Marketed by

INTERTEK, INC.

Naito Bldg. 7-2-8 Nishi-Shinjuku Shinjuku-ku TOKYO JAPAN

Telephone : TOKYO 03 (363) 6649

Telex : J23497

Manufactured by

TOYO TELESONICS CO., LTD.

Distributed by

PRINTED IN JAPAN 30510

pecker-I

EPROM PROGRAMMER PKW-5000

USER'S MANUAL

INTERTEK, INC.

INTRODUCTION

■ Introduction

The PECKER–I is a portable PROM programmer, and yet capable of full fledged functions.
Among its many features are the wide range of the PROMs it can program, having 16K byte RAM as a standard component, and secure operatability by using independent command keys.
Users are requested to read this manual prior to using the programmer to take full advantage of these features.

■ Outline

PECKER–I is a PROM programmer designed for programming Intel's EPROMs, 2704, 2708, 2758, 2716 and 2732, T.I's 2708, 2516 and 2532, and their equivalents.
The desired PROM to be programmed can be selected by a combination of settings of the two slide switches, while the key operations and display format are specified in an uniform way so that the users can easily learn how to use the PECKER–I. It uses the Z–80 μ-processor for the control, and users can execute or debug their programs on the machine language level of the Z–80, the 8080 and the like processors.
In order to give an overly annoying details of operations using many functions, this user's mannual is edited in two parts.
[1] Description of commands needed in programming.
[2] High level use of the PECKER–I
Useis are requested to read through the mannual in accordance with their needs and capabilities.

[Basic Description of the PECKER–I]

■ 1–1 Characteristics of the PECKER–I

Type:	PKW–5000
CPU used:	Z-80
RAM capacity:	16K byte
Programmable PROMs:	Intel's 2704, 2708, 2758, 2716, 2732, T.I.'s 2708, 2516, 2532, and their equivalents.
Display:	16 digit, 7 segment LEDs
ROM selection:	Combinaiton of the settings of two slide switches.
Key board:	15 command keys 20 data keys
Operational Temperature:	4 ~ 40°C
Operational Humidity:	30 ~ 80% (above dew point)
Errosive Gas:	SO_2, under 0.05ppm
Errosive Gas:	No harmful gas is emitted.
Dimension:	282W x 187D x 48 H m/m (11.1 x 7.4 x 1.9 in)
Weight:	1.8 Kg (4 lbs)
Line Voltage:	AC 100V, AC 117V or AC 220V ±10%, 50/60 Hz
Power Consumption:	30VA

■ 1–2 Programming Specifications

(1) 2708 type 8K EPROMs

Programming Voltage:	26V ±0.3V
Programming Current:	40mA min.
Programming Pulse Width:	670 ± 30μs
Pulse Rise Time:	Less than 1μs
Pulse Fall Time:	Less than 1μs
Programming Time per Address:	100 ~ 200ms

(After the programming is completed for all the addresses to be programmed, additional programming with twice the programming time should be performed in order to increase programming reliability.)

(2) 2716 type 16K EPROMs

Programming Voltage:	25V ±0.3V
Programming Time:	50 ±5ms

■ 1–3 External Appearances

The external appearences of PECKER–I is shown in Fig. 131.

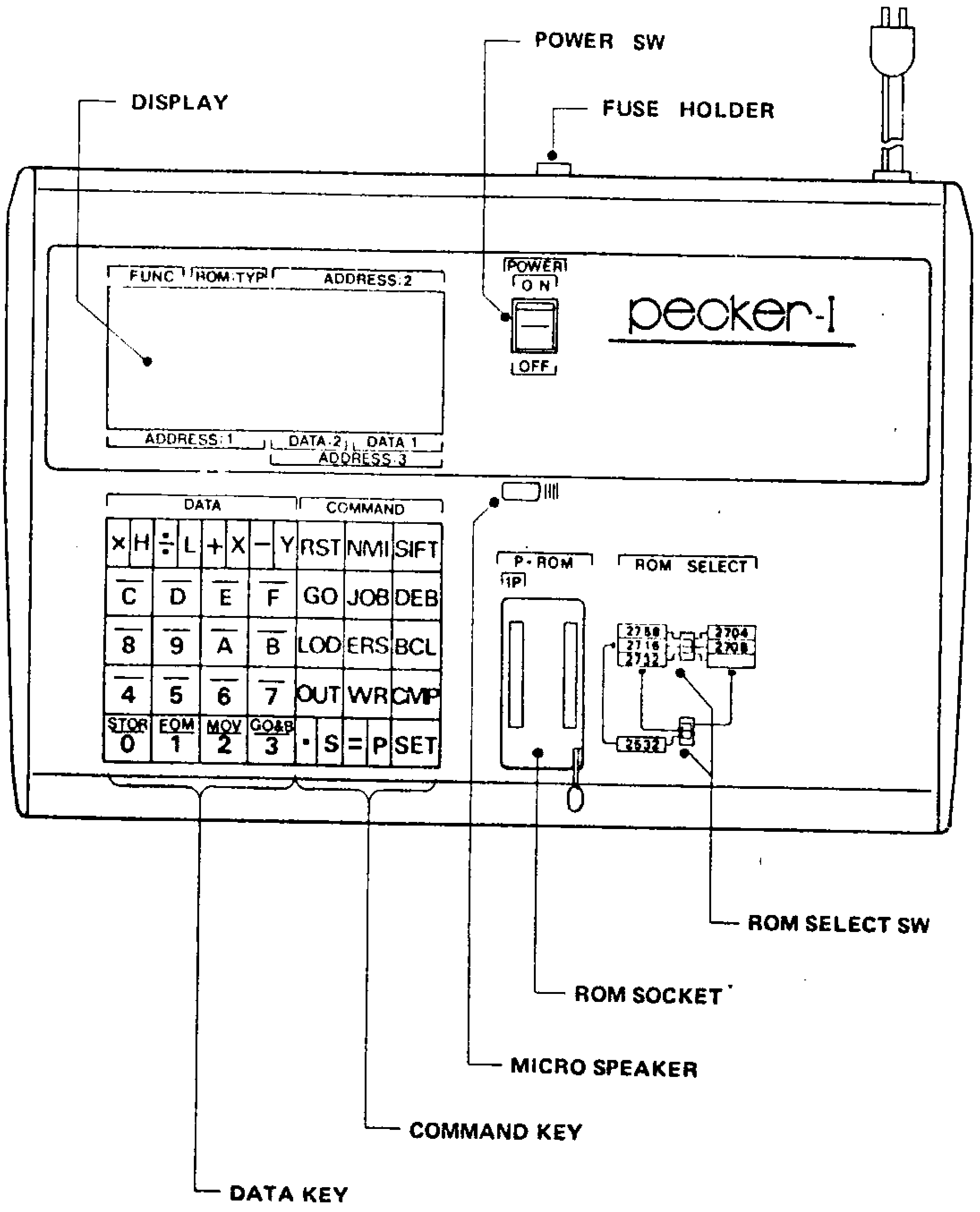
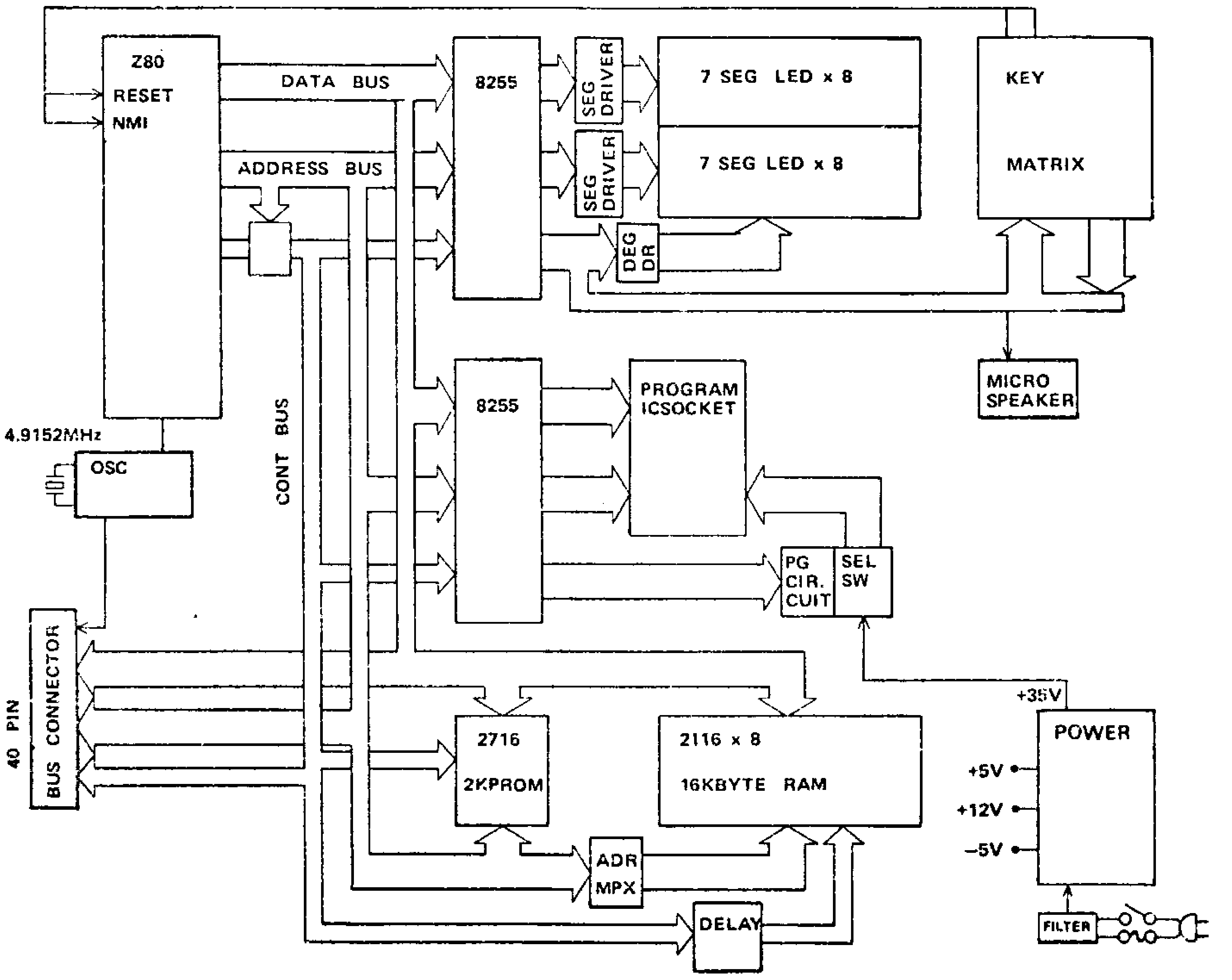


Fig. 131 External Appearances of PECKER–I

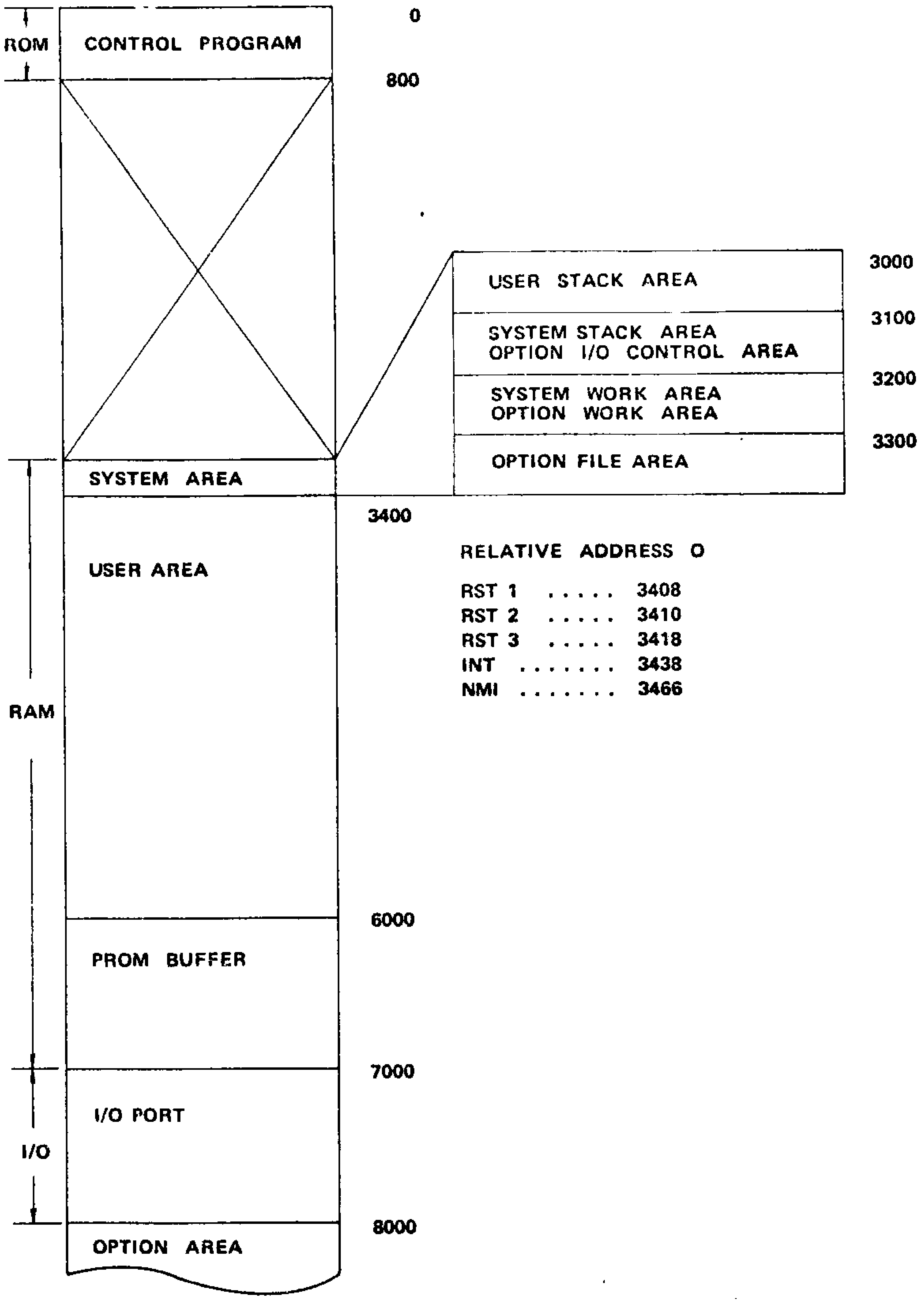
▪ 1–4 Function of the PECKER–I

(1) Loading the content of the master ROM into the RAM buffer.
(2) Program the content of the RAM buffer into the PROM.
(3) Comparing the data in the RAM buffer and the data in the PROM.
(4) Clearing the content of the RAM buffer.
(5) Erase-checking the content of the PROM.
(6) Displaying the content of the PROM and/or the RAM, and updating the content of the RAM.
(7) Block updating the content of the RAM.
(8) Block transfer of the data between RAMs.
(9) Displaying the Z-80 registers (FLAG, A, B, C, D, E, H, L, IX, IY, SP, PC)
(10) Updating the content of the above registers.
(11) Execution of programs, and designating up to two breakpoints.
(12) Turning ON and OFF the sound from the speaker by software.
(13) Indication of the key input confirmation, completion of the programming, error,etc., by a microspeaker.
(14) Displaying the type of PROM selected.
(15) Has the I/O control command used when the optional board is connected.
(16) Has the address increment and decrement comands.

▪ 1–5 PECKER–I BLOCK DIAGRAM



■ 1-6 MEMORY MAP



RELATIVE ADDRESS O

RST 1	.....	3408
RST 2	.....	3410
RST 3	.....	3418
INT	....	3438
NMI	....	3466

[2. How to operate the PECKER –I]

This chapter describes how to operate the PECKER–I.

▪ 2–1 Power On

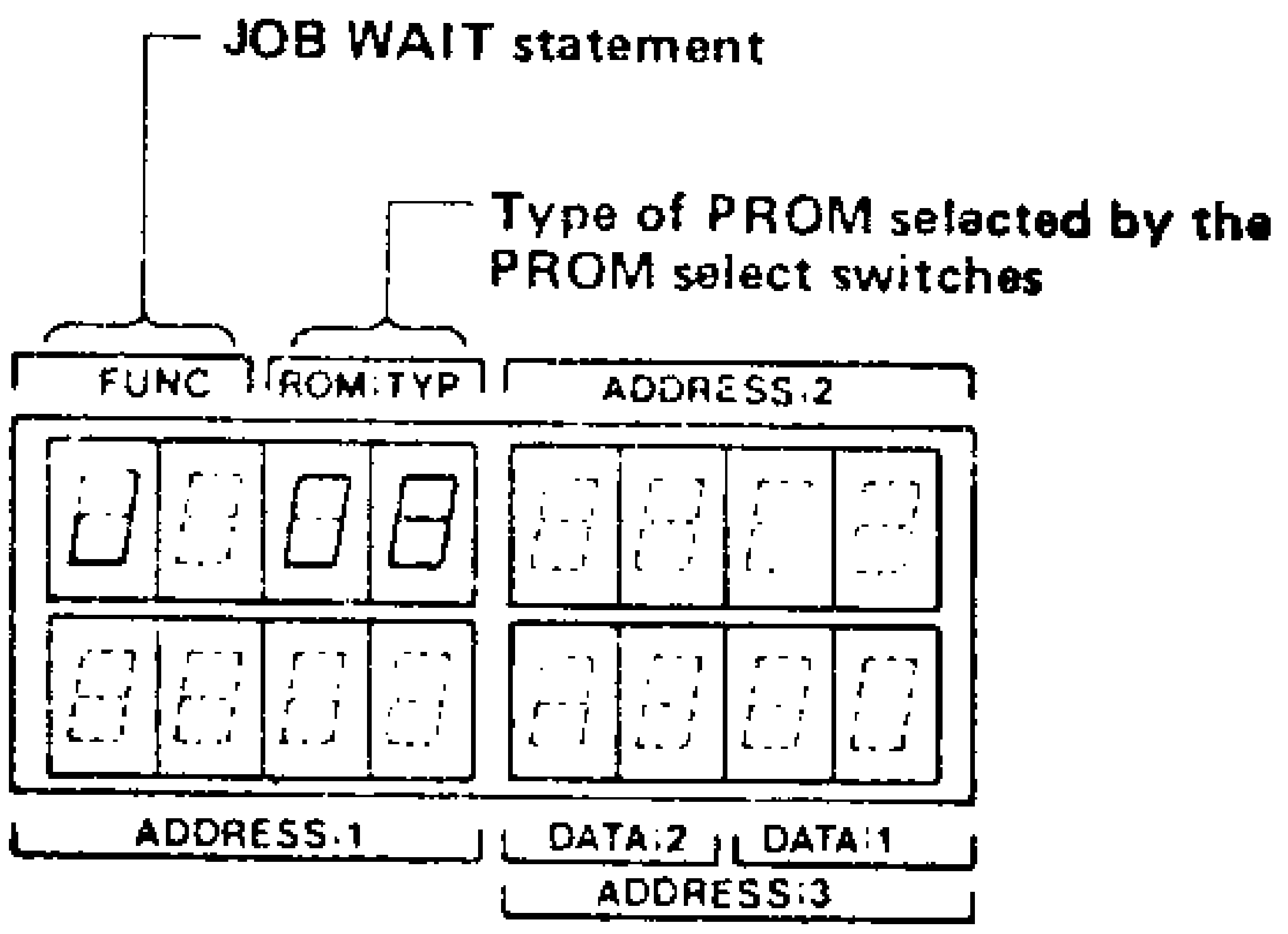
Power is turned on as follows.

