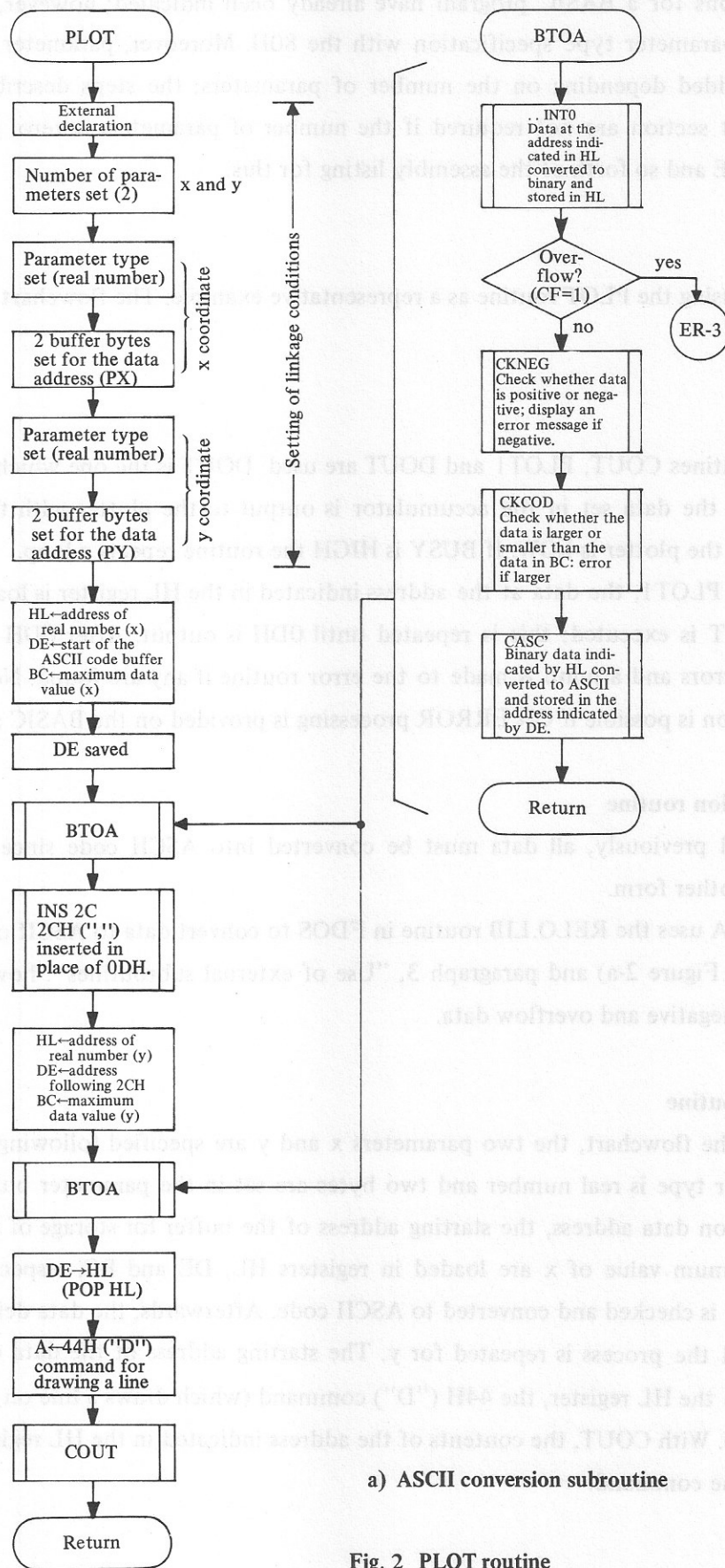
As was indicated previously, all data must be converted into ASCII code since the plotter will not accept data in any other form.

Subroutine BTOA uses the RELO.LIB routine in FDOS to convert data to ASCII code. This is as shown in the flowchart in Figure 2-a) and paragraph 3, "Use of external subroutines"; however, checks are also made for positive, negative and overflow data.

(c) PLOT x, y routine

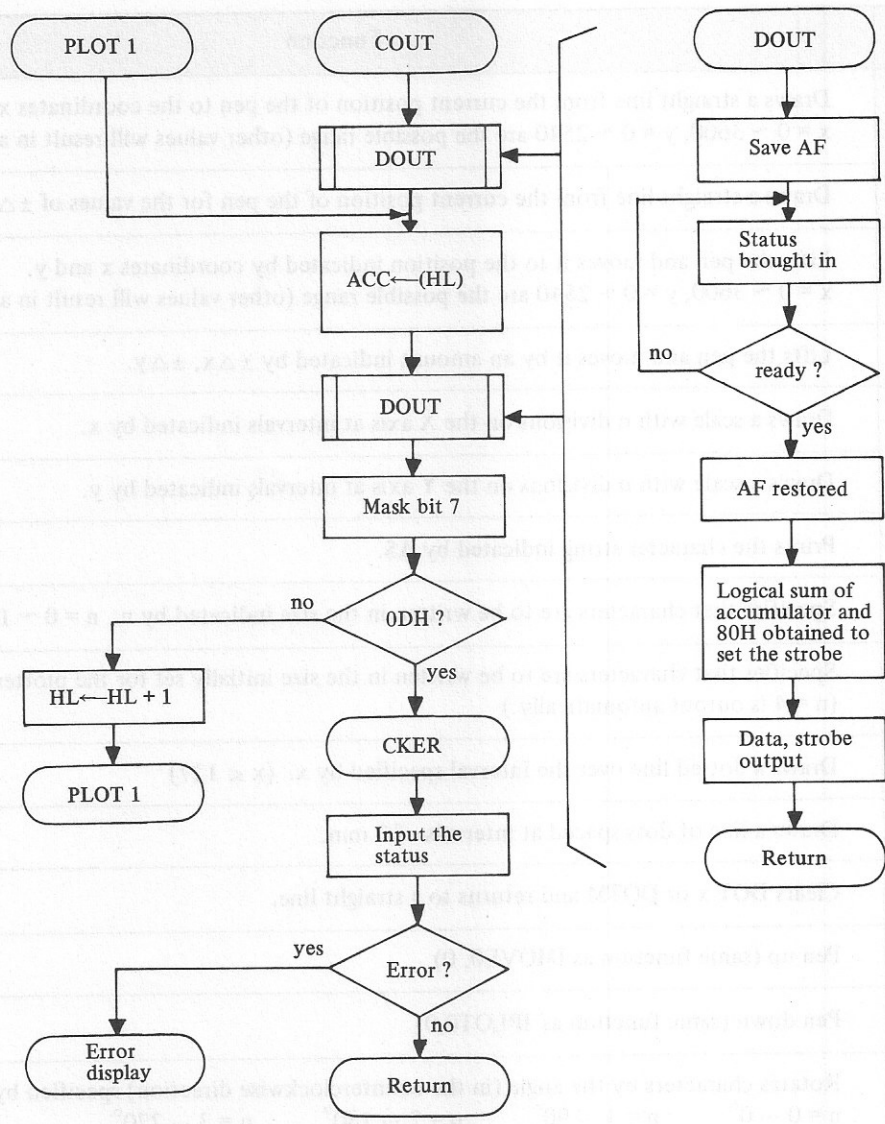
As is shown in the flowchart, the two parameters x and y are specified following the external declaration. The parameter type is real number and two bytes are set in the parameter buffer for both x and y.

