To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

RENESAS

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

APPLICATION NOTE

RENESAS

17K SERIES

4-BIT SINGLE-CHIP MICROCONTROLLER

FLOATING-POINT ARITHMETIC PACKAGE

Document No. IEA−1306
(O. D. No. IEA−757)
Date Published November 1994 P
Printed in Japan

The application circuits and their parameters are for references only and are not intended for use in actual design-in's

M4 92.6

Contents

List of Figures

List of Tables

Preface

Users:

This manual is for engineers who intend to learn the capabilities of the 17K Series for application program development.

Purpose:

The purpose of this manual is to help users understand the capabilities of the 17K Series by means of sample application programs.

Organization:

This manual includes the following chapters:

- General
- Hardware Configuration
- Pocket Calculator Specifications
- Overview of System Control Section
- RAM Layout and Variables
- Flowcharts of System Control Section
- Floating-Point Format
- Explanation of Arithmetic Package
- Pocket Calculator Programs

Notation:

Data weight: Higher digits on the left side and lower digits on the right side

Note: Explanation of an indicated part of text

Caution: Information requesting the user's special attention

Remark: Supplementary information

Numeric: Decimal: xxxx

Hexadecimal: xxxxH

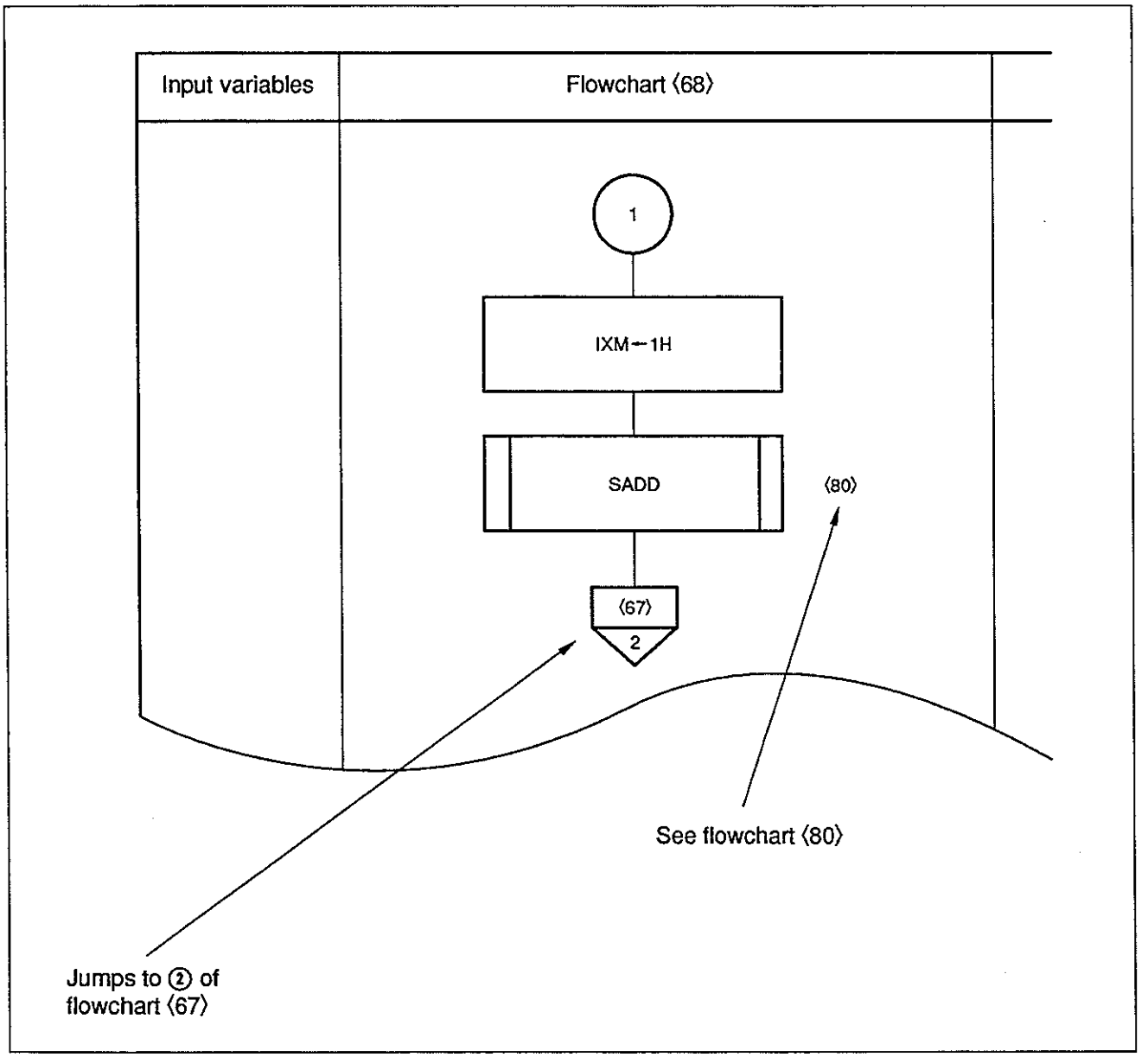
Related publications:

The related publications are listed below.

- μPD17201A, μPD17207 Data Sheet (IC-2773)
- μPD172xx Series User's Manual (IEU-1317)

Reading flowcharts:

An example of flowchart is provided below to illustrate how to read the flowcharts used in this manual.



Chapter 1
General

In the field of controllers, many applications use microcomputers to convert external analog input signals to digital signals, and use values produced by various operations.

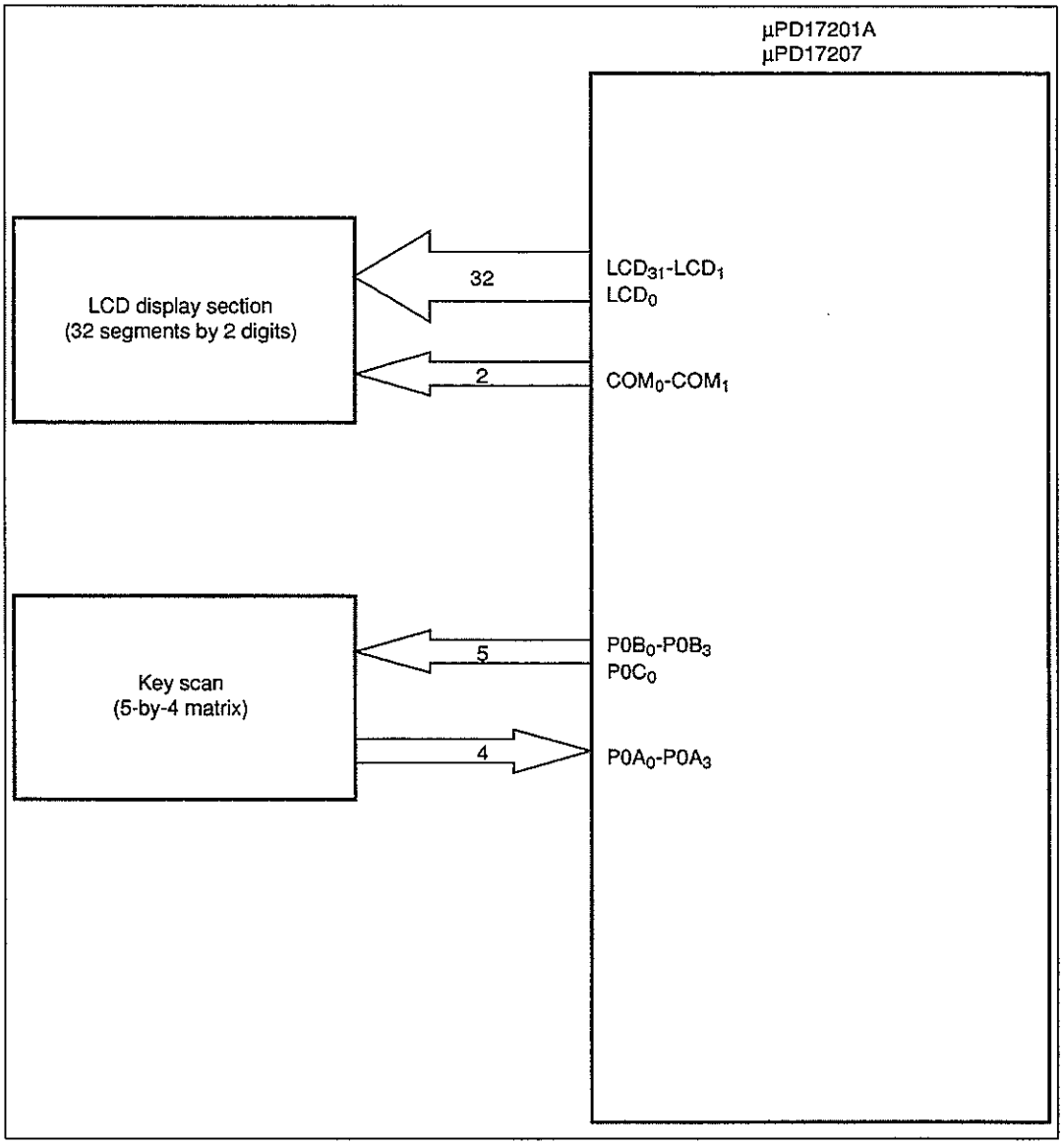
This manual introduces a floating-point arithmetic package and sample programs for controlling a pocket calculator, which are newly developed using the μPD17201A and μPD17207. The μPD17201A and μPD17207 are 17K Series 4-bit single-chip microcontrollers and are designed for infrared remote control.

Chapter 2
Hardware Configuration

2.1 System Block Diagram

Figure 2-1 shows the system block diagram of the pocket calculator introduced in this manual.

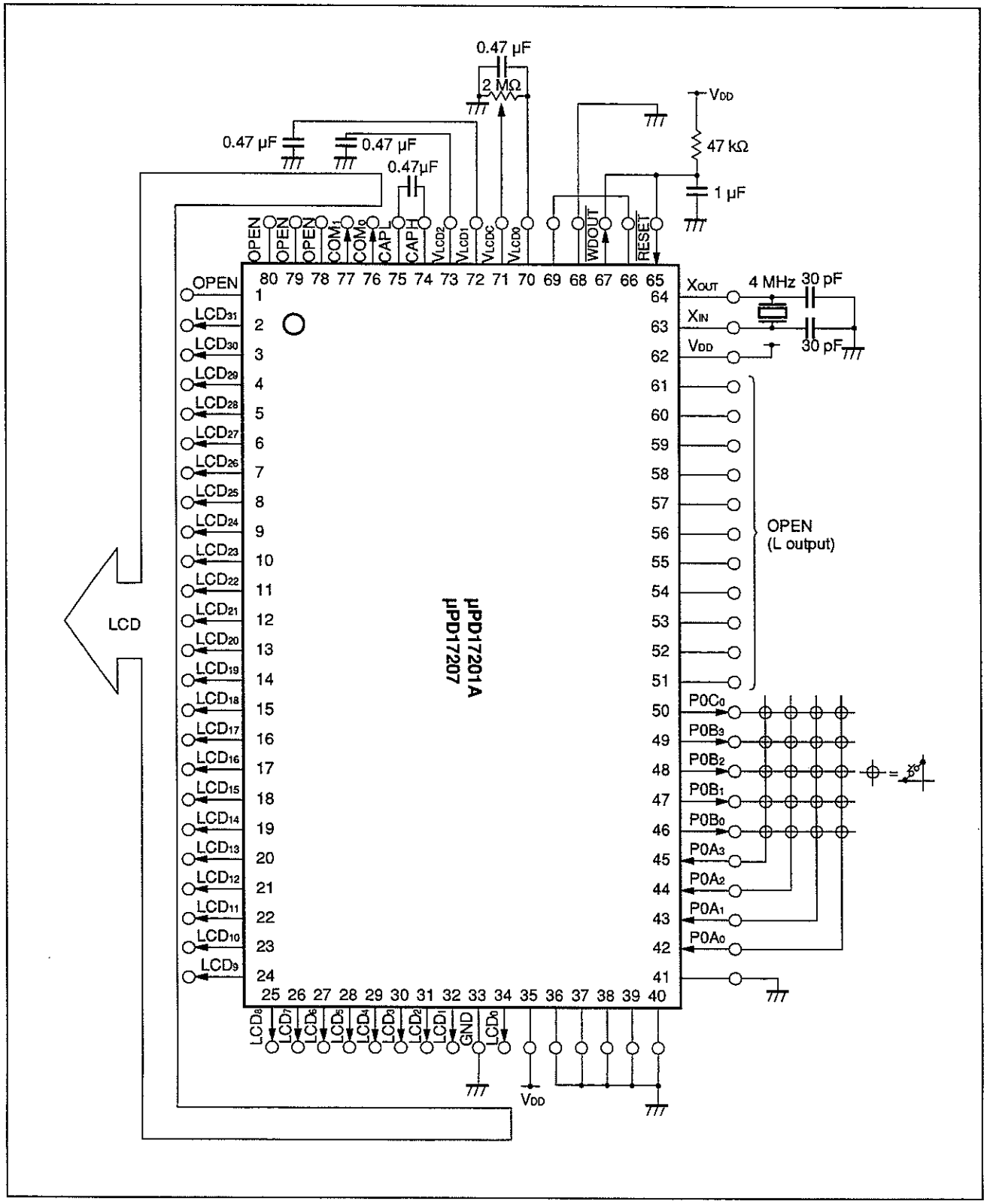
Figure 2-1. System Block Diagram



2.2 Circuit Diagram

Figure 2-2 shows the circuit diagram of the pocket calculator.

Figure 2-2. Circuit Diagram



2.3 Pin Assignment

Table 2-1. Pin Assignment (1/3)

Pin No.	Pin name	I/O	Logic	Initialization	Function	Mask option
1	LCD_{32}	Output	—	—	Not used (open)	
2	LCD_{31}	Output	—	—	Segment signal output from LCD controller/driver	
3	LCD_{30}	Output	—	—	Segment signal output from LCD controller/driver	
4	LCD_{29}	Output	—	—	Segment signal output from LCD controller/driver	
5	LCD_{28}	Output	—	—	Segment signal output from LCD controller/driver	
6	LCD_{27}	Output	—	—	Segment signal output from LCD controller/driver	
7	LCD_{26}	Output	—	—	Segment signal output from LCD controller/driver	
8	LCD_{25}	Output	—	—	Segment signal output from LCD controller/driver	
9	LCD_{24}	Output	—	—	Segment signal output from LCD controller/driver	
10	LCD_{23}	Output	—	—	Segment signal output from LCD controller/driver	
11	LCD_{22}	Output	—	—	Segment signal output from LCD controller/driver	
12	LCD_{21}	Output	—	—	Segment signal output from LCD controller/driver	
13	LCD_{20}	Output	—	—	Segment signal output from LCD controller/driver	
14	LCD_{19}	Output	—	—	Segment signal output from LCD controller/driver	
15	LCD_{18}	Output	—	—	Segment signal output from LCD controller/driver	
16	LCD_{17}	Output	—	—	Segment signal output from LCD controller/driver	
17	LCD_{16}	Output	—	—	Segment signal output from LCD controller/driver	
18	LCD_{15}	Output	—	—	Segment signal output from LCD controller/driver	
19	LCD_{14}	Output	—	—	Segment signal output from LCD controller/driver	
20	LCD_{13}	Output	—	—	Segment signal output from LCD controller/driver	
21	LCD_{12}	Output	—	—	Segment signal output from LCD controller/driver	
22	LCD_{11}	Output	—	—	Segment signal output from LCD controller/driver	
23	LCD_{10}	Output	—	—	Segment signal output from LCD controller/driver	
24	LCD_{9}	Output	—	—	Segment signal output from LCD controller/driver	
25	LCD_{8}	Output	—	—	Segment signal output from LCD controller/driver	
26	LCD_{7}	Output	—	—	Segment signal output from LCD controller/driver	
27	LCD_{6}	Output	—	—	Segment signal output from LCD controller/driver	
28	LCD_{5}	Output	—	—	Segment signal output from LCD controller/driver	
29	LCD_{4}	Output	—	—	Segment signal output from LCD controller/driver	
30	LCD_{3}	Output	—	—	Segment signal output from LCD controller/driver	
31	LCD_{2}	Output	—	—	Segment signal output from LCD controller/driver	
32	LCD_{1}	Output	—	—	Segment signal output from LCD controller/driver	
33	GND	—	—	—	Ground	

Table 2-1. Pin Assignment (2/3)

Pin No.	Pin name	I/O	Logic	Initialization	Function	Mask option
34	LCD_0	Output	—	—	Segment signal output from LCD controller/driver	
35	V_{ADC}	—	—	—	Not used (connected to V_{DD})	
36	ADC_0	Input	—	—	Not used (connected to GND)	
37	ADC_1	Input	—	—	Not used (connected to GND)	
38	ADC_2	Input	—	—	Not used (connected to GND)	
39	ADC_3	Input	—	—	Not used (connected to GND)	
40	GND_{ADC}	—	—	—	Not used (connected to GND)	
41	INT	Input	—	—	Not used (connected to GND)	
42	$P0A_0$	Input	—	—	Key input	
43	$P0A_1$	Input	—	—	Key input	
44	$P0A_2$	Input	—	—	Key input	
45	$P0A_3$	Input	—	—	Key input	
46	$P0B_0$	Output	L	H	Key scan output	
47	$P0B_1$	Output	L	H	Key scan output	
48	$P0B_2$	Output	L	H	Key scan output	
49	$P0B_3$	Output	L	H	Key scan output	
50	$P0C_0$	Output	L	H	Key scan output	
51	$P0C_1$	Output	—	L	Not used (open)	
52	$P0C_2$	Output	—	L	Not used (open)	
53	$P0C_3$	Output	—	L	Not used (open)	
54	$P0D_0$/LED	Output	—	L	Not used (open)	
55	$P0D_1$/TMOUT	Output	—	L	Not used (open)	
56	$P0D_2$	Output	—	L	Not used (open)	
57	$P0D_3$	Output	—	L	Not used (open)	
58	$P1A_2/\overline{SCK}$	Output	—	L	Not used (open)	
59	$P1A_1$/SO	Output	—	L	Not used (open)	
60	$P1A_2$/SI	Output	—	L	Not used (open)	
61	REM	Output	—	L	Not used (open)	
62	V_{DD}	—	—	—	Main power supply (+5 V)	
63	X_{IN}	—	—	—	Main clock connection	
64	X_{OUT}	—	—	—	Main clock connection	
65	$\overline{RESET}$	Input	L	—	Reset input	Pull-up
66	V_{REG}	—	—	—	Not used (connected to XT_{OUT})	
67	$\overline{WDOUT}$	Output	L	—	Output for crash detection	
68	XT_{IN}	—	—	—	Not used (connected to GND)	

Table 2-1. Pin Assignment (3/3)

Pin No.	Pin name	I/O	Logic	Initialization	Function	Mask option
69	XT_{OUT}	—	—	—	Not used (connected to V_{REG})	
70	V_{LCD0}	Output	—	—	Output for LCD drive reference voltage	
71	V_{LCDC}	Output	—	—	Output for LCD drive reference voltage adjustment	
72	V_{LCD1}	Output	—	—	Doubler output for LCD drive	
73	V_{LCD2}	Output	—	—	Tripler output for LCD drive	
74	CAPH	—	—	—	Capacitor connection for increasing LCD drive voltage	
75	CAPL	—	—	—	Capacitor connection for increasing LCD drive voltage	
76	COM_0	Output	—	—	Common signal output from LCD controller/driver	
77	COM_1	Output	—	—	Common signal output from LCD controller/driver	
78	LCD_{35}/COM_2	Output	—	—	Not used (open)	
79	LCD_{34}/COM_3	Output	—	—	Not used (open)	
80	LCD_{33}	Output	—	—	Not used (open)	

2.4 Pin Functions

Pin No.	Pin name	Function
2 | 32 34	LCD_{31} | LCD_1 LCD_0	These pins are used to output the segment signals from the LCD controller/driver. The signals output on these pins and COM_0-COM_1 form a matrix (32 x 2).
33	GND	Ground pin
42 | 45	$P0A_0$ | $P0A_3$	These pins are used for key input. The signals applied to these pins, and $P0B_0$-$P0B_3$ and $P0C_0$ form a matrix (5 x 4).
46 | 49 50	$P0B_0$ | $P0B_3$ $P0C_0$	These pins are used for key scan output. The signals output on these pins and $P0A_0$-$P0A_3$ form a matrix (5 x 4).
62	V_{DD}	Main power supply pin
63 64	X_{IN} X_{OUT}	These pins are used to connect a 4 MHz ceramic resonator for main clock generation.
65	$\overline{RESET}$	This pin is used for system reset input. When the low level is applied to this pin, the system is reset. During low level input, main clock generation stops.
67	$\overline{WDOUT}$	This output pin is used for crash detection. The low level is output when a watchdog timer overflow or stack overflow/underflow occurs. Connect this pin to the $\overline{RESET}$ pin.
71	V_{LCDC}	This output pin is used to adjust LCD drive reference voltage adjustment.
70 71 72	V_{LCD0} V_{LCD1} V_{LCD2}	These pins are used for LCD drive reference voltage output. • V_{LCD0} : Reference voltage output • V_{LCD1} : Doubler output (voltage two times higher) • V_{LCD2} : Tripler output (voltage three times higher)
74 75	CAPH CAPL	This pin is used to connect a capacitor for increasing LCD drive voltage.
76 77	COM_0 COM_1	These pins are used to output the common signals from the LCD controller/driver. The signals output on these pins and LCD_{31}-LCD_0 form a matrix (32 x 2).

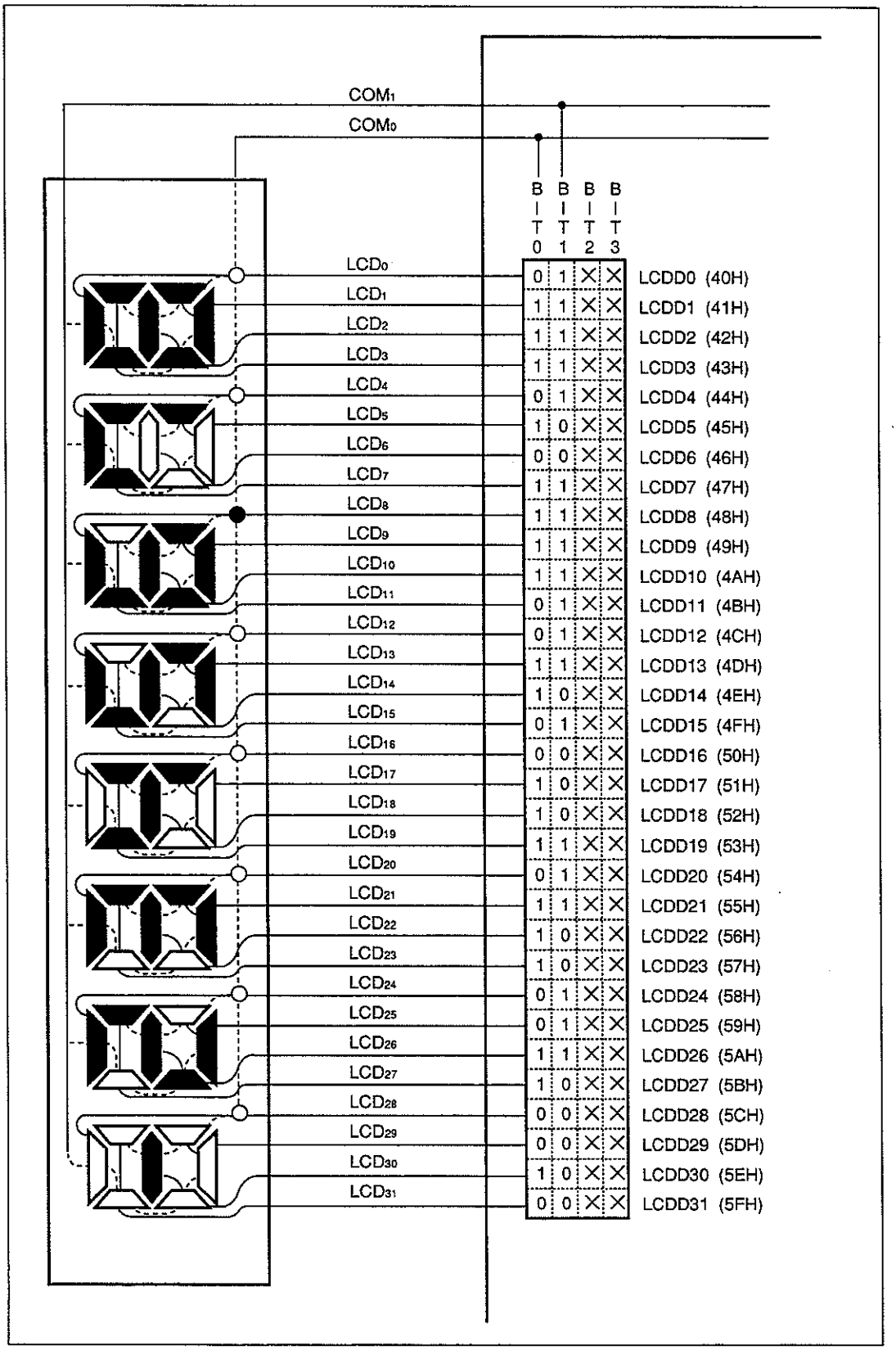
2.5 Display

For display of the pocket calculator introduced in this manual, LCD display based on a 2-by-32 matrix is used.

For display control, the LCD controller/driver contained in the μPD17201A and μPD17207 is used. Based on direct memory access (DMA) operation, the LCD controller/driver automatically reads segment data from display data memory to generate segment signals and common signals. The display data memory is mapped to LCDD0-LCDD35 (addresses 40H-63H of BANK0); the pocket calculator uses LCDD0-LCDD31 (addresses 40H-5FH of BANK0).

Figure 2-3 shows an example of LCD panel connections of the pocket calculator.

Figure 2-3. Example of LCD Panel Connections



Remark For the pocket calculator introduced in this manual, time-division display with two channels is used, so that bits 2 and 3 (marked with X) of the display data memory are not used.

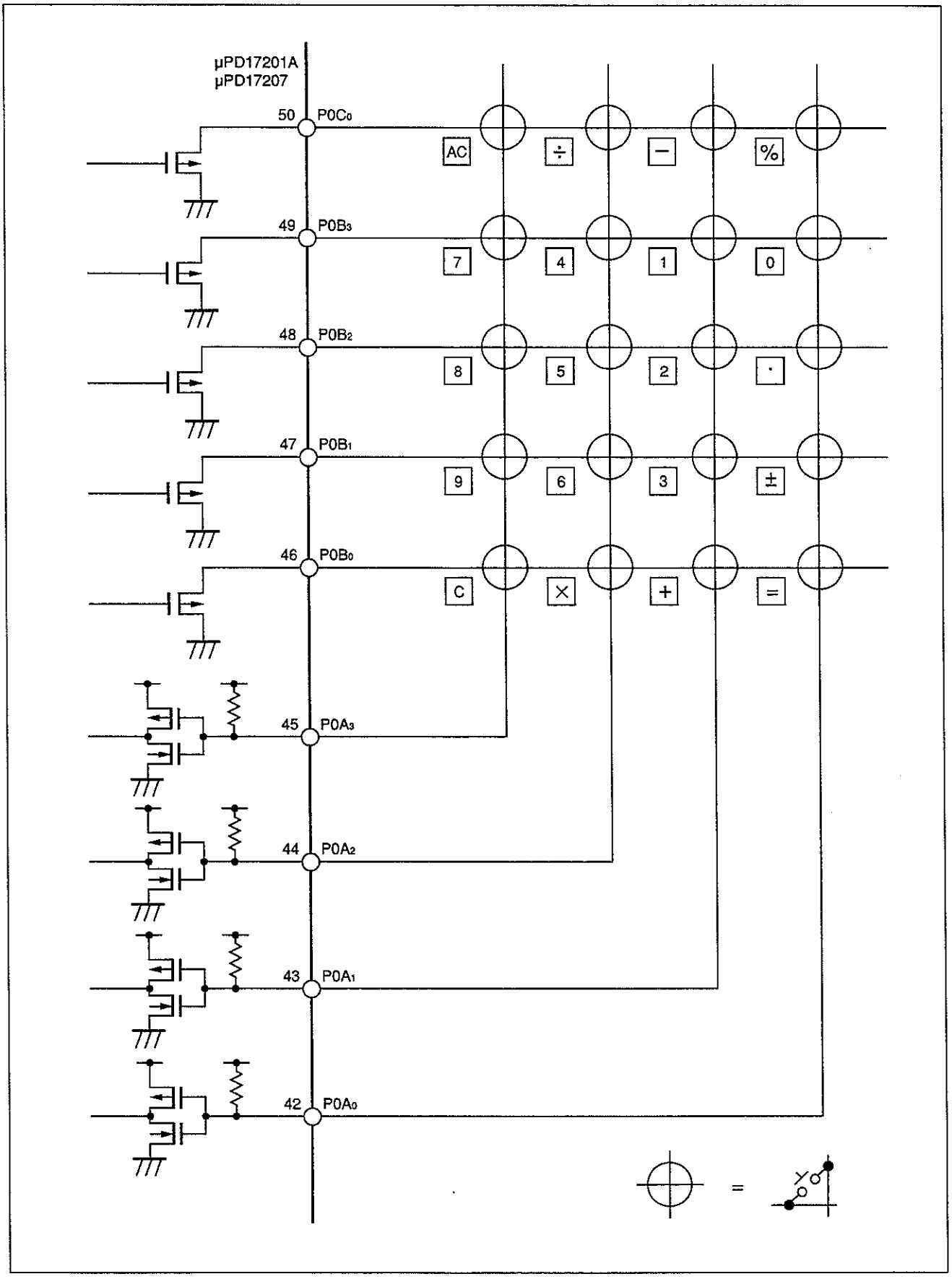
2.6 Key Input

The key input circuit of the pocket calculator introduced in this manual consists of a 5-by-4 matrix. Key scan signals are output on $P0B_0$-$P0B_3$ and $P0C_0$ every 10 ms, and port states are applied to $P0A_0$-$P0A_3$ to determine which key is pressed. Key scan operation is based on the active low mode. Each of the $P0B_0$-$P0B_3$ and $P0C_0$ pins is made low at the respective time interval, and $P0A_0$-$P0A_3$ are read in synchronism; the input of a key is determined from low level detection.

When the same key state is detected three times, the input of a key is determined, and a keying operation is executed. When the input of a key is determined, the input of any other keys is rejected until the key is released (until the state where no key is pressed is determined). If two or more keys are pressed at a time, multiple-key pressing is assumed; no key input is accepted until all keys are released.

Figure 2-4 shows the key input circuit diagram of the pocket calculator.

Figure 2-4. Key Input Circuit Diagram



Chapter 3

Pocket Calculator Specifications

3.1 Specifications

Display function	: 8-digit LCD display (1 digit for a sign, and 7 digits for a numeric)
Input method	: 5-by-4 matrix key input Lock-out method(Note)
Number of input/display digits	: 7 digits maximum (±0.000001 to ±9999999)
Special function	: Automatic-power-off function (when no key input is performed for three minutes)
Types of operations	: • Arithmetic operations • Percentage calculation, percentage-increase calculation, percentage-decrease calculation

(Note) Until a key pressed is released, the input of any other key is rejected.

3.2 Keys

The pocket calculator has the keys indicated below.

(1) Numeric keys ([0] [1] [2] [3] [4] [5] [6] [7] [8] [9])

These keys are used to enter a numeric. A numeric as long as 7 digits excluding a sign can be entered.

(2) Decimal-point key ([·])

This key is used to enter the decimal point.

(3) Plus/Minus key ([±])

This key is used to change the sign of a numeric being entered.

(4) Operator keys ([+] [−] [×] [÷])

An operator key is used to enter the respective operator. When operator keys are pressed in succession, the last key pressed is selected.

Key input	Display
[2] [5] [0]	2 5 0.
[+]	2 5 0.
[−]	2 5 0.
[×]	2 5 0.
[2]	2.
[=]	5 0 0.

In this example, [250 x 2] is executed, and the result of operation is 500.

(5) Percent key ([%])

This key is used for percentage calculation, percentage-increase calculation, and percentage-decrease calculation.

(6) Equal key ([=])

This key is to be pressed to find the result of operation.

(7) Clear key ([C])

This key is used to correct a numeric being entered. Any numeric and operator entered before a numeric corrected are preserved.

Key input	Display
[5] [0] [0]	5 0 0.
[+]	5 0 0.
[4] [0] [0]	4 0 0.
[C]	0.
[3] [0] [0]	3 0 0.
[=]	8 0 0.

In this example, [500 + 400] is corrected to [500 + 300], and the result of operation is 800.

Key input	Display
[7] [5] [0]	7 5 0.
[×]	7 5 0.
[C]	0.
[=]	0.

In this example, [750 x 0] is executed, and the result of operation is 0.

(8) All clear key ([AC])

This key is used to start a new calculation, clearing all data entered so far.

Key input	Display
[9] [0] [0]	9 0 0.
[−]	9 0 0.
[1] [0] [0]	1 0 0.
[AC]	0.
[7] [0] [0]	7 0 0.
[×]	7 0 0.
[5]	5.
[=]	3 5 0 0.

In this example, the input of [900 – 100] is entirely cleared. Instead, [700 x 5] is executed, and the result of operation is 3500.

This key also functions as the power-on switch.

3.3 State Transition

The pocket calculator has six modes as described below.

(1) First-term input mode

This mode is used to enter the first term of an operation.

(2) Operator selection mode

This mode is used to enter an operator.

(3) Second-term input mode

This mode is used to enter the second term of an operation.

(4) Percentage calculation mode

This mode is used for percentage calculation, percentage-increase calculation, and percentage-decrease calculation.

(5) Error mode

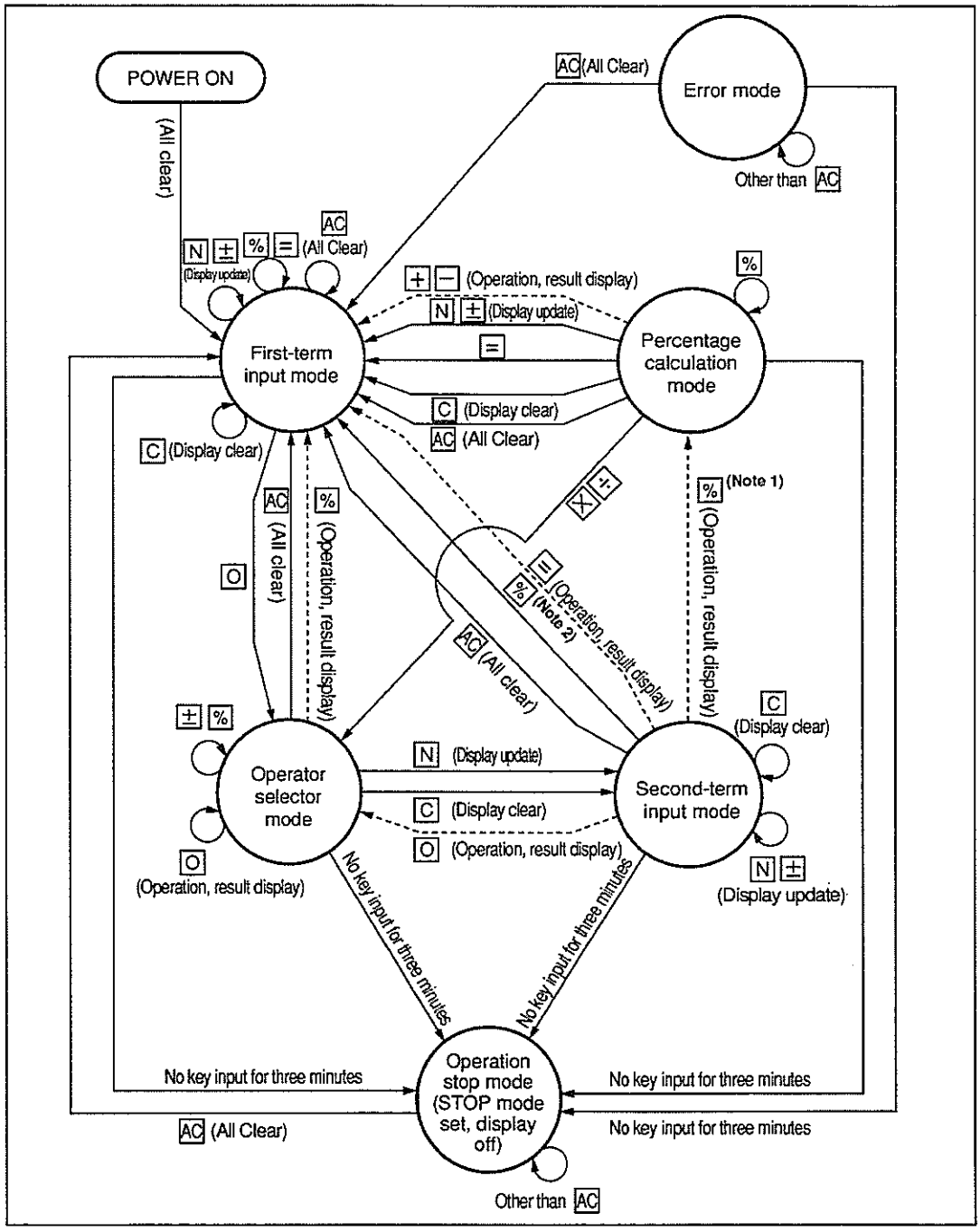
This mode provides an error indication when a division by zero is attempted or an overflow occurs as the result of an operation.

(6) Operation stop mode

The STOP mode is set if no key input is performed for three minutes.

Figure 3-1 shows the state transition diagram.

Figure 3-1. State Transition Diagram



(Note 1) Processing when the operator × or ÷ is entered

(Note 2) Processing when the operator + or − is entered

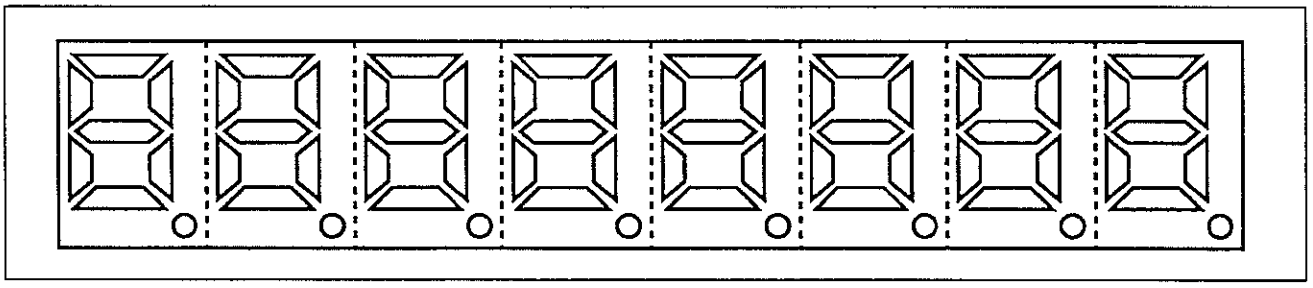
Remarks 1. N: 0 – 9, ·
O: +, −, ×, ÷

2. ( ) : Indicates processing by key input.
——→ : Indicates normal state transition.
- - - - → : Enters the error mode when an operation produces an error.

3.4 Display

As shown in Figure 3-2, an 8-segment by 8-digit LCD is used to display an entered numeric and the result of an operation.

Figure 3-2. Display



3.5 Automatic Power-off

The power to the pocket calculator is automatically turned off when no key input is performed for three minutes. To turn on the power again, press the [AC] key.

3.6 Examples of Operations

Examples of key input procedures and display are given below for operations that can be performed according to the specifications of the pocket calculator.

(1) Addition

123 + 4560.7 = 4683.7

Key input	Display
[1] [2] [3]	1 2 3.
[+]	1 2 3.
[4] [5] [6] [0] [·] [7]	4 5 6 0.7
[=]	4 6 8 3.7

–852 + 147 = –705

Key input	Display
[8] [5] [2] [±]	– 8 5 2.
[+]	– 8 5 2.
[1] [4] [7]	1 4 7.
[=]	– 7 0 5.

611 + 0 = 611

Key input	Display
[6] [1] [1]	6 1 1.
[+]	6 1 1.
[=]	6 1 1.

(2) Subtraction

951 – 627.84 = 323.16

Key input	Display
[9] [5] [1]	9 5 1.
[–]	9 5 1.
[6] [2] [7] [·] [8] [4]	6 2 7.8 4
[=]	3 2 3.1 6

–753.94 – 186 = –939.94

Key input	Display
[7] [5] [3] [·] [9] [4] [±]	– 7 5 3.9 4
[–]	– 7 5 3.9 4
[1] [8] [6]	1 8 6.
[=]	– 9 3 9.9 4

–2098 – 0 = –2098

Key input	Display
[2] [0] [9] [8] [±]	– 2 0 9 8.
[–]	– 2 0 9 8.
[=]	– 2 0 9 8.

(3) Multiplication

279 x (–0.58) = –161.82

Key input	Display
[2] [7] [9]	2 7 9.
[×]	2 7 9.
[·] [5] [8] [±]	– 0.5 8
[=]	– 1 6 1.8 2

35.704 x 22.194 = 792.414576

Key input	Display
[3] [5] [·] [7] [0] [4]	3 5.7 0 4
[×]	3 5.7 0 4
[2] [2] [·] [1] [9] [4]	2 2.1 9 4
[=]	7 9 2.4 1 4 5

Remark Only the seven digits starting from the most significant digit are displayed for decimals.

519 x 519 = 269361

Key input	Display
[5] [1] [9]	5 1 9.
[×]	5 1 9.
[=]	2 6 9 3 6 1.

(4) Division

–769.3 ÷ (–245) = 3.14

Key input	Display
[7] [6] [9] [·] [3] [±]	– 7 6 9.3
[÷]	– 7 6 9.3
[2] [4] [5] [±]	– 2 4 5.
[=]	3.1 4

412 ÷ (–618) = –0.666666···

Key input	Display
[4] [1] [2]	4 1 2.
[÷]	4 1 2.
[6] [1] [8] [±]	– 6 1 8.
[=]	– 0.6 6 6 6 6 6

Remark Only the seven digits starting from the most significant digit are displayed for decimals.

–9.98 ÷ (–9.98) = 1

Key input	Display
[9] [·] [9] [8] [±]	– 9.9 8
[÷]	– 9.9 8
[=]	1.

(5) Mixed calculation

{(45 + 38) x 0.75 – 25} ÷ (–50) = –0.745

Key input	Display
4 5	4 5.
+	4 5.
3 8	3 8.
×	8 3.
· 7 5	0.7 5
–	6 2.2 5
2 5	2 5.
÷	3 7.2 5
5 0 ±	– 5 0.
=	– 0.7 4 5

Remark In a calculation including a mixture of arithmetic operations, priority is not given to multiplication and division, but a calculation is made in the order of operator input.

(6) Percentage calculation

645 x 32.9% = 212.205

Key input	Display
6 4 5	6 4 5.
×	6 4 5.
3 2 · 9	3 2.9
%	2 1 2.2 0 5

(7) Percentage-increase calculation

400 + (400 x 60%) = 640

Key input	Display
[4] [0] [0]	4 0 0.
[×]	4 0 0.
[6] [0]	6 0.
[%]	2 4 0.
[+]	6 4 0.

(8) Percentage-decrease calculation

32.5 – (32.5 x 70%) = 9.75

Key input	Display
[3] [2] [·] [5]	3 2.5
[×]	3 2.5
[7] [0]	7 0.
[%]	2 2.7 5
[–]	9.7 5

(9) Examples of error occurrence

<1> When a division by zero is attempted

500 ÷ 0

Key input	Display
[5] [0] [0]	5 0 0.
[÷]	5 0 0.
[0]	0.
[=]	E.

<2> When the result of calculation ≥ 1.0 x 10^7, or the result of calculation ≤ 1.0 x 10^7

4500 x 4500 = 2.025 x 10^7

Key input	Display
[4] [5] [0] [0]	4 5 0 0.
[×]	4 5 0 0.
[=]	E.

(10) Example of underflow occurrence

When –1.0 x 10^{-6} < result of calculation < 1.0 x 10^{-6}

0.0063 ÷ (–9000) = –7 x 10^{-7}

Key input	Display
[·] [0] [0] [6] [3]	0.0 0 6 3
[÷]	0.0 0 6 3
[9] [0] [0] [0] [±]	– 9 0 0 0.
[=]	0.

(11) Others

Key input	Display
[3] [8]	3 8.
[%]	3 8.
[×]	3 8.
[5] [7]	5 7.
[=]	2 1 6 6.
[1] [9]	1 9.
[=]	1 9.
[−]	1 9.
[%]	1 9.
[+]	1 9.
[4] [8]	4 8.
[=]	6 7.

In the example above, [38 x 57] is first executed, then [19 + 48] is executed.

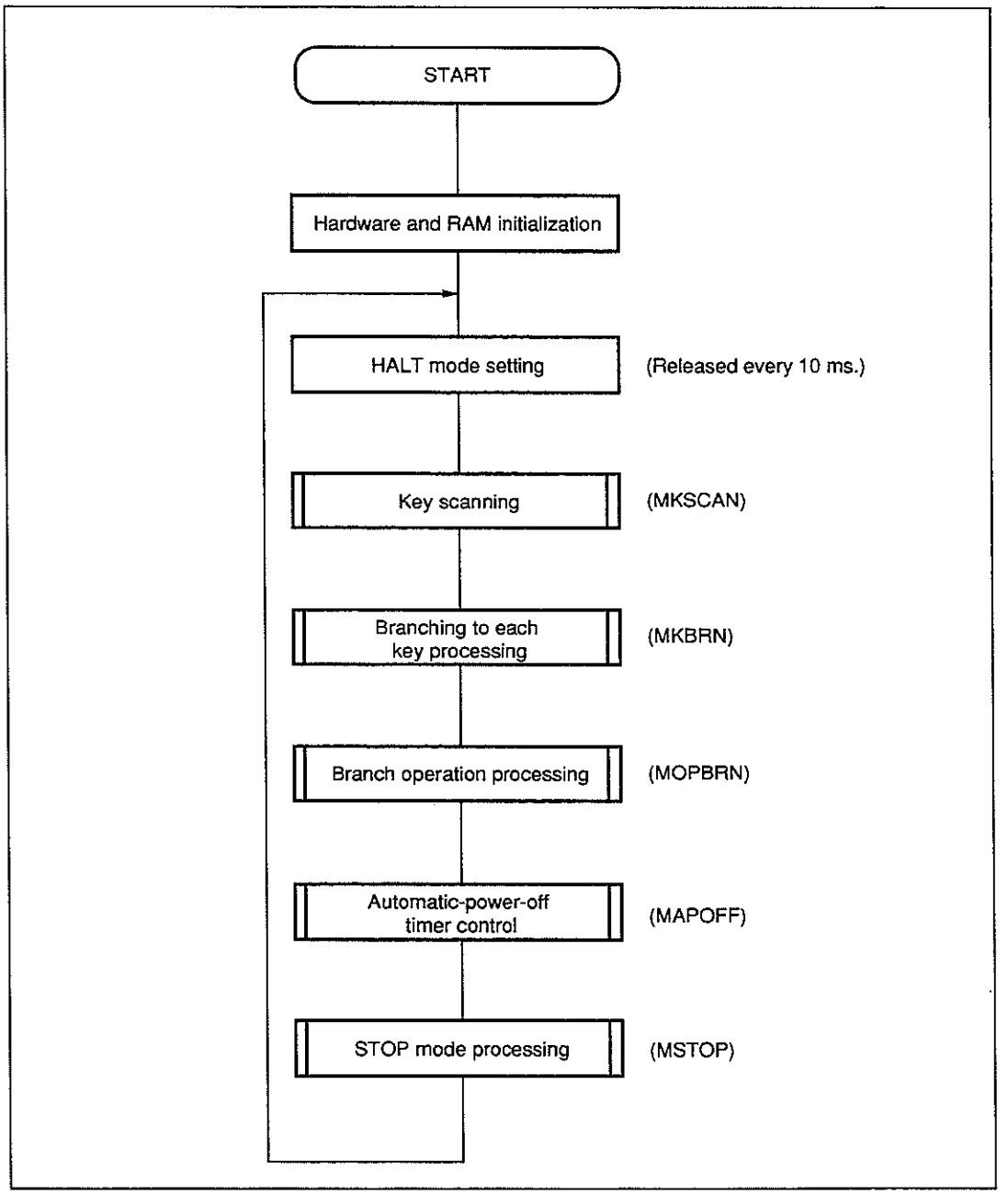
Chapter 4

Overview of System Control Section

This chapter outlines the major processing of the system control section of the pocket calculator, and also provides general flowcharts.

4.1 Main Processing

When the pocket calculator program starts, the hardware and RAM are initialized before main processing is performed. The pocket calculator performs main processing every 10 ms to reduce supply current. Each time main processing is completed, the HALT mode is set.



4.2 Key Scanning

Key scan signals are output on $P0B_0$-$P0B_3$ and $P0C_0$, and key state data is applied to $P0A_0$-$P0A_3$. Chattering processing is performed three times; each time lasts for 10 ms. A key code is generated at the end of chattering. At this time, the key-processing request flag is set for a valid key input.

Table 4-1 indicates the correspondence between input keys and key codes. Table 4-2 indicates the valid key inputs.

Table 4-1. Correspondence between Input Keys and Key Codes

Low-order digits \ High-order digits		RKCODH (0.30H) value					
		7H	BH	DH	EH	0H	FH
RKCODL (0.31H) value	7H	7	8	9	C	AC	–
	BH	4	5	6	×	÷	–
	DH	3	2	1	+	–	–
	EH	0	·	±	=	%	–
	FH	–	–	–	–	–	Key-off(Note)
	0H	–	–	–	–	–	Multiple-key pressing

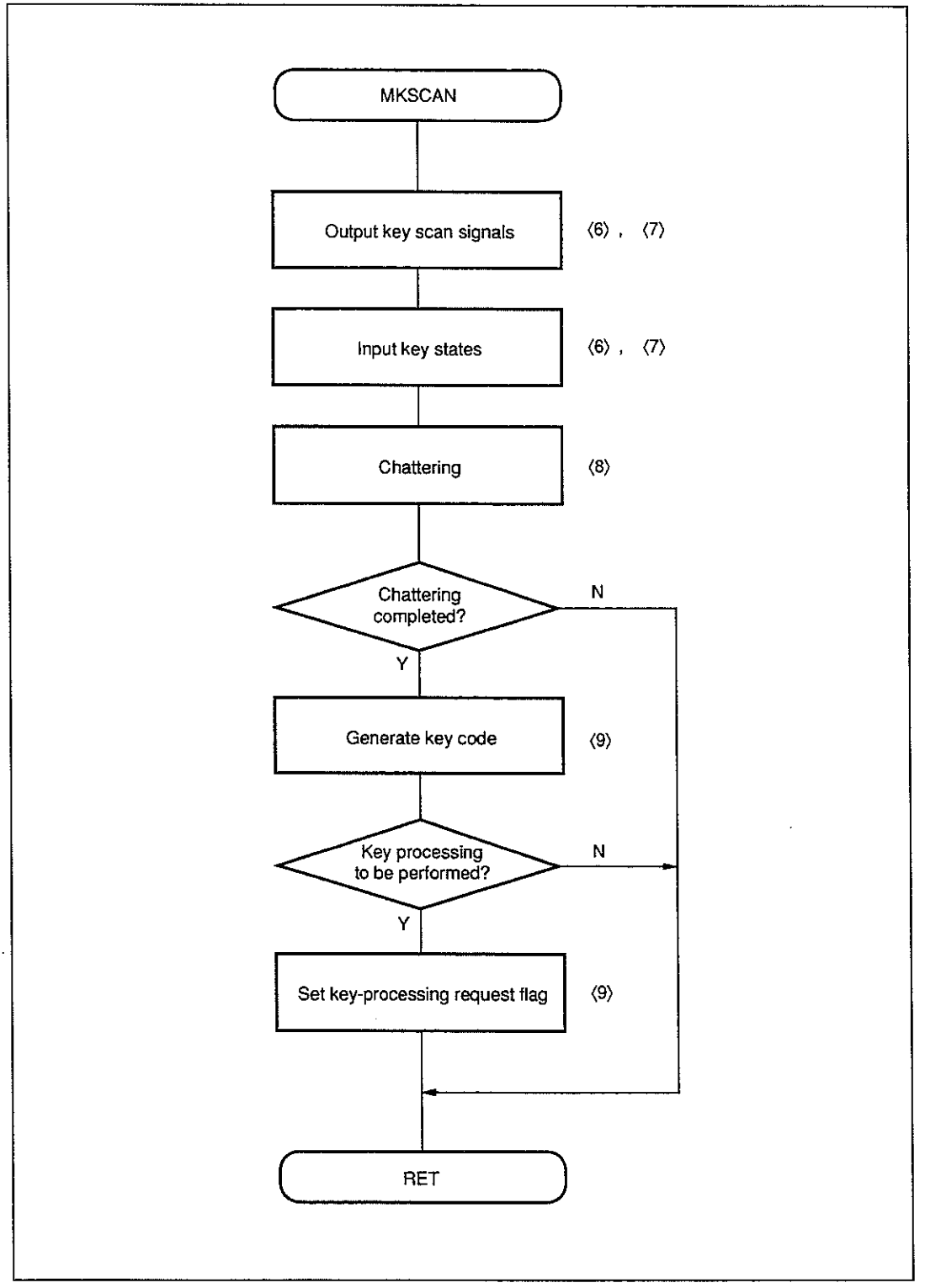
(Note) State where no key is pressed

Table 4-2. Valid Key Inputs

		Previous key code		
		Key-off (Note)	Single-key pressing	Multiple-key pressing
Current key code	Key-off (Note)	–	○	○
	Single-key pressing	○	×	×
	Multiple-key pressing	×	×	–

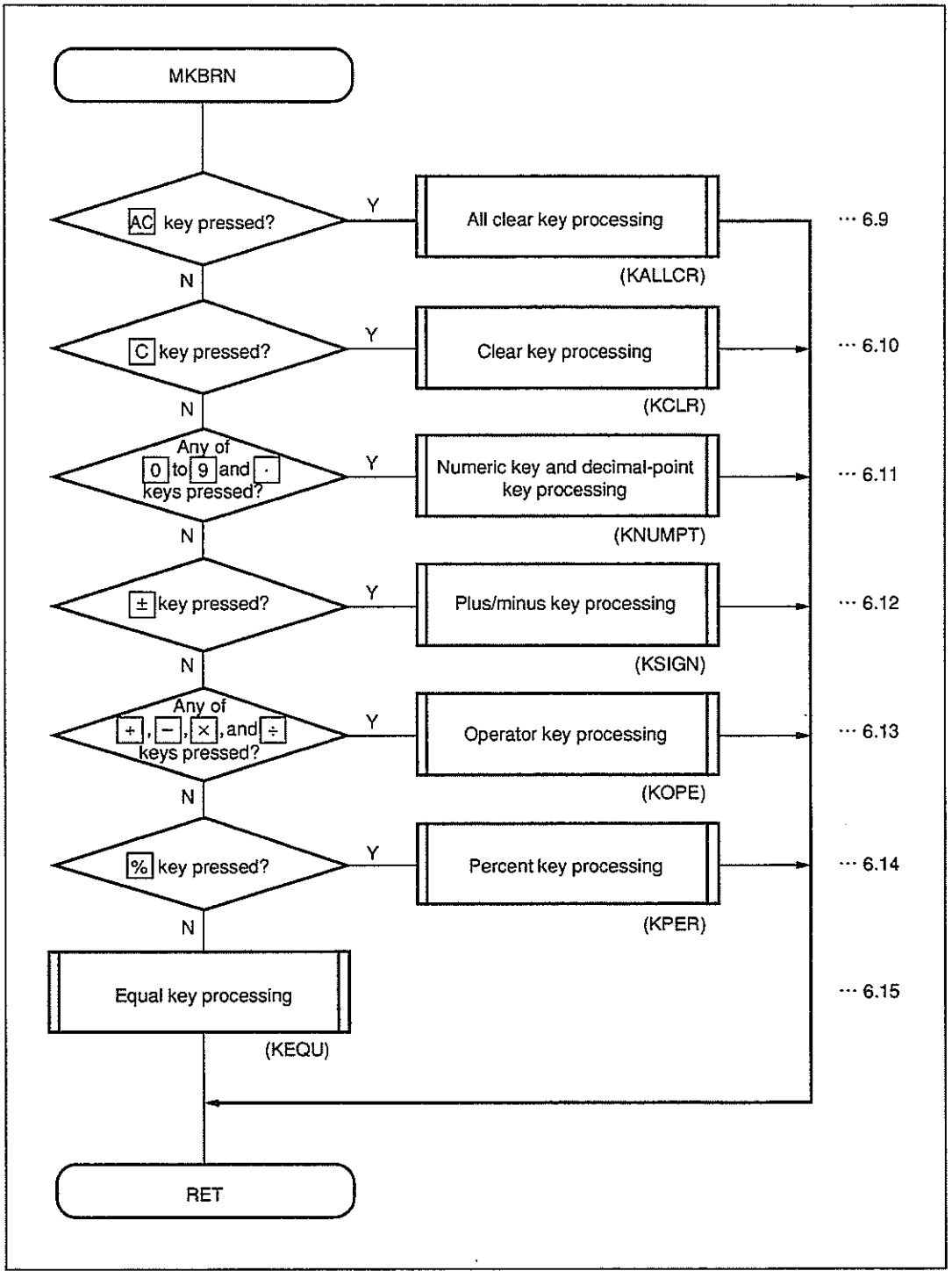
(Note) State where no key is pressed

Remarks A circle (○) represents a valid key input.
A hyphen (–) represents a case that cannot occur with the pocket calculator.


MKSCAN
Output key scan signals
⟨6⟩ , ⟨7⟩
Input key states
⟨6⟩ , ⟨7⟩
Chattering
⟨8⟩
Chattering completed?
N
Y
Generate key code
⟨9⟩
Key processing to be performed?
N
Y
Set key-processing request flag
⟨9⟩
RET

4.3 Branching to Each Key Processing

Each key code is decoded, and the respective key processing is called.