(1) Confirm that the POWER-SW is in the OFF position (thrown toward the user).

(2) Confirm that no PROM is inserted in the PROM socket.

(3) Connect the AC plug to the plug socket.

(4) Set the POWER-SW on.
These procedures make the displays look as follows. (JOB WAIT state).



(Note)
In case JOB WAIT state is not displayed, the user is requested to press the $\boxed{\text{RST}}$ key to return to the JOB WAIT state.

Fig. 211 The display for the JOB WAIT state

▪ 2–2 Microspeaker

A key input acknowledgement, error, completion of a command execution, etc. are signaled by an audible sounds from a microspeaker.
The audible sounds are emitted as follows.

- ○ Key input acknowledgement . . . 0.1 sec.
- ○ Error . . . . 3 x 0.1 sec.
- ○ Completion of programming . . . 3 x 0.4 sec.
- ○ Completion of command . . . . . 0.4 sec.

▪ 2–3 How to Select the PROM

The type of PROM is selected by a combination of the settings of the A–SW, and B–SW as shown in Fig. 231.

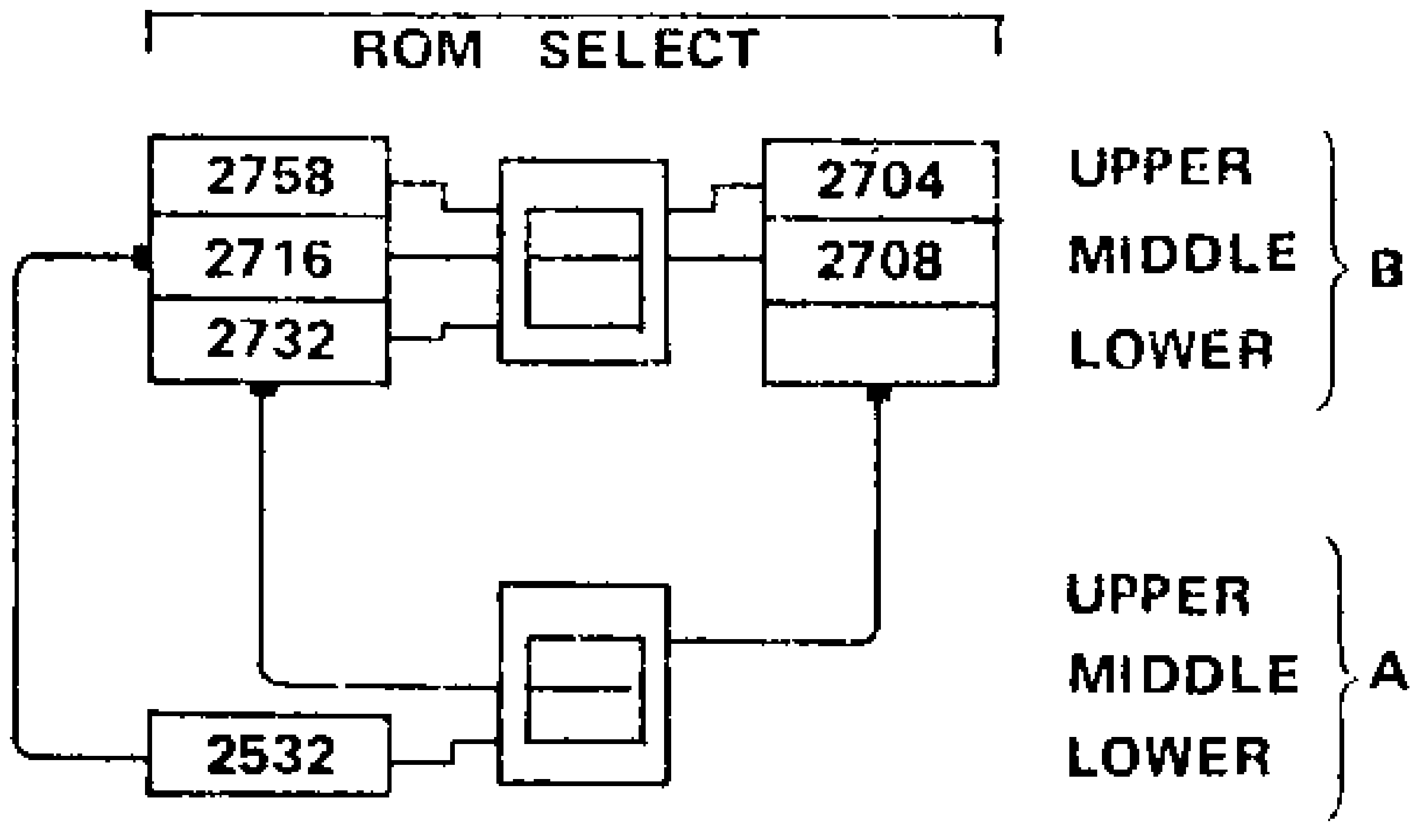


Fig. 231 PROM Select SW

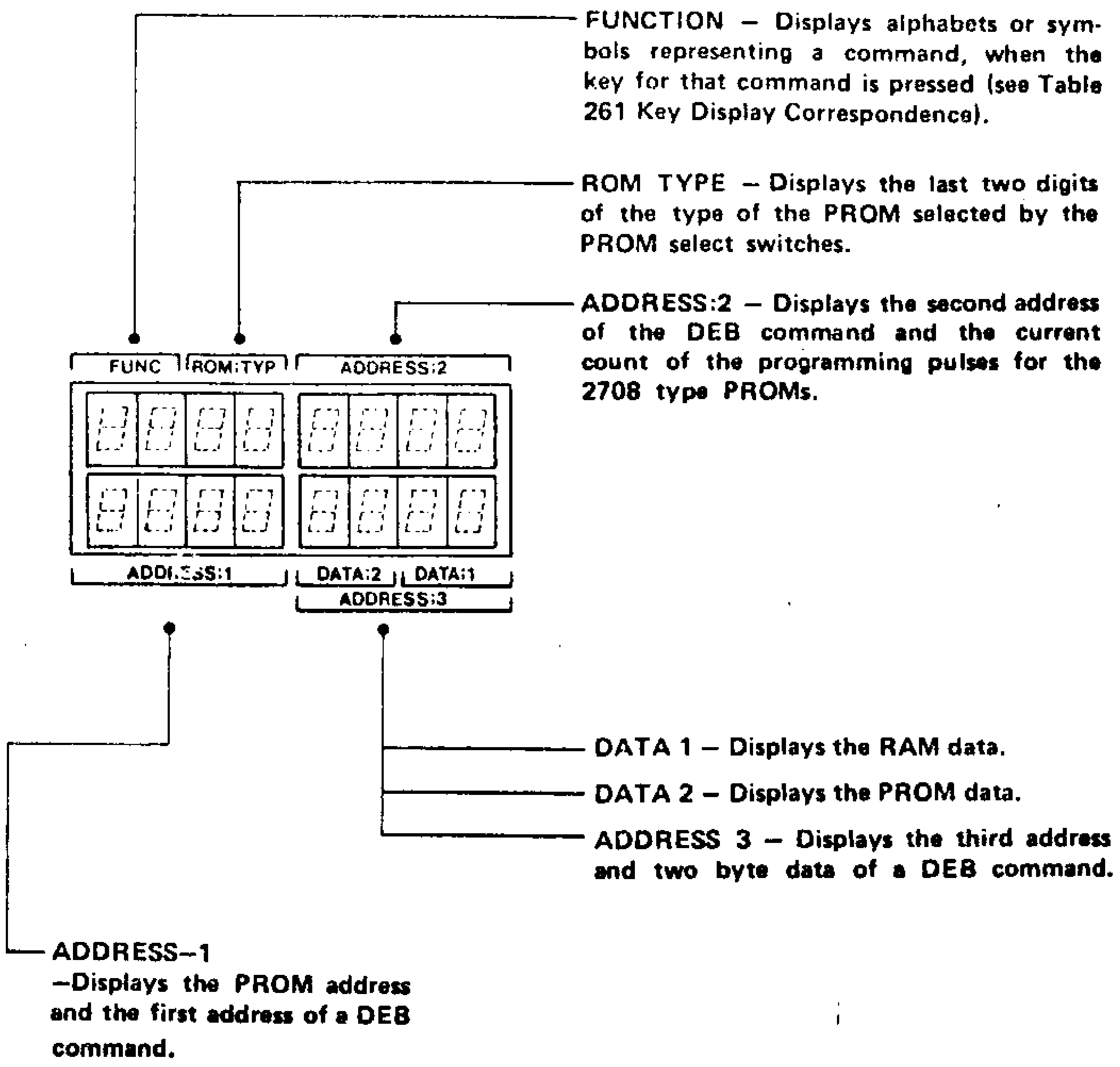
The A–SW is for selecting the product line, the upper position for the 2708 and its equivalents the middle position for the 2716 and its equivalents, and the lower position for the 2532 from T.I. The B–SW selects the actual type of the PROM of the product line selected by the A–SW.
The type of the selected PROM is displayed in the last two digits of the [PROM: TYP] display. Fig.232 shows how the selection and the display go.

PROM	A-SW	B-SW	ROMTYP
2704	UPPER	UPPER	04
2708	UPPER	MIDDLE	08
2758	MIDDLE	UPPER	58
2516 2716	MIDDLE	MIDDLE	16
2732	MIDDLE	LOWER	32
2532	LOWER	MIDDLE	3.2.

Fig. 232 PROM type, switch settings, and the display

■ 2–4 Function of the Display

The display consists of 16 7-segment LEDs.



FUNC
ROM:TYP
ADDRESS:2
ADDRESS:1
DATA:2
DATA:1
ADDRESS:3
FUNCTION – Displays alphabets or symbols representing a command, when the key for that command is pressed (see Table 261 Key Display Correspondence).
ROM TYPE – Displays the last two digits of the type of the PROM selected by the PROM select switches.
ADDRESS:2 – Displays the second address of the DEB command and the current count of the programming pulses for the 2708 type PROMs.
DATA 1 – Displays the RAM data.
DATA 2 – Displays the PROM data.
ADDRESS 3 – Displays the third address and two byte data of a DEB command.
ADDRESS–1 –Displays the PROM address and the first address of a DEB command.

Fig. 242 Function of the Display

■ 2–5 Mounting and Dismounting a PROM

Insert a PROM into the 24 pin DIP lever socket, as shown in Fig. 251, with the lever set upright. Then pull down the lever toward you, while holding the PROM by hand.

For dismounting the PROM, set the lever upright and remove the PROM.

Users are requested to be careful not to insert the PROM in a wrong direction, as it would damage the PROM.

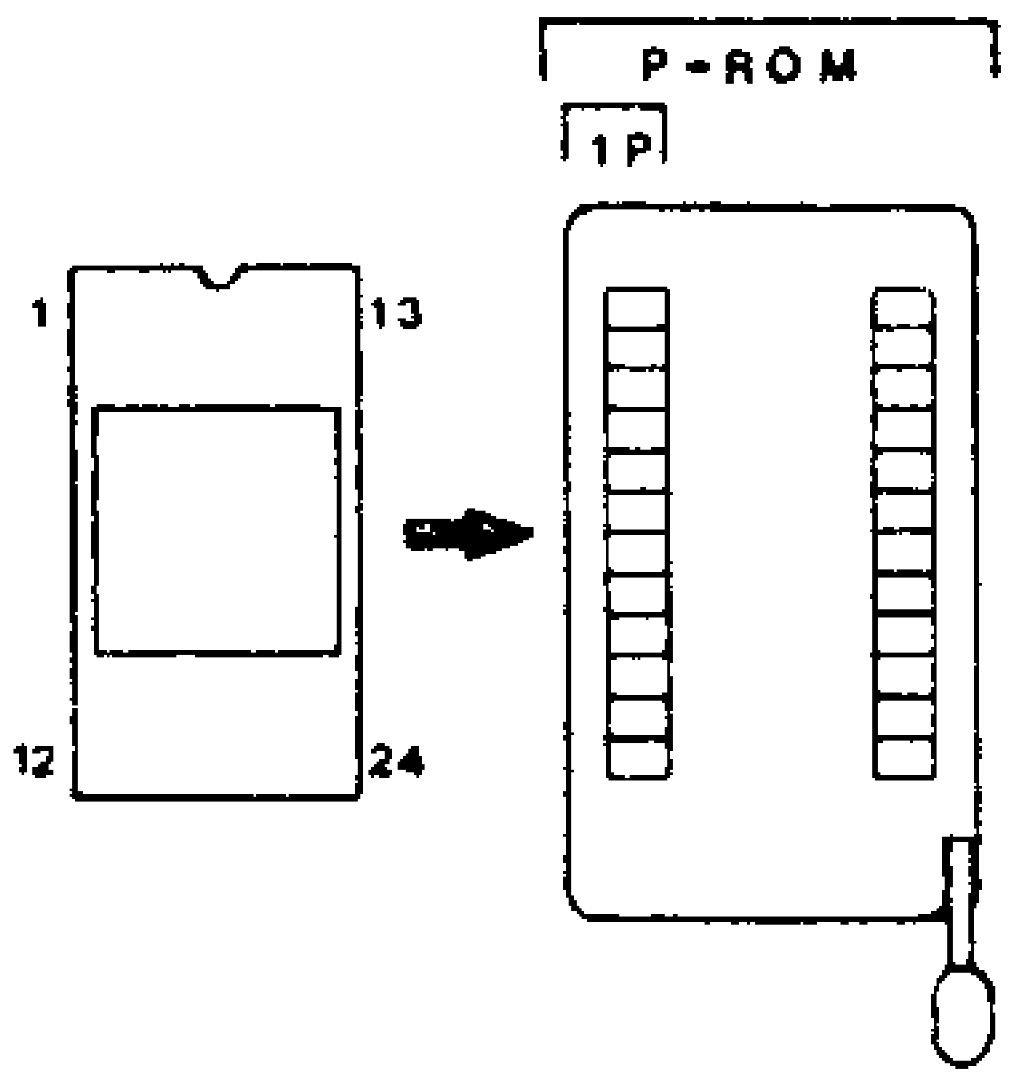


Fig. 251 How to mount a PROM

■ 2–6 The Role of Keys and Corresponding Displays

The keys and their corresponding displays are shown in Table 261.

Table 261 Key – Display Correspondence

DISPLAY	0	1	2	3	4	5	6	7	8	9	A	b	C	d	E	F
KEY	STOR 0	FOM 1	MOV 2	GO&B 3	4	5	6	7	8	9	A	B	C	D	E	F

DISPLAY	‖	L	‾ _	‖	5	P	Ξ	o	U	C.	L	E	b	G	J	d
KEY	× H	÷ L	+ X	− Y	• S	= P	SET	OUT	WR	CMP	LOD	ERS	BCL	G O	JOB	DEB

DISPLAY	╲	╲	╲
KEY	RST	NMI	SIFT

[3. Description of Commands needed for Programming]

Commands needed for programming and operations of these commands are described in detail in this section.

■ Definition of the Symbols Used in the Description.

(1) $\boxed{}$: Indicates that the command key or data key specified in the $\boxed{}$ is to be pressed or has been pressed.

Examples

$\boxed{\text{SET}}$	Press the SET key
$\boxed{3}\boxed{F}\boxed{9}$	Press the data keys in the order of 3, F, and 9.

(2)	DD :	Implies arbitrary data to be input in "HEX" (hexa decimal).
	XXXX :	Implies an arbitrary first address to be input in "HEX".
	YYYY :	Implies an arbitrary second address to be input in "HEX".
	ZZZZ :	Implies an arbitrary third address to be input in "HEX".

Examples

$\boxed{D}\boxed{D}$	Implies to key in arbitrary "HEX" data.
$\boxed{Y}\boxed{Y}\boxed{Y}\boxed{Y}$	Implies to key in an arbitrary second address in "HEX".

(3) (   ) : Implies the sound (accoustic emission) from the microspeaker, and a "• " in the (   ) indicates a short sound while a "—" implies a long sound.

Examples

(– – –)	Implies a state of the microspeaker having emitted three long sounds.
(•)	Implies a state of the microspeaker having emitted a short sound.

(4) [   ] A1 : The numbers and the letters in [   ] imply what is displayed, while the abbreviations which follow that imply where it is displayed.

Each abbreviation stands for as follows.

FN – FUNC	A1 – ADDRESS : 1
RT – ROM : TYP	A2 – ADDRESS : 2
D1 – DATA : 1	A3 – ADDRESS : 3
D2 – DATA : 2	

Example

[5B]D1	"5B" is displayed on the "DATA:1" section of the display.

(5) ↓ and ↑ : Implies mounting and dismounting of a PROM into and from the socket. ↓ implies mounting a PROM into the socket, while ↑ implies a removal from the socket. When a ↓ or ↑ is followed by a letter M, W, or E, that implies mounting or dismounting of a master (=M), programmed (=written=W) or erased (=E) PROM.

Example

↓M	Implies an insertion of a master PROM into the socket.

(6) <JOB>: Implies that the PECKER is in the JOB WAIT state.

(7) ○ : Indicates an execution of a command or a branch resulting from the execution.

Note: In most of the cases, the system drives the microspeaker for acknowledging a key input to emit a short sound. This audible signal, however, is not mentioned in the following descriptions of each command.

■ 3–2 JOB Command

A command must always be input in the JOB WAIT state. When the PECKER is not in the JOB WAIT state, or when the user wants to abort the execution of the current command, he can force the PECKER back into the JOB WAIT state by pressing the JOB key.

An erroneous key operation also forces the PECKER into the JOB WAIT state with an audible error signal (• • •).

