NEC

Application Note

μPD6133 SERIES

REMOTE CONTROL TRANSMISSION

SAMPLE PROGRAMS

Document No.   U11726EJ1V1AN00 (1st edition)
Date Published January 1998 N CP(K)

[MEMO]

The export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

The application circuits and their parameters are for reference only and are not intended for use in actual design-ins.

NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.

While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.

NEC devices are classified into the following three quality grades:
"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated “quality assurance program“ for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.

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

Special: 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: Aircrafts, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books. If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.

Anti-radioactive design is not implemented in this product.

M7 96.5

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Santa Clara, California
Tel: 408-588-6000
800-366-9782
Fax: 408-588-6130
800-729-9288

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.1.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Cumbica-Guarulhos-SP, Brasil
Tel: 011-6465-6810
Fax: 011-6465-6829

J97. 8

INTRODUCTION

Readers

These application notes are intended for engineers who understand the functions of μPD6133 Series infrared remote control transmission microcontrollers and who design application programs using these microcontrollers.

Purpose

The purpose of these application notes is to use application program examples to explain the functions of NEC μPD6133 Series microcontrollers.

Organization

These application notes can be broadly divided into the following sections.

● **GENERAL DESCRIPTION**

- Fundamentals of infrared remote control
- Overview of remote control transmission program

● **48-key program**

- Hardware configuration
- Transmission waveform
- Timing charts
- Output code
- Software configuration
- Program description
- Cautions on program revisions
- Program list

● **80-key program**

- Hardware configuration
- Transmission waveform
- Timing charts
- Output code
- Software configuration
- Program description
- Cautions on program revisions
- Cautions on use of this program
- Program list

Legend

Data representation weight	:	High-order digits are indicated at left and low-order digits at right.
Note	:	Explanation of (Note) in the text
Caution	:	Item deserving extra attention
Remark	:	Supplementary explanation to the text
Number representation	:	Binary number is XXXXB Decimal number is XXXX Hexadecimal number is XXXXH

Related Documents

The following documentation should be referred to when using this manual.
The related documents indicated in this publication may include preliminary versions.
However, preliminary versions are not marked as such.

Document name	Document Number	
	Japanese version	English version
μPD6133, 6134 DATA SHEET	U10454J	U10454E
μPD6604 DATA SHEET	U11281J	U11281E
μPD63, 64 DATA SHEET	U11371J	U11371E
SM6133 VERSION 1.0, USER’S MANUAL	U11128J	U11128E
AS6133 ASSEMBLER USER’S MANUAL	U10115J	U10115E

Caution

The programs in this document are intended as examples only and are not intended for production design purposes.

CONTENTS

LIST OF FIGURES (1/2)

Figure No. Title Page

PART 1 GENERAL

PART 2 48-KEY PROGRAM

PART 3 80-KEY PROGRAM

LIST OF FIGURES (2/2)

LIST OF TABLES

Table No. Title Page

PART 2 48-KEY PROGRAM

PART 3 80-KEY PROGRAM

PART 1

GENERAL

[MEMO]

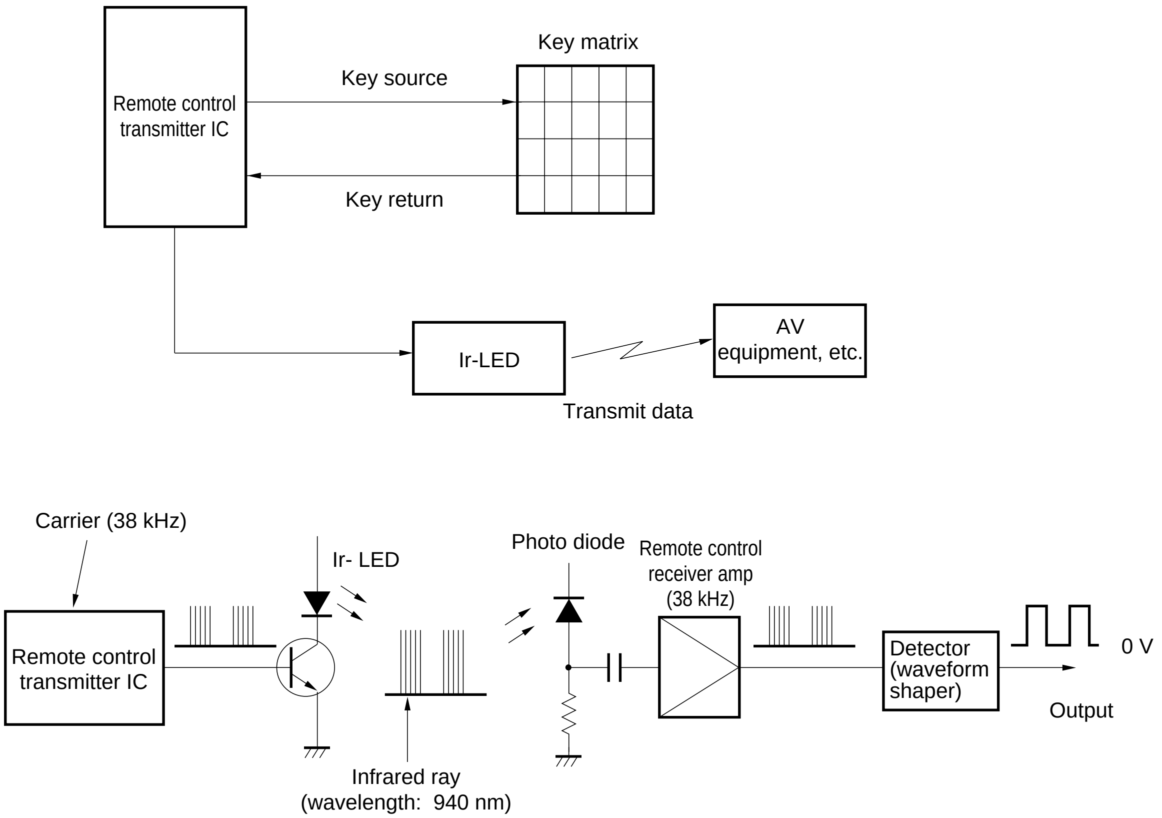
CHAPTER 1 FUNDAMENTALS OF INFRARED REMOTE CONTROL

Infrared remote control transmitters (hereafter referred to as “IR remote control units” or “remote control units”) interrupt an infrared ray to produce transmission of binary (“0” and “1”) data whereby various electronic control devices can be operated under remote control.

The IR remote control unit’s use of an infrared LED (Ir-LED) as the transmitter diode makes for a smaller remote control transmitter that can be manufactured at less cost.

A typical infrared remote control device configuration is shown below.

Figure 1-1. Configuration of IR Remote Control Unit


Remote control transmitter IC
Key matrix
Key source
Key return
Ir-LED
AV equipment, etc.
Transmit data
Carrier (38 kHz)
Remote control transmitter IC
Ir- LED
Infrared ray (wavelength: 940 nm)
Photo diode
Remote control receiver amp (38 kHz)
Detector (waveform shaper)
0 V
Output

To ensure an effective distance of 7 to 10 meters for remote control transmissions, the current supplied to the Ir-LED is generally in the range of 300 to 500 mA. However, since this amount of current would shorten the battery life, modulation is used to produce a pulse current wherein the average current is reduced to one tenth or less of the peak current.

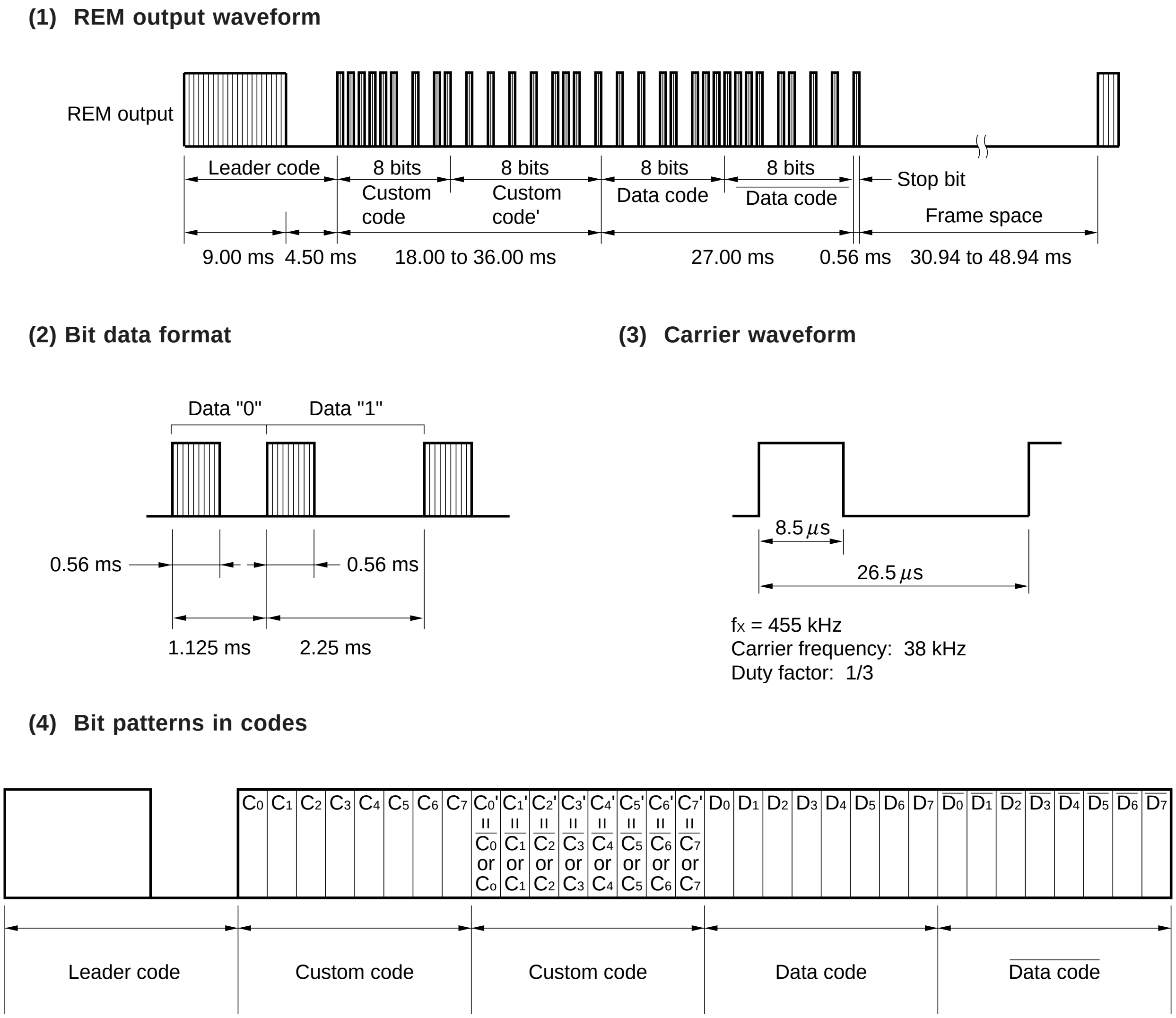
The infrared rays that are output from the Ir-LED are re-converted to electrical signals by a photo diode in the receiver.

Since the received signal is very weak, it is amplified via a receive amplifier before being detected and demodulated into “0” and “1” signals.

An NEC-format transmit waveform is shown below as an example of transmit data from an infrared remote control unit.

The Ir-LED is ON (lit) during high-level periods and OFF (not lit) during low-level periods.

Figure 1-2. Example of NEC-format Transmit Waveform



(1) REM output waveform

(2) Bit data format

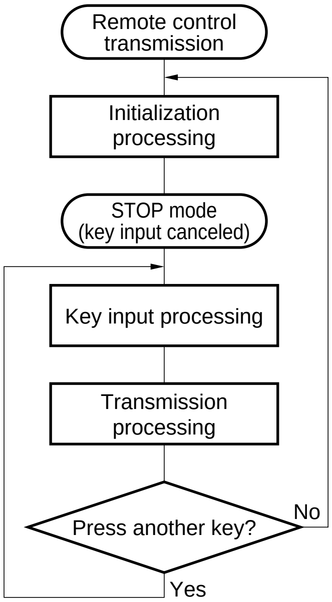
(3) Carrier waveform

(4) Bit patterns in codes

REM output
Leader code
8 bits
Custom code
8 bits
Custom code'
8 bits
Data code
8 bits
$\overline{\text{Data code}}$
Stop bit
Frame space
9.00 ms
4.50 ms
18.00 to 36.00 ms
27.00 ms
0.56 ms
30.94 to 48.94 ms
Data "0"
Data "1"
0.56 ms
0.56 ms
1.125 ms
2.25 ms
8.5 μs
26.5 μs
f_X = 455 kHz
Carrier frequency: 38 kHz
Duty factor: 1/3
C_0 C_1 C_2 C_3 C_4 C_5 C_6 C_7
$C_0' = \overline{C_0}$ or C_0
$C_1' = \overline{C_1}$ or C_1
$C_2' = \overline{C_2}$ or C_2
$C_3' = \overline{C_3}$ or C_3
$C_4' = \overline{C_4}$ or C_4
$C_5' = \overline{C_5}$ or C_5
$C_6' = \overline{C_6}$ or C_6
$C_7' = \overline{C_7}$ or C_7
D_0 D_1 D_2 D_3 D_4 D_5 D_6 D_7
$\overline{D_0}$ $\overline{D_1}$ $\overline{D_2}$ $\overline{D_3}$ $\overline{D_4}$ $\overline{D_5}$ $\overline{D_6}$ $\overline{D_7}$
Leader code
Custom code
Custom code
Data code
$\overline{\text{Data code}}$

CHAPTER 2 OVERVIEW OF REMOTE CONTROL TRANSMISSION PROGRAM

2.1 General Flow Chart


Remote control transmission
Initialization processing
STOP mode (key input canceled)
Key input processing
Transmission processing
Press another key?
No
Yes

2.2 Overview of Processing

2.2.1 Initialization processing

Since most remote control units use batteries as their power source, it should be assumed that RAM and port contents may undergo sudden changes due to power supply voltage fluctuations during the Ir-LED's infrared ray emissions.

Consequently, a program has been implemented to execute RAM and port initialization processing when each remote control signal is transmitted.

Initialization processing includes the following settings.

- Port settings, carrier frequency/duty factor, carrier setting
- Carrier frequency and timer/clock division settings
- I/O mode settings and pull-down resistor settings for each port

2.2.2 STOP mode

When in STOP mode, program execution is stopped along with system clock oscillation to enable low current consumption.

In the remote control program, setting STOP mode as a condition for canceling key input enables current consumption to be reduced as long as no key is pressed (while in standby mode).

2.2.3 Key input processing

Key input processing includes the following.

- Key scanning to determine if any key has been pressed
- Ensured reliability in key operations (elimination of chattering)
- Generation of key data corresponding to each pressed key, which is stored in RAM.

2.2.4 Transmission processing

Transmission processing includes the following.

- Transmit data corresponding to the key data generated by key input processing is transmitted according to the specific manufacturer's remote control transmission format.
- After the first transmission is completed, the unit detects whether or not the key is still being pressed.
- If it is determined that the key is still being pressed, key input processing checks whether or not the pressed key has changed.
- If it is determined that the key has been released, initialization processing is executed, after which the unit enters STOP mode to reduce power consumption until a key is pressed.