The pre-conversion data address, the starting address of the buffer for storage of the data after conversion and the maximum value of x are loaded in registers HL, DE and BC, respectively, then BTOA is called and the data is checked and converted to ASCII code. Afterwards, the data delimiter for the plotter " , " is loaded and the process is repeated for y. The starting address of the data converted into ASCII code is loaded into the HL register, the 44H ("D") command (which draws a line on the plotter) is loaded and COUT is called. With COUT, the contents of the address indicated in the HL register are output to the plotter following the command.



a) ASCII conversion subroutine

Fig. 2 PLOT routine



b) Data output subroutine

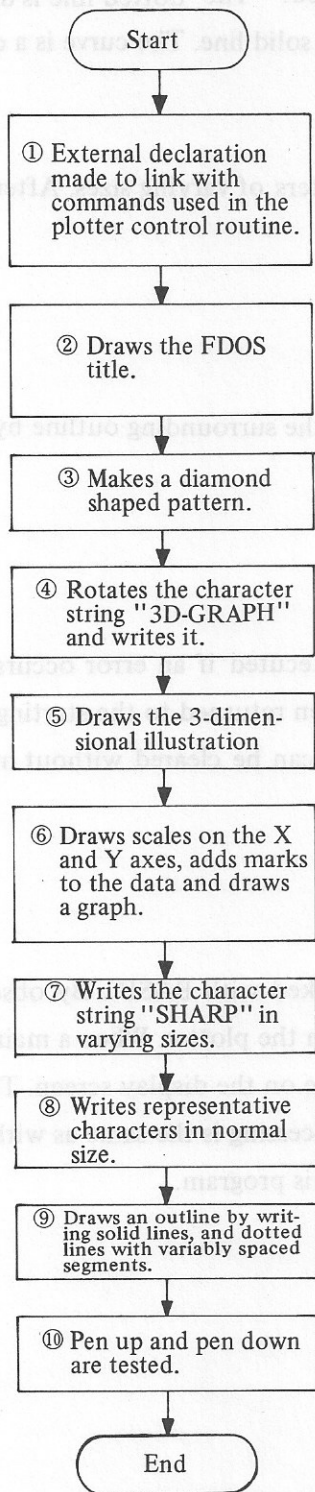
Fig. 2 PLOT routine

—Command Table—

Table 1. Plotter control commands

Command name	Function
PLOT x, y	Draws a straight line from the current position of the pen to the coordinates x, y; x = 0 ~ 3600, y = 0 ~ 2540 are the possible range (other values will result in an error).
IPLOT $\Delta x, \Delta y$	Draws a straight line from the current position of the pen for the values of $\pm \Delta x, \pm \Delta y$ only.
MOVE x, y	Lifts the pen and moves it to the position indicated by coordinates x and y. x = 0 ~ 3600, y = 0 ~ 2540 are the possible range (other values will result in an error).
IMOVE $\Delta x, \Delta y$	Lifts the pen and moves it by an amount indicated by $\pm \Delta x, \pm \Delta y$.
XAXIS x, n	Draws a scale with n divisions on the X axis at intervals indicated by x.
YAXIS x, n	Draws a scale with n divisions on the Y axis at intervals indicated by y.
CTYPE A\$	Prints the character string indicated by A\$.
SIZE n	Specifies that characters are to be written in the size indicated by n; n = 0 ~ 15
CSIZE	Specifies that characters are to be written in the size initially set for the plotter. (n = 4 is output automatically.)
DOT x	Draws a dotted line over the interval specified by x. (x \leq 127)
DOTM	Draws a line of dots spaced at intervals of 3 mm.
DOT0	Clears DOT x or DOTM and returns to a straight line.
PUP	Pen up (same function as IMOVE0, 0)
PDOWN	Pen down (same function as IPLOT0,0)
ANGL n	Rotates characters by the angle (in the counterclockwise direction) specified by n. n = 0 ~ 3 n = 0 ... 0° n = 1 ... 90° n = 2 ... 180° n = 3 ... 270°
MARK n	Writes the mark specified by n. n = 1 ~ 6 n = 1 ... · n = 2 ... ◊ n = 3 ... ◻ n = 4 ... ▲ n = 5 ... ✕ n = 6 ... ⊕
HOME	Moves the pen to the x, y coordinates it was at when the power was turned on, clears the error lamp and clears plotter error.
ERCLR	Clears plotter errors when they occur; the error lamp is not cleared, however.
(Note)	Values indicated by x and y are specified as integral multiples of 0.1 mm; for example, PLOT 100, 2000 results in a line being drawn to x = 10 mm and y = 200 mm.

-Outline of the BASIC Program-



As is indicated in the flowchart at left, the program consists of 10 subroutines.

At ①, the EXTERNAL statement is used for linking the program with the plotter control program commands. Although it is not necessary to use line numbers with statements other than GOTO and GOSUB, they are used in other locations to make the program easier to understand.

At ②, the title of FDOS is drawn. This routine uses the MOVE, IMOVE, PLOT and IPLOT commands.

At ③, a diamond pattern is drawn. MOVE X, Y is used to return the pen to the starting point, while IPLOT A, B is used to draw the pattern.

At ④, "3D-GRAPH" specified by A\$ is written at 90° angles with ANGL 1 and CTYPE A\$.

At ⑤, repetitions of attenuated SIN(X) and a 3-dimensional graph are drawn. The 3-dimensional graph is drawn in dots using the PUP and PDOWN commands.

Fig. 3 Flowchart of the BASIC program

At ⑥, scales are drawn on the X and Y axes with the XAXIS and YAXIS commands and marks are drawn on the graph with the MARK command; this processing is performed. The dotted line is drawn with the DOTM command, and the DOT0 command is used to return to a solid line. The curve is a drawing of $\text{SIN}(X)/X$. Refer to the command table for these commands.

At ⑦, the SIZE command is used to write the word SHARP in characters of varying sizes. Afterwards, the CSIZE command is used to return to standard size.

At ⑧, representative characters of standard size are drawn.