4.4 Branch Operation Processing and Error Handling

Each arithmetic processing (addition, subtraction, multiplication, or division) is called, and the result of calculation is displayed. An error is displayed when division by zero is attempted, or an overflow occurs (result of calculation $\geq 1.0 \times 10^{7}$, or result of calculation $\leq -1.0 \times 10^{7}$). When an underflow occurs (-1.0×10^{-6} < result of calculation < 1.0×10^{-6}), 0 is displayed as the result of calculation.

Which arithmetic operation is to be called is determined using operator data. Operator data changes with operator key input. Two operator data storage areas are available: ROPE (60H of BANK0) and RCOM (61H of BANK0).

Table 4-3 indicates an example of change in operator data.

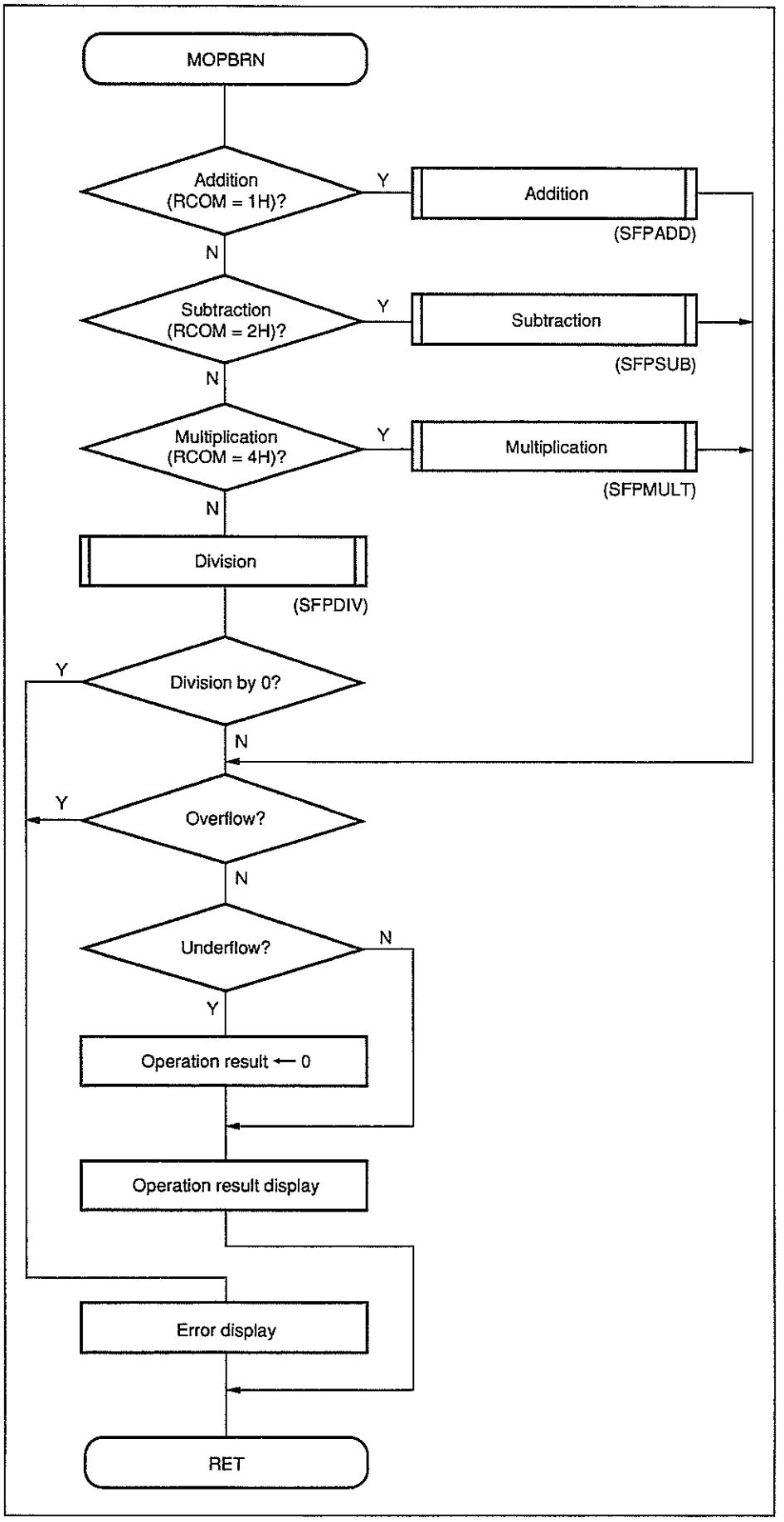
Table 4-3. Change in Operator Data (1/2)

Input key	ROPE (0.60H) value	RCOM (0.61H) value
AC	0H	0H
RAM all clear processing writes 0H to ROPE and RCOM.		
3 0 +	1H	0H
−	2H	0H
When an operator key is pressed after the first numeric, the operator data is stored at ROPE. If a separate operator key is pressed successively, the value of ROPE is updated; the value of RCOM remains unchanged. The correspondence between the operator keys and operator data is as follows: + key : 1H − key : 2H × key : 4H ÷ key : 8H		
2 0 =	2H	2H
2 0 ×	4H	2H
When the equal key is pressed after the second numeric, the operator data at ROPE is transferred to RCOM. The branch operation processing determines the operator from the value of RCOM, then calls the corresponding operation processing. Then, the result of calculation is displayed. If a separate operator key is pressed after the second numeric, the new operator data is stored at ROPE.		

Table 4-3. Change in Operator Data (2/2)

Input key	ROPE (0.60H) value	RCOM (0.61H) value
AC	0H	0H
5 0 ×	4H	0H
7 0 %	4H	4H
+	1H	1H

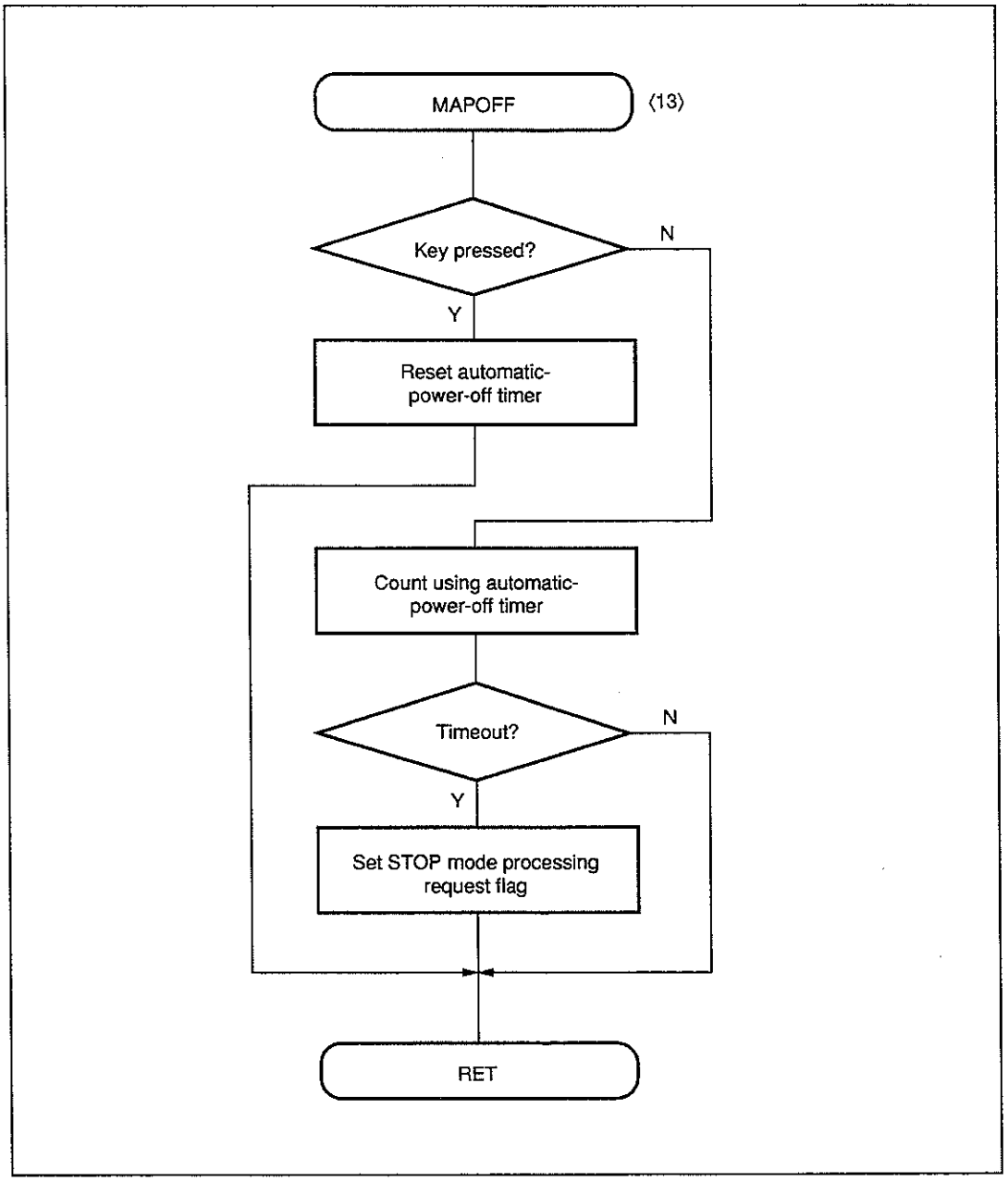
In precentage calculation, the value of ROPE is determined when the percent key is pressed. Only when the value of ROPE is 4H or 8H (that is, only when the operator key for × or ÷ is pressed), the operator data at ROPE is transferred to RCOM to execute a percentage calaculation. If the + or − key is pressed after a percentage calculation, the operator data is stored at ROPE and RCOM to execute a percentage-increase calculation or percentage-decrease calaculation.



Remark
For details of each operation, see **Section 8.2**.

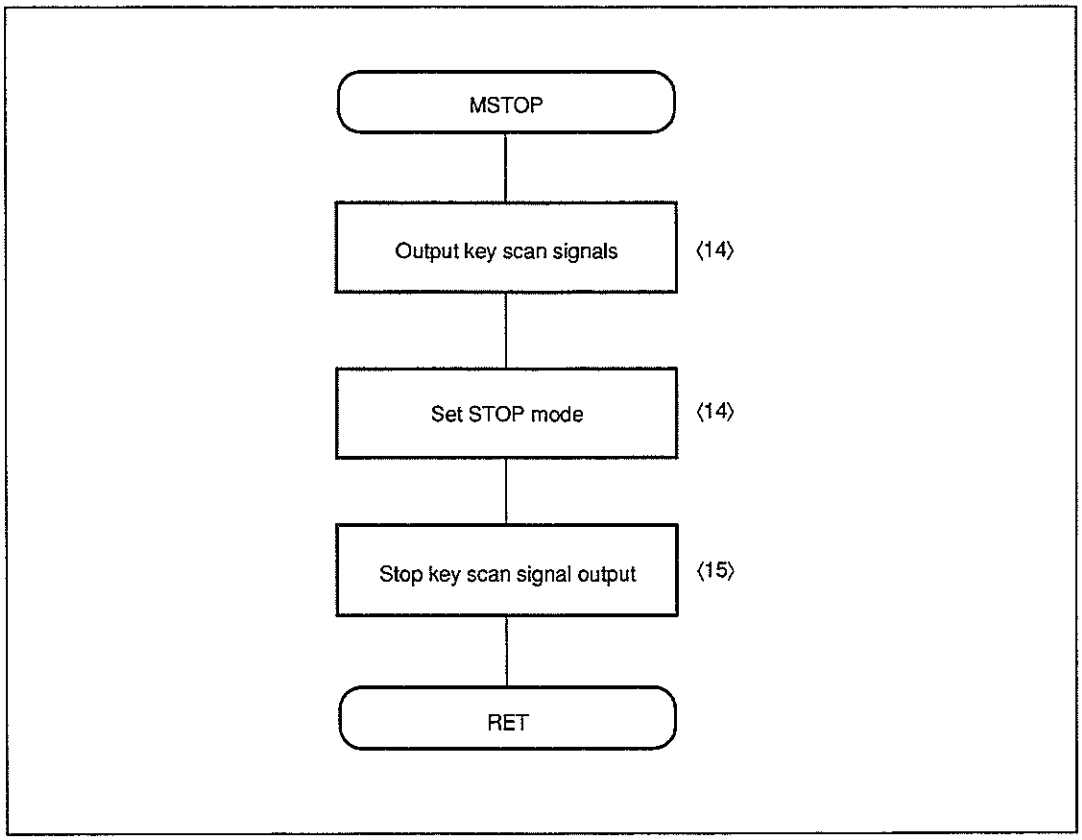
4.5 Automatic-Power-off Timer Control

When a key is pressed, the automatic-power-off timer is reset. In the key-off state, a count operation is performed using the automatic-power-off timer. In the case of a timeout, the STOP mode processing request flag is set.



4.6 STOP Mode Processing

The STOP mode is set by outputting all key scan signals on $P0B_0$-$P0C_0$. After the STOP mode is released by key input, key scan signal output is stopped.



Chapter 5
RAM Layout and Variables

5.1 RAM Layout

Figure 5-1 indicates the RAM layout for the programs introduced in this manual.

Figure 5-1. RAM Layout (1/2)

[BANK 0]

ROW ADDRESS	COLUMN ADDRESS 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0 3	RREG0	RREG1	RREG2	RREG3	RREG4	RREG5	RREG6	RREG7	RREG8	RREG9	RREG10	RREG11	DBF3	DBF2	DBF1	DBF0
0 2																
0 1																
0 0																
1 3	RNUMC	RPTLOC	RSINLOC	RDSIGN	RDEXP				RDLSD							RDMSD
1 2																
1 1																
1 0																
2 3				RZLSD												RZMSD
2 2																
2 1																
2 0																
3 3	RKCODH	RKCODL	RCHCODH	RCHCODL	RIPCODH	RIPCODL	RCHATC		RSTOP3	RSTOP2	RSTOP1	RSTOP0				
3 2																
3 1																
3 0																
4 3	LCDD0	LCDD1	LCDD2	LCDD3	LCDD4	LCDD5	LCDD6	LCDD7	LCDD8	LCDD9	LCDD10	LCDD11	LCDD12	LCDD13	LCDD14	LCDD15
4 2																
4 1																
4 0																
5 3	LCDD16	LCDD17	LCDD18	LCDD19	LCDD20	LCDD21	LCDD22	LCDD23	LCDD24	LCDD25	LCDD26	LCDD27	LCDD28	LCDD29	LCDD30	LCDD31
5 2																
5 1																
5 0																
6 3	ROPE	RCOM			RMODE				RSYSFLG FSTOP FFALSE FOPEND FPER	REVEFLG FSTPREQ FOPREQ FMULTI FKEYREQ						
6 2																
6 1																
6 0																
7 3	P0A	P0B	P0C	P0D	AR3	AR2	AR1	AR0	WR	BANK	IXH/MPH MPE	IXM/MPL	IXL	RPH	RPL	PSW CMP CY Z IXE
7 2															BCD	
7 1																
7 0																

Figure 5-1. RAM Layout (2/2)

[BANK 1]

		COLUMN ADDRESS 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	3				RXSIGN	RXEXP		RXLSD									RXMSD
	2																
	1																
	0																
1	3				RYSIGN	RYEXP		RYLSD									RYMSD
	2																
	1																
	0																
2	3				RWSIGN	RWEXP		RWLSD									RWMSD
	2																
	1																
	0																
3	3	ROPFLG															
	2	FOVER															
	1	FDVERR															
	0	FZERO															
4	3	FEXCHG															
	2																
	1																
	0																
5	3																
	2																
	1																
	0																
6	3																
	2																
	1																
	0																
7	3	P1A				AR3	AR2	AR1	AR0	WR	BANK	IXH/MPH MPE	IXM/MPL	IXL	RPH	RPL	PSW
	2																CMP
	1																CY
	0															BCD	Z IXE

R O W A D D R E S S

5.2 RAM and Flags

RAM and flag name	RAM address	Name	Description
RREG0 | RREG11	0.00H | 0.0BH	Register 0 to register 11	These registers are used as work registers.
RNUMC	0.10H	Numeric key counter	This counter counts the number of numeric key pressing operations.
RPTLOC	0.11H	Decimal-point position area	This area stores the display position of the decimal point.
RSINLOC	0.12H	Sign display position area	This area stores the display position of a sign.
REGD		Display data register	
RDSIGN	0.13H	Operation result sign area	This area is used to check the sign of the result of operation. When bit 0 is set to 0: Positive When bit 0 is set to 1: Negative
RDEXP	0.14H	Display data exponent area	This area is used for display data normalization and for conversion of the result of operation to display data.
RDLSD | RDMSD	0.18H | 0.1FH	Display data area	This area stores display data.
RZLSD | RZMSD	0.23H | 0.2FH	Saving register (REGZ)	This area is a data save area used to save data to be operated in a percentage calculation.
RKCODH RKCODL	0.30H 0.31H	Key code area	This area stores a key code for which chattering processing is completed.
RCHCODH RCHCODL	0.32H 0.33H	Chattering code area	This area stores a key code for which chattering processing is being performed.
RIPCODH RIPCODL	0.34H 0.35H	Input code area	This area stores the key code of the key currently being pressed.
RCHATC	0.36H	Chattering counter	This counter counts the number of chattering processing occurrences.
RSTOP3 RSTOP2 RSTOP1 RSTOP0	0.38H 0.39H 0.3AH 0.3BH	Automatic-power-off timer	This timer is a three-minute timer used for automatic power-off.
LCDD0 | LCDD31	0.40H | 0.5FH	LCD segment data register	These registers store segment data to be displayed on the LCD.

RAM and flag name	RAM address	Name	Description
ROPE	0.60H	Operator area	This area stores the data of an operator key pressed at the end. 1H: Addition (+ key) 2H: Subtraction (− key) 4H: Multiplication (× key) 8H: Division (÷ key)
RCOM	0.61H	Area for the operator to be executed	This area stores an operator to be executed.
RMODE	0.64H	Mode area	This area stores the current mode. 1H: Second-term input mode 2H: Operator selection mode 4H: First-term input mode 8H: Error mode
RSYSFLG	0.68H	System flag area	This area contains flags used with the system control section.
FPER	0.68H.0	Percent flag	This flag is set when the percent key is pressed.
FOPEND	0.68H.1	Operation end flag	This flag is set at the end of an operation.
FFALSE	0.68H.2	Illegal-input flag	This flag is set when the percent key or equal key is pressed at an illegal position.
FSTOP	0.68H.3	Operation stop mode flag	This flag is set when the STOP mode is set.
REVEFLG	0.69H	Event flag area	This area contains various processing request flags.
FKEYREQ	0.69H.0	Key-processing request flag	This flag is set when valid key inputs (including key-off) are determined.
FMULTI	0.69H.1	Multiple-key flag	This flag is set when multiple-key pressing is detected in key scanning.
FOPREQ	0.69H.2	Operation request flag	This flag is set when a key that allows operation is pressed.
FSTPREQ	0.69H.3	STOP mode processing request flag	This flag is set when the automatic-power-off timer times out.
REGX		Floating-point register 1	This is a register for operation. This register stores data to be operated and the result of an operation.
RXSIGN	1.03H		Sign part of REGX
RXEXP	1.04H 1.05H		Characteristic of REGX
RXLSD | RXMSD	1.06H | 1.0FH		Mantissa of REGX

RAM and flag name	RAM address	Name	Description
REGY		Floating-point register 2	This is a register for operation. This register stores operation data.
RYSIGN	1.13H		Sign part of REGY
RYEXP	1.14H 1.15H		Characteristic of REGY
RYLSD | RYMSD	1.16H | 1.1FH		Mantissa of REGY
REGW		Floating-point register 3	This is a work register for operation.
RWSIGN	1.23H		Sign part of REGW
RWEXP	1.24H 1.25H		Characteristic of REGW
RWLSD | RWMSD	1.26H | 1.2FH		Mantissa of REGW
ROPFLG	1.30H	Operation flag	This flag is used for floating-point operation.
FEXCHG	1.30H.0	Register exchange flag	This flag is set when the values of floating-point register 1 and floating-point register 2 are exchanged with each other in addition or subtraction.
FZERO	1.30H.1	Operation result zero flag	This flag is set when the result of operation is 0.
FDVERR	1.30H.2	Zero-division error flag	This flag is set when division by 0 is attempted.
FOVER	1.30H.3	Overflow flag	This flag is set when an operation has produced an overflow.

5.3 Processing Names and RAM Names

To improve efficiency in the coding of the pocket calculator programs introduced in this manual, an uppercase letter is prefixed to all processing names, label names, and RAM names for identification.

The meanings of the uppercase letters prefixed to the processing, label, and RAM names are explained below.

(1) Processing names

M **** : Main routine

S **** : Subroutine

I **** : Interrupt routine

(2) Label names

L **** : Branch destination label in a main routine

J **** : Branch destination label in a subroutine

H **** : Branch destination label in an interrupt routine

T **** : Table label

(3) RAM names

R **** : Name of a RAM area used in units of nibbles (4 bits)

F **** : Flag name

Chapter 6

Flowcharts of System Control Section

This chapter provides the flowcharts of the system control section of the pocket calculator.

6.1 Initialization

Input variables	Flowchart ⟨1⟩	Processing and remarks	Output variable
	IRESET		
	DI	Disables interrupt.	
	SP ← 5H	Sets stack pointer.	
	PCC ← 3H	Sets system clock (main clock).	
	P1A ← 0000B P0B ← 1111B P0C ← 0001B P0D ← 0000B	Initializes output ports.	P1A (1.70H) P0B (0.71H) P0C (0.72H) P0D (0.73H)
	PM1 ← 1110B PM2 ← 1111B	Sets port I/O. Output : P0B : P0C : P0D : P1A Input : P0A	
	TMCT ← 0110B	Sets 8-bit timer operating clock (system clock/256).	
	TMM ← 9CH	Sets 8-bit timer modulo register.	
	WTC ← 0000B	Sets clock-timer selector.	
	⟨2⟩ 1 (Note)		

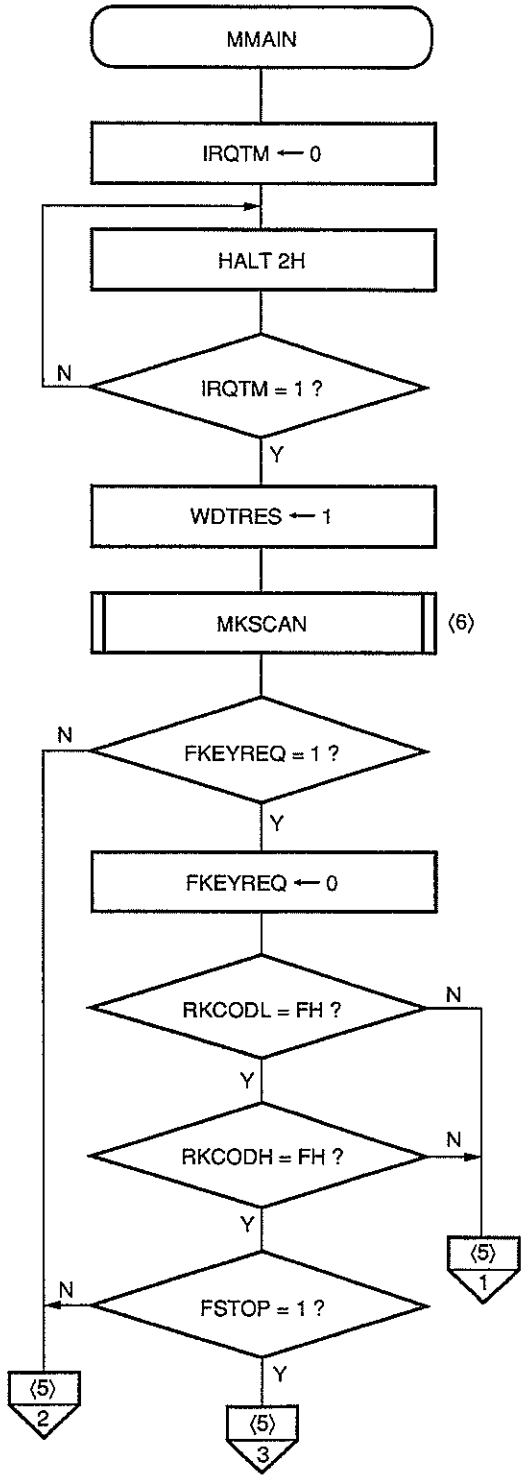
(Note) Jumps to <1> of flowchart ⟨2⟩.

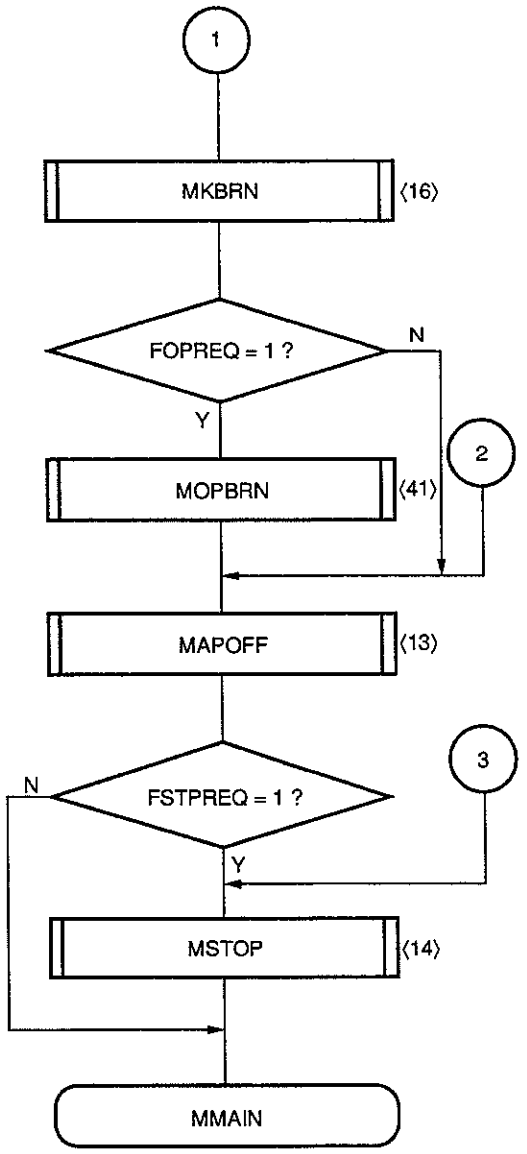
Input variables	Flowchart ⟨2⟩	Processing and remarks	Output variable
	1		
	LCDM ← 0001B LCDC ← 0001B	Sets LCD controller/driver. Display mode : 1/2 duty frame frequency : Main clock/2^{13}	
	ADM ← 0000B	Sets A/D converter to standby state.	
	PSL ← 0000B SIC ← 0000B	Sets serial interface (for nonuse).	
	IPF ← 0000B	Sets request enable flag (to disable all interrupts).	
	RPH ← 0000B RPL ← 0000B	Sets general-purpose register (00H-0FH of BANK0).	RPH RPL
	SRAMCR ⟨45⟩	Clears RAM.	
	⟨3⟩ 1		

Input variables	Flowchart ⟨3⟩	Processing and remarks	Output variable
	1		
	RSTOP0 ← FH RSTOP1 ← 4H RSTOP2 ← 6H RSTOP3 ← 4H	Resets automatic-power-off timer.	RSTOP0 RSTOP1 RSTOP2 RSTOP3 (0.38H-0.3BH)
	RDEXP ← 8H	Resets display data exponent area.	RDEXP (0.14H)
	RKCODL ← FH RKCODH ← FH RCHCODL ← FH RCHCODH ← FH	Resets key code and chattering code.	RKCODL RKCODH (0.30H-0.31H) RCHCODL RCHCODH (0.32H-0.33H)
	RMODE ← 4H	Sets first-term input mode.	RMODE (0.64H)
	SDPINI ⟨47⟩	Initializes display data area.	
	SDISP ⟨61⟩	Converts display data to LCD segment data.	
	LCDEN ← 1	Turns on LCD display.	
	TMEN ← 1	Starts count operation using 8-bit timer.	
	MMAIN		

6.2 Main Processing

Input variables	Flowchart ⟨4⟩	Processing and remarks	Output variable
	MMAIN		
	IRQTM ← 0	Clears 8-bit timer interrupt request flag.	
	HALT 2H	Sets HALT mode (released every 10 ms).	
	IRQTM = 1 ? (N / Y)		
	WDTRES ← 1	Resets watchdog timer.	
	MKSCAN ⟨6⟩	Key scanning	
FKEYREQ (0.69H.0)	FKEYREQ = 1 ? (N / Y)	Valid key input (including key-off) determined?	
	FKEYREQ ← 0	Clears key-processing request flag.	
RKCODL (0.31H)	RKCODL = FH ? (N / Y)	Key-off?	
RKCODH (0.30H)	RKCODH = FH ? (N / Y)		
FSTOP (0.68H.3)	FSTOP = 1 ? (N / Y)	Operation stop mode?	

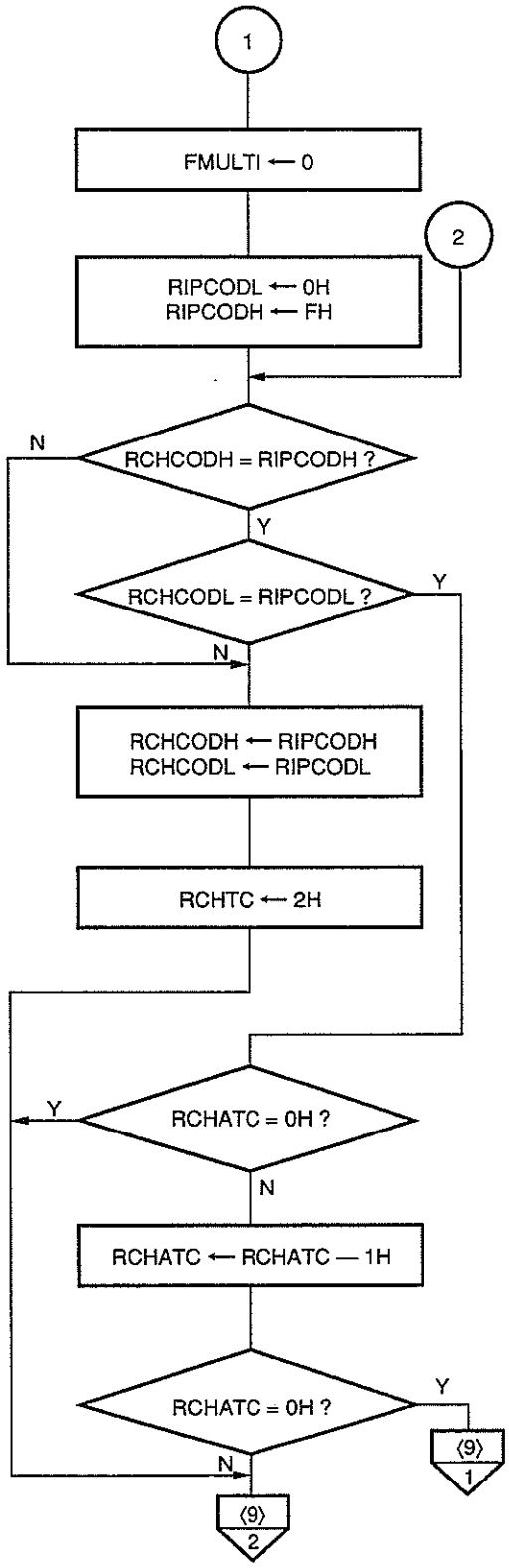


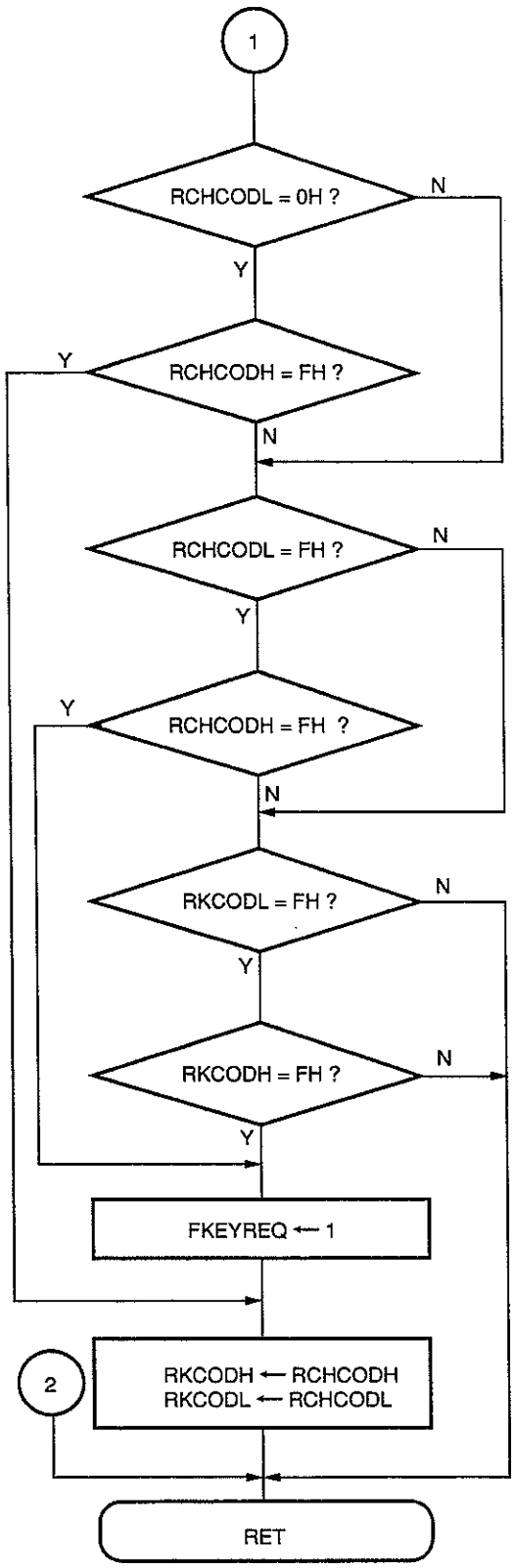
Input variables	Flowchart ⟨5⟩	Processing and remarks	Output variable
			
		Branching to each key processing	
FOPREQ (0.69H.2)		Operation possible?	
		Branch operation processing	
		Automatic-power-off timer control	
FSTPREQ (0.69H.3)		Timeout?	
		STOP mode processing	

6.3 Key Scanning

Input variables	Flowchart ⟨6⟩	Processing and remarks	Output variable
	MKSCAN		
	RIPCODL ← FH RIPCODH ← FH	Resets input code.	
	RREG0 ← 7H	Sets initial key scan signal value in RREG0.	
	1 RREG2 ← 0H	Resets keying counter.	
	P0B ← RREG0	Outputs key scan signals on P0B.	
	NOP 3 times	Waits 12 μs.	
	RREG9 ← P0A	Reads key return value.	
	P0B ← FH	Stops key scan signal output on P0B.	
	SKRET ⟨10⟩	Key return value decision	
FMULTI (0.69H.1)	FMULTI = 1 ? N → ⟨7⟩ 1 Y → ⟨8⟩ 1	Multiple-key pressing?	

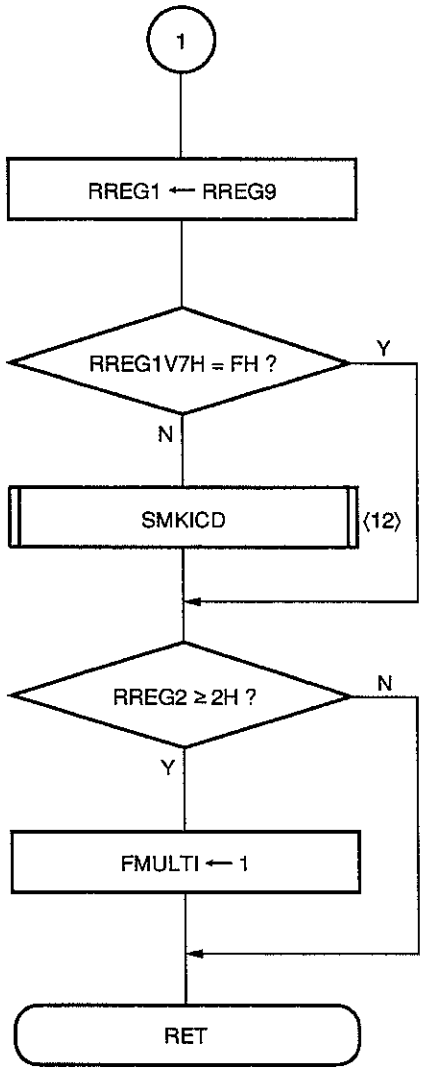
Input variables	Flowchart ⟨7⟩	Processing and remarks	Output variable
	1		
	CY ← 1	Key scan signal output on P0B completed?	
	Shift RREG0 and CY data 1 bit right		
	CY = 0 ? (N → ⟨6⟩ 1; Y)		
	RREG0 ← 0H P0C ← RREG0	Outputs key scan signal on P0C.	
	NOP 3 times	Waits 12 μs.	
	RREG9 ← P0A	Reads key return value.	
	P0C ← 1H	Stops key scan signal output on P0C.	
	SKRET ⟨10⟩	Key return value decision	
FMULTI (0.69H.1)	FMULTI = 1 ? (N → ⟨8⟩ 2; Y → ⟨8⟩ 1)	Multiple-key pressing?	

Input variables	Flowchart ⟨8⟩	Processing and remarks	Output variable
		Clears multiple-key flag. Sets multiple-key code as input code. Key state unchanged? Updates chattering code. Resets chattering counter. Chattering processing completed? Counts with chattering counter. Chattering processing completed?	FMULTI (0.69H.1)

Input variables	Flowchart ⟨9⟩	Processing and remarks	Output variable
		Multiple-key pressing? Key-off? Previous key code being key-off? Sets key-processing request flag. Sets chattering code as key code.	FKEYREQ (0.69H.0) RKCODH (0.30H) RKCODL (0.31H)

6.4 Key Return Value Decision Processing

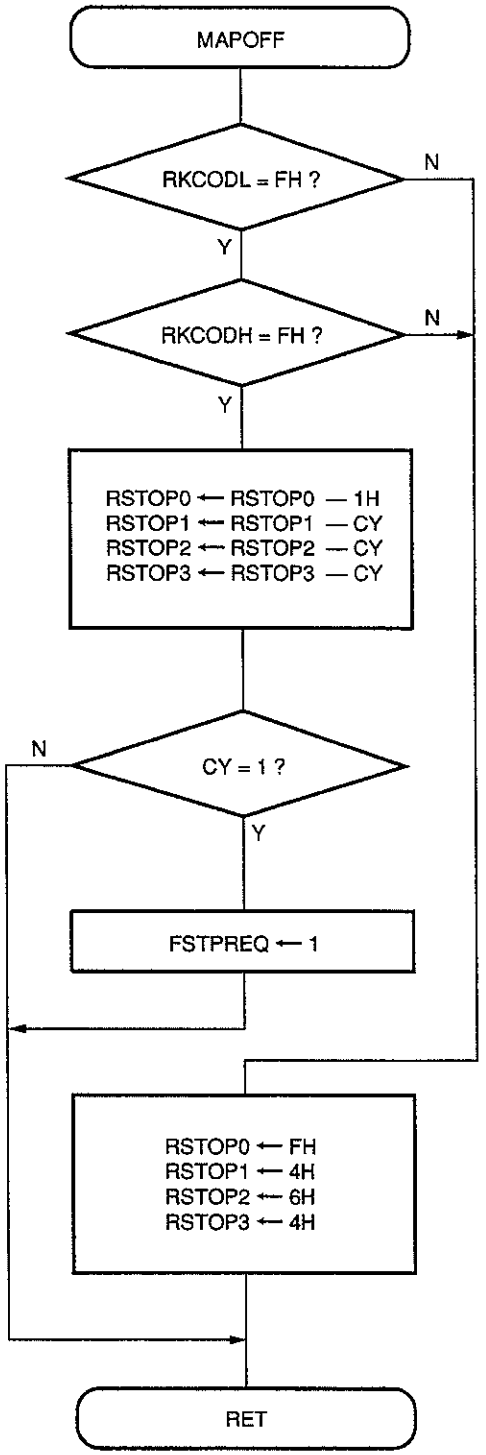
Input variables	Flowchart ⟨10⟩	Processing and remarks	Output variable
	SKRET		
RREG9 (0.09H)	RREG1 ← RREG9	Key return signal on $P0A_0$ high?	
	RREG1VEH = FH ? (Y / N)		
	SMKICD ⟨12⟩	Generates input code.	
RREG9 (0.09H)	RREG1 ← RREG9	Key return signal on $P0A_1$ high?	
	RREG1VDH = FH ? (Y / N)		
	SMKICD ⟨12⟩	Generates input code.	
RREG9 (0.09H)	RREG1 ← RREG9	Key return signal on $P0A_2$ high?	
	RREG1V7H = FH ? (Y / N)		
	SMKICD ⟨12⟩	Generates input code.	
	⟨11⟩ 1		

Input variables	Flowchart ⟨11⟩	Processing and remarks	Output variable
RREG9 (0.09H)		Key return signal on $P0A_3$ high? Generates input code. Multiple-key pressing? Sets multiple-key flag.	FMULTI (0.69H.1)

6.5 Input Code Generation Processing

Input variables	Flowchart ⟨12⟩	Processing and remarks	Output variable
RREG0 (0.00H) RREG9 (0.09H)	SMKICD		
	RIPCODH ← RREG0 RIPCODL ← RREG9	Generates input code.	RIPCODH RIPCODL (0.34H-0.35H)
	RREG2 ← RREG2 + 1H	Counts the number of keys.	RREG2 (0.02H)
	RET		

6.6 Automatic-Power-off Timer Control

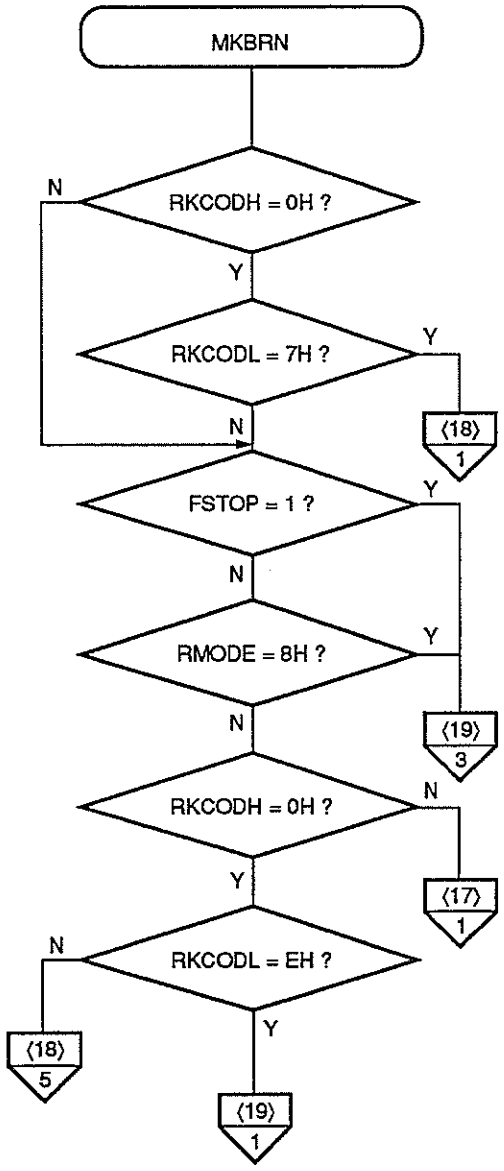
Input variables	Flowchart ⟨13⟩	Processing and remarks	Output variable
RKCODL (0.31H) RKCODL (0.30H)		Key-off currently? Counts using automatic-power-off timer. Timeout? Sets STOP mode processing request flag. Resets automatic-power-off timer.	

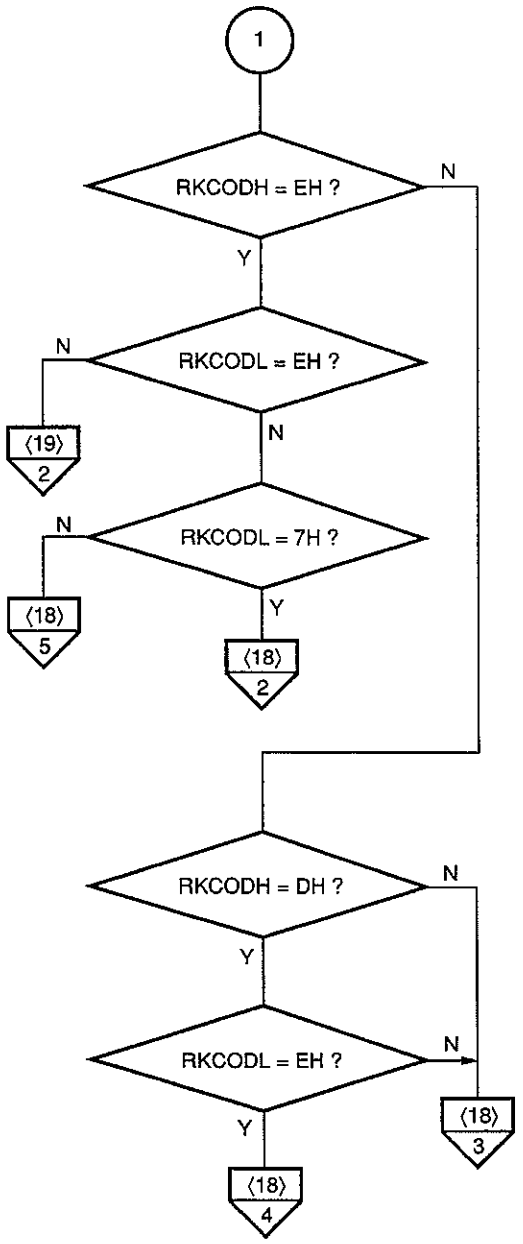
6.7 STOP Mode Processing

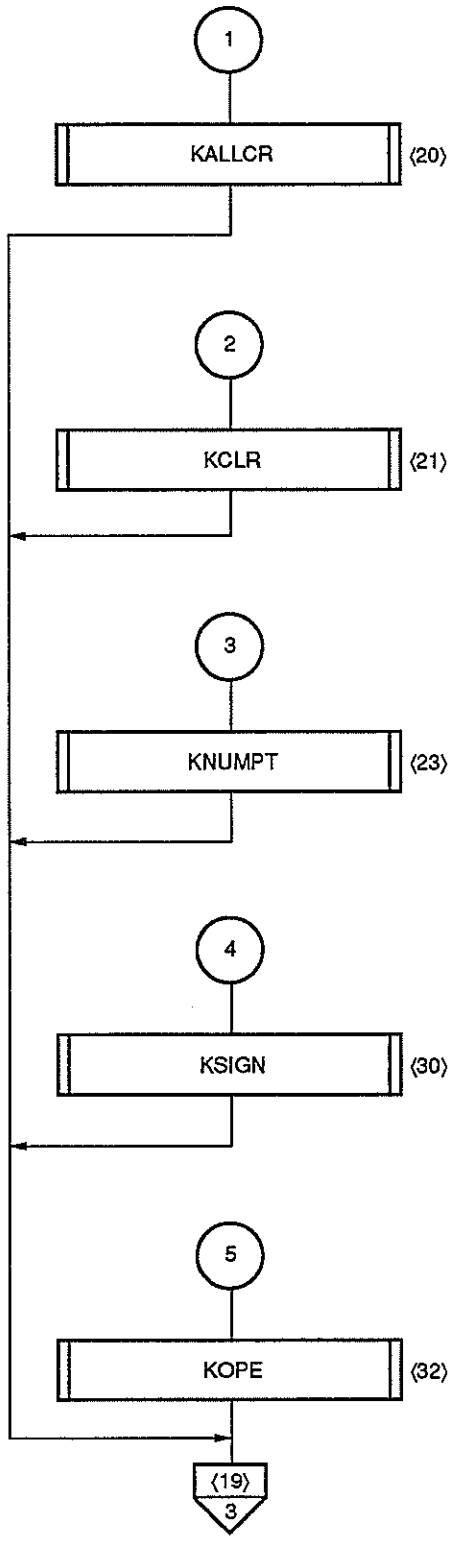
Input variables	Flowchart ⟨14⟩	Processing and remarks	Output variable
	MSTOP		
	FSTPREQ ← 1	Clears STOP mode processing request flag.	FSTPREQ (0.69H.3)
	LCDEN ← 0	Turns off LCD display.	
	TMEN ← 0	Stops 8-bit timer operation.	
	FSTOP ← 1	Sets operation stop mode flag.	FSTOP (0.68H.3)
	P0B ← 0H P0C ← 0H	Outputs key scan signals.	
	TMM ← 00H	Sets wait value for STOP mode release in 8-bit timer modulo register.	
	NOP		
	STOP 8H	Sets STOP mode (released by key input).	
	⟨15⟩ 1		

Input variables	Flowchart (15)	Processing and remarks	Output variable
	1		
	TMEN ← 0	Stops 8-bit timer operation.	
	P0B ← FH P0C ← 1H	Stops key scan signal output.	
	TMM ← 9CH	Resets 8-bit timer modulo register.	
	TMEN ← 1	Starts 8-bit timer operation.	
	RET		

6.8 Branching to Each Key Processing

Input variables	Flowchart (16)	Processing and remarks	Output variable
RKCODH (0.30H) RKCODL (0.31H)		AC key pressed?	
FSTOP (0.68H.3)		Operation stop mode?	
RMODE (0.64H)		Error mode?	
RKCODH (0.30H) RKCODL (0.31H)		% key pressed?	

Input variables	Flowchart ⟨17⟩	Processing and remarks	Output variable
RKCODH (0.30H) RKCODL (0.31H)		[=] key pressed?	
RKCODL (0.31H)		[C] key pressed?	
RKCODH (0.30H) RKCODL (0.31H)		[±] key pressed?	

Input variables	Flowchart ⟨18⟩	Processing and remarks	Output variable
			

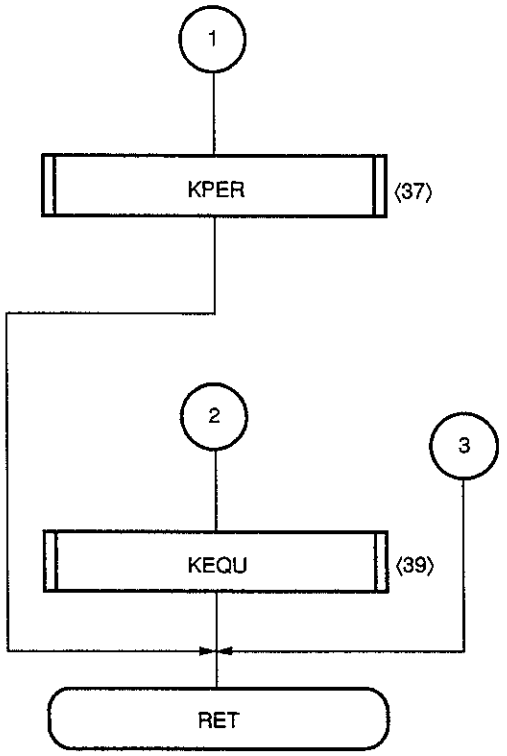
All clear key processing

Clear key processing

Numeric key and decimal-point key processing

Plus/minus key processing

Operator key processing

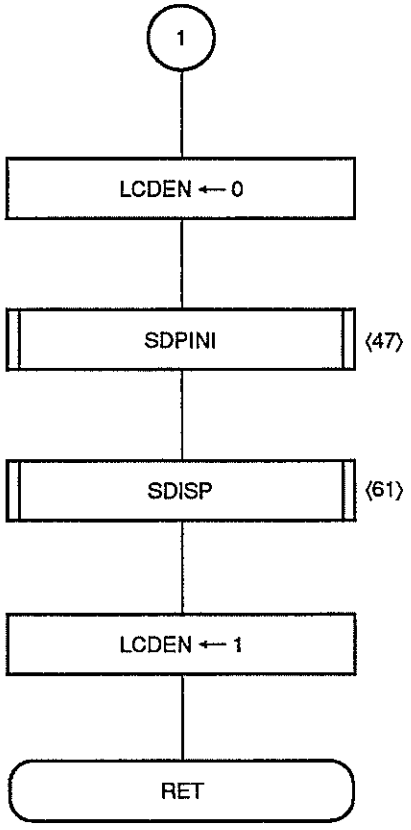
Input variables	Flowchart ⟨19⟩	Processing and remarks	Output variable
		Percent key processing Equal key processing	

6.9 All Clear Key Processing

Input variables	Flowchart ⟨20⟩	Processing and remarks	Output variable
	KALLCR		
	LCDEN ← 0	Turns off LCD display.	
	SRAMCR ⟨45⟩	Clears RAM.	
	RDEXP ← 8H	Resets display data exponent area.	RDEXP (0.14H)
	RKCODL ← 7H RKCODH ← 0H RCHCODL ← 7H RKCODH ← 0H	Restores AC key code as key code and chattering code.	
	RMODE ← 4H	Sets first-term input mode.	RMODE (0.64H)
	SDPINI ⟨47⟩	Initializes display data area.	
	SDISP ⟨61⟩	Converts display data to LCD segment data.	
	LCDEN ← 1	Turns on LCD display.	
	RET		

6.10 Clear Key Processing

Input variables	Flowchart ⟨21⟩	Processing and remarks	Output variable
	KCLR		
FPER (0.68H.0)	FPER = 1 ? (N / Y)	% key pressed?	
	FPER ← 0	Clears percent flag.	FPER (0.68H.0)
	RMODE ← 4H	Sets first-term input mode.	RMODE (0.64H)
	FOPEND ← 0 FFALSE ← 0	Clears operation end flag and illegal-input flag.	FOPEND (0.68H.1) FFALSE (0.68H.2)
	RNUMC ← 0H	Resets numeric key counter.	RNUMC (0.10H)
	RDEXP ← 8H	Resets display data exponent area.	RDEXP (0.14H)
RMODE (0.64H)	RMODE = 2H ? (N / Y)	Operator selection mode?	
	RMODE ← 1H	Sets second-term input mode.	RMODE (0.64H)
	⟨22⟩ 1		

Input variables	Flowchart ⟨22⟩	Processing and remarks	Output variable
		Turns off LCD display. Initializes display data area. Converts display data to LCD segment data. Turns on LCD display.	

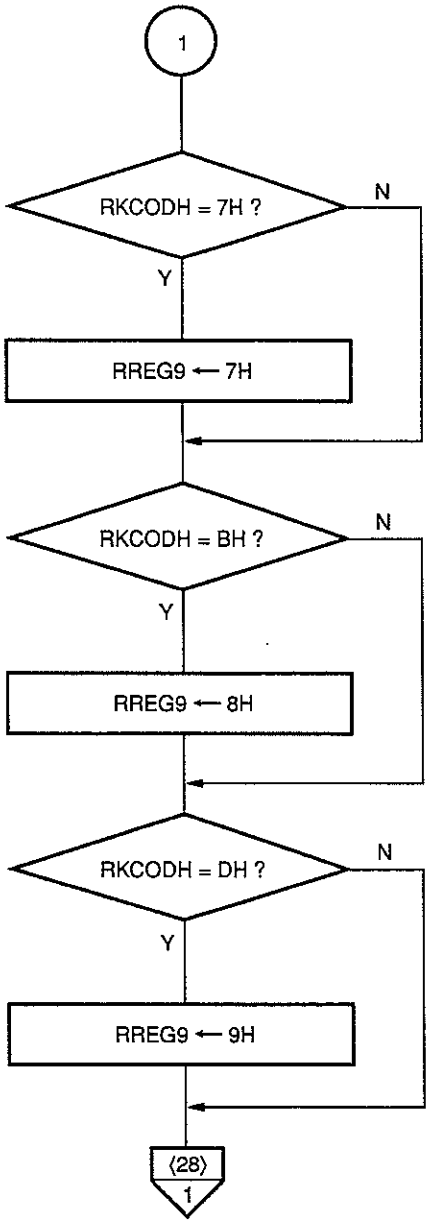
6.11 Numeric Key and Decimal-Point Key Processing

Input variables	Flowchart ⟨23⟩	Processing and remarks	Output variable
	KNUMPT		
FFALSE (0.68H.2)	FFALSE = 1 ? (N / Y)	% key or = key pressed incorrectly?	
	FFLASE ← 0	Clears illegal-input flag.	FFALSE (0.68H.2)
	RNUMC ← 0H	Resets numeric key counter.	RNUMC (0.10H)
	RDEXP ← 8H	Resets display data exponent area.	RDEXP (0.14H)
FPER (0.68H.0)	FPER = 1 ? (N / Y)	% key pressed?	
	RMODE ← 4H	Sets first-term input mode.	RMODE (0.64H)
	FPER ← 0 FOPEND ← 0	Clears percent flag and operation end flag.	FPER (0.68H.0) FOPEND (0.68H.1)
RKCODH (0.30H)	RKCODH = BH ? (N / Y)	· key pressed?	
RKCODL (0.31H)	RKCODL = EH ? (N / Y)		
	⟨25⟩ 1		
	⟨24⟩ 1		

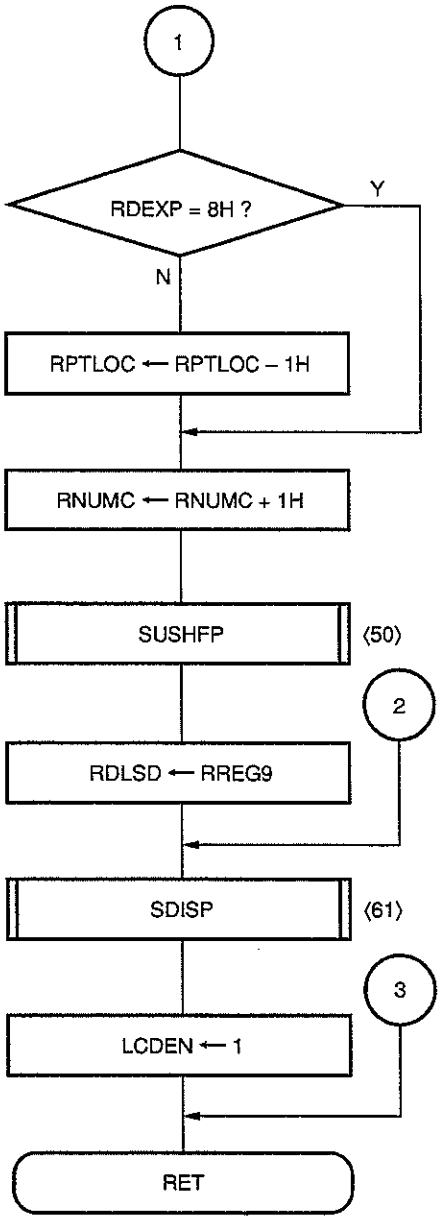
Input variables	Flowchart ⟨24⟩	Processing and remarks	Output variable
	1		
RDEXP (0.14H)	RDEXP = 8H ? (N / Y)	Decimal-point key pressed for the first time?	
RNUMC (0.10H)	RNUMC = 0H ? (N / Y)	Decimal-point key pressed before numeric key is pressed?	
	RNUMC ← RNUMC + 1H	Increments numeric key counter.	RNUMC (0.10H)
	LCDEN ← 0	Turns off LCD display.	
	SDPINI ⟨47⟩	Initializes display data area.	
	SDISP ⟨61⟩	Converts display data to LCD segment data.	
	LCDEN ← 1	Turns on LCD display.	
RNUMC (0.10H)	RDEXP ← RNUMC	Sets value of numeric key counter in display data exponent area.	RDEXP (0.14H)
RMODE (0.64H)	RMODE = 2H ? (N / Y)	Operator selection mode?	
	RMODE ← 1H	Sets second-term input mode.	RMODE (0.64H)
	⟨29⟩ 3		

Input variables	Flowchart ⟨25⟩	Processing and remarks	Output variable
RNUMC (0.10H)	1 → RNUMC = 7H ? → ⟨29⟩ 3	Numeric keys already pressed seven times?	
RKCODL (0.31H)	RKCODL = EH ? (N / Y)	0 key pressed?	
	RREG9 ← 0H → ⟨28⟩ 1	Sets display data "0" in RREG9.	
RKCODL (0.31H)	RKCODL = DH ? (N → ⟨26⟩ 2 / Y)	1 key pressed?	
RKCODH (0.30H)	RKCODH = 7H ? (N / Y)		
	RREG9 ← 1H	Sets display data "1" in RREG9.	
RKCODH (0.30H)	RKCODH = BH ? (N / Y)	2 key pressed?	
	RREG9 ← 2H → ⟨26⟩ 1	Sets display data "2" in RREG9.	