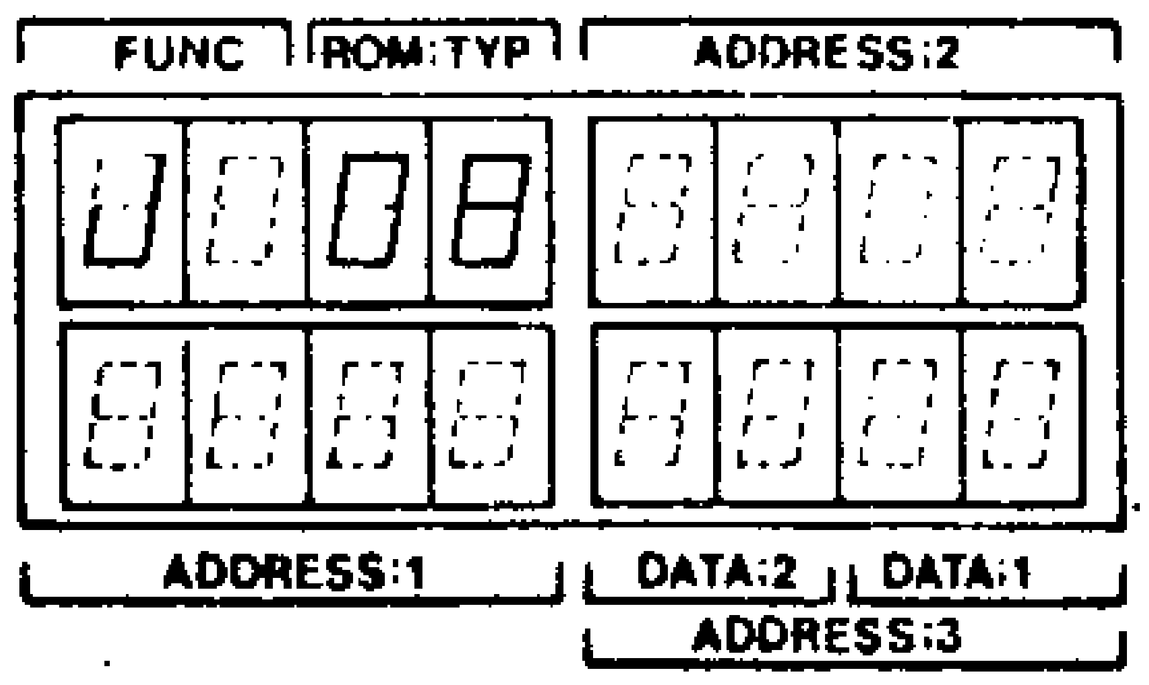


Fig. 321 JOB WAIT STATE

▪ 3–3 LOAD Command

A LOAD command is used for loading the content of the PROM on the PROM socket into the RAM.

≪Operation≫ $\boxed{\text{L O D}}$ $\boxed{\text{S E T}}$

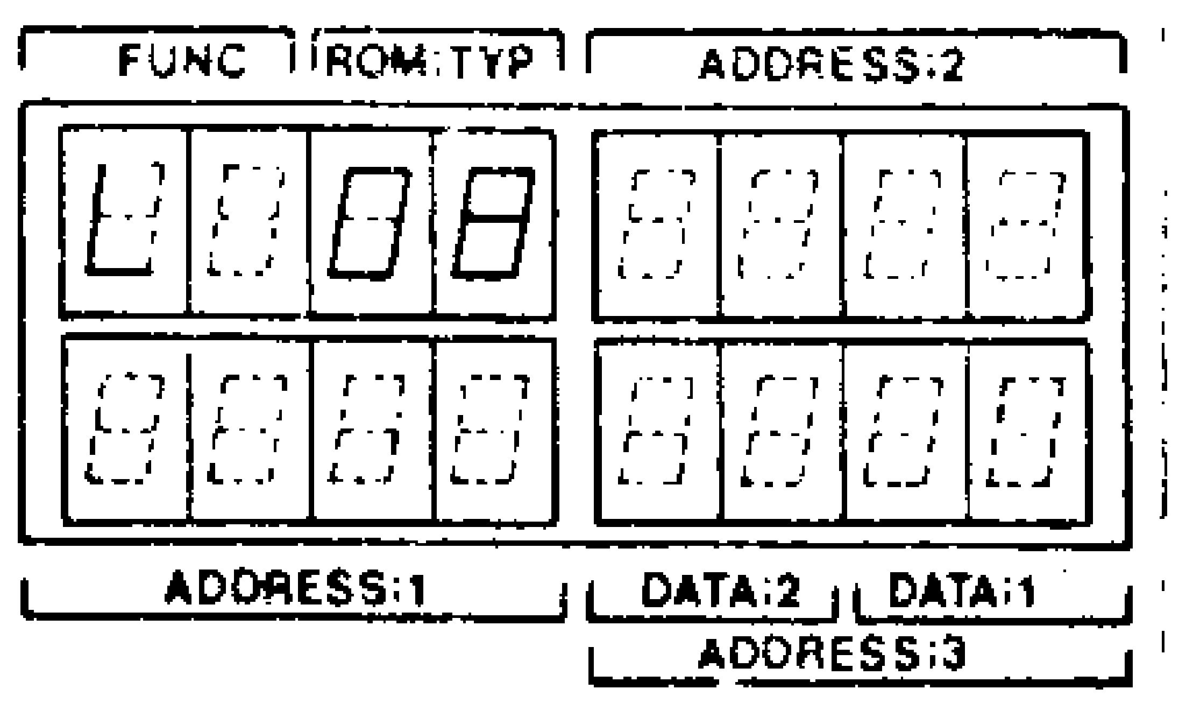


Fig. 322 The Display for a LOAD

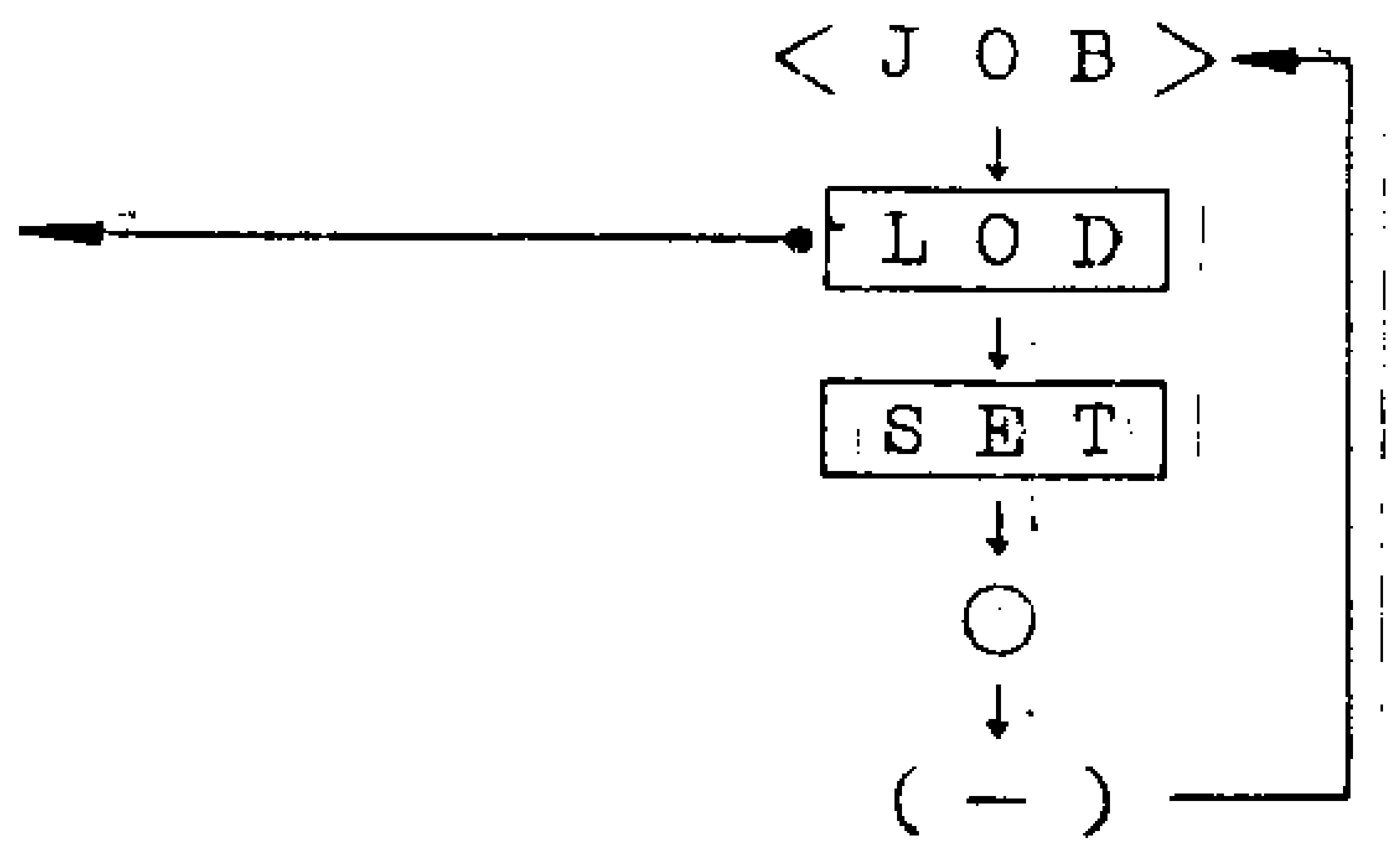


Fig. 323 The Flow for the LOAD Command

In case the LOAD and other commands which make use of the RAM (buffer) are executed, the PECKER automatically secures the RAM address and area equivalent to the size (number) of bytes of the PROM selected by the PROM select switches.

The PROM capacity and memory allocation are related as shown in Table 324.

Table 324 PROM Capacity and Buffer Address

BYTE	RAM AREA
0.5 K	6000 – 61FF
1 K	6000 – 63FF
2 K	6000 – 67FF
4 K	6000 – 6FFF

■-3-4 ERASE Command

An ERASE command is used for checking if the data in the PROM on the PROM socket are completely erased into the state of "FF".

≪Operation≫ [ERS] [SET]

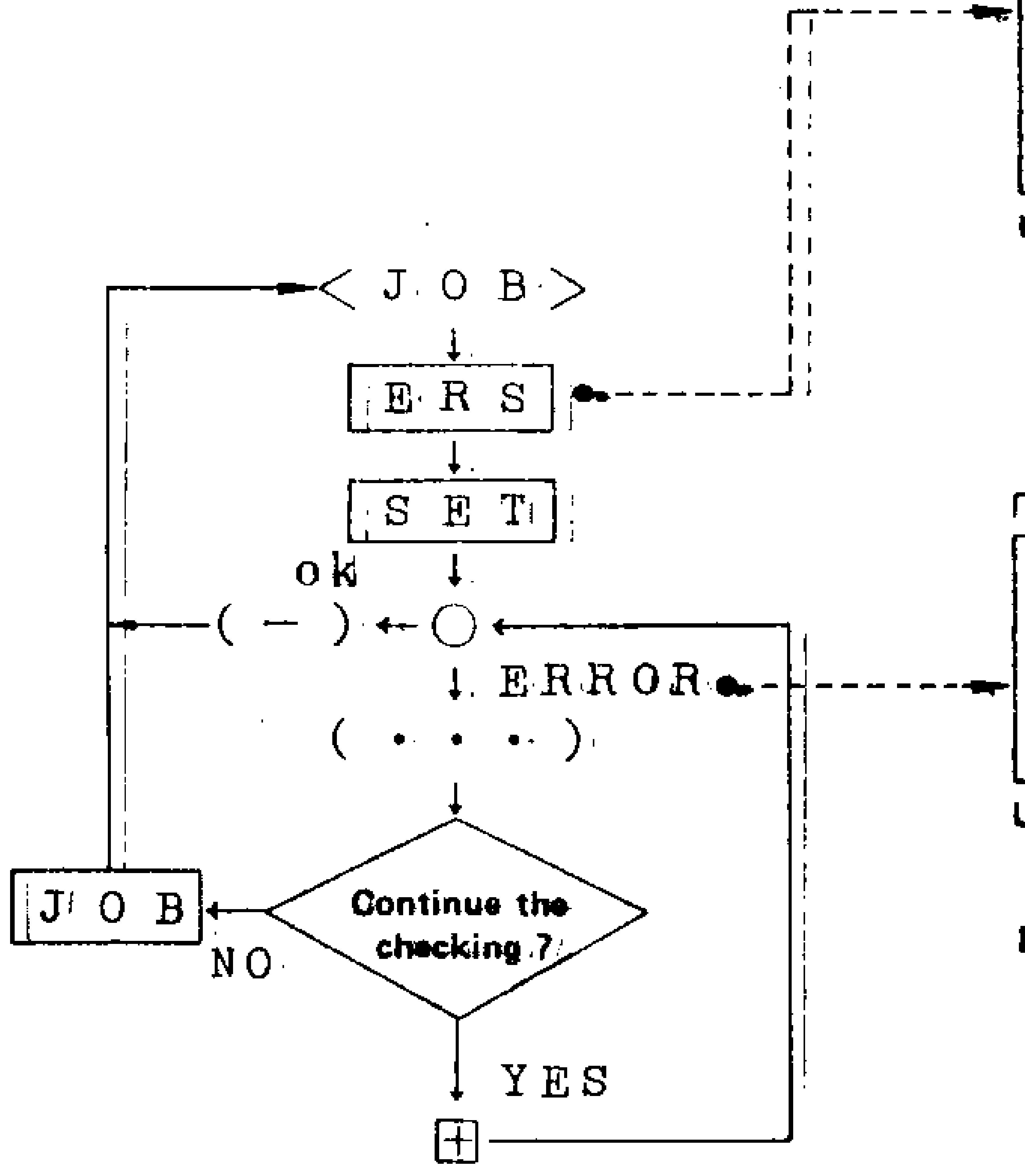


Fig. 341 The Flow for an ERASE Command

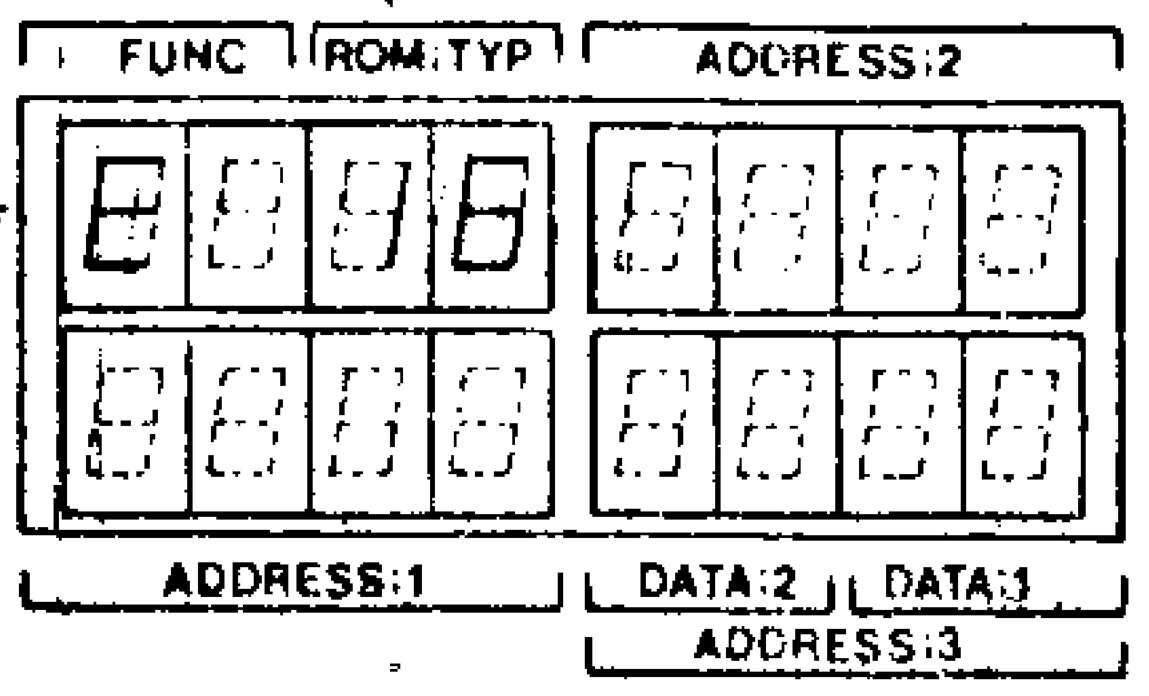


Fig. 342 ERASE Command

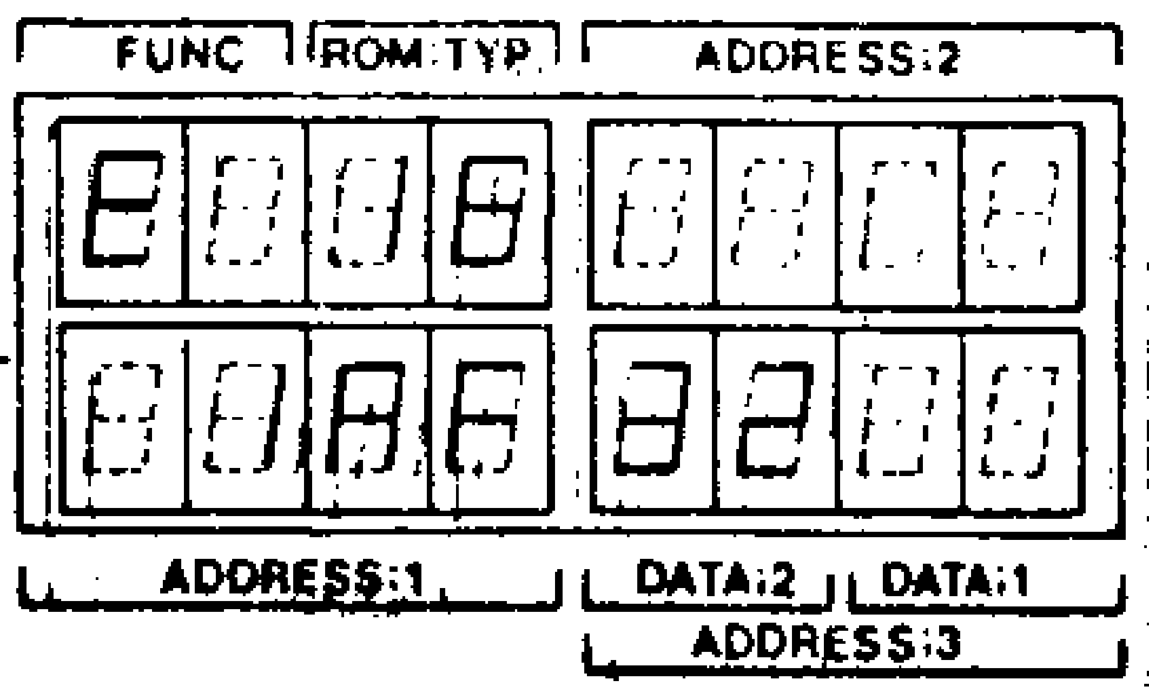


Fig. 343 The Display for an ERASE

If any location has not been erased, that address and the data are displayed on the ADDRESS : 1 and DATA : 2 respectively, and the checking activity is suspended.

The [+] key is to be pressed for continuing the checking, and the [JOB] key for aborting the check. Fig. 343 shows that the content of the address 1AF of the 2716 PROM under the erase-check is "32" instead of "FF".

■ 3–5 COMPARE Command

A COMPARE command is used for comparing the data in the PROM on the PROM socket with the data in the RAM (buffer). When there is a COMPARE error, the comparison is suspended while the address, the data in the RAM, and the data in the PROM are displayed on the ADDRESS : 1, DATA : 1, and DATA : 2.

For continuing the comparison, the user is to press the $\boxed{+}$ key, while for aborting the comparison, the $\boxed{\text{JOB}}$ key is to be pressed.