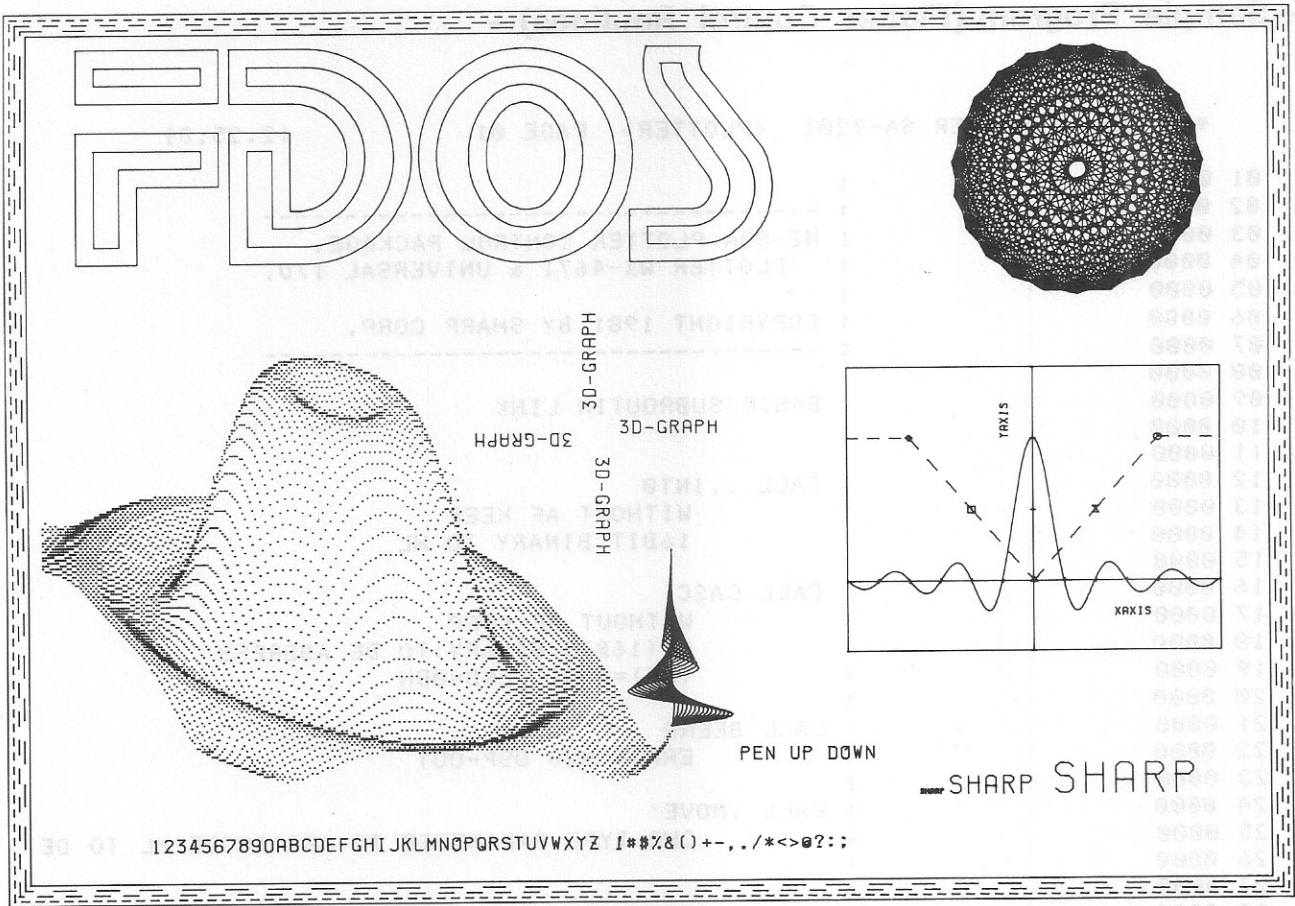
At ⑨, the HOME, DOT and DOT0 commands are used to draw the surrounding outline by changing between solid lines and dotted lines of varying pitch.

At ⑩, pen up and pen down are tested.

If the program is written so that the HOME command can be executed if an error occurs with the plotter, the error can be cleared, the error lamp extinguished and the pen returned to the starting position. If the ERCLR command is used (it is not in this program), the error can be cleared without moving the pen; however, in this case the error lamp is not extinguished.

—Conclusion—

As has been shown above, user-written programs can be easily linked with BASIC. By observing the conditions for linkage, it is possible to connect many devices other than the plotter. When a main program is created using the assembler, it may be necessary to output a picture on the display screen. The BASIC compiler and FDOS are very useful for this purpose. Further, the processing is the same as with machine language. The photograph on page 9 shows the result of execution of this program.



—References—

1. Watanabe Manufacturing Co., MILOT WX4671 Instruction Manual
2. Sharp Corp., Universal Interface Card Instruction Manual

—Sample Program (Plotter Control Routines)—

** Z80 ASSEMBLER SA-7201 <PLOTTER> PAGE 01

12.25.81

```

01 0000 ;
02 0000 ; -----
03 0000 ; MZ-80A PLOTTER CONTROL PACKAGE.
04 0000 ; PLOTTER WX-4671 & UNIVERSAL I/O.
05 0000 ;
06 0000 ; COPYRIGHT 1981 BY SHARP CORP.
07 0000 ; -----
08 0000 ;
09 0000 ; BASIC SUBROUTIN LINK
10 0000 ;
11 0000 ;
12 0000 ; CALL ..INT0
13 0000 ; WITHOUT AF KEEP
14 0000 ; 16BIT BINARY TO HL
15 0000 ;
16 0000 ; CALL CASC'
17 0000 ; WITHOUT AF KEEP
18 0000 ; HL(16BIT BINARY)TO DE ADDRESS
19 0000 ; (DE)=ASCII END=0DH
20 0000 ;
21 0000 ; CALL BEERR
22 0000 ; ERROR MSG DSP-OUT
23 0000 ;
24 0000 ; CALL .MOVE'
25 0000 ; CHR TYPE PARAMETER TO ASCII CODE(HL TO DE)
26 0000 ;
27 0000 ;
28 0000 ; PORT
29 0000 ;
30 0000 P PLTS: EQU 0EH ;PLOTTER STATUS READ
31 0000 P PLTD: EQU 0FH ;PLOTTER DATA OUT
32 0000 ;
33 0000 ; MOVE X,Y(PEN UP)
34 0000 ;
35 0000 MBUFF: DEFS 16
36 0010 ;
37 0010 MOVE: ENT
38 0010 02 DEFEB 2 ;NO OF PARAMETER
39 0011 00 DEFEB 0 ;REAL NO.
40 0012 MX: DEFS 2
41 0014 00 DEFEB 0 ;REAL NO.
42 0015 MY: DEFS 2
43 0017 ;
44 0017 2A1200 LD HL,(MX)
45 001A 110000 LD DE,MBUFF
46 001D D5 PUSH DE
47 001E 01110E LD BC,3601
48 0021 CD4E02 CALL BTOA
49 0024 CD7F02 CALL INS2C
50 0027 2A1500 LD HL,(MY)
51 002A 01ED09 LD BC,2541
52 002D CD4E02 CALL BTOA
53 0030 E1 POP HL
54 0031 3E4D LD A,4DH ;'M' MOVE CMD
55 0033 C34201 JP COUT
56 0036 ;
57 0036 ; INCREASE MOVE(PEN UP)DX,DY
58 0036 ;
59 0036 IMBUFF: DEFS 16
60 0046 ;