Input variables	Flowchart ⟨26⟩	Processing and remarks	Output variable
RKCODH (0.30H)	1 → RKCODH = DH ? (N / Y) → RREG9 ← 3H	3 key pressed?	
		Sets display data "3" in RREG9.	
RKCODL (0.31H)	2 → RKCODL = BH ? (N → ⟨27⟩ 1 / Y)	4 key pressed?	
RKCODH (0.30H)	RKCODH = 7H ? (N / Y) → RREG9 ← 4H		
		Sets display data "4" in RREG9.	
RKCODH (0.30H)	RKCODH = BH ? (N / Y) → RREG9 ← 5H	5 key pressed?	
		Sets display data "5" in RREG9.	
RKCODH (0.30H)	RKCODH = DH ? (N / Y) → RREG9 ← 6H → ⟨28⟩ 1	6 key pressed?	
		Sets display data "6" in RREG9.	

Input variables	Flowchart ⟨27⟩	Processing and remarks	Output variable
RKCODH (0.30H)		7 key pressed?	
		Sets display data "7" in RREG9.	
RKCODH (0.30H)		8 key pressed?	
		Sets display data "8" in RREG9.	
RKCODH (0.30H)		9 key pressed?	
		Sets display data "9" in RREG9.	

Input variables	Flowchart ⟨28⟩	Processing and remarks	Output variable
	1		
	LCDEN ← 0	Turns off LCD display.	
RMODE (0.64H)	RMODE = 4H ? (Y / N)	First-term input mode?	
	RMODE ← 1H	Sets second-term input mode.	RMODE (0.64H)
RNUMC (0.10H)	RNUMC = 0H ? (N / Y)	First numeric key input?	
	SDPINI ⟨47⟩	Initializes display data area.	
	RREG9 = 0H ? (Y → ⟨29⟩ 2 / N)	0 key pressed?	
	RDLSD ← AH	Sets display data of space in least significant digit of display data area.	RDLSD (0.18H)
	⟨29⟩ 1		

Input variables	Flowchart ⟨29⟩	Processing and remarks	Output variable
RDEXP (0.14H)		Decimal-point key pressed? Decrements decimal-point display position data by 1. Increments numeric key counter. Shifts display data 1 byte left. Sets new display data in least significant digit of display data area. Converts display data to LCD segment data. Turns on LCD display.	 RPTLOC (0.11H) RNUMC (0.10H) RDLSD (0.18H)

6.12 Plus/Minus Key Processing

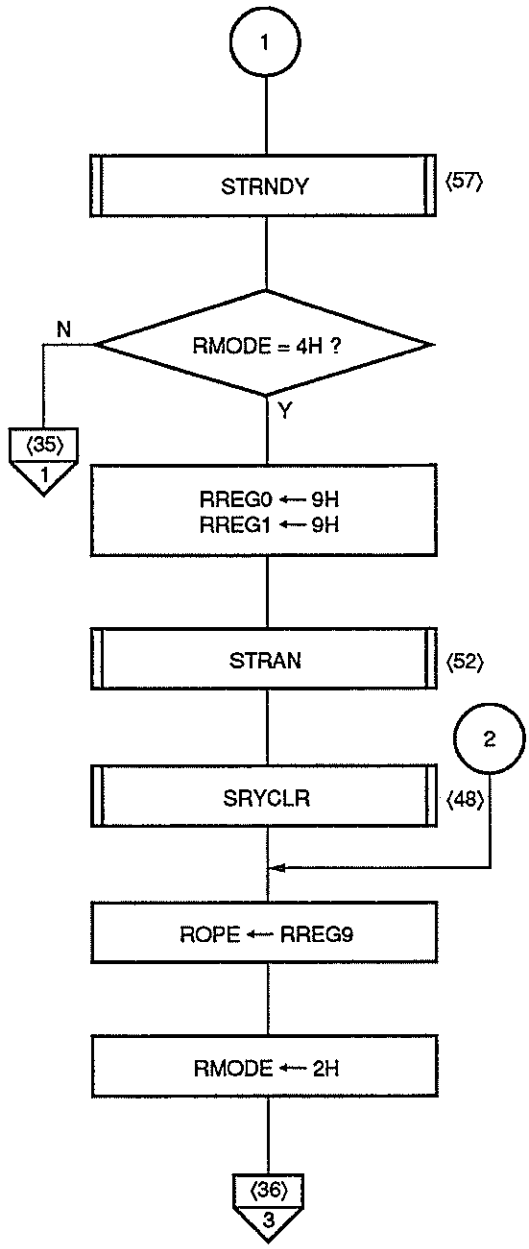
Input variables	Flowchart (30)	Processing and remarks	Output variable
	KSIGN		
RMODE (0.64H)	RMODE = 2H ? (Y / N)	Operator selection mode?	
FPER (0.68H.0)	FPER = 1 ? (N / Y)	% key pressed?	
	FPER ← 0	Clears percent flag.	FPER (0.68H.0)
	FOPEND ← 1	Sets operation end flag.	FOPEND (0.68H.1)
	RMODE ← 4H	Sets first-term input mode.	RMODE (0.64H)
FOPEND (0.68H.1)	FOPEND = 1 ? (Y / N)	Operation completed?	
	(31) 1		
RNUMC (0.10H)	RNUMC = 0H ? (Y / N)	Value of numeric key counter being 0?	
	(31) 3		
	RREG0 ← RNUMC + 8H RSINLOC ← RREG0	Finds sign display position, and sets that position in RREG0 and RSINLOC.	RSINLOC (0.12H)
	(31) 2		

Input variables	Flowchart ⟨31⟩	Processing and remarks	Output variable
FZERO (1.30H.1)	1 → FZERO = 1 ?	Operation result being 0?	
	RXSIGN ← RXSIGN⊕1H	Inverts sign data of REGX.	RXSIGN (1.03H)
RSINLOC (0.12H)	2 → RREG0 ← RSINLOC	Sets sign display position in RREG0.	
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	LCDEN ← 0	Turns off LCD display.	
	MPH ← 8H MPL ← 1H	Inverts sign of display data.	REGD [RRGEO]
	[(MP) , (RREG0)] ← [(MP) , (RREG0)] ⊕ 1H		
	MPH ← RREG10 MPL ← RREG11	Restores value of MP.	
	SDISP ⟨61⟩	Converts display data to LCD segment data.	
	3 → LCDEN ← 1	Turns on LCD display.	
	RET		

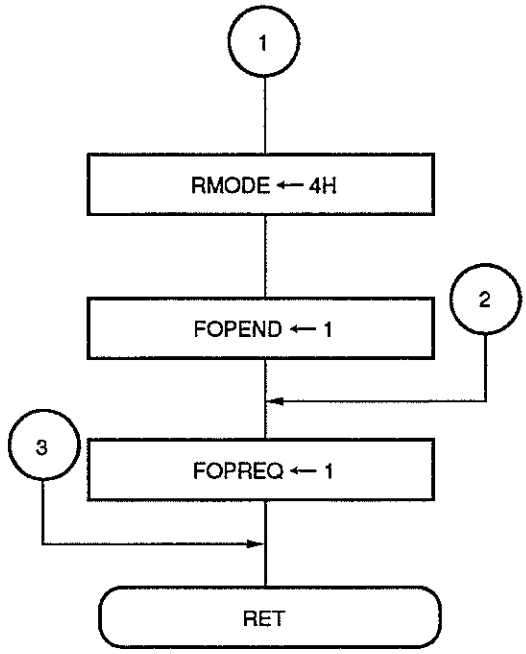
6.13 Operator Key Processing

Input variables	Flowchart ⟨32⟩	Processing and remarks	Output variable
	KOPE		
RKCODL (0.31H)	RKCODL = DH ? (N / Y)		
RKCODH (0.30H)	RKCODH = EH ? (N / Y)	⊞ key pressed?	
	RREG9 ← 1H	Sets addition operator data in RREG9.	
RKCODH (0.30H)	RKCODH = 0H ? (N / Y)	⊟ key pressed?	
	RREG9 ← 2H	Sets subtraction operator data in RREG9.	
RKCODH (0.30H)	RKCODH = EH ? (N / Y)	⊠ key pressed?	
	RREG9 ← 4H	Sets multiplication operator data in RREG9.	
RKCODH (0.30H)	RKCODH = 0H ? (N / Y)	⊡ key pressed?	
	RREG9 ← 8H	Sets division operator data in RREG9.	
	⟨33⟩ 1		

Input variables	Flowchart ⟨33⟩	Processing and remarks	Output variable
	1		
	FFALSE ← 0	Clears illegal-input flag.	FFALSE (0.68H.2)
RMODE (0.64H)	RMODE = 2H ? (Y / N)	Operator selection mode?	
FOPEND (0.68H.1)	FOPEND = 1 ? (N / Y)	Operation completed?	
	FOPEND ← 0	Clears operation end flag.	FOPEND (0.68H.1)
FPER (0.68H.0)	FPER = 1 ? (N → ⟨34⟩ 1 / Y)	% key pressed?	
	FPER ← 0	Clears percent flag.	FPER (0.68H.0)
	RREG9 < 4H ? (N → ⟨34⟩ 2 / Y → ⟨35⟩ 2)	+ key or − key pressed?	

Input variables	Flowchart ⟨34⟩	Processing and remarks	Output variable
RMODE (0.64H)		Normalizes display data, then transfers normalized display data to REGY. First-term input mode? Transfers data of REGY to REGX. Clears REGY. Sets operator data of RREG9 in operator area. Sets operator selection mode.	ROPE (0.60H) RMODE (0.64H)

Input variables	Flowchart ⟨35⟩	Processing and remarks	Output variable
	1		
	RCOM ← ROPE ROPE ← RREG9	Sets operator area data in area for operator to be executed, then sets operator data of RREG9 in operator area.	RCOM (0.61H) ROPE (0.60H)
	RMODE ← 2H	Sets operator selection mode.	RMODE (0.64H)
	⟨36⟩ 2		
	2		
	ROPE ← RREG9 RCOM ← RREG9	Sets operator data of RREG9 in both operator area and area for operator to be executed.	ROPE (0.60H) RCOM (0.61H)
	RREG0 ← 8H RREG1 ← 9H	Transfers data of REGX to REGY.	
	STRAN ⟨52⟩		
	RREG0 ← 2H RREG1 ← 8H	Restores data of REGZ in REGX.	
	STRAN ⟨52⟩		
	⟨36⟩ 1		

Input variables	Flowchart ⟨36⟩	Processing and remarks	Output variable
		Sets first-term input mode. Sets operation end flag. Sets operation request flag.	RMODE (0.64H) FOPEND (0.68H.1) FOPREQ (0.69H.2)

6.14 Percent Key Processing

Input variables	Flowchart ⟨37⟩	Processing and remarks	Output variable
FPER (0.68H.0)	KPER FPER = 1 ? (Y / N)	% key already pressed?	
RMODE (0.64H)	RMODE = 2H ? (Y / N)	Operator selection mode?	
RMODE (0.64H)	RMODE = 1H ? (Y / N)	Second-term input mode?	
	FOPEND ← 0	Clears Operator end flag.	FOPEND (0.68H.1)
ROPE (0.60H)	ROPE ≥4H ? (Y → ⟨38⟩ 1 / N)	× key or ÷ key pressed?	
	RMODE ← 4H	Sets first-term input mode.	RMODE (0.64H)
	FFALSE ← 1	Sets illegal-input flag.	FFALSE (0.68H.2)
	⟨38⟩ 2		

Input variables	Flowchart (38)	Processing and remarks	Output variable
	1		
	RREG0 ← 8H RREG1 ← 2H	Saves data of REGX to REGZ.	
	STRAN (52)		
	STRNDY (57)	Normalizes display data, then transfers normalized display data to REGY.	
	RYEXP ← RYEXP − 2H (RYEXP + 1H) ← (RYEXP + 1H) − CY	Subtracts 2 from characteristic data of REGY.	RYEXP RYEXP + 1H (1.14H-1.15H)
	FPER ← 1	Sets percent flag.	FPER (0.68H.0)
ROPE (0.60H)	RCOM ← ROPE	Sets operator area data in area for operator to be executed.	RCOM (0.61H)
	FOPREQ ← 1 (2)	Sets operator request flag.	FOPREQ (0.69H.2)
	RET		

6.15 Equal Key Processing

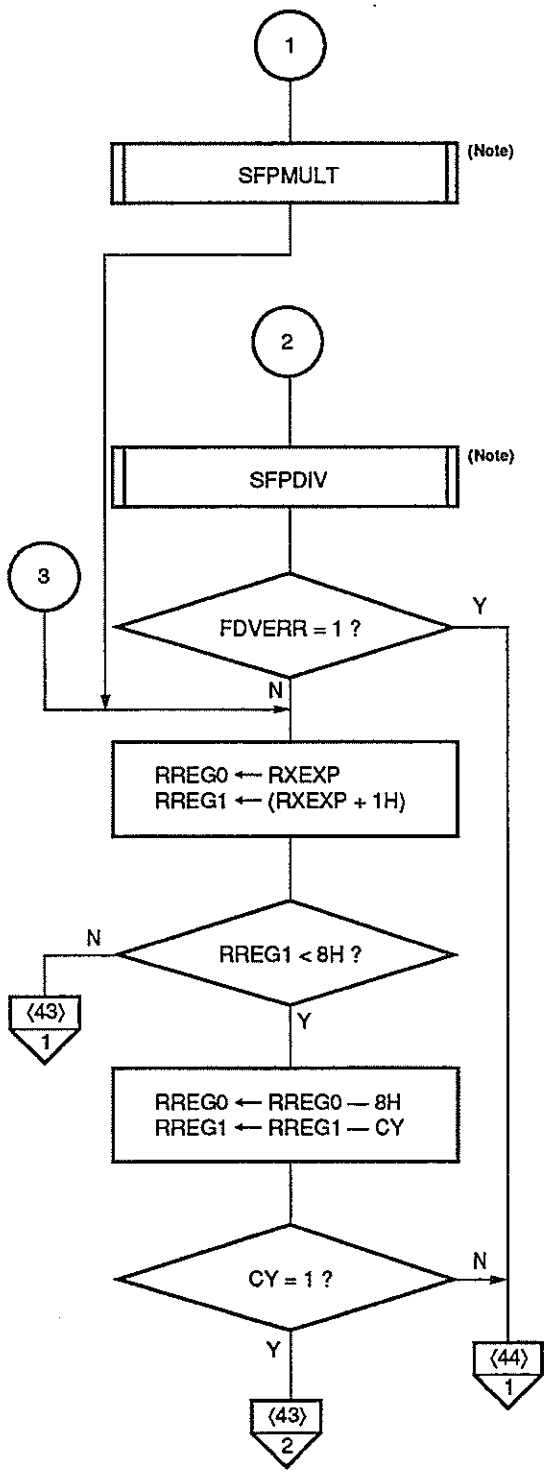
Input variables	Flowchart ⟨39⟩	Processing and remarks	Output variable
FPER (0.68H.0)	KEQU FPER = 1 ? (N / Y) FPER ← 0 ⟨40⟩ 4	% key pressed? Clears percent flag.	FPER (0.68H.0)
RMODE (0.64H)	RMODE = 4H ? (N / Y) FFALSE ← 1 ⟨40⟩ 5	First-term input mode? Sets illegal-input flag.	FFALSE (0.68H.2)
ROPE (0.60H)	RCOM ← ROPE	Sets operator area data in area for operator to be executed.	RCOM (0.61H)
RMODE (0.64H)	RMODE = 2H ? (N: ⟨40⟩ 3 / Y) RCOM < 4H ? (N: ⟨40⟩ 2 / Y) ⟨40⟩ 1	Operator input mode? + key or − key pressed?	

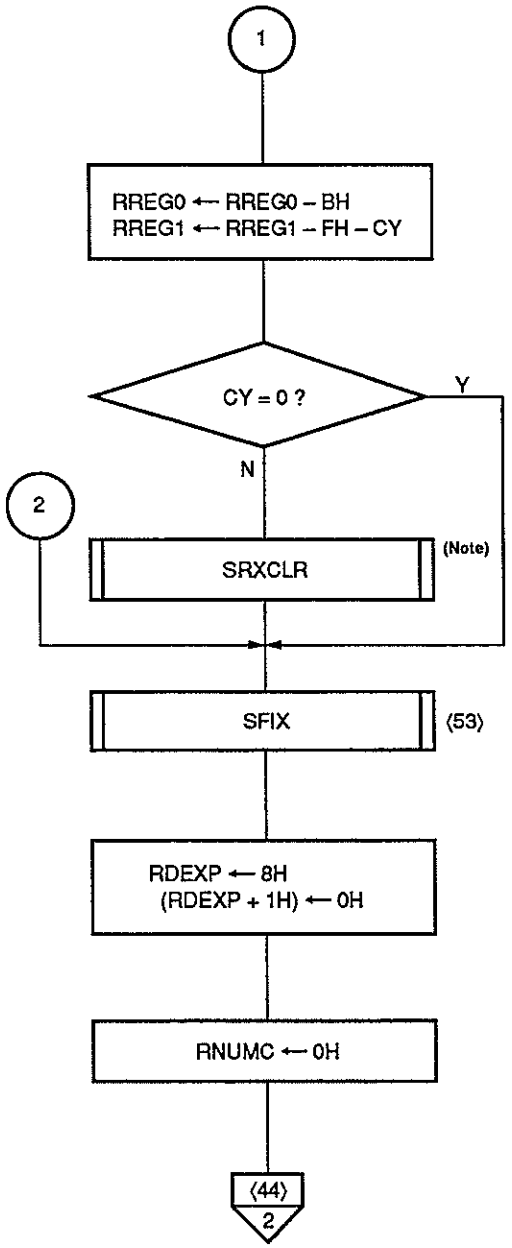
Input variables	Flowchart ⟨40⟩	Processing and remarks	Output variable
	1		
	SRYCLR ⟨48⟩	Clears REGY.	
	2		
	RREG0 ← 8H RREG1 ← 9H	Transfers data of REGX to REGY.	
	STRAN ⟨52⟩		
	3		
	STRNDY ⟨57⟩	Normalizes display data, then transfers normalized display data to REGY.	
	4		
	FOPREQ ← 1	Sets operation request flag.	FOPREQ (0.69H.2)
	FOPEND ← 1	Sets operation end flag.	FOPEND (0.68H.1)
	5		
	RMODE ← 4H	Sets first-term input mode.	RMODE (0.64H)
	RET		

6.16 Branch Operation Processing and Error Handling

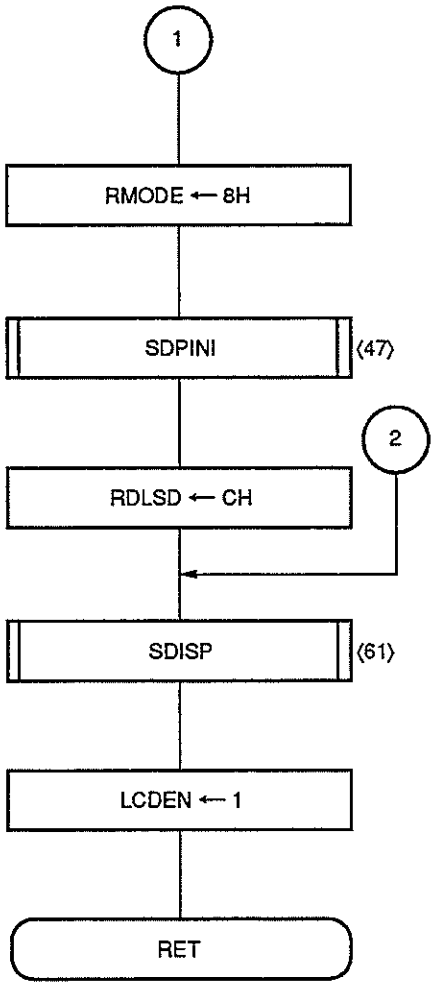
Input variables	Flowchart ⟨41⟩	Processing and remarks	Output variable
	MOPBRN		
	FOPREQ ← 0	Clears operation request flag.	FOPREQ (0.69H.2)
	LCDEN ← 0	Turns off LCD display.	
	ROPFLG ← 0H	Clears all operation flags.	ROPFLG (1.30H)
RCOM (0.61H)	RCOM = 8H ? Y → ⟨42⟩ 2	Divide instruction?	
	N		
RCOM (0.61H)	RCOM = 4H ? Y → ⟨42⟩ 1	Multiply instruction?	
	N		
RCOM (0.61H)	RCOM = 2H ? Y	Subtract instruction?	
	N		
	SFPADD (Note)	Addition	
	SFPSUB (Note)	Subtraction	
	⟨42⟩ 3		

(Note) For SFPADD and SFPSUB, see **Chapter 8.**

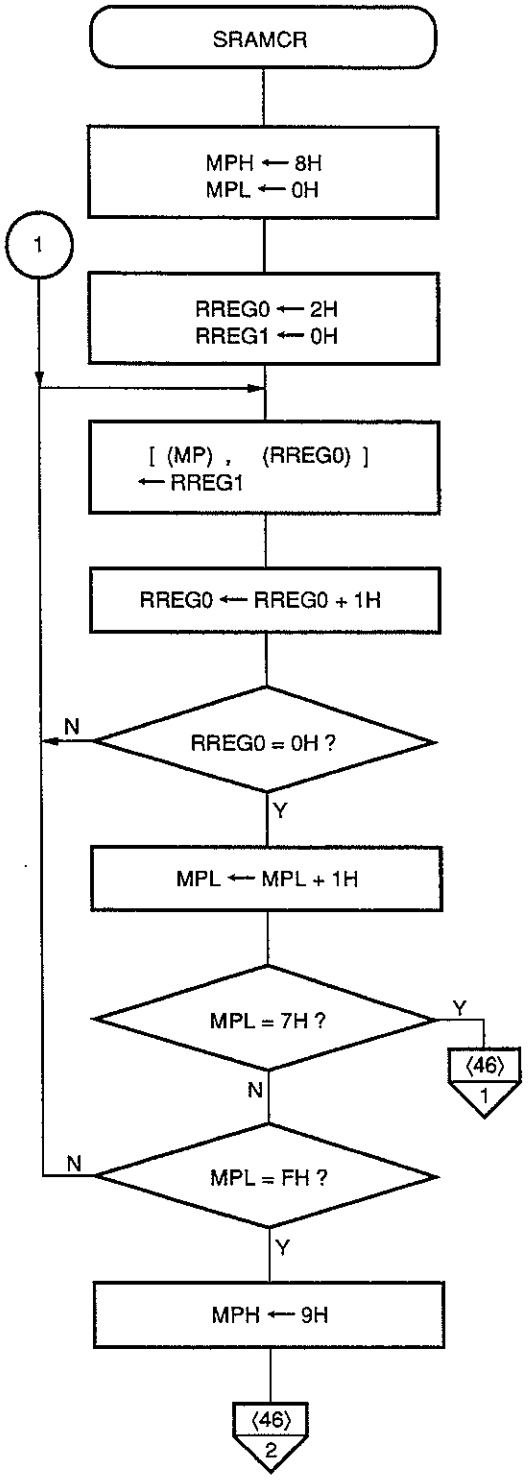
Input variables	Flowchart ⟨42⟩	Processing and remarks	Output variable
			
	(Note) For SFPMULT and SFPDIV, see **Chapter 8**.		

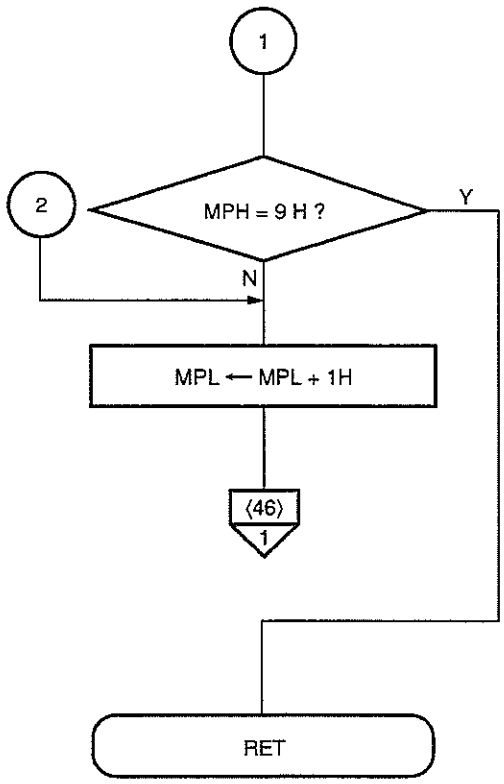
Input variables	Flowchart ⟨43⟩	Processing and remarks	Output variable
		Operation result without underflow (operation result represented by 6 or less decimal places)? Clears REGX. Converts result of operation to display data. Resets display data exponent area. Resets numeric key counter.	RXEXP RXEXP + 1H (0.14H-0.15H) RNUMC (0.10H)

(Note) For SRXCLR, see **Chapter 8.**

Input variables	Flowchart ⟨44⟩	Processing and remarks	Output variable
		Sets error mode. Initializes display data area. Sets error display data in least significant digit of display data area. Converts display data to LCD segment data. Turns on LCD display.	RMODE (0.64H) RDLSD (0.18H)

6.17 RAM All Clear Processing

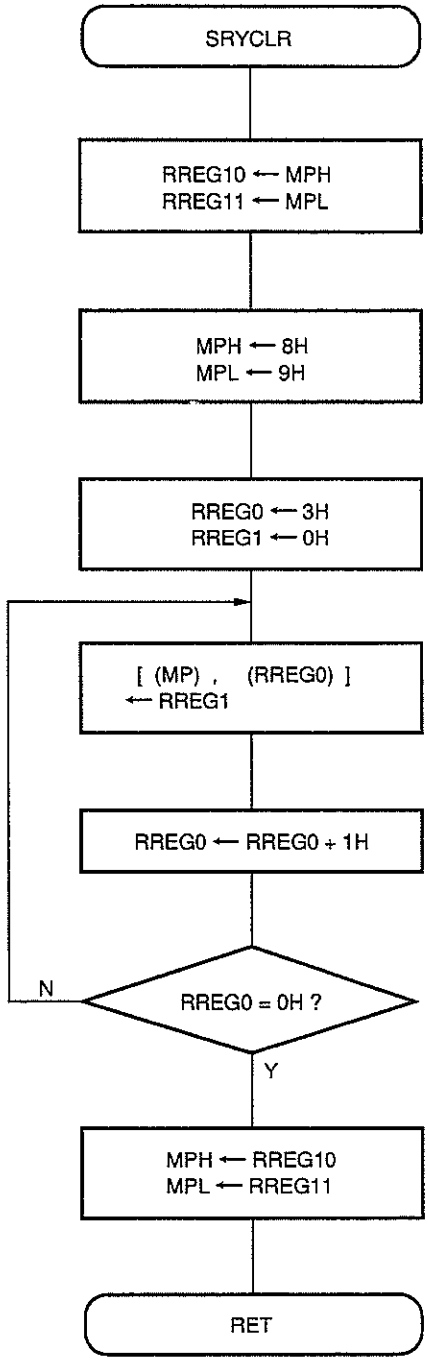
Input variables	Flowchart ⟨45⟩	Processing and remarks	Output variable
		Clears all RAM areas (except 0.00H and 0.01H) to 0.	MPL (0.7BH) MPH (0.7AH)

Input variables	Flowchart ⟨46⟩	Processing and remarks	Output variable
		Clears all RAM areas (except 0.00H and 0.01H) to 0.	

6.18 Display Data Area Initialization Processing

Input variables	Flowchart (47)	Processing and remarks	Output variable
	SDPINI		
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	RDLSD ← 0H	Sets 0 in least significant digit of display data area.	RDLSD (0.18H)
	MPH ← 8H MPL ← 1H		
	RREG0 ← 9H RREG1 ← AH		
	[(MP) , (RREG0)] ← RREG1	Sets data of space in all digits other than least significant digit of display data area.	RDLSD + 1H -RDMSD (0.19-0.1FH)
	RREG0 ← RREG0 + 1H		
	RREG0 = 0H ? (N / Y)		
	RPTLOC ← 7H	Resets decimal-point display position.	RPTLOC (0.11H)
	MPH ← RREG10 MPL ← RREG11	Restores value of MP.	
	RET		

6.19 REGY Clear Processing

Input variables	Flowchart ⟨48⟩	Processing and remarks	Output variable
		Saves value of MP. Clears REGY to 0. Restores value of MP.	REGY (1.13H-1.1FH)

6.20 Display Data Area Shifting Down

Input variables	Flowchart ⟨49⟩	Processing and remarks	Output variable
	SDSHFD		
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	MPH ← 8H MPL ← 1H		
	RREG0 ← 7H		(RDLSD – 2H) - (RDMSD – 1H) (0.16H-0.1EH)
	RREG1 ← [(MP) , (RREG0)]		
	RREG0 ← RREG0 – 1H	Shifts data of display data area 1 byte left.	
	[(MP) , (RREG0)] ← RREG1		
	RREG0 ← RREG0 + 2H		
	RREG0 = 0H ? (N / Y)		
	RDMSD ← 0H	Clears most significant digit of display data area to 0.	RDMSD (0.1FH)
	MPH ← RREG10 MPL ← RREG11	Restores value of MP.	
	RET		

6.21 Display Data Area Shifting Up

Input variables	Flowchart (50)	Processing and remarks	Output variable
	SUSHFD		
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	MPH ← 8H MPL ← 1H		
	RREG0 ← EH		(RDLSD + 1H) -RDMSD (0.19H-0.1FH)
	RREG1 ← [(MP) , (RREG0)]		
	RREG0 ← RREG0 + 1H	Shifts data of display data area 1 byte right.	
	[(MP) , (RREG0)] ← RREG1		
	RREG0 ← RREG0 – 2H		
	RREG0 < 8H ? (N / Y)		
	RDLSD ← 0H	Clears least significant digit of display data area to 0.	RDLSD (0.18H)
	MPH ← RREG10 MPL ← RREG11	Restores value of MP.	
	RET		

6.22 REGY Shifting Up

Input variables	Flowchart (51)	Processing and remarks	Output variable
	SUSHFY		
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	MPH ← 8H MPL ← 9H		
	RREG0 ← EH		(RYLSD + 1H) -RYMSD (1.17H-1.1FH)
	RREG1 ← [(MP) , (RREG0)]		
	RREG0 ← RREG0 + 1H	Shifts mantissa data of REGY 1 byte right.	
	[(MP) , (RREG0)] ← RREG1		
	RREG0 ← RREG0 – 2H		
	N ← RREG0 < 6H ? → Y		
	RYLSD ← 0H	Clears least significant digit of REGY to 0.	RYLSD (1.16H)
	MPH ← RREG10 MPL ← RREG11	Restores value of MP.	
	RET		

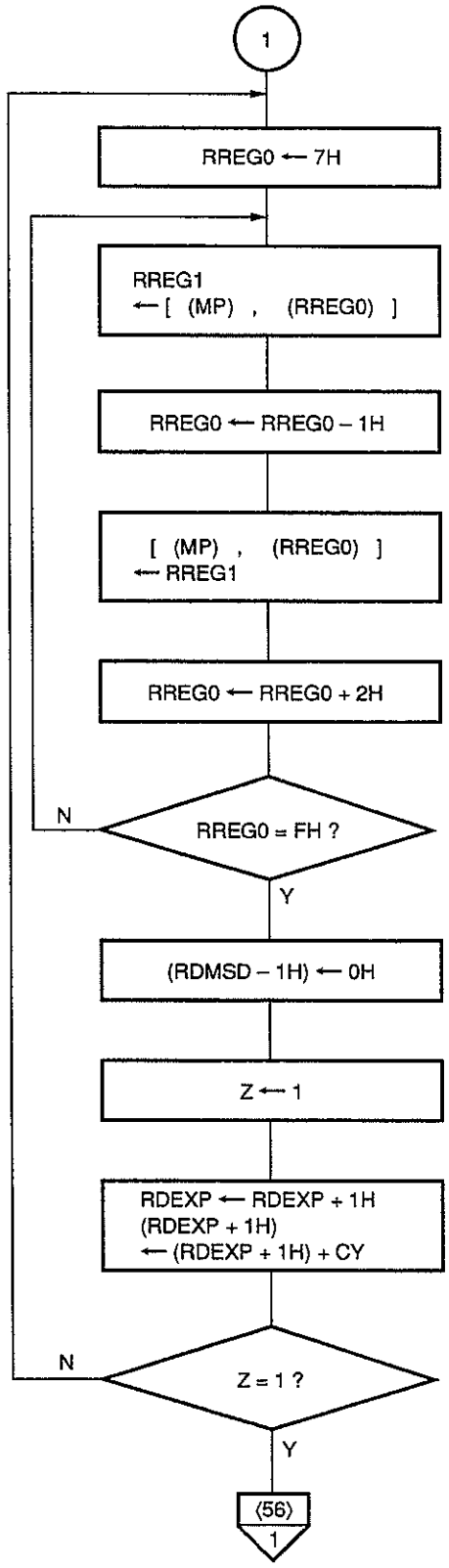
6.23 Data Transfer Processing

Input variables	Flowchart ⟨52⟩	Processing and remarks	Output variable
	STRAN		
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	MPH ← 8H		
RREG0 (0.00H)	RREG2 ← 3H	Transfers data among display data register, saving registers, and floating-point registers.	REGD (0.13H-0.1FH)
RREG1 (0.01H)	MPL ← RREG0		REGZ (0.23H-0.2FH)
REGZ (0.23H-0.2FH)	RREG3 ← [(MP) , (RREG2)]	RREG0: Row address of transfer source	REGX (1.03H-1.0FH)
REGX (1.03H-1.0FH)	MPL ← RREG1		REGY (1.13H-1.1FH)
REGY (1.13H-1.1FH)	[(MP) , (RREG2)] ← RREG3	RREG1: Row address of transfer destination	
	RREG2 ← RREG2 + 1H		
	N ← RREG2 = 0H ? → Y		
	MPH ← RREG10 MPL ← RREG11	Restores value of MP.	
	RET		

6.24 Operation Result Conversion Processing

Input variables	Flowchart ⟨53⟩	Processing and remarks	Output variable
	SFIX		
	RREG0 ← 8H RREG1 ← 1H	Transfers data of REGX to REGD.	
	STRAN ⟨52⟩		
RDEXP (0.14H)	RDEXP = 0H ? (N / Y)	Characteristic data of operation result 00H?	
RDEXP+1H (0.15H)	(RDEXP + 1H) = 0H ? (N / Y)		
	RDEXP ← 1H	Sets characteristic data to 1.	RDEXP (0.14H)
RDEXP+1H (0.15H)	(RDEXP + 1H) = FH ? (N / Y)	Characteristic data less than 0?	
	SDSHFD ⟨49⟩	Shifts data of display data area 1 byte left.	
	⟨54⟩ 1		

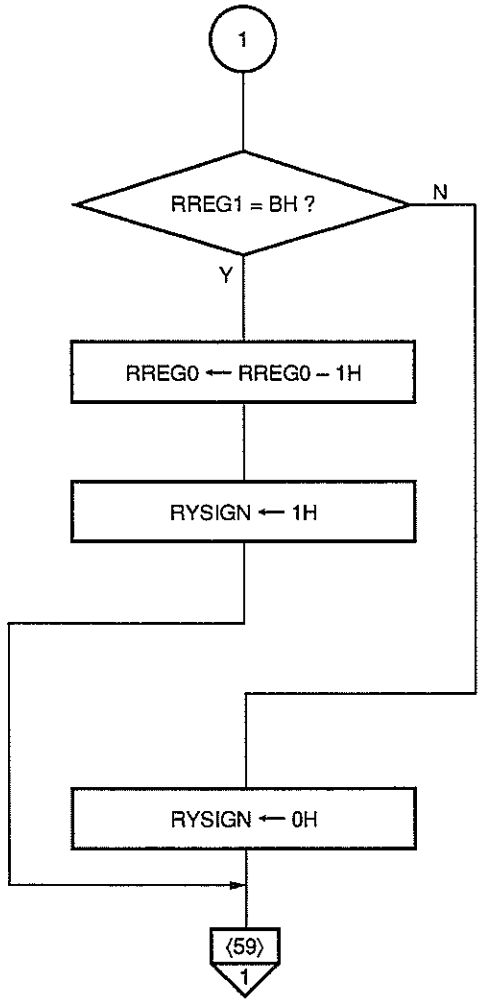
Input variables	Flowchart ⟨54⟩	Processing and remarks	Output variable
RDSIGN (0.13H)	1 → RDSIGN = 1H ? (N / Y)	Result of operation less than 0?	
	RDMSD ← BH	Sets display data "-" in most significant digit of display data area.	RDMSG (0.1FH)
	RDMSD ← AH	Sets display data of space in most significant digit of display data area.	RDMSG (0.1FH)
RDEXP+1H (0.15H)	(RDEXP + 1H) = FH ? (Y / N)	Characteristic data less than 0?	
	RPTLOC ← RDEXP → ⟨56⟩ 2	Sets low-order digits of characteristic data in decimal-point position area.	RPTLOC (0.11H)
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	MPH ← 8H MPL ← 1H → ⟨55⟩ 1	Sets MP.	

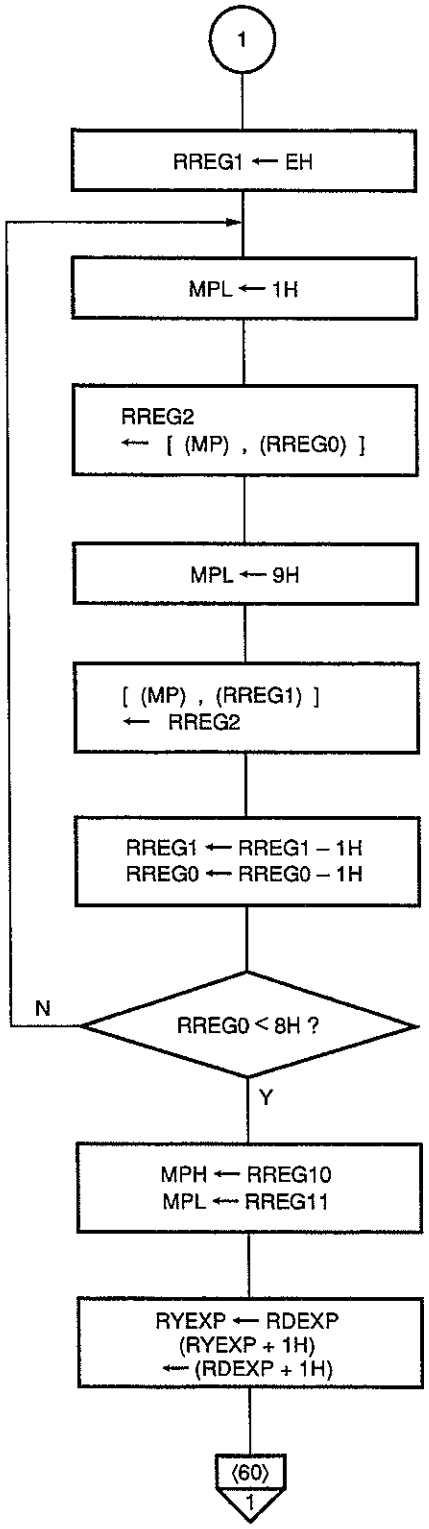
Input variables	Flowchart ⟨55⟩	Processing and remarks	Output variable
		Converts floating-point format data to fixed-point format data.	RDLSD - (RDMSD – 1H) (0.18H-0.1EH) RDEXP RDEXP + 1H (0.14H-0.15H)

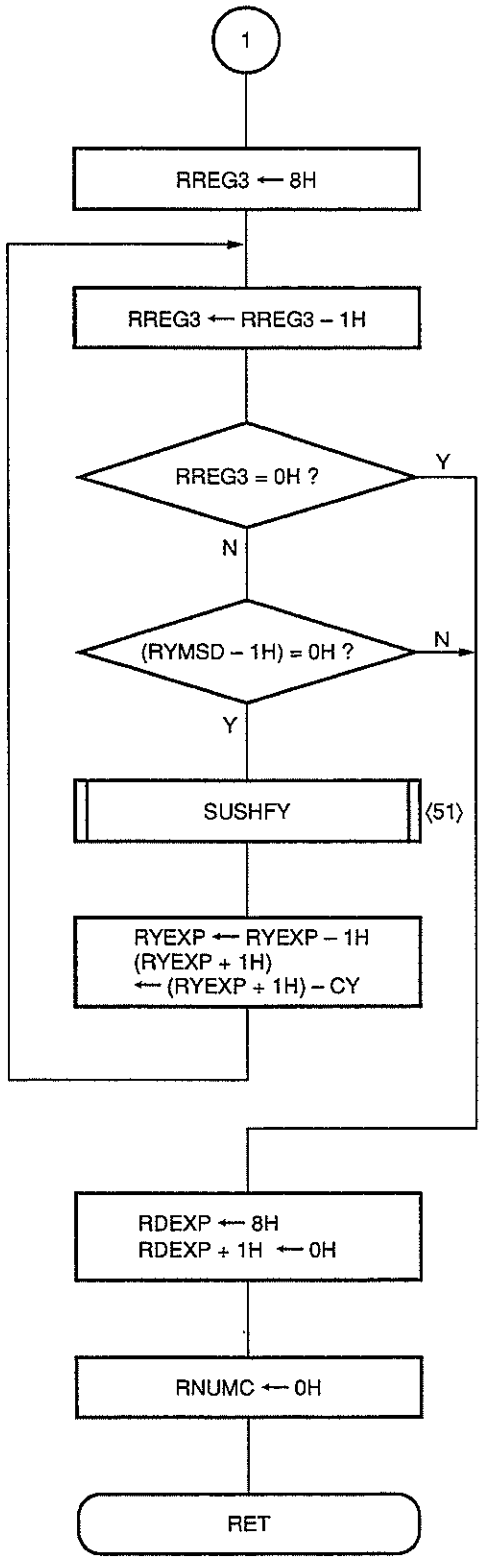
Input variables	Flowchart ⟨56⟩	Processing and remarks	Output variable
	1		
	MPH ← RREG10 MPL ← RREG11	Restores value of MP.	
	RDEXP ← 1H	Sets 1H in display data exponent area.	RDEXP (0.14H)
	RPTLOC ← 1H (2)	Sets 1H in decimal-point position area.	RPTLOC (0.11H)
	RSINLOC ← FH	Sets FH in sign display position area.	RSINLOC (0.12H)
	RDLSD = 0H ? (N / Y)	Display data zero suppression	
	RPTLOC = 7H ? (Y → RET / N)		
	RPTLOC ← RPTLOC + 1H		RPTLOC (0.11H)
	SDSHFD ⟨49⟩		
	RDMSD ← AH		RDMSD (0.1FH)
	RSINLOC ← RSINLOC − 1H		RSINLOC (0.12H)

6.25 Display Data Conversion Processing

Input variables	Flowchart ⟨57⟩	Processing and remarks	Output variable
	STRNDY		
	SRYCLR ⟨48⟩	Clears REGY.	
RDEXP (0.14H)	RDEXP = 8H ? (N / Y)	Decimal-point key pressed?	
RNUMC (0.10H)	RDEXP ← RNUMC	Sets value of numeric key counter in display data exponent area.	RDEXP (0.14H)
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	MPH ← 8H MPL ← 1H		
RDLSD -RDMSD (0.18H-0.1FH)	RREG0 ← FH		
	RREG1 ← [(MP) , (RREG0)]	Repeats checking of display data starting with most significant digit until data other than space is found.	
	RREG1 = AH ? (N → ⟨58⟩ 1 / Y)		
	RREG0 ← RREG0 — 1H		

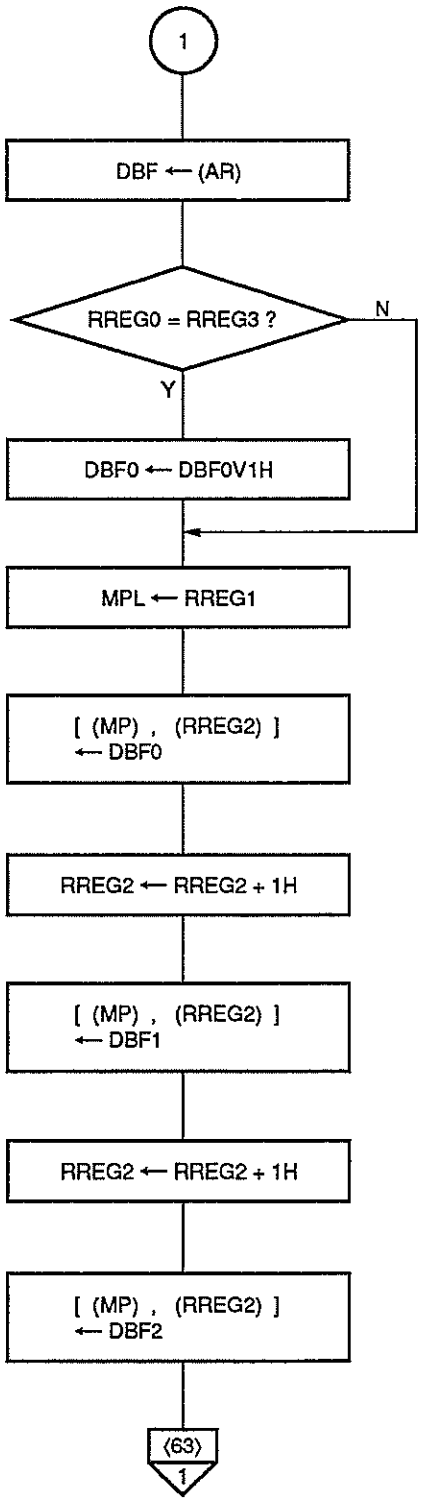
Input variables	Flowchart ⟨58⟩	Processing and remarks	Output variable
		Value of display data less than 0? Decrements address pointer of display data area by 1. Sets minus sign in sign area of REGY. Sets plus sign in sign area of REGY.	 RYSIGN (1.13H) RYSIGN (1.13H)

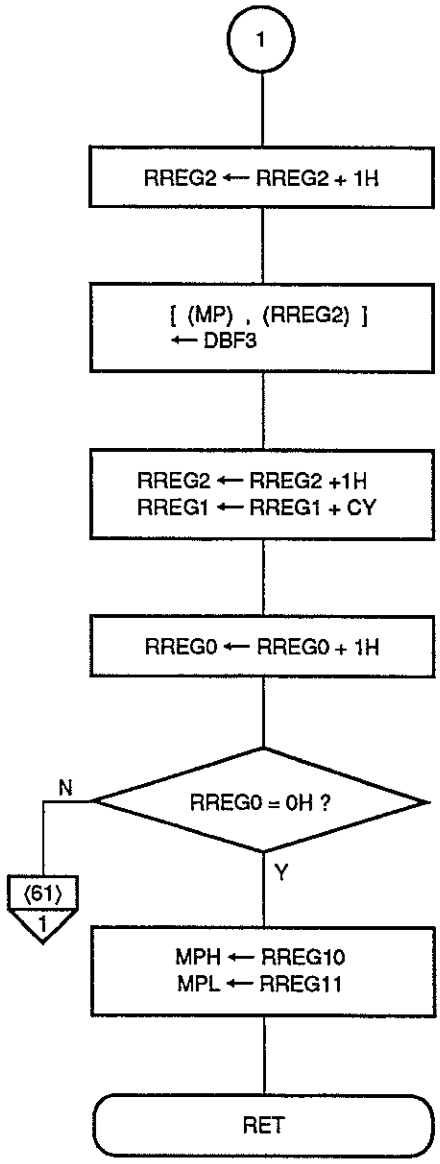
Input variables	Flowchart ⟨59⟩	Processing and remarks	Output variable
RDLSD -RDMSD (0.18H-0.1FH)		Transfers numeric data of display data area to mantissa of REGY. Restores MP data. Transfers data of display data exponent area to characteristic of REGY.	RYLSD -RYMSD (1.16H-1.1FH)
RDEXP RDEXP + 1H (0.14H-0.15H)			RYEXP RYEXP + 1H (1.14H-1.15H)

Input variables	Flowchart ⟨60⟩	Processing and remarks	Output variable
		REGY zero suppression Resets display data exponent area. Resets numeric key counter.	RYEXP RYEXP + 1H (1.14H-1.15H) RDEXP RDEXP + 1H (0.14H-0.15H)

6.26 Display Data Output Processing

Input variables	Flowchart (61)	Processing and remarks	Output variable
	SDISP		
	RREG10 ← MPH RREG11 ← MPL	Saves value of MP.	
	MPH ← 8H	Sets MPE.	
	RREG0 ← 8H RREG1 ← 4H RREG2 ← 0H	Sets start column address of display data area in RREG0, and sets start address of segment data area in RREG1-RREG2.	
RPTLOC (0.11H)	RREG3 ← FH – RPTLOC (1)	Sets decimal-point display position in RREG3.	
	AR ← TSEGDAT	Sets start address of segment data table in AR.	
	MPL ← 1H	Reads display data.	
RDLSD –RDMSD (0.18H-0.1FH)	RREG4 ← [(MP) , (RREG0)]		
	RREG4 = 0H ? Y → (62) 1 N → AR ← AR + 1H RREG4 ← RREG4 – 1H	Specifies address of segment data to be referenced, from read display data.	

Input variables	Flowchart ⟨62⟩	Processing and remarks	Output variable
		Reads segment data. Decimal-point display position? Converts segment data to data with decimal point. Outputs segment data to segment data area.	LCDD0 -LCDD31 (0.40H-0.5FH)

Input variables	Flowchart ⟨63⟩	Processing and remarks	Output variable
		Outputs segment data to segment data area. Segment data output completed? Restores value of MP.	

Chapter 7

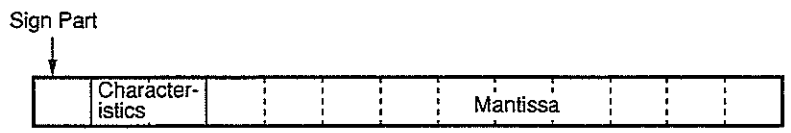
Floating-Point Format (Data Storage Format on RAM)

7.1 Numeric Control Format

The arithmetic package handles data to be operated, operation data, and operation results as floating-point decimals as shown in Figure 7-1.

For control, operation data and data to be operated are divided into three parts: a sign part (1 bit) for expressing the sign (plus or minus) of a numeric, a characteristic (8 bits) for storing a decimal-point position (n of 10^n), and a mantissa (40 bits) for expressing a decimal excluding a sign and decimal point.

Figure 7-1. Numeric Control Format



- Sign part : 1 bit (One nibble is allocated on RAM.)
- Characteristics (2 hexadecimal digits) : 2 by 4 bits
- Mantissa (10 decimal digits) : 10 by 4 bits, 1 integer digit
  9 decimal fraction digits

The numeric control format is expressed as follows:

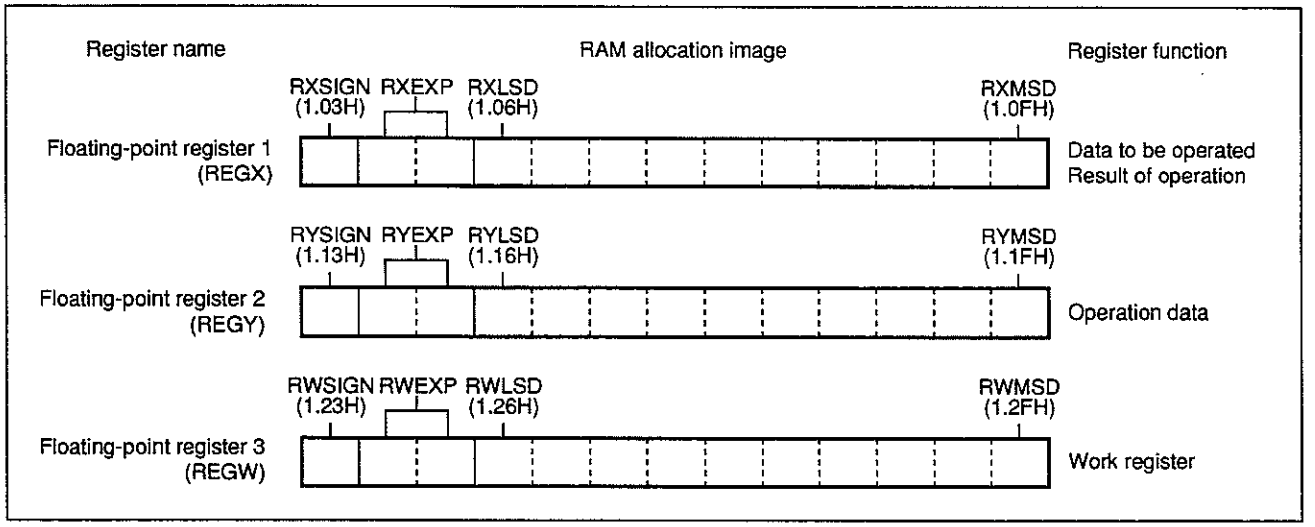
- $(-1)^{(\text{value of sign part})} \times (\text{value of mantissa}) \times 10^{(\text{value of characteristic})}$

7.2 Floating-Point Registers

With the arithmetic package, three floating-point registers are available which can handle floating-point decimals.

Figure 7-2 shows the RAM allocation image of the floating-point registers.

Figure 7-2. RAM Allocation Image of Floating-Point Registers



When the arithmetic package is used, each operation function is called after data to be operated is set in REGX, and operation data is set in REGY. Each operation ends after storing the result of operation in REGX.

For details of the flow of data between registers, see **Section 8.2.**

7.3 Parts of Floating-Point Format

This section describes each part of floating-point decimals.

(1) Mantissa

The mantissa represents a significant part of a numeric by using 10 digits: a 1-digit integer part and a 9-digit decimal fraction part. The mantissa holds the absolute value of a normalized decimal.

Normalization adjusts the characteristic and mantissa to bring the mantissa into a specified range. For all numerics except 0, the arithmetic package sets the range of the mantissa as follows: $0.1 \leq \text{mantissa} \leq 1$. So, when normalization is performed, the characteristic is adjusted so that the integer part holds 0, and the most significant digit of the decimal fraction part holds a number other than 0.

Table 7-1 indicates examples of normalization.

Table 7-1 Examples of Normalization

Number before normalization	Normalized Number
3729.45	0.372945×10^4
0.8765	0.8765×10^0
0.00054	0.54×10^{-3}

(2) Characteristic

The characteristic uses 2 hexadecimal digits to represent a base-10 exponent for a numeric to be expressed.

For a negative numeric, a twos complement is used.

Figure 7-3 shows the correspondence between characteristic representations and exponents.

Figure 7-3 Correspondence between Characteristic Representations and Exponents

Exponent	··· 10^7	10^6	···	10^2	10^1	10^0	10^{-1}	10^{-2}	···	10^{-5}	10^{-6} ···
Characterisctic	··· 07	06	···	02	01	00	FF	FE	···	FB	FA ···

Remark: The twos complement of an n-digit binary number is 2^n less the binary number.

(3) Sign part

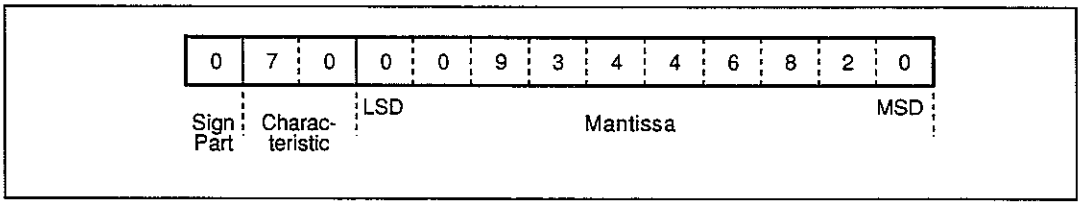
The sign of a numeric is represented using 1 bit.