PART 2

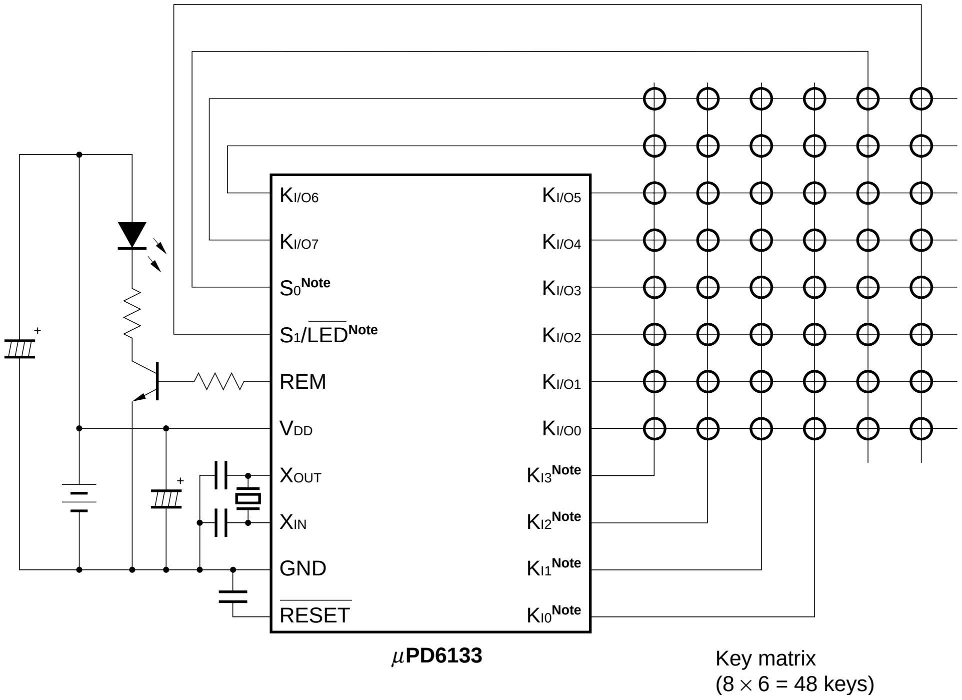
48-KEY PROGRAM

[MEMO]

1.1 Application Circuit Example

Figure 1-1 shows an application circuit example.

Figure 1-1. Application Circuit Example



Note The program is set for "on-chip pull-down resistors".

The K_I, $K_{I/O}$, S_0, and $S_1/\overline{LED}$ pins are used to configure a 48-key key matrix. The transmission code is output via the REM pin with a carrier signal.

Table 1-1 lists the various pin functions.

Table 1-1. Pin Functions

Pin Name	I/O	Function
$K_{I/O0}$ to $K_{I/O7}$	Output	Key source (active high)
K_{I0} to K_{I3}	Input	Key return (active high)
S_0	Input	
$S_1/\overline{LED}$	Input	
REM	Output	Infrared remote control signal (with carrier)

1.2 Key Matrix

Figure 1-2 illustrates the key matrix.

The “Kn” symbol (in which n = 1 to 48) indicates each key’s position.

Figure 1-2. Key Matrix

$K_{I/O7}$	K29	K30	K31	K32	K47	K48
$K_{I/O6}$	K25	K26	K27	K28	K45	K46
$K_{I/O5}$	K21	K22	K23	K24	K43	K44
$K_{I/O4}$	K17	K18	K19	K20	K41	K42
$K_{I/O3}$	K13	K14	K15	K16	K39	K40
$K_{I/O2}$	K9	K10	K11	K12	K37	K38
$K_{I/O1}$	K5	K6	K7	K8	K35	K36
$K_{I/O0}$	K1	K2	K3	K4	K33	K34
	K_{I3}	K_{I2}	K_{I1}	K_{I0}	S_1	S_0

CHAPTER 2 TRANSMISSION WAVEFORM

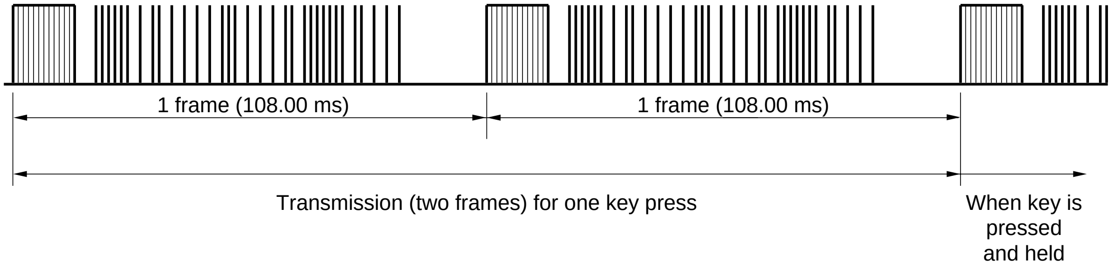
The transmission waveform that is output from the REM pin uses the NEC-R (NEC continuous) format. For description of the output data code, see “CHAPTER 4 OUTPUT CODES”.

2.1 NEC-R Format

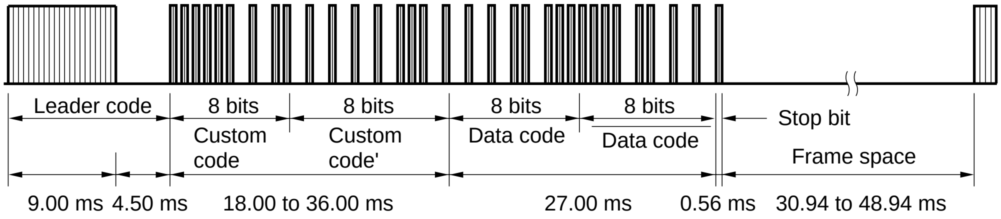
Figure 2-1 illustrates the NEC-R format.

Figure 2-1. NEC-R Format

- **Transmission waveform**

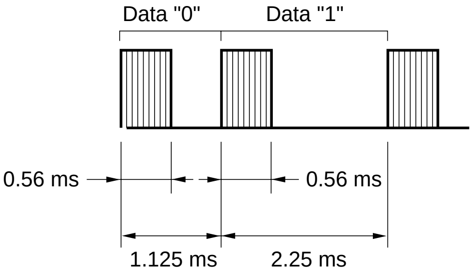

1 frame (108.00 ms)
1 frame (108.00 ms)
Transmission (two frames) for one key press
When key is pressed and held

- **Transmission waveform for first frame**

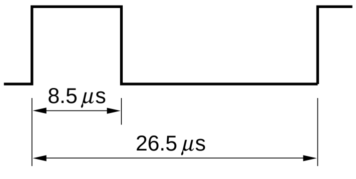


- **Transmission waveform for second and subsequent frames**...same as transmission waveform for first frame

- **Bit data format**



- **Carrier waveform**



fX = 455 kHz
Carrier frequency: 38 kHz
Duty factor: 1/3

[MEMO]

CHAPTER 3 TIMING CHARTS

3.1 Timing Charts for Key Input to REM Output

"ON chattering" elimination processing checks for key input every 9.00 ms and if it detects ON status three consecutive times, it determines that the key is ON. "ON chattering" elimination processing also checks for key status changes (between ON and OFF or when the key is pressed and held).

The OFF chattering elimination processing is described below.

Key OFF status is checked during low-level output of the bit data.

During transmission of one frame (108 ms), key input is checked ten times with reference to the timing (34 times) of the low-level output from the REM pin.

(1) If key OFF status is detected all ten times
... Key OFF status is determined.

(2) If key ON status is detected during at least one of the ten times
... When key ON status is detected, the check counter is cleared and key input is checked another ten times.

(3) If (1) and (2) above do not determine key OFF status
... Key press and hold status (key ON status) is determined.

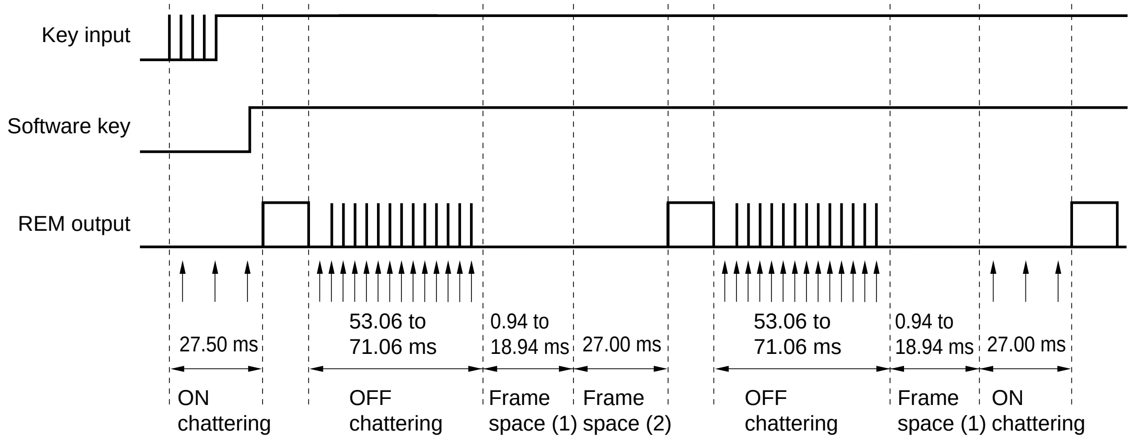
Even if key OFF status is determined during transmission of the first frame, initialization processing does not begin until after the second or a subsequent frame is transmitted.

3.1.1 Timing when a key is pressed and held

Figure 3-1 shows a timing chart for when a key is pressed and held.

For details of ON chattering elimination, see "6.2.1 (1) Chattering Elimination Processing".

Figure 3-1. Timing Chart when Key is Pressed and Held



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1): This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

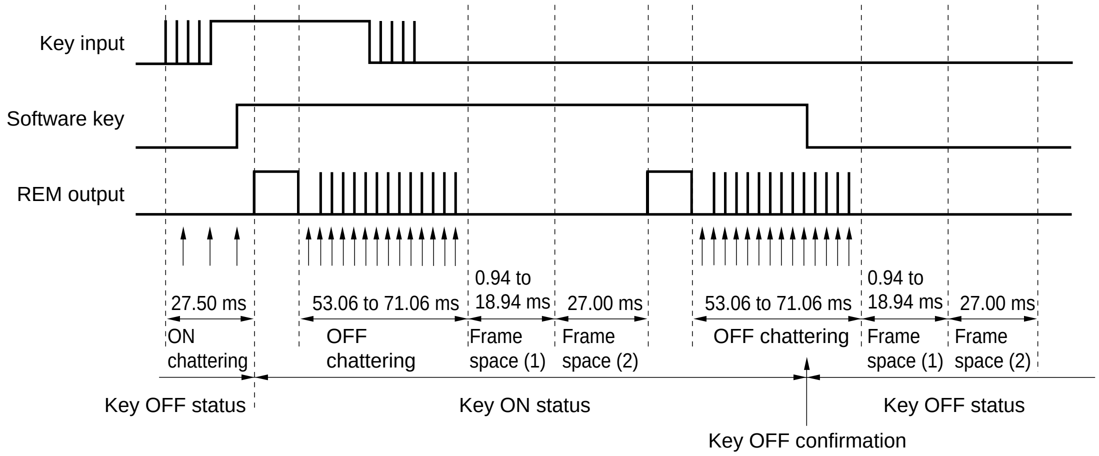
Frame space (2): When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

3.1.2 Timing during key OFF status

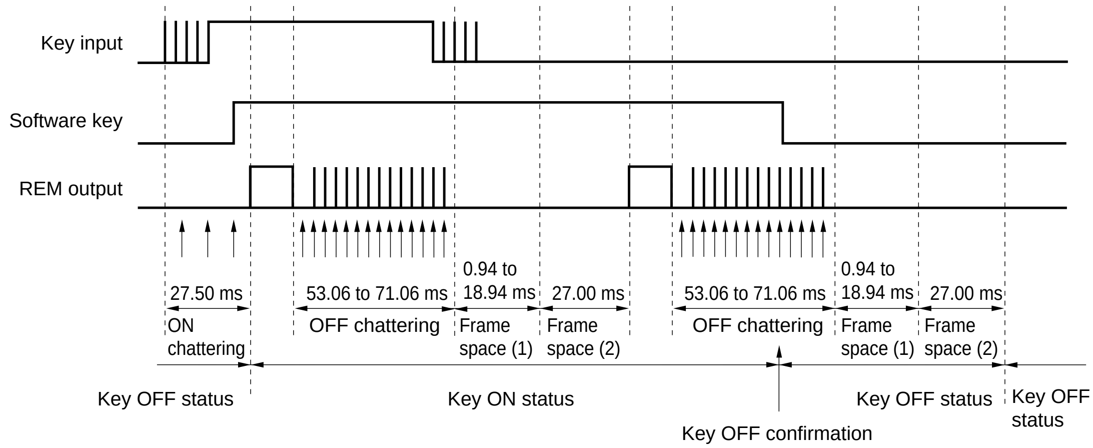
Figure 3-2 shows timing charts for when a key is released during a frame transmission.

Figure 3-2. Timing Chart for Key OFF Status (1/3)

**(1) When key is released during transmission of the first frame
(key OFF status has been determined ten times consecutively during transmission of the first frame)
... Key OFF status is confirmed during transmission of second frame**



**(2) When key is released during transmission of the first frame
(key OFF status has not been determined ten times consecutively during transmission of the first frame)
... Key OFF status is confirmed during transmission of second frame**



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

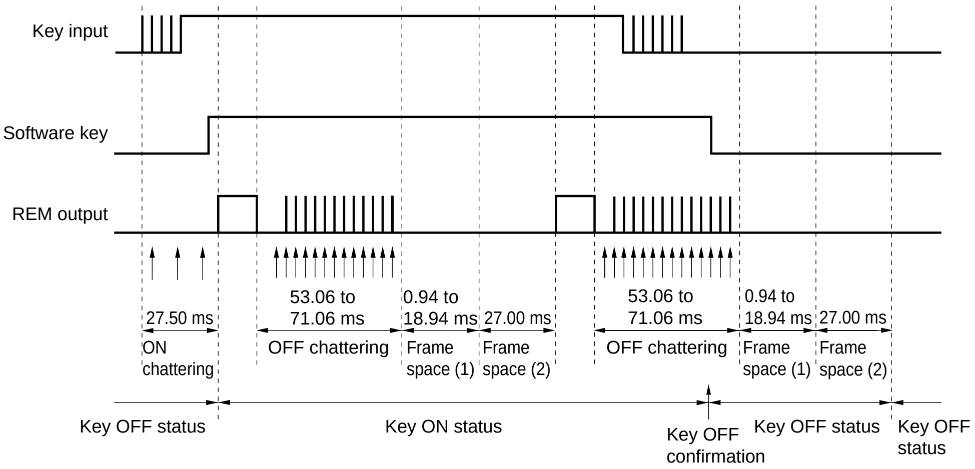
OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1): This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

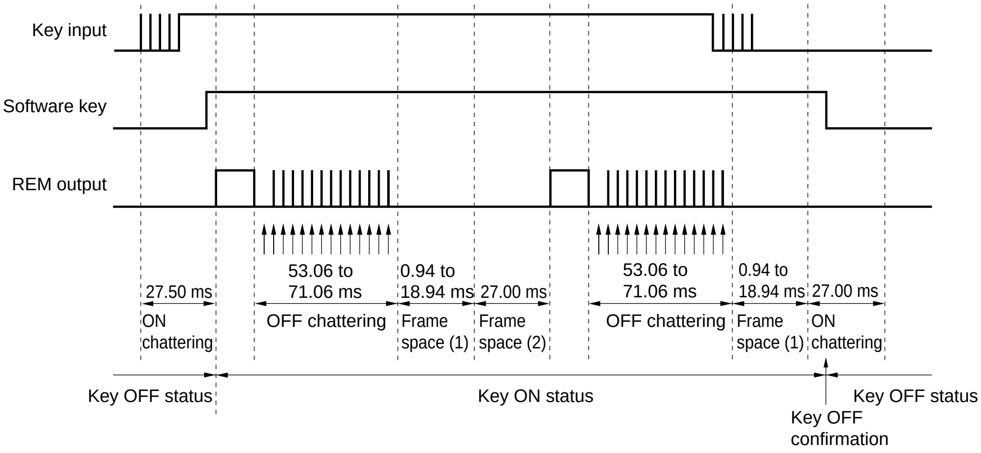
Frame space (2): When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

Figure 3-2. Timing Chart for Key OFF Status (2/3)

(3) When key is released during transmission of the second frame (key OFF status has been determined ten times consecutively during transmission of the second frame)
... Key OFF status is confirmed during transmission of second frame



(4) When key is released during transmission of the second frame (key OFF status has not been determined ten times consecutively during transmission of the second frame)
... Key OFF status is confirmed by the key's ON chattering