```

```

01 0046          IMOVE:  ENT
02 0046 02      DEFEB  2          ;NO OF PARAMETER
03 0047 00      DEFEB  0          ;REAL
04 0048          IMX:    DEFS   2
05 004A 00      DEFEB  0          ;REAL
06 004B          IMY:    DEFS   2
07 004D          ;
08 004D 2A4800  LD      HL,(IMX)
09 0050 113600  LD      DE,IMBUFF
10 0053 D5      PUSH   DE
11 0054 01110E  LD      BC,3601
12 0057 CD5E02  CALL   NSBTOA
13 005A CD7F02  CALL   INS2C
14 005D 2A4B00  LD      HL,(IMY)
15 0060 01ED09  LD      BC,2541
16 0063 CD5E02  CALL   NSBTOA
17 0066 E1      POP    HL
18 0067 3E52    LD      A,'R'          ;RELATIVE MOVE CMD
19 0069 C34201  JP     COUT
20 006C          ;
21 006C          ; X-AXIS
22 006C          ;
23 006C          QRXBUF: DEFS  16
24 007C          ;
25 007C          XAXIS:  ENT
26 007C 02      DEFEB  2          ;NO OF PARAMETER
27 007D 00      DEFEB  0          ;REAL
28 007E          QX:    DEFS   2
29 0080 00      DEFEB  0          ;REAL
30 0081          RX:    DEFS   2
31 0083          ;
32 0083 2A7E00  LD      HL,(QX)
33 0086 116C00  LD      DE,QRXBUF
34 0089 D5      PUSH   DE
35 008A 01110E  LD      BC,3601
36 008D CD4E02  CALL   BTOA
37 0090 CD7F02  CALL   INS2C
38 0093 2A8100  LD      HL,(RX)
39 0096 01110E  LD      BC,3601
40 0099 CD4E02  CALL   BTOA
41 009C 3E31    LD      A,'1'
42 009E 1832    JR     AXOUT
43 00A0          ;
44 00A0          ; Y-AXIS
45 00A0          ;
46 00A0          QRYBUF: DEFS  16
47 00B0          ;
48 00B0          YAXIS:  ENT
49 00B0 02      DEFEB  2          ;NO OF PARAMETER
50 00B1 00      DEFEB  0          ;REAL
51 00B2          QY:    DEFS   2
52 00B4 00      DEFEB  0          ;REAL
53 00B5          RY:    DEFS   2
54 00B7          ;
55 00B7 2AB200  LD      HL,(QY)
56 00BA 11A000  LD      DE,QRYBUF
57 00BD D5      PUSH   DE
58 00BE 01ED09  LD      BC,2541
59 00C1 CD4E02  CALL   BTOA
60 00C4 CD7F02  CALL   INS2C

```

```

01 00C7 2AB500          LD      HL,(RY)
02 00CA 01ED09          LD      BC,2541
03 00CD CD4E02          CALL   BTOA
04 00D0 3E30            LD      A,'0'
05 00D2 CD2302          AXOUT:  CALL  AXIS
06 00D5 E1              POP     HL
07 00D6 186D            JR      PLOT1
08 00D8                ;
09 00D8                ; INCREASE PLOT X+DX,Y+DY
10 00D8                ;
11 00D8                IPBUFF: DEFS   16
12 00E8                ;
13 00E8                IPLOT:  ENT
14 00E8 02              DEFB   2          ;NO OF PARAMETER
15 00E9 00              DEFB   0          ;REAL
16 00EA                IPX:    DEFS   2
17 00EC 00              DEFB   0          ;REAL
18 00ED                IPY:    DEFS   2
19 00EF                ;
20 00EF 2AEA00          LD      HL,(IPX)
21 00F2 11D800          LD      DE,IPBUFF
22 00F5 D5              PUSH   DE
23 00F6 01110E          LD      BC,3601
24 00F9 CD5E02          CALL   NSBTOA
25 00FC CD7F02          CALL   INS2C
26 00FF 2AED00          LD      HL,(IPY)
27 0102 01ED09          LD      BC,2541
28 0105 CD5E02          CALL   NSBTOA
29 0108 E1              POP     HL
30 0109 3E49            LD      A,'I'
31 010B 1835            JR      COUT          ;INCREASE CMD
32 010D                ;
33 010D                ; PLOT X,Y
34 010D                ;
35 010D                PTBUFF: DEFS  16
36 011D                ;
37 011D                ;
38 011D                PLOT:   ENT
39 011D 02              DEFB   2          ;NO OF PARAMETER
40 011E 00              DEFB   0          ;REAL
41 011F                PX:    DEFS   2
42 0121 00              DEFB   0          ;REAL
43 0122                PY:    DEFS   2
44 0124                ;
45 0124 2A1F01          LD      HL,(PX)
46 0127 110D01          LD      DE,PTBUFF
47 012A D5              PUSH   DE
48 012B 01110E          LD      BC,3601
49 012E CD4E02          CALL   BTOA
50 0131 CD7F02          CALL   INS2C
51 0134 2A2201          LD      HL,(PY)
52 0137 01ED09          LD      BC,2541
53 013A CD4E02          CALL   BTOA
54 013D E1              POP     HL
55 013E 3E44            LD      A,'D'
56 0140 1800            JR      COUT          ;DRAW CMD
57 0142                ;
58 0142                ; ASCII CODE OUT SUB
59 0142                ;
60 0142 CD2F02          COUT:  CALL   DOUT