For a positive numeric, the sign bit is reset to 0. For a negative numeric, the sign bit is set to 1.

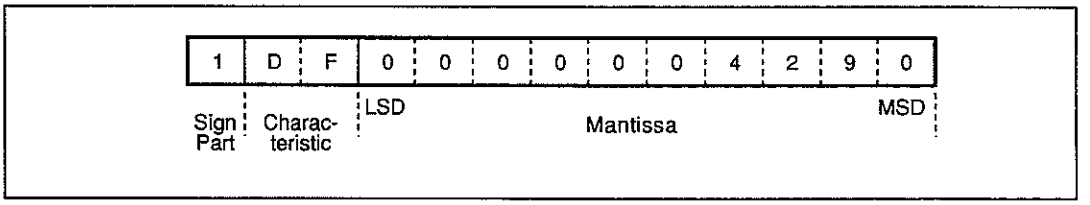
7.4 Examples of Storage in Floating-Point Registers

This section provides some examples of storing floating-point decimals in floating-point registers.

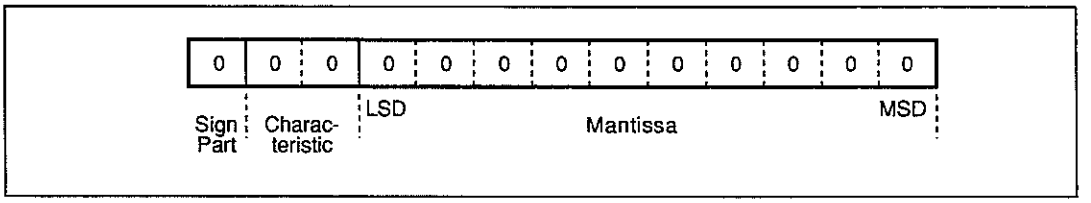
(1) 2864439 → 0.286443900 x 10^7



(2) –0.000924 → –0.924000000 x 10^{-3}



(3) 0 → 0.000000000 x 10^0



Chapter 8

Explanation of Arithmetic Package

8.1 List of Subroutines

Table 8-1 lists up the subroutines.

Table 8-1. Subroutines

Subroutine name		Processing	Relevant section
SFPADD	Floating-point addition	Adds the floating-point number of REGY to the floating-point number of REGX in decimal, then stores and normalizes the result of operation in REGX.	8.2.1
SFPSUB	Floating-point subtraction	Subtracts the floating-point number of REGY from the floating-point number of REGX in decimal, then stores and normalizes the result of operation in REGX.	8.2.2
SFPMULT	Floating-point multiplication	Multiplies the floating-point number of REGY by the floating-point number of REGX in decimal, then stores and normalizes the result of operation in REGX.	8.2.3
SFPDIV	Floating-point division	Divides the floating-point number of REGX by the floating-point number of REGY in decimal, then stores and normalizes the result of operation in REGX.	8.2.4
SNML	Normalization	Normalizes the result of operation stored in REGX.	8.3.1
SADD	Mantissa addition	Adds the mantissa of REGY to the mantissa of REGX in decimal, then stores the result of operation in REGX.	8.3.2
SSUB	Mantissa subtraction	Subtracts the mantissa of REGY or REGW from the mantissa of REGX in decimal, then stores the result of operation in REGX.	8.3.3
SADDEX	Characteristic addition	Adds the characteristic of REGY to the characteristic of REGX in hexadecimal, then stores the result of operation in the characteristic of REGX.	8.3.4
SSUBEX	Characteristic subtraction	Subtracts the characteristic of REGY from the characteristic of REGX in hexadecimal, then stores the result of operation in the characteristic of REGX.	8.3.5
SCHGXYEX	Register exchange 1	Exchanges the characteristic and mantissa of REGX with the characteristic and mantissa of REGY.	8.3.6 (1)
SCHGXY	Register exchange 2	Exchanges the mantissa of REGX with the mantissa of REGY.	8.3.6 (2)
SCHGXW	Register exchange 3	Exchanges the mantissa of REGX with the mantissa of REGW.	8.3.6 (3)
SUSHFX	Register shift-up 1	Shifts up the mantissa of REGX 1 digit.	8.3.7 (1)
SUSHFW	Register shift-up 2	Shifts up the mantissa of REGW 1 digit.	8.3.7 (2)
SDSHFX	Register shift-down 1	Shifts down the mantissa of REGX 1 digit.	8.3.8 (1)
SDSHFY	Register shift-down 2	Shifts down the mantissa of REGY 1 digit.	8.3.8 (2)
SDSHFW	Register shift-down 3	Shifts down the mantissa of REGW 1 digit.	8.3.8 (3)
SRXCLR	Register clear (0) 1	Clears the sign part, characteristic, and mantissa of REGX to 0.	8.3.9 (1)
SRWCLR	Register clear (0) 2	Clears the mantissa of REGW to 0.	8.3.9 (2)

8.2 Arithmetic Operations

The arithmetic package provides four arithmetic operation functions.

(1) Floating-point addition (SFPADD)

This function adds the value of REGY as an addend to the value of REGX as an augend.

(2) Floating-point subtraction (SFPSUB)

This function subtracts the value of REGY as a subtrahend from the value of REGX as a minuend.

(3) Floating-point multiplication (SFPMULT)

This function multiplies the value of REGY as a multiplicand by the value of REGX as a multiplier.

(4) Floating-point division (SFPDIV)

This function divides the value of REGX as a dividend by the value of REGY as a divisor.

Remark The result of operation is stored in REGX at the end of operation.

8.2.1 Floating-Point Addition (SFPADD)

(1) Processing

The floating-point number of REGY is added to the floating-point number of REGX in decimal, then the result of operation is stored in REGX and normalized.

(2) Input condition

A normalized augend and addend are to be stored in REGX and REGY, respectively.

(3) Output result

- The normalized result of operation is stored in REGX.
- REGY holds an addend which, however, is not normalized.

(4) Work area used

REGX

REGY

REGW (used to save a sign and exponent difference, and find a twos complement for a negative operation result)

RREG0, RREG1 (used to exchange REGX data with REGY data)

(5) Nesting level

2 levels

(6) Processing procedure

Addition is performed using steps (a), (b), (c), (d), and (e) in this order.

(a) Digit alignment of a mantissa and mantissa to be operated

First, REGX and REGY digit alignment is performed. When the fixed-point format is used, digit alignment is performed for operation. The floating-point format used with the arithmetic package allows fixed-point format addition to be performed by adjusting the exponents.

(i) A difference between the exponents of REGX and REGY is found.

(ii) Then, the value of REGX is exchanged with the value of REGY as required so that REGX holds the value with a greater exponent.

When add processing is performed later, the mantissa of REGY is shifted down so that the characteristic of REGY can match the characteristic of REGX. For this purpose, REGY must hold a smaller absolute value than REGX. For register exchange, work registers RREG0 and RREG1 are used.

The values of REGX and REGY are not exchanged with each other when there is no exponent difference or when the characteristic data of REGX is greater than the characteristic data of REGY.

(iii) Operation processing is continued when the exponent difference is less than 10.

When the exponent difference is 10 or greater, the operation ends, treating the value of REGX as the result of operation. This is because the mantissa available with the arithmetic package is 10 digits long, and smaller values cannot be stored.

(iv) In add processing, the mantissa of REGY is shifted down (for digit alignment) so that the characteristic value of REGY can match the characteristic value of REGX to allow the mantissa of REGY to be added to the mantissa of REGX.

An example of digit alignment is given below.

68.372 (REGX) + 245.91 (REGY)

$= (0.68372 \times \underline{10^2}) + (0.24591 \times \underline{10^3})$

; REGY is exchanged with REGX because the exponent of REGY is greater than the exponent of REGX.

$= \quad (0.24591 \quad \times 10^3) \quad$ (REGX)

$+ \ (0.68372 \quad \times 10^2) \quad$ (REGY)

$= \quad (0.24591 \quad \times 10^3)$

$+ \ (\underline{0.068372 \ \times 10^3})$

; The mantissa of REGY is shifted down so that the exponent of REGY can match the exponent (10^3) of REGX.

(b) Add processing when the operation data and data to be operated have the same sign

When REGX and REGY have the same sign, the absolute value of the mantissa of REGY can be added to the absolute value of the mantissa of REGY to find the result of operation. So, the SADD subroutine is called for mantissa addition in decimal.

(c) Add processing when the operation data and data to be operated do not have the same sign

When REGX and REGY do not have the same sign, the addend and augend are treated as a subtrahend and minuend, respectively, to find the result of operation by subtraction. So, SSUB subroutine is called for mantissa subtraction in decimal. The result of operation is stored in the mantissa of REGX.

(d) Finding a twos complement for the result of operation

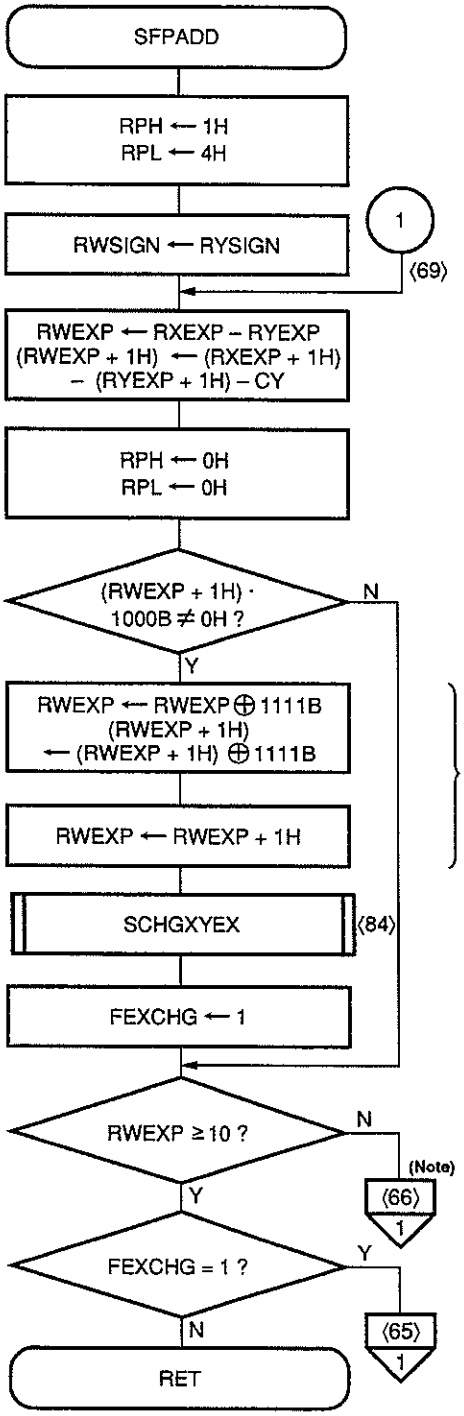
When the processing of (c) produces a negative value (when the mantissa of REGY is greater than the mantissa of REGX before operation), the twos complement of the resultant mantissa is found, then the sign bit is inverted to produce the result of operation.

(e) Normalization

Before processing termination, the SNML subroutine is called to normalize the result of operation held in REGX.

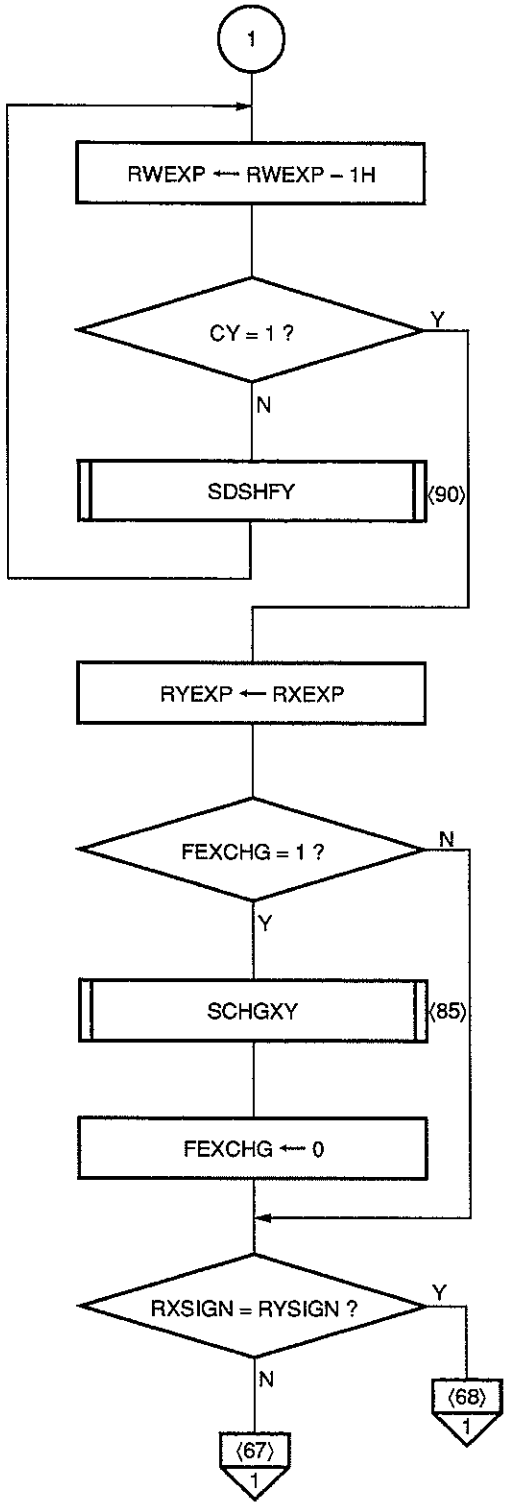
(7) Flowchart of floating-point addition

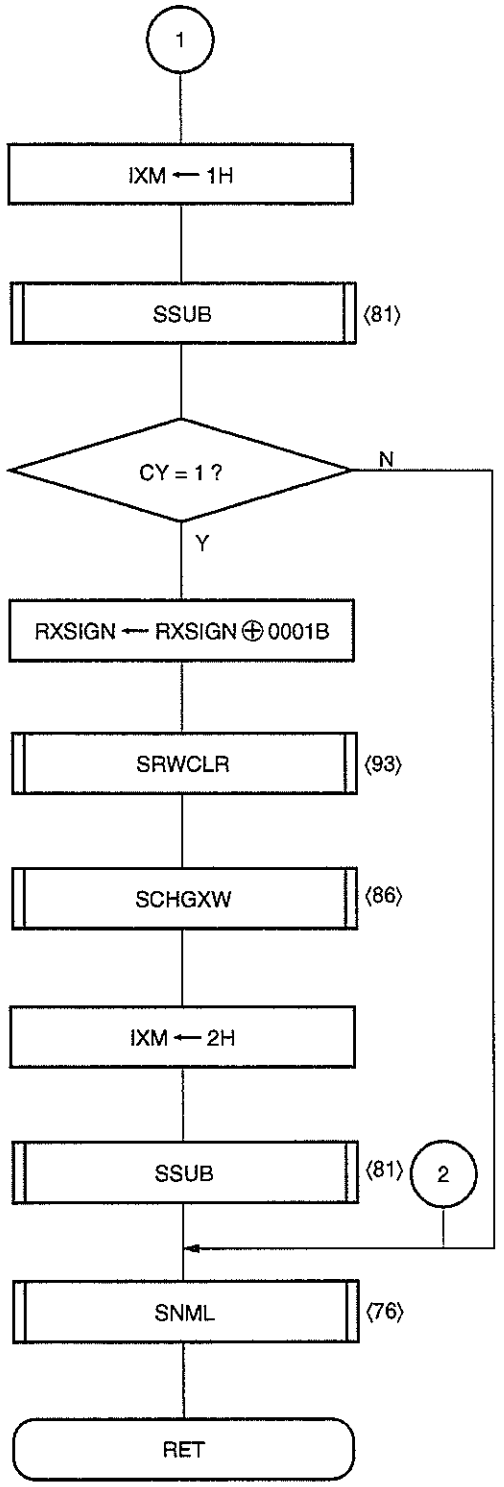
Input variables	Flowchart ⟨64⟩	Processing and remarks	Output variable
	SFPADD		
	RPH ← 1H RPL ← 4H	Sets general-purpose register at row address 2H of BANK1.	RPH RPL
REGY (1.13H–1.1FH)	RWSIGN ← RYSIGN	Holds sign of addend.	
REGX (1.03H–1.0FH)	RWEXP ← RXEXP – RYEXP (RWEXP + 1H) ← (RXEXP + 1H) – (RYEXP + 1H) – CY	Finds exponent difference (for digit alignment to allow additional and subtraction).	
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK 0.	RPH RPL
	(RWEXP + 1H) · 1000B ≠ 0H ? (N / Y)	Exponent of REGY (addend) greater?	
	RWEXP ← RWEXP ⊕ 1111B (RWEXP + 1H) ← (RWEXP + 1H) ⊕ 1111B RWEXP ← RWEXP + 1H	Finds absolute value of exponent difference (for digit alignment).	
	SCHGXYEX ⟨84⟩	REGX (augend) ⟷ REGY (addend)	
	FEXCHG ← 1	Sets register exchange flag.	
	RWEXP ≥ 10 ? (N / Y)	Exponent difference equal to or greater than 10?	
	FEXCHG = 1 ? (Y / N)	Register exchange flag set on?	
	RET		

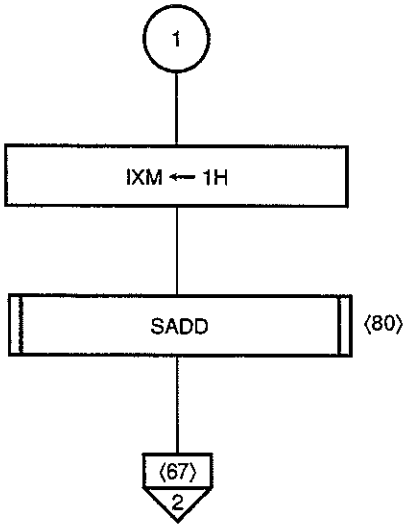


(Note) Jumps to ① of flowchart <66>.

Input variables	Flowchart (65)	Processing and remarks	Output variable
	1		
	RPH ← 1H RPL ← 0H	Sets general-purpose register at row address 0H of BANK1.	RPH RPL
	RYEXP ← RXEXP (RYEXP + 1H) ← (RXEXP + 1H)	Restores characteristic of REGY.	
	RYLSD ← RXLSD (RYLSD + 1H) ← (RXLSD + 1H) (RYLSD + 2H) ← (RXLSD + 2H) (RYLSD + 3H) ← (RXLSD + 3H) (RYLSD + 4H) ← (RXLSD + 4H) (RYLSD + 5H) ← (RXLSD + 5H) (RYLSD + 6H) ← (RXLSD + 6H) (RYLSD + 7H) ← (RXLSD + 7H) (RYLSD + 8H) ← (RXLSD + 8H) RYMSD ← RXMSD	Restores mantissa of REGY.	
	RPH ← 0H	Sets general-purpose register at row address 0H of BANK 0.	RPH
	RXSIGN ← RWSIGN	Stores sign of result of operation.	REGX (1.03H–1.0FH)
	FEXCHG ← 0	Clears register exchange flag.	
	RET		

Input variables	Flowchart ⟨66⟩	Processing and remarks	Output variable
		Decrements exponent difference. Exponent difference eliminated? Shifts down operation data. (Digit alignment (to increase smaller exponent to match greater exponent))	
		Match characteristic values.	
		REGX (augend) exchanged with REGY (addend)?	
		REGX ⟷ REGY (to return to original state)	
		Clears register exchange flag.	
		REGX and REGY have same sign?	

Input variables	Flowchart ⟨67⟩	Processing and remarks	Output variable
		Sets row address of REGY in index register.	IXM
		REGX (augend) – REGY (addend)	
		Borrow produced by subtraction?	
		Inverts sign of result of operation.	REGX (1.03H–1.0FH)
		REGW ← ALL 0	
		REGX (result of operation) ⟷ REGW	
		Index register ← row address of REGW	IXM
		REGX (0) – REGW (result of operation)	
		Changes representation including complement to representation including no complement.	
		Normalizes result of operation.	

Input variables	Flowchart ⟨68⟩	Processing and remarks	Output variable
		Sets general-purpose register at row address 2H of BANK1. REGX (augend) + REGY (addend)	IXM

8.2.2 Floating-Point Subtraction (SFPSUB)

(1) Processing

The floating-point number of REGY is subtracted from the floating-point number of REGX in decimal, then the result of operation is stored in REGX and normalized.

(2) Input condition

A normalized minuend and subtrahend are to be stored in REGX and REGY, respectively.

(3) Output result

- The normalized result of operation is stored in REGX.
- REGY holds a subtrahend which, however, is not normalized.

(4) Work area used

REGX

REGY

REGW (used to save a sign and exponent difference, and find a twos complement for a negative operation result)

RREG0, RREG1 (used to exchange REGX data with REGY data)

(5) Nesting level

2 levels

(6) Processing procedure

Subtraction can be handled as addition by inverting the sign bit of a subtrahend.

So, subtraction is performed using steps (a) and (b) in this order.

(a) Inverting the sign bit of a subtrahend

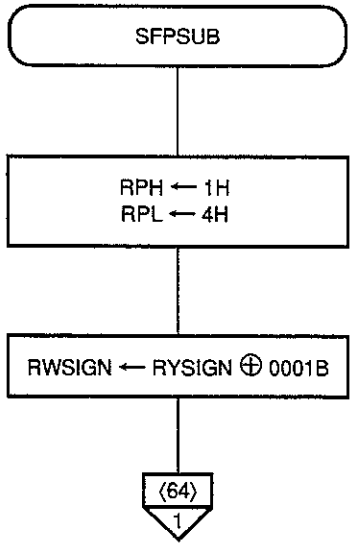
Invert the sign bit of REGY.

(b) Floating-point add processing

Processing after (a) is the same as for addition. So, floating-point addition (SFPADD) is used for subsequent processing.

For details, see the processing procedure of floating-point addition (SFPADD).

(7) Flowchart of floating-point subtraction

Input variables	Flowchart ⟨69⟩	Processing and remarks	Output variable
REGY (1.13H–1.1FH)		Sets general-purpose register at row address 2H of BANK1. Saves inverted sign of subtrahend.	RPH RPL

8.2.3 Floating-Point Multiplication (SFPMULT)

(1) Processing

The floating-point number of REGY is multiplied by the floating-point number of REGX in decimal, then the result of operation is stored is REGX and normalized.

(2) Input condition

A normalized multiplier and multiplicand are to be stored in REGX and REGY, respectively.

Remark For multiplication only, REGX is to hold an operation data (multiplier) and REGY is to hold data to be operated (multiplicand).

---

(3) Output result

- The normalized result of operation is stored in REGX.
- REGY holds a multiplier.

(4) Work area used

REGX

REGY

REGW (used to save the sign of the result of operation, save an exponent, and store a multiplier, and also used as a counter)

RREG0 (used to restore a saved sign)

(5) Nesting level

2 levels

(6) Processing procedure

The multiplication function of the arithmetic package follows a procedure used by the human to do calculation on a piece of paper with a pencil.

An example of multiplication (25.7 x (–0.32)) is given below to review the hand-writing multiplication procedure.

Figure 8-1. Example of Hand-Writing Multiplication

```
    +  2 5 7
 x( -  0.3 2 )
 ------------
       5 1 4   ···② (1)
     7 7 1     ···② (2)
 ------------
   -8.2 2 4
   ┴ ┴ ┴
   ① ④ ③
```

<1> The sign of the result of operation is found. When a multiplicand and multiplier have the same sign, the result of operation has the plus sign. When a multiplicand and multiplier do not have the same sign, the result of operation has the minus sign. This example multiplies a positive number by a negative number, so that the result of operation has the minus sign.

<2> The digits of a multiplicand are multiplied by each digit of a multiplier, starting with the least significant digit of the multiplier. In this operation, the decimal point is ignored.

(1) 257 x 2 is performed.

(2) 257 x 3 is performed. Multiplier 3 in this multiplication is in the tens digit position, so that the result of multiplication is shifted 1 digit left when written down.

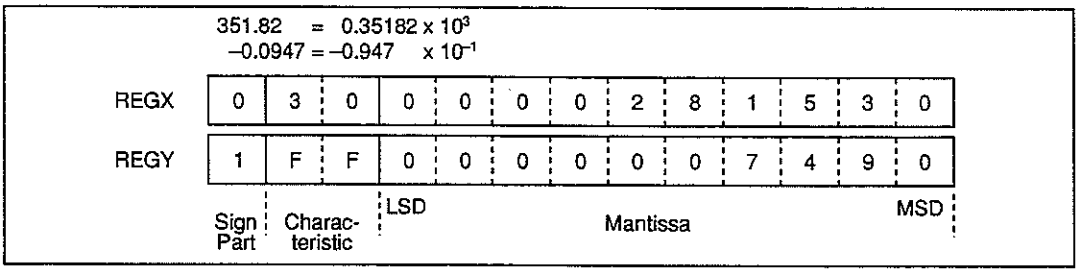
<3> The result of multiplication by one digit of the multiplier is added to the result of multiplication by the other digit of the multiplier.

<4> The position of the decimal point in the result of operation is found. In this example, the multiplicand is 257 x 10^{-1}, and the multiplier is 32 x 10^{-2}, so that the result of addition of 3 above is multiplied by 10^{-3}. This means that the decimal-point position is placed between the third digit and fourth digit counting from the least significant digit.

From the above, the result of 25.7 x (–0.32) is –8.224.

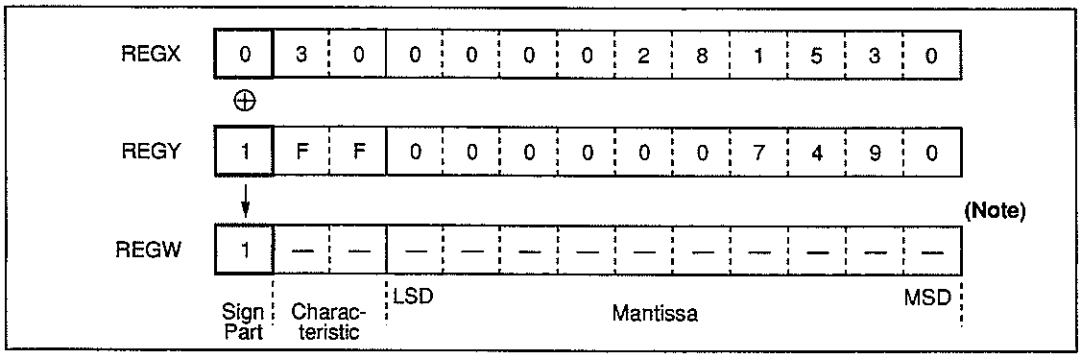
The arithmetic package carries out multiplication as described below.

The example below uses 351.82(REGX) x (–0.0947)(REGY) for explanation.



(a) Multiplication of the sign part

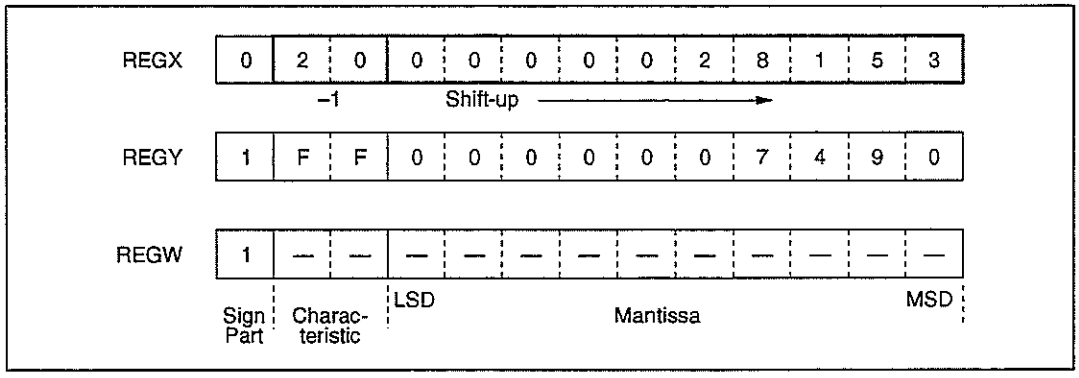
The sign of the result of operation is found (by (REGX sign part) ⊕ (REGY sign part)). When the multiplier and multiplicand have the same sign, the result of operation has the plus sign (with the sign bit set to 0). When the multiplier and multiplicand do not have the same sign, the result of operation has the minus sign (with the sign bit set to 1). The resultant sign is saved to the sign part of REGW.



(Note) Each position marked with "–" holds an undefined value.

(b) Shifting up the multiplier 1 digit

The mantissa of REGX is shifted up 1 digit with SUSHFX, and 1 is subtracted from the characteristic of REGX.



The reason for shifting up REGX is described below.

The purpose is to minimize the error of operation. If multiplication is performed using a value just normalized, the resultant mantissa can be a lower-digit decimal fraction. In such a case, normalization performed at a later stage stores 0 in the least significant digit of the mantissa, thus resulting in a greater operation error. So, the multiplier is shifted up 1 digit before multiplication to minimize an error that can be caused by the mantissa resulting in a lower-digit decimal fraction.

Example of a mantissa resulting in a lower-digit decimal fraction:

0.33333(REGY) x 0.22222(REGX)

$= (0.33333 \times 0.22222) \times 10^{(0+0)}$

$= 0.\underline{0}74072592 \times 10^{0}$; Causes the mantissa to result in a lower-digit decimal fraction.

$= 0.740725920 \times 10^{-1}$; Stores 0 in the least significant digit by normalization

↓

Example of operation with a multiplier shifted up 1 digit:

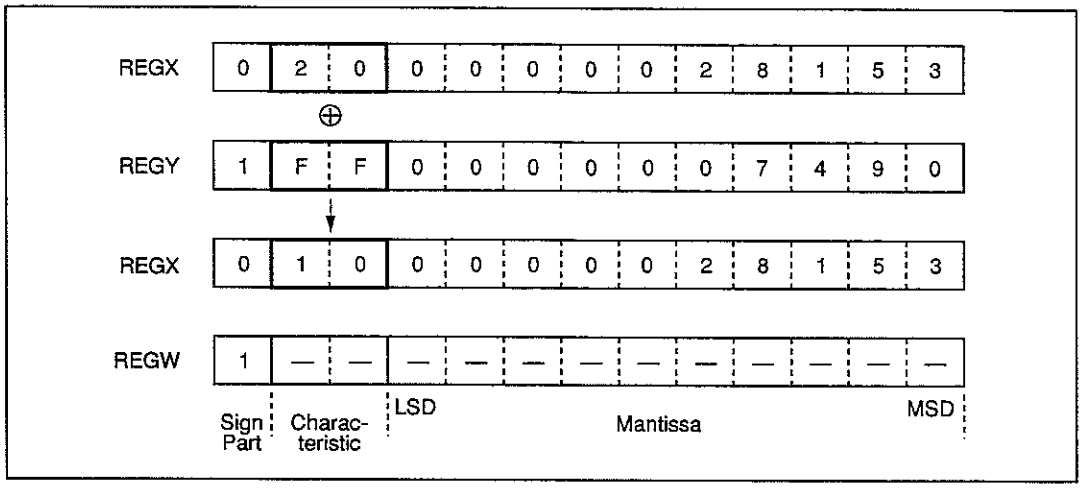
$= 0.33333 \times (\underline{2.2222 \times 10^{-1}})$; Shifts up REGX 1 digit.

$= (0.33333 \times 2.2222) \times 10^{(0+(-1))}$

$= 0.740725926 \times 10^{-1}$

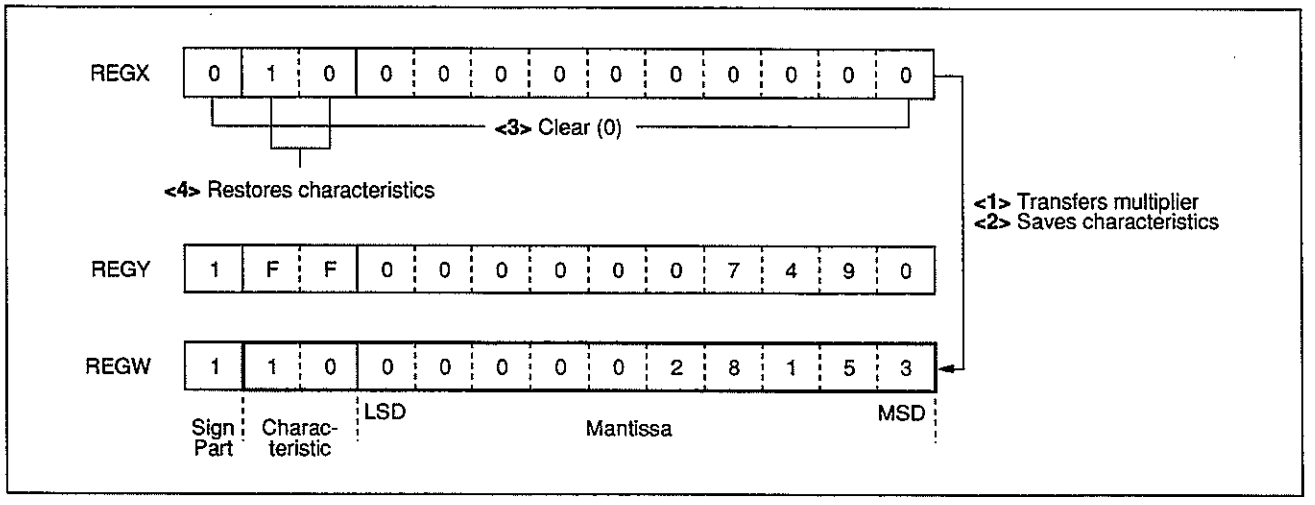
(c) Characteristic addition

The characteristic of the result of operation is found. The characteristic of the multiplier (REGX) is added to the characteristic of the multiplicand (REGY) with SADDEX, then the result of addition is stored in the characteristic of REGX.



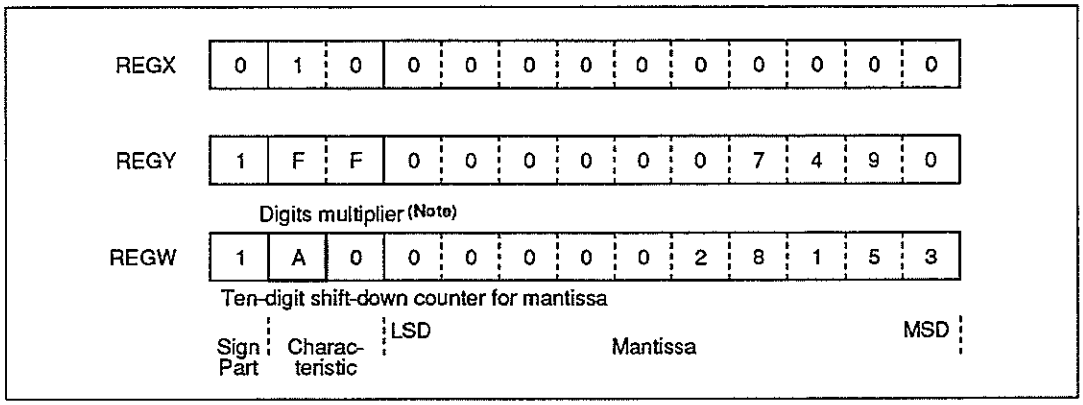
(d) Multiplier transfer and operation result area clearing

The mantissa of REGX is exchanged with the mantissa of REGW by using SCHGXW, then REGX is cleared to 0 with SRXCLR. This processing is required to store the result of operation in REGX. After REGX is cleared to 0, the exponent of the result of operation is restored in the characteristic of REGX.



(e) Setting of a ten-digit shift-down counter for mantissa

The low-order digit of the characteristic of REGW is set as a ten-digit shift-down counter for the mantissa.

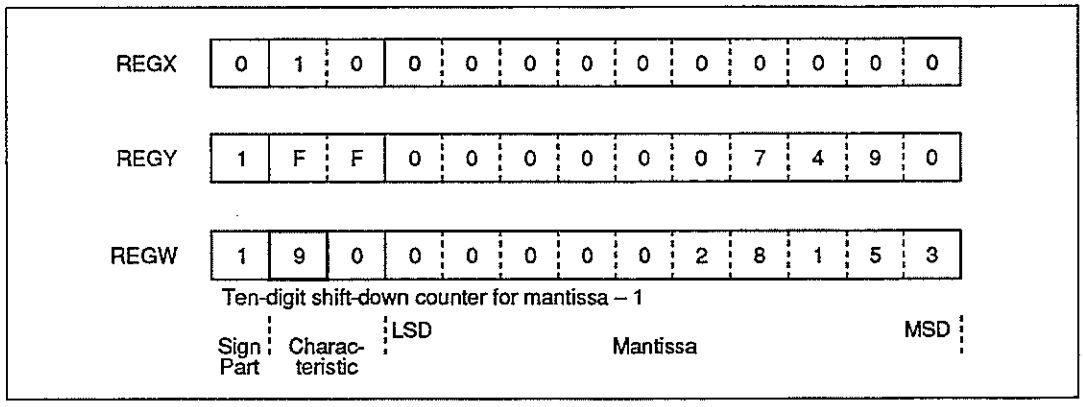


The ten-digit shift-down counter for the mantissa is initialized to the number of mantissa digits. The purpose is to count the number of mantissa shift-down occurrences because multiplication ((digit multiplier)(Note)XREGY) is performed for all mantissa digits starting with the least significant digit of the multiplier.

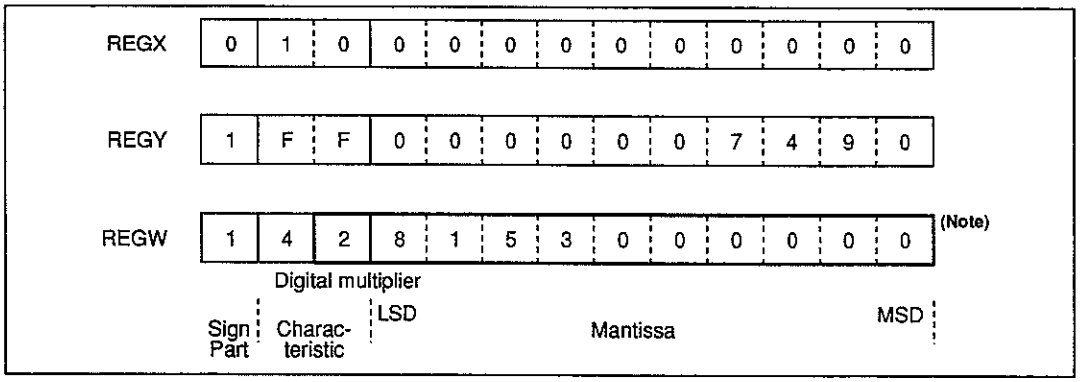
(Note) Used in step (f) as an area for storing a least significant digit overflow caused by shift-down.

(f) Multiplication by each digit of the multiplier

(i) The ten-digit shift-down counter for the mantissa counts down. When the value of the counter reaches FH, processing of (f) is terminated.

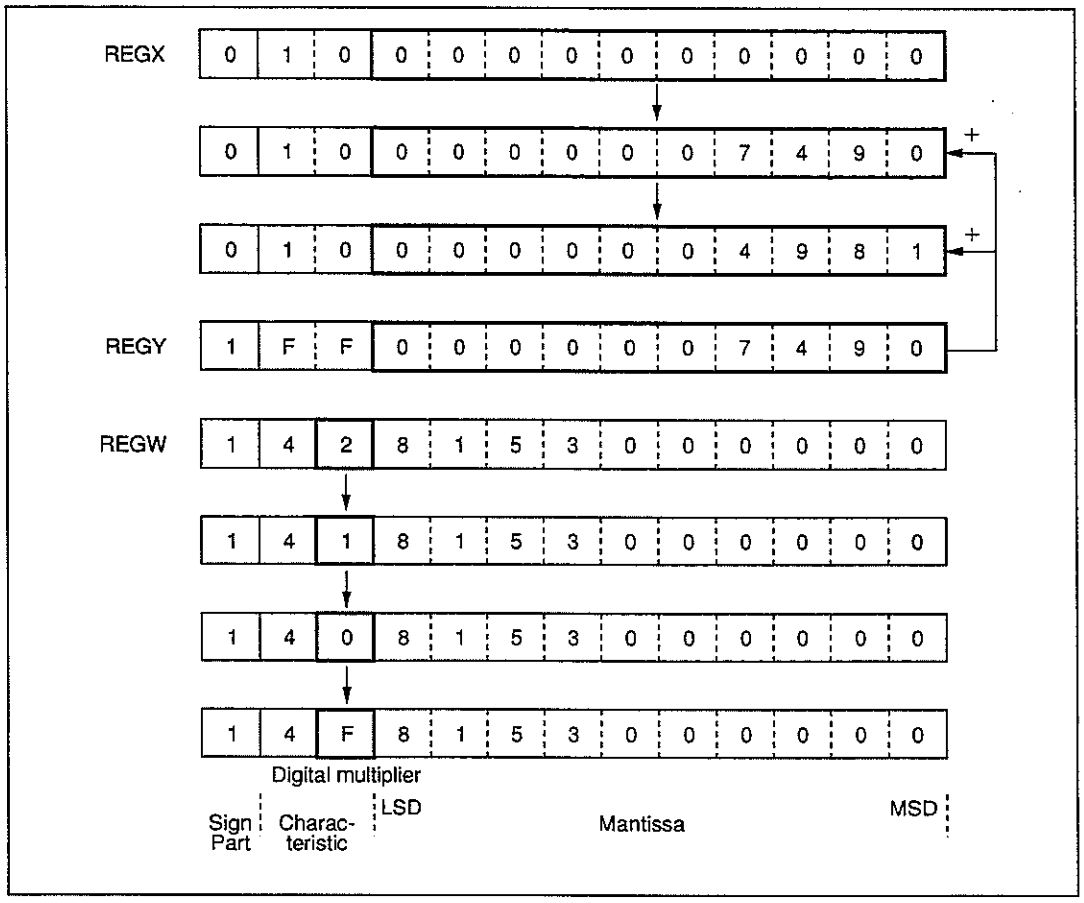


(ii) The mantissa of REGW storing the multiplier is shifted down 1 digit with SDSHFW. The least significant digit that overflowed as the result of shift-down is stored in the high-order digit of the characteristic. Such a least significant digit that overflowed serves as a multiplier (digit multiplier) for digit-by-digit multiplication.

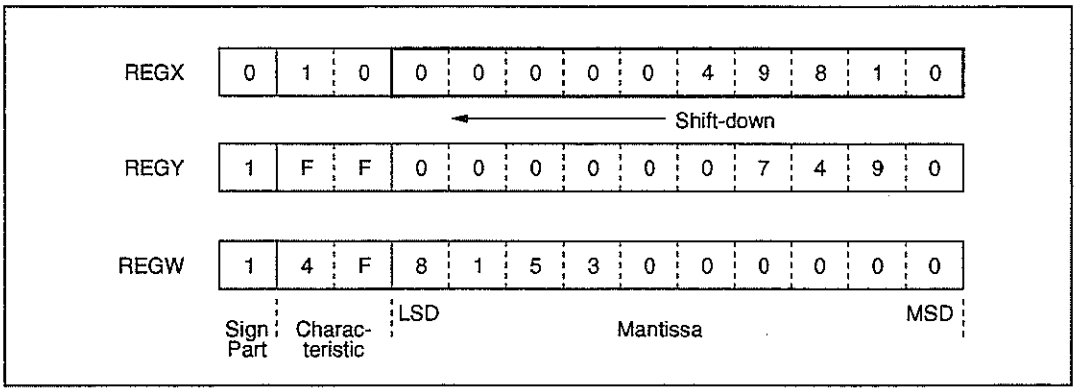


(Note) Processing of the 6th digit

(iii) The multiplicand is multiplied by each digit of the multiplier. Specifically, the multiplicand of REGY is added by SADD as many times as the value of each digit multiplier to REGX storing the result of operation.



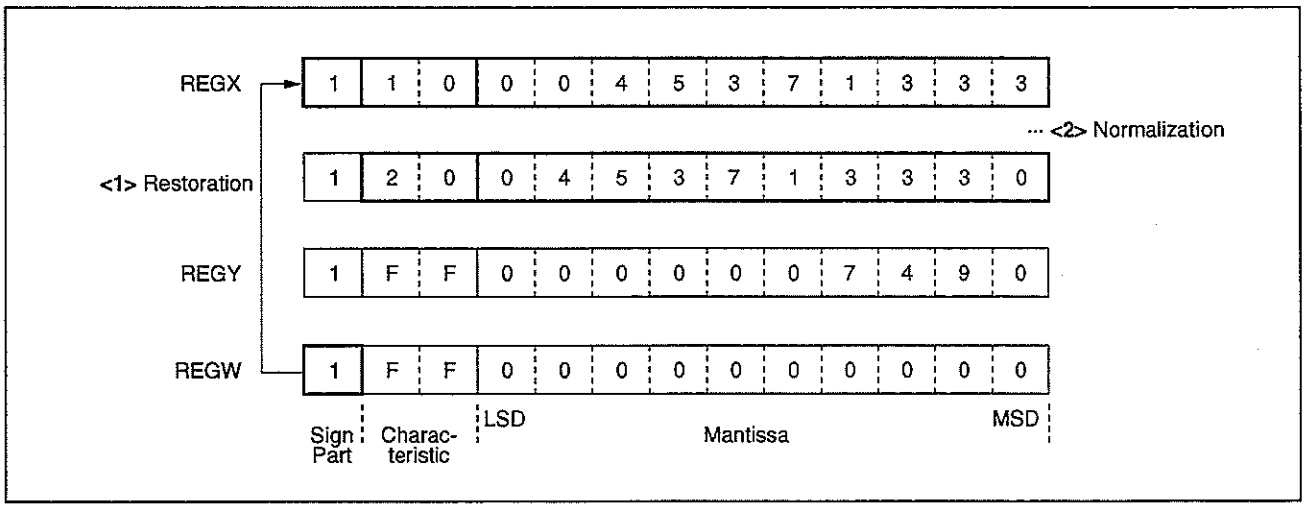
(iv) The mantissa of REGX is shifted down 1 digit with SDSHFX. In hand-writing multiplication, the result of operation is shifted up 1 digit for each multiplier digit. On the other hand, the arithmetic package achieves the same effect by shifting down the sum of the results of operations performed thus far.



(v) Steps (i) through (iv) are performed for the 10 digits of the mantissa (until the value of the ten-digit shift-down counter for the mantissa reaches FH.)

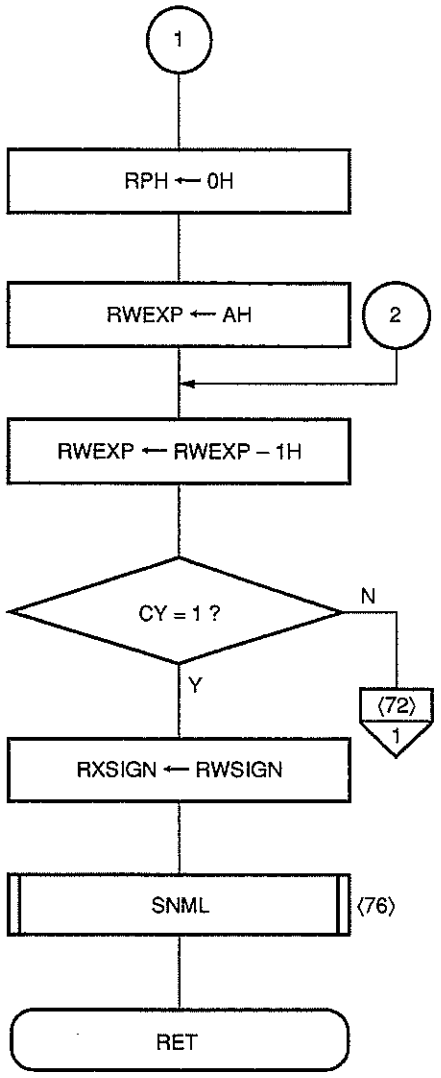
(g) Normalization

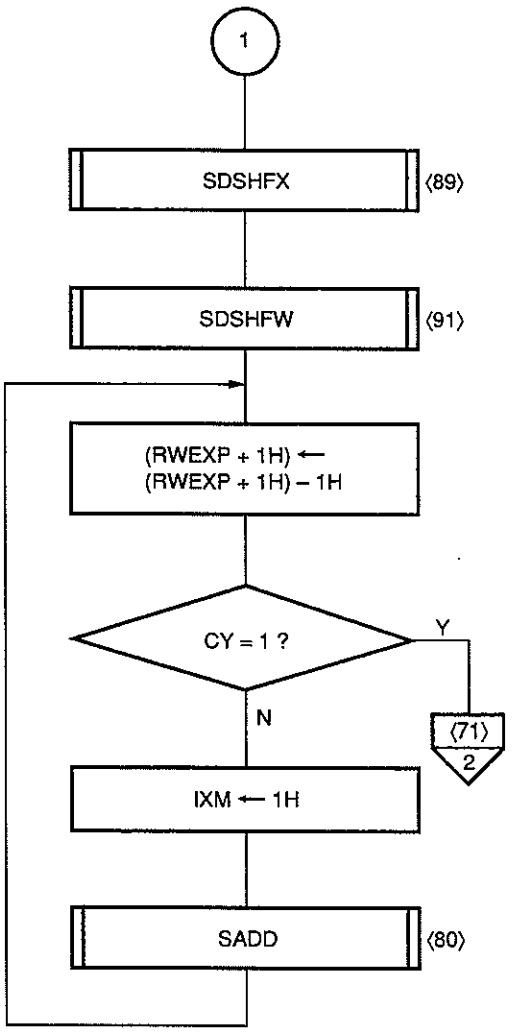
The resultant sign saved in REGW is restored in REGX, then normalization is performed to complete the processing.



(7) Flowchart of floating-point multiplication

Input variables	Flowchart ⟨70⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH) REGY (1.13H–1.1FH)	SFPMULT		
	RPH ← 1H RPL ← 0H	Sets general-purpose register at row address 0H of BANK1.	RPH RPL
	RWSIGN ← RXSIGN ⊕ RYSIGN	Finds and saves sign of result of operation. RXSIGN RYSIGN → RWSIGN + + + + − − − + − − − +	
	RPH ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH
	SUSHFX ⟨87⟩	Shifts up multiplier.	
	RXEXP ← RXEXP + FH (RXEXP + 1H) ← (RXEXP + 1H) + FH + CY	Decrements exponent by 1.	
	SADDEX ⟨82⟩	Characteristic addition (REGX – REGY) (to find exponent for product)	
	SCHGXW ⟨86⟩	REGX ⟷ REGW	
	RPH ← 1H	Sets general-purpose register at row address 0H of BANK1.	RPH
	RWEXP ← RXEXP (RWEXP + 1H) ← (RXEXP + 1H)	Save characteristic (to clear REGX to 0).	
	SRXCLR ⟨92⟩	Clears entire REGX (product) to 0.	
	RXEXP ← RWEXP (RXEXP + 1H) ← (RWEXP + 1H)	Restores characteristic.	
	⟨71⟩ 1		

Input variables	Flowchart ⟨71⟩	Processing and remarks	Output variable
		Sets general-purpose register at row address 0H of BANK0. Initializes ten-digit shift-up counter for mantissa. Decrements ten-digit shift-up counter for mantissa. Add operations for 10 mantissa digits completed? Stores sign of result of operation. Normalizes product.	RPH REGX (1.03H–1.0FH)

Input variables	Flowchart ⟨72⟩	Processing and remarks	Output variable
REGW (1.23H–1.2FH)		Shifts down product area. Shifts down multiplier area. Decrements digit multiplier. Add operations performed as many times as required? Sets row address of REGY in index register. REGX (product) + REGY (multiplicand)	IXM

8.2.4 Floating-Point Division (SFPDIV)

(1) Processing

The floating-point number of REGX is divided by the floating-point number of REGY in decimal, then the result of operation is stored in REGX and normalized.

(2) Input condition

A normalized dividend and divisor are to be in REGX and REGY, respectively.

(3) Output result

- The normalized result of operation is stored in REGX. When division by zero is attempted, an error occurs (the FDVERR flag is set); the processing is terminated, preserving the dividend stored in REGX.
- REGY holds a divisor.

(4) Work area used

REGX

REGY

REGW (used to save the result of operation, and also used as a counter)

RREG0 (used to check for division by zero)

(5) Nesting level

2 levels

(6) Processing procedure

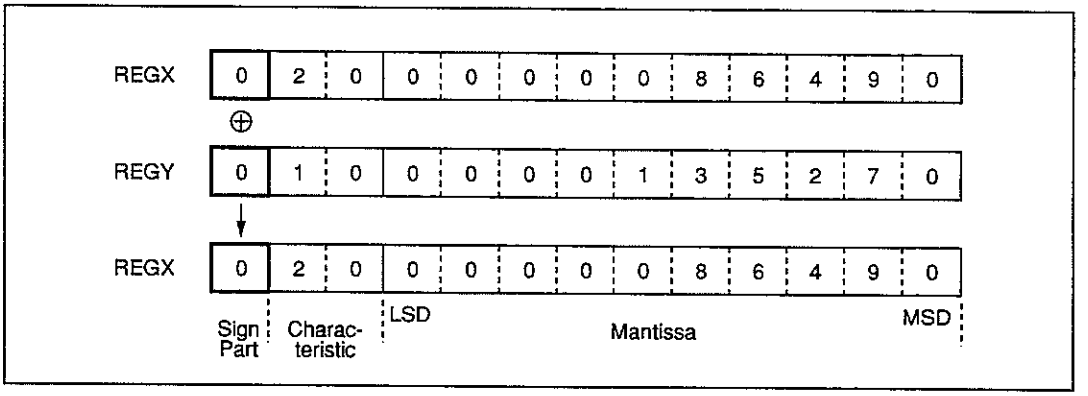
An example of division (94.68 (REGX) / 7.2531 (REGY)) is given below for explanation.

$94.68 = 0.9468 \times 10^2$
$7.2531 = 0.72531 \times 10^1$

REGX	0	2	0	0	0	0	0	0	8	6	4	9	0
REGY	0	1	0	0	0	0	0	1	3	5	2	7	0
	Sign Part	Charac-teristic		LSD					Mantissa				MSD

(a) Sign part division

The sign of the result of operation is found (by (REGX sign part) ⊕ (REGY sign part)), then is stored in the sign part of REGX. When the dividend (REGX) and divisor (REGY) have the same sign, the result of operation has the plus sign (with the sign bit set to 0). When the dividend (REGX) and divisor (REGY) do not have the same sign, the result of operation has the minus sign (with the sign bit set to 1).

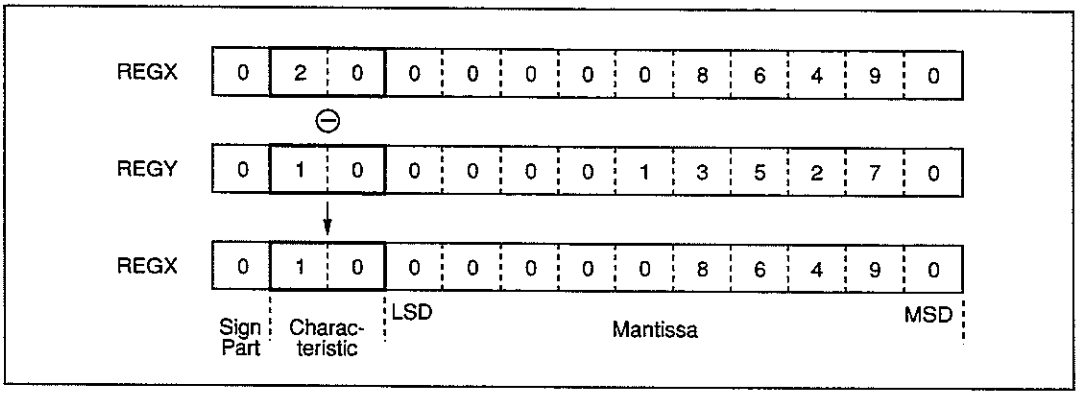


(b) Zero-division check

When the mantissa of REGY is 0, an error occurs (the FDVERR flag is set); the processing is terminated.

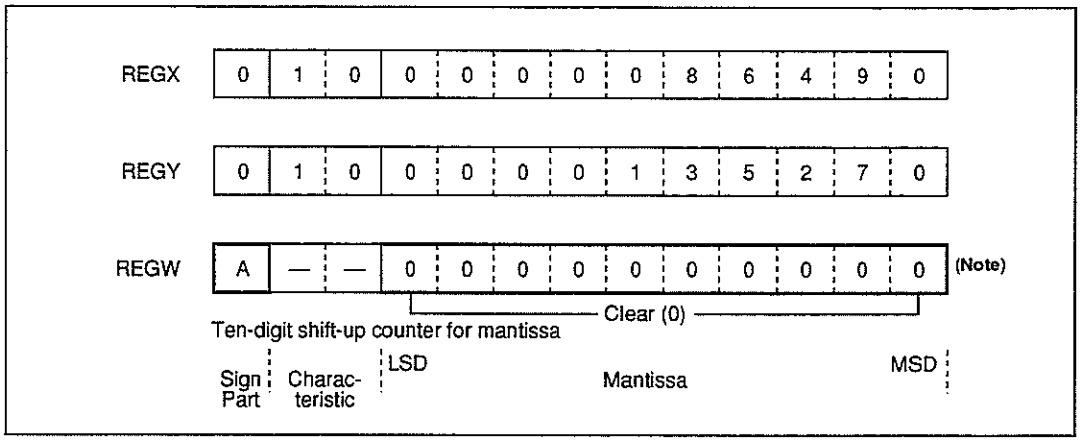
(c) Characteristic subtraction

The exponent of the result of operation is found. The characteristic of the divisor (REGY) is subtracted from the characteristic of the dividend (REGX) with SSUBEX, then the result of subtraction is stored in the characteristic of REGX.