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

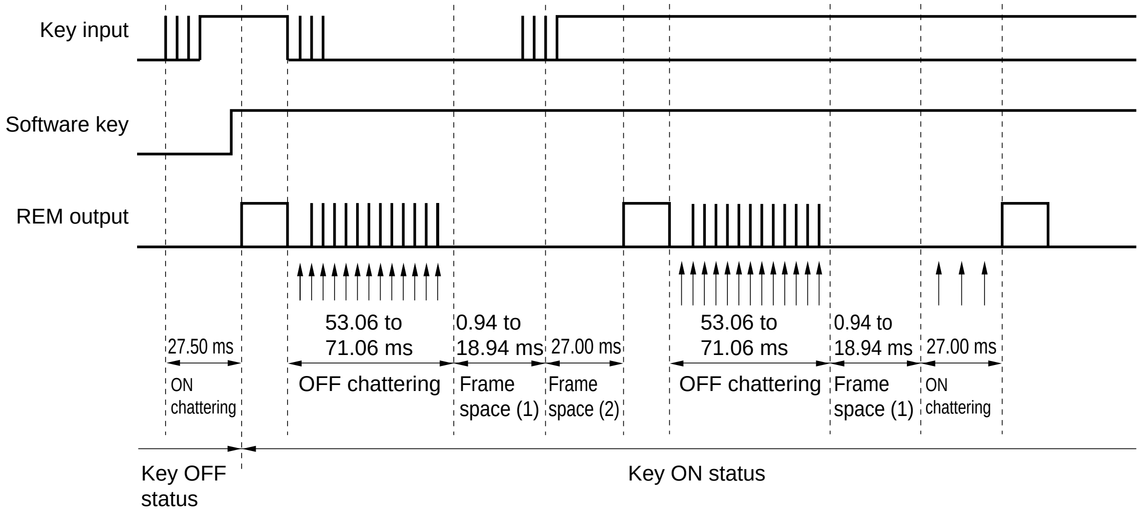
OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1): This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

Frame space (2): When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

Figure 3-2. Timing Chart for Key OFF Status (3/3)

(5) When key is released during transmission of the first frame and is pressed again later ... Key OFF cannot be confirmed since a key ON status is detected during the second frame



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1): This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

Frame space (2): When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

3.2 Timing Charts of Key Operations

3.2.1 Output patterns prior to key confirmation

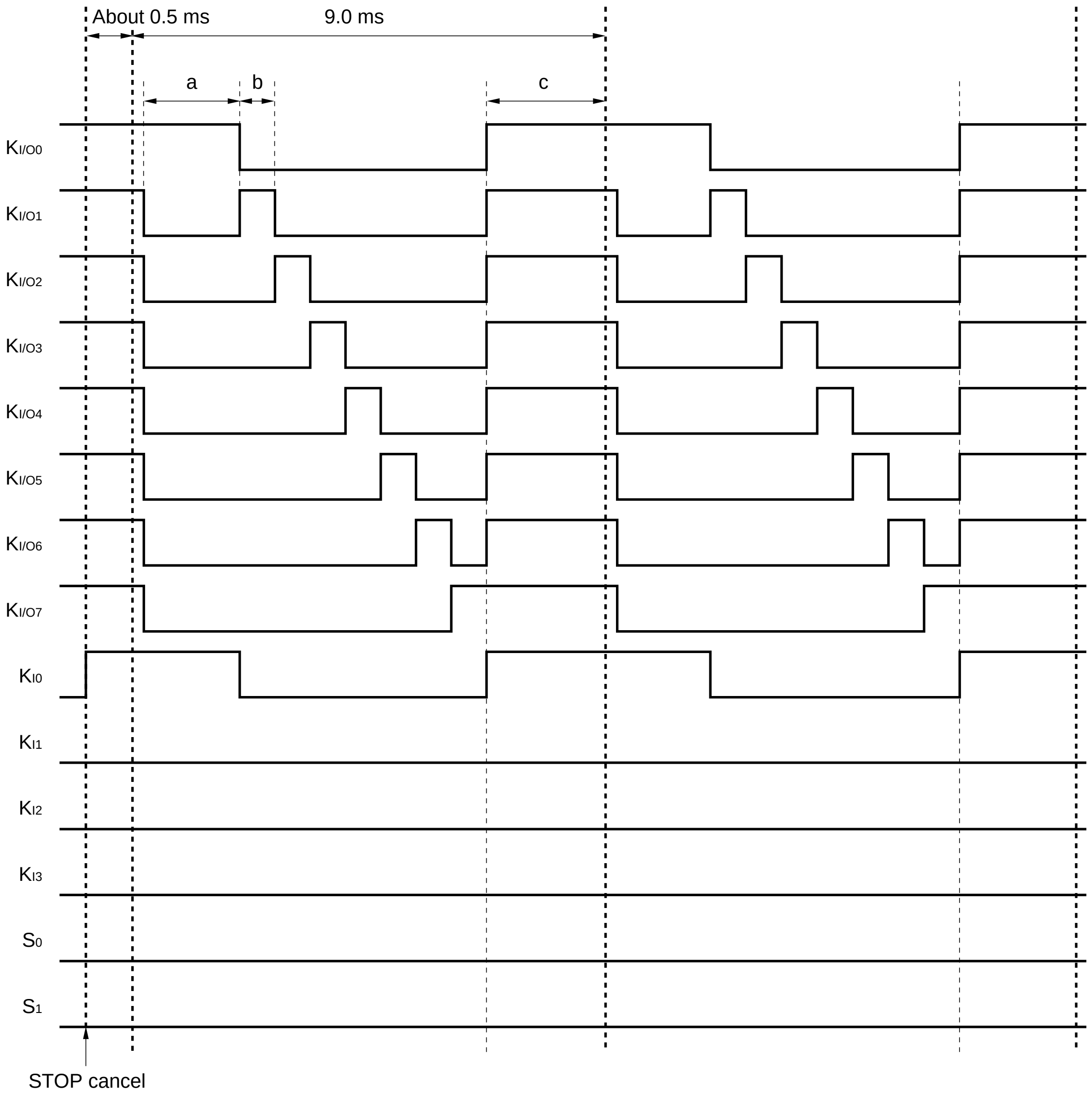
Prior to key confirmation, ON chattering elimination processing (9.00 ms × 3 times = 27.00 ms) is performed and a key is confirmed when it has the same status all 3 times.

Key input confirmation is determined after at least 100 μs (at least 6 instructions when operating at 455 kHz) has elapsed after the key source output pin goes to high level.

Figure 3-3 shows the detailed output patterns prior to key confirmation.

Figure 3-3. Output Patterns Prior to Key Confirmation (1/5)

(1) When K4 is pressed



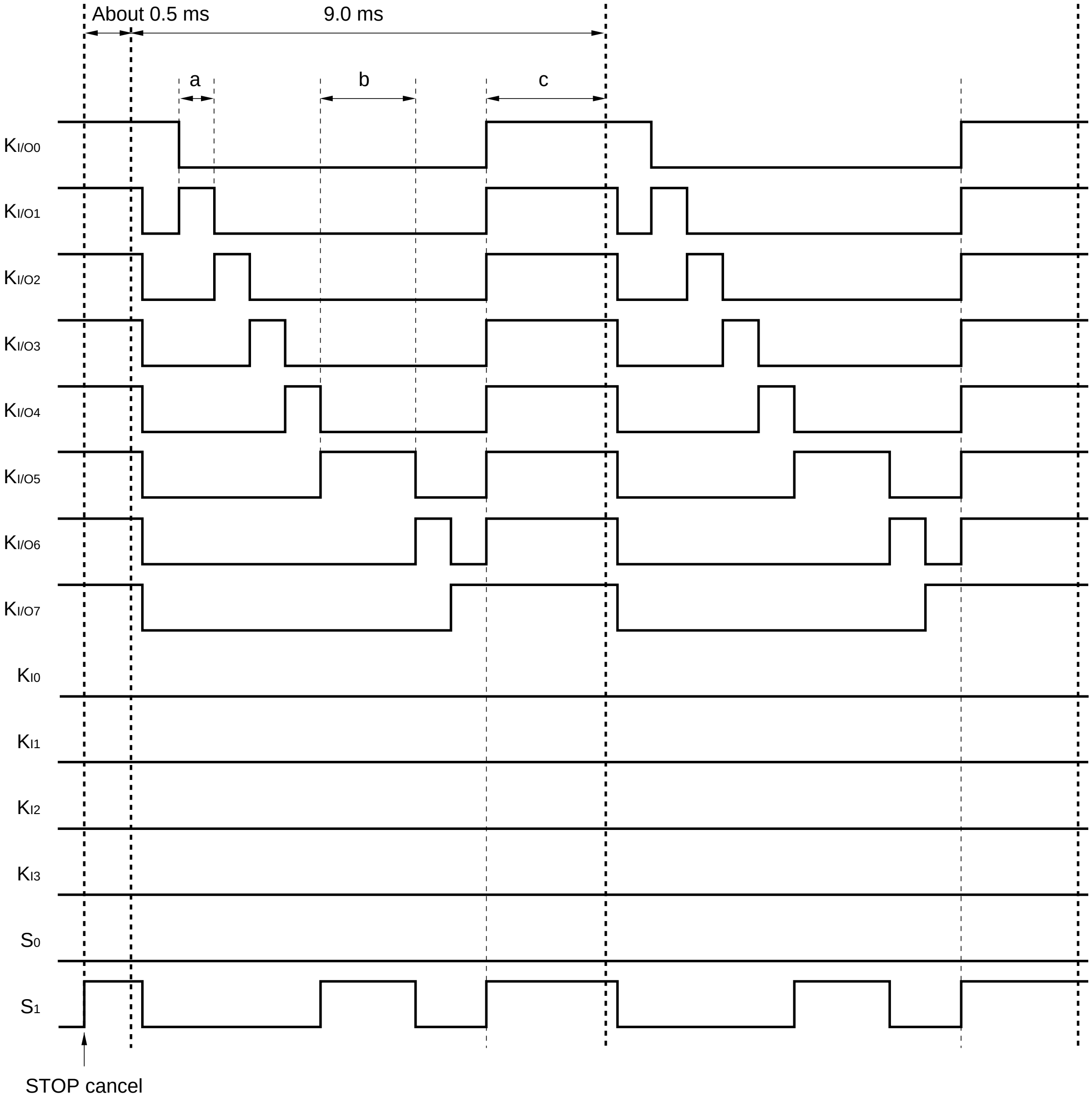
a: When a key ON status has been determined, the key return is checked and the key data is calculated. Execution time ranges from approximately 1.34 ms to 1.95 ms.

b: Key input is checked. Execution time is approximately 0.41 ms.

c: This is the total output time for key scanning. This time varies according to the value of “a”. Execution time ranges from approximately 4.23 ms to 4.84 ms.

Figure 3-3. Output Patterns Prior to Key Confirmation (2/5)

(2) When K43 is pressed



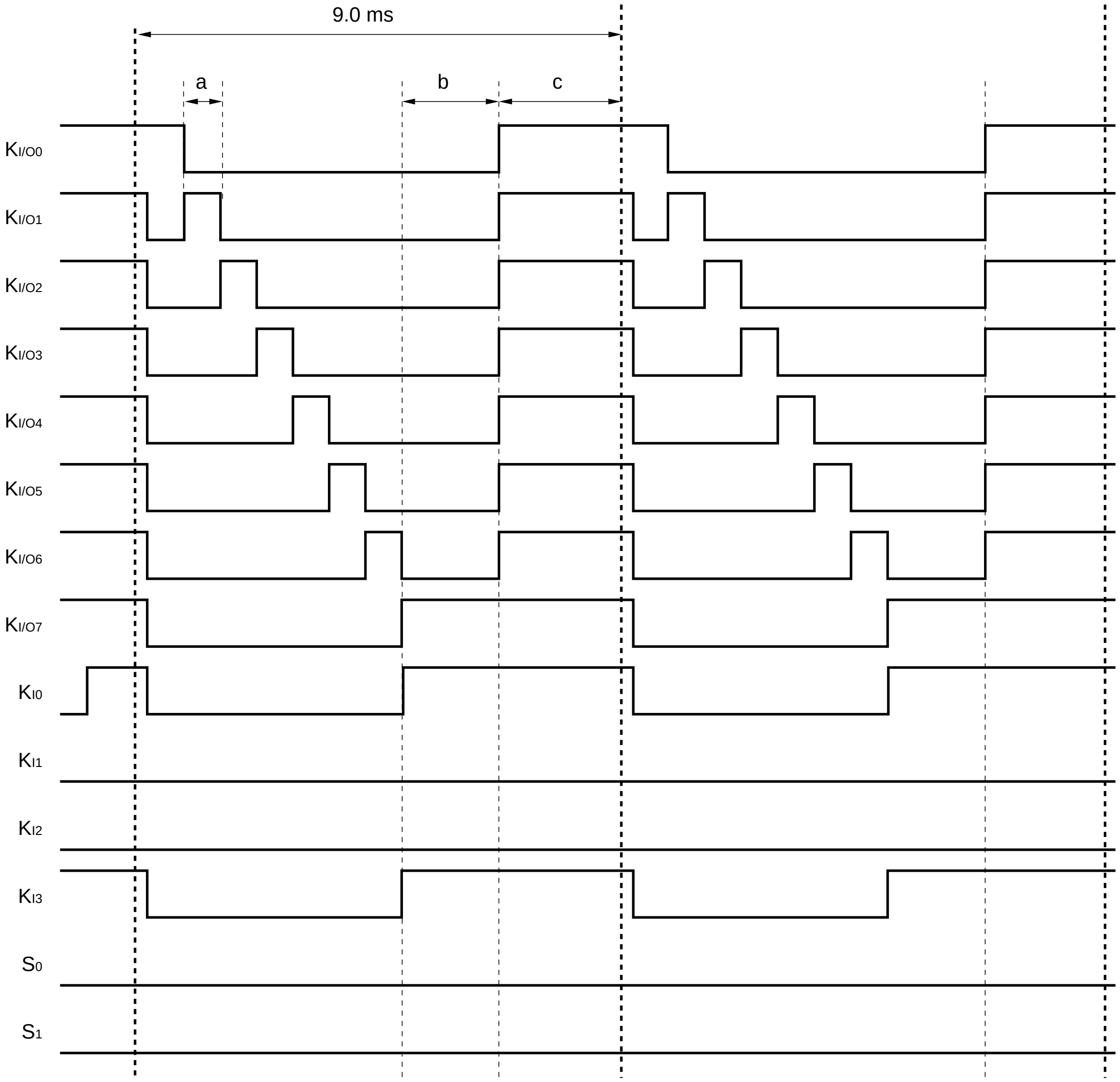
a : When a key ON status has been determined, the key return is checked and the key data is calculated. Execution time ranges from approximately 1.34 ms to 1.95 ms.

b : Key input is checked. Execution time is approximately 0.41 ms.

c : This is the total output time for key scanning. This time varies according to the value of “a”. Execution time ranges from approximately 4.23 ms to 4.84 ms.