```

```

01 0145 7E          PLOT1: LD      A,(HL)
02 0146 CD2F02     CALL    DOUT
03 0149 E67F      AND     7FH
04 014B FE0D      CP      0DH
05 014D 2803      JR      Z,CKER
06 014F 23        INC     HL
07 0150 18F3      JR      PLOT1
08 0152           ;
09 0152           ; ERROR CHECK
10 0152           ;
11 0152 CDCA02     CKER:  CALL    DLY
12 0155 DB0E      IN      A,(PLTS)
13 0157 E602      AND     02H          ;CKER(BIT 1=0?)
14 0159 C0        RET     NZ
15 015A CD0000     E      CKER1: CALL    BEERR
16 015D 50        DEFB   80
17 015E 504C4F54   DEFM   'PLOTTER ERROR'
18 0162 54455220
19 0166 4552524F
20 016A 52
21 016B 0D        DEFB   0DH
22 016C           ;
23 016C           ; PRINT ASCII
24 016C           ;
25 016C           CTYPE: ENT
26 016C 01        DEFB   1          ;NO OF PARAMETER
27 016D 80        DEFB   80H       ;CHR
28 016E           TP:  DEFS   2
29 0170           ;
30 0170 2A6E01     LD      HL,(TP)
31 0173 CD0000     E      CALL    .MOVE'
32 0176 EB        EX     DE,HL
33 0177 3E50      LD     A,'P'
34 0179 18C7      JR     COUT          ;PRINT CMD
35 017B           ;
36 017B           ; SIZE ALPHA
37 017B           ;
38 017B           SBUF: DEFS   5
39 0180           ;
40 0180           SIZE:  ENT
41 0180 01        DEFB   1          ;NO OF PARAMETER
42 0181 00        DEFB   0          ;REAL
43 0182           SE:  DEFS   2
44 0184           ;
45 0184 2A8201     LD     HL,(SE)
46 0187 117B01     LD     DE,SBUF
47 018A 011000     LD     BC,16
48 018D CD4E02     CALL   BTOA
49 0190 EB        EX     DE,HL
50 0191 3E53      LD     A,'S'
51 0193 18AD      JR     COUT          ;SCALE CMD
52 0195           ;
53 0195           ; DOT LINE
54 0195           ;
55 0195           DBUF: DEFS   5
56 019A           ;
57 019A           DOT:  ENT
58 019A 01        DEFB   1          ;NO OF PARAMETER
59 019B 00        DEFB   0          ;REAL
60 019C           DT:  DEFS   2

```

```

01 019E          ;
02 019E 2A9C01  LD      HL,(DT)
03 01A1 119501  LD      DE,DBUF
04 01A4 018000  LD      BC,128
05 01A7 CD4E02  CALL   BTOA
06 01AA EB      EX      DE,HL
07 01AB 3E42    DOTWR: LD      A,'B'          ;LINE GAGE CMD
08 01AD CD4201  CALL   COUT
09 01B0 219E02  LD      HL,DOTMSG
10 01B3 188D    JR      COUT
11 01B5          ;
12 01B5          DOTM:  ENT
13 01B5 00      DEFB   0          ;NO PARAMETER
14 01B6 219B02  LD      HL,DOIMSG
15 01B9 18F0    JR      DOTWR
16 01BB          DOT0:  ENT
17 01BB 00      DEFB   0          ;NO PARAMETER
18 01BC 219802  LD      HL,LTMSG
19 01BF 180A    JR      CNCT
20 01C1          CSIZE: ENT
21 01C1 00      DEFB   0          ;NO PARAMETER
22 01C2 21A102  LD      HL,ISMSG
23 01C5 1804    JR      CNCT
24 01C7          ;
25 01C7          ; PEN UP
26 01C7          ;
27 01C7          PUP:   ENT
28 01C7 00      DEFB   0          ;NO PARAMETER
29 01C8 21A402  LD      HL,PUMSG
30 01CB C34501  CNCT:  JP     PLOT1
31 01CE          ;
32 01CE          ; PEN DOWN
33 01CE          ;
34 01CE          PDOWN: ENT
35 01CE 00      DEFB   0          ;NO PARAMETER
36 01CF 21A902  LD      HL,PDMSG
37 01D2 18F7    JR      CNCT
38 01D4          ;
39 01D4          ; ERROR CLEAR
40 01D4          ;
41 01D4          ERCLR: ENT
42 01D4 00      DEFB   0          ;NO PARAMETER
43 01D5 1806    JR      LFOUT
44 01D7          ;
45 01D7          ; HOME
46 01D7          ;
47 01D7          HOME:  ENT
48 01D7 00      DEFB   0          ;NO PARAMETER
49 01D8 3E48    LD      A,'H'          ;HOME
50 01DA CD2F02  CALL   DOUT
51 01DD 3E0A    LFOUT: LD      A,0AH        ;ER CLEAR CMD
52 01DF 184E    JR      DOUT
53 01E1          ;
54 01E1          ; ROTATE ALPHA
55 01E1          ;
56 01E1          PBUF:  DEFS   5
57 01E6          ;
58 01E6          ANGL:  ENT
59 01E6 01      DEFB   1          ;NO OF PARAMETER
60 01E7 00      DEFB   0          ;REAL

```



```

01 01E8          AL:   DEFS   02
02 01EA          ;
03 01EA 2AE801          LD   HL,(AL)
04 01ED 11E101          LD   DE,PBUF
05 01F0 010400          LD   BC,4
06 01F3 CD4E02          CALL BTOA
07 01F6 EB             EX   DE,HL
08 01F7 3E51          LD   A,'Q'
09 01F9 C34201          JP   COUT ; ROTATE CMD
10 01FC          ;
11 01FC          ; MARK
12 01FC          ;
13 01FC          MBUF:  DEFS   5
14 0201          ;
15 0201          MARK:  ENT   ;
16 0201 01          DEFB  1 ; NO OF PARAMETER
17 0202 00          DEFB  0 ; REAL
18 0203          MK:   DEFS   12
19 0205          ;
20 0205 2A0302          LD   HL,(MK)
21 0208 CD0000          CALL ..INT0
22 020B DA0000          JP   C,ER3
23 020E CDAE02          CALL CKNEG
24 0211 010700          LD   BC,7
25 0214 CD8902          CALL CKCOD0
26 0217 11FC01          LD   DE,MBUF
27 021A CD0000          CALL CASC'
28 021D EB             EX   DE,HL
29 021E 3E4E          LD   A,'N'
30 0220 C34201          JP   COUT ; MARK CND
31 0223          ;
32 0223          ; AXIS SUB
33 0223          ;
34 0223 F5           AXIS:  PUSH  AF
35 0224 3E58          LD   A,'X'
36 0226 CD2F02          CALL DOUT ; AXIS CMD
37 0229 F1           POP   AF
38 022A CD2F02          CALL DOUT
39 022D 3E2C          AXIS1: LD   A,2CH ; ',' DELIMITER
40 022F          ;
41 022F          ;
42 022F          ;;;;;;;;;;
43 022F          ;
44 022F          ; PLOTTER DATA CONTROL
45 022F          ;   A=DATA
46 022F          ;
47 022F          ;;;;;;;;;;
48 022F          ;
49 022F F5           DOUT:   PUSH  AF
50 0230 CD0000          DOUTP: CALL AT BREAK
51 0233 DA0000          JP   C,ER37
52 0236 DB0E          IN   A,(PLTS)
53 0238 CB47          BIT  0,A
54 023A 20F4          JR   NZ,DOUTP
55 023C F1           POP   AF
56 023D D30E          OUT  (PLTD),A
57 023F CDCA02          CALL DLY
58 0242 F680          OR   80H
59 0244 D30E          OUT  (PLTD),A
60 0246 CDCA02          CALL DLY