≪ Operation ≫ $\boxed{\text{CMP}}$ $\boxed{\text{SET}}$

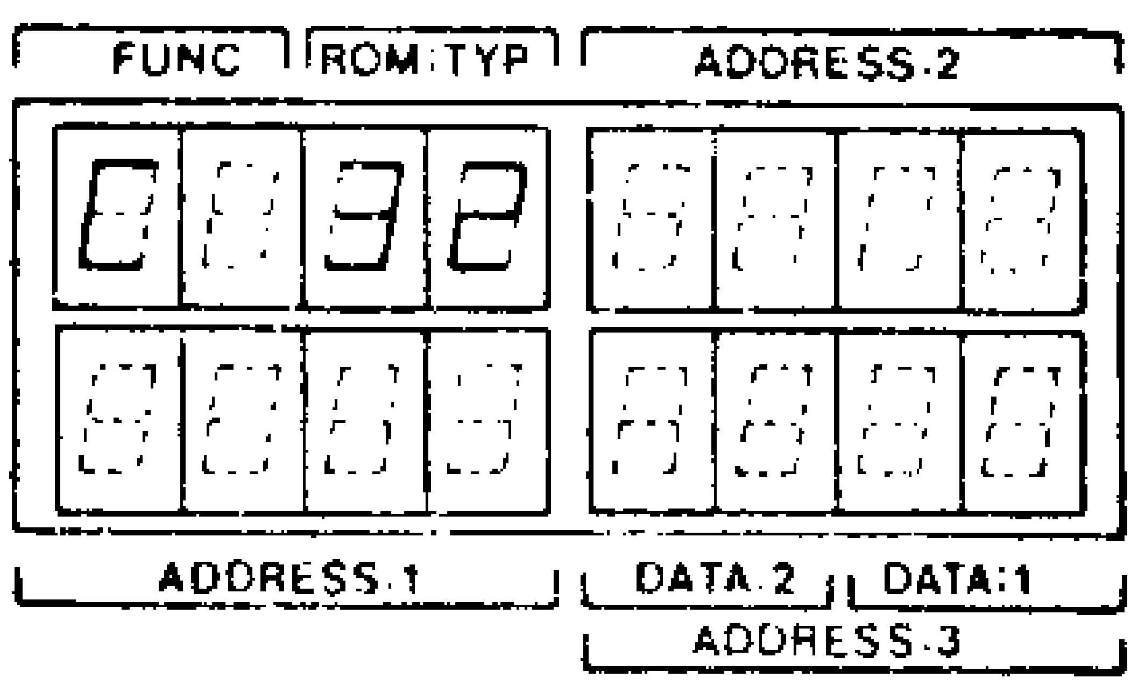


Fig. 352 COMPARE Command

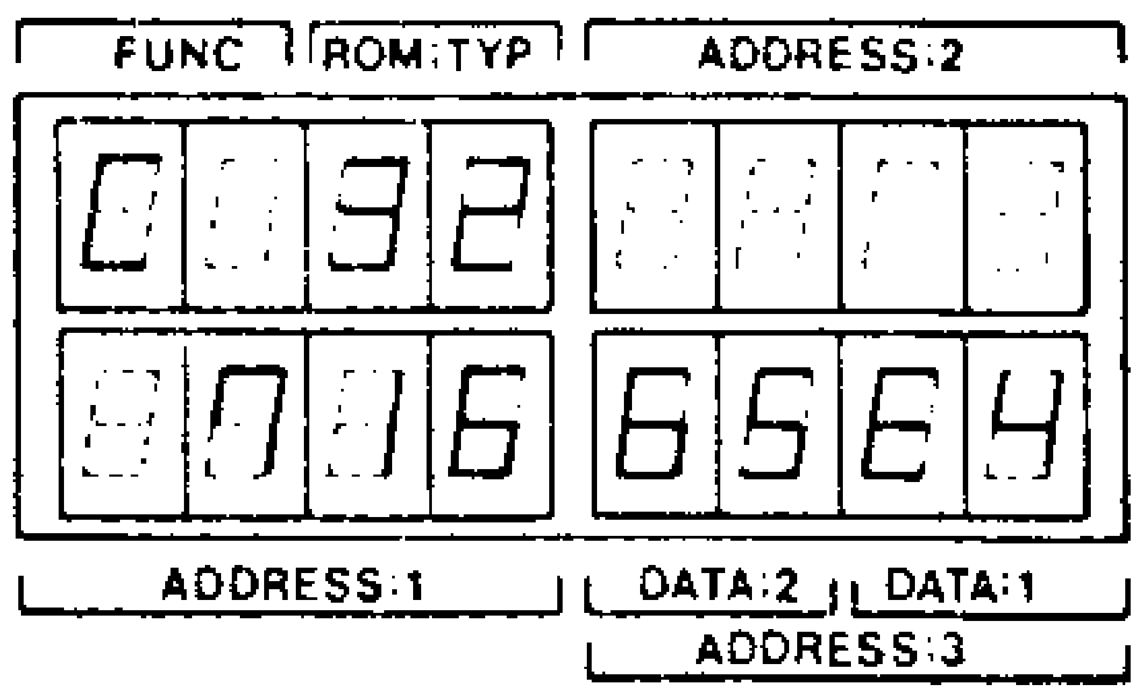


Fig. 353 The Display for a COMPARE Error

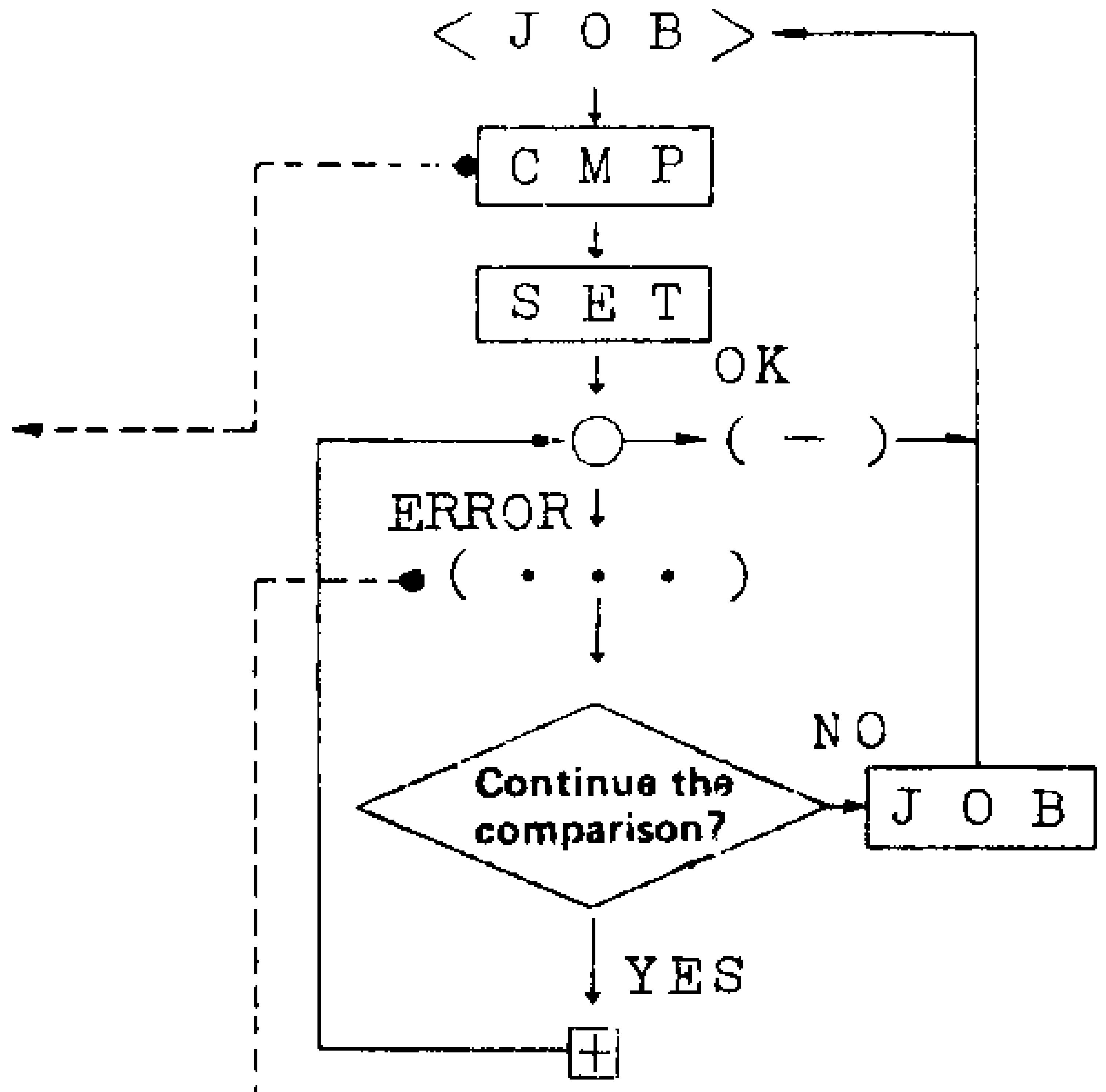


Fig. 351 The Flow for the COMPARE Command

■ 3–6 BUFFER CLEAR Command

A BUFFER CLEAR command is used for clearing (writing "FF"'s onto) the RAM (buffer) area equivalent in size to the number of bytes of the PROM specified by the PROM select SW's (see Table 324). This command is convenient for preparing new data in the buffer, and also for other operations.

≪ Operation ≫ $\boxed{\text{BCL}}$ $\boxed{\text{SET}}$

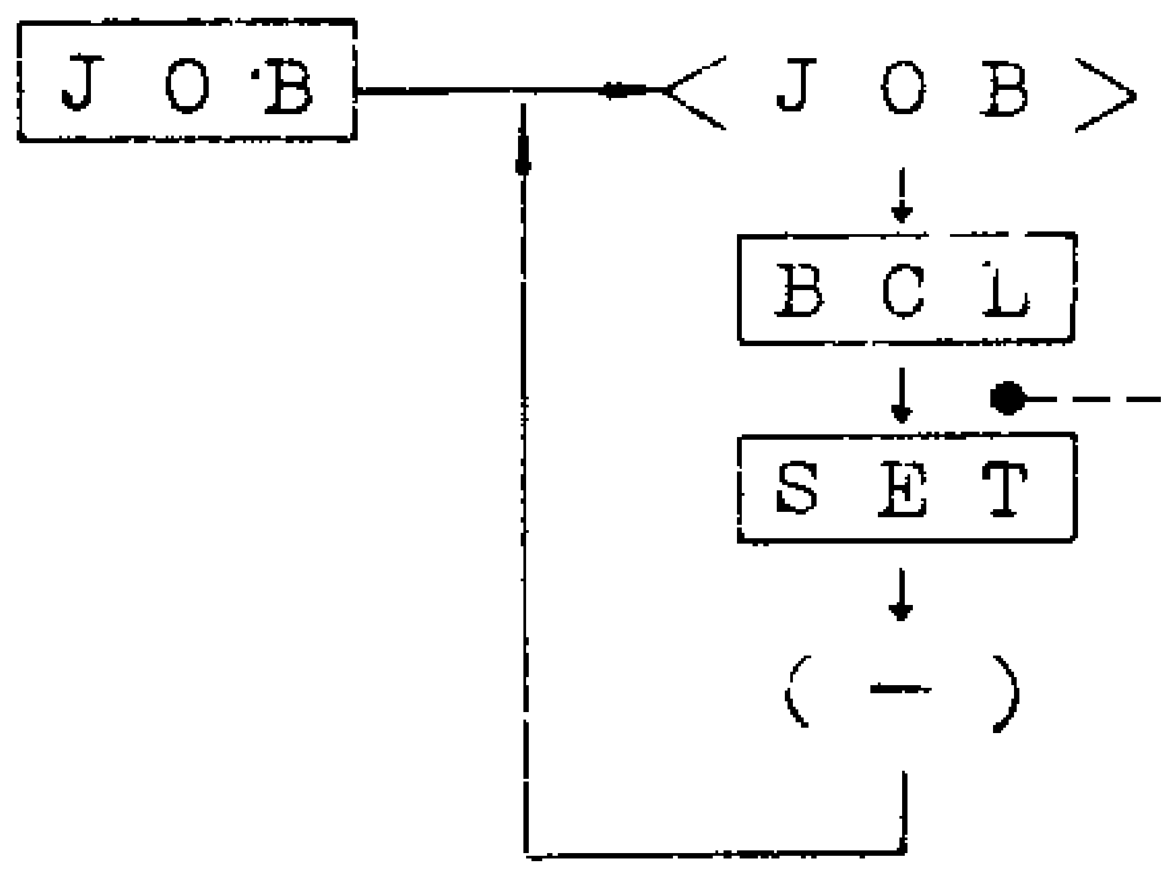


Fig. 361 The Flow for the BUFFER CLEAR Command

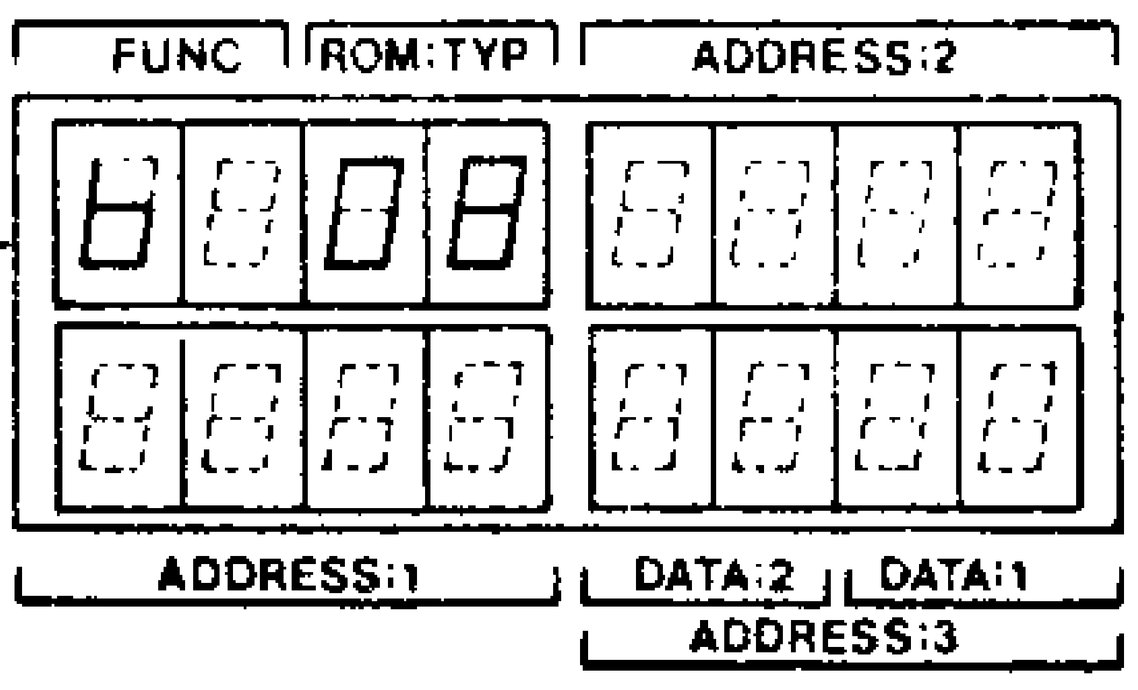


Fig. 362 The Display for a BUFFER CLEAR Command

■ 3–7 WRITE Command

A WRITE command is used for programming the data in the RAM (buffer) onto the PROM on the PROM socket.

The programming procedures for the 2708 type PMOM and the 2716 type device are somewhat different. In other words, although the internal operations and displays for these two kinds of PROMs are a little different, the external (actual) operations are identical.

(Caution: Please note that the PECKER cannot be used for programming a PROM other than the type specified by the switches and trying to do so may damage the PROM.)

Upon completion of programming, a COMPARE operation is automatically performed. In case a COMPARE error occurs, please refer to ■ 3–5 COMPARE Command.

≪ Operation ≫ $\boxed{\text{WR}}$ $\boxed{\text{SET}}$

(1) Programming the 2708 type PROM

For programming the 2708-type, the PROM manufactures specify that pulses of 100 μs ~ 1ms should be applied in a loop programming manner (starting from the first address to the last and repeating the process) until the total (cummulative) programming time per address reaches 100 ms.

The PECKER–I is designed to repeat the loop programming until the total programming pulse duration per address reaches 100 ms using programming pulses of 670 μs.

During programming, "ADDRESS:2" is used for the display, using the last two digits for displaying the total programming time, while the first two digits for the programming time at the point when it has been confirmed for the first time that all of the addresses have been programmed. Given N for the display for the total programming , and n for the time when programming was confirmed, the following hold.

Total Programming Time = N × 2 ms
Programming Confirmaiton Time = n × 2 ms
2ms= 670 μs (pulse width) × 3 pulses

(Note: Actual width of the programming pulse of the PECKER is 670 μs, and single programming pulse is applied to each address each time in a round-robin fasion, repeating the process 3 times to increment the displayed value of the N by 1. Upon incrementing the N, a comparison is made for all the data, and if OK, that N becomes the n, representing the programming confirmation time to be displayed, and there will be no further change in the display for the n.)

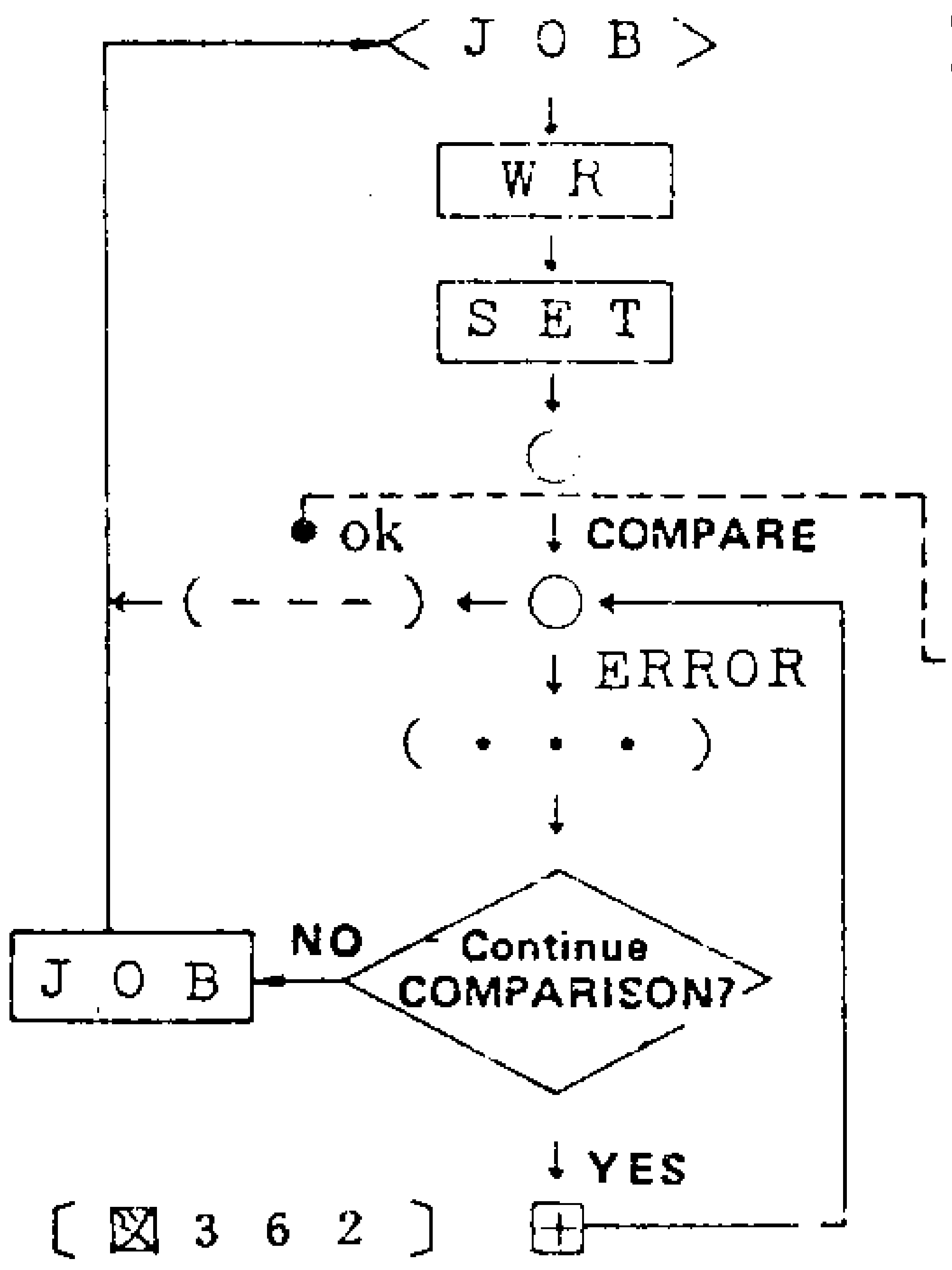
Programming is achieved by meeting the following conditions.