(d) Clearing the operation result area to 0

The mantissa of REGW used to store the result of operation is cleared to 0 with SRWCLR, and the sign part of REGW is used as a ten-digit shift-up counter for the mantissa.



(Note) Each position marked with "–" holds an undefined value.

(e) Division by a restoration method

The arithmetic package uses a restoration method for division. With this method, a subtraction is carried out, treating a dividend and divisor as a minuend and subtrahend, respectively. At a second and later steps, the divisor is repeatedly subtracted from the result of each subtraction; the number of subtractions performed until the result of subtraction becomes negative serves as the quotient for the digit.

When the result of subtraction becomes negative, the divisor is added to the result of subtraction to return the result of subtraction to a positive number. Then, that positive result is shifted up 1 digit for continued subtraction. This shift-up operation means that a quotient obtained in the next stage of subtractions is lower by 1 digit than the previous quotient.

This processing is continued until the result of subtraction becomes 0.

Figure 8-2 shows an example of division using the restoration method.

Figure 8-2. Example of Division with Restoration Method

```
      1.2
5 ) 6      ←
    5      ···<1>
   10      ···<2>
   10
    0      ]—<3>
```

Example of division	Processing	Quotient
<1> 6 −5 1	The divisor could be subtracted from the dividend, so quotient 1 is obtained.	1
1 −5 −4	The result of subtraction is negative, so no quotient is obtained.	1
<2> −4 + 5 1 ↓ 10	The divisor is added to the result of subtraction to return the result of operation to a positive number. Then, that positive result is shifted up 1 digit. This shift-up operation makes subsequent quotients 1 digit lower.	1
<3> 10 −5 5	The divisor could be subtracted, so quotient 0.1 is obtained, and is added to the previous quotient of 1.	1.1
<3> 5 −5 0	The divisor could be subtracted, so quotient 0.1 is added to the previous quotient of 1.1. Because the result of operation becomes 0, division processing is terminated.	1.2

(i) The ten-digit shift-up counter for the mantissa counts down.

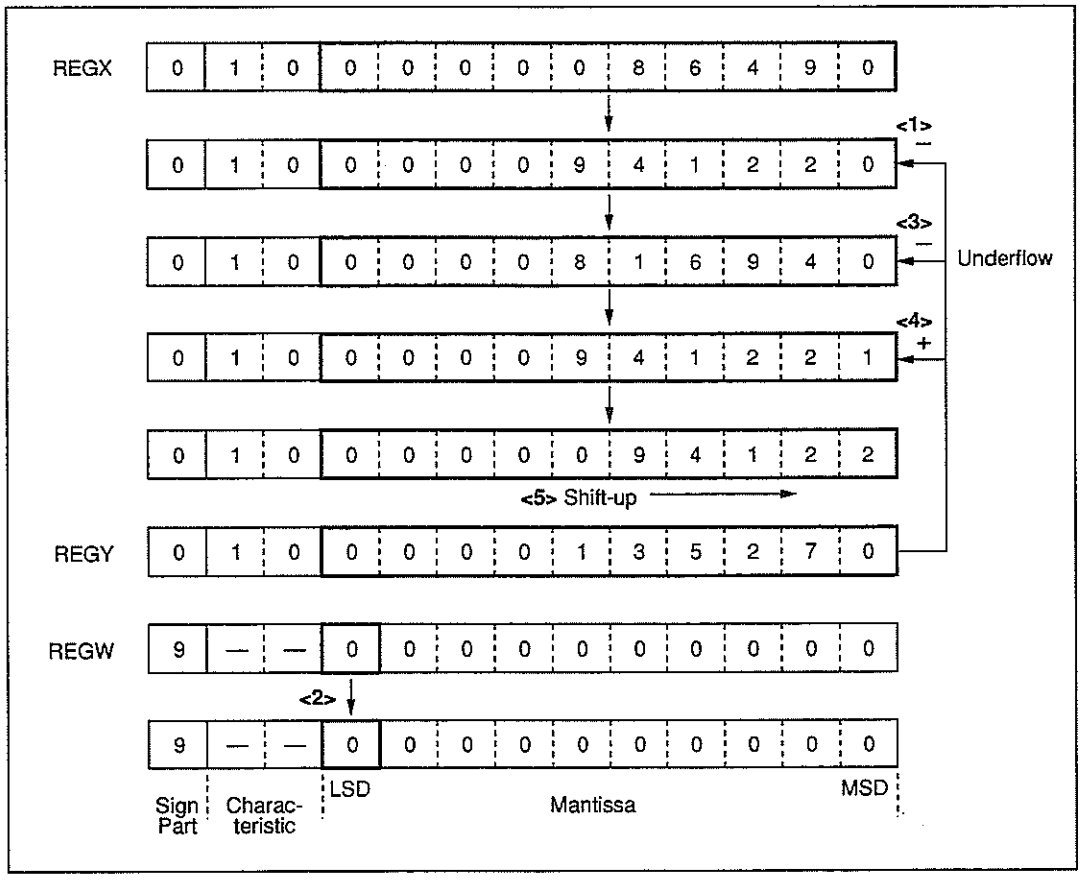
REGX	0	1	0	0	0	0	0	0	0	0	0	0	0
REGY	0	1	0	0	0	0	0	1	3	5	2	7	0
REGW	9	—	—	0	0	0	0	0	2	8	1	5	3

Ten-digit shift-up counter for mantissa – 1

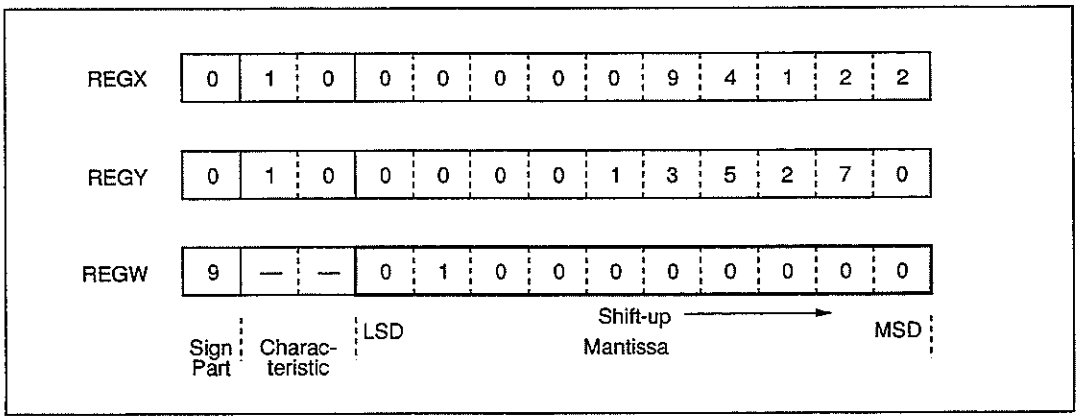
Sign Part | Characteristic | LSD | Mantissa | MSD

(ii) REGY is subtracted from REGX with SSUB. If no underflow occurs, 1 is added as a quotient to the least significant digit of REGW.

If an underflow occurs, REGY is added to REGX, then REGX is shifted up 1 digit with SUSHFX. This shift-up operation allows subsequent subtractions.



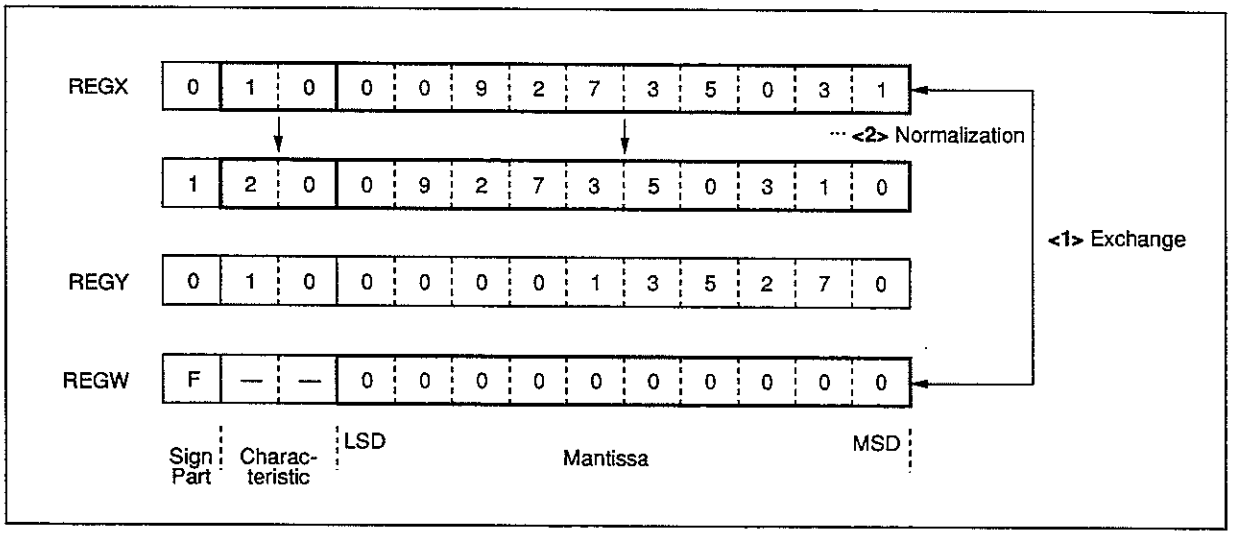
(iii) The mantissa of REGW is shifted up 1 digit with SUSHFW. This operation shifts up the result of operation to update the quotient digit.



(iv) Steps (i) through (iii) are performed for the 10 digits of the mantissa (until the value of the ten-digit shift-up counter for the mantissa reaches FH.)

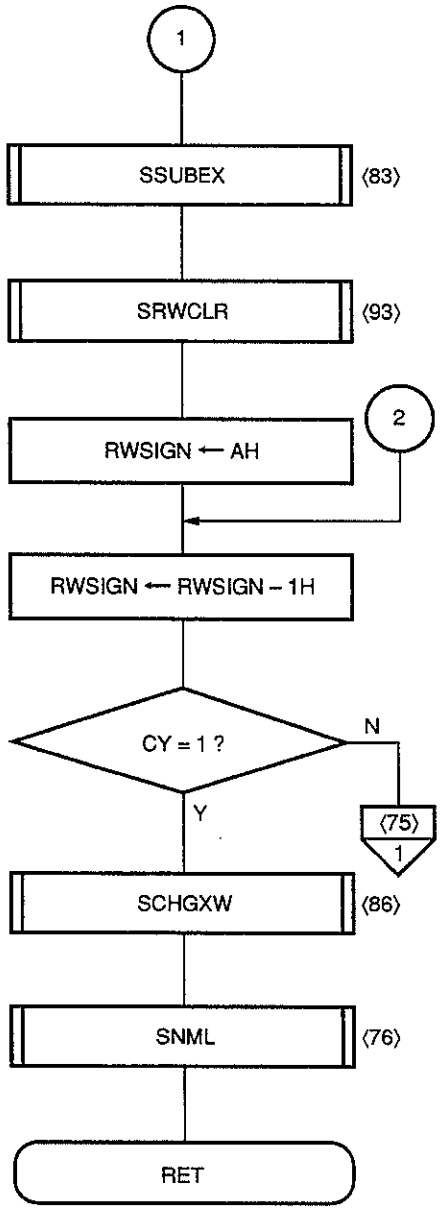
(f) Normalization

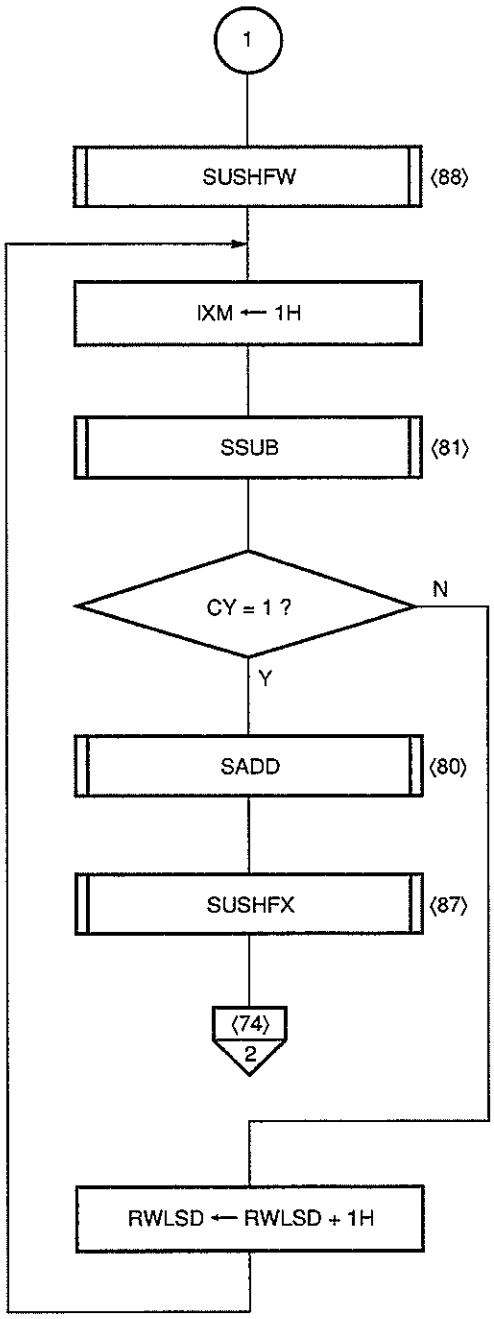
The mantissa of REGX is exchanged with the mantissa of REGW by using SCHGXW, then the result of operation is stored in REGX. Next, the result of operation is normalized to complete the processing.



(7) Flowchart of floating-point division

Input variables	Flowchart ⟨73⟩	Processing and remarks	Output variable
	SFPDIV		
	FDVERR ← 0	Clears zero-division error flag.	FDVERR (1.30H.2)
	RPH ← 1H RPL ← 0H	Sets general-purpose register at row address 0H of BANK1.	RPH RPL
REGX (1.30H–1.0FH) REGY (1.13H–1.1FH)	RXSIGN ← RXSIGN ⊕ RYSIGN	Finds sign of result of operation.	REGX (1.30H–1.0FH)
	RPH ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH
	IXH ← 0H IXM ← 8H IXL ← 6H	Sets index register at column address 6H of BANK1.	
	IXE ← 1	Check divisor (mantissa of REGY) for zero.	
	RREG0 ← REGY : Divisor [IX]		
	IXE ← 0		
	RREG0 = 0H ? N → ⟨74⟩ 1		
	Y: IX ← IX + 1H		
	IXL = 0H ? N → IXL ← IXL + 1		
	Y: FDVERR ← 1	Sets zero-division error flag.	FDVERR (1.30H.2)
	RET		

Input variables	Flowchart ⟨74⟩	Processing and remarks	Output variable
			
		Characteristic subtraction (REGX – REGY) (to find quotient exponent value)	
		Clears entire REGW (quotient) to 0.	
		Initializes ten-digit shift-up counter for mantissa.	
		Decrements ten-digit shift-up counter for mantissa.	
		Subtract operations for 10 mantissa digits completed?	
		REGX ⟷ REGW (to transfer quotient in REGW to REGX)	
		Normalizes quotient.	

Input variables	Flowchart ⟨75⟩	Processing and remarks	Output variable
			
		Shifts up quotient area.	
		Sets row address of REGY in index register.	IXM
		REGX (dividend) – REGY (divisor)	
		Borrow produced by subtraction?	
		REGX (dividend) + REGY (divisor)	
		Shifts up dividend area.	
		Adds 1 to least significant digit of quotient.	

8.3 Other Subroutines (Internal Routines Not Called by Users)

8.3.1 Normalization (SNML)

- **Processing**

The result of operation stored in REGX is normalized.

When the result of operation is 0, the operation result zero flag (FZERO) is set to 1.

When the result of operation involves an overflow or underflow, the overflow flag (FOVER) is set to 1.

An overflow and underflow is detected using the exponent of REGX. An exponent is represented as a twos complement using two hexadecimal digits, so that the representable range of exponents is: $-128 \leq$ exponent ≤ 127. The arithmetic package assumes the occurrence of an overflow or underflow when an exponent beyond the following range results: $-64 \leq$ exponent ≤ 63. In such a case, the overflow flag is set. By checking the overflow flag with the system control section, the result of operation can be prevented from being destructed by an overflow or underflow.

When an operation produces a number represented using a higher digit, the least significant digit of the mantissa is discarded.

0.77777×0.88888

$= 0.77777 \times (8.8888 \times 10^{-1})$

$= (0.77777 \times 8.8888) \times 10^{(0 + (-1))}$

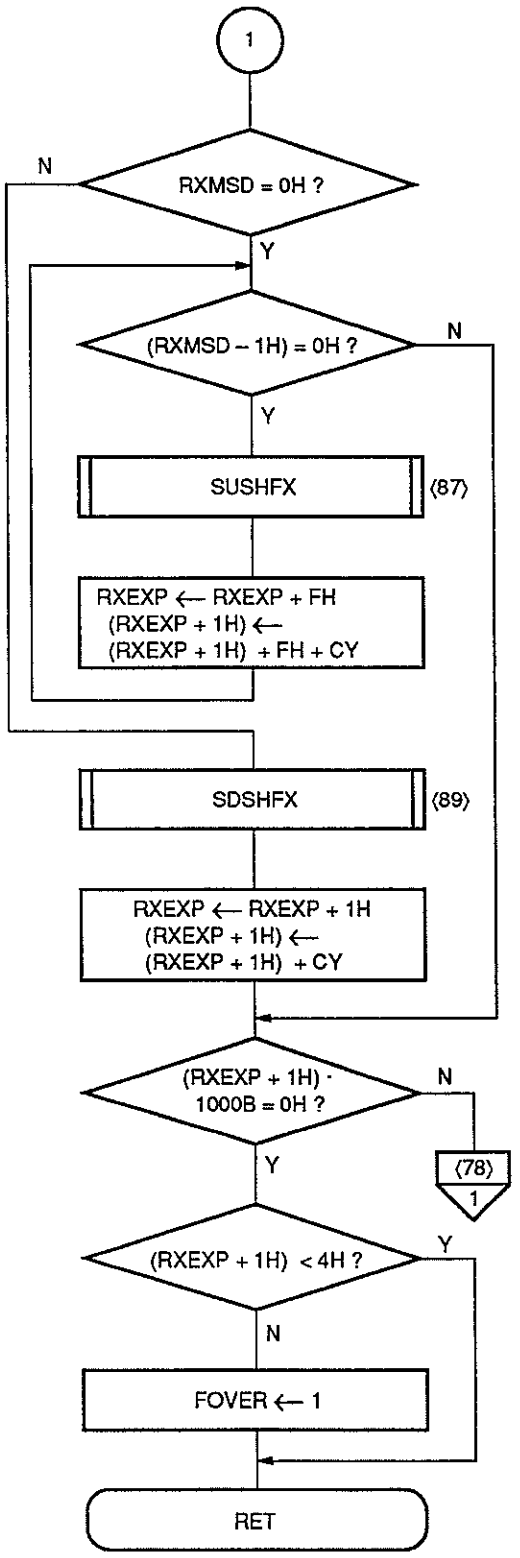
$= 6.91344197\underline{6} \times 10^{-1}$; Discards least significant digit.

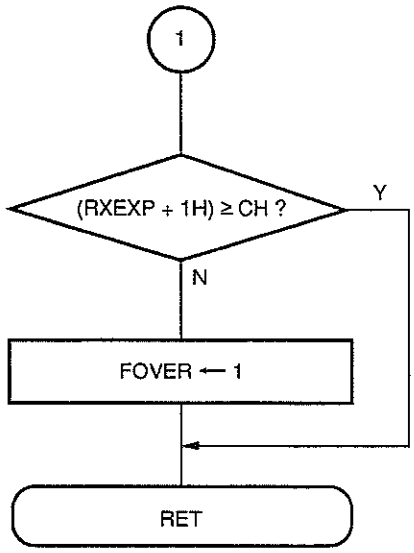
$= 0.691344197 \times 10^{0}$

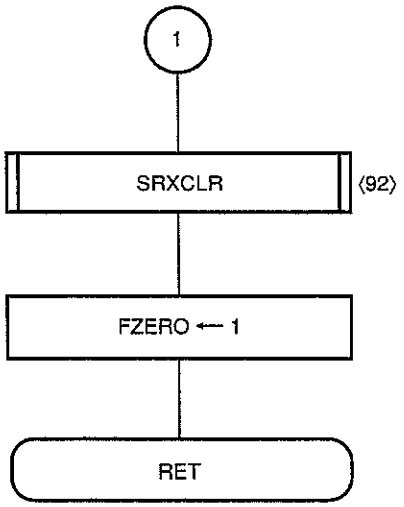
Remark In this manual introducing pocket calculator applications based on the arithmetic package, the overflow flag is ignored.

• Flowchart of normalization

Input variables	Flowchart ⟨76⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH)	SNML		
	FZERO ← 0	Clears operation result zero flag.	FZERO (1.30H.1)
	FOVER ← 0	Clears overflow flag.	FOVER (1.30H.3)
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH RPL
	IXH ← 0H IXM ← 8H IXL ← 6H	Sets index register at least significant digit of mantissa.	
	IXE ← 1	Check result of operation (mantissa of REGX) for zero.	
	RREG0 ← REGX : Operation result [IX]		
	IXE ← 0		
	RREG0 = 0H ? N → ⟨77⟩ 1		
	Y → IX ← IX + 1H		
	IXL = 0H ? N → IX ← IX + 1H		
	Y → ⟨79⟩ 1		

Input variables	Flowchart (77)	Processing and remarks	Output variable
		Integer part of result of operation being 0? Most significant digit of decimal fraction part of result of operation being 0? Shifts up result of operation. Decrements exponent by 1. Shifts down result of operation. Increments exponent by 1. Exponent positive? Exponent less than 64? Sets overflow flag. Overflow decision	REGX (1.03H–1.0FH) REGX (1.03H–1.0FH) FOVER (1.30H.3)

Input variables	Flowchart ⟨78⟩	Processing and remarks	Output variable
		Exponent equal to or greater than −64? Sets overflow flag. } Underflow decision	FOVER (1.30H.3)

Input variables	Flowchart ⟨79⟩	Processing and remarks	Output variable
		Clears result of operation. Sets operation result zero flag.	FZERO (1.30H.1)

8.3.2 Mantissa Addition (SADD)

- **Processing**

The mantissa of REGY is added to the mantissa of REGX in decimal, then the result of operation is stored in REGX. Decimal addition is performed by setting the BCD flag of the program status word.

- **Flowchart of mantissa addition**

Input variables	Flowchart (80)	Processing and remarks	Output variable
	SADD		
	IXH ← 0H IXL ← 0H	Clears index register.	
	RPH ← 1H RPL ← 1H	Sets general-purpose register at row address 0H of BANK1. Sets BCD flag.	RPH RPL
IXM	IXE ← 1		
REGX (1.03H–1.0FH)	REGX mantissa ← REGX mantissa + REGX mantissa [IX]	Adds index-modified register mantissa to mantissa of REGX in decimal.	REGX (1.03H–1.0FH)
	IXE ← 0		
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0. Clears BCD flag.	RPH RPL
	RET		

8.3.3 Mantissa Subtraction (SSUB)

- **Processing**

The mantissa of REGY or REGW is subtracted from the mantissa of REGX in decimal, then the result of operation is stored in REGX. Decimal subtraction is performed by setting the BCD flag of the program status word.

- **Flowchart of mantissa subtraction**

Input variables	Flowchart (81)	Processing and remarks	Output variable
	SSUB		
	IXH ← 0H IXL ← 0H	Clears index register.	
	RPH ← IH RPL ← 1H	Sets general-purpose register at row address 0H of BANK1. Sets BCD flag.	RPH RPL
IXM	IXE ← 1		
REGX (1.03H–1.0FH)	REGX mantissa ← REGX mantissa – REGX mantissa [IX]	Subtracts index-modified register mantissa from mantissa of REGX in decimal.	REGX (1.03H–1.0FH)
	IXE ← 0		
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0. Clears BCD flag.	RPH RPL
	RET		

8.3.4 Characteristic Addition (SADDEX)

- **Processing**

The characteristic of REGY is added to the characteristic of REGX in hexadecimal, then the result of operation is stored in the characteristic of REGX.

- **Flowchart of characteristic addition**

Input variables	Flowchart ⟨82⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH) REGY (1.13H–1.1FH)	SADDEX		
	RPH ← 1H RPL ← 0H	Sets general-purpose register at row address 0H of BANK1.	RPH RPL
	RXEXP ← RXEXP + RYEXP (RXEXP + 1H) ← (RXEXP + 1H) + (RYEXP + 1H) + CY	Adds characteristic of REGY to characteristic of REGX.	REGX (1.03H–1.0FH)
	RPH ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH
	RET		

8.3.5 Characteristic Subtraction (SSUBEX)

- **Processing**

  The characteristic of REGY is subtracted from the characteristic of REGX in hexadecimal, then the result of operation is stored in the characteristic of REGX.

- **Flowchart of characteristic subtraction**

Input variables	Flowchart ⟨83⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH) REGY (1.13H–1.1FH)	SSUBEX		
	RPH ← 1H RPL ← 0H	Sets general-purpose register at row address 0H of BANK1.	RPH RPL
	RXEXP ← RXEXP – RYEXP (RXEXP + 1H) ← (RXEXP + 1H) – (RYEXP + 1H) – CY	Subtracts characteristic of REGY from characteristic of REGX.	REGX (1.03H–1.0FH)
	RPH ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH
	RET		

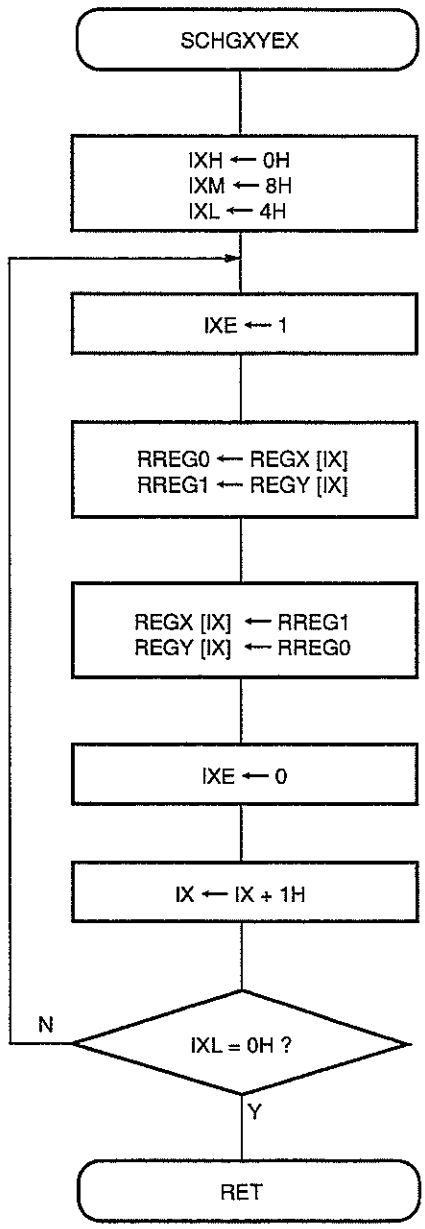
8.3.6 Register Exchange

(1) Register exchange 1 (SCHGXYEX)

- **Processing**

  The characteristic and mantissa of REGX are exchanged with the characteristic and mantissa of REGY.

- **Flowchart of register exchange 1 processing**

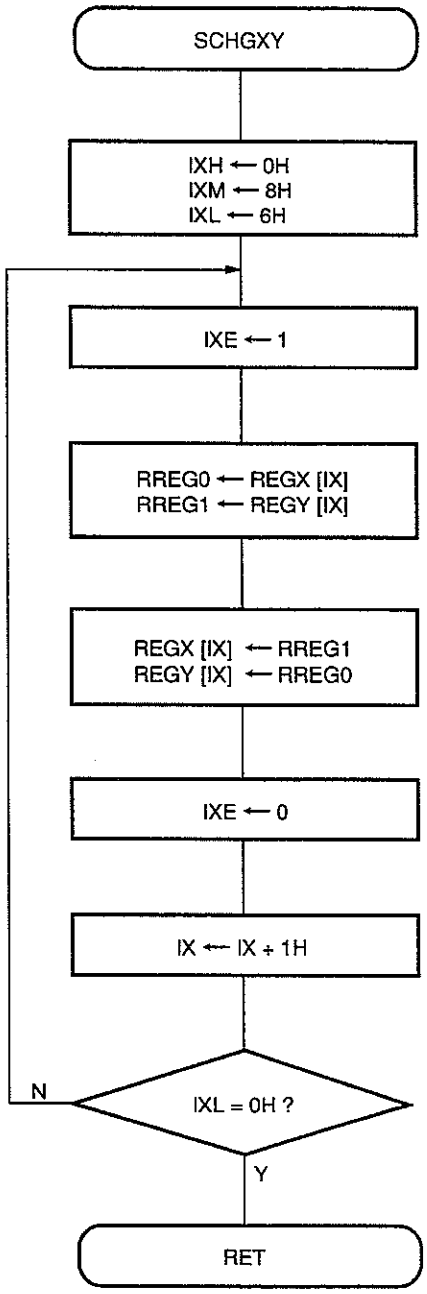
Input variables	Flowchart ⟨84⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH) REGY (1.13H–1.1FH)		Sets index register at column address 4H of BANK1. Exchanges characteristic and mantissa of REGX with characteristic and mantissa of REGY.	REGX (1.03H–1.0FH) REGY (1.13H–1.1FH)

(2) Register exchange 2 (SCHGXY)

- **Processing**

  The mantissa of REGX is exchanged with the mantissa of REGY.

- **Flowchart of register exchange 2 processing**

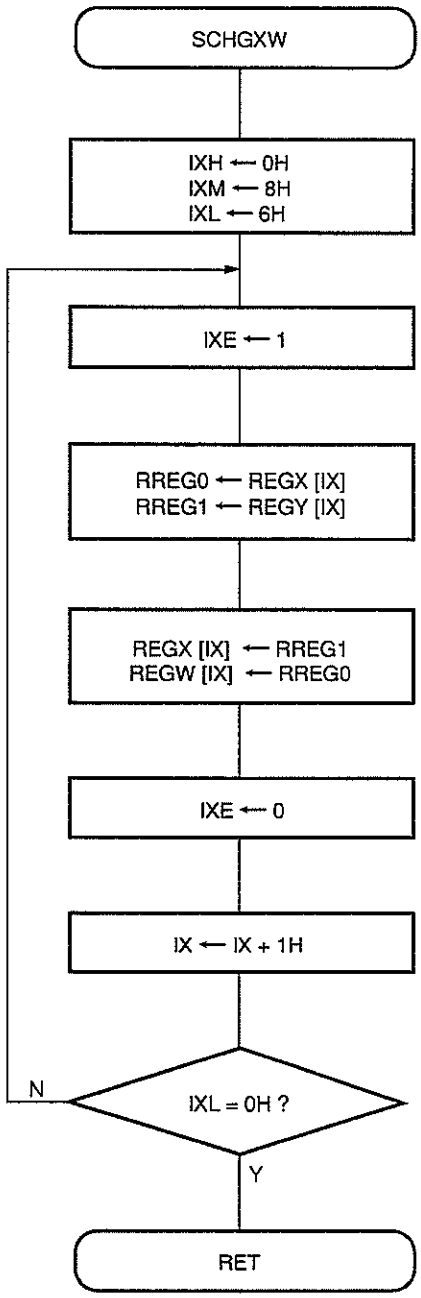
Input variables	Flowchart ⟨85⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH) REGY (1.13H–1.1FH)		Sets index register at column address 6H of BANK1.	
		Exchanges mantissa of REGX with mantissa of REGY.	REGX (1.03H–1.0FH) REGY (1.13H–1.1FH)

(3) Register exchange 3 (SCHGXW)

- **Processing**

  The mantissa of REGX is exchanged with the mantissa of REGW.

- **Flowchart of register exchange 3 processing**

Input variables	Flowchart ⟨86⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH) REGW (1.23H–1.2FH)		Sets index register at column address 6H of BANK1. Exchanges mantissa of REGX with mantissa of REGW.	REGX (1.03H–1.0FH) REGW (1.23H–1.2FH)

8.3.7 Register Shift-Up

(1) Register shift-up 1 (SUSHFX)

- **Processing**

  The mantissa of REGX is shifted up 1 digit.

- **Flowchart of register shift-up 1 processing**

Input variables	Flowchart ⟨87⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH)	SUSHFX		
	RPH ← 1H RPL ← 0H	Sets general-purpose register at row address 0H of BANK1.	RPH RPL
	RXMSD ← (RXMSD – 1H) (RXMSD – 1H) ← (RXMSD – 2H) (RXMSD – 2H) ← (RXMSD – 3H) (RXMSD – 3H) ← (RXMSD – 4H) (RXMSD – 4H) ← (RXMSD – 5H) (RXMSD – 5H) ← (RXMSD – 6H) (RXMSD – 6H) ← (RXMSD – 7H) (RXMSD – 7H) ← (RXMSD – 8H) (RXMSD – 8H) ← (RXMSD – 9H) RXLSD ← 0H	Shifts up mantissa of REGX 1 digit.	REGX (1.03H–1.0FH)
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH RPL
	RET		

(2) Register shift-up 2 (SUSHFW)

- **Processing**

  The mantissa of REGW is shifted up 1 digit.

- **Flowchart of register shift-up 2 processing**

Input variables	Flowchart (88)	Processing and remarks	Output variable
REGW (1.23H–1.2FH)	SUSHFW		
	RPH ← 1H RPL ← 4H	Sets general-purpose register at row address 2H of BANK1.	RPH RPL
	RWMSD ← (RWMSD – 1H) (RWMSD – 1H) ← (RWMSD – 2H) (RWMSD – 2H) ← (RWMSD – 3H) (RWMSD – 3H) ← (RWMSD – 4H) (RWMSD – 4H) ← (RWMSD – 5H) (RWMSD – 5H) ← (RWMSD – 6H) (RWMSD – 6H) ← (RWMSD – 7H) (RWMSD – 7H) ← (RWMSD – 8H) (RWMSD – 8H) ← (RWMSD – 9H) RWLSD ← 0H	Shifts up mantissa of REGW 1 digit.	REGW (1.23H–1.2FH)
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH RPL
	RET		

8.3.8 Register Shift-Down

(1) Register shift-down 1 (SDSHFX)

- **Processing**

  The mantissa of REGX is shifted down 1 digit.

- **Flowchart of register shift-down 1 processing**

Input variables	Flowchart (89)	Processing and remarks	Output variable
	SDSHFX		
	RPH ← 1H RPL ← 0H	Sets general-purpose register at row address 0H of BANK1.	RPH RPL
REGX (1.03H–1.0FH)	RXLSD ← (RXLSD + 1H) (RXLSD + 1H) ← (RXLSD + 2H) (RXLSD + 2H) ← (RXLSD + 3H) (RXLSD + 3H) ← (RXLSD + 4H) (RXLSD + 4H) ← (RXLSD + 5H) (RXLSD + 5H) ← (RXLSD + 6H) (RXLSD + 6H) ← (RXLSD + 7H) (RXLSD + 7H) ← (RXLSD + 8H) (RXLSD + 8H) ← (RXLSD + 9H) RXMSD ← 0H	Shifts down mantissa of REGX 1 digit.	REGX (1.03H–1.0FH)
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH RPL
	RET		

(2) Register shift-down 2 (SDSHFY)

- **Processing**

  The mantissa of REGY is shifted down 1 digit.

- **Flowchart of register shift-down 2 processing**

Input variables	Flowchart (90)	Processing and remarks	Output variable
REGY (1.13H–1.1FH)	SDSHFY		
	RPH ← 1H RPL ← 4H	Sets general-purpose register at row address 1H of BANK1.	RPH RPL
	RYLSD ← (RYLSD + 1H) (RYLSD + 1H) ← (RYLSD + 2H) (RYLSD + 2H) ← (RYLSD + 3H) (RYLSD + 3H) ← (RYLSD + 4H) (RYLSD + 4H) ← (RYLSD + 5H) (RYLSD + 5H) ← (RYLSD + 6H) (RYLSD + 6H) ← (RYLSD + 7H) (RYLSD + 7H) ← (RYLSD + 8H) (RYLSD + 8H) ← (RYLSD + 9H) RYMSD ← 0H	Shifts down mantissa of REGY 1 digit.	REGY (1.13H–1.1FH)
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH RPL
	RET		

(3) Register shift-down 3 (SDSHFW)

- **Processing**

  The mantissa of REGW is shifted down 1 digit.

- **Flowchart of register shift-down 3 processing**

Input variables	Flowchart ⟨91⟩	Processing and remarks	Output variable
REGW (1.23H–1.2FH)	SDSHFW		
	(RWEXP + 1H) ← RWLSD	Saves least significant digit (up-counter) of mantissa.	
	RPH ← 1H RPL ← 4H	Sets general-purpose register at row address 2H of BANK1.	RPH RPL
	RWLSD ← (RWLSD + 1H) (RWLSD + 1H) ← (RWLSD + 2H) (RWLSD + 2H) ← (RWLSD + 3H) (RWLSD + 3H) ← (RWLSD + 4H) (RWLSD + 4H) ← (RWLSD + 5H) (RWLSD + 5H) ← (RWLSD + 6H) (RWLSD + 6H) ← (RWLSD + 7H) (RWLSD + 7H) ← (RWLSD + 8H) (RWLSD + 8H) ← (RWLSD + 9H) RWMSD ← 0H	Shifts down mantissa of REGW 1 digit.	REGW (1.23H–1.2FH)
	RPH ← 0H RPL ← 0H	Sets general-purpose register at row address 0H of BANK0.	RPH RPL
	RET		

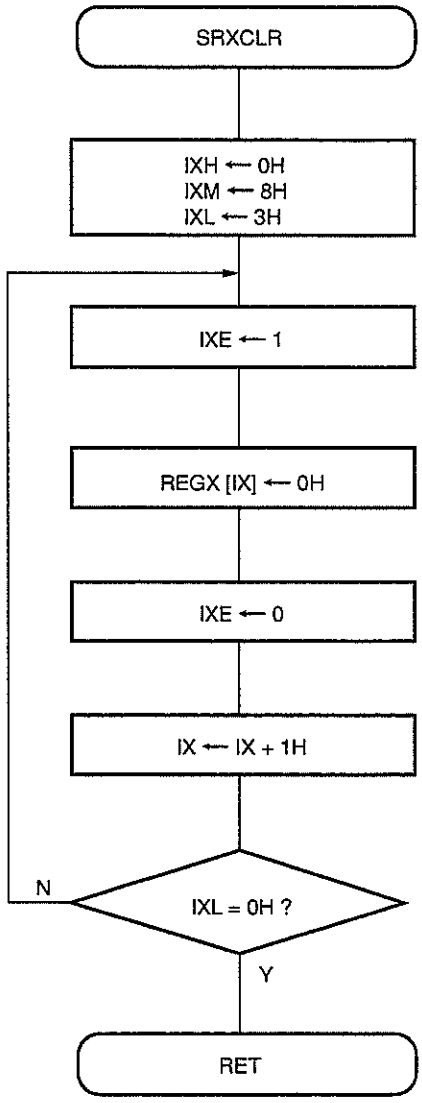
8.3.9 Register Clear (0)

(1) Register clear (0) 1 (SRXCLR)

- **Processing**

  The sign part, characteristic, and mantissa of REGX are cleared to 0.

- **Flowchart of register clear (0) 1 processing**

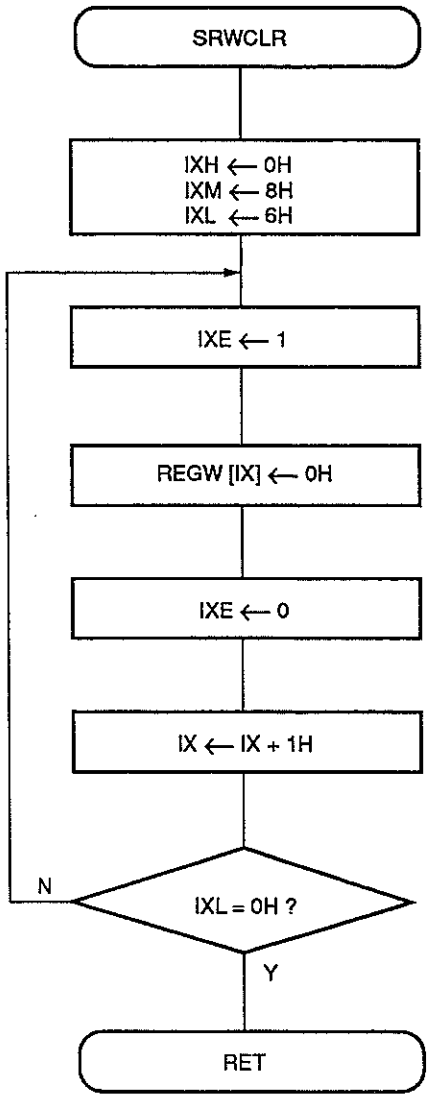
Input variables	Flowchart ⟨92⟩	Processing and remarks	Output variable
REGX (1.03H–1.0FH)		Sets index register at column address 3H of BANK1. Clears sign part, characteristic, and mantissa of REGX to 0.	REGX (1.03H–1.0FH)

(2) Register clear (0) 2 (SRWCLR)

- **Processing**

  The mantissa of REGW is cleared to 0.

- **Flowchart of register clear (0) 2 processing**

Input variables	Flowchart (93)	Processing and remarks	Output variable
REGW (1.23H–1.2FH)		Sets index register at column address 6H of BANK1. Clears mantissa of REGW to 0.	REGW (1.23H–1.2FH)

Chapter 9
Calculator Programs

9.1 System Control Section Program

This section shows the program listing of the system control section of the calculator described in the application notes.

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-001

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
     1                 ;*********************************************
     2                 ;*                                           *
     3                 ;*  USER NAME   : NEC CORPORATION            *
     4                 ;*                                           *
     5                 ;*  SYSTEM NAME : 17K APPLICATION (CALCULATOR) *
     6                 ;*                                           *
     7                 ;*  CPU         : µPD17201A                  *
     8                 ;*                                           *
     9                 ;*  LAST UPDATE : '93/12/20 11:00            *
    10                 ;*                                           *
    11                 ;*  FILE NAME   : CALCDEF.ASM                *
    12                 ;*                                           *
    13                 ;*********************************************
    14                 ;*********************************************
    15                 ;*                                           *
    16                 ;*        Control register definitions       *
    17                 ;*                                           *
    18                 ;*********************************************
    19                        PUBLIC   BELOW
    20      0082       PCC    MEM      0.82H        ;System clock control register
    21      0083       WTC    MEM      0.83H        ;Clock-timer/watch-dog-timer
    22                                               ;control register
    23      00A1       ADM    MEM      0.0A1H       ;A/D converter operation mode/pin selection
    24      00A2       SIC    MEM      0.0A2H       ;Serial interface control register
    25      00A3       PSL    MEM      0.0A3H       ;Port/timer output, LED output, serial
    26                                               ;interface input/output control register
    27      00A7       PM0    MEM      0.0A7H       ;Bit I/O control register
    28      00AF       IPF    MEM      0.0AFH       ;Interrupt enable flag
    29      00B1       LCDC   MEM      0.0B1H       ;LCD controller/driver control register
    30      00B2       LCDM   MEM      0.0B2H       ;Display mode register
    31      00B3       TMCT   MEM      0.0B3H       ;8-bit timer/counter control register
    32      00B7       PM1    MEM      0.0B7H       ;Group I/O control register
    33
    34                        EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-002

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    35                 ;*********************************************
    36                 ;*                                           *
    37                 ;*                RAM definition             *
    38                 ;*                                           *
    39                 ;*********************************************
    40                 ;*********************************************
    41                 ;*                    BANK0                  *
    42                 ;*********************************************
    43                 ;+++    General-purpose registers          +++
    44      0000       RREG0   MEM      0.00H
    45      0001       RREG1   MEM      0.01H
    46      0002       RREG2   MEM      0.02H
    47      0003       RREG3   MEM      0.03H
    48      0004       RREG4   MEM      0.04H
    49                 ;RREG5  MEM      0.05H
    50                 ;RREG6  MEM      0.06H
    51                 ;RREG7  MEM      0.07H
    52                 ;RREG8  MEM      0.08H
    53      0009       RREG9   MEM      0.09H
    54      000A       RREG10  MEM      0.0AH
    55      000B       RREG11  MEM      0.0BH
    56
    57                 ;+++    Display data registers (REGD)       +++
    58      0010       RNUMC   MEM      0.10H         ;Numeric key counter
    59      0011       RPTLOC  MEM      0.11H         ;Decimal-point position area
    60      0012       RSINLOC MEM      0.12H         ;Sign position area
    61      0013       RDSIGN  MEM      0.13H         ;Operation result sign area
    62      0014       RDEXP   MEM      0.14H         ;Display data exponent area
    63      0018       RDLSD   MEM      0.18H         ;Display data area (least significant digit)
    64      001F       RDMSD   MEM      0.1FH         ;Display data area (most significant digit)
    65
    66                 ;+++    Saving registers (REGZ)             +++
    67      0023       RZLSD   MEM      0.23H
    68                 ;RZMSD  MEM      0.2FH
    69
    70                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-003

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    71                 ;+++            Key scan                    +++
    72      0030       RKCODH  MEM     0.30H       ;Key code area (4 high-order bits)
    73      0031       RKCODL  MEM     0.31H       ;Key code area (4 low-order bits)
    74      0032       RCHCODH MEM     0.32H       ;Chattering code area (4 high-order bits)
    75
    76      0033       RCHCODL MEM     0.33H       ;Chattering code area (4 low-order bits)
    77      0034       RIPCODH MEM     0.34H       ;Input code area (4 high-order bits)
    78      0035       RIPCODL MEM     0.35H       ;Input code area (4 low-order bits)
    79      0036       RCHATC  MEM     0.36H       ;Chattering counter
    80
    81                 ;+++    Automatic-power-off timer           +++
    82      0038       RSTOP3  MEM     0.38H       ;3-minute counter (10 ms x 18.000)
    83      0039       RSTOP2  MEM     0.39H
    84      003A       RSTOP1  MEM     0.3AH
    85      003B       RSTOP0  MEM     0.3BH
    86
    87                 ;+++            Operator                    +++
    88      0060       ROPE    MEM     0.60H       ;Operator area
    89      0061       RCOM    MEM     0.61H       ;Area for the operator to be executed
    90
    91                 ;+++            Mode
    92      0064       RMODE   MEM     0.64H
    93
    94                 ;+++    System flag area                    +++
    95      0068       RSYSFLG MEM     0.68H
    96      0681       FPER    FLG     RSYSFLG.0   ;Percent flag
    97      0682       FOPEND  FLG     RSYSFLG.1   ;Operation end flag
    98      0684       FFALSE  FLG     RSYSFLG.2   ;Illegal-input flag
    99      0688       FSTOP   FLG     RSYSFLG.3   ;Operation stop mode flag
   100
   101                 ;+++    Event flag area                     +++
   102      0069       REVEFLG MEM     0.69H
   103      0691       FKEYREQ FLG     REVEFLG.0   ;Key-processing request flag
   104      0692       FMULTI  FLG     REVEFLG.1   ;Multiple key press flag
   105      0694       FOPREQ  FLG     REVEFLG.2   ;Operation request flag
   106      0698       FSTPREQ FLG     REVEFLG.3   ;STOP mode processing request flag
   107
   108                 ;+++    Port registers                      +++
   109      0070       P0A     MEM     0.70H
   110      0071       P0B     MEM     0.71H
   111      0072       P0C     MEM     0.72H
   112      0073       P0D     MEM     0.73H
   113      0170       PIA     MEM     1.70H
   114
   115                        EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-004

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   116                 ;*********************************************
   117                 ;*                    BANK1                    *
   118                 ;*********************************************
   119                 ;+++    Floating-point register 1      +++
   120      0103       RXSIGN  MEM      1.03H                 ;Sign part
   121      0104       RXEXP   MEM      1.04H                 ;Characteristic
   122      0106       RXLSD   MEM      1.06H                 ;Mantissa (least significant digit)
   123      010F       RXMSD   MEM      1.0FH                 ;Mantissa (most significant digit)
   124
   125                 ;+++    Floating-point register 2      +++
   126      0113       RYSIGN  MEM      1.13H                 ;Sign part
   127      0114       RYEXP   MEM      1.14H                 ;Characteristic
   128      0116       RYLSD   MEM      1.16H                 ;Mantissa (least significant digit)
   129      011F       RYMSD   MEM      1.1FH                 ;Mantissa (most significant digit)
   130
   131                 ;+++    Floating-point register 3      +++
   132      0123       RWSIGN  MEM      1.23H                 ;Sign part
   133      0124       RWEXP   MEM      1.24H                 ;Characteristic
   134      0126       RWLSD   MEM      1.26H                 ;Mantissa (least significant digit)
   135      012F       RWMSD   MEM      1.2FH                 ;Mantissa (most significant digit)
   136
   137                 ;+++    Operation flags                +++
   138      0130       ROPFLG  MEM      1.30H
   139      1301       FEXCHG  FLG      ROPFLG.0              ;Register exchange flag
   140      1302       FZERO   FLG      ROPFLG.1              ;Operation result zero flag
   141      1304       FDVERR  FLG      ROPFLG.2              ;Zero-division error flag
   142      1308       FOVER   FLG      ROPFLG.3              ;Overflow flag
   143
   144                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-005

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   145                 ;*********************************************
   146                 ;*                                           *
   147                 ;*        Constant value definition          *
   148                 ;*                                           *
   149                 ;*********************************************
   150                 ;*********************************************
   151                 ;*                  Hardware                 *
   152                 ;*********************************************
   153                 ;+++        8-bit modulo register            +++
   154      009C       CRLHLT  DAT      9CH            ;Releases the HALT mode (10 ms).
   155      0000       CRLSTP  DAT      0H             ;Releases the STOP mode.
   156
   157                 ;*********************************************
   158                 ;*              System register              *
   159                 ;*********************************************
   160                 ;+++    Data memory row address
   161                         pointer (MP)                          +++
   162      0008       CSTMPE  DAT      1000B          ;MPH:  Indirect addressing (BANK0)
   163      0009       CSTBK2  DAT      1001B          ;MPH:  Indirect addressing (BANK2)
   164      0008       CSTBK1  DAT      1000B          ;MPL:  Indirect addressing (BANK1)
   165
   166                 ;*********************************************
   167                 ;*      Port input/output data/key code      *
   168                 ;*********************************************
   169                 ;+++    Key scan signal
   170                 ;              -> Key code (4 high-order bits)  +++
   171      000E       CKSRB0  DAT      1110B          ;P0B0 output
   172      000D       CKSRB1  DAT      1101B          ;P0B1 output
   173      000B       CKSRB2  DAT      1011B          ;P0B2 output
   174      0007       CKSRB3  DAT      0111B          ;P0B3 output
   175      0000       CKSRC0  DAT      0000B          ;P0C0 output
   176      0000       CALLLOW DAT      0000B
   177      000F       CALLHI  DAT      1111B
   178      0001       C0C0HI  DAT      0001B
   179
   180                 ;+++    Key return value
   181                 ;              -> Key code (4 low-order bits)   +++
   182      000E       CKRET0  DAT      1110B          ;P0A0 input
   183      000D       CKRET1  DAT      1101B          ;P0A1 input
   184      000B       CKRET2  DAT      1011B          ;P0A2 input
   185      0007       CKRET3  DAT      0111B          ;P0A3 input
   186
   187                 ;+++             Key code (others)            +++
   188      00F0       CKMLT   DAT      0F0H           ;Multiple key press
   189      00FF       CKOFF   DAT      0FFH           ;Key-off
   190
   191                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-006

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   192                 ;*********************************************
   193                 ;*                  RAM data                 *
   194                 ;*********************************************
   195                 ;+++    Numeric key counter                  +++
   196      0000       CNONUM  DAT      0H         ;Number of numeric key inputs: 0
   197      0007       CFLNUM  DAT      7H         ;Number of numeric key inputs: 7
   198
   199                 ;+++    Decimal-point display position area  +++
   200      0007       CPLINI  DAT      7H         ;Initial value of the decimal-point display position area
   201
   202                 ;+++    Display data exponent area           +++
   203      0008       CEXPINI DAT      08H        ;Initial value of the display data exponent area
   204
   205                 ;+++    Overflow judgment                    +++
   206      00FB       CEXPMIN DAT      0FBH       ;Range of exponent: -5 to 7
   207      0007       CEXPMAX DAT      07H
   208
   209                 ;+++    Sign data                            +++
   210      0000       CSINOFF DAT      0000B      ;No sign display
   211      0001       CSINMN  DAT      0001B      ;Minus sign display
   212      0001       CCHSIN  DAT      0001B      ;Sign inversion
   213
   214                        EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-007

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   215                  ;+++        Display data                    +++
   216      0000        CZERO   DAT     0H          ;[0]
   217      0001        CONE    DAT     1H          ;[1]
   218      0002        CTWO    DAT     2H          ;[2]
   219      0003        CTHREE  DAT     3H          ;[3]
   220      0004        CFOUR   DAT     4H          ;[4]
   221      0005        CFIVE   DAT     5H          ;[5]
   222      0006        CSIX    DAT     6H          ;[6]
   223      0007        CSEVEN  DAT     7H          ;[7]
   224      0008        CEIGHT  DAT     8H          ;[8]
   225      0009        CNINE   DAT     9H          ;[9]
   226      000A        CSPACE  DAT     0AH         ;Space
   227      000B        CMINUS  DAT     0BH         ;[-]
   228      000C        CERRDP  DAT     0CH         ;[E]
   229
   230                  ;+++        Chattering                      +++
   231      0002        CCHATS  DAT     2H          ;Start of chattering
   232      0000        CCHATE  DAT     0H          ;End chattering
   233
   234                  ;+++        Automatic-power-off timer       +++
   235      464F        CSTSTP  DAT     464FH       ;3-minute timer (10 ms x 18,000)
   236                                              ;       -> Sets the STOP mode.
   237
   238                  ;+++        LCD segment data                +++
   239      0001        CDPON   DAT     0001B       ;Decimal-point display
   240
   241                  ;+++            Operators                   +++
   242      0001        CADD    DAT     0001B       ;[+]
   243      0002        CSUB    DAT     0010B       ;[-]
   244      0004        CMUL    DAT     0100B       ;[x]
   245      0008        CDIV    DAT     1000B       ;[÷]
   246
   247                  ;+++        Modes                           +++
   248      0001        CRIGHT  DAT     0001B       ;Second-term input mode
   249      0002        COPSEL  DAT     0010B       ;Operator selection mode
   250      0004        CLEFT   DAT     0100B       ;First-term input mode
   251      0008        CERROR  DAT     1000B       ;Error mode
   252
   253                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-008

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   254                 ;*********************************************
   255                 ;*             Floating-point unit             *
   256                 ;*********************************************
   257      0000       CGB0     DAT     0H      ;General-purpose register BANK 0
   258      0001       CGB1     DAT     1H      ;General-purpose register BANK 1
   259      0000       CROWG    DAT     0H      ;General-purpose register row address 0H
   260      0000       CROWX    DAT     0H      ;General-purpose register row address 0H
   261      0002       CROWY    DAT     2H      ;General-purpose register row address 10H
   262      0004       CROWW    DAT     4H      ;General-purpose register row address 20H
   263      0001       CROWXBCD DAT     1H      ;General-purpose register row address 0H
   264                                          ;                Sets the BCD flag.
   265      0001       CIXMRY   DAT     1H      ;Index modification (REGY)
   266      0002       CIXMRW   DAT     2H      ;Index modification (REGW)
   267      0008       CIXMB1   DAT     8H      ;Index modification (BANK1)
   268      0006       CIXLMANT DAT     6H      ;Index modification (mantissa LSD)
   269      0004       CIXLEXP  DAT     4H      ;Index modification (characteristic)
   270      0003       CIXLSIGN DAT     3H      ;Index modification (sign part)
   271      0000       CMDRX    DAT     0H    ;Column address to be indexed  0H
   272      0010       CMDRY    DAT     10H   ;Column address to be indexed  10H
   273      0020       CMDRW    DAT     20H   ;Column address to be indexed  20H
   274      0008       CJUDGE   DAT     1000B   ;Positive/negative judgment of the characteristic value
   275      000F       CCPTURN  DAT     1111B   ;Bit inversion mask
   276      0001       CSUBTURN DAT     0001B   ;Subtrahend sign inversion
   277      0001       CRXTURN  DAT     0001B   ;REGX sign inversion
   278      000A       CEXDIF   DAT     0AH     ;Exponent difference judgment
   279      0001       CSIGNCK  DAT     0001B   ;Sign part check
   280      000A       CMANTCNT DAT     0AH     ;Ten-digit counter for mantissa
   281      0004       CEXOVER  DAT     0100B   ;Characteristic value overflow
   282      000C       CEXUNDER DAT     1100B   ;Characteristic value underflow
   283
   284                         ENDP
   285
   286                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:22:42 12/20/93 PAGE 01-009

PROG =

SOURCE = CALCDEF.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   287                 ;*********************************************
   288                 ;*                                           *
   289                 ;*           Mask option definition          *
   290                 ;*                                           *
   291                 ;*********************************************
   292                         OPTION
   293            1            OPTRES   RESPLUP      ;Reset pin:  Has built-in pull-up resistor
 +  12      0001   1                                 ;System clock:  Main clock
   294            1            OPTCK    USEX,NOXT
 +  24      0006   1
   295                         ENDOP
   296
   297                         END