```

```

01 0249 E67F          AND      7FH
02 024B D30E          OUT      (PLTD),A
03 024D C9           RET
04 024E              ;
05 024E              ; BINARY TO ASCII
06 024E              ;
07 024E CD0000      E   BTOA:  CALL  ..INT0
08 0251 DA0000      E           JP      C,ER3
09 0254 CDAE02          CALL  CKNEG
10 0257 CD8E02          CALL  CKCOD
11 025A CD0000      E           CALL  CASC'
12 025D C9           RET
13 025E              ;
14 025E              ;
15 025E CD0000      E   NSBTOA: CALL ..INT0
16 0261 DA0000      E           JP      C,ER3
17 0264 CD6E02          CALL  INS2D
18 0267 CD8E02          CALL  CKCOD
19 026A CD0000      E           CALL  CASC'
20 026D C9           RET
21 026E              ;
22 026E              ; IF NEG... INSERT '-'
23 026E              ;
24 026E 7C           INS2D: LD      A,H
25 026F CB7F          BIT      7,A
26 0271 C8           RET      Z
27 0272 3E2D          LD      A,'-'
28 0274 12           LD      (DE),A
29 0275 13           INC     DE
30 0276 7D           LD      A,L
31 0277 2F           CPL
32 0278 6F           LD      L,A
33 0279 7C           LD      A,H
34 027A 2F           CPL
35 027B 67           LD      H,A
36 027C 23           INC     HL
37 027D C9           RET
38 027E              ;
39 027E              ; SEEK 0D,INSERT 2C
40 027E              ;
41 027E 13           INC     DE
42 027F 1A           INS2C: LD      A,(DE)
43 0280 FE0D          CP      0DH
44 0282 20FA          JR      NZ,INS2C-1
45 0284 3E2C          LD      A,2CH
46 0286 12           LD      (DE),A
47 0287 13           INC     DE
48 0288 C9           RET
49 0289              ;
50 0289              ; CHECK DATA VALUE
51 0289              ;
52 0289 7C           CKCOD0: LD      A,H
53 028A B5           OR      L
54 028B CA0000      E           JP      Z,ER3
55 028E E5           CKCOD:  PUSH  HL
56 028F B7           OR      A
57 0290 ED42          CKMAX:  SBC   HL,BC
58 0292 F20000      E           JP      P,ER3
59 0295 E1           POP   HL
60 0296 B7           OR      A

```

```

01 0297 C9          RET
02 0298          ;
03 0298          ; LINE MSG
04 0298          ;
05 0298 4C30      LTMSG:  DEFM  'L0'
06 029A 0D          DEFB  0DH
07 029B          ;
08 029B          ; BETWEEN DOT
09 029B          ;
10 029B 3330      DOIMSG: DEFM  '30'
11 029D 0D          DEFB  0DH
12 029E          ;
13 029E          ; DOT MSG
14 029E          ;
15 029E 4C31      DOTMSG: DEFM  'L1'
16 02A0 0D          DEFB  0DH
17 02A1          ;
18 02A1          ; INITIAL SIZE SET
19 02A1          ;
20 02A1 5334      ISMSG:  DEFM  'S4'
21 02A3 0D          DEFB  0DH
22 02A4          ;
23 02A4          ; PEN UP MSG
24 02A4          ;
25 02A4 52302C30  PUMSG:  DEFM  'R0,0'      ; RELATIVE CMD
26 02A8 0D          DEFB  0DH
27 02A9          ;
28 02A9          ; PEN DOWN MSG
29 02A9          ;
30 02A9 49302C30  PDMSG:  DEFM  '10,0'      ; INCREASE CMD
31 02AD 0D          DEFB  0DH
32 02AE          ;
33 02AE          ; CHECK DATA SIGN
34 02AE          ;
35 02AE 7C          CKNEG:  LD      A,H
36 02AF CB7F          BIT      7,A
37 02B1 C8          RET      Z
38 02B2 CD0000      E      CKER2: CALL  BEERR
39 02B5 51          DEFB  81
40 02B6 554E464F          DEFM  'UNFORMAT DATA ERROR'
41 02BA 524D4154
42 02BE 20444154
43 02C2 41204552
44 02C6 524F52
45 02C9 0D          DEFB  0DH
46 02CA          ;
47 02CA C5          DLY:   PUSH  BC
48 02CB F5          PUSH  AF
49 02CC 060A          LD      B,10
50 02CE AF          DLY2:  XOR    A
51 02CF 3D          DEC    A
52 02D0 20FD          JR     NZ,-1
53 02D2 10FA          DJNZ  DLY2
54 02D4 F1          POP   AF
55 02D5 C1          POP   BC
56 02D6 C9          RET
57 02D7          END

```

AL	01E8	ANGL	01E6	AXIS	0223	AXIS1	022D	AXOUT	00D2
BTOA	024E	CKCOD	028E	CKCOD0	0289	CKER	0152	CKER1	015A
CKER2	02B2	CKMAX	0290	CKNEG	02AE	CNCT	01CB	COUT	0142
Csize	01C1	CTYPE	016C	DBUF	0195	DLY	02CA	DLY2	02CE
DOIMSG	029B	DOT	019A	DOT0	01BB	DOTM	01B5	DOTMSG	029E
DOTWR	01AB	DOUT	022F	DOUTP	0230	DT	019C	ERCLR	01D4
HOME	01D7	IMBUFF	0036	IMOVE	0046	IMX	0048	IMY	004B
INS2C	027F	INS2D	026E	IPBUFF	00D8	IPLOT	00E8	IPX	00EA
IPY	00ED	ISMSG	02A1	LFOUT	01DD	LTMMSG	0298	MARK	0201
MBUF	01FC	MBUFF	0000	MK	0203	MOVE	0010	MX	0012
MY	0015	NSBTOA	025E	PBUF	01E1	PDMSG	02A9	PDOWN	01CE
PLOT	011D	PLOT1	0145	PLTD	000F	PLTS	000E	PTBUFF	010D
PUMSG	02A4	PUP	01C7	PX	011F	PY	0122	QRXBUF	006C
QRYBUF	00A0	QX	007E	QY	00B2	RX	0081	RY	00B5
SBUF	017B	SE	0182	SIZE	0180	TP	016E	XAXIS	007C
YAXIS	00B0								

```

1 17 02A1
1 18 02A1
1 19 02A1
1 20 02A1 0094
1 21 02A3 0D
1 22 02A4
1 23 02A4
1 24 02A4
1 25 02A4 2302C0B
1 26 02A8 0D
1 27 02A9
1 28 02A9
1 29 02A9
1 30 02A9 4980C3B
1 31 02AD 0D
1 32 02AE
1 33 02AE
1 34 02AE
1 35 02AE 7C
1 36 02AF C87F
1 37 02B1 C8
1 38 02B3 C0800
1 39 02B5 21
1 40 02B8 02AE447F
1 41 02BA 234D124
1 42 02BE 2044124
1 43 02C2 41284223
1 44 02C6 25A723
1 45 02C9 0D
1 46 02CA
1 47 02CA C8
1 48 02CB F8
1 49 02CC 0A0A
1 50 02CE AF
1 51 02CF 3B
1 52 02D0 507D
1 53 02D3 107A
1 54 02D4 F1
1 55 02D5 C1
1 56 02D8 C9
1 57 02D7
    
```