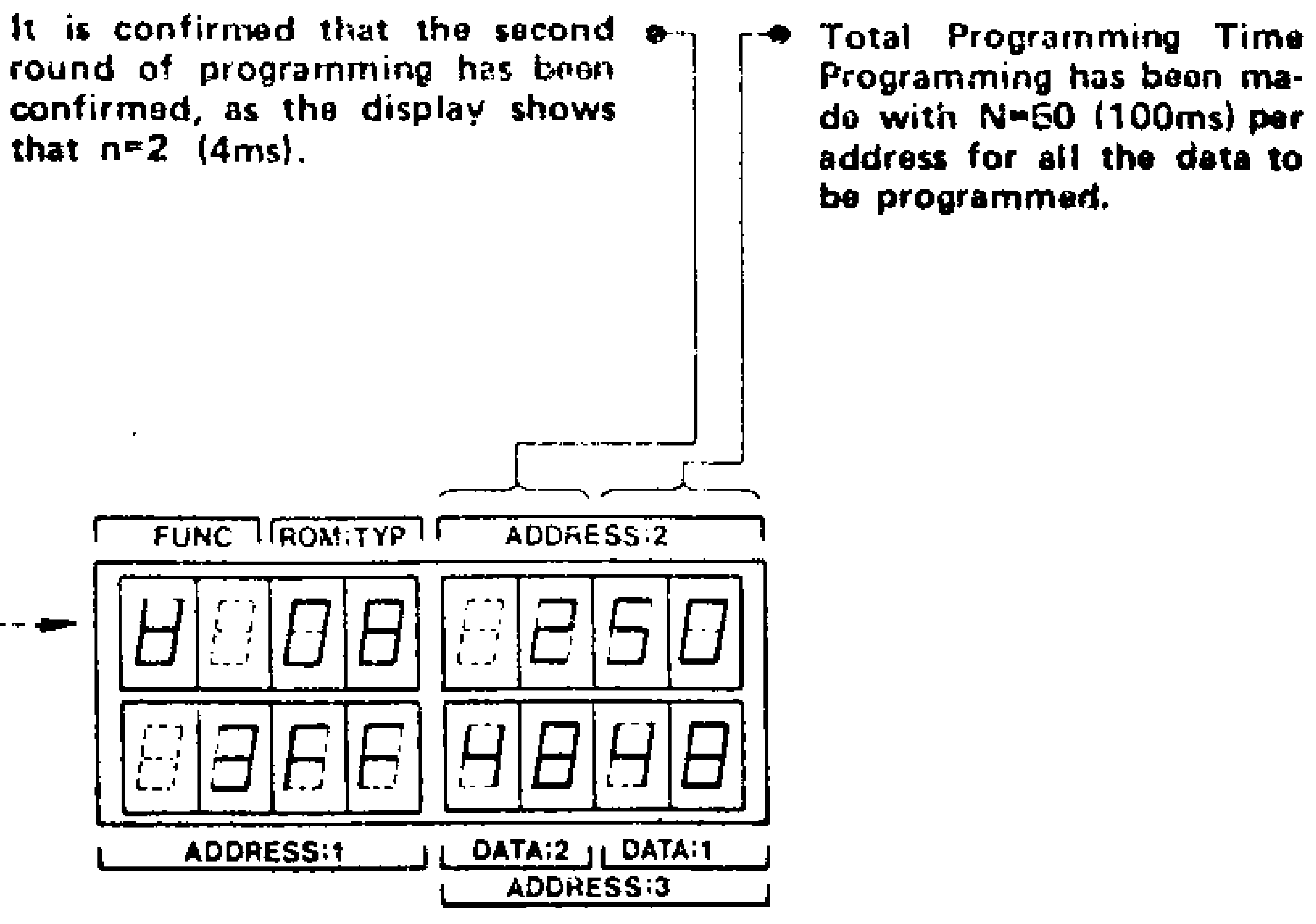
1. $50 \leqq N \leqq 100$ (N must be between 50 and 100. Please note that 100 is displayed as 00.)
2. $n \leqq 25$ N=50 (If n is equal to or smaller than 25, total programming time of 100 μs per address must be applied.)
3. $25 < n \leqq 50$ N=2n (If n is greater than ms and equal to or smaller than 50, programming time of twice as long as the confirmation time must be applied.)

In case programming is not completed when n has reached 50, the PECKER enters into the programming error state (with the tone from the micro-speaker).

(2) Programming the 2716 type PROM

The programming procedure for the 2716 type 16K PROMs is different from that for the 2708-type, and programming is achieved by applying a 50 ms programming pulse to each address. Thus the displays given are different from those for the 2708-type; namely the PROM address being programmed is displayed on "ADDRESS:1", while the data in the RAM are displayed on "DATA:1"

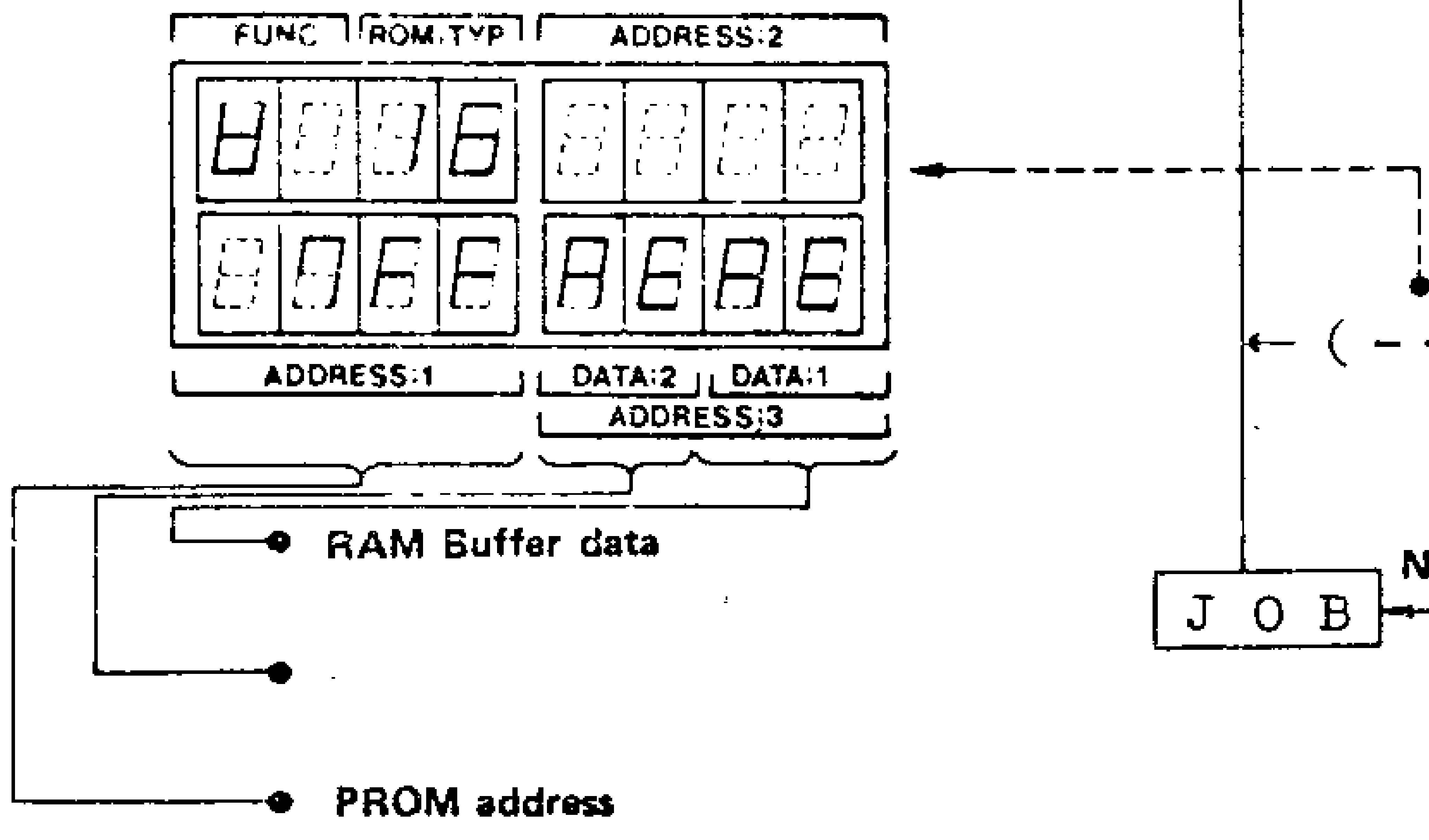




It is confirmed that the second round of programming has been confirmed, as the display shows that n=2 (4ms).
Total Programming Time Programming has been made with N=50 (100ms) per address for all the data to be programmed.
FUNC
ROM.TYP
ADDRESS:2
ADDRESS:1
DATA:2
DATA:1
ADDRESS:3

Fig. 361 Display for programming the 2708 type PROM

Fig. 362 Flow for programming the 2708 type PROM



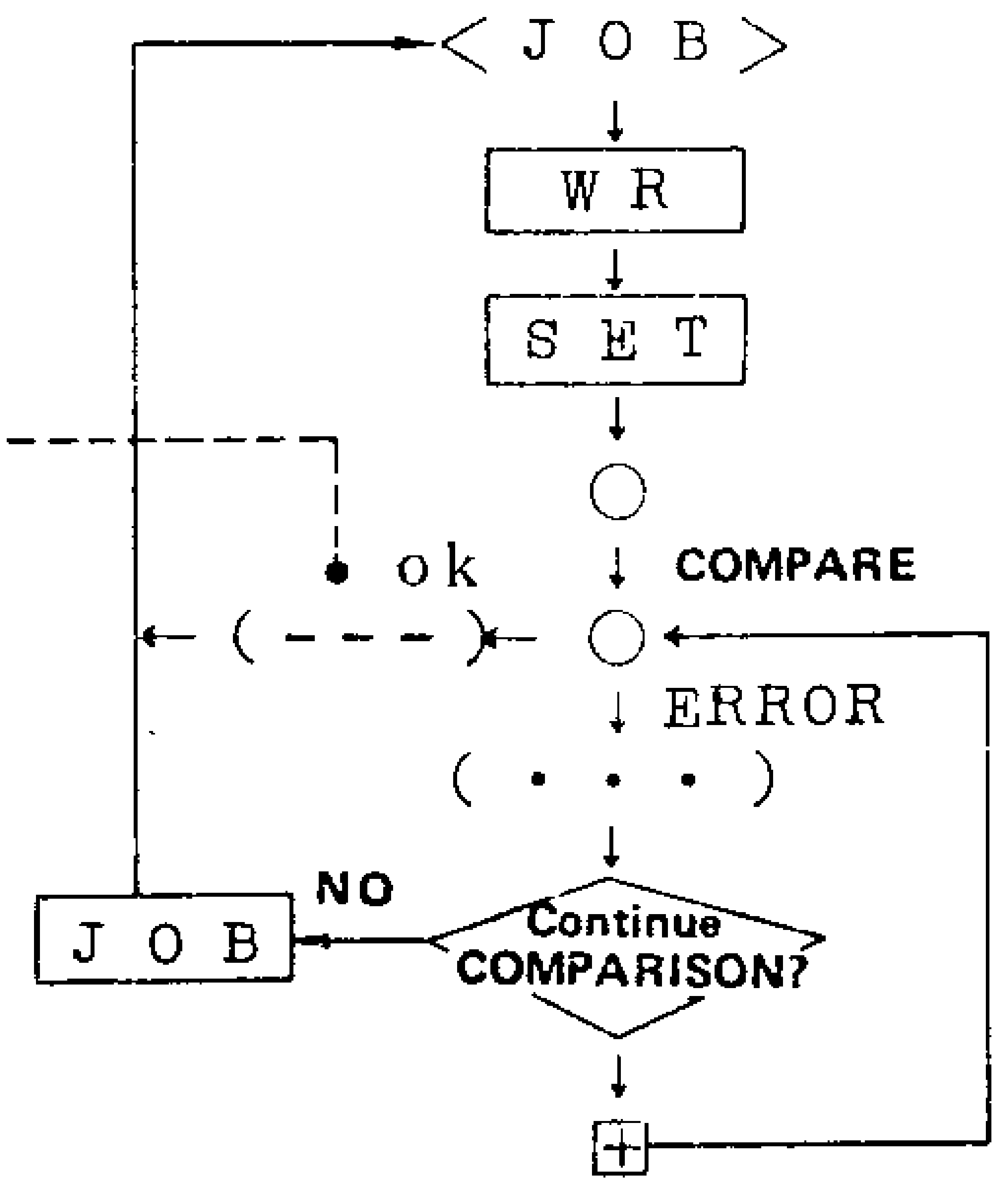


Fig. 364 Display for programming the 2716 type PROM

Fig. 363 Flow for programming the 2716 type

■ 3-8 DEBUG Command

A number of DEBUG commands are defined by a combination of the [DEB] key and data keys. Only 4 commands are described in this chapter.

- o [DEB] [0] → STORE Command
- o [DEB] [1] → FORMAL Command
- o [DEB] [2] → TRANSFER Command
- o [DEB] [4] → BUFFER UPDATE Command

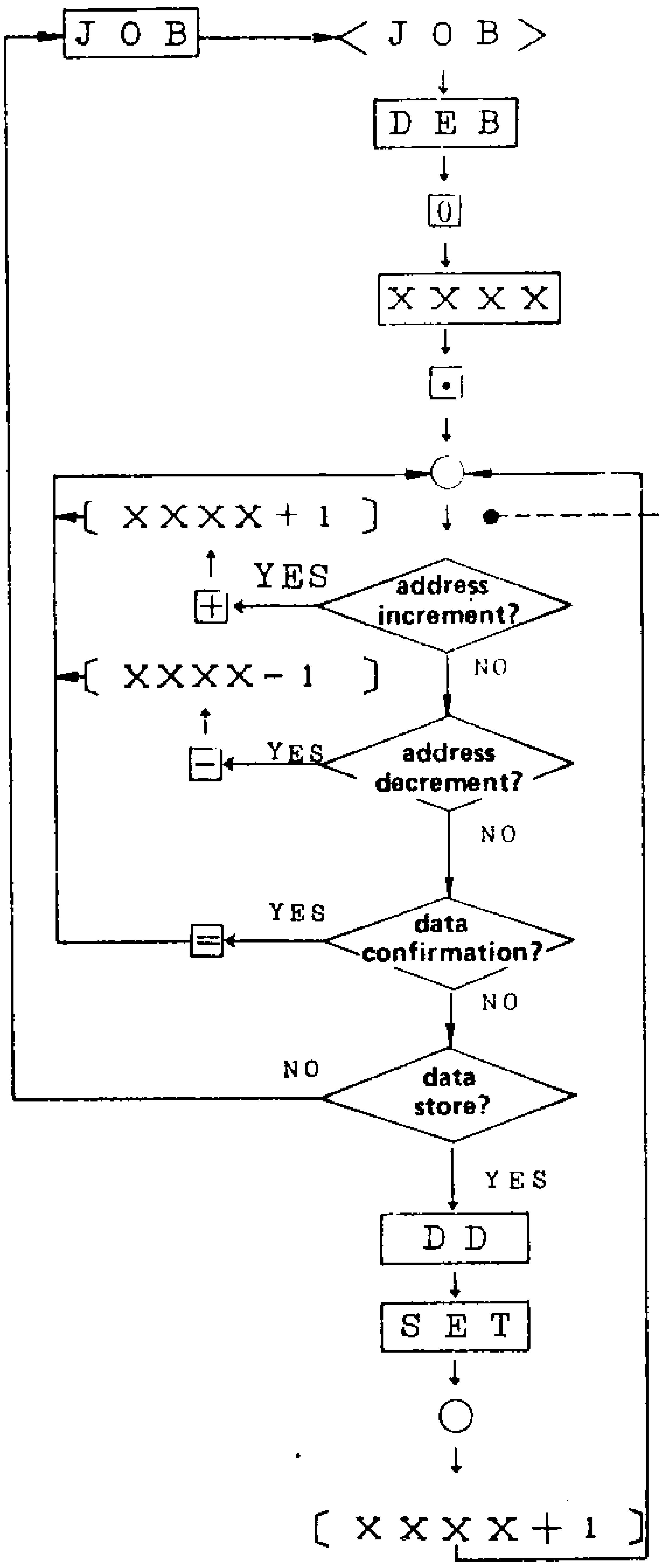


Fig. 371 Flow for the STORE Command

All of these commands are the editing commands which make use of the RAM buffer and are used for editing jobs such as producing or updating the data to be programmed. Keying-in (Inputing) through the data and address keys has zero-suppress and flow-out functions. Therefore, even if you press both 0 and A keys, only A is memorized and displayed (zero-suppression). Likewise, even if you press many keys, only the last two of the data keys and the last four of the address keys are memorized and displayed while the preceding inputs are all flowed out.

(1) STORE Command

A STORE command is used to see the content of the ROM and RAM, or to update the content of the RAM.

≪Operation≫

[DEB] [0] [X|X|X|X] [·]

([·] can be substituted by [SET] .)