Figure 3-3. Output Patterns Prior to Key Confirmation (3/5)

(3) When K29 is pressed along with valid combination key K32 (K29 + K32)



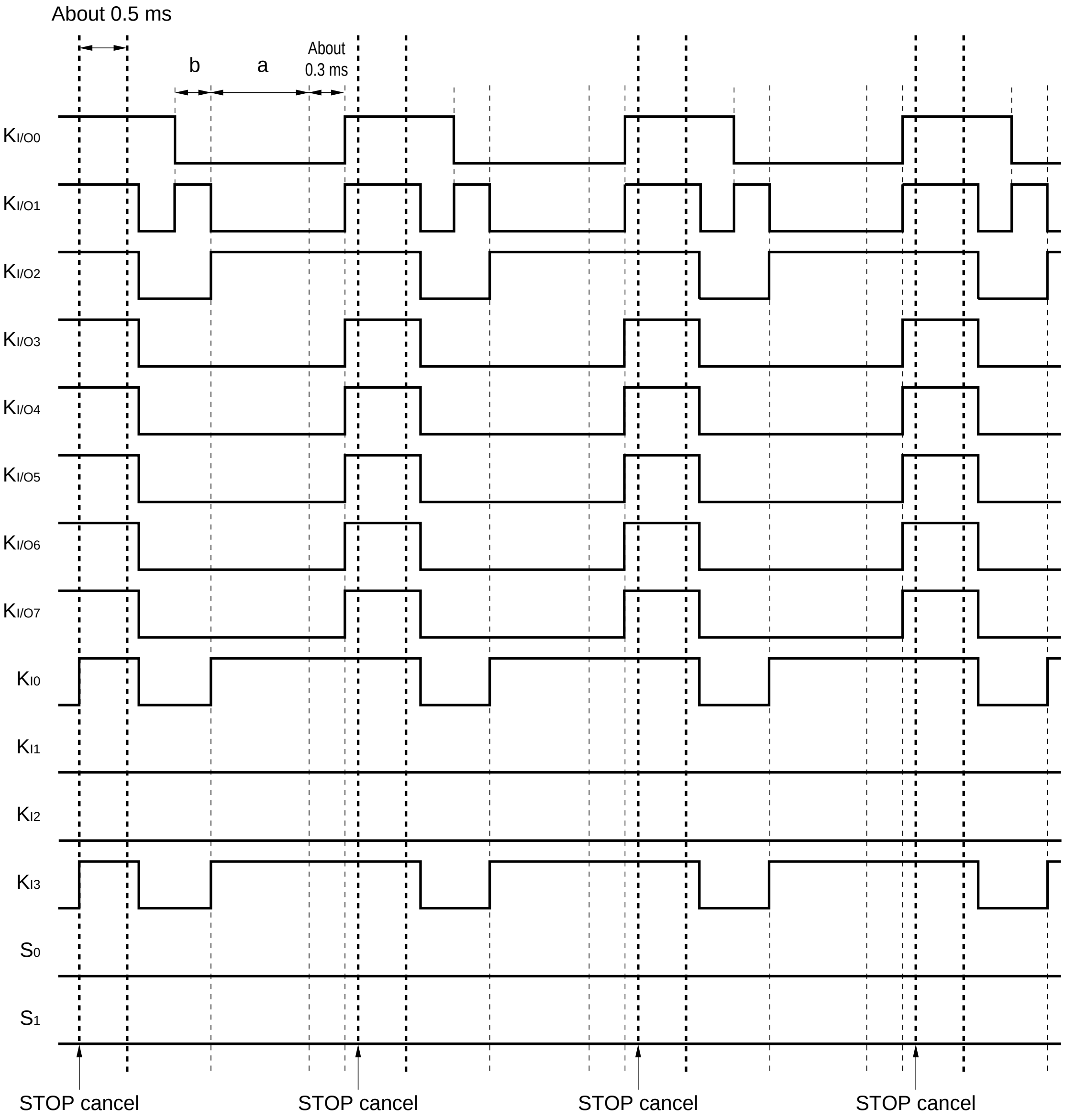
a: When a key ON status has been determined, the key return is checked and the key data is calculated.
Execution time ranges from approximately 1.34 ms to 1.95 ms.

b: Key input is checked.
Execution time is approximately 0.41 ms.

c: This is the total output time for key scanning. This time varies according to the value of “a”.
Execution time ranges from approximately 4.23 ms to 4.84 ms.

Figure 3-3. Output Patterns Prior to Key Confirmation (4/5)

(4) When K9 and K12 are pressed but are invalid in identical key sources

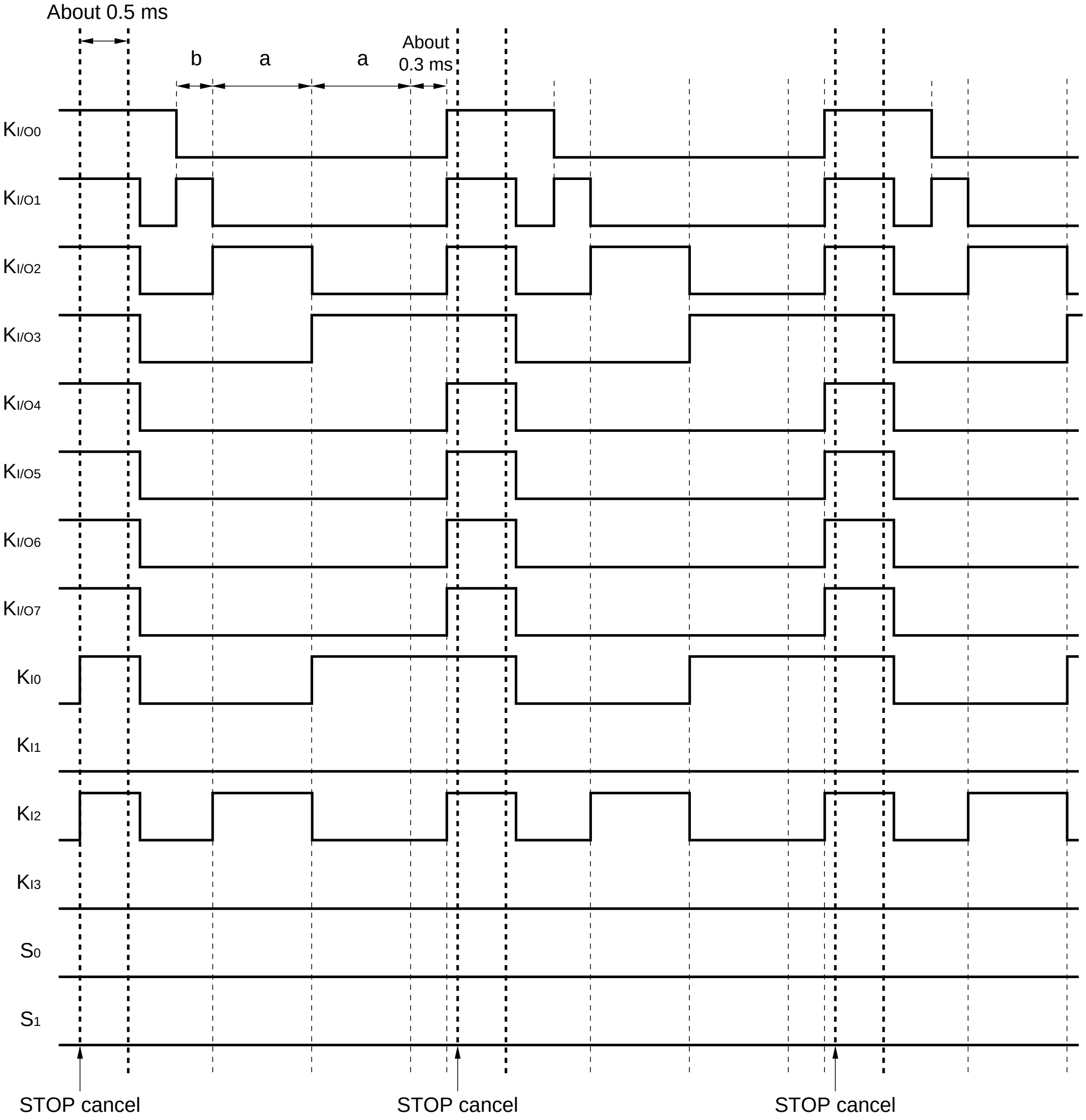


a: When a key ON status has been determined, the key return is checked and the key data is calculated.
Execution time ranges from approximately 1.34 ms to 1.95 ms.

b: Key input is checked.
Execution time is approximately 0.41 ms.

Figure 3-3. Output Patterns Prior to Key Confirmation (5/5)

(5) When K10 and K16 are pressed but are invalid in identical key sources



a: When a key ON status has been determined, the key return is checked and the key data is calculated. Execution time ranges from approximately 1.34 ms to 1.95 ms.

b: Key input is checked. Execution time is approximately 0.41 ms.

3.2.2 Operation of combination key

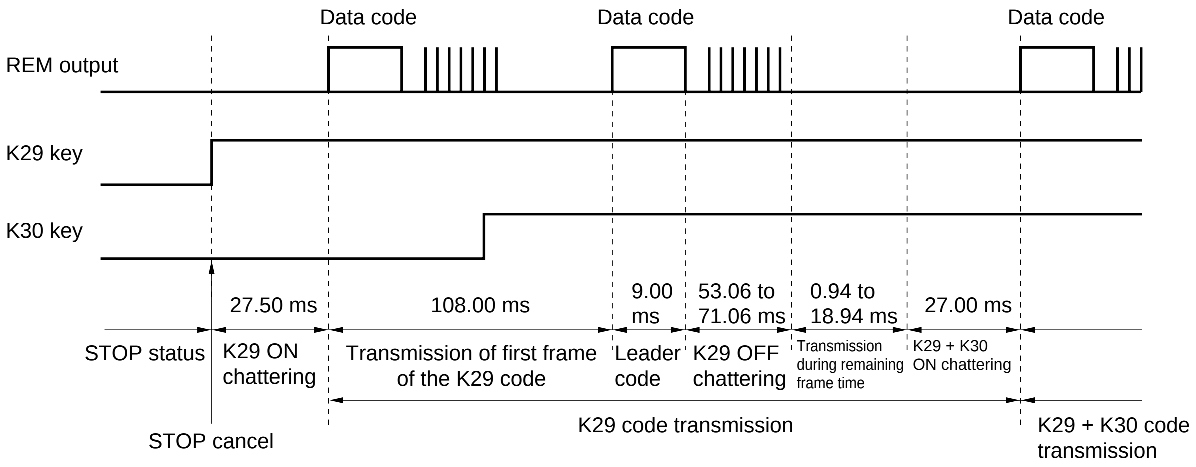
K29 is the combination key. There are three valid key combination pairs: K29 + K30, K29 + K31, and K29 + K32. For any pattern, a key combination is valid only when K29 is pressed first.

In other words, if another key is pressed before or at the same time as K29, the key combination is not valid.

Figure 3-4 shows examples of key operations when a key combination is entered.

Figure 3-4. Timing Chart of Key Combination Operation (1/2)

(1) If another valid key (K30 to K32) is pressed during transmission of the first frame of the K29 code, after the second frame of the K29 code is transmitted, the key combination becomes valid and the combination's code is transmitted.



(2) If another valid key (from K30 to K32) is pressed during transmission of the third frame of the K29 code, after the K29 code and the third frame are transmitted, the key combination becomes valid and the combination's code is transmitted.

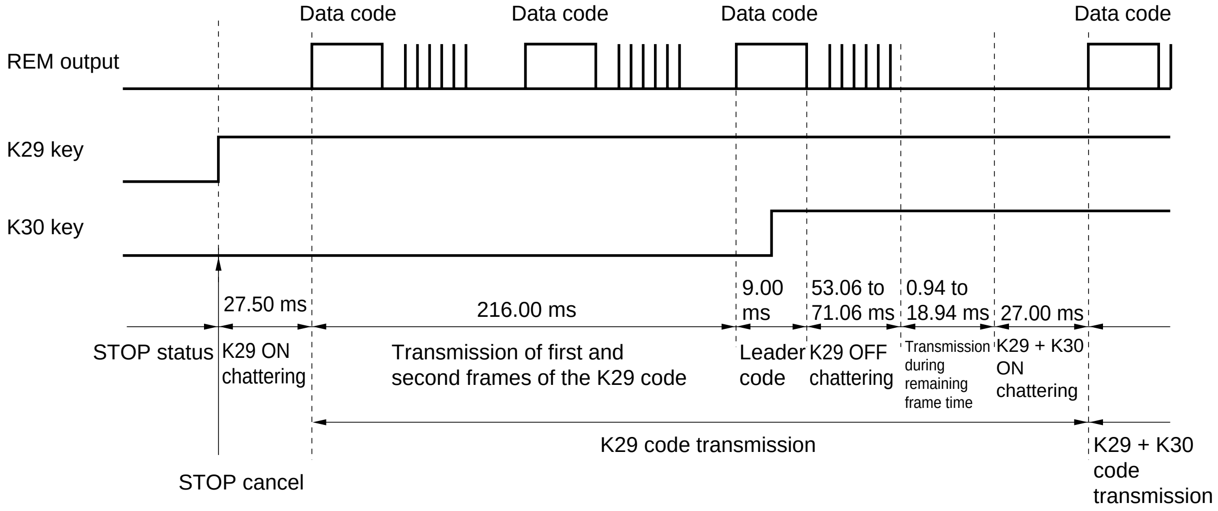
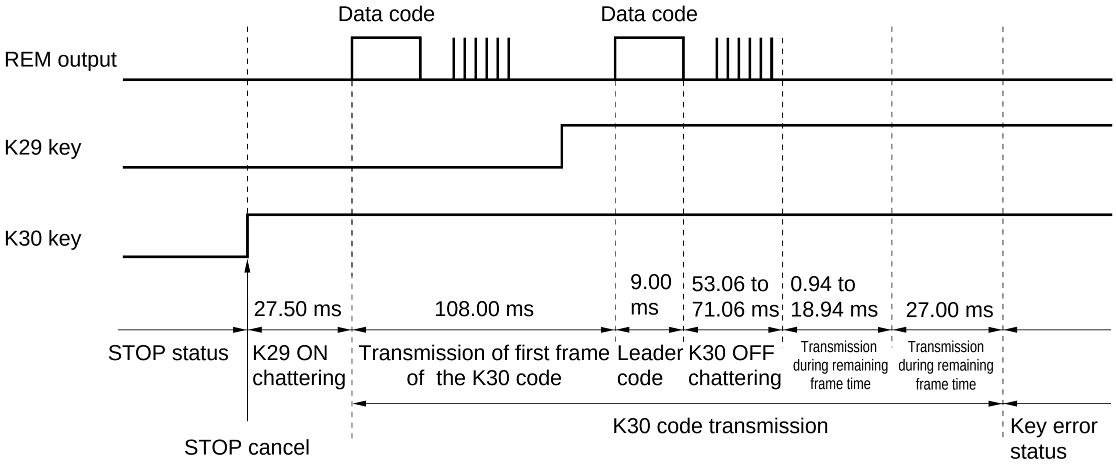
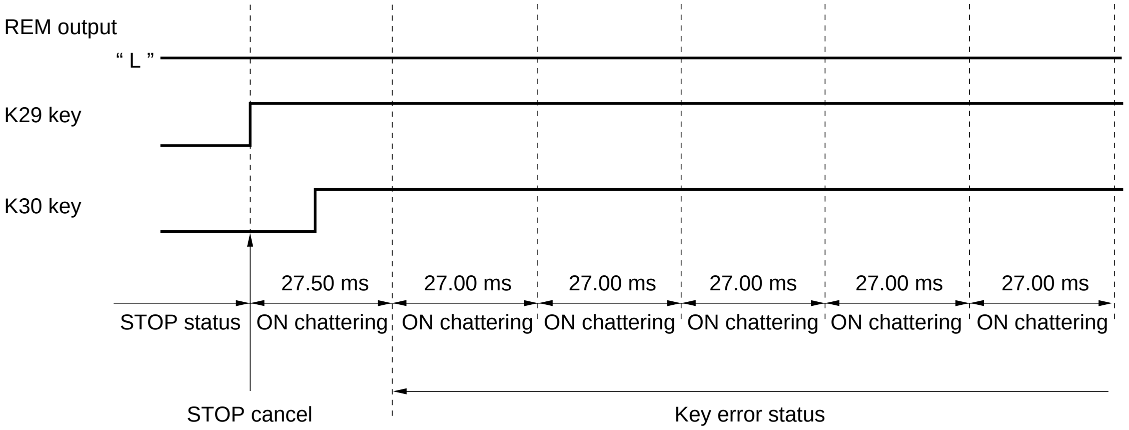


Figure 3-4. Timing Chart of Key Combination Operation (2/2)

(3) If K29 is pressed after another valid key (K30 to K32) has been confirmed, a key check produces a key error and no code is transmitted (the key combination is not valid).



(4) If another valid key (K30 to K32) is pressed before K29 has been confirmed, a key check produces a key error and no code is transmitted (the key combination is not valid).