-BASIC Main Program-

BASIC compiler SA-7701 <X-YDEMO> page 1

12.25.81

```
000F EXTERNAL I PLOT,PLOT,IMOVE,MOVE,PUP,PDOWN,XAXIS,YAXIS,CTYPE
000F EXTERNAL CSIZE,SIZE,HOME,ANGL,MARK,ERCLR,DOTM,DOT0,DOT
000F DIM D(1,255),E(50)
0042 DIM A(23),B(23)
005F HOME : CSIZE : ANGL 0: DOT0
00A8 REM *FDOS NO SAKUGA*
00A8 REM * F *
00A8 MOVE 200,2400
00DD I PLOT 400,0: I PLOT 0,-150: I PLOT -400,0: I PLOT 0,150
018C IMOVE 50,-50: I PLOT 300,0: I PLOT 0,-50: I PLOT -300,0: I PLOT 0,50
0256 MOVE 200,2170: I PLOT 400,0: I PLOT 0,-150: I PLOT -230,0
0300 I PLOT 0,-220: I PLOT -170,0: I PLOT 0,370
038A IMOVE 50,-50: I PLOT 300,0: I PLOT 0,-50
03FF I PLOT -230,0: I PLOT 0,-220: I PLOT -70,0
047B I PLOT 0,270
04A9 REM * D *
04A9 MOVE 630,2400: I PLOT 210,0
0505 A=840: B=300: C=2100: GOSUB 1100
0540 I PLOT -210,0: I PLOT 0,370: I PLOT 170,0: I PLOT 0,-220
05DC A=810: B=150: C=2100: GOSUB 1000
060B I PLOT -180,0: I PLOT 0,150: IMOVE 50,-50: I PLOT 160,0
06B5 A=840: B=250: C=2100: GOSUB 1100
06E4 I PLOT -160,0: I PLOT 0,270: I PLOT 70,0: I PLOT 0,-220: I PLOT 60,0
07AE A=810: B=200: C=2100: GOSUB 1000
07D6 I PLOT -130,0: I PLOT 0,50: GOTO 1200
0836 1000 FOR I=-π/2 TO π/2 STEP π/60
0890 X=INT(A+B*COS(I)): Y=INT(C+B*SIN(I))
090A PLOT X,Y
091F NEXT I
092A RETURN
0930 1100 FOR I=π/2 TO -π/2 STEP -π/60
0978 X=INT(A+B*COS(I)): Y=INT(C+B*SIN(I))
09DE PLOT X,Y
09F3 NEXT I
09F9 RETURN
09FF 1200 REM * 0 *
0A05 X=0: Y=0: A=0: B=0: C=0: D=0: MOVE 1760,2100
0A69 A=1460: B=300: C=2100: D=300: GOSUB 1300
0AA1 IMOVE -50,0
0AC8 A=1460: B=250: C=2100: D=250: GOSUB 1300
0AF9 IMOVE -90,0
0B27 A=1460: B=160: C=2100: D=200: GOSUB 1300
0B58 IMOVE -50,0
0B7F A=1460: B=110: C=2100: D=150: GOSUB 1300
0BB7 MOVE 1840,2350: GOTO 1400
0BF2 1300 FOR I=0*π TO 2*π STEP π/60
0C28 X=INT(A+B*COS(I)): Y=INT(C+D*SIN(I))
0C8E PLOT X,Y
0CA3 NEXT I: A=0: B=0: C=0: D=0: X=0: Y=0: RETURN
0CE5 1400 REM * S *
0CEB FOR I=π TO π*7/6 STEP π/60
0D32 X=INT(1960+120*COS(I)): Y=INT(2350+120*SIN(I))
0DA6 PLOT X,Y: NEXT I: X=0: Y=0
0DD3 A=1920: B=110: C=2010: GOSUB 1500
0E09 I PLOT -120,0: I PLOT 0,-50: I PLOT 140,0
0E85 A=1940: B=160: C=2010: GOSUB 1600
0EB4 FOR I=π*7/6 TO π*186/180 STEP -π/60
0F0C X=INT(2030+135*COS(I)): Y=INT(2370+135*SIN(I))
0F87 PLOT X,Y
0F9C NEXT I: I PLOT -60,0: X=0: Y=0: IMOVE -38,45
```

```

1010      I PLOT 150,0
1037      FOR I=π*8/9 TO π*7/6 STEP π/60
108D      X=INT(2030+85*COS(I)): Y=INT(2370+85*SIN(I))
10FA      PLOT X,Y
110F      NEXT I: X=0: Y=0
1127      A=1940: B=210: C=2010: GOSUB 1500
114F      I PLOT -190,0: I PLOT 0,150: I PLOT 170,0
11CB      A=1920: B=60: C=2010: GOSUB 1600
11F3      FOR I=π*7/6 TO π*8/9 STEP -π/60
1244      X=INT(1960+170*COS(I)): Y=INT(2350+170*SIN(I))
12AA      PLOT X,Y
12BF      NEXT I: X=0: Y=0: GOTO 1700
12DD 1500  REM * SUB A *
12E3      FOR I=π/6 TO -π/2 STEP -π/60
1325      X=INT(A+B*COS(I)): Y=INT(C+B*SIN(I))
138B      PLOT X,Y
13A0      NEXT I: X=0: Y=0: RETURN
13BE 1600  REM * SUB B *
13C4      FOR I=-π/2 TO π/6 STEP π/60
13FD      X=INT(A+B*COS(I)): Y=INT(C+B*SIN(I))
1463      PLOT X,Y
1478      NEXT I: X=0: Y=0: RETURN
1496      REM * DIAMOND *
1496      MOVE 3000,2050
14CB 1700  FOR Z=1 TO 23
14F1      A(Z)=INT(350*COS(2*π*(Z-1)/23+π/2)+3000)
158D      B(Z)=INT(-350*SIN(2*π*(Z-1)/23+π/2)+2050)
1621      NEXT Z
1627      FOR S=2 TO 11
1649      L=2-S
1655      X=A(23): Y=B(23)
1682      MOVE X,Y
1697      FOR I=1 TO 24
16B9      J=L+S
16C5      IF J<24 THEN 1800
16E7      J=J-23
16F3 1800  X1=A(J): L=J: Y1=B(J)
172A      A=X1-X: B=Y1-Y
1742      I PLOT A,B
1757      X=X1: Y=Y1
1769      NEXT I: NEXT S: X=0: Y=0: X1=0: Y1=0: A=0: B=0
17AB      MOVE 1650,1320
17E0      REM *PRINT " 3D-GRAPH"*
17E0      A$=" 3D-GRAPH"
1802      FOR I=3 TO 0 STEP -1
1830      MOVE 1650,1320
1857      ANGL I
1866      CTYPE A$
187D      NEXT I: A$=""
1891      REM * 3D-GRAPHIC *
1891      T=0: S=0
18A3      FOR L=0 TO 255
18BE      D(0,L)=-1: D(1,L)=-1: NEXT L
18F2      MOVE 100,250
1920      FOR Y=-180 TO 180 STEP 4
194E      FOR X=-180 TO 180 STEP 4
1975      IF (Y=-180)*(X=180) THEN 2600
19B3 1900  R=π/180*SQR(X*X+Y*Y)
19FE      Z=100*COS(R)-30*COS(3*R)
1A47      A=INT(110+X/2+(16-Y/2)/2)