 TOTAL ERRORS   = 0
 TOTAL WARNINGS = 0

 END OF LIST
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-001

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
     1                 ;*************************************************
     2                 ;*                                               *
     3                 ;*  USER NAME    : NEC CORPORATION               *
     4                 ;*                                               *
     5                 ;*  SYSTEM NAME : 17K APPLICATION (CALCULATOR)  *
     6                 ;*                                               *
     7                 ;*  CPU          : µPD17201A                     *
     8                 ;*                                               *
     9                 ;*  LAST UPDATE : '93/12/20  11:                 *
    10                 ;*                                               *
    11                 ;*************************************************
    12                 ;*************************************************
    13                 ;*                                               *
    14                 ;*   FILE NAME        : CALC1.ASM                *
    15                 ;*                                               *
    16                 ;*   INCLUDE 7 ROUTINES:                         *
    17                 ;*            IRESET   : RESET ROUTINE           *
    18                 ;*            MMAIN    : MAIN ROUTINE            *
    19                 ;*            MKSCAN   : KEY SCAN                *
    20                 ;*            SKRET    : KEY RETURN              *
    21                 ;*            SMKIPC   : MAKE INPUT CODE         *
    22                 ;*            MAPOFF   : AUTO POWER OFF          *
    23                 ;*                             COUNTER CONTROL   *
    24                 ;*            MSTOP    : STOP MODE PROCESS       *
    25                 ;*                                               *
    26                 ;*************************************************
    27
    28                      EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-002

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    29                 ;*********************************************
    30                 ;*              External reference           *
    31                 ;*********************************************
    32                 EXTRN   LAB: MKBRN            ;Branches to processing for each key
    33                 EXTRN   LAB: MOPBRN           ;Branch operation processing/error handling
    34                 EXTRN   LAB: SRAMCR           ;RAM all-clear processing
    35                 EXTRN   LAB: SDPINI           ;Display data area initialization
    36                 EXTRN   LAB: SDISP            ;Display data output processing
    37
    38                 EXTRN   MEM: PCC       ;System clock control register
    39                 EXTRN   MEM: WTC       ;Clock-timer/watch-dog-timer control register
    40                 EXTRN   MEM: ADM       ;A/D converter operation mode/pin selection
    41                 EXTRN   MEM: SIC       ;Serial interface control register
    42                 EXTRN   MEM: PSL       ;Port/timer output, LED output, serial interface
    43                                        ;input/output control register
    44                 EXTRN   MEM: PM0       ;Bit I/O control register
    45                 EXTRN   MEM: IPF       ;Interrupt enable flag
    46                 EXTRN   MEM: LCDC      ;LCD controller/driver control register
    47                 EXTRN   MEM: LCDM      ;Display mode register
    48                 EXTRN   MEM: TMCT      ;8-bit timer/counter control register
    49                 EXTRN   MEM: PM1       ;Group I/O control register
    50
    51                 EXTRN   MEM: RREG0,RREG1,RREG2,RREG9   ;General-purpose register
    52                 EXTRN   MEM: RDEXP                     ;Display data exponent area
    53                 EXTRN   MEM: RKCODH,RKCODL             ;Key code area
    54                 EXTRN   MEM: RCHCODH,RCHCODL           ;Chattering code area
    55                 EXTRN   MEM: RIPCODH,RIPCODL           ;Input code area
    56                 EXTRN   MEM: RCHATC                    ;Chattering counter
    57                 EXTRN   MEM: RSTOP3,RSTOP2             ;Automatic-power-off timer
    58                 EXTRN   MEM: RSTOP1,RSTOP0
    59                 EXTRN   MEM: RMODE                     ;Mode area
    60                 EXTRN   MEM: P0A,P0B,P0C,P0D,P1A       ;Port register
    61
    62                 EXTRN   FLG: FSTOP                     ;STOP mode flag
    63                 EXTRN   FLG: FKEYREQ                   ;Key-processing request flag
    64                 EXTRN   FLG: FMULTI                    ;Multiple-key flag
    65                 EXTRN   FLG: FOPREQ                    ;Operation request flag
    66                 EXTRN   FLG: FSTPREQ                   ;STOP mode processing request flag
    67
    68                 EXTRN   DAT: CKSRB3,CKSRC0                        ;Port output data
    69                 EXTRN   DAT: CALLOW,CALLHI,C0C0HI
    70                 EXTRN   DAT: CKRET0,CKRET1,CKRET2,CKRET3          ;Port input data
    71                 EXTRN   DAT: CKMLT,CKOFF                          ;Key code
    72                 EXTRN   DAT: CRLHLT                    ;8-bit modulo
    73                 EXTRN   DAT: CRLSTP                    ;register data
    74                 EXTRN   DAT: CEXPINI                   ;Initial value of the display data exponent area
    75                 EXTRN   DAT: CCHATS,CCHATE             ;Chattering start/end count
    76                 EXTRN   DAT: CSTSTP                    ;Automatic-power-off count
    77                 EXTRN   DAT: CLEFT                     ;First-term input mode
    78
    79                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-003

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    80                 ;*********************************************
    81                 ;*                                           *
    82                 ;*              Reset processing             *
    83                 ;*                                           *
    84                 ;*********************************************
    85 0000
    86 0000 0C005              ORG     000H
    87                         BR      IRESET
    88
    89 0001 0C005      ;+++    Interrupt vector address       +++
    90 0002 0C005              BR      IRESET
    91 0003 0C005              BR      IRESET
    92 0004 0C005              BR      IRESET
    93                         BR      IRESET
    94
    95 0005 071F0      IRESET:
    96 0006 1D785              DI
    97 0007 07021              MOV     WR,#5H
    98 0008 1D783              POKE    SP,WR          ;System clock:  Main clock
    99 0009 07022              MOV     WR,#3H
   100                         POKE    PCC,WR
   101                 *********************************************
   102                 ;*              Port initialization          *
   103                 ;*********************************************
   104            1    ;       BANK1
 +   1 000A 1D791  1           MOV     BANK,#01H
   105 000B 1D700              MOV     P1A,#CALLOW AND 0FH
   106            1            BANK0                          ;P1A:  Not used
 +   1 000C 1D790  1           MOV     BANK,#00H
   107 000D 1D71F              MOV     P0B,#CALLHI AND 0FH
   108 000E 1D721              MOV     P0C,#C0C0HI AND 0FH    ;P0B, P0C0:  Key source
   109 000F 1D730              MOV     P0D,#CALLOW AND 0FH    ;(Active low)
   110 0010 1D78E              MOV     WR,#1110B              ;P0D:  Not used
   111 0011 07327              POKE    PM1,WR                 ;P0B, P0C, P1A:  Output
   112 0012 1D78F              MOV     WR,#1111B              ;P0A:  Input (key return)
   113 0013 07227              POKE    PM0,WR                 ;P0D0—P0D3:  Output
   114
   115                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-004

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   116                 ;*********************************************
   117                 ;*            Timer initialization           *
   118                 ;*********************************************
   119                 ;+++         8-bit timer                     +++
   120 0014 1D786              MOV     WR,#0110B           ;Operating clock cycle:  fSYS/256
   121 0015 07323              POKE    TMCT,WR
   122 0016 1D0FC              MOV     DBF0,#CRLHLT AND 0FH            ;Modulo register
   123 0017 1D0E9              MOV     DBF1,#CRLHLT SHR 4 AND 0FH      ;reset (10 ms)
   124 0018 070A2              PUT     TMM,DBF
   125
   126                 ;+++         Clock timer/watch dog timer     +++
   127 0019 1D780              MOV     WR,#0000B                       ;Clock timer:  Not used
   128 001A 07023              POKE    WTC,WR
   129
   130                 ;*********************************************
   131                 ;*      LCD controller/driver initialization   *
   132                 ;*********************************************
   133 001B 1D781              MOV     WR,#0001B           ;Display mode:  1/2 duty
   134 001C 07322              POKE    LCDM,WR
   135                 ;       MOV     WR,#0001B           ;Frame frequency:  512 Hz
   136 001D 07321              POKE    LCDC,WR
   137
   138                 ;*********************************************
   139                 ;*      Initialization of other hardware       *
   140                 ;*********************************************
   141                 ;+++         A/D converter                   +++
   142 001E 1D780              MOV     WR,#0000B                       ;Operation mode:
   143 001F 07221              POKE    ADM,WR                          ;Standby status
   144
   145                 ;+++         Serial interface                +++
   146                 ;       MOV     WR,#0000B                       ;Not used
   147 0020 07223              POKE    PSL,WR
   148                 ;       MOV     WR,#0000B
   149 0021 07222              POKE    SIC,WR
   150
   151                 ;*********************************************
   152                 ;*          Interrupt initialization           *
   153                 ;*********************************************
   154                 ;       MOV     WR,#0000B                       ;Disables all interrupts.
   155 0022 0722F              POKE    IPF,WR
   156
   157                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-005

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   158                 ;*********************************************
   159                 ;*              RAM initialization             *
   160                 ;*********************************************
   161 0023 1D7D0      ;       MOV     RPH,#0000B              ;General-purpose register:
   162 0024 1D7E0              MOV     RPL,#0000B              ; 0.00H-0.0FH
   163 0025 EXTRN              CALL    SRAMCR                  ;RAM all clear
   164
   165                 ;+++    Automatic-power-off timer reset      +++
   166 0026 1D3BF              MOV     RSTOP0,#CSTSTP AND 0FH          ;10ms*18000 count
   167 0027 1D3A4              MOV     RSTOP1,#CSTSTP SHR 4 AND 0FH
   168 0028 1D396              MOV     RSTOP2,#CSTSTP SHR 8 AND 0FH
   169 0029 1D384              MOV     RSTOP3,#CSTSTP SHR 12 AND 0FH
   170
   171 002A 1D148              MOV     RDEXP,#CEXPINI AND 0FH
   172 002B 1D31F              MOV     RKCODL,#CKOFF AND 0FH           ;Key code
   173 002C 1D30F              MOV     RKCODH,#CKOFF SHR 4 AND 0FH     ;Chattering code
   174 002D 1D33F              MOV     RCHCODL,#CKOFF AND 0FH          ;<- Key-off code
   175 002E 1D32F              MOV     RCHCODH,#CKOFF SHR 4 AND 0FH
   176 002F 1D644              MOV     RMODE,#CLEFT AND 0FH
   177
   178 0030 EXTRN              CALL    SDPINI          ;Display data area initialization
   179 0031 EXTRN              CALL    SDISP           ;Display data output
   180             1           SET1    LCDEN           ;LCD display on
 +   1 0032 07331  1           PEEK    WR,.MF.LCDEN SHR 4
 +   2 0033 16788  1           OR      WR,#.DF.LCDEN AND 0FH
 +   3 0034 07321  1           POKE    .MF.LCDEN SHR 4,WR
   181             1           SET1    TMEN            ;8-bit timer count start
 +   1 0035 07333  1           PEEK    WR,.MF.TMEN SHR 4
 +   2 0036 16788  1           OR      WR,#.DF.TMEN AND 0FH
 +   3 0037 07323  1           POKE    .MF.TMEN SHR 4,WR
   182
   183                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-006

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   184                 ;*********************************************
   185                 ;*                                           *
   186                 ;*              Main processing              *
   187                 ;*                                           *
   188                 ;*********************************************
   189                 MMAIN:
   190            1            CLR1    IRQTM
 +   1 0038 1D780  1            MOV     WR,#0000B
 +   2 0039 0732E  1            POKE    .MF.IRQTM SHR 4,WR
   191                 LMN200:
   192 003A 073F2               HALT    2H                      ;Executes a release every 10 ms.
   193            1            SKT1    IRQTM
 +   1 003B 0733E  1            PEEK    WR,.MF.IRQTM SHR 4
 +   2 003C 1E781  1            SKT     WR,#.DF.IRQTM AND 0FH
   194 003D 0C03A               BR      LMN200
   195            1            SET1    WDTRES                  ;Resets the watch dog timer.
 +   1 003E 07033  1            PEEK    WR,.MF.WDTRES SHR 4
 +   2 003F 16788  1            OR      WR,#.DF.WDTRES AND 0FH
 +   3 0040 07023  1            POKE    .MF.WDTRES SHR 4,WR
   196 0041 1C052               CALL    MKSCAN                  ;Key scan
   197            1            SKT1    FKEYREQ                 ;Does chattering terminate?
 +   1 0042 1E691  1            SKT     .MF.FKEYREQ SHR 4,#.DF.FKEYREQ AND 0FH
   198 0043 0C04E               BR      LMN600                  ;No -> Automatic-power-off
   199                                                          ;timer control
   200
   201                 ;*********************************************
   202                 ;*          Operation stop mode check        *
   203                 ;*********************************************
   204            1            CLR1    FKEYREQ
 +   1 0044 1469E  1            AND     .MF.FKEYREQ SHR 4,#.DF.(NOT FKEYREQ AND 0FH)
   205 0045 0B31F               SKNE    RKCODL,#CKOFF AND 0FH          ;Does off-chattering
   206 0046 0930F               SKE     RKCODH,#CKOFF SHR 4 AND 0FH    ;terminate?
   207 0047 0C04B               BR      LMN400                         ;No:  Single key
   208            1            SKF1    FSTOP                   ;Operation stop mode?
 +   1 0048 1F688  1            SKF     .MF.FSTOP SHR 4,#.DF.FSTOP AND 0FH
   209 0049 0C050               BR      LMN800                  ;Yes -> Resets the STOP mode.
   210 004A 0C04E               BR      LMN600                  ;No -> Automatic-power-off
   211                                                          ;timer control
   212
   213                 LMN400:
   214 004B EXTRN               CALL    MKBRN                   ;Branches to processing for each key
   215            1            SKF1    FOPREQ                  ;Is the operation possible?
 +   1 004C 1F694  1            SKF     .MF.FOPREQ SHR 4,#.DF.FOPREQ AND 0FH
   216 004D EXTRN               CALL    MOPBRN                  ;Yes:  Branch operation processing
   217                 LMN600:
   218 004E 1C0AF               CALL    MAPOFF                  ;Automatic-power-off
   219                                                          ;timer control
   220            1            SKF1    FSTPREQ                 ;Timeout?
 +   1 004F 1F698  1            SKF     .MF.FSTPREQ SHR 4,#.DF.FSTPREQ AND 0FH
   221                 LMN800:
   222 0050 1C0BF               CALL    MSTOP                   ;Yes -> Sets the STOP mode.
   223 0051 0C038               BR      MMAIN
   224
   225                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-007

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   226                  ;*********************************************
   227                  ;*                                           *
   228                  ;*      Key scanning                         *
   229                  ;*                                           *
   230                  ;*      INPUT   : Nothing                    *
   231                  ;*      OUTPUT  : RKCODH - RKCODL (Key code) *
   232                  ;*              : FKEYREQ (Key-processing    *
   233                  ;*                         request flag)     *
   234                  ;*********************************************
   235                  MKSCAN:
   236 0052 1D35F               MOV     RIPCODL,#CKOFF AND 0FH          ;Resets the
   237 0053 1D34F               MOV     RIPCODH,#CKOFF SHR 4 AND 0FH    ;input code.
   238 0054 1D007               MOV     RREG0,#CKSRB3 AND 0FH
   239 0055 1D020               MOV     RREG2,#0H                       ;Resets the key counter.
   240
   241                  ;*********************************************
   242                  ;*            P0B0 - P0B3 output             *
   243                  ;*********************************************
   244                  LKS100:
   245 0056 18710               ST      P0B,RREG0                 ;P0B <- Key scan signal
   246 0057 074F0               NOP                               ;Waits for 12 us.
   247 0058 074F0               NOP
   248 0059 074F0               NOP
   249 005A 08709               LD      RREG9,P0A                 ;Saves the key return value.
   250 005B 1D71F               MOV     P0B,#CALLHI AND 0FH
   251 005C 1C094               CALL    SKRET                     ;Judges the key return value.
   252            1             SKF1    FMULTI                    ;Multiple key?
 +   1 005D 1F692  1            SKF     .MF.FMULTI SHR 4,.#.DF.FMULTI AND 0FH
   253 005E 0C06D               BR      LKS200                    ;Yes
   254            1             SET1    CY
 +   1 005F 167F4  1            OR      .MF.CY SHR 4,#.DF.CY AND 0FH
   255 0060 07070               RORC    RREG0
   256            1             SKF1    CY                        ;Does P0B3 - P0B0 terminate?
 +   1 0061 1F7F4  1            SKF     .MF.CY SHR 4,#.DF.CY AND 0FH
   257 0062 0C056               BR      LKS100                    ;No
   258
   259                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-008

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   260                   ;*********************************************
   261                   ;*              P0C0 output                  *
   262                   ;*********************************************
   263 0063 1D000                MOV     RREG0,#CKSRC0 AND 0FH
   264 0064 18720                ST      P0C,RREG0                    ;P0C <- Key scan signal
   265 0065 074F0                NOP                                  ;Waits for 12 us.
   266 0066 074F0                NOP
   267 0067 074F0                NOP
   268 0068 08709                LD      RREG9,P0A                    ;Saves the key return value.
   269 0069 1D721                MOV     P0C,#C0C0HI AND 0FH
   270 006A 1C094                CALL    SKRET                        ;Judges the key return value.
   271              1            SKT1    FMULTI                       ;Multiple key?
 +   1 006B 1E692   1            SKT     .MF.FMULTI SHR 4,#.DF.FMULTI AND 0FH
   272 006C 0C070                BR      LKS300                       ;No
   273
   274                   ;*********************************************
   275                   ;*              Multiple key                 *
   276                   ;*********************************************
   277                   LKS200:
   278              1            CLR1    FMULTI
 +   1 006D 1469D   1            AND     .MF.FMULTI SHR 4,#.DF.(NOT FMULTI AND 0FH)
   279 006E 1D350                MOV     RIPCODL,#CKMLT AND 0FH          ;Input code
   280 006F 1D34F                MOV     RIPCODH,#CKMLT SHR 4 AND 0FH    ;<- Multiple-key code
   281
   282                   ;*********************************************
   283                   ;*              Key status judgment          *
   284                   ;*********************************************
   285                   LKS300:
   286 0070 08320                LD      RREG0,RCHCODH                ;Does the input code change?
   287 0071 05340                XOR     RREG0,RIPCODH
   288 0072 09000                SKE     RREG0,#0H
   289 0073 0C078                BR      LKS400
   290 0074 08330                LD      RREG0,RCHCODL
   291 0075 05350                XOR     RREG0,RIPCODL
   292 0076 0B000                SKNE    RREG0,#0H
   293 0077 0C07E                BR      LKS500                       ;No -> Chattering
   294                   LKS400:
   295 0078 08340                LD      RREG0,RIPCODH                ;Chattering code
   296 0079 18320                ST      RCHCODH,RREG0                ;-> Input code
   297 007A 08350                LD      RREG0,RIPCODL
   298 007B 18330                ST      RCHCODL,RREG0
   299 007C 1D362                MOV     RCHATC,#CCHATS AND 0FH       ;Chattering counter reset
   300 007D 0C093                BR      LKS999                       ;RET
   301
   302                           EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-009

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   303                   ;*********************************************
   304                   ;*              Chattering                   *
   305                   ;*********************************************
   306                   LKS500:
   307 007E 0B360               SKNE    RCHATC,#CCHATE AND 0FH  ;Has chattering already terminated?
   308 007F 0C093               BR      LKS999                  ;Yes -> RET
   309 0080 11361               SUB     RCHATC,#1H
   310 0081 09360               SKE     RCHATC,#CCHATE AND 0FH  ;Does chattering terminate?
   311 0082 0C093               BR      LKS999                  ;No -> RET
   312
   313                   ;*********************************************
   314                   ;*              End chattering               *
   315                   ;*********************************************
   316 0083 0B330               SKNE    RCHCODL,#CKMLT AND 0FH          ;Multiple key?
   317 0084 0932F               SKE     RCHCODH,#CKMLT SHR 4 AND 0FH
   318 0085 0C087               BR      LKS600                          ;No
   319 0086 0C08F               BR      LKS900                          ;Yes
   320                   LKS600:
   321 0087 0B33F               SKNE    RCHCODL,#CKOFF AND 0FH          ;Does off-chattering
   322 0088 0932F               SKE     RCHCODH,#CKOFF SHR 4 AND 0FH    ;terminate?
   323 0089 0C08B               BR      LKS700                          ;No: Single key
   324 008A 0C08E               BR      LKS800                          ;Yes
   325
   326                   LKS700:                                       ;<- Single key
   327 008B 0B31F               SKNE    RKCODL,#CKOFF AND 0FH           ;Key-off -> Single key?
   328 008C 0930F               SKE     RKCODH,#CKOFF SHR 4 AND 0FH
   329 008D 0C093               BR      LKS999                          ;No -> RET
   330
   331                   LKS800:                                 ;<- Key-off
   332             1            SET1    FKEYREQ                 ;Sets a key-processing request.
 +   1 008E 16691  1            OR      .MF.FKEYREQ SHR 4,#DF.FKEYREQ AND 0FH
   333
   334                   LKS900:                                 ;<- Multiple key
   335 008F 08320               LD      RREG0,RCHCODH           ;Key code
   336 0090 18300               ST      RKCODH,RREG0            ;<- Chattering code
   337 0091 08330               LD      RREG0,RCHCODL
   338 0092 18310               ST      RKCODL,RREG0
   339
   340                   LKS999:
   341 0093 070E0               RET
   342
   343                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-010

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   344                 ;*********************************************
   345                 ;*                                           *
   346                 ;*      Key return value judgment            *
   347                 ;*                                           *
   348                 ;*      INPUT   : RREG9 (Key return value)   *
   349                 ;*      OUTPUT  : FMULTI (Multiple-key flag) *
   350                 ;*                                           *
   351                 ;*********************************************
   352                 SKRET:
   353                 ;+++      P0A0 check                    +++
   354 0094 08091              LD      RREG1,RREG9
   355 0095 1601E              OR      RREG1,#CKRET0 AND 0FH
   356 0096 0B01F              SKNE    RREG1,#CALLHI AND 0FH   ;Is P0A0 low?
   357 0097 0C099              BR      JKR100                  ;No
   358 0098 1C0AB              CALL    SMKICD                  ;Yes:  Generates an input code.
   359
   360                 ;+++      P0A1 check                    +++
   361                 JKR100:
   362 0099 08091              LD      RREG1,RREG9
   363 009A 1601D              OR      RREG1,#CKRET1 AND 0FH
   364 009B 0B01F              SKNE    RREG1,#CALLHI AND 0FH   ;Is P0A1 low?
   365 009C 0C09E              BR      JKR200                  ;No
   366 009D 1C0AB              CALL    SMKICD                  ;Yes:  Generates an input code.
   367
   368                 ;+++      P0A2 check                    +++
   369                 JKR200:
   370 009E 08091              LD      RREG1,RREG9
   371 009F 1601B              OR      RREG1,#CKRET2 AND 0FH
   372 00A0 0B01F              SKNE    RREG1,#CALLHI AND 0FH   ;Is P0A2 low?
   373 00A1 0C0A3              BR      JKR300                  ;No
   374 00A2 1C0AB              CALL    SMKICD                  ;Yes:  Generates an input code.
   375
   376                 ;+++      P0A3 check                    +++
   377                 JKR300:
   378 00A3 08091              LD      RREG1,RREG9
   379 00A4 16017              OR      RREG1,#CKRET3 AND 0FH
   380 00A5 0B01F              SKNE    RREG1,#CALLHI AND 0FH   ;Is P0A3 low?
   381 00A6 0C0A8              BR      JKR400                  ;No
   382 00A7 1C0AB              CALL    SMKICD                  ;Yes:  Generates an input code.
   383
   384                 ;+++      Multiple-key press check      +++
   385                 JKR400:
   386 00A8 1B022              SKLT    RREG2,#2H               ;Multiple key press?
   387            1            SET1    FMULTI                  ;Yes
 +   1 00A9 16692  1           OR      .MF.FMULTI SHR 4,#.DF.FMULTI AND 0FH
   388
   389 00AA 070E0              RET
   390
   391                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-011

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   392                 ;*********************************************
   393                 ;*                                           *
   394                 ;*      Input code generation                *
   395                 ;*                                           *
   396                 ;*      INPUT   : RREG0 (Key scan signal)    *
   397                 ;*              : RREG9 (Key return value)   *
   398                 ;*      OUTPUT  : RREG2 (Key counter)        *
   399                 ;*              : RIPCODH - RIPCODL          *
   400                 ;*                (Input code)               *
   401                 ;*                                           *
   402                 ;*********************************************
   403                 SMKICD:
   404 00AB 18340              ST      RIPCODH,RREG0          ;Generates an input code.
   405 00AC 18359              ST      RIPCODL,RREG9
   406 00AD 10021              ADD     RREG2,#1H              ;Counts the number of keys.
   407
   408 00AE 070E0              RET
   409
   410                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-012

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   411                   ;*********************************************
   412                   ;*                                           *
   413                   ;*      Automatic-power-off timer control    *
   414                   ;*                                           *
   415                   ;*      INPUT   : RKCODH - RKCODL (Key code) *
   416                   ;*      OUTPUT  : RSTOP3 - RSTOP0            *
   417                   ;*                (Automatic-power-off timer) *
   418                   ;*              : FSTPREQ(STOP mode processing *
   419                   ;*                        request flag)      *
   420                   ;*                                           *
   421                   ;*********************************************
   422                   MAPOFF:
   423             1             BANK0
 +   1 00AF 1D790  1             MOV     BANK,#00H
   424 00B0 0B31F                SKNE    RKCODL,#CKOFF AND 0FH          ;Key-off?
   425 00B1 0930F                SKE     RKCODH,#CKOFF SHR 4 AND 0FH
   426 00B2 0C0BA                BR      LAP200                         ;No
   427
   428                   ;*********************************************
   429                   ;*      Automatic-power-off timer count      *
   430                   ;*********************************************
   431                           SUB     RSTOP0,#1H
   432 00B3 113B1                SUBC    RSTOP1,#0H
   433 00B4 133A0                SUBC    RSTOP2,#0H
   434 00B5 13390                SUBC    RSTOP3,#0H
   435 00B6 13380                SKF1    CY                     ;3-minute timeout?
 +   1             1             SKF     .MF.CY SHR 4,#.DF.CY AND 0FH
   436 00B7 1F7F4  1             SET1    FSTPREQ                ;Yes
 +   1             1             OR      .MF.FSTPREQ SHR 4,#.DF.FSTPREQ AND 0FH
   437 00B8 16698  1             BR      LAP400                 ;RET
   438 00B9 0C0BE
   439                   ;*********************************************
   440                   ;*      Automatic-power-off timer reset      *
   441                   ;*********************************************
   442                   LAP200:
   443                           MOV     RSTOP0,#CSTSTP AND 0FH
   444 00BA 1D3BF                MOV     RSTOP1,#CSTSTP SHR 4 AND 0FH
   445 00BB 1D3A4                MOV     RSTOP2,#CSTSTP SHR 8 AND 0FH
   446 00BC 1D396                MOV     RSTOP3,#CSTSTP SHR 12 AND 0FH
   447 00BD 1D384
   448                   LAP400:
   449                           RET
   450 00BE 070E0
   451                           EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-013

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   452                   ;*********************************************
   453                   ;*                                           *
   454                   ;*      STOP mode processing                 *
   455                   ;*                                           *
   456                   ;*      INPUT   : Nothing                    *
   457                   ;*      OUTPUT  : FSTOP (Operation stop      *
   458                   ;*                       mode flag)          *
   459                   ;*              : FSTPREQ(STOP mode processing *
   460                   ;*                        request flag)      *
   461                   ;*********************************************
   462                   ;*********************************************
   463                   ;*              STOP mode setting            *
   464                   ;*********************************************
   465                   MSTOP:
   466             1             CLR1    FSTPREQ
 +   1 00BF 14697  1             AND     .MF.FSTPREQ SHR 4,#.DF.(NOT FSTPREQ AND 0FH)
   467             1             CLR1    LCDEN                   ;LCD display off
 +   1 00C0 07331  1             PEEK    WR,.MF.LCDEN SHR 4
 +   2 00C1 14787  1             AND     WR,#.DF.(NOT LCDEN) AND 0FH
 +   3 00C2 07321  1             POKE    .MF.LCDEN SHR 4,WR
   468             1             CLR1    TMEN                    ;8-bit timer count stop
 +   1 00C3 07333  1             PEEK    WR,.MF.TMEN SHR 4
 +   2 00C4 14787  1             AND     WR,#.DF.(NOT TMEN) AND 0FH
 +   3 00C5 07323  1             POKE    .MF.TMEN SHR 4,WR
   469             1             SET1    FSTOP
 +   1 00C6 16688  1             OR      .MF.FSTOP SHR 4,#.DF.FSTOP AND 0FH
   470 00C7 1D710                MOV     P0B,#CALLOW AND 0FH
   471 00C8 1D720                MOV     P0C,#CALLOW AND 0FH
   472 00C9 1D0F0                MOV     DBF0,#CRLSTP AND 0FH
   473 00CA 1D0E0                MOV     DBF1,#CRLSTP SHR 4 AND 0FH
   474 00CB 070A2                PUT     TMM,DBF
   475 00CC 074F0                NOP
   476 00CD 072F8                STOP    8H                      ;Sets the STOP mode.
   477
   478                           EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:23:25 12/20/93 PAGE 02-014

PROG =

SOURCE = CALC1.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   479                   ;*********************************************
   480                   ;*              STOP mode release            *
   481                   ;*********************************************
   482              1            CLR1    TMEN                     ;8-bit timer count stop
 +   1 00CE 07333   1            PEEK    WR,.MF.TMEN SHR 4
 +   2 00CF 14787   1            AND     WR,#.DF.(NOT TMEN) AND 0FH
 +   3 00D0 07323   1            POKE    .MF.TMEN SHR 4,WR
   483 00D1 1D71F               MOV     P0B,#CALLHI AND 0FH            ;Port reset
   484 00D2 1D721               MOV     P0C,#C0C0HI AND 0FH
   485 00D3 1D0FC               MOV     DBF0#CRLHLT AND 0FH            ;8-bit modulo
   486 00D4 1D0E9               MOV     DBF1,#CRLHLT SHR 4 AND 0FH     ;register reset
   487 00D5 070A2               PUT     TMM,DBF
   488              1            SET1    TMEN                     ;8-bit timer count start
 +   1 00D6 07333   1            PEEK    WR,.MF.TMEN SHR 4
 +   2 00D7 16788   1            OR      WR,#.DF.TMEN AND 0FH
 +   3 00D8 07323   1            POKE    .MF.TMEN SHR 4,WR
   489 00D9 070E0               RET
   490
   491                          END


  TOTAL ERRORS   = 0
  TOTAL WARNINGS = 0

  END OF LIST
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-001

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
     1                 ;*********************************************
     2                 ;*                                           *
     3                 ;*  USER NAME   : NEC CORPORATION            *
     4                 ;*                                           *
     5                 ;*  SYSTEM NAME : 17K APPLICATION (CALCULATOR) *
     6                 ;*                                           *
     7                 ;*  CPU         : µPD17201A                  *
     8                 ;*                                           *
     9                 ;*  LAST UPDATE : '93/12/20  11:00           *
    10                 ;*                                           *
    11                 ;*********************************************
    12                 ;*********************************************
    13                 ;*                                           *
    14                 ;*  FILE NAME       : CALC2.ASM              *
    15                 ;*                                           *
    16                 ;*  INCLUDE 9 ROUTINES:                      *
    17                 ;*          MKBRN  : KEY BRANCH              *
    18                 ;*          KALLCR : ALL CLEAR KEY PROCESS   *
    19                 ;*          KCLR   : CLEAR KEY PROCESS       *
    20                 ;*          KNUMPT : NUMBER AND POINT KEY    *
    21                 ;*                                PROCESS    *
    22                 ;*          KSIGN  : SIGN KEY PROCESS        *
    23                 ;*          KOPE   : OPERATOR KEY PROCESS    *
    24                 ;*          KPER   : PERCENT KEY PROCESS     *
    25                 ;*          KEQU   : EQUAL KEY PROCESS       *
    26                 ;*          MOPBRN : OPERATOR BRANCH         *
    27                 ;*                     AND ERROR PROCESS     *
    28                 ;*                                           *
    29                 ;*********************************************
    30
    31                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-002

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    32                 ;*********************************************
    33                 ;*              External reference           *
    34                 ;*********************************************
    35                 EXTRN   LAB: SRAMCR           ;RAM all-clear processing
    36                 EXTRN   LAB: SDPINI           ;Display data area initialization
    37                 EXTRN   LAB: SUSHFD           ;Display data area shift up
    38                 EXTRN   LAB: STRAN            ;Data transfer
    39                 EXTRN   LAB: SFIX             ;Operation result conversion
    40                 EXTRN   LAB: STRNDY           ;Display data conversion
    41                 EXTRN   LAB: SDISP            ;Display data output
    42                 EXTRN   LAB: SRXCLR
    43                 EXTRN   LAB: SRYCLR
    44                 EXTRN   LAB: SFPADD
    45                 EXTRN   LAB: SFPSUB
    46                 EXTRN   LAB: SFPMULT
    47                 EXTRN   LAB: SFPDIV
    48
    49                 EXTRN   MEM: RREG0,RREG2,RREG9     ;General-purpose register
    50                 EXTRN   MEM: RNUMC                 ;Numeric key counter
    51                 EXTRN   MEM: RPTLOC                ;Decimal-point position area
    52                 EXTRN   MEM: RSINLOC               ;Sign position area
    53                 EXTRN   MEM: RDEXP                 ;Display data exponent area
    54                 EXTRN   MEM: RDLSD                 ;Display data area
    55                 EXTRN   MEM: RZLSD                 ;Save register
    56                 EXTRN   MEM: RKCODH,RKCODL         ;Key code area
    57                 EXTRN   MEM: RCHCODH,RCHCODL       ;Chattering code area
    58                 EXTRN   MEM: ROPE                  ;Operator area
    59                 EXTRN   MEM: RCOM                  ;Area for the operator to be executed
    60                 EXTRN   MEM: RMODE                 ;Mode area
    61                 EXTRN   MEM: RXSIGN                ;Floating-point register 1, sign
    62                 EXTRN   MEM: RXEXP                 ;Floating-point register 1, characteristic
    63                 EXTRN   MEM: RYSIGN                ;Floating-point register 2, sign
    64                 EXTRN   MEM: RYEXP                 ;Floating-point register 2, characteristic
    65                 EXTRN   MEM: ROPFLG                ;Operation flag
    66
    67                 EXTRN   FLG: FPER                  ;Percent flag
    68                 EXTRN   FLG: FOPEND                ;Operation end flag
    69                 EXTRN   FLG: FFALSE                ;Illegal-input flag
    70                 EXTRN   FLG: FSTOP                 ;STOP mode flag
    71                 EXTRN   FLG: FOPREQ                ;Operation request flag
    72                 EXTRN   FLG: FZERO                 ;Operation result zero flag
    73                 EXTRN   FLG: FDVERR                ;Division error flag
    74
    75                 EXTRN   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-003

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    76                  EXTRN   DAT: CKSRB0,CKSRB1,CKSRB2       ;Key code (4 high-order bits)
    77                  EXTRN   DAT: CKSRB3,CKSRC0
    78                  EXTRN   DAT: CKRET0,CKRET1,CKRET2,CKRET3   ;Key code (4 low-order bits)
    79                  EXTRN   DAT: CSTMPE                     ;MPE set
    80                  EXTRN   DAT: CSTBK1                     ;MP <- BANK1
    81                  EXTRN   DAT: CNONUM,CFLNUM              ;Number of numeric key inputs:  0/7
    82                  EXTRN   DAT: CEXPINI                    ;Initial value of the display data exponent area
    83                  EXTRN   DAT: CEXPMIN,CEXPMAX            ;Overflow judgment
    84                  EXTRN   DAT: CCHSIN                     ;Sign data inversion
    85                  EXTRN   DAT: CZERO                      ;[0] Display data
    86                  EXTRN   DAT: CONE                       ;[1] Display data
    87                  EXTRN   DAT: CTWO                       ;[2] Display data
    88                  EXTRN   DAT: CTHREE                     ;[3] Display data
    89                  EXTRN   DAT: CFOUR                      ;[4] Display data
    90                  EXTRN   DAT: CFIVE                      ;[5] Display data
    91                  EXTRN   DAT: CSIX                       ;[6] Display data
    92                  EXTRN   DAT: CSEVEN                     ;[7] Display data
    93                  EXTRN   DAT: CEIGHT                     ;[8] Display data
    94                  EXTRN   DAT: CNINE                      ;[9] Display data
    95                  EXTRN   DAT: CSPACE                     ;Space display data
    96                  EXTRN   DAT: CERRDP                     ;[E] Display data
    97                  EXTRN   DAT: CADD                       ;[+] Operator data
    98                  EXTRN   DAT: CSUB                       ;[-] Operator data
    99                  EXTRN   DAT: CMUL                       ;[x] Operator data
   100                  EXTRN   DAT: CDIV                       ;[÷] Operator data
   101                  EXTRN   DAT: CRIGHT                     ;Second-term input mode
   102                  EXTRN   DAT: COPSEL                     ;Operator selection mode
   103                  EXTRN   DAT: CLEFT                      ;First-term input mode
   104                  EXTRN   DAT: CERROR                     ;Error mode
   105
   106                  ;*********************************************
   107                  ;*              External definition          *
   108                  ;*********************************************
   109                  PUBLIC  MKBRN,MOPBRN
   110
   111                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-004

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   112                  ;*************************************************
   113                  ;*                                               *
   114                  ;*      Macro definition file reading            *
   115                  ;*                                               *
   116                  ;*************************************************
   117                1         INCLUDE 'PUSHMP.ASM'
 +   1                1
 +   2                1         EXTRN   MEM: RREG10,RREG11
 +   3                1
 +   4                1 ;*************************************************
 +   5                1 ;*                                               *
 +   6                1 ;*      Macro definition for saving              *
 +   7                1 ;*      data memory row address pointer          *
 +   8                1 ;*                                               *
 +   9                1 ;*************************************************
 +  10                1 APUSHMP MACRO
 +  11                1         LD      RREG10,MPH
 +  12                1         LD      RREG11,MPL
 +  13                1         ENDM
 +  14                1
 +  15                1 ;*************************************************
 +  16                1 ;*                                               *
 +  17                1 ;*      Macro definition for restoring the       *
 +  18                1 ;*      data memory row address pointer          *
 +  19                1 ;*                                               *
 +  20                1 ;*************************************************
 +  21                1 APOPMP  MACRO
 +  22                1         ST      MPH,RREG10
 +  23                1         ST      MPL,RREG11
 +  24                1         ENDM
 +  25                1
 +  26                1         EOF
   118
   119                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-005

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   120                 ;*********************************************
   121                 ;*                                           *
   122                 ;*      Branch to processing for each key    *
   123                 ;*                                           *
   124                 ;*      INPUT    : RKCODH - RKCODL (Key code)*
   125                 ;*               : RMODE (Mode area)         *
   126                 ;*               : FSTOP (Operation stop     *
   127                 ;*                        mode flag)         *
   128                 ;*      OUTPUT   : Nothing                   *
   129                 ;*********************************************
   130                 MKBRN:
   131 00DA 0B300              SKNE    RKCODH,#CKSRC0 AND 0FH  ;Was the [AC] key pressed?
   132 00DB 09317              SKE     RKCODL,#CKRET3 AND 0FH
   133 00DC 0C0DE              BR      LKD050                  ;No
   134 00DD 0C0F1              BR      LKD250                  ;Yes
   135
   136                 LKD050:
   137            1            SKT1    FSTOP                   ;Operation stop mode?
 +   1 00DE 1E688  1           SKT     .MF.FSTOP SHR 4,#.DF.FSTOP AND 0FH
   138 00DF 0B648              SKNE    RMODE,#CERROR AND 0FH   ;Error mode?
   139 00E0 0C0FE              BR      LKD900                  ;Yes -> RET
   140
   141 00E1 09300              SKE     RKCODH,#CKSRC0 AND 0FH
   142 00E2 0C0E6              BR      LKD100
   143 00E3 0B31E              SKNE    RKCODL,#CKRET0 AND 0FH  ;Was the [%] key pressed?
   144 00E4 0C0FB              BR      LKD700                  ;Yes
   145 00E5 0C0F9              BR      LKD600                  ;No -> Operator key processing
   146                 LKD100:
   147 00E6 0930E              SKE     RKCODH,#CKSRB0 AND 0FH
   148 00E7 0C0ED              BR      LKD200
   149 00E8 0B31E              SKNE    RKCODL,#CKRET0 AND 0FH  ;Was the [=] key pressed?
   150 00E9 0C0FD              BR      LKD800                  ;Yes
   151 00EA 0B317              SKNE    RKCODL,#CKRET3 AND 0FH  ;Was the [C] key pressed?
   152 00EB 0C0F3              BR      LKD300                  ;Yes
   153 00EC 0C0F9              BR      LKD600                  ;No
   154                 LKD200:
   155 00ED 0B30D              SKNE    RKCODH,#CKSRB1 AND 0FH  ;Was the [±] key pressed?
   156 00EE 0931E              SKE     RKCODL,#CKRET0 AND 0FH
   157 00EF 0C0F5              BR      LKD400                  ;No -> Numeric key, decimal-point key processing
   158 00F0 0C0F7              BR      LKD500                  ;Yes
   159
   160                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-006

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   161                  ;+++   All-clear key                    +++
   162                  LKD250:
   163 00F1 1C0FF               CALL    KALLCR
   164 00F2 0C0FE               BR      LKD900                  ;RET
   165
   166                  ;+++   Clear key                        +++
   167                  LKD300:
   168 00F3 1C10F               CALL    KCLR
   169 00F4 0C0FE               BR      LKD900                  ;RET
   170
   171                  ;+++   Numeric key, decimal-point key   +++
   172                  LKD400:
   173 00F5 1C121               CALL    KNUMPT
   174 00F6 0C0FE               BR      LKD900                  ;RET
   175
   176                  ;+++   Plus/minus key                   +++
   177                  LKD500:
   178 00F7 1C171               CALL    KSIGN
   179 00F8 0C0FE               BR      LKD900                  ;RET
   180
   181                  ;+++   Operator key                     +++
   182                  LKD600:
   183 00F9 1C198               CALL    KOPE
   184 00FA 0C0FE               BR      LKD900                  ;RET
   185
   186                  ;+++   Percent key                      +++
   187                  LKD700:
   188 00FB 1C1D0               CALL    KPER
   189 00FC 0C0FE               BR      LKD900                  ;RET
   190
   191                  ;+++   Equal key                        +++
   192                  LKD800:
   193 00FD 1C1EA               CALL    KEQU
   194
   195                  LKD900:
   196 00FE 070E0               RET
   197
   198                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-007

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   199                   ;*********************************************
   200                   ;*                                           *
   201                   ;*      All-clear key processing             *
   202                   ;*                                           *
   203                   ;*      INPUT   : Nothing                    *
   204                   ;*      OUTPUT  : RDEXP (Display data        *
   205                   ;*                       exponent area)      *
   206                   ;*              : RMODE (Mode area)          *
   207                   ;*********************************************
   208                   KALLCR:
   209             1             CLR1    LCDEN                        ;LCD display off
 +   1 00FF 07331  1             PEEK    WR,.MF.LCDEN SHR 4
 +   2 0100 14787  1             AND     WR,#.DF.(NOT LCDEN) AND 0FH
 +   3 0101 07321  1             POKE    .MF.LCDEN SHR 4,WR
   210 0102 EXTRN                CALL    SRAMCR                       ;RAM all clear
   211
   212 0103 1D148                MOV     RDEXP,#CEXPINI AND 0FH
   213 0104 1D317                MOV     RKCODL,#CKRET3 AND 0FH       ;Key code,
   214 0105 1D300                MOV     RKCODH,#CKSRC0 SHR 4 AND 0FH ;chattering code
   215 0106 1D337                MOV     RCHCODL,#CKRET3 AND 0FH      ;<- Restores the code
   216 0107 1D320                MOV     RCHCODH,#CKSRC0 SHR 4 AND 0FH ;  of the [AC] key.
   217 0108 1D644                MOV     RMODE,#CLEFT AND 0FH
   218
   219 0109 EXTRN                CALL    SDPINI          ;Display data area initialization
   220 010A EXTRN                CALL    SDISP           ;Display data output
   221             1             SET1    LCDEN           ;LCD display on
 +   1 010B 07331  1             PEEK    WR,.MF.LCDEN SHR 4
 +   2 010C 16788  1             OR      WR,#.DF.LCDEN AND 0FH
 +   3 010D 07321  1             POKE    .MF.LCDEN SHR 4,WR
   222
   223 010E 070E0                RET
   224
   225                           EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-008

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   226                  ;*********************************************
   227                  ;*                                           *
   228                  ;*      Clear key processing                 *
   229                  ;*                                           *
   230                  ;*      INPUT    : RMODE (Mode area)         *
   231                  ;*               : FPER (Percent flag)       *
   232                  ;*      OUTPUT   : RNUMC (Numeric key counter) *
   233                  ;*               : RDEXP (Display data exponent area)*
   234                  ;*               : RMODE (Mode area)         *
   235                  ;*               : FPER (Percent flag)       *
   236                  ;*               : FOPEND (Operation end flag) *
   237                  ;*               : FFALSE (Illegal-input flag) *
   238                  ;*                                           *
   239                  ;*********************************************
   240                  KCLR:
   241            1             SKT1     FPER                        ;Was the [%] key pressed?
 +   1 010F 1E681     1        SKT      .MF.FPER SHR 4,#.DF.FPER AND 0FH
   242 0110 0C113                BR       NCR200                      ;No
   243            1             CLR1     FPER                        ;Yes
 +   1 0111 1468E     1        AND      .MF.FPER SHR 4,#.DF.(NOT FPER AND 0FH)
   244 0112 1D644                MOV      RMODE,#CLEFT AND 0FH        ;Sets the first-term input mode.
   245                  NCR200:
   246            1             CLR2     FOPEND,FFALSE
 +   1 0113 14689     1        AND      .MF.FOPEND SHR 4,#.DF.(NOT (FOPEND OR FFALSE) AND 0FH)
   247 0114 1D100                MOV      RNUMC,#CNONUM AND 0FH
   248 0115 1D148                MOV      RDEXP,#CEXPINI AND 0FH
   249 0116 0B642                SKNE     RMODE,#COPSEL AND 0FH       ;Operator selection mode?
   250 0117 1D641                MOV      RMODE,#CRIGHT AND 0FH       ;Yes:  Sets the second-term
   251            1             CLR1     LCDEN                                          input mode.
 +   1 0118 07331     1        PEEK     WR,.MF.LCDEN SHR 4
 +   2 0119 14787     1        AND      WR,#.DF.(NOT LCDEN) AND 0FH
 +   3 011A 07321     1        POKE     .MF.LCDEN SHR 4,WR
   252 011B EXTRN                CALL     SDPINI
   253 011C EXTRN                CALL     SDISP
   254            1             SET1     LCDEN
 +   1 011D 07331     1        PEEK     WR,.MF.LCDEN SHR 4
 +   2 011E 16788     1        OR       WR,#.DF.LCDEN AND 0FH
 +   3 011F 07321     1        POKE     .MF.LCDEN SHR 4,WR
   255
   256 0120 070E0                RET
   257
   258                           EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-009

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   259                   ;*********************************************
   260                   ;*                                           *
   261                   ;*      Numeric key, decimal-point key processing  *
   262                   ;*                                           *
   263                   ;*      INPUT    : RNUMC (Numeric key counter)  *
   264                   ;*               : RKCODH - RKCODL (Key code)   *
   265                   ;*               : RMODE (Mode area)            *
   266                   ;*               : FPER (Percent flag)          *
   267                   ;*               : FFALSE (Illegal-input flag)  *
   268                   ;*      OUTPUT   : RNUMC (Numeric key counter)  *
   269                   ;*               : RPTLOC (Decimal-point position area) *
   270                   ;*               : RDEXP - RDMSD (Display data) *
   271                   ;*               : RMODE (Mode area)            *
   272                   ;*               : FPER (Percent flag)          *
   273                   ;*               : FOPEND (Operation end flag)  *
   274                   ;*               : FFALSE (Illegal-input flag)  *
   275                   ;*                                           *
   276                   ;*********************************************
   277                   KNUMPT:
   278             1             SKT1     FFALSE
 +   1 0121 1E684  1             SKT      .MF.FFALSE SHR 4,#.DF.FFALSE AND 0FH
   279 0122 0C126                BR       NNP100
   280             1             CLR1     FFALSE
 +   1 0123 1468B  1             AND      .MF.FFALSE SHR 4,#.DF.(NOT FFALSE AND 0FH)
   281 0124 1D100                MOV      RNUMC,#CNONUM AND 0FH
   282 0125 1D148                MOV      RDEXP,#CEXPINI AND 0FH
   283                   NNP100:
   284             1             SKF1     FPER
 +   1 0126 1F681  1             SKF      .MF.FPER SHR 4,#.DF.FPER AND 0FH
   285 0127 1D644                MOV      RMODE,#CLEFT AND 0FH
   286             1             CLR2     FPER,FOPEND
 +   1 0128 1468C  1             AND      .MF.FPER SHR 4,#.DF.(NOT (FPER OR FOPEND) AND 0FH)
   287
   288 0129 0B30B                SKNE     RKCODH,#CKSRB2 AND 0FH  ;Was the decimal-point key pressed?
   289 012A 0931E                SKE      RKCODL,#CKRET0 AND 0FH
   290 012B 0C13E                BR       NNP300                  ;No -> Numeric key processing
   291
   292                           EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-010

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   293                  ;*****************************************************
   294          .       ;*            Decimal-point key processing            *
   295                  ;*****************************************************
   296 012C 09148               SKE     RDEXP,#CEXPINI AND 0FH  ;Has the decimal-point key already been pressed?
   297 012D 0C170               BR      NNP999                  ;Yes -> RET
   298 012E 09100               SKE     RNUMC, #CNONUM          ;Was the decimal-point key pressed
   299 012F 0C139               BR      NNP200                  ;before a numeric key?
   300 0130 10101               ADD     RNUMC,#1H               ;Yes
   301              1           CLR1    LCDEN
 +   1 0131 07331  1            PEEK    WR,.MF.LCDEN SHR 4
 +   2 0132 14787  1            AND     WR,#.DF.(NOT LCDEN) AND 0FH
 +   3 0133 07321  1            POKE    .MF.LCDEN SHR 4,WR
   302 0134 EXTRN               CALL    SDPINI
   303 0135 EXTRN               CALL    SDISP
   304              1           SET1    LCDEN
 +   1 0136 07331  1            PEEK    WR,.MF.LCDEN SHR 4
 +   2 0137 16788  1            OR      WR,#.DF.LCDEN AND 0FH
 +   3 0138 07321  1            POKE    .MF.LCDEN SHR 4,WR
   305                  NNP200:
   306 0139 08100               LD      RREG0,RNUMC
   307 013A 18140               ST      RDEXP,RREG0
   308 013B 0B642               SKNE    RMODE,#COPSEL AND 0FH   ;Operator selection mode?
   309 013C 1D641               MOV     RMODE,#CRIGHT AND 0FH   ;Yes:  Sets the second-term input mode.
   310 013D 0C170               BR      NNP999                  ;RET
   311
   312                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-011

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   313                 ;*********************************************
   314                 ;*            Numeric key processing           *
   315                 ;*********************************************
   316                 NNP300:
   317 013E 0B107              SKNE    RNUMC,#CFLNUM AND 0FH   ;Has the numeric key been already pressed seven times?
   318 013F 0C170              BR      NNP999                  ;Yes -> RET
   319
   320                 ;+++            Key decoding            +++
   321 0140 0931E              SKE     RKCODL,#CKRET0 AND 0FH          ;Was [0] pressed?
   322 0141 0C144              BR      NNP400
   323 0142 1D090              MOV     RREG9,#CZERO AND 0FH            ;Yes
   324 0143 0C15C              BR      NNP700
   325
   326                 NNP400:
   327 0144 0931D              SKE     RKCODL,#CKRET1 AND 0FH
   328 0145 0C14D              BR      NNP500
   329 0146 0B307              SKNE    RKCODH,#CKSRB3 AND 0FH          ;Was [1] pressed?
   330 0147 1D091              MOV     RREG9,#CONE AND 0FH             ;Yes
   331 0148 0B30B              SKNE    RKCODH,#CKSRB2 AND 0FH          ;Was [2] pressed?
   332 0149 1D092              MOV     RREG9,#CTWO AND 0FH             ;Yes
   333 014A 0B30D              SKNE    RKCODH,CKSRB1 AND 0FH           ;Was [3] pressed?
   334 014B 1D093              MOV     RREG9,#CHTREE AND 0FH           ;Yes
   335 014C 0C15C              BR      NNP700
   336
   337                 NNP500:
   338 014D 0931B              SKE     RKCODL,#CKRET2 AND 0FH
   339 014E 0C156              BR      NNP600
   340 014F 0B307              SKNE    RKCODH,#CKSRB3 AND 0FH          ;Was [4] pressed?
   341 0150 1D094              MOV     RREG9,#CFOUR AND 0FH            ;Yes
   342 0151 0B30B              SKNE    RKCODH,#CKSRB2 AND 0FH          ;Was [5] pressed?
   343 0152 1D095              MOV     RREG9,#CFIVE AND 0FH            ;Yes
   344 0153 0B30D              SKNE    RKCODH,#CKSRB1 AND 0FH          ;Was [6] pressed?
   345 0154 1D096              MOV     RREG9,#CSIX AND 0FH             ;Yes
   346 0155 0C15C              BR      NNP700
   347
   348                 NNP600:
   349 0156 0B307              SKNE    RKCODH,#CKSRB3 AND 0FH          ;Was [7] pressed?
   350 0157 1D097              MOV     RREG9,#CSEVEN AND 0FH           ;Yes
   351 0158 0B30B              SKNE    RKCODH,#CKSRB2 AND 0FH          ;Was [8] pressed?
   352 0159 1D098              MOV     RREG9,#CEIGHT AND 0FH           ;Yes
   353 015A 0B30D              SKNE    RKCODH,#CKSRB1 AND 0FH          ;Was [9] pressed?
   354 015B 1D099              MOV     RREG9,#CNINE AND 0FH            ;Yes
   355
   356                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-012

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   357                   NNP700:
   358              1            CLR1    LCDEN                    ;LCD display off
 +   1 015C 07331   1            PEEK    WR,.MF.LCDEN SHR 4
 +   2 015D 14787   1            AND     WR,#.DF.(NOT LCDEN) AND 0FH
 +   3 015E 07321   1            POKE    .MF.LCDEN SHR 4,WR
   359 015F 09644               SKE     RMODE,#CLEFT AND 0FH     ;First-term input mode?
   360 0160 1D641               MOV     RMODE,#CRIGHT AND 0FH    ;No : Sets the second-term input mode.
   361 0161 09100               SKE     RNUMC,#CNONUM AND 0FH    ;First numeric key input?
   362 0162 0C167               BR      NNP800                   ;No
   363 0163 EXTRN               CALL    SDPINI                   ;Yes:  Display data area
   364                                                                     initialization
   365 0164 0B090               SKNE    RREG9,#CZERO AND 0FH     ;Was [0] pressed?
   366 0165 0C16C               BR      NNP900                   ;Yes
   367 0166 1D18A               MOV     RDLSD,#CSPACE AND 0FH    ;No:  Least significant digit of the display data
   368                                                                      -> Space display data
   369                   NNP800:
   370 0167 09148               SKE     RDEXP,#CEXPINI AND 0FH   ;Has the decimal-point key already been pressed?
   371 0168 11111               SUB     RPTLOC,#1H               ;Yes
   372 0169 10101               ADD     RNUMC,#1H
   373 016A EXTRN               CALL    SUSHFD                   ;Display data area shift right
   374 016B 18189               ST      RDLSD,RREG9              ;Least significant digit of the display data
   375                                                           ; <- Display data of the pressed numeric value
   376                   NNP900:
   377 016C EXTRN               CALL    SDISP                    ;Display data output
   378              1            SET1    LCDEN                    ;LCD display on
 +   1 016D 07331   1            PEEK    WR,.MF.LCDEN SHR 4
 +   2 016E 16788   1            OR      WR,#.DF.LCDEN AND 0FH
 +   3 016F 07321   1            POKE    .MF.LCDEN SHR 4, WR
   379
   380                   NNP999:
   381 0170 070E0               RET
   382
   383                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-013

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   384                   ;*********************************************
   385                   ;*                                           *
   386                   ;*      Plus/minus key processing            *
   387                   ;*                                           *
   388                   ;*      INPUT    : RNUMC (Numeric key counter)  *
   389                   ;*               : RSINLOC (Sign position area) *
   390                   ;*               : RMODE (Mode area)            *
   391                   ;*               : FPER (Percent flag)          *
   392                   ;*               : FOPEND (Operation end flag)  *
   393                   ;*               : FZERO (Operation result zero flag)  *
   394                   ;*      OUTPUT   : RSINLOC (Sign position area) *
   395                   ;*               : RDLSD-RDMSD (Display data)   *
   396                   ;*               : RMODE (Mode area)            *
   397                   ;*               : RXSIGN (Floating-point       *
   398                   ;*                 register 1, sign)            *
   399                   ;*               : FOPEND (Operation end flag)  *
   400                   ;*                                           *
   401                   ;*********************************************
   402                   KSIGN:
   403 0171 0B642               SKNE    RMODE,#COPSEL AND 0FH          ;Operator selection mode?
   404 0172 0C197               BR      NSG500                         ;Yes -> RET
   405             1            SKT1    FPER                           ;Was the [%] key pressed?
 +   1 0173 1E681  1            SKT     .MF.FPER SHR 4,#.DF.FPER AND 0FH
   406 0174 0C178               BR      NSG100                         ;No
   407             1            CLR1    FPER                           ;Yes
 +   1 0175 1468E  1            AND     .MF.FPER SHR 4,#.DF.(NOT FPER AND 0FH)
   408             1            SET1    FOPEND
 +   1 0176 16682  1            OR      .MF.FOPEND SHR 4,#.DF.FOPEND AND 0FH
   409 0177 1D644               MOV     RMODE,#CLEFT AND 0FH           ;Sets the first-term input mode.
   410                   NSG100:
   411             1            SKF1    FOPEND
 +   1 0178 1F682  1            SKF     .MF.FOPEND SHR 4,#.DF.FOPEND AND 0FH
   412 0179 0C180               BR      NSG200
   413
   414 017A 0B100               SKNE    RNUMC,#CNONUM                  ;Is [_____0.] displayed?
   415 017B 0C197               BR      NSG500                         ;Yes -> RET
   416 017C 08100               LD      RREG0,RNUMC                    ;Sign position calculation
   417 017D 10008               ADD     RREG0,#.DM.RDLSD AND 0FH
   418 017E 18120               ST      RSINLOC,RREG0
   419 017F 0C186               BR      NSG300
   420
   421                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-014