By this operation the data at address "XXXX" are displayed on the "DATA 1", while address "XXXX", input through the key, is displayed on "ADDRESS:1". For continuing the confirmation of the data, press the [+] key to increment the address, and confirm the data sequentially. The [−] key is used to decrement the address, and the [=] key allows you to reconfirm the

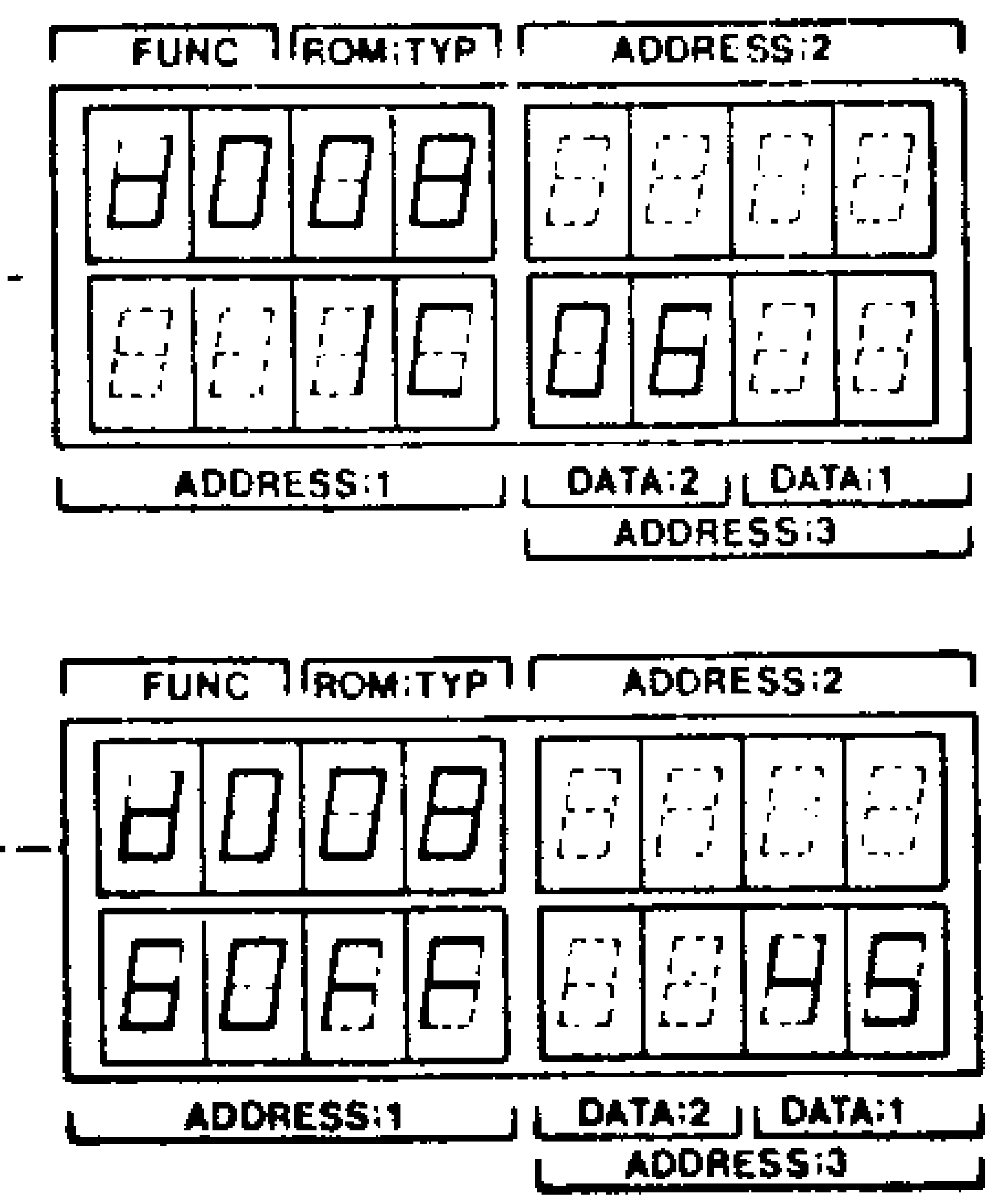


Fig. 372 Examples of the Display

data at the current address. For updating the data or storing new data, follow the above procedures and then input the desired data through the key and press the [SET] key. The address is automatically incremented upon pressing the [SET] key. For aborting a STORE command press the [JOB] command. A STORE command can be used to confirm the data in the PROM on the PROM socket. In this case the address equivalent to the number of bytes of the PROM selected by the PROM switches starts start from the virtual "0" address, instead of "6000" as specified for the system.

(2) FORMAL Command

A FORMAL command is used to block-store or to fill the specified address block with specified data.

≪ Operation ≫

$\boxed{\text{DEB}}\ \boxed{1}\ \boxed{\text{XXXX}}\ \boxed{.}\ \boxed{\text{YYYY}}\boxed{.}\boxed{\text{DD}}\boxed{\text{SET}}$

where XXXX ≤ YYYY is required.

"XXXX" is the first address, and "YYYY" is the last address of the area to be updated, while [DD] is the data to be programmed. They are displayed on "ADDRESS:1", "ADDRESS:2", and "DATA:1" respectively. A FORMAL command, like other commands, can be aborted instantaneously by pressing the [JOB] key, if so required.

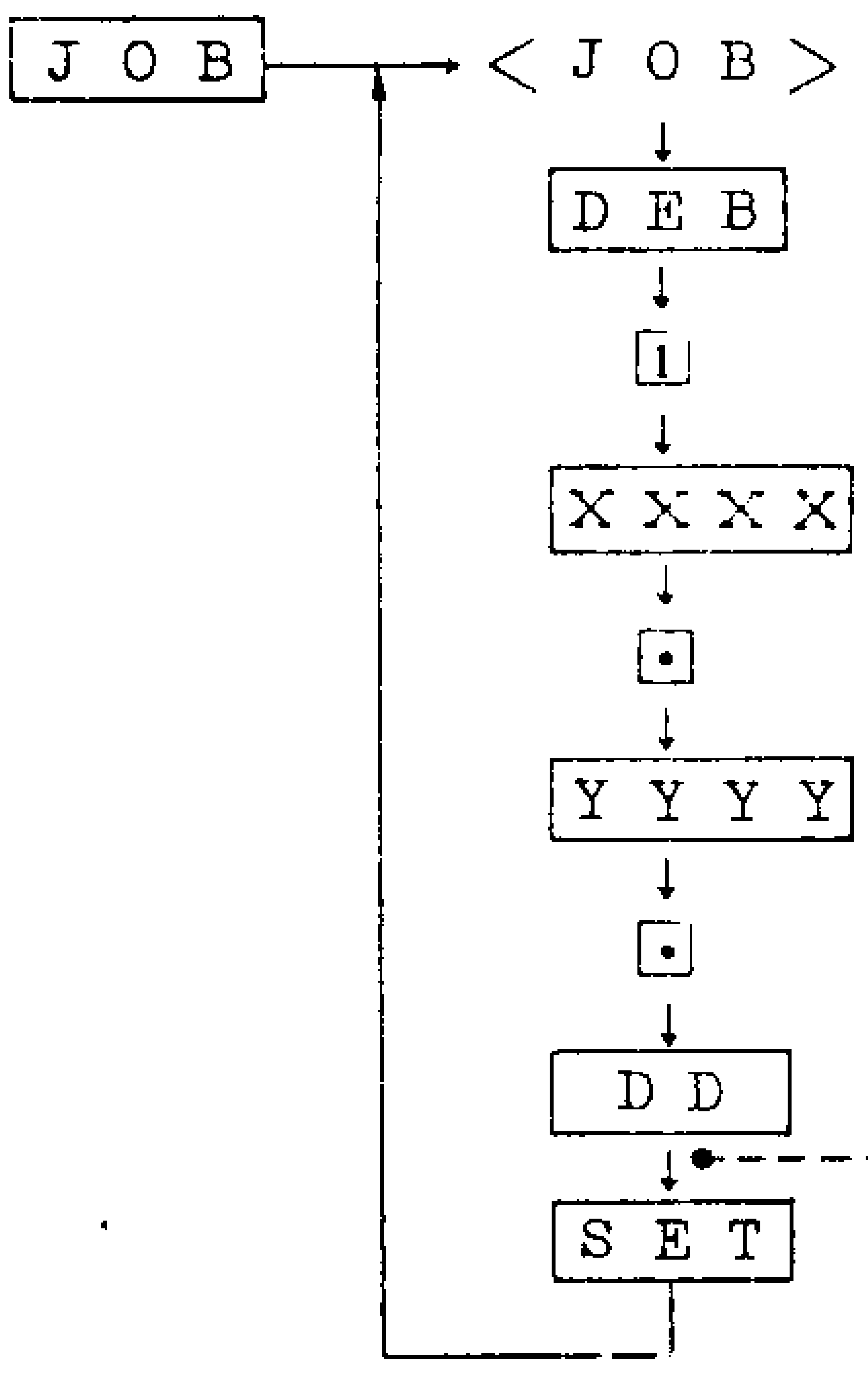


Fig. 373 Flow for a FORMAL Command

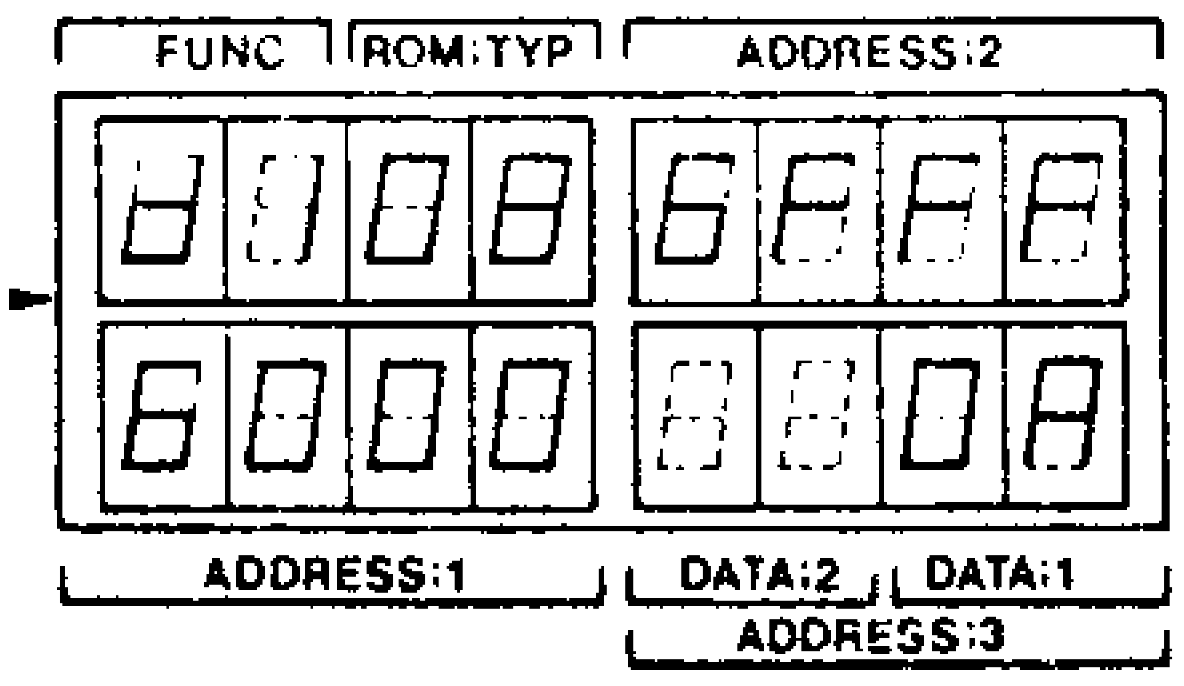


Fig. 374 An Example of the Display

(3) MOVE Command

A MOVE command is used to block-transfer the data on a specified address block to another address block starting with another specified address.

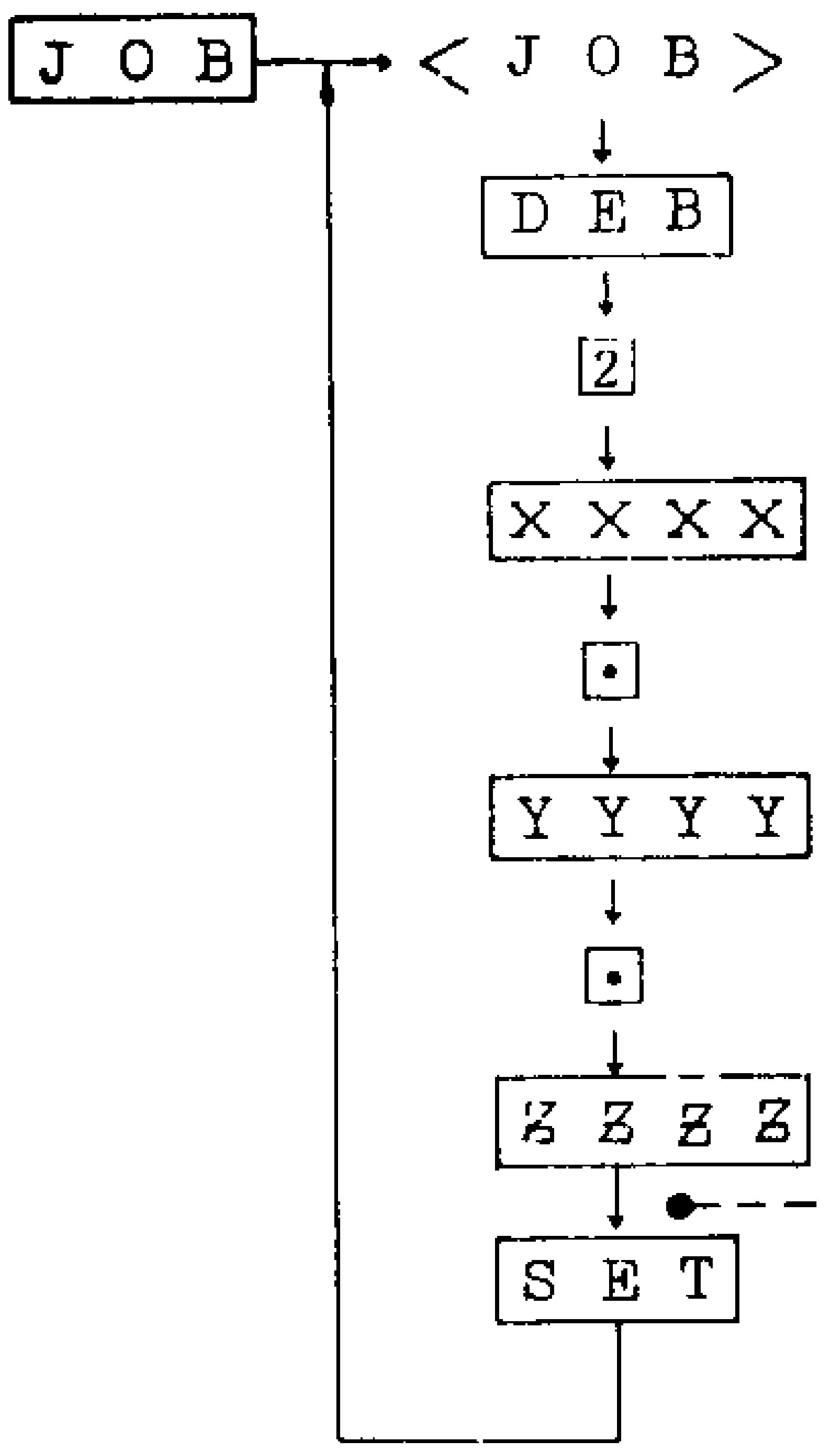


Fig. 375 Flow for a MOVE Command

≪ Operation ≫

$\boxed{\text{DEB}}$ $\boxed{2}$ $\boxed{\text{XXXX}}$ $\boxed{.}$ $\boxed{\text{YYYY}}$ $\boxed{.}$ $\boxed{\text{ZZZZ}}$

$\boxed{\text{SET}}$ where XXXX < YYYY is required.

The operation involves the transfer of the data on the address block "XXXX" through "YYYY" to the address block starting with address "ZZZZ". The three addresses are displayed on "ADDRESS:1", "ADDRESS:2", and "ADDRESS:3" respectively.

The destination address "ZZZZ" can be either in the upper or lower side of, or inside (to overlap) the address block.

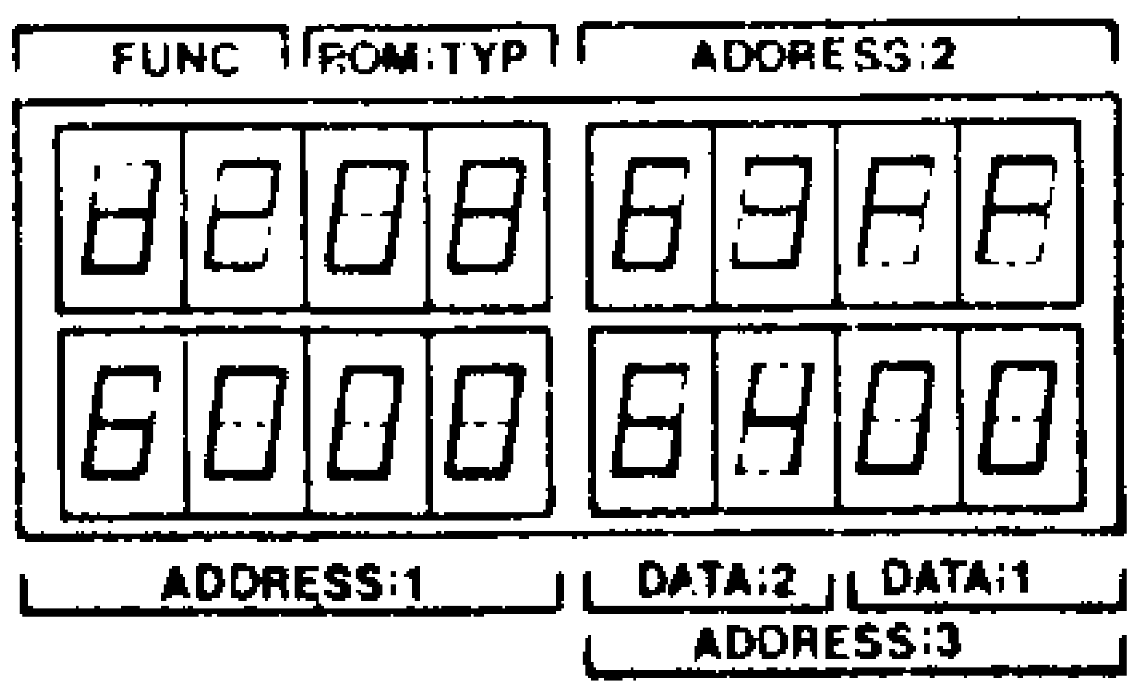


Fig. 376 An Example of the Display

(4) BUFFER REALLOCATE Command

A BUFFER REALLOCATE command is used to change the first address of the RAM (buffer) which is automatically set at "6000" during initialization.

≪ Operation ≫

$\boxed{\text{DEB}}$ $\boxed{4}$ $\boxed{\text{ZZZZ}}$ $\boxed{\text{SET}}$

The reallocated address, "ZZZZ", is displayed on "ADDRESS:3". This command allows you to write the data on any address(es) that may or may not be inside the address block starting with address "6000".