```

```

1AA8      B=INT((116-Y/2-Z)/2): B=192-B
1B04      IF (A<0)+(A>255) THEN 2000
1B39      IF D(0,A)=-1 THEN 2100
1B68      IF B<=D(0,A) THEN 2300
1B93      IF B>=D(1,A) THEN 2400
1BBE      2000 NEXT X
1BCA      IMOVE 0,0: T=0: S=0
1C03      NEXT Y
1C09      GOTO 2700
1C0F      2100 IF A=0 THEN 2200
1C2D      IF D(0,A-1)=-1 THEN 2200
1C68      IF D(0,A+1)=-1 THEN 2200
1CA3      D(0,A)=INT((D(0,A-1)+D(0,A+1))/2)
1D0F      D(1,A)=INT((D(1,A-1)+D(1,A+1))/2)
1D7B      GOSUB 2500: GOTO 2000
1D8E      2200 D(0,A)=B: D(1,A)=B: GOSUB 2500: GOTO 2000
1DD5      2300 GOSUB 2500: D(0,A)=B: IF D(1,A)=-1 THEN D(1,A)=B
1E42      GOTO 2000
1E48      2400 GOSUB 2500: D(1,A)=B: IF D(0,A)=-1 THEN D(1,A)=B
1EB5      GOTO 2000
1EBB      2500 REM *SUB DATA OUT *
1EC1      DX=100+7*A: DY=8*B
1EE5      MOVE DX,DY: PDOWN : PUP
1F0C      RETURN
1F12      2600 REM *SUB GENSUI *
1F18      FOR K=1 TO 50: E(K)=0: NEXT K
1F4D      FOR I=1 TO 50
1F68      E(I)=SIN(I*/12.5)
1FA4      NEXT I
1FAA      N=15
1FBA      FOR I=1 TO N
1FD5      D=5*I/N+5: G=(N-I+0.5)/N*4: O=DX: P=DY
2049      MOVE O,P
205E      FOR J=1 TO 50
2079      O=DX+E(J)*G*(50-J): P=P+D
20C0      PLOT O,P
20D5      NEXT J: NEXT I: MOVE DX,DY: GOTO 1900
20FC      2700 REM * SIN(X)/X *
2102      MOVE 2350,900
2130      XAXIS 87,12
2165      MOVE 2870,700
219A      YAXIS 200,4: MOVE 2350,900
21E8      FOR I=-6 TO 6 STEP 0.08
2216      IF I=0 THEN Y=1: X=0: GOTO 2800
2243      X=I*
224F      Y=SIN(X)/X
2264      2800 A=2870+INT(27.5*X+0.5)
229E      B=900+INT(400*Y)
22BF      PLOT A,B
22D4      NEXT I
22DA      MOVE 2350,1300: DOTM : MARK 1
2329      I PLOT 174,0: MARK 2: I PLOT 174,-200: MARK 3: I PLOT 174,-200: MARK 4
23ED      I PLOT 174,200: MARK 5: I PLOT 174,200: MARK 6: I PLOT 174,0
2492      DOT0
249B      MOVE 2350,700: I PLOT 0,800: I PLOT 1044,0: I PLOT 0,-800: I PLOT -1044,0
256C      MOVE 3100,800: SIZE 2: A$="XAXIS": CTYPE A$
25D7      A$="YAXIS": MOVE 2800,1300: ANGL 1: CTYPE A$: ANGL 0: A$=""
2668      REM * SHARP SIZE *
2668      MOVE 2550,300
2696      B$="SHARP "

```

```

26AA FOR I=1 TO 15 STEP 5
26D1 SIZE I
26E0 CTYPE B$
26F2 NEXT I: B$=""
2706 CSIZE
270F REM * CHARACTER SET *
270F MOVE 400,150
2736 C$="1234567890ABCDEFGHIJKLMNPOQRSTUVWXYZ !#$%&()+-.,/*<>??:;"
277C CTYPE C$
278E REM * DOT0 TYPE *
278E HOME : PLOT 0,2500: PLOT 3590,2500: PLOT 3590,0: PLOT 0,0: IMOVE 15,15
2868 H=3560: V=2470: R=15
2891 FOR I=20 TO 60 STEP 20
28BF DOT I
28CE IPLOT 0,V: IPLOT H,0: IPLOT 0,-V: IPLOT -H,0: IMOVE R,R
296D V=V-2*R: H=H-2*R
299D NEXT I
29A3 HOME : IMOVE 60,60
29D3 DOT0 : IPLOT 0,V: IPLOT H,0: IPLOT 0,-V: IPLOT -H,0
2A66 D$="PEN UP DOWN"
2A7F MOVE 2050,400
2AA6 CTYPE D$
2AB8 IMOVE 50,0
2ADF FOR I=1 TO 5: PUP : PDOWN : NEXT I
2B12 CSIZE : DOT0 : ANGL 0: HOME
2B45 END

```

** Compiler found no errors.