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   422                   NSG200:
   423              1            BANK1    BANK,#01H
 +   1 0180 1D791   1            MOV      FZERO
   424              1            SKF1     .MF.FZERO SHR 4,#.DF.FZERO AND 0FH
 +   1 0181 1F302   1            SKF      NSG400
   425 0182 0C196               BR       RXSIGN,#CCHSIN AND 0FH
   426 0183 15031               XOR
   427              1            BANK0
 +   1 0184 1D790   1            MOV      BANK, #00H
   428 0185 08120               LD       RREG0, RSINLOC
   429
   430                   NSG300:
   431              1            APUSHMP                            ;Saves the memory pointer
 +   1 0186 087AA   1            LD       RREG10, MPH
 +   2 0187 087BB   1            LD       RREG11, MPL
   432
   433              1            CLR1     LCDEN                     ;LCD display off
 +   1 0188 07331   1            PEEK     WR,.MF.LCDEN SHR 4
 +   2 0189 14787   1            AND      WR,#.DF.(NOT LCDEN) AND 0FH
 +   3 018A 07321   1            POKE     .MF.LCDEN SHR 4,WR
   434 018B 1D7A8               MOV      MPH,#CSTMPE AND 0FH
   435 018C 1D7B1               MOV      MPL,#.DM.RDLSD SHR 4 AND 0FH
   436 018D 1A010               MOV      RREG1,@RREG0
   437 018E 15011               XOR      RREG1,#CCHSIN AND 0FH      ;Inverts the sign of display data.
   438 018F 0A010               MOV      @RREG0, RREG1
   439              1            APOPMP                             ;Restores the memory pointer.
 +   1 0190 187AA   1            ST       MPH,RREG10
 +   2 0191 187BB   1            ST       MPL,RREG11
   440
   441 0192 EXTRN               CALL     SDISP                     ;Display data output
   442              1            SET1     LCDEN                     ;LCD display on
 +   1 0193 07331   1            PEEK     WR,.MF.LCDEN SHR 4
 +   2 0194 16788   1            OR       WR,#.DF.LCDEN AND 0FH
 +   3 0195 07321   1            POKE     .MF.LCDEN SHR 4,WR
   443
   444                   NSG400:
   445              1            BANK0
 +   1 0196 1D790   1            MOV      BANK,#00H
   446
   447                   NSG500:
   448 0197 070E0               RET
   449
   450                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-015

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   451                 ;*********************************************************
   452                 ;*                                                       *
   453                 ;*      OPERATOR KEY PROCESSING                          *
   454                 ;*                                                       *
   455                 ;*      INPUT    : RKCODH - RKCODL (Key Code)            *
   456                 ;*               : RZLSD - RZMSD (Saving Register)       *
   457                 ;*               : ROPE (Operator area)                  *
   458                 ;*               : RCOM (Area for operator to be executed) *
   459                 ;*               : RMODE (Mode Area)                     *
   460                 ;*               : RXSIGN - RXMSD                        *
   461                 ;*                 (Floating-point register 1)           *
   462                 ;*               : RYSIGN - RYMSD                        *
   463                 ;*                 (Floating-point register 2)           *
   464                 ;*               : FPER (Percent Flag)                   *
   465                 ;*               : FOPEND (Operation end flag)           *
   466                 ;*               : FFALSE (Illegal-input flag)           *
   467                 ;*      OUTPUT   : ROPE (Operator area)                  *
   468                 ;*               : RCOM (Area for operator to be executed) *
   469                 ;*               : RMODE (Mode area)                     *
   470                 ;*               : FOPEND (Operation end flag)           *
   471                 ;*               : FFALSE (Illegal-input flag)           *
   472                 ;*               : FOPREQ (Operation request flag)       *
   473                 ;*               : RXSIGN - RXMSD                        *
   474                 ;*                 (Floating-point register 1)           *
   475                 ;*               : RYSIGN - RYMSD                        *
   476                 ;*                 (Floating-point register 2)           *
   477                 ;*                                                       *
   478                 ;*********************************************************
   479
   480                        EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-016

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   481                  ;+++            KEY DECODING        +++
   482                  KOPE:
   483 0198 0931D               SKE     RKCODL,#CKRET1 AND 0FH
   484 0199 0C19F               BR      NOP100
   485 019A 0B30E               SKNE    RKCODH,#CKSRB0 AND 0FH            ;Was the [+] key pressed?
   486 019B 1D091               MOV     RREG9,#CADD AND 0FH               ;Yes
   487 019C 0B300               SKNE    RKCODH,#CKSRC0 AND 0FH            ;Was the [-] key pressed?
   488 019D 1D092               MOV     RREG9,#CSUB AND 0FH               ;Yes
   489 019E 0C1A3               BR      NOP200
   490                  NOP100:                                           ;Was the [x] key pressed?
   491 019F 0B30E               SKNE    RKCODH,#CKSRB0 AND 0FH            ;Yes
   492 01A0 1D094               MOV     RREG9,#CMUL AND 0FH               ;Was the [÷] key pressed?
   493 01A1 0B300               SKNE    RKCODH,#CKSRCO AND 0FH            ;Yes
   494 01A2 1D098               MOV     RREG9,#CDIV AND 0FH
   495
   496                  NOP200:
   497              1           CLR1    FFALSE
 +   1 01A3 1468B   1           AND     .MF.FFALSE SHR 4,#.DF(NOT FFALSE AND 0FH)
   498 01A4 0B642               SKNE    RMODE.#COPSEL AND 0FH             ;Operator selection mode?
   499 01A5 0C1B9               BR      NOP500                            ;Yes
   500              1           SKT1    FOPEND
 +   1 01A6 1E682   1           SKT     .MF.FOPEND SHR 4,#DF.FOPEND AND 0FH
   501 01A7 0C1AA               BR      NOP300
   502              1           CLR1    FOPEND
 +   1 01A8 1468D   1           AND     .MF.FOPEND SHR 4,#.DF.(NOT FOPEND AND 0FH)
   503 01A9 0C1B9               BR      NOP500
   504                  NOP300:
   505              1           SKT1    FPER                      ;Is percentage calculation in progress?
 +   1 01AA 1E681   1           SKT     .MF.FPER SHR 4,#.DF.FPER AND 0FH
   506 01AB 0C1B0               BR      NOP400                    ;No
   507              1           CLR1    FPER                      ;Yes
 +   1 01AC 1468E   1           AND     .MF.FPER SHR 4,#.DF.(NOT FPER AND 0FH)
   508 01AD 1B094               SKLT    RREG9,#CMUL AND 0FH       ;Was the [+] or [-] key pressed?
   509 01AE 0C1B9               BR      NOP500                    ;No
   510 01AF 0C1C1               BR      NOP700                    ;Yes -> Percentage calculation
   511
   512                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-017

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   513                  ;+++    Data transfer (display data area
   514                  ;                       -> floating-point register 2) +++
   515                  NOP400:
   516 01B0 EXTRN               CALL    STRNDY
   517
   518 01B1 09644               SKE     RMODE,#CLEFT AND OFH              ;First-term input mode?
   519 01B2 0C1BC               BR      NOP600                            ;No: Second-term input mode
   520
   521                  ;+++    Data transfer (floating-point register 2
   522                  ;                       -> floating-point register 1) +++
   523 01B3 1D001               MOV     RREG0,#.DM.RYSIGN SHR 4 AND 0FH
   524 01B4 16008               OR      RREG0,#CSTBK1 AND 0FH
   525 01B5 1D010               MOV     RREG1,#.DM.RXSIGN SHR 4 AND 0FH
   526 01B6 16018               OR      RREG1,#CSTBK1 AND 0FH
   527 01B7 EXTRN               CALL    STRAN
   528 01B8 EXTRN               CALL    SRYCLR                  ;Clears floating-point 2.
   529                  NOP500:
   530 01B9 18609               ST      ROPE,RREG9              ;Restore operator data which was saved.
   531 01BA 1D642               MOV     RMODE,#COPSEL AND 0FH   ;Sets the operator selection mode.
   532 01BB 0C1CF               BR      NOP900                  ;RET
   533
   534                  NOP600:
   535 01BC 08600               LD      RREG0,ROPE
   536 01BD 18610               ST      RCOM,RREG0
   537 01BE 18609               ST      ROPE,RREG9
   538 01BF 1D642               MOV     RMODE,#COPSEL AND 0FH   ;Sets the operator selection mode.
   539 01C0 0C1CE               BR      NOP800
   540
   541                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-018

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   542                   ;+++     Data restoration (saving register ->
   543                   ;         floating-point register 1) - percentage calculation +++
   544                   NOP700:
   545 01C1 18609                 ST       ROPE,RREG9
   546 01C2 18619                 ST       RCOM,RREG9
   547 01C3 1D000                 MOV      RREG0,#.DM.RXSIGN SHR 4 AND 0FH
   548 01C4 16008                 OR       RREG0,#CSTBK1 AND 0FH
   549 01C5 1D011                 MOV      RREG1,#.DM.RYSIGN SHR 4 AND 0FH
   550 01C6 16018                 OR       RREG1,#CSTBK1 AND 0FH
   551 01C7 EXTRN                 CALL     STRAN
   552 01C8 1D002                 MOV      RREG0,#.DM.RZLSD SHR 4 AND 0FH
   553 01C9 1D010                 MOV      RREG1,#.DM.RXSIGN SHR 4 AND 0FH
   554 01CA 16018                 OR       RREG1,#CSTBK1 AND 0FH
   555 01CB EXTRN                 CALL     STRAN
   556 01CC 1D644                 MOV      RMODE,#CLEFT AND 0FH              ;Sets the first-term input mode.
   557             1              SET1     FOPEND
 +   1 01CD 16682  1              OR       .MF.FOPEND SHR 4,#.DF.FOPEND AND 0FH
   558
   559                   NOP800:                                            ;Sets and operation request.
   560             1              SET1     FOPREQ
 +   1 01CE 16694  1              OR       .MF.FOPREQ SHR 4,#.DF.FOPREQ AND 0FH
   561
   562                   NOP900:
   563 01CF 070E0                 RET
   564
   565                            EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-019

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   566                  ;*************************************************************
   567                  ;*                                                           *
   568                  ;*    Percent Key Processing                                 *
   569                  ;*                                                           *
   570                  ;*       INPUT    : ROPE (Operator Area)                     *
   571                  ;*                : RMODE (Mode Area)                        *
   572                  ;*                : FPER (Percent Flag)                      *
   573                  ;*                : RXLSD - RXMSD (Floating-point            *
   574                  ;*                        Register 1, mantissa)              *
   575                  ;*       OUTPUT   : RZLSD - RZMSD (Saving Register)          *
   576                  ;*                : RCOM (Area for the operator to be executed)*
   577                  ;*                : RYEXP (Floating-point                    *
   578                  ;*                         Register 2, characteristic)       *
   579                  ;*                : FOPEND (Operation end flag)              *
   580                  ;*                : FOPREQ (Operation request flag)          *
   581                  ;*                : FPER (Percent Flag)                      *
   582                  ;*                : FFALSE (Illegal-input flag)              *
   583                  ;*                                                           *
   584                  ;*************************************************************
   585                  KPER:
   586              1           SKT1    FPER                      ;Has the [%] key already been pressed?
 +   1 01D0 1E681   1           SKT     .MF.FPER SHR 4,#.DF.FPER AND 0FH
   587 01D1 0B642               SKNE    RMODE,#COPSEL AND 0FH     ;Operator selection mode?
   588 01D2 0C1E9               BR      NPR800                    ;Yes -> RET
   589 01D3 0B641               SKNE    RMODE,#CRIGHT AND 0FH     ;Second-term input mode?
   590 01D4 0C1D7               BR      NPR200                    ;Yes
   591              1           CLR1    FOPEND                    ;No
 +   1 01D5 1468D   1           AND     .MF.FOPEND SHR 4,#.DF.(NOT FOPEND AND 0FH)
   592 01D6 0C1DA               BR      NPR400
   593
   594                  NPR200:
   595 01D7 1B604               SKLT    ROPE,#CMUL AND 0FH        ;Was the [x] or [÷] key pressed?
   596 01D8 0C1DC               BR      NPR600                    ;Yes -> Percentage calculation
   597 01D9 1D644               MOV     RMODE,#CLEFT AND 0FH      ;No: Sets the first-term input mode.
   598                  NPR400:
   599              1           SET1    FFALSE
 +   1 01DA 16684   1           OR      .MF.FFALSE SHR 4,#.DF.FFALSE AND 0FH
   600 01DB 0C1E9               BR      NPR800                    ;RET
   601
   602                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-020

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   603                  ;+++     Saves data (floating-point register 1
   604                  ;                -> saving register)-Percentage calculation  +++
   605                  NPR600:
   606 01DC 1D000               MOV      RREG0,#.DM.RXSIGN SHR 4 AND 0FH
   607 01DD 16008               OR       RREG0,#CSTBK1 AND 0FH
   608 01DE 1D012               MOV      RREG1,#.DM.RZLSD SHR 4 AND 0FH
   609 01DF EXTRN               CALL     STRAN
   610 01E0 EXTRN               CALL     STRNDY
   611              1           BANK1
 +   1 01E1 1D791   1           MOV      BANK,#01H
   612 01E2 11142               SUB      RYEXP,#2H
   613 01E3 13150               SUBC     RYEXP+1,#0H
   614              1           BANK0
 +   1 01E4 1D790   1           MOV      BANK,#00H
   615              1           SET1     FPER
 +   1 01E5 16681   1           OR       .MF.FPER SHR 4,#.DF.FPER AND 0FH
   616 01E6 08600               LD       RREG0,ROPE
   617 01E7 18610               ST       RCOM,RREG0
   618              1           SET1     FOPREQ              ;Sets an operation request.
 +   1 01E8 16694   1           OR       .MF.FOPREQ SHR 4,#.DF.FOPREQ AND 0FH
   619
   620                  NPR800:
   621 01E9 070E0               RET
   622
   623                          EJECT
```

```
AS17K V1.10 V4  << D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-021

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   624                   ;*************************************************************
   625                   ;*                                                           *
   626                   ;*    Equal Key Processing                                   *
   627                   ;*                                                           *
   628                   ;*       INPUT    : ROPE (Operator Area)                     *
   629                   ;*                : RMODE (Mode Area)                        *
   630                   ;*                : FPER (Percent Flag)                      *
   631                   ;*                : RXSIGN - RXMSD (Floating-point           *
   632                   ;*                        Register 1)                        *
   633                   ;*       OUTPUT   : RCOM (Area for the operator to be executed)*
   634                   ;*                : RMODE (Mode Area)                        *
   635                   ;*                : FOPEND (Operation end flag)              *
   636                   ;*                : FFALSE (Illegal-input flag)              *
   637                   ;*                : FOPREQ (Operation request flag)          *
   638                   ;*                : RYSIGN - RYMSD                           *
   639                   ;*                          (Floating-point register 2)      *
   640                   ;*                                                           *
   641                   ;*************************************************************
   642                   KEQU:
   643              1            SKT1     FPER                      ;Was the [%] key pressed?
 +   1 01EA 1E681   1            SKT      .MF.FPER SHR 4,#.DF.FPER AND 0FH
   644 01EB 0C1EE               BR       NEQ100                    ;No
   645              1            CLR1     FPER                      ;Yes
 +   1 01EC 1468E   1            AND      .MF.FPER SHR 4,#.DF.(NOT FPER AND 0FH)
   646 01ED 0C203               BR       NEQ600
   647
   648                   NEQ100:
   649 01EE 09644               SKE      RMODE,#CLEFT AND 0FH      ;First-term input mode?
   650 01EF 0C1F2               BR       NEQ200                    ;No
   651              1            SET1     FFALSE                    ;Yes
 +   1 01F0 16684   1            OR       .MF.FFALSE SHR 4,#.DF.FFALSE AND 0FH
   652 01F1 0C205               BR       NEQ700                    ;RET
   653
   654                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-022

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   655                 NEQ200:
   656 01F2 08600              LD      RREG0,ROPE
   657 01F3 18610              ST      RCOM, RREG0
   658 01F4 09642              SKE     RMODE,#COPSEL AND 0FH  ;Operator selector mode?
   659 01F5 0C201              BR      NEQ400                 ;No
   660 01F6 1B614              SKLT    RCOM,#CMUL             ;Was the [+] or [-] key pressed?
   661 01F7 0C1FB              BR      NEQ300                 ;No
   662 01F8 EXTRN              CALL    SRYCLR          ;Yes:  Clears floating-point register 2.
   663            1            BANK0
 +   1 01F9 1D790  1           MOV     BANK,#00H
   664 01FA 0C202              BR      NEQ500
   665
   666                 ;+++    Data transfer (floating-point register 1
   667                 ;               -> floating-point register 2)    +++
   668                 NEQ300:
   669 01FB 1D000              MOV     RREG0,#.DM.RXSIGN SHR 4 AND 0FH
   670 01FC 16008              OR      RREG0,#CSTBK1 AND 0FH
   671 01FD 1D011              MOV     RREG1,#.DM.RYSIGN SHR 4 AND 0FH
   672 01FE 16018              OR      RREG1,#CSTBK1 AND 0FH
   673 01FF EXTRN              CALL    STRAN
   674 0200 0C202              BR      NEQ500
   675
   676                 NEQ400:
   677 0201 EXTRN              CALL    STRNDY
   678
   679                 NSG500:
   680            1            SET1    FOPREQ                 ;Sets an operation request.
 +   1 0202 16694  1           OR      .MF.FOPREQ SHR 4,#.DF.FOPREQ AND 0FH
   681
   682                 NEQ600:
   683            1            SET1    FOPEND
 +   1 0203 16682  1           OR      .MF.FOPEND SHR 4,#.DF.FOPEND AND 0FH
   684 0204 1D644              MOV     RMODE,#CLEFT AND 0FH   ;Sets the first-term input mode.
   685
   686                 NEQ700:
   687 0205 070E0              RET
   688
   689                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-023

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   690                   ;*************************************************************
   691                   ;*                                                           *
   692                   ;*  Branch operation processing, error handling              *
   693                   ;*                                                           *
   694                   ;*      INPUT   : RCOM (Area for the operator to be executed) *
   695                   ;*      OUTPUT  : RNUMC (Numeric key counter)                *
   696                   ;*              : RDEXP - RDMSD (Display data)               *
   697                   ;*              : RMODE (Mode area)                          *
   698                   ;*              : FOPREQ (Operation request flag)            *
   699                   ;*                                                           *
   700                   ;*************************************************************
   701                   MOPBRN:
   702              1            CLR1    FOPREQ
 +   1 0206 1469B  1            AND     .MF. FOPREQ SHR 4,#.DF. (NOT FOPREQ AND OFH)
   703              1            CLR1    LCDEN                        ;LCD display off
 +   1 0207 07331  1            PEEK    WR,.MF.LCDEN SHR 4
 +   2 0208 14787  1            AND     WR,#.DF.(NOT LCDEN) AND 0FH
 +   3 0209 07321  1            POKE    .MF.LCDEN SHR 4,WR
   704              1            BANK1
 +   1 020A 1D791  1            MOV     BANK,#01H
   705 020B 1D300               MOV     ROPFLG,#0000B                ;Clears the operation flag.
   706              1            BANK0
 +   1 020C 1D790  1            MOV     BANK,#00H
   707 020D 0B618               SKNE    RCOM,#CDIV AND 0FH           ;Divide instruction?
   708 020E 0C219               BR      LOB300                       ;Yes
   709 020F 0B614               SKNE    RCOM,#CMUL AND 0FH           ;Multiply instruction?
   710 0210 0C217               BR      LOB200                       ;Yes
   711 0211 0B612               SKNE    RCOM,#CSUB AND 0FH           ;Subtract instruction?
   712 0212 0C215               BR      LOB100                       ;Yes
   713
   714                   ;+++           Add operation                +++
   715 0213 EXTRN               CALL    SFPADD
   716 0214 0C21D               BR      LOB400
   717
   718                   ;+++           Subtract operation           +++
   719                   LOB100:
   720 0215 EXTRN               CALL    SFPSUB
   721 0216 0C21D               BR      LOB400
   722
   723                   ;+++           Multiply operation           +++
   724                   LOB200:
   725 0217 EXTRN               CALL    SFPMULT
   726 0218 0C21D               BR      LOB400
   727
   728                   ;+++           Divide operation             +++
   729                   LOB300:
   730 0219 EXTRN               CALL    SFPDIV
   731              1            BANK1
 +   1 021A 1D791  1            MOV     BANK,#01H
   732              1            SKF1    FDVERR                       ;Is the divisor 0?
 +   1 021B 1F304  1            SKF     .MF.FDVERR SHR 4,#.DF.FDVERR AND 0FH
   733 021C 0C231               BR      LOB700                       ;Yes -> Eror handling
   734
   735                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-024

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   736                   ;+++       Overflow                          +++
   737                   LOB400:
   738              1            BANK1
 +   1 021D 1D791   1            MOV      BANK,#01H
   739 021E 08040               LD       RREG0,RXEXP
   740 021F 08051               LD       RREG1,RXEXP+1H
   741              1            BANK0
 +   1 0220 1D790   1            MOV      BANK,#00H
   742 0221 1B018               SKLT     RREG1,#8H                    ;Is the exponent smaller than 0?
   743 0222 0C228               BR       LOB500                       ;Yes
   744 0223 11008               SUB      RREG0,#(CEXPMAX+1H) AND 0FH  ;Is the exponent between 0 and 7?
   745 0224 13010               SUBC     RREG1,#(CEXPMAX+1H) SHR 4 AND 0FH
   746              1            SKF1     CY
 +   1 0225 1F7F4   1            SKF      .MF.CY SHR 4,#.DF.CY AND 0FH
   747 0226 0C22C               BR       LOB600                       ;Yes -> Operation result display
   748 0227 0C231               BR       LOB700                       ;No -> Error handling
   749
   750                   LOB500:
   751 0228 1100B               SUB      RREG0,#CEXPMIN AND 0FH       ;Is the exponent between -5 and -1?
   752 0229 1301F               SUBC     RREG1,#CEXPMIN SHR 4 AND 0FH
   753              1            SKF1     CY
 +   1 022A 1F7F4   1            SKF      .MF.CY SHR 4,#.DF.CY AND 0FH
   754 022B EXTRN               CALL     SRXCLR                       ;No:  Operation result
   755
   756                   ;+++       Operation result conversion       +++
   757                   LOB600:
   758 022C EXTRN               CALL     SFIX                         ;Operation result -> display data
   759 022D 1D148               MOV      RDEXP,#CEXPINI AND 0FH
   760 022E 1D150               MOV      RDEXP+1H,#CEXPINI SHR 4 AND 0FH
   761 022F 1D100               MOV      RNUMC,#CNONUM AND 0FH
   762 0230 0C235               BR       LOB800                       ;-> Operation result display
   763
   764                   ;+++       Error handling                    +++
   765                   LOB700:
   766              1            BANK0
 +   1 0231 1D790   1            MOV      BANK,#00H
   767 0232 1D648               MOV      RMODE,#CERROR AND 0FH        ;Sets the error mode.
   768 0233 EXTRN               CALL     SDPINI
   769 0234 1D18C               MOV      RDLSD,#CERRDP AND 0FH        ;Least significant digit of the display data
   770                                                                ;<- [E] display data
   771
   772                   LOB800:
   773 0235 EXTRN               CALL     SDISP                        ;Display data output
   774              1            SET1     LCDEN                        ;LCD display on
 +   1 0236 07331   1            PEEK     WR,.MF.LCDEN SHR 4
 +   2 0237 16788   1            OR       WR,#.DF.LCDEN AND 0FH
 +   3 0238 07321   1            POKE     .MF.LCDEN SHR 4,WR
   775
   776 0239 070E0               RET
   777
   778                          END

      TOTAL ERRORS    = 0
      TOTAL WARNINGS = 0
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:24:27 12/20/93 PAGE 03-025

PROG =

SOURCE = CALC2.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT

END OF LIST
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-001

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
     1                 ;*************************************************
     2                 ;*                                               *
     3                 ;*  USER NAME   : NEC CORPORATION                *
     4                 ;*                                               *
     5                 ;*  SYSTEM NAME : 17K APPLICATION (CALCULATOR)   *
     6                 ;*                                               *
     7                 ;*  CPU         : µPD17201A                      *
     8                 ;*                                               *
     9                 ;*  LAST UPDATE : '93/12/20  11:00               *
    10                 ;*                                               *
    11                 ;*************************************************
    12                 ;*************************************************
    13                 ;*                                               *
    14                 ;*  FILE NAME      : CALC3.ASM                   *
    15                 ;*                                               *
    16                 ;*  INCLUDE 10 ROUTINES:                         *
    17                 ;*          SRAMCR :  RAM ALL CLEAR              *
    18                 ;*          SDPINI :  DISPLAY DATA AREA          *
    19                 ;*                          INITIALIZE           *
    20                 ;*          SRYCLR :  SECOND OPERAND DATA        *
    21                 ;*                          AREA CLEAR           *
    22                 ;*          SDSHFD :  DISPLAY DATA AREA          *
    23                 ;*                          DOWN SHIFT           *
    24                 ;*          SUSHFD :  DISPLAY DATA AREA          *
    25                 ;*                          UP SHIFT             *
    26                 ;*          SUSHFY :  SECOND OPERAND DATA        *
    27                 ;*                          AREA DOWN SHIFT      *
    28                 ;*          STRAN  :  TRANSFER DATA              *
    29                 ;*          SFIX   :  TRANSFER FLOATING          *
    30                 ;*                      POINT NUMBER TO          *
    31                 ;*                      FIXED POINT NUMBER       *
    32                 ;*          STRNDY :  TRANSFER DISPLAY DATA      *
    33                 ;*                      TO SECOND OPERAND        *
    34                 ;*          SDISP  :  DISPLAY DATA OUTPUT        *
    35                 ;*                                               *
    36                 ;*************************************************
    37
    38                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-002

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    39                 ;*************************************************
    40                 ;*              External reference               *
    41                 ;*************************************************
    42                 EXTRN   MEM: RREG0,RREG1,RREG2   ;General-purpose register
    43                 EXTRN   MEM: RREG3,RREG4
    44                 EXTRN   MEM: RNUMC               ;Numeric key counter
    45                 EXTRN   MEM: RPTLOC              ;Decimal-point position area
    46                 EXTRN   MEM: RSINLOC             ;Sign position area
    47                 EXTRN   MEM: RDSIGN              ;Operation result sign area
    48                 EXTRN   MEM: RDEXP               ;Display data exponent area
    49                 EXTRN   MEM: RDLSD,RDMSD         ;Display data area
    50                 EXTRN   MEM: RXSIGN              ;Floating-point register 1, sign
    51                 EXTRN   MEM: RYSIGN              ;Floating-point register 2, sign
    52                 EXTRN   MEM: RYEXP               ;Floating-point register 2, characteristic
    53                 EXTRN   MEM: RYMSD,RYLSD         ;Floating-point register 2, mantissa
    54
    55                 EXTRN   DAT: CSTMPE           ;MPH: Indirect addressing (BANK0)
    56                 EXTRN   DAT: CSTBK2           ;     Indirect addressing (BANK2)
    57                 EXTRN   DAT: CSTBK1           ;MPL: Indirect addressing (BANK1)
    58                 EXTRN   DAT: CPLINI           ;Initial value of the decimal-point position area
    59                 EXTRN   DAT: CEXPINI          ;Initial value of the display data exponent area
    60                 EXTRN   DAT: CZERO            ;[0] display data
    61                 EXTRN   DAT: CSPACE           ;Space display data
    62                 EXTRN   DAT: CMINUS           ;[-] display data
    63                 EXTRN   DAT: CDPON            ;Decimal-point display segment data
    64                 EXTRN   DAT: CSINOFF,CSINMN   ;No sign display/minus
    65
    66                 ;*************************************************
    67                 ;*              External definition              *
    68                 ;*************************************************
    69                 PUBLIC  SRAMCR,SDPINI,SDSHFD,SUSHFD,STRAN,SFIX,STRNDY,SRYCLR,SDISP
    70
    71                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-003

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    72                 ;*************************************************
    73                 ;*                                               *
    74                 ;*        Macro definition file reading          *
    75                 ;*                                               *
    76                 ;*************************************************
    77               1         INCLUDE 'PUSHMP.ASM'
 +   1               1
 +   2               1         EXTRN   MEM:  RREG10,RREG11
 +   3               1
 +   4               1 ;*************************************************
 +   5               1 ;*                                               *
 +   6               1 ;*       Macro definition for saving the         *
 +   7               1 ;*       data memory row address pointer         *
 +   8               1 ;*                                               *
 +   9               1 ;*************************************************
 +  10               1 APUSHMP MACRO
 +  11               1         LD      RREG10,MPH
 +  12               1         LD      RREG11,MPL
 +  13               1         ENDM
 +  14               1
 +  15               1 ;*************************************************
 +  16               1 ;*                                               *
 +  17               1 ;*      Macro definition for restoring the       *
 +  18               1 ;*      data memory row address pointer          *
 +  19               1 ;*                                               *
 +  20               1 ;*************************************************
 +  21               1 APOPMP  MACRO
 +  22               1         ST      MPH,RREG10
 +  23               1         ST      MPL,RREG11
 +  24               1         ENDM
 +  25               1
 +  26               1         EOF
    78
    79                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-004

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    80                 ;*************************************************
    81                 ;*                                               *
    82                 ;*           RAM all clear processing            *
    83                 ;*                                               *
    84                 ;*           INPUT   : Nothing                   *
    85                 ;*           OUTPUT  : ALL RAM AREA              *
    86                 ;*                                               *
    87                 ;*************************************************
    88                 SRAMCR:
    89             1           BANK0
 +   1 023A 1D790  1           MOV     BANK,#00H
    90 023B 1D7A8               MOV     MPH,#CSTMPE AND 0FH             ;MPE set
    91 023C 1D7B0               MOV     MPL,#.DM.RREG2 SHR 4 AND 0FH
    92 023D 1D002               MOV     RREG0,#.DM.RREG2 AND 0FH
    93 023E 1D010               MOV     RREG1,#CZERO AND 0FH
    94                 JRC200:
    95 023F 0A010               MOV     @RREG0,RREG1
    96 0240 10001               ADD     RREG0,#1H
    97 0241 09000               SKE     RREG0,#0H
    98 0242 0C23F               BR      JRC200
    99
   100 0243 107B1               ADD     MPL,#1H
   101 0244 0B7B7               SKNE    MPL,#7H                 ;Did BANK0 or BANK2 terminate?
   102 0245 0C24A               BR      JRC400                          ;Yes
   103 0246 097BF               SKE     MPL,#0FH                        ;Did BANK1 terminate?
   104 0247 0C23F               BR      JRC200                          ;No -> Next column
   105 0248 1D7A9               MOV     MPH,#CSTBK2 AND 0FH             ;Yes -> BANK2
   106 0249 0C24C               BR      JRC600
   107
   108                 JRC400:
   109 024A 0B7A9               SKNE    MPH,#CSTBK2 AND 0FH             ;Did BANK2 terminate?
   110 024B 0C24E               BR      JRC800                          ;Yes -> RET
   111                 JRC600:
   112 024C 107B1               ADD     MPL,#1H
   113 024D 0C23F               BR      JRC200
   114
   115                 JRC800:
   116 024E 070E0               RET
   117
   118                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-005

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   119                   ;*********************************************
   120                   ;*                                           *
   121                   ;* Display data area initialization          *
   122                   ;*                                           *
   123                   ;* INPUT  : Nothing                          *
   124                   ;* OUTPUT : RPTLOC (Decimal-point position area)*
   125                   ;*        : RDLSD - RDMSD (Display data)     *
   126                   ;*                                           *
   127                   ;*********************************************
   128                   SDPINI:
   129              1            APUSHMP                          ;Saves the memory pointer.
 +   1 024F 087AA  1            LD      RREG10,MPH
 +   2 0250 087BB  1            LD      RREG11,MPL
   130
   131              1            BANK0
 +   1 0251 1D790  1            MOV     BANK,#00H
   132 0252 1D180               MOV     RDLSD,#CZERO AND 0FH     ;Least significant digit <- [0] display data
   133
   134 0253 1D7A8               MOV     MPH,#CSTMPE AND 0FH            ;MPE set
   135 0254 1D7B1               MOV     MPL,#.DM.RDLSD SHR 4 AND 0FH   ;Sets the row address of
   136                                                                 ;the display area.
   137 0255 1D009               MOV     RREG0,#.DM.(RDLSD+1H) AND 0FH  ;Sets the column address.
   138 0256 1D01A               MOV     RREG1,#CSPACE AND 0FH          ;Space display data
   139                   JDI200:
   140 0257 0A010               MOV     @RREG0,RREG1
   141 0258 10001               ADD     RREG0,#1H
   142 0259 09000               SKE     RREG0,#0H                      ;Did the initialization terminate?
   143 025A 0C257               BR      JDI200                         ;No -> Next digit
   144 025B 1D117               MOV     RPTLOC,#CPLINI AND 0FH         ;Resets the decimal-point position.
   145
   146              1            APOPMP                                 ;Restores the memory pointer.
 +   1 025C 187AA  1            ST      MPH,RREG10
 +   2 025D 187BB  1            ST      MPL,RREG11
   147
   148 025E 070E0               RET
   149
   150                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-006

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   151                  ;*********************************************
   152                  ;*                                           *
   153                  ;*     Clearing floating-point register 2    *
   154                  ;*                                           *
   155                  ;*     INPUT  : Nothing                      *
   156                  ;*     OUTPUT : RYSIGN - RDMSD               *
   157                  ;*              (Floating-point register 2)  *
   158                  ;*                                           *
   159                  ;*********************************************
   160                  SRYCLR:
   161             1            APUSHMP                        ;Saves the memory pointer.
 +   1 025F 087AA  1            LD      RREG10,MPH
 +   2 0260 087BB  1            LD      RREG11,MPL
   162
   163             1            BANK0
 +   1 0261 1D790  1            MOV     BANK,#00H
   164 0262 1D7A8               MOV     MPH,#CSTMPE AND 0FH            ;MPE set
   165 0263 1D7B1               MOV     MPL,#.DM.RYSIGN SHR 4 AND 0FH ;Sets the row address of
   166 0264 167B8               OR      MPL,#CSTBK1 AND 0FH            ;floating-point register 2.
   167 0265 1D003               MOV     RREG0,#.DM.RYSIGN AND 0FH      ;Sets the column address.
   168 0266 1D010               MOV     RREG1,#CZERO AND 0FH           ;Space display data
   169                  JYC200:
   170 0267 0A010               MOV     @RREG0,RREG1
   171 0268 10001               ADD     RREG0,#1H
   172 0269 09000               SKE     RREG0,#0H                      ;Clear terminated?
   173 026A 0C267               BR      JYC200                         ;No -> Next digit
   174
   175             1            APOPMP                         ;Restores the memory pointer.
 +   1 026B 187AA  1            ST      MPH, RREG10
 +   2 026C 187BB  1            ST      MPL,RREG11
   176
   177 026D 070E0               RET
   178
   179                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-007

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   180                 ;*************************************************
   181                 ;*                                               *
   182                 ;*    Display data area shift down               *
   183                 ;*                                               *
   184                 ;*    INPUT  : RDLSD - RDMSD (Display data)      *
   185                 ;*    OUTPUT : RDLSD - RDMSD (Display data)      *
   186                 ;*                                               *
   187                 ;*************************************************
   188                 SDSHFD:
   189             1           APUSHMP                        ;Saves the memory pointer.
 +   1 026E 087AA  1           LD      RREG10,MPH
 +   2 026F 087BB  1           LD      RREG11,MPL
   190
   191             1           BANK0
 +   1 0270 1D790  1           MOV     BANK,#00H
   192 0271 1D7A8              MOV     MPH,#CSTMPE AND 0FH            ;MPE set
   193 0272 1D7B1              MOV     MPL,#.DM.RDLSD SHR 4 AND 0FH   ;Sets the row address of the
   194                                                                ;display data area.
   195 0273 1D007              MOV     RREG0,#.DM.(RDLSD-1H) AND 0FH  ;Sets the column address.
   196                 JDD200:
   197 0274 1A010              MOV     RREG1,@RREG0                   ;One-byte shift down
   198 0275 11001              SUB     RREG0,#1H
   199 0276 0A010              MOV     @RREG0,RREG1
   200 0277 10002              ADD     RREG0,#2H
   201 0278 09000              SKE     RREG0,#0H                      ;End of shift down?
   202 0279 0C274              BR      JDD200                         ;No -> Next digit
   203 027A 1D1F0              MOV     RDMSD,#CZERO AND 0FH  ;Most significant digit <- [0] display data
   204
   205             1           APOPMP                         ;Restores the memory pointer.
 +   1 027B 187AA  1           ST      MPH,RREG10
 +   2 027C 187BB  1           ST      MPL,RREG11
   206
   207 027D 070E0              RET
   208
   209                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-008

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   210                   ;*************************************************
   211                   ;*                                               *
   212                   ;*    Display data area shift up                 *
   213                   ;*                                               *
   214                   ;*    INPUT  : RDLSD - RDMSD (Display data)      *
   215                   ;*    OUTPUT : RDLSD - RDMSD (Display data)      *
   216                   ;*                                               *
   217                   ;*************************************************
   218                   SUSHFD:
   219              1            APUSHMP                          ;Saves the memory pointer.
 +   1 027E 087AA  1            LD      RREG10,MPH
 +   2 027F 087BB  1            LD      RREG11,MPL
   220
   221              1            BANK0
 +   1 0280 1D790  1            MOV     BANK,#00H
   222 0281 1D7A8               MOV     MPH,#CSTMPE AND 0FH              ;MPE set
   223 0282 1D7B1               MOV     MPL,#.DM.RDLSD SHR 4 AND 0FH     ;Sets the row address of the
   224                                                                   ;display data area.
   225 0283 1D00E               MOV     RREG0,#.DM.(RDMSD-1H) AND 0FH    ;Sets the column address.
   226                   JUD200:
   227 0284 1A010               MOV     RREG1,@RREG0                     ;One-byte shift up
   228 0285 10001               ADD     RREG0,#1H
   229 0286 0A010               MOV     @RREG0, RREG1
   230 0287 11002               SUB     RREG0,#2H
   231 0288 1B008               SKLT    RREG0,#.DM.RDLSD AND 0FH         ;End of shift up?
   232 0289 0C284               BR      JUD200                           ;No -> Next digit
   233 028A 1D180               MOV     RDLSD,#CZERO AND 0FH     ;Least significant digit <- [0] display data
   234
   235              1            APOPMP                           ;Restores the memory pointer.
 +   1 028B 187AA  1            ST      MPH,RREG10
 +   2 028C 187BB  1            ST      MPL,RREG11
   236
   237 028D 070E0               RET
   238
   239                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-009

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   240                  ;*********************************************
   241                  ;*                                           *
   242                  ;* Floating-point register 2 shift up        *
   243                  ;*                                           *
   244                  ;* INPUT  : RYLSD - RYMSD                    *
   245                  ;*        (Floating-point register 2, mantissa) *
   246                  ;* OUTPUT : RYLSD - RYMSD                    *
   247                  ;*        (Floating-point register 2, mantissa) *
   248                  ;*                                           *
   249                  ;*********************************************
   250                  SUSHFY:
   251             1            APUSHMP                          ;Saves the memory pointer.
 +   1 028E 087AA  1            LD      RREG10,MPH
 +   2 028F 087BB  1            LD      RREG11,MPL
   252
   253             1            BANK0
 +   1 0290 1D790  1            MOV     BANK,#00H
   254 0291 1D7A8               MOV     MPH,#CSTMPE AND 0FH              ;MPE set
   255 0292 1D7B1               MOV     MPL,#.DM.RYLSD SHR 4 AND 0FH     ;Sets the row address of
   256 0293 167B8               OR      MPL,#CSTBK1 AND 0FH              ;floating-point register 2.
   257 0294 1D00E               MOV     RREG0,#.DM.(RYMSD-1H) AND 0FH    ;Sets the column address.
   258                  JUY200:
   259 0295 1A010               MOV     RREG1,@RREG0                     ;One-byte shift up
   260 0296 10001               ADD     RREG0,#1H
   261 0297 0A010               MOV     @RREG0,RREG1
   262 0298 11002               SUB     RREG0,#2H
   263 0299 1B006               SKLT    RREG0,#.DM.RYLSD AND 0FH         ;End of shift up?
   264 029A 0C295               BR      JUY200                           ;No -> Next digit
   265             1            BANK1
 +   1 029B 1D791  1            MOV     BANK,#01H
   266 029C 1D160               MOV     RYLSD,#CZERO AND 0FH     ;Least significant digit <- [0] display data
   267             1            BANK0
 +   1 029D 1D790  1            MOV     BANK,#00H
   268
   269             1            APOPMP                           ;Restores the memory pointer.
 +   1 029E 187AA  1            ST      MPH,RREG10
 +   2 029F 187BB  1            ST      MPL,RREG11
   270
   271 02A0 070E0               RET
   272
   273                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-010

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   274                 ;*************************************************
   275                 ;*                                               *
   276                 ;*   Data transfer                               *
   277                 ;*                                               *
   278                 ;*   INPUT  : RREG0 (Row address of the          *
   279                 ;*                  source area)                 *
   280                 ;*          : RREG1 (Row address of the          *
   281                 ;*                  destination area)            *
   282                 ;*          : RZLSD - RZMSD (Saving register)    *
   283                 ;*          : RXSIGN - RXMSD                     *
   284                 ;*            (Floating-point register 1)        *
   285                 ;*          : RYSIGN - RYMSD                     *
   286                 ;*            (Floating-point register 2)        *
   287                 ;*   OUTPUT : RDSIGN - RDMSD                     *
   288                 ;*            (Display data register)            *
   289                 ;*          : RZLSD - RZMSD (Saving register)    *
   290                 ;*          : RXSIGN - RXMSD                     *
   291                 ;*            (Floating-point register 1)        *
   292                 ;*          : RYSIGN - RYMSD                     *
   293                 ;*            (Floating-point register 2)        *
   294                 ;*                                               *
   295                 ;*************************************************
   296
   297                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-011

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   298                   STRAN:
   299              1            APUSHMP                    ;Saves the memory pointer.
 +   1 02A1 087AA  1            LD      RREG10,MPH
 +   2 02A2 087BB  1            LD      RREG11,MPL
   300
   301              1            BANK0
 +   1 02A3 1D790  1            MOV     BANK,#00H
   302 02A4 1D7A8               MOV     MPH,#CSTMPE AND 0FH         ;MPE set
   303 02A5 1D023               MOV     RREG2,#.DM.RDSIGN AND 0FH   ;Sets the column address.
   304                   JTR200:
   305 02A6 187B0               ST      MPL,RREG0          ;Sets the row address of the source area.
   306 02A7 1A032               MOV     RREG3,@RREG2
   307 02A8 187B1               ST      MPL,RREG1          ;Sets the row address of the destination area.
   308 02A9 0A032               MOV     @RREG2,RREG3
   309 02AA 10021               ADD     RREG2,#1H
   310 02AB 09020               SKE     RREG2,#0H          ;End of transfer?
   311 02AC 0C2A6               BR      JTR200             ;No -> Next digit
   312
   313              1            APOPMP                     ;Restores the memory pointer.
 +   1 02AD 187AA  1            ST      MPH,RREG10
 +   2 02AE 187BB  1            ST      MPL,RREG11
   314
   315 02AF 070E0               RET
   316
   317                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:47 12/20/93 PAGE 04-012

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   318                   ;*************************************************
   319                   ;*                                               *
   320                   ;*      Operation result conversion              *
   321                   ;*                                               *
   322                   ;* INPUT  : RXSIGN - RXMSD (Operation result)    *
   323                   ;* OUTPUT : RPTLOC (Decimal-point position area)*
   324                   ;*        : RSINLOC (Sign position area)          *
   325                   ;*        : RDSIGN - RDMSD                        *
   326                   ;*          (Display data register)               *
   327                   ;*                                               *
   328                   ;*************************************************
   329                   ;+++ Transfer of the operation result to the display data area +++
   330                   SFIX:
   331              1            BANK0
 +   1 02B0 1D790  1            MOV     BANK,#00H
   332 02B1 1D000               MOV     RREG0,#.DM.RXSIGN SHR 4 AND 0FH
   333 02B2 16008               OR      RREG0,#CSTBK1
   334 02B3 1D011               MOV     RREG1,#.DM.RDSIGN SHR 4 AND 0FH
   335 02B4 1C2A1               CALL    STRAN
   336
   337                   ;+++ Exponent data judgment                                  +++
   338 02B5 0B140               SKNE    RDEXP,#0H               ;Is exponent data 0?
   339 02B6 09150               SKE     RDEXP+1H,#0H
   340 02B7 0C2BA               BR      JFX100
   341 02B8 1D141               MOV     RDEXP,#1H               ;Yes:  Exponent data <- 1
   342 02B9 0C2BC               BR      JFX200
   343                   JFX100:
   344 02BA 0915F               SKE     RDEXP+1H,#0FH           ;Is exponent data smaller than 0?
   345 02BB 0C2BD               BR      JFX300                  ;No
   346                   JFX200:
   347 02BC 1C26E               CALL    SDSHFD
   348
   349                   ;+++ Positive/negative judgment of the operation result        +++
   350                   JFX300:
   351 02BD 09131               SKE     RDSIGN,#CSINMN AND 0FH ;Is the operation result smaller than 0?
   352 02BE 0C2C1               BR      JFX400                  ;No
   353 02BF 1D1FB               MOV     RDMSD,#CMINUS AND 0FH  ;Yes:  Sets the [-] display data in the
   354 02C0 0C2C2               BR      JFX500                  ;most significant digit of the display data.
   355                   JFX400:
   356 02C1 1D1FA               MOV     RDMSD,#CSPACE AND 0FH  ;Sets the space display data in the
   357                                                          ;most significant digit of the display data.
   358                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-013

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   359                  ;+++     Display data shift down
   360                  ;        (except the most significant digit)  +++
   361                  JFX500:
   362 02C2 0B15F               SKNE    RDEXP+1H,#0FH                 ;Is exponent data smaller than 0?
   363 02C3 0C2C7               BR      JFX600                        ;Yes -> Shift down
   364 02C4 08140               LD      RREG0,RDEXP                   ;No
   365 02C5 18110               ST      RPTLOC,RREG0
   366 02C6 0C2DC               BR      JFX800
   367
   368                  JFX600:
   369             1            APUSHMP                               ;Saves the memory pointer.
 +   1 02C7 087AA  1            LD      RREG10,MPH
 +   2 02C8 087BB  1            LD      RREG11,MPL
   370
   371 02C9 1D7A8               MOV     MPH,#CSTMPE AND 0FH           ;MPE set
   372 02CA 1D7B1               MOV     MPL,#.DM.RDLSD SHR 4 AND 0FH  ;Sets the row address of the
   373                                                                ;display data area.
   374                  JFX650:
   375 02CB 1D007               MOV     RREG0,#.DM.(RDLSD-1H) AND 0FH;Sets the column address.
   376                  JFX700:
   377 02CC 1A010               MOV     RREG1,@RREG0                  ;One-byte shift down
   378 02CD 11001               SUB     RREG0,#1H
   379 02CE 0A010               MOV     @RREG0,RREG1
   380 02CF 10002               ADD     RREG0,#2H
   381 02D0 0900F               SKE     RREG0,#.DM.RDMSD AND 0FH      ;End of shift down?
   382 02D1 0C2CC               BR      JFX700                        ;No -> Next digit
   383 02D2 1D1E0               MOV     RDMSD-1H,#CZERO AND 0FH
   384
   385             1            SET1    Z
 +   1 02D3 167F2  1            OR      .MF.Z SHR 4,#.DF.Z AND 0FH
   386 02D4 10141               ADD     RDEXP,#1H                     ;Increments exponent data.
   387 02D5 12150               ADDC    RDXEP+1H,#0H
   388             1            SKT1    Z                             ;Did exponent data become 0?
 +   1 02D6 1E7F2  1            SKT     .MF.Z SHR 4,#.DF.Z AND 0FH
   389 02D7 0C2CB               BR      JFX650                        ;No -> Repeats shift down.
   390
   391             1            APOPMP                                ;Yes:  Restores the memory pointer.
 +   1 02D8 187AA  1            ST      MPH,RREG10
 +   2 02D9 187BB  1            ST      MPL,RREG11
   392
   393 02DA 1D141               MOV     RDEXP,#1H
   394 02DB 1D111               MOV     RPTLOC,#1H
   395
   396                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-014

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   397                  +++          Zero suppression          +++
   398                  JFX800:
   399 02DC 1D12F               MOV     RSINLOC,#0FH
   400                  JFX850:
   401 02DD 0B180               SKNE    RDLSD,#CZERO AND 0FH   ;Did zero suppression terminate?
   402 02DE 0B117               SKNE    RPTLOC,#7H
   403 02DF 0C2E5               BR      JFX900                 ;Yes -> RET
   404 02E0 10111               ADD     RPTLOC,#1H             ;No:  Increments the
   405                                                         ;decimnal-point position.
   406 02E1 1C26E               CALL    SDSHFD                 ;Shifts display data down.
   407 02E2 1D1FA               MOV     RDMSD,#CSPACE AND 0FH  ;Most significant digit of the display
   408                                                         ;data area <- Space display data
   409 02E3 11121               SUB     RSINLOC,#1H
   410 02E4 0C2DD               BR      JFX850
   411
   412                  JFX900:
   413 02E5 070E0               RET
   414
   415                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-015

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   416                   ;*********************************************
   417                   ;*                                           *
   418                   ;*      Display data conversion              *
   419                   ;*                                           *
   420                   ;*      INPUT  : RNUMC (Numeric key counter) *
   421                   ;*             : RDEXP - RDMSD (Display data) *
   422                   ;*      OUTPUT : RYSIGN - RYMSD              *
   423                   ;*               (Floating-point register 2) *
   424                   ;*                                           *
   425                   ;*********************************************
   426                   STRNDY:
   427 02E6 1C25F                CALL    SRYCLR                      ;Clears floating-point register 2.
   428 02E7 09148                SKE     RDEXP,#CEXPINI AND 0FH      ;Was the decimal-point key pressed?
   429 02E8 0C2EB                BR      JTD100                      ;Yes
   430 02E9 08100                LD      RREG0,RNUMC                 ;No:  Display data exponent
   431 02EA 18140                ST      RDEXP,RREG0                 ;area <- Numeric key counter
   432
   433                   ;+++   Deletes a space from the display data  +++
   434                   JTD100:
   435              1            APUSHMP                             ;Saves the memory pointer.
 +   1 02EB 087AA  1             LD      RREG10,MPH
 +   2 02EC 087BB  1             LD      RREG11,MPL
   436
   437 02ED 1D7A8                MOV     MPH,#CSTMPE AND 0FH         ;MPE set
   438 02EE 1D7B1                MOV     MPL,#.DM.RDMSD SHR 4 AND 0FH ;Sets the row address of the
   439                                                               ;display data area.
   440 02EF 1D00F                MOV     RREG0,#.DM.RDMSD AND 0FH    ;Sets the column address.
   441                   JTD200:
   442 02F0 1A010                MOV     RREG1,@RREG0                ;Checks the space display
   443 02F1 0901A                SKE     RREG1,#CSPACE AND 0FH       ;data.
   444 02F2 0C2F5                BR      JTD300
   445 02F3 11001                SUB     RREG0,#1H
   446 02F4 0C2F0                BR      JTD200
   447
   448                           EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-016

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   449                  ;+++    Sign judgment of the display data    +++
   450                  JTD300:
   451 02F5 0901B               SKE     RREG1,#CMINUS AND 0FH         ;Is [-] displayed?
   452 02F6 0C2FB               BR      JTD400
   453 02F7 11001               SUB     RREG0,#1H
   454             1            BANK1
 +   1 02F8 1D791  1            MOV     BANK,#01H
   455 02F9 1D131               MOV     RYSIGN,#CSINMN AND 0FH        ;Yes
   456 02FA 0C2FD               BR      JTD500
   457                  JTD400:
   458             1            BANK1
 +   1 02FB 1D791  1            MOV     BANK,#01H
   459 02FC 1D130               MOV     RYSIGN,#CSINOFF AND 0FH       ;No
   460
   461                  ;+++    Data transfer                        +++
   462                  JTD500:
   463             1            BANK0
 +   1 02FD 1D790  1            MOV     BANK,#00H
   464 02FE 1D01E               MOV     RREG1,#.DM.(RYMSD-1H) AND 0FH
   465                  JTD600:
   466 02FF 1D7B1               MOV     MPL,#.DM.RDMSD SHR 4 AND 0FH  ;Transfers the display data to
   467 0300 1A020               MOV     RREG2,@RREG0                  ;floating-point register 2.
   468 0301 1D7B1               MOV     MPL,#.DM.RYMSD SHR 4 AND 0FH
   469 0302 167B8               OR      MPL,#CSTBK1
   470 0303 0A021               MOV     @RREG1,RREG2
   471 0304 11011               SUB     RREG1,#1H
   472 0305 11001               SUB     RREG0,#1H
   473 0306 1B008               SKLT    RREG0,#.DM.RDLSD AND 0FH      ;End of transfer?
   474 0307 0C2FF               BR      JTD600                        ;No -> Next digit
   475
   476             1            APOPMP                                ;Restores the memory pointer.
 +   1 0308 187AA  1            ST      MPH,RREG10
 +   2 0309 187BB  1            ST      MPL,RREG11
   477
   478                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-017

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   479                  ;+++  Exponent data transfer                +++
   480 030A 08140              LD      RREG0,RDEXP
   481 030B 08151              LD      RREG1,RDEXP+1H
   482             1           BANK1
 +   1 030C 1D791  1           MOV     BANK,#01H
   483 030D 18140              ST      RYEXP,RREG0
   484 030E 18151              ST      RYEXP+1H,RREG1
   485
   486                  ;+++  Zero suppression                      +++
   487             1           BANK0
 +   1 030F 1D790  1           MOV     BANK,#00H
   488 0310 1D038              MOV     RREG3,#8H
   489                  JTD700:
   490 0311 11031              SUB     RREG3,#1H
   491 0312 0B030              SKNE    RREG3,#0H             ;End of zero suppression?
   492 0313 0C31F              BR      JTD800                ;Yes
   493             1           BANK1
 +   1 0314 1D791  1           MOV     BANK,#01H
   494 0315 081E4              LD      RREG4,RYMSD-1H
   495             1           BANK0
 +   1 0316 1D790  1           MOV     BANK,#00H
   496 0317 09040              SKE     RREG4,#0H
   497 0318 0C31F              BR      JTD800                ;Yes
   498 0319 1C28E              CALL    SUSHFY                ;No:  Floating-point register 2
   499                                                       ;shift up
   500             1           BANK1
 +   1 031A 1D791  1           MOV     BANK,#01H
   501 031B 11141              SUB     RYEXP,#1H
   502 031C 13150              SUBC    RYEXP+1H,#0H
   503             1           BANK0
 +   1 031D 1D790  1           MOV     BANK,#00H
   504 031E 0C311              BR      JTD700
   505
   506                  ;+++  Resets the display data exponent area  +++
   507                  JTD800:
   508 031F 1D148              MOV     RDEXP,#CEXPINI AND 0FH
   509 0320 1D150              MOV     RDEXP+1H,#CEXPINI SHR 4 AND 0FH
   510 0321 1D100              MOV     RNUMC,#0H
   511
   512 0322 070E0              RET
   513
   514                         EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-018

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   515                  ;*************************************************
   516                  ;*                                               *
   517                  ;*   Display data output processing              *
   518                  ;*                                               *
   519                  ;*   INPUT  : RPTLOC (Decimal-point position area)  *
   520                  ;*          : RDLSD - RDMSD (Display data)       *
   521                  ;*   OUTPUT : LCDD0 - LCDD31                     *
   522                  ;*            (LCD segment data)                 *
   523                  ;*                                               *
   524                  ;*************************************************
   525                  ;*************************************************
   526                  ;*              Segment data table               *
   527                  ;*************************************************
   528 0323 3232        TSEGDAT:DW      3232H                   ;[0]
   529 0324 1010                DW      1010H                   ;[1]
   530 0325 1322                DW      1322H                   ;[2]
   531 0326 1132                DW      1132H                   ;[3]
   532 0327 3110                DW      3110H                   ;[4]
   533 0328 2132                DW      2132H                   ;[5]
   534 0329 2332                DW      2332H                   ;[6]
   535 032A 3012                DW      3012H                   ;[7]
   536 032B 3332                DW      3332H                   ;[8]
   537 032C 3132                DW      3132H                   ;[9]
   538 032D 0000                DW      0000H                   ;Space
   539 032E 0100                DW      0100H                   ;[-]
   540 032F 2322                DW      2322H                   ;[E]
   541
   542                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-019

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   543                  ;*********************************************
   544                  ;*           Initial value setting           *
   545                  ;*********************************************
   546                  SDISP:
   547             1            APUSHMP                        ;Saves the memory pointer.
 +   1 0330 087AA  1            LD      RREG10,MPH
 +   2 0331 087BB  1            LD      RREG11,MPL
   548
   549 0332 1D7A8               MOV     MPH,#CSTMPE AND 0FH            ;MPE set
   550 0333 1D008               MOV     RREG0,#.DM.RDLSD AND 0FH       ;Sets the column address of
   551                                                                 ;the display data area.
   552 0334 1D014               MOV     RREG1,#.DM.LCDD0 SHR 4 AND 0FH ;Sets the address of the LCD
   553 0335 1D020               MOV     RREG2,#.DM.LCDD0 AND 0FH       ;segment data area.
   554 0336 1D03F               MOV     RREG3,#0FH
   555 0337 01113               SUB     RREG3,RPTLOC
   556
   557                  ;*********************************************
   558                  ;*            Segment data reading           *
   559                  ;*********************************************
   560                  JDP200:
   561 0338 1D773               MOV     AR0,#.DL.TSEGDAT AND 0FH       ;Sets the first
   562 0339 1D762               MOV     AR1,#.DL.TSEGDAT SHR 4 AND 0FH ;address of the segment
   563 033A 1D753               MOV     AR2,#.DL.TSEGDAT SHR 8 AND 0FH ;data table.
   564 033B 1D740               MOV     AR3,#.DL.TSEGDAT SHR 12 AND 0FH
   565
   566 033C 1D7B1               MOV     MPL,#.DM.RDLSD SHR 4 AND 0FH   ;Sets the row address of
   567                                                                 ;the display data area.
   568 033D 1A040               MOV     RREG4,@RREG0                   ;Reads the display data.
   569                  JDP400:
   570 033E 0B040               SKNE    RREG4,#0H
   571 033F 0C343               BR      JDP600
   572 0340 07090               INC     AR                     ;Specifies the address of the segment
   573 0341 11041               SUB     RREG4,#1H              ;data to be referenced.
   574 0342 0C33E               BR      JDP400
   575
   576                  JDP600:
   577 0343 07010               MOVT    DBF,@AR                ;Reads the segment data.
   578
   579                          EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:25:27 12/20/93 PAGE 04-020

PROG =

SOURCE = CALC3.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   580                   ;*********************************************
   581                   ;*         Decimal-point position check        *
   582                   ;*********************************************
   583 0344 08004               LD      RREG4,RREG0
   584 0345 05034               XOR     RREG4,RREG3                     ;Decimal-point position?
   585 0346 0B040               SKNE    RREG4,#0H
   586 0347 160F1               OR      DBF0,#CDPON AND 0FH             ;Yes
   587
   588                   ;*********************************************
   589                   ;*              Segment data output            *
   590                   ;*********************************************
   591 0348 187B1               ST      MPL,RREG1
   592 0349 0A0F2               MOV     @RREG2,DBF0
   593 034A 10021               ADD     RREG2,#1H
   594 034B 0A0E2               MOV     @RREG2,DBF1
   595 034C 10021               ADD     RREG2,#1H
   596 034D 0A0D2               MOV     @RREG2,DBF2
   597 034E 10021               ADD     RREG2,#1H
   598 034F 0A0C2               MOV     @RREG2,DBF3
   599 0350 10021               ADD     RREG2,#1H
   600 0351 12010               ADDC    RREG1,#0H
   601                                                          ;Increments the column address
   602 0352 10001               ADD     RREG0,#1H               ;of the display data area.
   603 0353 09000               SKE     RREG0,#0H               ;End of data output?
   604 0354 0C338               BR      JDP200                  ;No -> Next digit
   605
   606             1            APOPMP                          ;Restores the memory pointer.
 +   1 0355 187AA  1            ST      MPH,RREG10
 +   2 0356 187BB  1            ST      MPL,RREG11
   607
   608 0357 070E0               RET
   609
   610                          END

TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

9.2 Floating-Point Section Program

This section shows the program listing of the floating-point section of the pocket calculator described in the application notes.