3.2.3 Key transfer operation

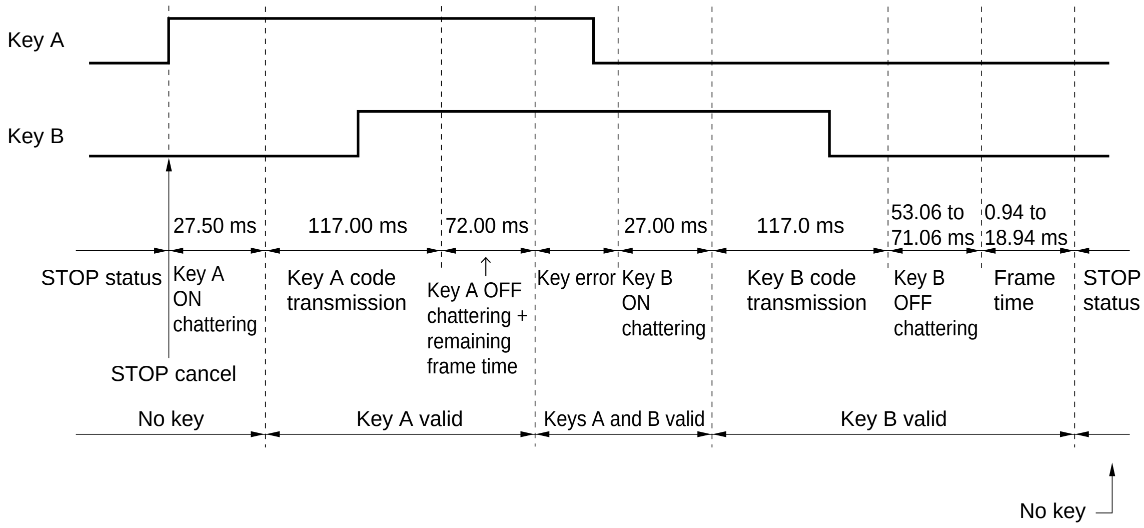
A key transfer operation can be performed when a valid combination key has been pressed.

The key transfer operation is an operation that occurs when a second key is pressed before a previously pressed key is released. The operation for the second key is performed when the previous key is released.

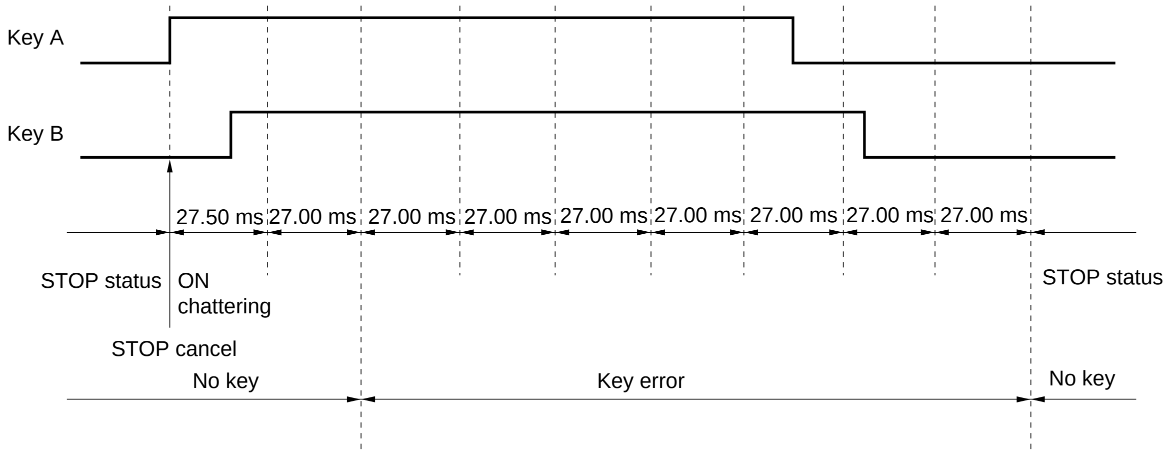
Figure 3-5 shows the operation timing of a key transfer operation.

Figure 3-5. Operation Timing of Key Transfer Operation

(1) When a key transfer operation is performed to transfer to key B during transmission of the code for key A, a key check is performed after all of the code has been transmitted. A key error occurs if it is during a key combination period.
When key A is released, key B becomes valid and its code is transmitted.



(2) If the transfer to key B occurs before key A is confirmed, a key error occurs during the key check. A key error also occurs if it is during a key combination period.
After key A is released, a key check is performed. Key B was released before it could be checked, so the check result is "no key".



[MEMO]

CHAPTER 4 OUTPUT CODES

Table 4-1 lists the output codes.

For output using the NEC format, NEC provides each customer with a custom code to avoid the risk of interfering with output from another remote control unit that outputs using the NEC format.

This program is set to output 0AH as the custom code and F5H as the custom code'.

Contact your NEC sales representative for information on obtaining a custom code.

Table 4-1. Output Codes

Key No.	Custom code	Custom code'	Data	Key No.	Custom code	Custom code'	Data	Key No.	Custom code	Custom code'	Data
K1	0AH	F5H	00H	K17	0AH	F5H	50H	K33	0AH	F5H	A0H
K2	0AH	F5H	01H	K18	0AH	F5H	51H	K34	0AH	F5H	A1H
K3	0AH	F5H	02H	K19	0AH	F5H	52H	K35	0AH	F5H	A2H
K4	0AH	F5H	03H	K20	0AH	F5H	53H	K36	0AH	F5H	A3H
K5	0AH	F5H	04H	K21	0AH	F5H	54H	K37	0AH	F5H	A4H
K6	0AH	F5H	05H	K22	0AH	F5H	55H	K38	0AH	F5H	A5H
K7	0AH	F5H	06H	K23	0AH	F5H	56H	K39	0AH	F5H	A6H
K8	0AH	F5H	07H	K24	0AH	F5H	57H	K40	0AH	F5H	A7H
K9	0AH	F5H	08H	K25	0AH	F5H	58H	K41	0AH	F5H	A8H
K10	0AH	F5H	09H	K26	0AH	F5H	59H	K42	0AH	F5H	A9H
K11	0AH	F5H	0AH	K27	0AH	F5H	5AH	K43	0AH	F5H	AAH
K12	0AH	F5H	0BH	K28	0AH	F5H	5BH	K44	0AH	F5H	ABH
K13	0AH	F5H	0CH	K29	0AH	F5H	5CH	K45	0AH	F5H	ACH
K14	0AH	F5H	0DH	K30	0AH	F5H	5DH	K46	0AH	F5H	ADH
K15	0AH	F5H	0EH	K31	0AH	F5H	5EH	K47	0AH	F5H	AEH
K16	0AH	F5H	0FH	K32	0AH	F5H	5FH	K48	0AH	F5H	AFH

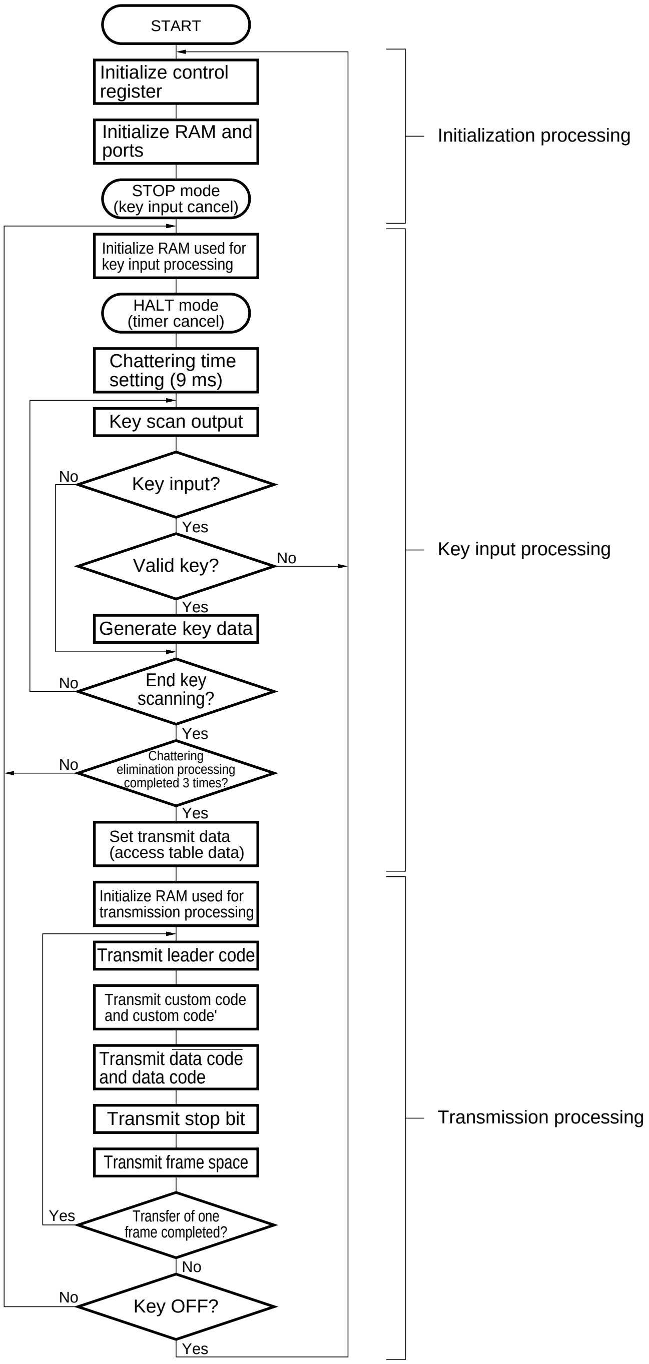
Key No.	Custom code	Custom code'	Data
K29 + K30	0AH	F5H	90H
K29 + K31	0AH	F5H	91H
K29 + K32	0AH	F5H	92H

[MEMO]

5.1 General Flow Chart

Figure 5-1 shows a general flow chart of this program.

Figure 5-1. General Flow Chart


START
Initialize control register
Initialize RAM and ports
STOP mode (key input cancel)
Initialize RAM used for key input processing
HALT mode (timer cancel)
Chattering time setting (9 ms)
Key scan output
Key input?
No
Yes
Valid key?
No
Yes
Generate key data
End key scanning?
No
Yes
Chattering elimination processing completed 3 times?
No
Yes
Set transmit data (access table data)
Initialize RAM used for transmission processing
Transmit leader code
Transmit custom code and custom code'
Transmit data code and data code
Transmit stop bit
Transmit frame space
Transfer of one frame completed?
Yes
No
Key OFF?
No
Yes
Initialization processing
Key input processing
Transmission processing

5.2 Program Memory (ROM) Configuration

The μPD6133's program memory (ROM) consists of 512 steps × 10 bits. Figure 5-2 shows a ROM map for this program and Figure 5-3 shows a table data map.

Figure 5-2. ROM Map

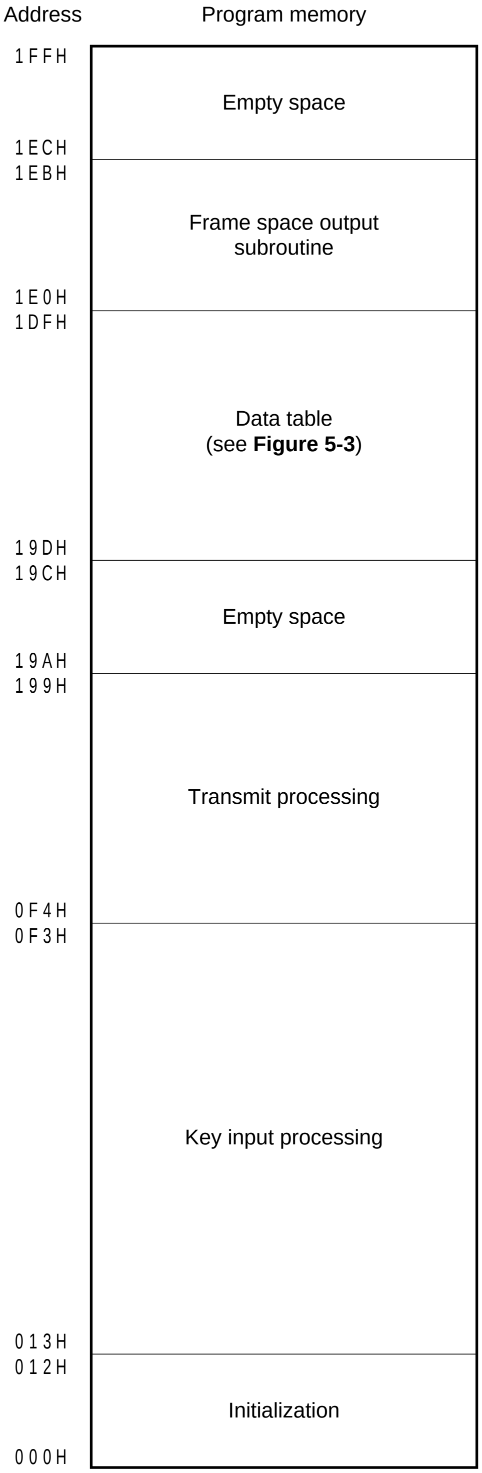
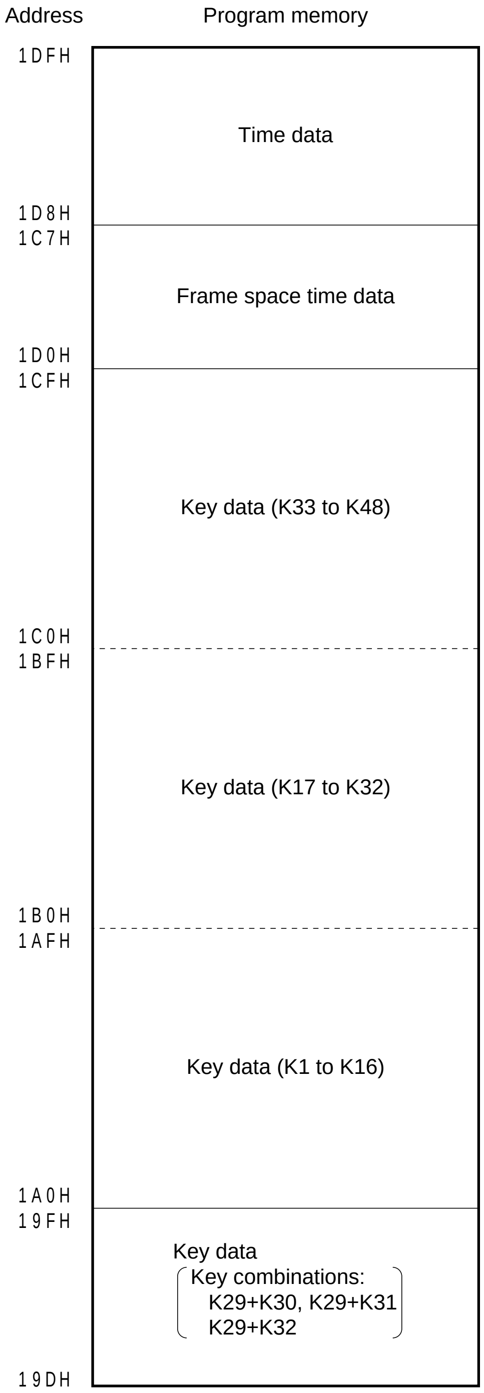


Figure 5-3. Data Table Map



5.3 Data Memory (RAM) Configuration

The data memory (RAM) consists of 32 × 4 bits static RAM, which is used to store processing data. Some instructions enable RAM contents to be manipulated in 8 bits units.

R0 can function as a data pointer for ROM addresses. ROM contents can be accessed once a ROM address is set to this data pointer. This is called a table lookup for ROM data. When reset, the value of R0 becomes 00H.

RF can also be used as an address stack register. When reset, the values of R1 to RF are undefined.

Tables 5-1 to 5-3 show RAM maps for the entire program (key input processing and transmission processing), for key input processing alone, and for transmission processing alone. Table 5-4 describes RAM usage.