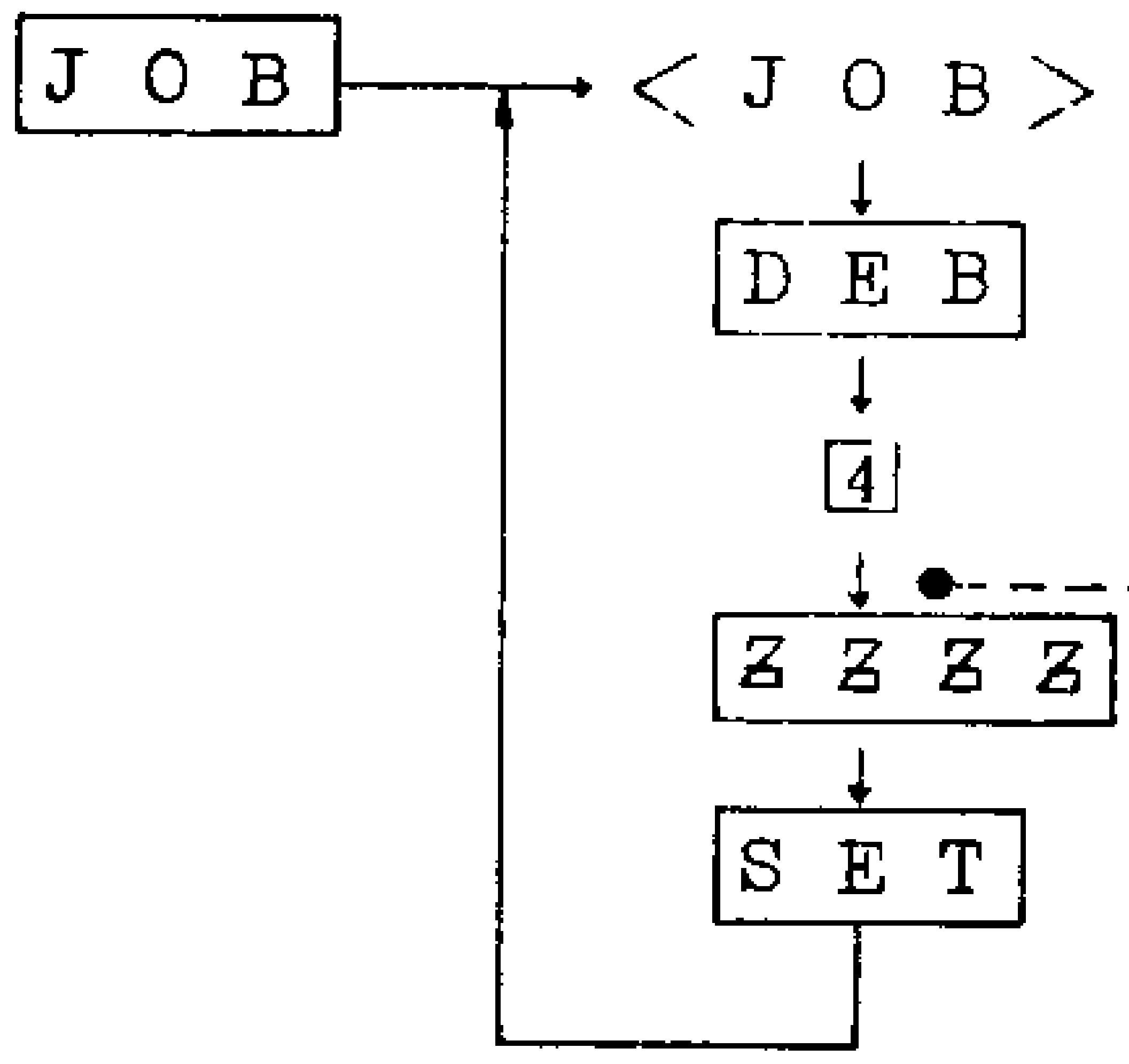


Fig. 374 Flow for a BUFFER REALLOCATE Command

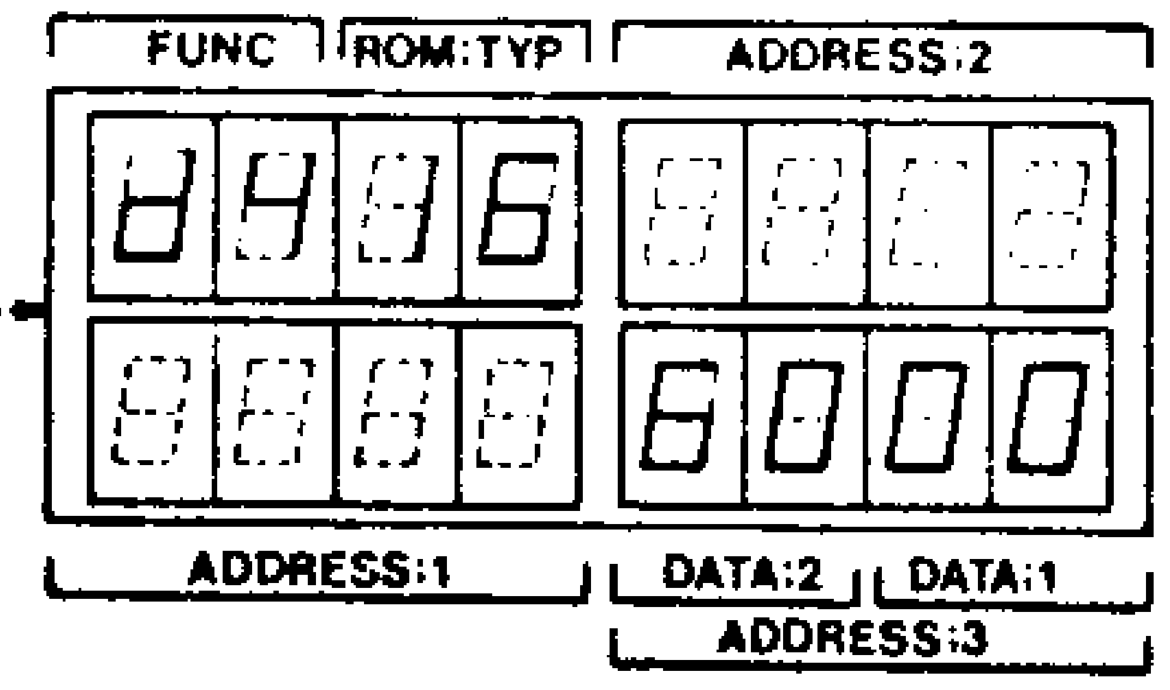


Fig. 375 An Example of the Display

■ 3–9 Examples of Programming Operation

Practical examples and operations required in programming and editing the RAM are given in the following. Please make it sure before you start operating the Pecker that the PROM select SW's are appropriately set and the PECKER in the JOB WAIT state.

A. Copying a Master ROM

A1. $\downarrow$M $\boxed{\text{LOD}}$ $\boxed{\text{SET}}$

A2. $\boxed{\text{CMP}}$ $\boxed{\text{SET}}$ $\uparrow$M

A3. $\downarrow$E $\boxed{\text{ERS}}$ $\boxed{\text{SET}}$

A4. $\boxed{\text{WR}}$ $\boxed{\text{SET}}$ $\uparrow$W

A1 is for inserting the master ROM into the PROM socket and loading the RAM with a LOAD command.
A2 is for comparing for the sake of confirmation.
A3 is for substituting the master ROM with an erased PROM and making an erase check.
A4 is for programming by using a write command.

B. Referring the data on "6000" through "6003" of the RAM (buffer), and updating the data on "6002" and "6003" into "14", and "BC". (It is assumed that the data on "6000" and up are "FF", "A4", "23", and "EC").

B1. $\boxed{\text{DEB}}$ $\boxed{0}$ $\boxed{6}\boxed{0}\boxed{0}\boxed{0}$ $\boxed{\cdot}$
($\boxed{\cdot}$ can be $\boxed{\text{SET}}$)
The display shows [6000] A1 [FF] D1.
B2. $\boxed{+}$ The display shows [6001] A1 [A4] D1.
B3. $\boxed{+}$ The display shows [6002] A1 [23] D1.
B4. $\boxed{1}\boxed{4}$ The display shows [6002] A2 [14] D1.
B5. $\boxed{\text{SET}}$ The display shows [6003] A1 [EC] D1.
B6. $\boxed{\text{B}}\boxed{\text{C}}$ $\boxed{\text{SET}}$ The display shows [6004] A1 [60] D1.
B7. $\boxed{\text{JOB}}$

B1 is for displaying the data at "6000" by using a STORE command. B2 and B3 are for incrementing the address twice and confirming the data each time. B4 and B5 are for rewriting the data at "6002" to "14". B6 is for rewriting the data at "6003" to "BC" [the $\boxed{-}$ key can be used to decrement the address(es) and confirm the data in it(them). B7 is for returning to the JOB WAIT state.

C. Producing the data to be programmed on "6000" and up of the RAM buffer area and programming them into a PROM.

C1. $\boxed{\text{BCL}}$ $\boxed{\text{SET}}$
C2. $\boxed{\text{DEB}}$ $\boxed{0}$ $\boxed{6}\boxed{0}\boxed{0}\boxed{0}$ $\boxed{\cdot}$
C3. $\boxed{5}\boxed{1}$ $\boxed{\text{SET}}$
C4. $\boxed{3}\boxed{\text{E}}$ $\boxed{\text{SET}}$
C5. $\boxed{\text{JOB}}$
C6. Perform A3, and then A4.

C1 is for clearing the buffer ("FF").
C2 is for displaying the data at "6000".
C3 and C4 are for storing the intended data succesively. C5 is for returning to the JOB WAIT state. (Note: It is better to confirm the produced data by using a STORE command). C6 is for programming the data into the PROM per procedures A3 and A4 of Example A.

D. Altering the content of "6000" through "63FF" to "00".

D1. $\boxed{\text{DEB}}$ $\boxed{1}$ $\boxed{6}\boxed{0}\boxed{0}\boxed{0}$ $\boxed{\cdot}$ $\boxed{6}\boxed{3}\boxed{\text{F}}\boxed{\text{F}}$ $\boxed{\cdot}$
$\boxed{0}\boxed{0}$ $\boxed{\text{SET}}$

D1 is for rewriting by using a FORMAL command.

E. Block-transferring the data on "3800" through "4000" on the RAM to the address block starting with address "6000".

E1. $\boxed{\text{DEB}}$ $\boxed{2}$ $\boxed{3}\boxed{8}\boxed{0}\boxed{0}$ $\boxed{\cdot}$ $\boxed{4}\boxed{0}\boxed{0}\boxed{0}$ $\boxed{\cdot}$
$\boxed{6}\boxed{0}\boxed{0}\boxed{0}$ $\boxed{\text{SET}}$

E1 is for block-transferring by using a MOVE command.

F. Transcribing the data in two 2708's onto one 2716.

F1. $\downarrow$M2 $\boxed{\text{LOD}}$ $\boxed{\text{SET}}$ $\boxed{\text{CMP}}$ $\boxed{\text{SET}}$ $\uparrow$M2
F2. $\boxed{\text{DEB}}$ $\boxed{2}$ $\boxed{6}\boxed{0}\boxed{0}\boxed{0}$ $\boxed{\cdot}$ $\boxed{6}\boxed{3}\boxed{\text{F}}\boxed{\text{F}}$ $\boxed{\cdot}$
$\boxed{6}\boxed{4}\boxed{0}\boxed{0}$ $\boxed{\text{SET}}$
F3. $\downarrow$M1 $\boxed{\text{LOD}}$ $\boxed{\text{SET}}$ $\boxed{\text{CMP}}$ $\boxed{\text{SET}}$ $\uparrow$M1
F4. Set the PROM SW's for the 2716.
F5. $\downarrow$E $\boxed{\text{ERS}}$ $\boxed{\text{SET}}$ $\boxed{\text{WR}}$ $\boxed{\text{SET}}$ $\uparrow$W

F1 is for loading the content of the 2708 for the upper half addresses onto the RAM buffer. F2 is for transferring the data to "6400" and beyond by using a MOVE command. F3 is for loading the content of

2708 for the lower half addresses onto the RAM buffer. F4 is for setting the PROM select SW's for the 2716. F5 is for erase-checking the 2716, and for programming if it is erased.

G. Copying the data in a 2716 onto two 2708's

G1. ↓M $\boxed{\text{LOD}}$ $\boxed{\text{SET}}$ $\boxed{\text{CMP}}$ $\boxed{\text{SET}}$ ↑M
G2. Set the PROM SW's for the 2708
G3. ↓E1 $\boxed{\text{ERS}}$ $\boxed{\text{SET}}$ $\boxed{\text{WR}}$ $\boxed{\text{SET}}$ ↑W1
G4. $\boxed{\text{DEB}}$ $\boxed{4}$ $\boxed{6}\boxed{4}\boxed{0}\boxed{0}$ $\boxed{\text{SET}}$
G5. ↓E2 $\boxed{\text{ERS}}$ $\boxed{\text{SET}}$ $\boxed{\text{WR}}$ $\boxed{\text{SET}}$ ↑W2
G6. $\boxed{\text{DEB}}$ $\boxed{4}$ $\boxed{6}\boxed{0}\boxed{0}\boxed{0}$ $\boxed{\text{SET}}$

Chargement depuis une adresse de départ autre que 6000

G1 is for loading the data in the 2716 onto the RAM buffer. G2 is for setting the PROM SW's for the 2708. G3 is for mounting the 2708 for the lower half addresses and for programming after erase-checking. G4 is for allocating "6400" and beyond for the RAM buffer. G5 is for programming the 2708 for the upper half addresses. G6 is for returning the buffer address to 6,000.

→ Remplir une zone mémoire par la même valeur

→ Déplacement d'une zone RAM-PROM

[4. High Level Use of the PECKER]

The PECKER has a number of auxiliary functions in addition to the PROM programming functions described so far, and can be used for executing and debugging Z80 and 8080 programs.

■ 4–1 SPEAKER Command

This command disables the microspeaker that gives the audible signals (tones) for acknowledging the key input operations and other purposes as well. DEB 5 issues this command. If the keyed-in data which follows this is anything but "0" (zero), the microspeaker is turned on. (The initial setting automatically sets it for a "0" (zero)).

≪ Operation ≫

Disabling (OFF)

DEB 5 1 SET

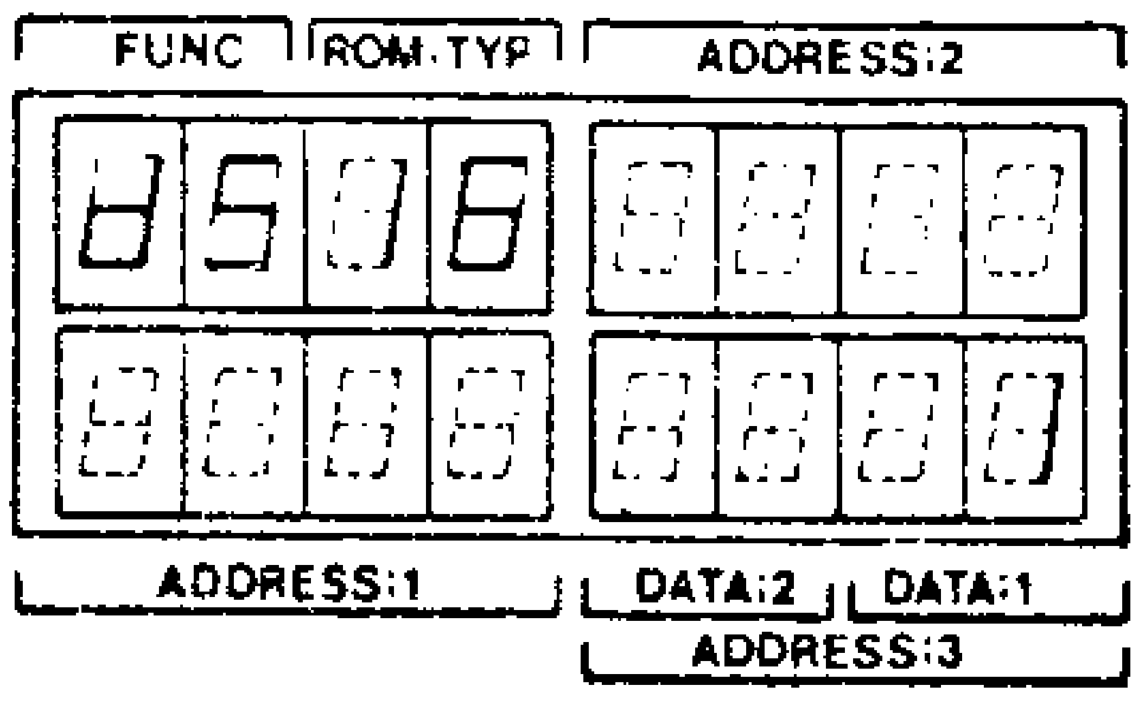


Fig. 411

≪ Operation ≫

Enabling (ON)

DEB 5 0 SET

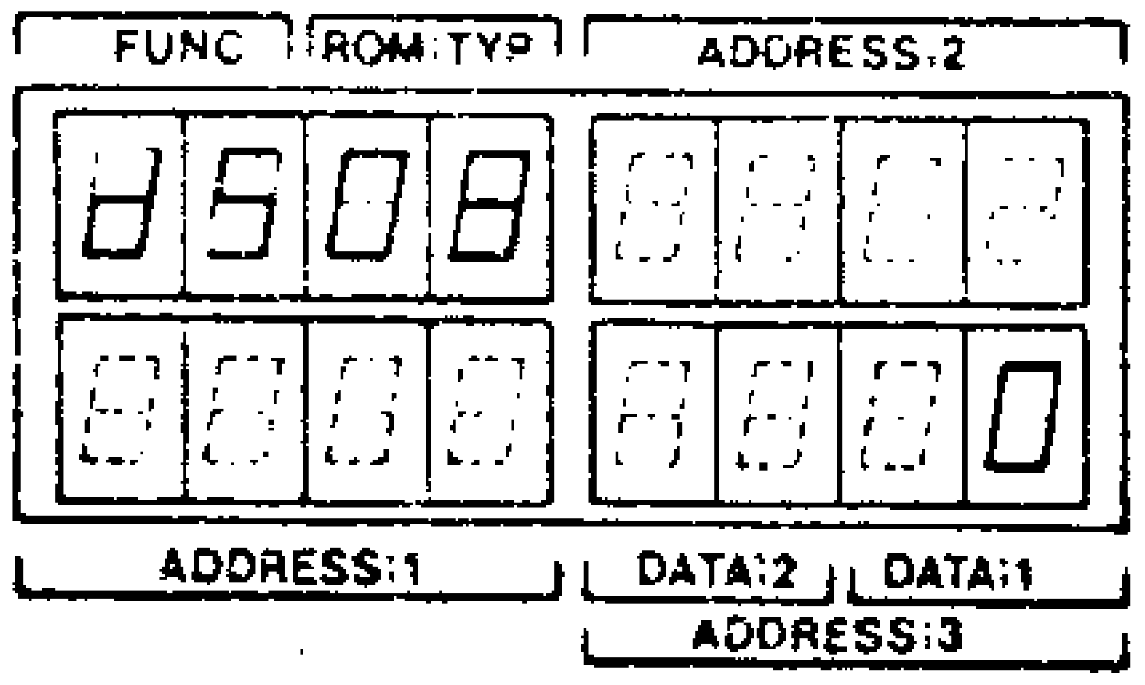


Fig. 412

■ 4–2 REGISTER Commands

The PECKER uses the Z–80 for its CPU, and these commands can be used for referring each register of the Z–80 for various purposes including program debugging.