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-001

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
     1                 ;*************************************************************
     2                 ;*                                                           *
     3                 ;*                                                           *
     4                 ;*                                                           *
     5                 ;*       SYSTEM NAME : FLOATING POINT ARITHMETIC PACKAGE     *
     6                 ;*                                                           *
     7                 ;*                                                           *
     8                 ;*       CPU         : µPD17201A                             *
     9                 ;*                                                           *
    10                 ;*                                                           *
    11                 ;*       FILE NAME   : ARITH.ASM                             *
    12                 ;*                                                           *
    13                 ;*                                                           *
    14                 ;*************************************************************
    15
    16                 ;*************************************************************
    17                 ;*                            PUBLIC                         *
    18                 ;*************************************************************
    19
    20                 PUBLIC  SFPADD,SFPSUB,SFPMULT,SFPDIV
    21                 PUBLIC  SRXCLR
    22
    23                 ;*************************************************************
    24                 ;*                            EXTRN                          *
    25                 ;*************************************************************
    26                 ;****** RAM data ***********************************************
    27
    28                 EXTRN   MEM: RREG0,RREG1                ;General-purpose register
    29
    30                                                         ;Floating-point register 1
    31                 EXTRN   MEM: RXSIGN                     ;Sign
    32                 EXTRN   MEM: RXEXP                      ;Characteristic
    33                 EXTRN   MEM: RXLSD                      ;Least significant digit of mantissa
    34                 EXTRN   MEM: RXMSD                      ;Most significant digit of mantissa
    35                 EXTRN   MEM: RYSIGN,RYEXP,RYLSD,RYMSD   ;Floating-point register 2
    36                 EXTRN   MEM: RWSIGN,RWEXP,RWLSD,RWMSD   ;Floating-point register 3
    37
    38                 ;****** Flag **************************************************
    39
    40                 EXTRN   FLG: FEXCHG                     ;Register exchange flag
    41                 EXTRN   FLG: FZERO                      ;Operation result zero flag
    42                 EXTRN   FLG: FOVER                      ;Overflow flag
    43                 EXTRN   FLG: FDVERR                     ;Zero-division flag
    44                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-002

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    45                 ;****** Constant  ********************************************
    46
    47                 EXTRN   DAT: CGB0          ;General-purpose register BANK0
    48                 EXTRN   DAT: CGB1          ;General-purpose register BANK1
    49                 EXTRN   DAT: CROWG         ;General-purpose register row address 0H
    50                 EXTRN   DAT: CROWX         ;General-purpose register row address 0H
    51                 EXTRN   DAT: CROWY         ;General-purpose register row address 10
    52                 EXTRN   DAT: CROWW         ;General-purpose register row address 20
    53                 EXTRN   DAT: CROWXBCD      ;General-purpose register row address 0H
    54                                            ;Sets the BCD flag.
    55                 EXTRN   DAT: CIXMRY        ;Index modification (REGY)
    56                 EXTRN   DAT: CIXMRW        ;Index modification (REGW)
    57                 EXTRN   DAT: CIXMB1        ;Index modification (BANK1)
    58                 EXTRN   DAT: CIXLMANT      ;Index modification (mantissa, LSD)
    59                 EXTRN   DAT: CIXLEXP       ;Index modification (characteristic)
    60                 EXTRN   DAT: CIXLSIGN      ;Index modification (sign)
    61                 EXTRN   DAT: CMDRX         ;Column address
    62                                            ;to be indexed 0H
    63                 EXTRN   DAT: CMDRY         ;Column address
    64                                            ;to be indexed 10H
    65                 EXTRN   DAT: CMDRW         ;Column address
    66                                            ;to be indexed 20H
    67                 EXTRN   DAT: CJUDGE        ;Positive/negative judgment of the characteristic value
    68                 EXTRN   DAT: CCPTURN       ;Bit inversion mask
    69                 EXTRN   DAT: CSUBTURN      ;Subtrahend sign inversion
    70                 EXTRN   DAT: CRXTURN       ;REGX sign inversion
    71                 EXTRN   DAT: CEXDIF        ;Exponent difference judgment
    72                 EXTRN   DAT: CSIGNCK       ;Sign part check
    73                 EXTRN   DAT: CMANTCNT      ;Ten-digit counter for mantissa
    74                 EXTRN   DAT: CEXOVER       ;Characteristic value overflow
    75                 EXTRN   DAT: CEXUNDER      ;Characteristic value underflow
    76
    77                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-003

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
    78                 ;*************************************************************
    79                 ;*                                                           *
    80                 ;*         Floating-point arithmetic                         *
    81                 ;*                                                           *
    82                 ;*         (Addition, subtraction)                           *
    83                 ;*                                                           *
    84                 ;*         REGX   [I]  :Augend, minuend (normalized)         *
    85                 ;*         REGY   [I]  :Addend, subtrahend (normalized)      *
    86                 ;*         REGX   [O]  :Operation result (normalized)        *
    87                 ;*         IXM    [O]  :Index register                       *
    88                 ;*         FEXCHG [I/O]:Register exchange flag               *
    89                 ;*                                                           *
    90                 ;*         General-purpose registers used:  RREG0,RREG1      *
    91                 ;*                                                           *
    92                 ;*************************************************************
    93
    94                 SFPSUB:
    95                 ;*************************************************************
    96                 ;*                          Subtraction                      *
    97                 ;*************************************************************
    98             1           BANK1
 +   1 0358 1D791  1           MOV      BANK,#01H
    99 0359 1D7D1              MOV      RPH,#CGB1 AND 0FH            ;General-purpose register BANK1
   100 035A 1D7E4              MOV      RPL,#CROWW AND 0FH           ;Row address 20H
   101 035B 08133              LD       RWSIGN,RYSIGN                ;Stores the inverted sign
   102 035C 15231              XOR      RWSIGN,#CSUBTURN AND 0FH     ;of the subtrahend in RWSIGN.
   103                                                               ;Retains RYSIGN.
   104 035D 0C362              BR       JAD005
   105                 SFPADD:
   106                 ;*************************************************************
   107                 ;*                           Addition                        *
   108                 ;*************************************************************
   109             1           BANK1
 +   1 035E 1D791  1           MOV      BANK,#01H
   110 035F 1D7D1              MOV      RPH,#CGB1 AND 0FH            ;General-purpose register BANK1
   111 0360 1D7E4              MOV      RPL,#CROWW AND 0FH           ;Row address 20H
   112 0361 08133              LD       RWSIGN,RYSIGN                ;Retains RYSIGN.
   113                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-004

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
  114                    ;****** Smoothing characteristics ******************************
  115                    JAD005:
  116 0362 08044                LD      RWEXP,RXEXP
  117 0363 08055                LD      RWEXP+1H,RXEXP+1H
  118 0364 01144                SUB     RWEXP,RYEXP                 ;
  119 0365 03155                SUBC    RWEXP+1H,RYEXP+1H           ;RXEXP - RYEXP
  120 0366 1D7D0                MOV     RPH,#CGB0 AND 0FH           ;General-purpose register BANK0
  121 0367 1D7E0                MOV     RPL,#CROWG AND 0FH          ;Row address 0H
  122 0368 1E258                SKT     RWEXP+1H,#CJUDGE AND 0FH    ;RYEXP > RXEXP?
  123 0369 0C370                BR      JAD010
  124                                                               ;If RYEXP is greater
  125 036A 1524F                XOR     RWEXP,#CCPTURN AND 0FH      ;
  126 036B 1525F                XOR     RWEXP+1H,#CCPTURN AND 0FH   ;
  127 036C 10241                ADD     RWEXP,#1H                   ;
  128 036D 12250                ADDC    RWEXP+1H,#0H                ;Obtains the complement of the exponent difference.
  129 036E 1C467                CALL    SCHGXYEX                    ;Exchange REGX and REGY.
  130                                                               ;(characteristic, mantissa)
  131             1             SET1    FEXCHG                      ;Sets the register exchange flag.
+   1 036F 16301  1             OR      .MF.FEXCHG SHR 4,#.DF.FEXCHG AND 0FH
  132                    JAD010:
  133             1             SET1    CMP                         ;Sets the CMP flag.
+   1 0370 167F8  1             OR      .MF.CMP SHR 4,#.DF.CMP AND 0FH
  134 0371 1124A                SUB     RWEXP,#CEXDIF AND 0FH       ;
  135 0372 13250                SUBC    RWEXP+1H,#0H                ;Exponent difference ≧ 10?
  136             1             CLR1    CMP
+   1 0373 147F7  1             AND     .MF.CMP SHR 4,#.DF.(NOT CMP AND 0FH)
  137
  138                    ;****** Terminates with the greater value as the answer. ******
  139
  140             1             SKF1    CY
+   1 0374 1F7F4  1             SKF     .MF.CY SHR 4,#.DF.CY AND 0FH
  141 0375 0C38B                BR      JAD020                      ;Exponent difference ≦ 9
  142             1             SKT1    FEXCHG                      ;Register exchange flag = 0?
+   1 0376 1E301  1             SKT     .MF.FEXCHG SHR 4,#.DF.FEXCHG AND 0FH
  143 0377 070E0                RET                                 ;Terminates with REGX as the answer.
  144                    EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-005

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   145                  ;****** Transfer (REGX -> REGY)*******************************
   146
   147 0378 1D7D1              MOV     RPH,#CGB1 AND 0FH       ;General-purpose register BANK1
   148 0379 1D7E0              MOV     RPL,#CROWX AND 0FH      ;Row address 0H
   149 037A 18144              ST      RYEXP,RXEXP
   150 037B 18155              ST      RYEXP+1H,RXEXP+1H
   151 037C 18166              ST      RYLSD,RXLSD
   152 0C7D 18177              ST      RYLSD+1H,RXLSD+1H
   153 037E 18188              ST      RYLSD+2H,RXLSD+2H
   154 037F 18199              ST      RYLSD+3H,RXLSD+3H
   155 0380 181AA              ST      RYLSD+4H,RXLSD+4H
   156 0381 181BB              ST      RYLSD+5H,RXLSD+5H
   157 0382 181CC              ST      RYLSD+6H,RXLSD+6H
   158 0383 181DD              ST      RYLSD+7H,RXLSD+7H
   159 0384 181EE              ST      RYLSD+8H,RXLSD+8H
   160 0385 181FF              ST      RYMSD,RXMSD             ;Restores the previous value in REGY.
   161 0386 1D7D0              MOV     RPH,#CGB0 AND 0FH       ;General-purpose register BANK0
   162 0387 08230              LD      RREG0,RWSIGN            ;
   163 0388 18030              ST      RXSIGN,RREG0            ;Stores the result flag which was saved.
   164              1          CLR1    FEXCHG
 +   1 0389 1430E  1           AND     .MF.FEXCHG SHR 4,#.DF.(NOT FEXCHG AND 0FH)
   165 038A 070E0              RET
   166                  JAD020:
   167                  ;****** Mantissa adjustment **********************************
   168
   169              1          BANK1
 +   1 038B 1D791  1           MOV     BANK,#01H
   170 038C 11241              SUB     RWEXP,#1H               ;Decrements the exponent difference.
   171              1          SKT1    CY
     1 038D 1E7F4  1           SKT     .MF.CY SHR 4,#.DF.CY AND 0FH
   172 038E 0C398              BR      JAD030
   173 038F 08040              LD      RREG0,RXEXP             ;
   174 0390 08051              LD      RREG1,RXEXP+1H          ;
   175 0391 18140              ST      RYEXP,RREG0             ;
   176 0392 18151              ST      RYEXP+1H,RREG1          ;RYEXP <- RXEXP
   177              1          SKT1    FEXCHG                  ;Register exchange flag = 1?
 +   1 0393 1E301  1           SKT     .MF.FEXCHG SHR 4,#.DF.FEXCHG AND 0FH
   178 0394 0C39A              BR      JAD040
   179 0395 1C475              CALL    SCHGXY                  ;Exchanges REGX and REGY (mantissa).
   180              1          CLR1    FEXCHG
 +   1 0396 1430E  1           AND     .MF.FEXCHG SHR 4,#.DF.(NOT FEXCHG AND 0FH)
   181 0397 0C39A              BR      JAD040
   182                  JAD030:
   183 0398 1C4C1              CALL    SDSHFY                  ;Shifts REGY down by one digit.
   184 0399 0C38B              BR      JAD020
   185                  EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-006

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   186                  JAD040:
   187                  ;****** Sign check  ******************************************
   188
   189             1            BANK1
 +   1 039A 1D791  1            MOV     BANK,#01H
   190 039B 08230               LD      RREG0,RWSIGN
   191 039C 05030               XOR     RREG0,RXSIGN
   192             1            BANK0
 +   1 039D 1D790  1            MOV     BANK,#00H
   193 039E 1E001               SKT     RREG0,#CSIGNCK AND 0FH ;Skips if the signs are different.
   194 039F 0C3AB               BR      JAD999
   195
   196                  ;****** Mantissa calculation (if the signs are different)*******
   197
   198 03A0 1D7B1               MOV     IXM,#CIXMRY AND 0FH    ;Index modification (REGY)
   199 03A1 1C444               CALL    SSUB                   ;Mantissa subtraction (REGX - REGY)
   200             1            SKT1    CY                     ;Did subtraction cause a borrow?
 +   1 03A2 1E7FA  1            SKT     .MF.CY SHR 4,#.DF.CY AND 0FH
   201 03A3 0C3A9               BR      JAD050
   202 03A4 15031               XOR     RXSIGN,#CRXTURN AND 0FH;Sign inversion
   203 03A5 1C4EE               CALL    SRWCLR                 ;Clears REGW (0).
   204 03A6 1C483               CALL    SCHGXW                 ;Exchanges REGX and REGW (mantissa).
   205 03A7 1D7B2               MOV     IXM,#CIXMRW AND 0FH    ;Index modification (REGW)
   206 03A8 1C444               CALL    SSUB                   ;Mantissa subtraction (REGX - REGW)
   207                                                         ;Obtains the complement of the mantissa.
   208                  JAD050:
   209                  ;****** Normalization  ***************************************
   210
   211 03A9 1C3FC               CALL    SNML                   ;Normalization
   212 03AA 070E0               RET
   213                  JAD999:
   214                  ;****** Mantissa calculation (if the signs are the same)********
   215
   216 03AB 1D7B1               MOV     IXM,#CIXMRY AND 0FH    ;Index modification (REGY)
   217 03AC 1C42F               CALL    SADD                   ;Mantissa addition (REGX + REGY)
   218 03AD 0C3A9               BR      JAD050
   219                  EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-007

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   220                 ;*************************************************************
   221                 ;*                                                           *
   222                 ;*          Floating-point arithmetic                        *
   223                 ;*                                                           *
   224                 ;*          (Multiplication)                                 *
   225                 ;*                                                           *
   226                 ;*          RegX   [I] :Multiplicand (normalized)            *
   227                 ;*          RegY   [I] :Multiplier (normalized)              *
   228                 ;*          RegX   [O] :Operation result (normalized)        *
   229                 ;*          IXM    [O] :Index register                       *
   230                 ;*                                                           *
   231                 ;*          General-purpose registers used : RREG0           *
   232                 ;*                                                           *
   233                 ;*************************************************************
   234                 SFPMULT:
   235                 ;******  Obtaining the sign of the result *********************
   236
   237            1            BANK1
 +   1 03AE 1D791  1            MOV     BANK,#01H
   238 03AF 1D7D1               MOV     RPH,#CGB1 AND 0FH      ;General-purpose register BANK1
   239 03B0 1D7E0               MOV     RPL,#CROWX AND 0FH     ;Row address 0H
   240 03B1 05133               XOR     RXSIGN,RYSIGN          ;Obtains the sign of the result.
   241 03B2 18233               ST      RWSIGN,RXSIGN
   242
   243                 ;****** Exponent calculation **********************************
   244
   245 03B3 1D7D0               MOV     RPH,#CGB0 AND 0FH      ;General-purpose register BANK0
   246 03B4 1C491               CALL    SUSHFX                 ;Shifts REGX up by one digit
   247                                                         ;(provision for error).
   248 03B5 1004F               ADD     RXEXP,#0FH             ;
   249 03B6 1205F               ADDC    RXEXP+1H,#0FH          ;Exponent - 1
   250 03B7 1C459               CALL    SADDEX                 ;Exponent addition
   251
   252                 ;****** Generating an increasing loop counter  ****************
   253
   254 03B8 1C483               CALL    SCHGXW                 ;Exchanges REGX and REGW (mantissa).
   255 03B9 1D7D1               MOV     RPH,#CGB1 AND 0FH      ;General-purpose register BANK1
   256 03BA 18244               ST      RWEXP,RXEXP            ;
   257 03BB 18255               ST      RWEXP+1H,RXEXP+1H      ;Saves the exponent.
   258 03BC 1C4E3               CALL    SRXCLR                 ;Clears REGX (0).
   259 03BD 08244               LD      RXEXP,RWEXP            ;
   260 03BE 08255               LD      RXEXP+1H,RWEXP+1H      ;Restores the exponent.
   261 03BF 1D7D0               MOV     RPH,#CGB0 AND 0FH      ;General-purpose register BANK0
   262 03C0 1D24A               MOV     RWEXP,#CMANTCNT AND 0FH;Ten-digit counter for mantissa
   263                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-008

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   264                  JMT010:
   265                  ;****** Multiplication  ************************************
   266
   267            1             BANK1
 +   1 03C1 1D791  1            MOV      BANK,#01H
   268 03C2 11241               SUB      RWEXP,#1H
   269            1             SKT1     CY
 +   1 03C3 1E7F4  1            SKT      .MF.CY SHR 4,#.DF.CY AND 0FH
   270 03C4 0C3C9               BR       JMT020
   271 03C5 08230               LD       RREG0,RWSIGN            ;
   272 03C6 18030               ST       RXSIGN,RREG0            ;Restores the sign of the result.
   273 03C7 1C3FC               CALL     SNML                    ;Normalization
   274 03C8 070E0               RET
   275                  JMT020:
   276 03C9 1C4B1               CALL     SDSHFX                  ;Shifts REGX down by one digit.
   277 03CA 1C4D1               CALL     SDSHFW                  ;Shifts REGW down by one digit.
   278                  JMT030:
   279 03CB 11251               SUB      RWEXP+1H,#1H            ;RWEXP + 1 = Least significant digit of REGW
   280                                                           ;<Forward counter>
   281            1             SKT1     CY
 +   1 03CC 1E7F4  1            SKT      .MF.CY SHR 4,#.DF.CY AND 0FH
   282 03CD 0C3CF               BR       JMT999
   283 03CE 0C3C1               BR       JMT010
   284                  JMT999:
   285 03CF 1D7B1               MOV      IXM,#CIXMRY AND 0FH     ;Index modification (REGY)
   286 03D0 1C42F               CALL     SADD                    ;Mantissa addition (REGX + REGY)
   287 03D1 0C3CB               BR       JMT030
   288
   289                  EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-009

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   290                   ;*************************************************************
   291                   ;*                                                           *
   292                   ;*         Floating-point arithmetic                         *
   293                   ;*                                                           *
   294                   ;*         (Division)                                        *
   295                   ;*                                                           *
   296                   ;*         RegX     [I] :Dividend (normalized)               *
   297                   ;*         RegY     [I] :Divisor (normalized)                *
   298                   ;*         RegX     [O] :Operation result (normalized)       *
   299                   ;*         FDVERR   [O] :Zero-division flag                  *
   300                   ;*         IXM      [O] :Index register                      *
   301                   ;*                                                           *
   302                   ;*         General-purpose registers used: RREG0             *
   303                   ;*                                                           *
   304                   ;*************************************************************
   305                   SFPDIV:
   306              1            BANK1
 +   1 03D2 1D791   1            MOV      BANK,#01H
   307              1            CLR1     FDVERR                  ;Clears the zero-division flag.
 +   1 03D3 1430B   1            AND      .MF.FDVERR SHR 4,#.DF.(NOT FDVERR AND 0FH)
   308 03D4 1D7D1               MOV      RPH,#CGB1 AND 0FH       ;General-purpose register BANK1
   309 03D5 1D7E0               MOV      RPL,#CROWX AND 0FH      ;Row address 0H
   310 03D6 05133               XOR      RXSIGN,RYSIGN           ;Obtains the sign of the result.
   311 03D7 1D7D0               MOV      RPH,#CGB0 AND 0FH       ;General-purpose register BANK0
   312
   313                    ;***** Zero-division check  ***********************************
   314
   315 03D8 1D7A0               MOV      IXH,#0H                 ;
   316 03D9 1D7B8               MOV      IXM,#CIXMB1 AND 0FH     ;BANK1
   317 03DA 1D7C6               MOV      IXL,#CIXLMANT AND 0FH   ;Column address (mantissa)
   318                                                           ;IX <- 00010000110B
   319                   JDV010:
   320              1            BANK1
 +   1 03DB 1D791   1            MOV      BANK,#01H
   321              1            SET1     IXE
 +   1 03DC 167F1   1            OR       .MF.IXE SHR 4,#.DF.IXE AND 0FH
   322 03DD 08100               LD       RREG0,.MD.CMDRY         ;RREG0 <- REGY
   323              1            CLR1     IXE
 +   1 03DE 147FE   1            AND      .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   324              1            BANK0
 +   1 03DF 1D790   1            MOV      BANK,#00H
   325 03E0 09000               SKE      RREG0,#0H               ;Is the mantissa 0?
   326 03E1 0C3E8               BR       JDV020                  ;Not 0
   327 03E2 07080               INC      IX
   328 03E3 097C0               SKE      IXL,#0H                 ;Is the transfer up to 1FH completed?
   329 03E4 0C3DB               BR       JDV010
   330              1            BANK1
 +   1 0E35 1D791   1            MOV      BANK,#01H
   331              1            SET1     FDVERR                  ;Zero-division flag
 +   1 03E6 16304   1            OR       .MF.FDVERR SHR 4,#.DF.FDVERR AND 0FH
   332 03E7 070E0               RET
   333                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-010

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   334                   JDV020:
   335                   ;****** Exponent calculation   ********************************
   336
   337 03E8 1C460                CALL     SSUBEX                  ;Exponent subtraction
   338
   339                   ;****** Division   *******************************************
   340
   341 03E9 1C4EE                CALL     SRWCLR                       ;Clears REGW (0).
   342 03EA 1D23A                MOV      RWSIGN,#CMANTCNT AND 0FH     ;Ten-digit counter for mantissa
   343                   JDV030:
   344              1            BANK1
 +   1 03EB 1D791   1            MOV      BANK,#01H
   345 03EC 11231                SUB      RWSIGN,#1H
   346              1            SKT1     CY
 +   1 03ED 1E7F4   1            SKT      .MF.CY SHR 4,#.DF.CY AND 0FH
   347 03EE 0C3F2                BR       JDV040
   348 03EF 1C483                CALL     SCHGXW                  ;Exchanges REGX and REGW (mantissa).
   349 03F0 1C3FC                CALL     SNML                    ;Normalization
   350 03F1 070E0                RET
   351                   JDV040:
   352 03F2 1C4A1                CALL     SUSHFW                  ;Shifts REGW up by one digit.
   353                   JDV050:
   354 03F3 1D7B1                MOV      IXM,#CIXMRY AND 0FH     ;Index modification (REGY)
   355 03F4 1C444                CALL     SSUB                    ;Mantissa subtraction (REGX - REGY)
   356              1            SKT1     CY
 +   1 03F5 1E7F4   1            SKT      .MF.CY SHR 4,#.DF.CY AND 0FH
   357 03F6 0C3FA                BR       JDV999
   358 03F7 1C42F                CALL     SADD                    ;Mantissa addition (REGX + REGY)
   359 03F8 1C491                CALL     SUSHFX                  ;Shifts REGX up by one digit.
   360 03F9 0C3EB                BR       JDV030
   361                   JDV999:
   362 03FA 10261                ADD      RWLSD,#1H               ;Quotient + 1
   363 03FB 0C3F3                BR       JDV050
   364                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-011

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   365                 ;*************************************************************
   366                 ;*                                                           *
   367                 ;*   Normalization                                           *
   368                 ;*                                                           *
   369                 ;*                                                           *
   370                 ;*   RegX  [I]:Operation data before normalization (result)  *
   371                 ;*   RegX  [O]:Normalized operation data (result)            *
   372                 ;*   FZERO [O]:Operation result zero flag                    *
   373                 ;*   FOVER [O]:Overflow flag                                 *
   374                 ;*             Overflow   :   63 < Exponent value            *
   375                 ;*             Underflow  :  -64 > Exponent value            *
   376                 ;*                                                           *
   377                 ;*   General-purpose registers used:  RREG0                  *
   378                 ;*                                                           *
   379                 ;*************************************************************
   380
   381                 SNML:
   382            1            BANK1
 +   1 03FC 1D791  1            MOV      BANK,#01H
   383            1            CLR1     FZERO                  ;Clears the operation result zero flag.
 +   1 03FD 1430D  1            AND      .MF.FZERO SHR 4,#.DF.(NOT FZERO AND 0FH)
   384            1            CLR1     FOVER                  ;Clears the overflow flag.
     1 03FE 14307  1            AND      .MF.FOVER SHR 4,#.DF.(NOT FOVER AND 0FH)
   385            1            BANK0
 +   1 03FF 1D790  1            MOV      BANK.#00H
   386 0400 1D7D0               MOV      RPH,#CGB0 AND 0FH      ;General-purpose register BANK0
   387 0401 1D7E0               MOV      RPL,#CROWG AND 0FH     ;Row address 0H
   388
   389                 ;****** Zero-mantissa check  ***********************************
   390
   391 0402 1D7A0               MOV      IXH,#0H                ;
   392 0403 1D7B8               MOV      IXM,#CIXMB1 AND 0FH    ;BANK1
   393 0404 1D7C6               MOV      IXL,#CIXLMANT AND 0FH  ;Column address (mantissa)
   394                                                          ;IX <- 00010000110B
   395                 JNM010:
   396            1            BANK1
 +   1 0405 1D791  1            MOV      BANK,#01H
   397            1            SET1     IXE
 +   1 0406 167F1  1            OR       .MF.IXE SHR 4,#.DF.IXE AND 0FH
   398 0407 08000               LD       RREG0,.MD.CMDRX        ;RREG0 <- REGX
   399            1            CLR1     IXE
 +   1 0408 147FE  1            AND      .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   400            1            BANK0
     1 0409 1D790  1            MOV      BANK,#00H
   401 040A 09000               SKE      RREG0,#0H              ;Is the mantissa 0?
   402 040B 0C410               BR       JNM020                 ;Not 0
   403 040C 07080               INC      IX
   404 040D 097C0               SKE      IXL,#0H                ;Is the transfer up to 0FH completed?
   405 040E 0C405               BR       JNM010
   406 040F 0C42C               BR       JNM999                 ;Mantissa is all 0.
   407                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-012

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   408                   JNM020:
   409                   ;****** Normalization  ****************************************
   410
   411              1            BANK1
 +   1 0410 1D791   1            MOV     BANK,#01H
   412 0411 090F0               SKE     RXMSD,#0H          ;Is the integer part 0?
   413 0412 0C419               BR      JNM040             ;Not 0
   414                   JNM030:
   415 0413 090E0               SKE     RXMSD-1H,#0H       ;Is the most significant digit of the fractional part 0?
   416 0414 0C41C               BR      JNM050             ;Not 0
   417 0415 1C491               CALL    SUSHFX             ;Shifts REGX up by one digit.
   418 0416 1004F               ADD     RXEXP,#0FH         ;
   419 0417 1205F               ADDC    RXEXP+1H,#0FH      ;Exponent - 1
   420 0418 0C413               BR      JNM030
   421                   JNM040:
   422 0419 1C4B1               CALL    SDSHFX             ;Shifts REGX down by one digit.
   423 041A 10041               ADD     RXEXP,#1H          ;
   424 041B 12050               ADDC    RXEXP+1H,#0H       ;Exponent + 1
   425                   JNM050:
   426 041C 1F058               SKF     RXEXP+1H,#CJUDGE AND 0FH          ;Is the exponent positive?
   427 041D 0C425               BR      JNM060
   428
   429                   ;****** Overflow judgment  ************************************
   430              1            SET1    CMP                                ;Sets the CMP flag.
 +   1 041E 167F8   1            OR      .MF.CMP SHR 4,#.DF.CMP AND 0FH
   431 041F 11054               SUB     RXEXP+1H,#CEXOVER AND 0FH          ;Exponent value < 64?
   432              1            CLR1    CMP
 +   1 0420 147F7   1            AND     .MF.CMP SHR 4,#.DF.(NOT CMP AND 0FH)
   433              1            SKF1    CY
 +   1 0421 1F7F4   1            SKF     .MF.CY SHR 4,#.DF.CY AND 0FH
   434 0422 070E0               RET
   435              1            SET1    FOVER              ;Sets the overflow flag.
 +   1 0423 16308   1            OR      .MF.FOVER SHR 4,#.DF.FOVER AND 0FH
   436 0424 070E0               RET
   437                   JNM060:
   438                   ;****** Underflow judgment ************************************
   439              1            SET1    CMP
 +   1 0425 167F8   1            OR      .MF.CMP SHR 4,#.DF.CMP AND 0FH
   440 0426 1105C               SUB     RXEXP+1H,#CEXUNDER AND 0FH         ;Exponent value > -65?
   441              1            CLR1    CMP
 +   1 0427 147F7   1            AND     .MF.CMP SHR 4,#.DF.(NOT CMP AND 0FH)
   442              1            SKT1    CY
 +   1 0428 1E7F4   1            SKT     .MF.CY SHR 4,#.DF.CY AND 0FH
   443 0429 070E0               RET
   444              1            SET1    FOVER              ;Sets the overflow flag.
 +   1 042A 16308   1            OR      .MF.FOVER SHR 4,#.DF.FOVER AND 0FH
   445 042B 070E0               RET
   446                   JNM999:
   447                   ;****** Clear (0)  ****************************************
   448 042C 1C4E3               CALL    SRXCLR             ;Clears REGX (0).
   449              1            SET1    FZERO
 +   1 042D 16302   1            OR      .MF.FZERO SHR 4,#.DF.FZERO AND 0FH
   450 042E 070E0               RET
   451                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-013

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   452                   ;*************************************************************
   453                   ;*                                                           *
   454                   ;*        Mantissa addition                                  *
   455                   ;*                                                           *
   456                   ;*        RegX    [I]:Operation data                         *
   457                   ;*        IXM     [I]:Index register                         *
   458                   ;*        RegX    [O]:Result of addition of REGX and the     *
   459                   ;*                    register to be indexed                 *
   460                   ;*************************************************************
   461
   462
   463                   SADD:
   464             1             BANK1
 +   1 042F 1D791  1             MOV      BANK,#01H
   465 0430 1D7A0                MOV      IXH,#0H
   466 0431 1D7C0                MOV      IXL,#0H
   467 0432 1D7D1                MOV      RPH,#CGB1 AND 0FH       ;General-purpose register BANK1
   468 0433 1D7E1                MOV      RPL,#CROWXBCD AND 0FH   ;Row address 0H
   469                                                            ;Sets the BCD flag.
   470             1             SET1     IXE
 +   1 0434 167F1  1             OR       .MF.IXE SHR 4,#.DF.IXE AND 0FH
   471             1             CLR1     CY
 +   1 0435 147FB  1             AND      .MF.CY SHR 4,#.DF.(NOT CY AND 0FH)
   472 0436 00066                ADD      RXLSD,RXLSD
   473 0437 02077                ADDC     RXLSD+1H,RXLSD+1H
   474 0438 02088                ADDC     RXLSD+2H,RXLSD+2H
   475 0439 02099                ADDC     RXLSD+3H,RXLSD+3H
   476 043A 020AA                ADDC     RXLSD+4H,RXLSD+4H
   477 043B 020BB         *      ADDC     RXLSD+5H,RXLSD+5H
   478 043C 020CC                ADDC     RXLSD+6H,RXLSD+6H
   479 043D 020DD                ADDC     RXLSD+7H,RXLSD+7H
   480 043E 020EE                ADDC     RXLSD+8H,RXLSD+8H
   481 043F 020FF                ADDC     RXLSD+9H,RXLSD+9H
   482             1             CLR1     IXE
 +   1 0440 147FE  1             AND      .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   483 0441 1D7D0                MOV      RPH,#CGB0 AND 0FH       ;General-purpose register BANK0
   484 0442 1D7E0                MOV      RPL,#CROWG AND 0FH      ;Row address 0H
   485                                                            ;Clears the BCD flag.
   486 0443 070E0                RET
   487                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-014

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   488                   ;*************************************************************
   489                   ;*                                                           *
   490                   ;*       Mantissa subtraction                                *
   491                   ;*                                                           *
   492                   ;*       RegX  [I]:Operation data                            *
   493                   ;*       IXM   [I]:Index register                            *
   494                   ;*       RegX  [O]:Result of subtraction of REGX and the     *
   495                   ;*                 register to be indexed                    *
   496                   ;*                                                           *
   497                    *************************************************************
   498
   499                   SSUB:
   500              1            BANK1
 +   1 0444 1D791  1            MOV      BANK,#01H
   501 0445 1D7A0               MOV      IXH,#0H
   502 0446 1D7C0               MOV      IXL,#0H
   503 0447 1D7D1               MOV      RPH,#CGB1 AND 0FH       ;General-purpose register BANK1
   504 0448 1D7E1               MOV      RPL,#CROWXBCD AND 0FH   ;Row address 0H
   505                                                           ;Sets the BCD flag.
   506              1            SET1     IXE
 +   1 0449 167F1  1            OR       .MF.IXE SHR 4,#.DF.IXE AND 0FH
   507              1            CLR1     CY
 +   1 044A 147FB  1            AND      .MF.CY SHR 4,#.DF.(NOT CY AND 0FH)
   508 044B 01066               SUB      RXLSD,RXLSD
   509 044C 03077               SUBC     RXLSD+1H,RXLSD+1H
   510 044D 03088               SUBC     RXLSD+2H,RXLSD+2H
   511 044E 03099               SUBC     RXLSD+3H,RXLSD+3H
   512 044F 030AA               SUBC     RXLSD+4H,RXLSD+4H
   513 0450 030BB               SUBC     RXLSD+5H,RXLSD+5H
   514 0451 030CC               SUBC     RXLSD+6H,RXLSD+6H
   515 0452 030DD               SUBC     RXLSD+7H,RXLSD+7H
   516 0453 030EE               SUBC     RXLSD+8H,RXLSD+8H
   517 0454 030FF               SUBC     RXLSD+9H,RXLSD+9H
   518              1            CLR1     IXE
 +   1 0455 147FE  1            AND      .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   519 0456 1D7D0               MOV      RPH,#CGB0 AND 0FH       ;General-purpose register BANK0
   520 0457 1D7E0               MOV      RPL,#CROWG AND 0FH      ;Row address 0H
   521                                                           ;Clears the BCD flag.
   522 0458 070E0               RET
   523                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-015

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   524                   ;*************************************************************
   525                   ;*                                                           *
   526                   ;*         Characteristic operation (addition)               *
   527                   ;*                                                           *
   528                   ;*         RegX   [I]:Augend                                 *
   529                   ;*         RwgY   [I]:Addend                                 *
   530                   ;*         RegX   [O]:Characteristic operation result        *
   531                   ;*                                                           *
   532                   ;*************************************************************
   533                   SADDEX:
   534              1            BANK1
 +   1 0459 1D791  1            MOV      BANK,#01H
   535 045A 1D7D1               MOV      RPH,#CGB1 AND 0FH      ;General-purpose register BANK1
   536 045B 1D7E0               MOV      RPL,#CROWX AND 0FH     ;Row address 0H
   537 045C 00144               ADD      RXEXP,RYEXP
   538 045D 02155               ADDC     RXEXP+1H,RYEXP+1H
   539 045E 1D7D0               MOV      RPH,#CGB0 AND 0FH      ;General-purpose register BANK0
   540 045F 070E0               RET
   541                   ;*************************************************************
   542                   ;*                                                           *
   543                   ;*         Characteristic operation (subtraction)            *
   544                   ;*                                                           *
   545                   ;*         RegX    [I]:Minuend                               *
   546                   ;*         RwgY    [I]:Subtrahend                            *
   547                   ;*         RegX    [O]:Characteristic operation result       *
   548                   ;*                                                           *
   549                   ;*************************************************************
   550                   SSUBEX:
   551              1            BANK1
 +   1 0460 1D791  1            MOV      BANK,#01H
   552 0461 1D7D1               MOV      RPH,#CGB1 AND 0FH      ;General-purpose register BANK1
   553 0462 1D7E0               MOV      RPL,#CROWX AND 0FH     ;Row address 0H
   554 0463 01144               SUB      RXEXP,RYEXP            ;
   555 0464 03155               SUBC     RXEXP+1H,RYEXP+1H      ;RXEXP - RYEXP
   556 0465 1D7D0               MOV      RPH,#CGB0 AND 0FH      ;General-purpose register BANK0
   557 0466 070E0               RET
   558                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-016

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   559                 ;*************************************************************
   560                 ;*                                                           *
   561                 ;*      Register exchange 1  (REGX <--> REGY)                *
   562                 ;*                              (characteristic, mantissa)   *
   563                 ;*      RegX    [I]:Operation data                           *
   564                 ;*      RegY    [I]:Operation data                           *
   565                 ;*      RegX    [O]:REGY                                     *
   566                 ;*      RegY    [O]:REGX                                     *
   567                 ;*                                                           *
   568                 ;*                                                           *
   569                 ;*      General-purpose registers used:  RREG0,RREG1         *
   570                 ;*                                                           *
   571                 ;*************************************************************
   572
   573                 SCHGXYEX:
   574             1           BANK1
 +   1 0467 1D791  1           MOV     BANK,#01H
   575 0468 1D7A0              MOV     IXH,#0H                 ;
   576 0469 1D7B8              MOV     IXM,#CIXMB1 AND 0FH     ;BANK1
   577 046A 1D7C4              MOV     IXL,#CIXLEXP AND 0FH    ;Column address (characteristic)
   578                                                         ;IX <- 00010000100B
   579                 JEXY999:
   580             1           SET1    IXE
 +   1 046B 167F1  1           OR      .MF.IXE SHR 4,#.DF.IXE AND 0FH
   581 046C 08000              LD      RREG0,.MD.CMDRX         ;RREG0 <- REGX
   582 046D 08101              LD      RREG1,.MD.CMDRY         ;RREG1 <- REGY
   583 046E 18001              ST      .MD.CMDRX,RREG1         ;REGX <- REGY
   584 046F 18100              ST      .MD.CMDRY,RREG0         ;REGY <- REGX
   585             1           CLR1    IXE
 +   1 0470 147FE  1           AND     .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   586 0471 07080              INC     IX
   587 0472 097C0              SKE     IXL,#0H                 ;Is the transfer up to *FH completed?
   588 0473 0C46B              BR      JEXY999
   589 0474 070E0              RET
   590
   591                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-017

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   592                   ;*************************************************************
   593                   ;*                                                           *
   594                   ;*      Register exchange 2  (REGX <--> REGY)                *
   595                   ;*                           (mantissa)                      *
   596                   ;*      RegX  [I]:Operation data                             *
   597                   ;*      RegY  [I]:Operation data                             *
   598                   ;*      RegX  [O]:REGY                                       *
   599                   ;*      RegY  [O]:REGX                                       *
   600                   ;*                                                           *
   601                   ;*                                                           *
   602                   ;*      General-purpose registers used:  RREG0,RREG1         *
   603                   ;*                                                           *
   604                   ;*************************************************************
   605
   606                   SCHGXY:
   607              1            BANK1
 +   1 0475 1D791  1            MOV     BANK,#01H
   608 0476 1D7A0               MOV     IXH,#0H                 ;
   609 0477 1D7B8               MOV     IXM,#CIXMB1 AND 0FH     ;BANK1
   610 0478 1D7C6               MOV     IXL,#CIXLMANT AND 0FH   ;Column address (mantissa)
   611                                                          ;IX <- 00010000110B
   612                   JXY999:
   613              1            SET1    IXE
 +   1 0479 167F1  1            OR      .MF.IXE SHR 4,#.DF.IXE AND 0FH
   614 047A 08000               LD      RREG0,.MD.CMDRX         ;RREG0 <- REGX
   615 047B 08101               LD      RREG1,.MD.CMDRY         ;RREG1 <- REGY
   616 047C 18001               ST      .MD.CMDRX,RREG1         ;REGX <- REGY
   617 047D 18100               ST      .MD.CMDRY,RREG0         ;REGY <- REGX
   618              1            CLR1    IXE
 +   1 047E 147FE  1            AND     .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   619 047F 07080               INC     IX
   620 0480 097C0               SKE     IXL,#0H                 ;Is the transfer up to *FH completed?
   621 0481 0C479               BR      JXY999
   622 0482 070E0               RET
   623                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-018

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   624                  ;*************************************************************
   625                  ;*                                                           *
   626                  ;*                                                           *
   627                  ;*      Register exchange 3 (REGX <--> REGW)                 *
   628                  ;*                          (mantissa)                       *
   629                  ;*      RegX  [I]: Operation data                            *
   630                  ;*      RegW  [I]: Operation work data                       *
   631                  ;*      RegX  [O]: REGW                                      *
   632                  ;*      RegW  [O]: REGX                                      *
   633                  ;*                                                           *
   634                  ;*      General-purpose registers used:  RREG0,RREG1         *
   635                  ;*                                                           *
   636                  ;*                                                           *
   637                   *************************************************************
   638
   639                  SCHGXW:
   640              1           BANK1
 +   1 0483 1D791       1       MOV     BANK,#01H
   641 0484 1D7A0               MOV     IXH,#0H                 ;
   642 0485 1D7B8               MOV     IXM,#CIXMB1 AND 0FH     ;BANK1
   643 0486 1D7C6               MOV     IXL,#CIXLMANT AND 0FH   ;Column address (mantissa)
   644                                                          ;IX <- 00010000110B
   645                  JXW010:
   646              1           SET1    IXE
 +   1 0487 167F1       1       OR      .MF.IXE SHR 4,#.DF.IXE AND 0FH
   647 0488 08000               LD      RREG0,.MD.CMDRX         ;RREG0 <- REGX
   648 0489 08201               LD      RREG1,.MD.CMDRW         ;RREG1 <- REGW
   649 048A 18001               ST      .MD.CMDRX,RREG1         ;REGX <- REGW
   650 048B 18200               ST      .MD.CMDRW,RREG0         ;REGW <- REGX
   651              1           CLR1    IXE
 +   1 048C 147FE       1       AND     .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   652 048D 07080               INC     IX
   653 048E 097C0               SKE     IXL,#0H                 ;Is the transfer up to *FH completed?
   654 048F 0C487               BR      JXW010
   655 0490 070E0               RET
   656                  EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-019

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   657                 ;*********************************************
   658                 ;*                                           *
   659                 ;*      Shift up register 1 (REGX)           *
   660                 ;*                                           *
   661                 ;*      RegX    [I]:  Operation data (result) *
   662                 ;*      RegX    [O]:  Shift up completed     *
   663                 ;*                                           *
   664                 ;*********************************************
   665
   666                 SUSHFX:
   667            1            BANK1
 +   1 0491 1D791  1            MOV     BANK,#01H
   668 0492 1D7D1               MOV     RPH,#CGB1 AND 0FH   ;General-purpose register BANK1
   669 0493 1D7E0               MOV     RPL,#CROWX AND 0FH  ;Row address 0H
   670 0494 080EF               LD      RXMSD,RXMSD-1H      ;
   671 0495 080DE               LD      RXMSD-1H,RXMSD-2H   ;
   672 0496 080CD               LD      RXMSD-2H,RXMSD-3H   ;
   673 0497 080BC               LD      RXMSD-3H,RXMSD-4H   ;
   674 0498 080AB               LD      RXMSD-4H,RXMSD-5H   ;
   675 0499 0809A               LD      RXMSD-5H,RXMSD-6H   ;
   676 049A 08089               LD      RXMSD-6H,RXMSD-7H   ;
   677 049B 08078               LD      RXMSD-7H,RXMSD-8H   ;
   678 049C 08067               LD      RXMSD-8H,RXMSD-9H   ;Shift up
   679 049D 1D060               MOV     RXLSD,#0H           ;LSD <- 0
   680 049E 1D7D0               MOV     RPH,#CGB0 AND 0FH   ;General-purpose register BANK0
   681 049F 1D7E0               MOV     RPL,#CROWG AND 0FH  ;Row address 0H
   682 04A0 070E0               RET
   683                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-020

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   684                   ;*********************************************
   685                   ;*                                           *
   686                   ;*     Shift up register 2 (REGW)            *
   687                   ;*                                           *
   688                   ;*     RegW     [I]:  Operation data (result)  *
   689                   ;*     RegW     [O]:  Shift up completed       *
   690                   ;*                                           *
   691                   ;*********************************************
   692
   693                   SUSHFW:
   694              1            BANK1
 +   1 04A1 1D791  1            MOV      BANK,#01H
   695 04A2 1D7D1               MOV      RPH,#CGB1 AND 0FH    ;General-purpose register BANK1
   696 04A3 1D7E4               MOV      RPL,#CROWW AND 0FH   ;Row address 2H
   697 04A4 082EF               LD       RWMSD,RWMSD-1H       ;
   698 04A5 082DE               LD       RWMSD-1H,RWMSD-2H    ;
   699 04A6 082CD               LD       RWMSD-2H,RWMSD-3H    ;
   700 04A7 082BC               LD       RWMSD-3H,RWMSD-4H    ;
   701 04A8 082AB               LD       RWMSD-4H,RWMSD-5H    ;
   702 04A9 0829A               LD       RWMSD-5H,RWMSD-6H    ;
   703 04AA 08289               LD       RWMSD-6H,RWMSD-7H    ;
   704 04AB 08278               LD       RWMSD-7H,RWMSD-8H    ;
   705 04AC 08267               LD       RWMSD-8H,RWMSD-9H    ;Shift up
   706 04AD 1D260               MOV      RWLSD,#0H            ;LSD <- 0
   707 04AE 1D7D0               MOV      RPH,#CGB0 AND 0FH    ;General-purpose register BANK0
   708 04AF 1D7E0               MOV      RPL,#CROWG AND 0FH   ;Row address 0H
   709 04B0 070E0               RET
   710                   EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-021

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.    M  I SOURCE STATEMENT
   711                   ;*********************************************
   712                   ;*                                           *
   713                   ;*      Shift down register 1 (REGX)         *
   714                   ;*                                           *
   715                   ;*      RegX    [I]:  Operation data         *
   716                   ;*      RegX    [O]:  Shift down completed   *
   717                   ;*                                           *
   718                   ;*********************************************
   719
   720                   SDSHFX:
   721              1            BANK1
 +   1 04B1 1D791  1            MOV     BANK,#01H
   722 04B2 1D7D1               MOV     RPH,#CGB1 AND 0FH   ;General-purpose register BANK1
   723 04B3 1D7E0               MOV     RPL,#CROWX AND 0FH  ;Row address 0H
   724 04B4 08076               LD      RXLSD,RXLSD+1H      ;
   725 04B5 08087               LD      RXLSD+1H,RXLSD+2H   ;
   726 04B6 08098               LD      RXLSD+2H,RXLSD+3H   ;
   727 04B7 080A9               LD      RXLSD+3H,RXLSD+4H   ;
   728 04B8 080BA               LD      RXLSD+4H,RXLSD+5H   ;
   729 04B9 080CB               LD      RXLSD+5H,RXLSD+6H   ;
   730 04BA 080DC               LD      RXLSD+6H,RXLSD+7H   ;
   731 04BB 080ED               LD      RXLSD+7H,RXLSD+8H   ;
   732 04BC 080FE               LD      RXLSD+8H,RXLSD+9H   ;Shift down
   733 04BD 1D0F0               MOV     RXMSD,#0H           ;MSD <- 0
   734 04BE 1D7D0               MOV     RPH,#CGB0 AND 0FH   ;General-purpose register BANK0
   735 04BF 1D7E0               MOV     RPL,#CROWG AND 0FH  ;Row address 0H
   736 04C0 070E0               RET
   737                   EJECT
```

```
AS17K V1.10 V4  << D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-022

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   738                 ;*********************************************
   739                 ;*                                           *
   740                 ;*      Shift down register 2 (REGY)         *
   741                 ;*                                           *
   742                 ;*      RegY    [I]:  Operation data         *
   743                 ;*      RegY    [O]:  Shift down completed   *
   744                 ;*                                           *
   745                 ;*********************************************
   746
   747                 SDSHFY:
   748            1            BANK1
 +   1 04C1 1D791  1            MOV     BANK,#01H
   749 04C2 1D7D1               MOV     RPH,#CGB1 AND 0FH   ;General-purpose register BANK1
   750 04C3 1D7E2               MOV     RPL,#CROWY AND 0FH  ;Row address 1H
   751 04C4 08176               LD      RYLSD,RYLSD+1H      ;
   752 04C5 08187               LD      RYLSD+1H,RYLSD+2H   ;
   753 04C6 08198               LD      RYLSD+2H,RYLSD+3H   ;
   754 04C7 081A9               LD      RYLSD+3H,RYLSD+4H   ;
   755 04C8 081BA               LD      RYLSD+4H,RYLSD+5H   ;
   756 04C9 081CB               LD      RYLSD+5H,RYLSD+6H   ;
   757 04CA 081DC               LD      RYLSD+6H,RYLSD+7H   ;
   758 04CB 081ED               LD      RYLSD+7H,RYLSD+8H   ;
   759 04CC 081FE               LD      RYLSD+8H,RYLSD+9H   ;Shift down
   760 04CD 1D1F0               MOV     RYLSD,#0H           ;MSD <- 0
   761 04CE 1D7D0               MOV     RPH,#CGB0 AND 0FH   ;General-purpose register BANK0
   762 04CF 1D7E0               MOV     RPL,#CROWG AND 0FH  ;Row address 0H
   763 04D0 070E0               RET
   764                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-023

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   765                 ;*********************************************
   766                 ;*                                           *
   767                 ;*   Shift down register 3 (REGW)            *
   768                 ;*                                           *
   769                 ;*   RegW     [I]:  Operation data work area  *
   770                 ;*   RegW     [O]:  Shift down completed      *
   771                 ;*   RWEXP+1H[O]:  Least significant digit    *
   772                 ;*                 obtained by shift down     *
   773                 ;*   General-purpose registers used:  RREG0  *
   774                 ;*                                           *
   775                 ;*********************************************
   776
   777                 SDSHFW:
   778             1           BANK1
 +   1 04D1 1D791  1           MOV     BANK,#01H          ;
   779 04D2 08260              LD      RREG0,RWLSD        ;Saves the least significant digit.
   780 04D3 18250              ST      RWEXP+1H,RREG0     ;General-purpose register BANK1
   781 04D4 1D7D1              MOV     RPH,#CGB1 AND 0FH  ;Row address 2H
   782 04D5 1D7E4              MOV     RPL,#CROWW AND 0FH ;
   783 04D6 08276              LD      RWLSD,RWLSD+1H     ;
   784 04D7 08287              LD      RWLSD+1H,RWLSD+2H  ;
   785 04D8 08298              LD      RWLSD+2H,RWLSD+3H  ;
   786 04D9 082A9              LD      RWLSD+3H,RWLSD+4H  ;
   787 04DA 082BA              LD      RWLSD+4H,RWLSD+5H  ;
   788 04DB 082CB              LD      RWLSD+5H,RWLSD+6H  ;
   789 04DC 082DC              LD      RWLSD+6H,RWLSD+7H  ;
   790 04DD 082ED              LD      RWLSD+7H,RWLSD+8H  ;
   791 04DE 082FE              LD      RWLSD+8H,RWLSD+9H  ;Shift down
   792 04DF 1D2F0              MOV     RWMSD,#0H          ;MSD <- 0
   793 04E0 1D7D0              MOV     RPH,#CGB0 AND 0FH  ;General-purpose register BANK0
   794 04E1 1D7E0              MOV     RPL,#CROWG AND 0FH ;Row address 0H
   795 04E2 070E0              RET
   796                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-024

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   797                 ;*************************************************************
   798                 ;*                                                           *
   799                 ;*      Register clear (0)(REGX)                             *
   800                 ;*                    (sign, characteristic, mantissa)       *
   801                 ;*      RegX   [I]:  Operation data (result)                 *
   802                 ;*      RwgX   [O]:  Clear (0) completed                     *
   803                 ;*                                                           *
   804                 ;*                                                           *
   805                 ;*************************************************************
   806
   807                 SRXCLR:
   808            1            BANK1
 +   1 04E3 1D791  1           MOV     BANK,#01H                ;
   809 04E4 1D7A0               MOV     IXH,#0H                  ;BANK1
   810 04E5 1D7B8               MOV     IXM,#CIXMB1 AND 0FH      ;Column address (sign)
   811 04E6 1D7C3               MOV     IXL,#CIXLSIGN AND 0FH    ;IX <- 000100000110B
   812
   813                 JXC999:
   814            1            SET1    IXE
 +   1 04E7 167F1  1           OR      .MF.IXE SHR 4,#.DF.IXE AND 0FH
   815 04E8 1D000               MOV     .MD.CMDRX,#0H            ;Clear (0)
   816            1            CLR1    IXE
 +   1 04E9 147FE  1           AND     .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   817 04EA 07080               INC     IX                       ;Is the transfer up to 0FH completed?
   818 04EB 097C0               SKE     IXL,#0H
   819 04EC 0C4E7               BR      JXC999
   820 04ED 070E0               RET
   821                 EJECT
```

```
AS17K V1.10 V4 <<  D17201A ASSEMBLE LIST >>  11:26:25 12/20/93 PAGE 05-025

PROG =

SOURCE = ARITH.ASM

E STNO LOC. OBJ.   M  I SOURCE STATEMENT
   822                  ;*************************************************************
   823                  ;*                                                           *
   824                  ;*      Register clear (0) (REGW)                            *
   825                  ;*                    (mantissa)                             *
   826                  ;*                                                           *
   827                  ;*      RegW  [I]:  Operation work area                      *
   828                  ;*      RegW  [O]:  Clear (0) completed                      *
   829                  ;*                                                           *
   830                  ;*************************************************************
   831
   832                  SRWCLR:
   833             1            BANK1
 +   1 04EE 1D791  1            MOV     BANK,#01H
   834 04EF 1D7A0               MOV     IXH,#0H                 ;
   835 04F0 1D7B8               MOV     IXM,#CIXMB1 AND 0FH     ;BANK1
   836 04F1 1D7C6               MOV     IXL,#CIXLMANT AND 0FH   ;Column address (mantissa)
   837                                                          ;IX <- 00010000110B
   838                  JWC999:
   839             1            SET1    IXE
 +   1 04F2 167F1  1            OR      .MF.IXE SHR 4,#.DF.IXE AND 0FH
   840 04F3 1D200               MOV     .MD.CMDRW,#0H           ;Clear (0)
   841             1            CLR1    IXE
 +   1 04F4 147FE  1            AND     .MF.IXE SHR 4,#.DF.(NOT IXE AND 0FH)
   842 04F5 07080               INC     IX
   843 04F6 097C0               SKE     IXL,#0H                 ;Is the transfer up to 2FH completed?
   844 04F7 0C4F2               BR      JWC999
   845 04F8 070E0               RET
   846
   847                          END


 TOTAL ERRORS   = 0
 TOTAL WARNINGS = 0

 END OF LIST
```