Table 5-1. RAM Map

	H (1)	L (0)
R0	Work area 1 Data pointer H	Chattering counter Data pointer L
R1	Confirmation key data H	Confirmation key data L
R2	For immediate setting (0EH)	For immediate setting (1H)
R3	K29 ON flag	Continue flag
R4	Key data H Custom code H	Key data L Custom code L
R5	Custom code' H	Custom code' L
R6	Key ON flag Data code H	Key scan counter Data code L
R7	For immediate setting (0FH)	Key OFF check counter
R8	Key scan data H For immediate setting (0FH) Data 1	Key scan data L For immediate setting (0H) Data 0
R9	Compare key, data H Data 1 transmit counter H	Compare key, data L Data 1 transmit counter L
RA	Work area 2	Key return check counter Transmit bit counter
RB	For immediate setting (0CH)	For immediate setting (3H)
RC	Not used	Not used
RD	Not used	Not used
RE	Not used	Not used
RF	Address stack register	

Table 5-2. Map of RAM Used for Key Input Processing

	H (1)	L (0)
R0	Work area 1 Data pointer H	Chattering counter Data pointer L
R1	Confirmation key data H	Confirmation key data L
R2	For immediate setting (0EH)	For immediate setting (1H)
R3	K29 ON flag	Continue flag
R4	Key data H Custom code H	Key data L Custom code L
R5	Custom code’ H	Custom code’ L
R6	Key ON flag Data code H	Key scan counter Data code L
R7	▨	▨
R8	Key scan data H For immediate setting (0FH)	Key scan data L For immediate setting (0H)
R9	Compare key, data H	Compare key, data L
RA	▨	Key return check counter
RB	For immediate setting (0CH)	For immediate setting (3H)
RC	Not used	Not used
RD	Not used	Not used
RE	Not used	Not used
RF	Address stack register	

▨ : Used for transmission processing

Table 5-3. RAM Map Used for Transmission Processing

	H (1)	L (0)
R0	Data pointer H Work area 1	Data pointer L
R1	▨	▨
R2	▨	▨
R3	▨	Continue flag
R4	Custom code H	Custom code L
R5	Custom code’ H	Custom code’ L
R6	Data code H	Data code L
R7	For immediate setting (0FH)	Key OFF check counter
R8	Data 1	Data 0
R9	Data 1 transmit counter H	Data 1 transmit counter L
RA	Work area 2	Transmit bit counter
RB	▨	▨
RC	Not used	Not used
RD	Not used	Not used
RE	Not used	Not used
RF	Address stack register	

▨ : Used for key input processing

Table 5-4. Description of RAM Usage (1/2)

<table>
<tr><th>Name</th><th>RAM</th><th>Description</th></tr>
<tr><td>Data pointer</td><td>R0</td><td>This pointer is used for indicating ROM addresses.
It is used when performing a table lookup for ROM data.</td></tr>
<tr><td>Chattering counter</td><td>R00</td><td>This counter is used to count the number of times chattering occurs during key input processing. 0CH is set to this counter as the initial value for a count of three. 0FH indicates completion of three times.</td></tr>
<tr><td>Work area 1</td><td>R10</td><td>This is a work area that is used to temporarily store data.</td></tr>
<tr><td>Confirmation key data</td><td>R1</td><td>This is used to store confirmation key data after ON chattering elimination processing has been completed.
If other confirmation key data has already been stored, it is compared with the key data (R4) when chattering elimination processing is completed. If the two sets of data match, the previous data is retained. If they do not match, the new key data (R4) is stored.</td></tr>
<tr><td>For immediate setting (1H)</td><td>R02</td><td>This is used to clear the high-order three bits or to set 1H.</td></tr>
<tr><td>For immediate setting (0EH)</td><td>R12</td><td>This is used to clear the lowest three bits or to set 0EH.</td></tr>
<tr><td>Continue flag</td><td>R03</td><td>During transmission processing, this flag is used to determine whether it is the first frame or a second or subsequent frame that is currently being transmitted.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0FH</td><td>Second or subsequent frame is being transmitted</td></tr>
<tr><td>0H</td><td>First frame is being transmitted</td></tr>
</table>
0FH: Set, 0H: Clear</td></tr>
<tr><td>K29 ON flag</td><td>R13</td><td>This flag is used to determine whether or not the K29 key (combination key) has been pressed.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0FH</td><td>K29 key has been pressed</td></tr>
<tr><td>0H</td><td>Key other than K29 key has been pressed</td></tr>
</table>
0FH: Set, 0H: Clear</td></tr>
<tr><td>Key data</td><td>R4</td><td>This is used to store the previous key data during ON chattering. This data is compared with the compare key data (R9). If they match, the previous data is retained. If they do not match, the new compare key data (R9) is stored.</td></tr>
<tr><td>Custom code</td><td>R4</td><td>This is used to store the custom code.</td></tr>
<tr><td>Custom code’</td><td>R5</td><td>This is used to store the custom code’.</td></tr>
<tr><td>Data code</td><td>R6</td><td>This is used to store the data code.</td></tr>
<tr><td>Key scan counter</td><td>R06</td><td>This counter is used to count the number of key scans. Since a count of eight is required, 8H (8 times) is set as the counter’s initial value.</td></tr>
<tr><td>Key ON flag</td><td>R16</td><td>This flag is used to determine whether or not a key has been pressed.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0FH</td><td>Key has been pressed</td></tr>
<tr><td>0H</td><td>Key has been released</td></tr>
</table>
0FH: Set, 0H: Clear</td></tr>
</table>

Table 5-4. Description of RAM Usage (2/2)

<table>
<tr><th>Name</th><th>RAM</th><th>Description</th></tr>
<tr><td>Key OFF check counter</td><td>R07</td><td>This counter is used to count the number of times key off status occurs during bit data transmission.
Although a count of ten is required, 5H is set as the initial value. When the value becomes 0FH (meaning ten continuous times of key OFF status), key OFF status is confirmed. If key OFF status is not determined at this time, the counter value is initialized.</td></tr>
<tr><td>For immediate setting (0FH)</td><td>R17</td><td>This is used to set various flags.</td></tr>
<tr><td>Key scan data</td><td>R8</td><td>This is used to store output data for key scanning.</td></tr>
<tr><td>Data 0
Data1</td><td>R08
R18</td><td>This is used to store the row addresses of time data (table data) for transmitting a bit data value of “0” or “1”.
Data 0: Bit data “0”
Data 1: Bit data “1”</td></tr>
<tr><td>For immediate setting (0H)</td><td>R08</td><td>This is used to set various flags.</td></tr>
<tr><td>For immediate setting (0FH)</td><td>R18</td><td>This is used for flag setting and data inversion.
The setting is “0FH”.</td></tr>
<tr><td>Compare key data</td><td>R9</td><td>This is used to store key data during ON chattering.</td></tr>
<tr><td>Data 1 transmit counter</td><td>R9</td><td>This counter is used to count the number of times bit data “1” is transmitted.</td></tr>
<tr><td>Key return check counter</td><td>R0A</td><td>This counter is used to calculate the key data based on the key return data.
Since a count of four is required for K_I input and a count of two is required for input of S_0 or S_1, a initial value of 0CH (for four times) or 0EH (for two times) is set.</td></tr>
<tr><td>Transmit bit counter</td><td>R0A</td><td>During bit transmission processing, this checks whether the required number of transmit bits are in the bit data stored in work area 2 (R1A).
The counter counts from one to four times.
<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>0FH</td><td>1-bit transmission (for transmitting leader code and stop bit)</td></tr>
<tr><td>0EH</td><td>2-bit transmission (not used)</td></tr>
<tr><td>0DH</td><td>3-bit transmission (not used)</td></tr>
<tr><td>0CH</td><td>4-bit transmission (for transmitting custom code, custom code’, data code, and $\overline{\text{data code}}$)</td></tr>
</table></td></tr>
<tr><td>Work area 2</td><td>R1A</td><td>This is used to store output data for bit transmissions.</td></tr>
<tr><td>For immediate setting (3H)</td><td>R0B</td><td>This is used for data judgments and for setting 3H.</td></tr>
<tr><td>For immediate setting (0CH)</td><td>R1B</td><td>This is used for data judgments and for setting 0CH.</td></tr>
</table>

5.4 Flag Maps

Table 5-5 shows flag operations during various types of processing.

Table 5-5. Flag Map

(1) Continue flag (R03)

Processing		R03
Initialization processing		Clear (0H)
Key input processing	When the key data used for ON chattering elimination processing differs from the confirmation key data from the previous ON chattering elimination processing.	Clear (0H)
Transmission processing	During frame space transmission of all frames	Judgment
	During frame space transmission of only one frame	Set (0FH)

Judgment: A judgment is made during this processing.

(2) K29 ON flag (R13)

Processing		R13
Initialization processing		Clear (0H)
Key input processing	When key is ON and flag has been set	Judgment
	When chattering has been completed 3 times and the confirmation key data is K29	Set (0FH)
	When chattering has been completed 3 times and the confirmation key data is not K29, nor is there a valid key combination	Clear (0H)
Transmission processing		–

Judgment: A judgment is made during this processing.
—: Not used

(3) Key ON flag (R16)

Processing		R16
Initialization processing		–
Key input processing	During RAM initialization	Clear (0H)
	Start of key return check	Judgment
	After key data calculation	Judgment
	During key return check or when there is key input	Set (0FH)
	After completion of key scanning (8 times)	Judgment
Transmission processing		×

Judgment: A judgment is made during this processing.
—: Not used, ×: Used by other application

[MEMO]

CHAPTER 6 PROGRAM DESCRIPTION

6.1 Initial Settings

Microcontrollers used in infrared remote control transmitters generally use batteries as their power source. However, the lighting of infrared LEDs requires a large current consumption. The abrupt change in the power supply voltage when infrared LEDs are being lit can cause sudden changes in the contents of RAM, ports, etc., which must be taken into consideration.

To prevent operation faults that may occur as a result of sudden changes in the contents of RAM, ports, etc., the program can be designed to reset initial settings after each transmission.

6.1.1 Description of processing

(1) Port settings and control register initialization

(a) $K_{I/O}$ port (P0)

This is an 8-bit I/O port that is used for key scan output.

This port's initial setting is FFH. All of the port's bits ($K_{I/O1}$ to $K_{I/O7}$) are set for high level output.

(b) Control register 0 (P3)

Tables 6-1 and 6-2 list the contents of control register 0.

The initial setting is 13H. The initial settings are shown in shaded areas in Tables 6-1 and 6-2.

Table 6-1. Control Register 0 (P3)

Bit		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
Name		–	–	DP (data pointer)		TCTL	CARY	MOD_1	MOD_0
				DP_9	DP_8				
Setting	0	Fixed as "0"	Fixed as "0"	▨ 0	0	▨ 1/1	▨ ON	See **Table 6-2**.	
	1			1	▨ 1	1/2	OFF		
After reset		0	0	0	0	0	0	1	1

b_0 and b_1 .... Specify the REM output's carrier frequency and duty factor.

b_2 .... Indicates presence/absence of carrier for frequency specified by b_0 and b_1.
"0" = ON (with carrier), "1" = OFF (no carrier, high level)

b_3 .... Changes the carrier frequency and the timer clock division ratio.
"0" = 1/1 (carrier frequency: values set to b_0 and b_1, timer clock: $f_X/8$)
"1" = 1/2 (carrier frequency: one half of values set to b_0 and b_1, timer clock: $f_X/16$)

Table 6-2. Time Clock and Carrier Frequency Settings

b_3	b_2	b_1	b_0	Timer clock	Carrier frequency (duty factor)
▨ 0	▨ 0	0	0	$f_X/8$	f_X (Duty 1/2)
		0	1		$f_X/8$ (Duty 1/2)
		1	0		$f_X/12$ (Duty 1/2)
		▨ 1	▨ 1		$f_X/12$ (Duty 1/3)
	1	×	×		No carrier (high level)
1	0	0	0	$f_X/16$	$f_X/2$ (Duty 1/2)
		0	1		$f_X/16$ (Duty 1/2)
		1	0		$f_X/24$ (Duty 1/2)
		1	1		$f_X/24$ (Duty 1/3)
	1	×	×		No carrier (high level)

b_4 and b_5 .... Specify the high-order two bits (DP_8 and DP_9) of the ROM data pointer.

Remarks 1. × : don't care

2. ▨ : Initial setting (13H)

3. f_X : System clock frequency

(c) Control register 1 (P4)

Table 6-3 lists the contents of control register 1.

The initial setting is 33H, which is shown in the shaded areas of the table.

Table 6-3. Control Register 1 (P4)

Bit		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
Name		–	–	K_I pull-down	S_0/S_1 pull-down	–	$S_1/\overline{\text{LED}}$ mode	$K_{I/O}$ mode	S_0 mode
Setting	0	Fixed as “0”	Fixed as “0”	OFF	OFF	Fixed as “0”	S_1	IN	OFF
	1			ON	ON		$\overline{\text{LED}}$	OUT	IN
After reset		0	0	1	0	0	1	1	0

b_0 .... Specifies the S_0 port’s input mode. “0” = OFF mode (high impedance), “1” = IN (input mode).

b_1 .... Specifies the $K_{I/O}$ port’s I/O mode. “0” = IN (input mode), “1” = OUT (output mode).

b_2 .... Specifies the $S_1/\overline{\text{LED}}$ port’s I/O mode. “0” = S_1 (input mode), “1” = $\overline{\text{LED}}$ (output mode)

b_4 .... Specifies presence/absence of pull-down resistor when S_0/S_1 port is in input mode. “0” = OFF (no pull-down), “1” = ON (pull-down).

b_5 .... Specifies presence/absence of pull-down resistor when K_1 port is in input mode. “0” = OFF (no pull-down), “1” = ON (pull-down).

Remarks 1. All pull-down resistors are automatically switched off during output mode and OFF mode.

2. ▨ : Initial setting (33H)

(2) Initialize RAM

The following RAM contents are cleared to “0”.

- Confirmation key data (R1)
- Continue flag (R03)
- K29 ON flag (R13)

(3) Set STOP mode

Table 6-4 lists the cancellation conditions for the HALT instruction.

The initial setting is 8H. When initialized, the (STOP mode) cancellation conditions are set as shown in the shaded areas of the table.

Table 6-4. Cancellation Conditions for HALT Instruction

HALT instruction operand value				Mode	Precondition for settings	Cancellation condition
b_3	b_2	b_1	b_0	Setting		
0	0	0	0	STOP	High-level output from all $K_{I/O}$ pins	High-level input via at least one K_I pin
	0	1	1	STOP	High-level output from all $K_{I/O}$ pins	High-level input via at least one K_I pin
	1	1	0	STOP[Note 1]	High-level output from $K_{I/O0}$ pin	High-level input via at least one K_I pin
1	Any combination of b_2, b_1, and b_0 above			STOP	[The following conditions in addition to the above conditions]	
					–	High-level input via at least one pin between S_0 and S_1[Note 2]
0/1	1	0	1	HALT	–	When timer's down counter reaches 0

Notes 1. When HALT #X110B is set, use the $K_{I/O0}$ pin and the K_I pin to configure a key matrix so that an internal reset is executed whenever a runaway (control loss) condition occurs.

2. S_0 and/or S_1 (at least one of these pins that are used to cancel standby mode) must be set to input mode (an internal reset will not be executed if both are set to output mode).

Cautions 1. An internal reset is executed if the HALT instruction is executed using operand values other than those specified above or when the precondition for HALT instruction execution has not been met.

2. If STOP mode is set when the timer's down counter has not yet reached "0" (i.e., when the timer is operating), all 10 bits of the enable flags for the timer's down counter and the timer output are cleared to zero and STOP mode is set.