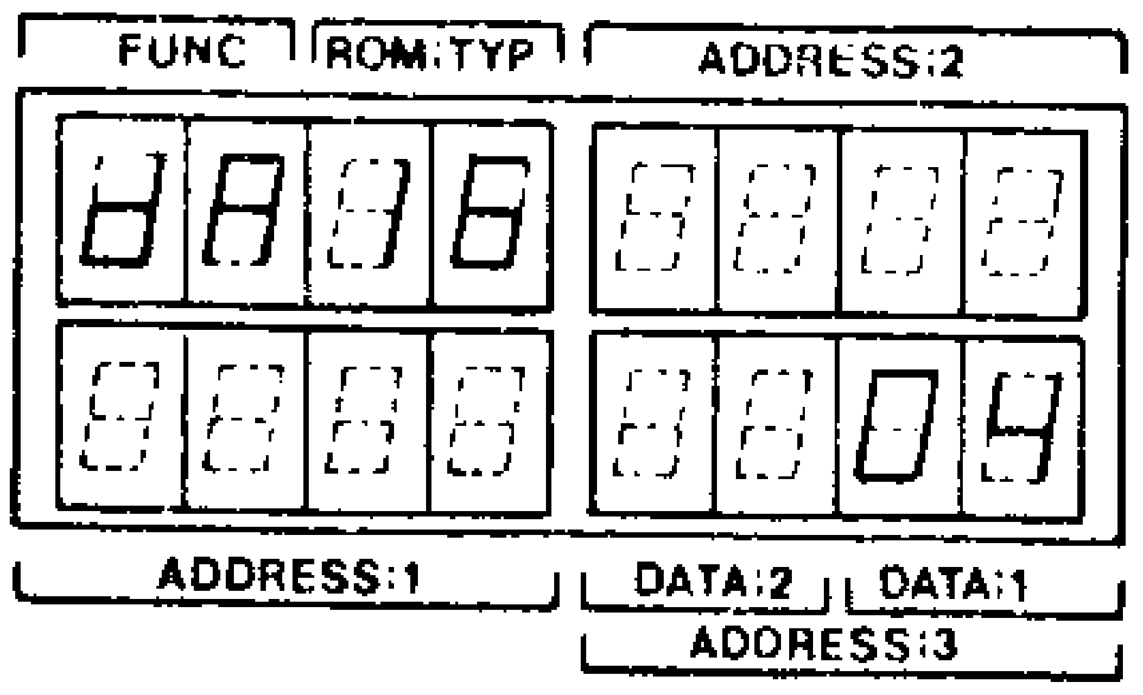
These commands are specified by a combination of the DEB key and other keys as shown in Table 421.

Table 421

Key		Key	
A	Accummulator	XIH	H Register
B	B Register	÷IL	L Register
C	C Register	+IX	IX Register
D	D Register	–IY	IY Register
E	E Register	· IS	Stack Pointer
F	F Register	=IP	Program Counter

≪ Operation ≫

DEB A followed by DD SET (for updating).



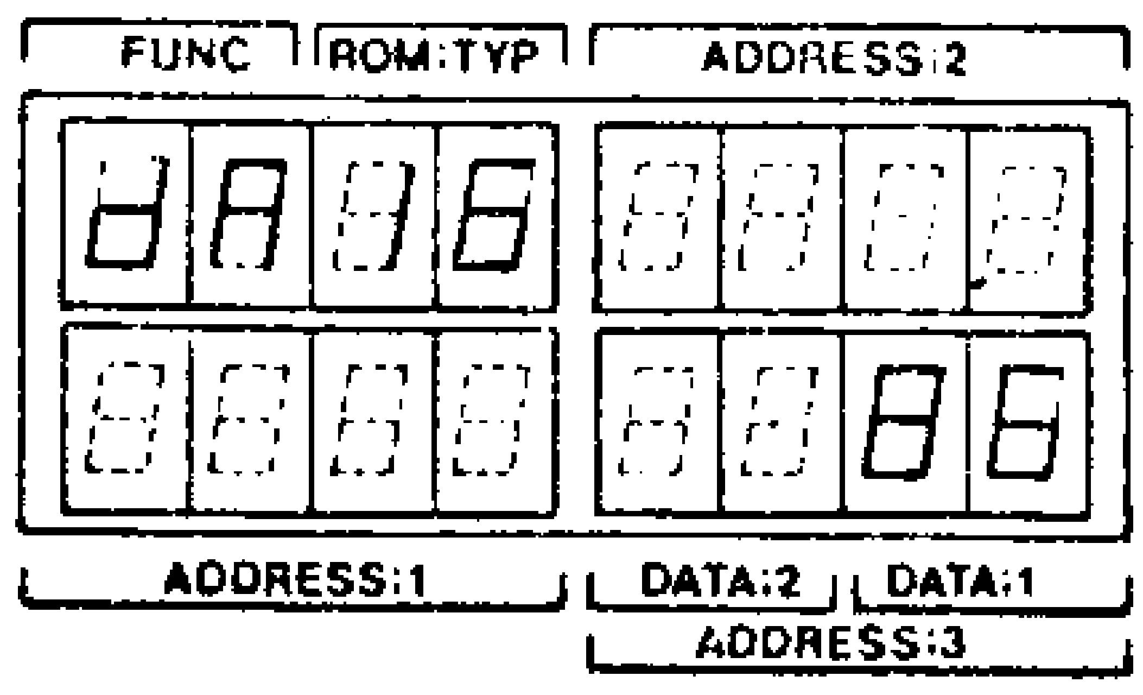


Fig. 422 Updating "A" Register

■ 4–3 EXECUTE Command

If it is desired to actually execute the program on the PECKER, this command is used to execute it starting from the address specified. $\boxed{\text{GO}}$ command and $\boxed{\text{DEB}}$ $\boxed{3}$ are the two commands of the same function as to the execution. The latter allows to specify two break points.

(1) $\boxed{\text{GO}}$ Command

≪Operation≫

$\boxed{\text{GO}}$ $\boxed{\text{XXXX}}$ $\boxed{\text{SET}}$

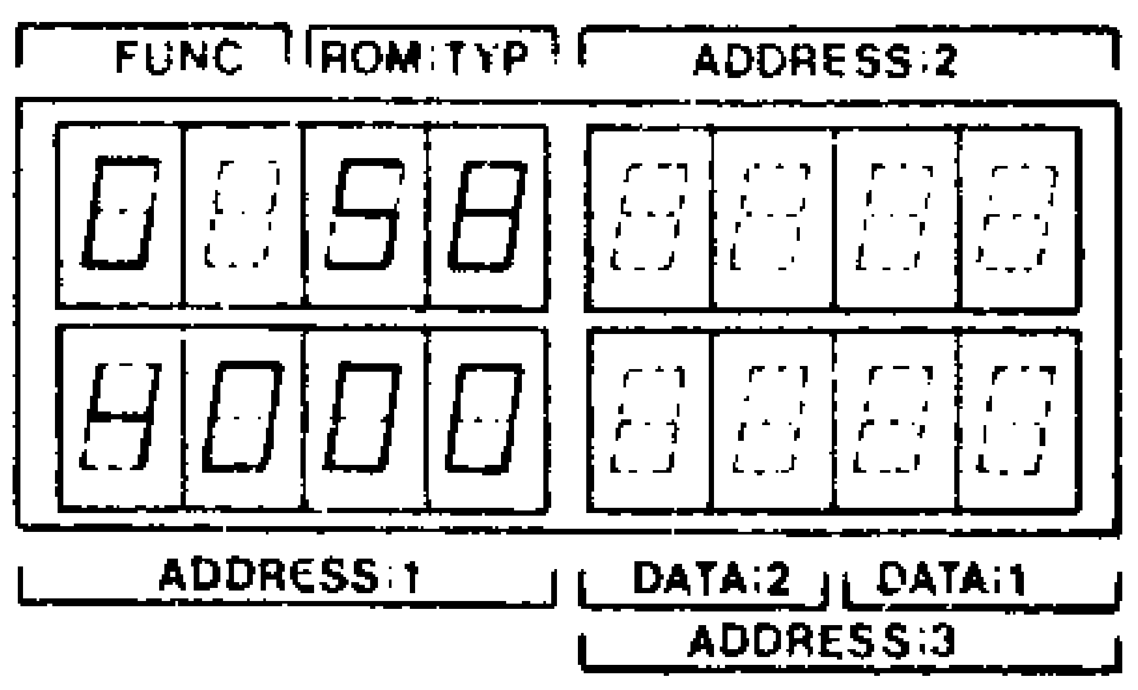


Fig. 431 GO Command

The program is executed starting from the address XXXX.

(2) Command given by $\boxed{\text{DEB}}$ $\boxed{3}$

The feature of the execution by $\boxed{\text{DEB}}$ $\boxed{3}$ is that up to two break points can be specified. The CPU returns to the JOB WAIT state, keeping all the data of the register, flag, SP, PC, etc. in the CPU, when it has passed the specified break point address. So you can either check or alter the CPU status (registers, flags, SP, PC, RAM etc.) preceding the break point, and then execute the program again. (Note: The break point is implemented by inserting a restart instruction (RST6) at the specified address, and restoring the original data when transferring to the JOB WAIT state.)

≪Operation≫

For one break point:

$\boxed{\text{DEB}}$ $\boxed{3}$ $\boxed{\text{XXXX}}$ $\boxed{\cdot}$ $\boxed{\text{YYYY}}$ $\boxed{\text{SET}}$

For two break points:

$\boxed{\text{DEB}}$ $\boxed{3}$ $\boxed{\text{XXXX}}$ $\boxed{\cdot}$ $\boxed{\text{YYYY}}$ $\boxed{\cdot}$ $\boxed{\text{ZZZZ}}$ $\boxed{\text{SET}}$

The "XXXX" is the address where the execution is started, while the "YYYY" and "ZZZZ" are the break point addresses.

For resuming the execution from the JOB WAIT state, you need not input the address for the execution. The execution resumes at the break point that the CPU previously passed.

≪Operation≫

Resuming the execution after a break:

$\boxed{\text{DEB}}$ $\boxed{3}$ $\boxed{\text{SET}}$

This operation automatically displays the program counter on the "ADDRESS:1", and the execution resumes at that address.

≪Operation≫

Resuming the execution and specifying a new break point:

$\boxed{\text{DEB}}$ $\boxed{3}$ $\boxed{\cdot}$ $\boxed{\text{YYYY}}$ $\boxed{\text{SET}}$

$\boxed{\text{DEB}}$ $\boxed{3}$ $\boxed{\cdot}$ $\boxed{\text{YYYY}}$ $\boxed{\cdot}$ $\boxed{\text{ZZZZ}}$ $\boxed{\text{SET}}$

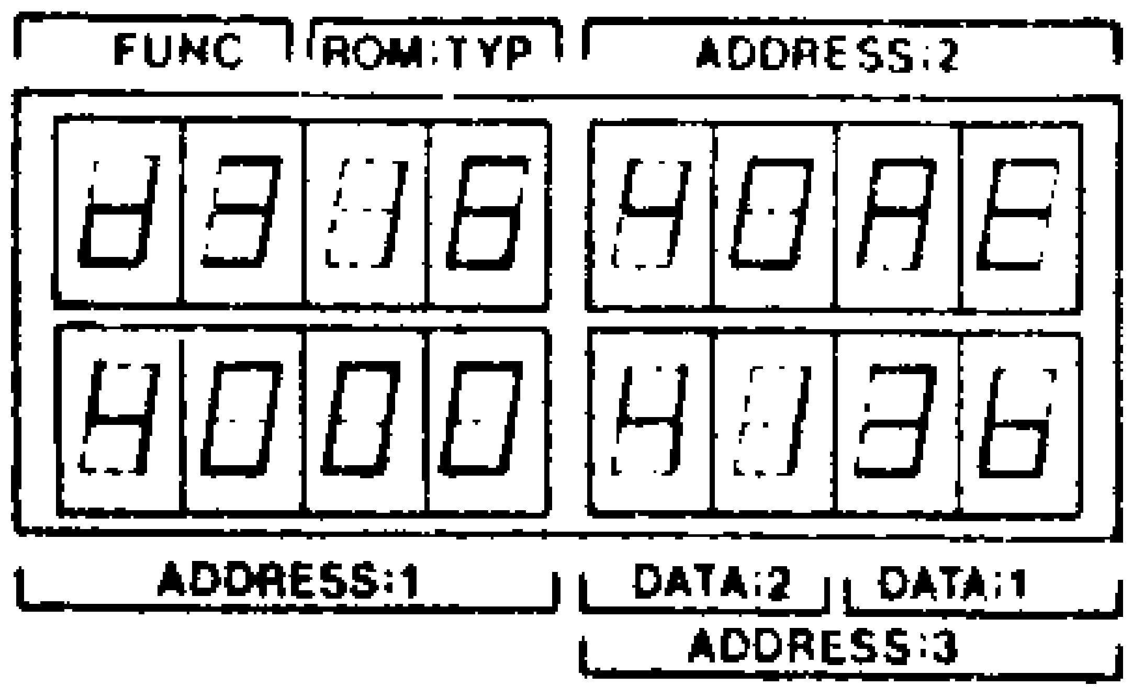


Fig. 432 **Execution starts at 4000**
Break Point 1 – 40AE
Break Point 2 – 413B

(Note)

After RESET, the first and only the first execution automatically starts at "6000", which is the starting address of the programming buffer, by using the $\boxed{\text{DEB}}$ $\boxed{3}$ $\boxed{\text{SET}}$ combination. A break point must be specified in the RAM area.

You take the advantage of the use of "RST6" in the system for a break point to obtain a virtual break point by inserting a "RST6 (F7)" in the program under execution. The program counter is set at the address next to where the RST6 is inserted, so that the execution can be resumed by using the $\boxed{\text{DEB}}$ $\boxed{3}$ $\boxed{\text{SET}}$ combination in this case also.

■ 4–4 NMI (Nonmaskable Interrupt)

■ 4–5 Program Routines Open to Users

(1) Key-in

CALL OCH: CD OC OO

The data fed through the keys are set onto the accummulator by calling the address OC. The data fed through the keys are encoded as in Table 453.

Note: $\boxed{\text{SIFT}}$ key sets the bit 7 to bit 1.

Note: Conditions

Table 451

KEY	ZERO FLAG	CARRY FLAG
JOB	SET	SET
SET	SET	RESET
OTHERS	RESET	RESET

(2) LED Display

LEDs are used as dynamic display, and the data are displayed by storing them on the appropriate RAM. The data to be stored for display are shown in Table 453. "10" gives a blank, and a DP (Display Point) is displayed by setting the bit 7 to 1. The way the display image is generated is shown in Table 261. The LED units and the RAM addresses are related as follows.

0 ~ 7 :	3280 ~ 3287
8 ~ F :	3288 ~ 328F

(Note: You cannot use addresses, "3282" and "3283", because they are always used for displaying the setting of the PROM SW's.)

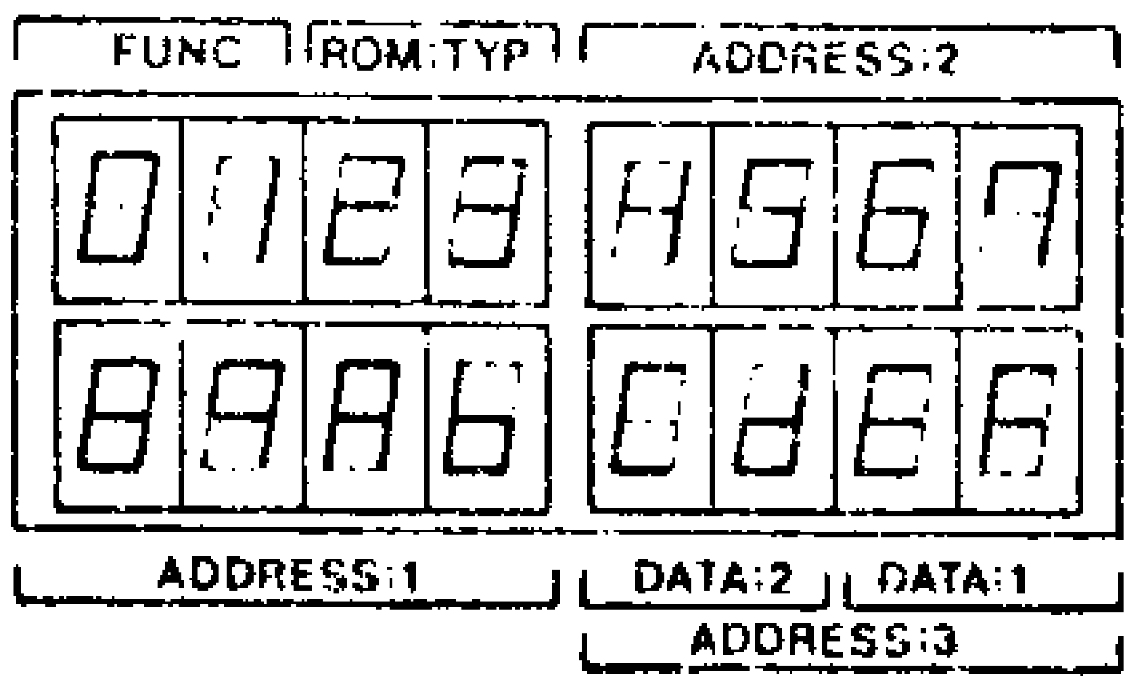


Fig. 452 The LED Unit Number

(3) Excitation of the Microspeaker

The microspeaker is driven as follows.

(3–1) Excitation

```
MVI    C, XX    0E XX
CALL   1CH      CD 1C00
```

The data set on register "C" specifies the duration of the excitation.

Duration = "C Register" x 5ms

The registers used are the accummulator and registers "B" and "C", and both of these registers will be cleared upon completion.

(3–2) Repeating the Excutation and Disabling

```
LXI    B   XXXX    01 XXXX
CALL   14H         CD1400
```

The data to be set are the number of loops on register "B" and the duration of the excitation on register "C" ("C register" x 5 ms). Likewise, the OFF duration is set on "C" register also. The registers used are accumulator and registers, "B" and "C". Register "B" is cleared while the register "C" remains unchanged upon return.

The operation for giving audible tones is to repeat the OFF–ON operation.

KEY	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	4	5	6	7	8	9	A	B	C	D	E	F
CODE	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F

KEY	×|H	÷|L	+|X	−|Y	•|S	=|P	SET	OUT	WR	CMP	LOD	ERS	BCL	GO	JOB	DEB
CODE	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20

Table 453 Key Input Encoding for the Display