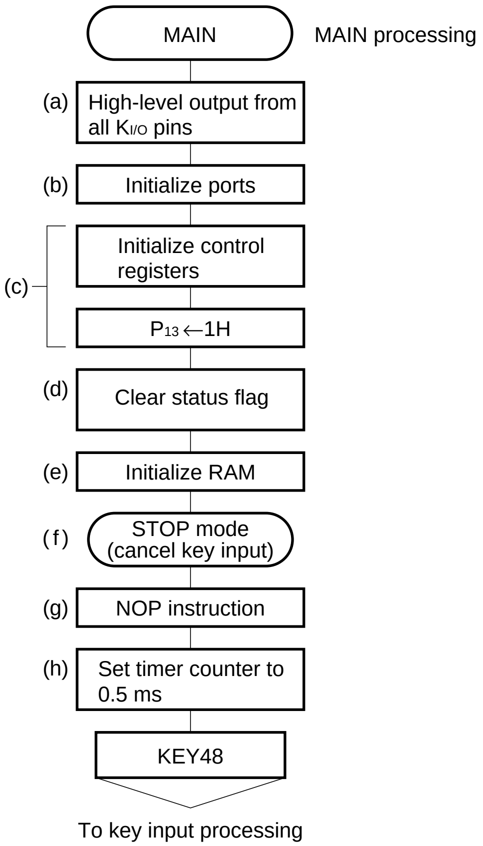
3. Specify a NOP instruction as the first instruction following cancellation of STOP mode.

Remark ▨ : Initial setting (8H)

(4) After cancellation of STOP mode, execute an NOP instruction, then initialize the timer.

The timer's initial setting is 1CH (= 0.5 ms).

6.1.2 Detailed flow chart



```
******** INITIALIZATION PROCESSING ********

        TIME05M         EQU     01CH               ;0.5ms
        DamyTime        EQU     512-1              ;9.00ms

;##########   P   U   B   L   I   C   ##########
        PUBLIC          MAIN
        ;
;##########   E   X   T   E   R   N   ##########
        ;

;##########   S    T    A    R    T   ##########
;***********************************************
;       Control Register Initialize
;***********************************************
MAIN:
        OUT     P0,#0FFH                        (a) Sets all key scan outputs (KI/O) to high level
        OUT     P4,#033H                        (b) Initializes ports
                                                    S0, S1:  Input mode, KI/O:  output mode
        OUT     P3,#013H                        (c) Initializes control registers 0 and 1 (P3 and P4)
                                                    With carrier, frequency:  fX/12, duty factor:  1/3
                                                    TCTL:  1/1, KI:  with pull-down resistor
                                                    S0, S1: with pull-down resistor
                                                    Data pointer:  DP8, DP9 = 01H (MSB of table lookup address)
        MOV     T,#DamyTime ⎫
                            ⎬..................... (d) Clears status flags
        STTS    #0101B      ⎭
;***********************************************
;       RAM Initialize Routine
;***********************************************  (e) Initializes RAM
        MOV     R1,#000H                            Confirmation key data (R1): 00H
        MOV     R3,#000H                            K29 ON flag (R13), continue flag (R03): 00H
        HALT    #008H                           (f) STOP mode:  Canceled by high-level input via KI, S0, and S1
        NOP                                     (g) NOP instruction
        MOV     T,#TIME05M                      (h) Initializes timer value
        ;                                           1CH = 0.5 ms
END
```

6.2 Key Input Processing

6.2.1 Description of processing

Key input processing includes chattering elimination processing, key scan processing, custom code generation processing, and key data generation processing.

(1) Chattering elimination processing

When switching to key ON or key OFF status, an unstable condition called chattering (ON chattering or OFF chattering) exists until the key signal is stabilized as a high level or low level signal (see Figure 6-1). Since key input during this unstable condition is also unstable, the program must provide a means of eliminating chattering.

Figure 6-1. Chattering of Key Input Signal

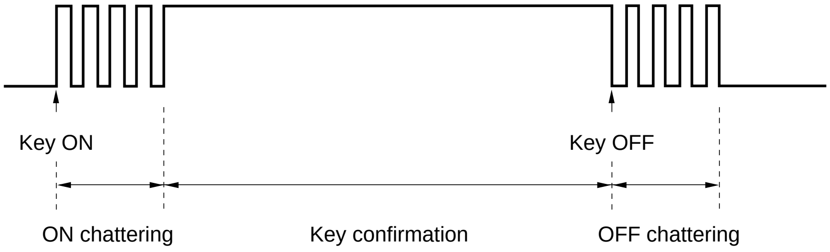
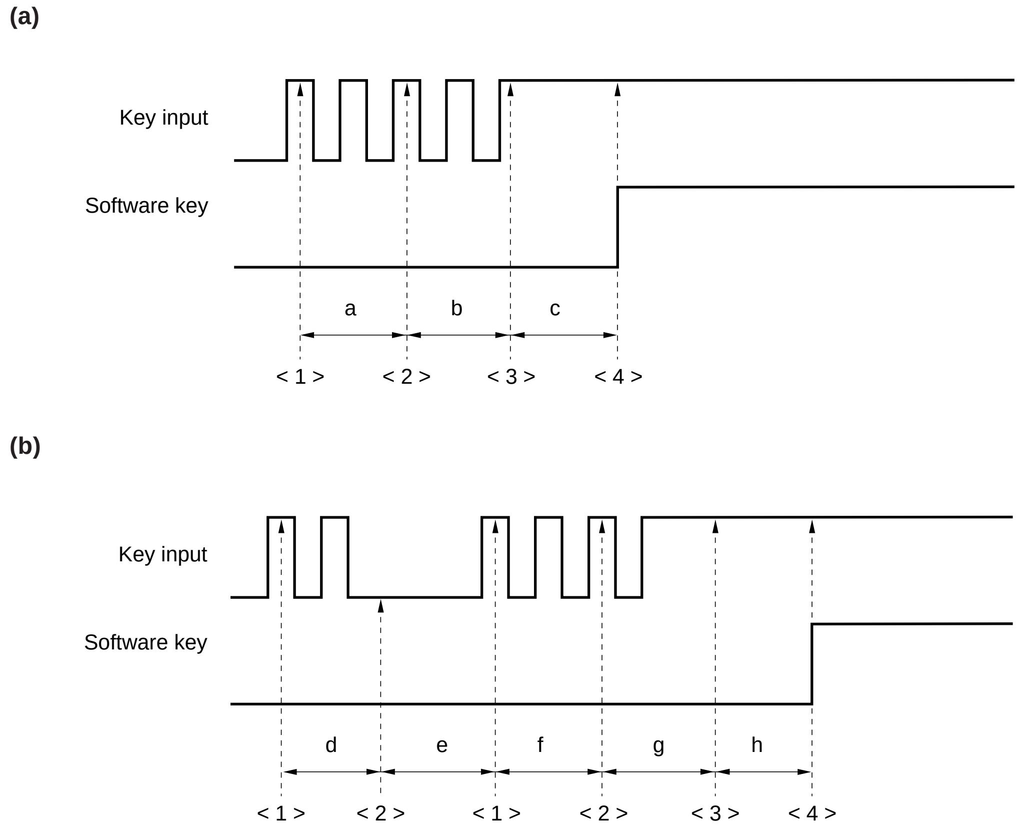


Figure 6-2 illustrates an example of chattering elimination processing (key ON judgment example).
At point <1>, key ON status is detected when STOP mode is canceled, and key input is checked during a set time period (from <1> to <4>: 9.00 ms × 3).
In part (a) of the figure, key ON status is detected at all check points, so the key signal is judged to be at high level at the software key's <4>.
The wait period (a + b + c) after key ON status is detected at point <1> and before the software judges the signal's high level status is called the chattering elimination period. In part (a), the chattering elimination period is "a + b + c" since the key input status is always ON when checked. In part (b), the chattering elimination period is "d + e + f + g + h".

Figure 6-2. Key ON Judgment when Chattering Occurs



(2) Key scan processing

Key scan processing is described in sections (a) and (b) below.

(a) Key matrix

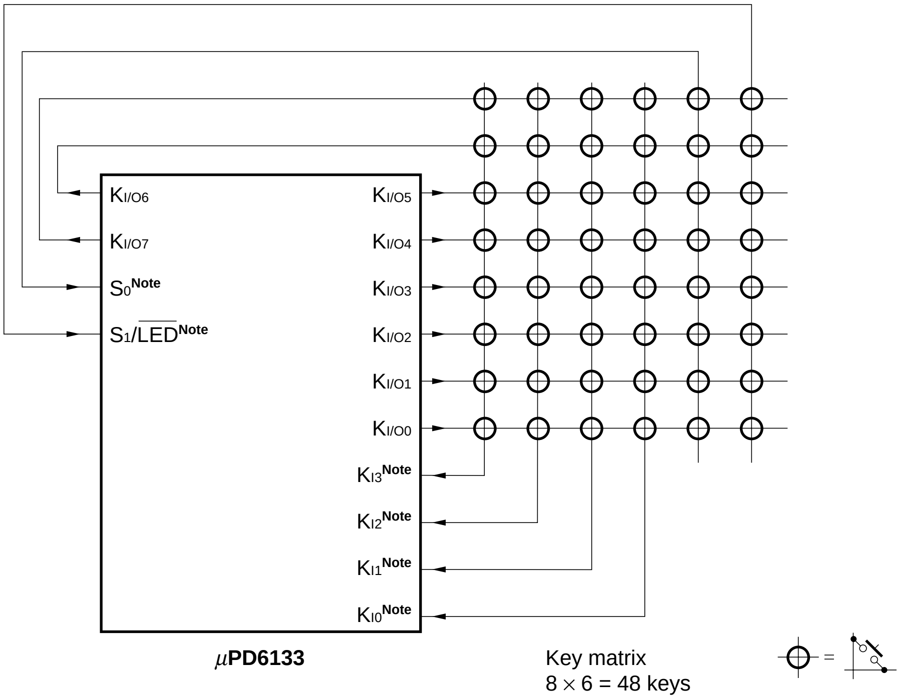
Figure 6-3 shows an example of a 48-key key matrix.

In this example, $K_{I/O0}$ to $K_{I/O7}$ are output ports that output key scan signals.

These signals are captured (as key return signals) via input ports comprised of K_{I0} to K_{I3}, S_0 and S_1.

Since the program connects these key return signal input ports (K_{I0} to K_{I3}, S_0 and S_1) to an internal pull-down resistor, low-level signals are input when no keys are being pressed.

Figure 6-3. 48-key Key Matrix



Note The program is set for “internal pull-down resistor”.

(b) Key scan

To judge which of the keys in the 48-key key matrix is being pressed, the STTS instruction is used to check for high-level signal input via K_{I0} to K_{I3}, S_0, and S_1.

Next, the key scan signal's output ports ($K_{I/O0}$ to $K_{I/O7}$) are set to high level one at a time starting from $K_{I/O0}$ to determine which output among $K_{I/O0}$ to $K_{I/O7}$ corresponds to the detected input.

During the key scan processing part of the program, the key scan counter (key source position) and key return check counter (key return position) are used to detect the key data (key position). Figure 6-5 shows the correspondence between key data and the values of these two counters.

For description of the key data's bit configuration in data memory, see "(4) (a) Bit configuration of key data".

When performing a key scan, factors such as stray capacitance in the keyboard and key source delay due to line impedance must be taken into consideration. Therefore, this program waits for about 100 μs (six steps when at 455 kHz) following high-level output before capturing the key input.

Figure 6-4. Key Matrix

$K_{I/O7}$	K29	K30	K31	K32	K47	K48
$K_{I/O6}$	K25	K26	K27	K28	K45	K46
$K_{I/O5}$	K21	K22	K23	K24	K43	K44
$K_{I/O4}$	K17	K18	K19	K20	K41	K42
$K_{I/O3}$	K13	K14	K15	K16	K39	K40
$K_{I/O2}$	K9	K10	K11	K12	K37	K38
$K_{I/O1}$	K5	K6	K7	K8	K35	K36
$K_{I/O0}$	K1	K2	K3	K4	K33	K34
	K_{I3}	K_{I2}	K_{I1}	K_{I0}	S_1	S_0

Figure 6-5. Key Data (Key Position)

Key scan counter (R06) ↓						
F	K29	K30	K31	K32	K47	K48
E	K25	K26	K27	K28	K45	K46
D	K21	K22	K23	K24	K43	K44
C	K17	K18	K19	K20	K41	K42
B	K13	K14	K15	K16	K39	K40
A	K9	K10	K11	K12	K37	K38
9	K5	K6	K7	K8	K35	K36
8	K1	K2	K3	K4	K33	K34
Key return check counter (R0A) →	C	D	E	F	E	F

(3) Custom code generation processing

This program is set to output 0AH as the custom code and F5H as the custom code'.

Specifically, 50H is set to data memory R4 as the custom code and AFH is set to R5 as the custom code'.

Caution In the NEC format, the LSB is transmitted first, so values are set in opposite order to the bit string.

(4) Key data generation processing

Key data generation processing is described in (a) and (b) below.

(a) Bit configuration of key data

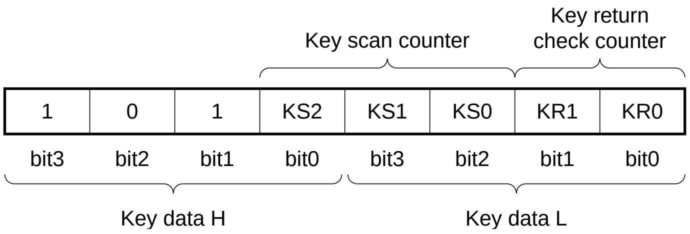
Key data consists of eight bits. Each bit indicates a key source and key return status.

To accommodate the eight types from $K_{I/O0}$ to $K_{I/O7}$, the key scan counter uses the LSB of key data H and the high-order two bits of key data L when input is via K_{I0} to K_{I3}, and uses the high-order three bits of key data L when input is via S_0 and S_1. The key return check counter accommodates the four types from K_{I0} to K_{I3} by using the low-order two bits of key data L and accommodates the two types S_1 and S_0 by using the LSB of key data L.

When the first key input is detected, 0CH is set to key data H if the input is via S_0 and S_1. This key data H is also used to determine the format used to generate the key data, as shown in Figures 6-6 and 6-7.

As is described in "(b) Data code" below, this key data is also used as an address for table reference. Therefore, the key data is configured as shown in Figures 6-6 and 6-7.

Figure 6-6. Bit Configuration of Key Data when Input is via K_{I0} to K_{I3}



1	0	1	KS2	KS1	KS0	KR1	KR0
bit3	bit2	bit1	bit0	bit3	bit2	bit1	bit0

Figure 6-7. Bit Configuration of Key Data when Input is via S_0 and S_1

				Key scan counter			Key return check counter
1	1	0	0	KS2	KS1	KS0	KR0
bit3	bit2	bit1	bit0	bit3	bit2	bit1	bit0
Key data H				Key data L			

(b) Data code

The data code can be calculated by using the key data obtained via key input processing (as shown in Figures 6-6 and 6-7) as an address for table lookup.

The table lookup method uses the contents of ROM as a transmit code by setting 1H to the high-order four bits of control register 0 (P3) and by setting the key data to the data pointer (see Figure 6-8).

Figure 6-8. Configuration of Data Pointer

Control register 0 (P3)

b7	b6	b5	b4
Fixed as "0"	Fixed as "0"	DP_9	DP_8

Data pointer (DP)

R_{10}				R_{00}			
DP_7	DP_6	DP_5	DP_4	DP_3	DP_2	DP_1	DP_0

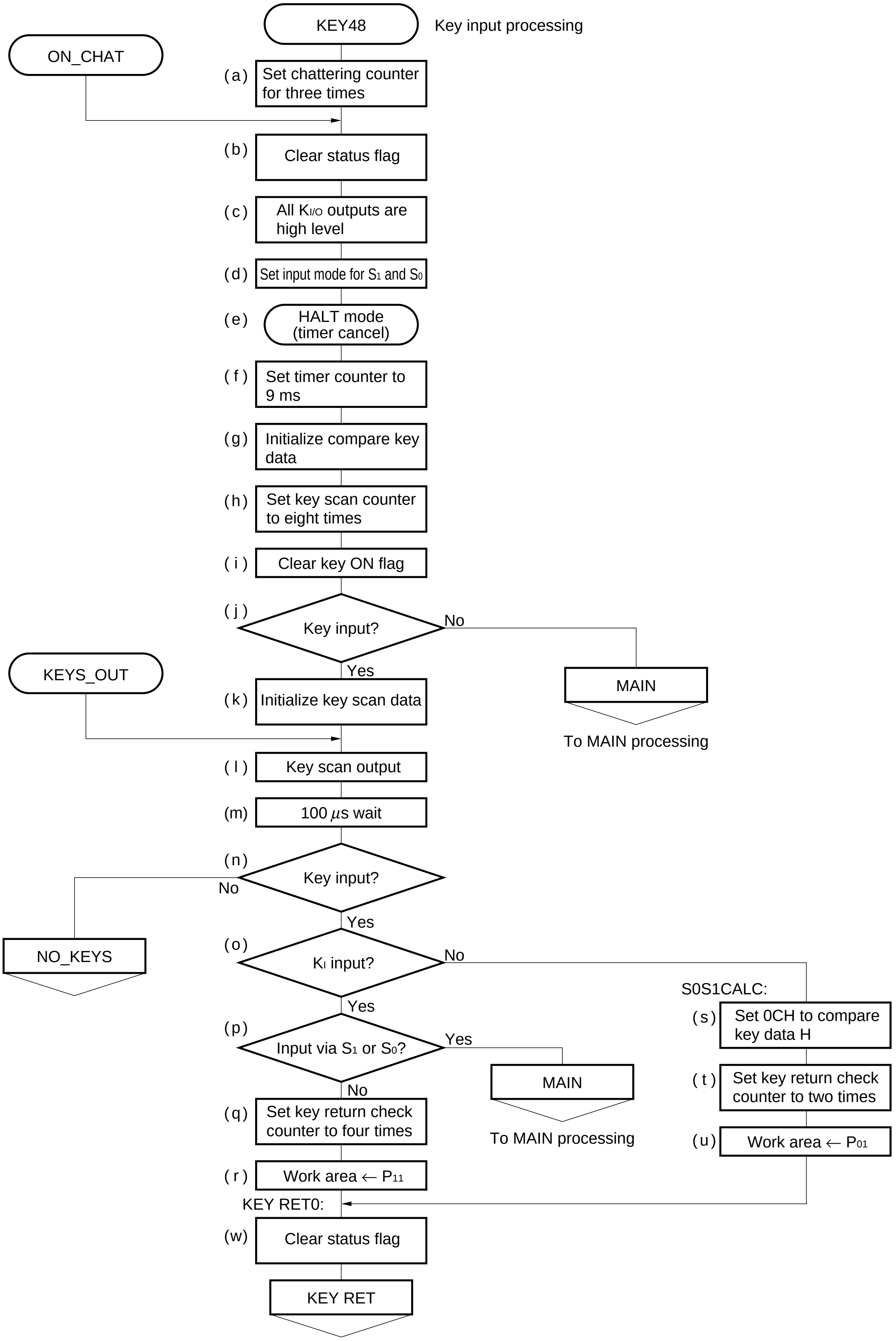
Example When key position is K26 (key input K_{I2})

1. Key scan counter (R06) = EH
   Key return check counter (R0A) becomes DH (see Figure 6-5).
2. The low-order three bits of the key scan counter and the low-order two bits of the key return check counter are used to configure the key data, as was shown in Figure 6-6.
   - Key scan counter 1 $\underline{1\ 1\ 0}$
   - Key return check counter 1 1 $\underline{0\ 1}$
   - Bit configuration of key data 1 0 1 $\underline{1\ 1\ 0}$ $\underline{0\ 1}$
3. Set key data to data pointer.
   - Configuration of data pointer $\underline{0\ 0\ 0\ 1}$ $\underline{1\ 0\ 1\ 1}$ $\underline{1\ 0\ 0\ 1}$ → $\underline{1}$ $\underline{B}$ $\underline{9}$ H

The following table reference addresses are generated when a key is generated using one of the bit configurations shown in Figures 6-6 and 6-7. As mentioned above, 1H is set to the high-order four bits of control register 0 (P3).

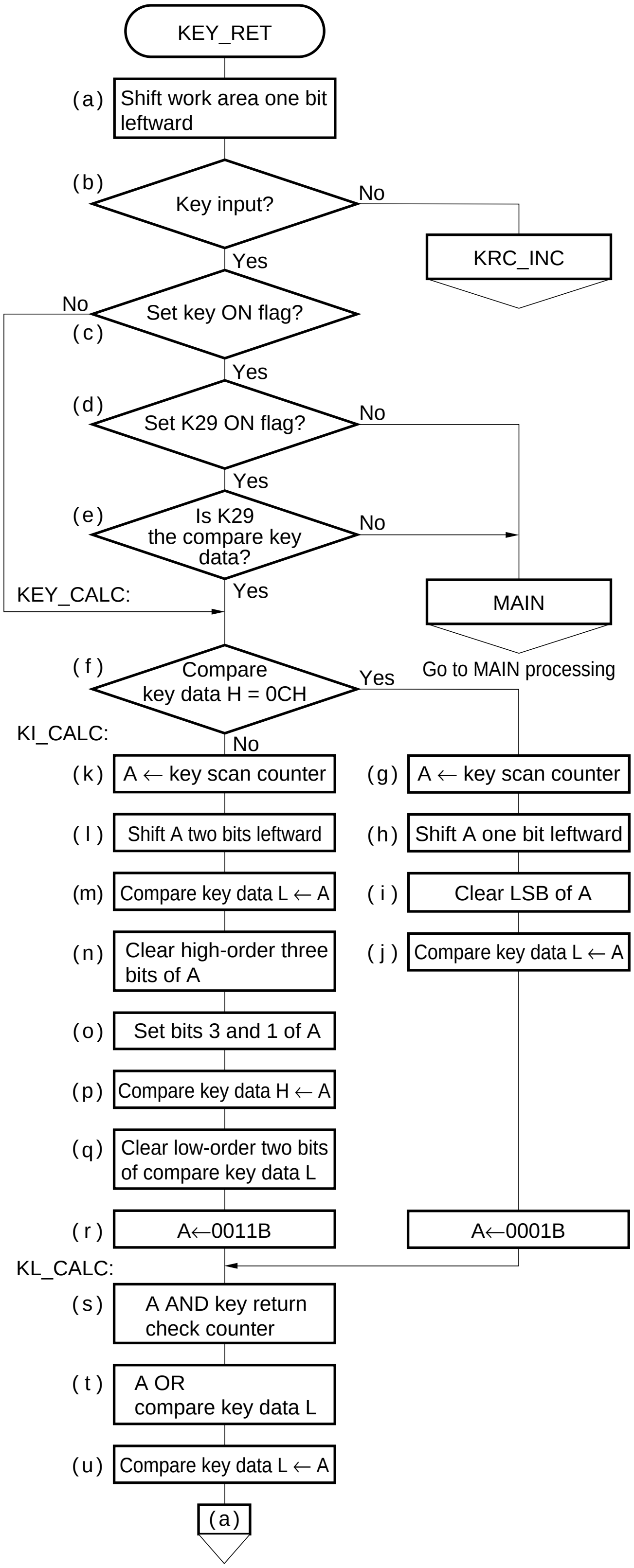
- Key input: K_{I0} to K_{I3}, key position: K1 to K32
  ... reference addresses = 1A0H to 1BFH
- Key input: S_0 and S_1, key position: K33 to K48
  ... reference addresses = 1C0H to 1CFH
- Key combination (K29 + K30, K29 + K31, K29 + K32)
  When a key combination has been confirmed, 9H is set to key data H.
  K29 + K30 ... reference address = 19DH
  K29 + K31 ... reference address = 19EH
  K29 + K32 ... reference address = 19FH

6.2.2 Detailed flow chart



```
******** KEY INPUT PROCESSING ********
        TIME9M  EQU     1FFH                       ;9.00ms(9.002ms)
        CUSTM1  EQU     050H                       ;Custom Code  = 0AH
        CUSTM2  EQU     0AFH                       ;Custom Code’ = F5H

;##########   P   U   B   L   I   C   ##########
        PUBLIC          KEY48
        ;

;##########   E   X   T   E   R   N   ##########
        EXTRN           MAIN                       ;MAIN Routine
        ;

;##########   S    T    A    R    T   ##########
KEY48:
        MOV     R0,#00CH                           (a) Sets (0CH to) chattering counter (R00)
                                                       (3 count end at 0FH.)
;************************************************
;               ON Chattering
;************************************************
ON_CHAT:
        STTS    #0101B                             (b) Clears status flag
        OUT     P0,#0FFH                           (c) Sets all KI/O outputs to high level
        OUT     P4,#033H                           (d) Sets input mode for S1 and S0
        HALT    #005H                              (e) HALT mode:  timer is canceled when count reaches 00H
        MOV     T,#TIME9M                          (f) Sets timer counter.  (9 ms = 1FFH)
        MOV     R9,#000H                           (g) Initializes (sets to “00H”) compare key data (R9)
        MOV     R6,#008H                           (h) (i) Sets 8H to key scan counter (R06) and sets 0H to clear
                                                       key ON flag (R16)
        STTS    #1110B                             (j) Determines when there is key input (via KI, S0, or S1)
        JNF     MAIN                                   If no key input, processing branches to MAIN
        MOV     R8,#001H                           (k) Initializes (sets 01H to) key scan data (R8)
;************************************************
;                  Key Scan
;************************************************
KEYS_OUT:
        MOV     A,R08    ⎫
        OUT     P00,A    ⎬........................ (l) Outputs key scan data (P0)
        MOV     A,R18    ⎪
        OUT     P10,A    ⎭
    ;++++++++++++++++++++++++++++++++++
    ;++ 100-µs wait before key input ++
    ;++++++++++++++++++++++++++++++++++
        MOV     R2,#0E1H ⎫                         Remark  R2 is used to set immediate data (R12:  0EH, R02:  1H).
        MOV     RB,#0C3H ⎪                         Remark  RB is used to set immediate data (R1B:  0CH, R0B:  3H).
        NOP              ⎬........................ (m) 100-µs wait before performing key input check
        NOP              ⎪
        NOP              ⎪
        NOP              ⎭
```

```
;************************************************
;               Key Return Check
;************************************************
        STTS    #1011B                                  (n) Determines whether or not there is key input.
        JNF     NO_KEYS                                     If no key input, processing branches to NO_KEYS.
        STTS    #0011B                                  (o) Determines whether input is via KI or S0 and S1
        JNF     S0S1CALC                                    If via S0 and S1, processing branches to S0S1CALC.
;********************************
;       KI Data Calculate
;********************************
        IN      A,P01  ┐
        RL      A      ├ ·························  (p) When there is input via KI, it determines whether there is
        JC      MAIN   │                                    also input via S0 and S1.
        RL      A      │                                    If there is also input via S0 or S1, a key error is detected
        JC      MAIN   ┘                                    and it goes to MAIN processing.
        MOV     A,R1B                                   (q) Initializes the key return check counter (R0A).
        MOV     R0A,A                                       0CH = 4 times (In this case, R1B = 0CH).
        IN      A,P11                                   (r) The pin status (P11) of KI is transferred to the accumulator.
        JMP     KEY_RET0
;********************************
;       S0,S1 Data Calculate
;********************************
S0S1CALC:
        MOV     R9,#0C0H                                (s) A judgment value (0CH) for K33 to K48 (for input via S0
                                                            and S1) is set to the compare key data H (R19).

        MOV     A,R12
        MOV     R0A,A                                   (t) Initializes the key return check counter (R0A).
                                                            0EH = 2 times (in this case, R12 = 0EH).

        IN      A,P01                                   (u) The pin status (P01) of S0 or S1 is transferred to the
                                                            accumulator.

KEY_RET0:
        MOV     R10,A                                   Remark  The accumulator's value (P00 or P01) is stored in the work area.
        STTS    #0101B                                  (v) Clears status flag.
```


KEY_RET
(a) Shift work area one bit leftward
(b) Key input?
No
KRC_INC
Yes
No
Set key ON flag?
(c)
Yes
(d) Set K29 ON flag?
No
Yes
(e) Is K29 the compare key data?
No
KEY_CALC:
Yes
MAIN
Go to MAIN processing
(f) Compare key data H = 0CH
Yes
KI_CALC:
No
(k) A ← key scan counter
(g) A ← key scan counter
(l) Shift A two bits leftward
(h) Shift A one bit leftward
(m) Compare key data L ← A
(i) Clear LSB of A
(n) Clear high-order three bits of A
(j) Compare key data L ← A
(o) Set bits 3 and 1 of A
(p) Compare key data H ← A
(q) Clear low-order two bits of compare key data L
(r) A←0011B
A←0001B
KL_CALC:
(s) A AND key return check counter
(t) A OR compare key data L
(u) Compare key data L ← A
(a)

```
KEY_RET:
        MOV     A,R10  }
        RLZ     A      }............................ (a) Shifts work area 1 (R10) one bit leftward
        MOV     R10,A  }
        JNC     KRC_INC                              (b) Determines whether or not key input exists (CY = 1).
    ;**  Key input exists  **                            If there is no key input, processing branches to KRC_INC.
        MOV     A,R16     }
        SCAF              }......................... (c) Determines whether or not key has been pressed (combination key).
        JNC     KEY_CALC  }                              If no combination key (only one key) has been pressed,
                                                         processing branches to KEY_CALC.
        MOV     A,R13  }
        SCAF           }............................ (d) Determines whether or not the K29 ON flag has been set.
        JNC     MAIN   }                                 If it has been cleared, processing branches to MAIN.
        MOV     A,#0100B  }
        XRL     A,R19     }
        SCAF              }
        JNC     MAIN      }......................... (e) Determines whether or not the compare key data (R9)
        MOV     A,R09     }                              matches K29's key data.
        XRL     A,R0B     }                              If they do not match, processing branches to MAIN.
        SCAF              }
        JNC     MAIN      }
;*******************************
;       Key Data Calculate
;*******************************
KEY_CALC:
        MOV     A,R0B     }                       { (f) Determines whether or not the pressed key is between
        XRL     A,R19     }                       {     K33 and K48 (input via S0 or S1).
        SCAF              } ......................{     If a key between K1 and K32 has been pressed,
        JNC     KI_CALC   }                       {     processing branches to KI_CALC.

                                                                                   Accumulator values
;**  S0,S1 data calculation  **
        MOV     A,R06                                (g) Transfers key scan counter (R06)  | KS3 | KS2 | KS1 | KS0 |
                                                         to accumulator.
        RL      A                                    (h) Shifts accumulator one bit        | KS2 | KS1 | KS0 | KS3 |
                                                         leftward.
        ANL     A,R12                                (i) Clears the accumulator's LSB.     | KS2 | KS1 | KS0 |  0  |
                                                         (In this case, R12 = 0EH).
        MOV     R09,A                                (j) Stores accumulator values in                         R09
                                                         compare key data L (R09).
        MOV     A,R02                                (k) Sets 0001B to accumulator.        |  0  |  0  |  0  |  1  |
                                                         (In this case, R02 = 1H).
        JMP     KL_CALC                                  To KL_CALC
        ;
;**  KI data calculation  **
KI_CALC:
        MOV     A,R06                                (l) Transfers key scan data (R06) to  | KS3 | KS2 | KS1 | KS0 |
        RL      A  }                                     accumulator.
                   } ............................... (m) Shifts accumulator two bits leftward | KS1 | KS0 | KS3 | KS2 |
        RL      A  }
        MOV     R09,A                                (n) Stores accumulator values in                         R09
                                                         compare key data L (R09).
        ANL     A,R02                                (o) Clears the high-order three bits of the |  0  |  0  |  0  | KS2 |
                                                         accumulator.  (In this case, R02 = 1H)
        ORL     A,#1010B                             (p) Sets bit 3 and bit 1 of accumulator.  |  1  |  0  |  1  | KS2 |

        MOV     R19,A                                (q) Stores accumulator values in                         R19
                                                         compare key data H (R19).
        MOV     A,R09  }
        ANL     A,R1B  }............................ (r) Clears the low-order two bits of  | KS1 | KS0 |  0  |  0  |
        MOV     R09,A  }                                 the compare key data L (R1B).
                                                         (In this case, R1B = 0CH)                            R09
        MOV     A,R0B                                (s) Sets 0011B to accumulator.        |  0  |  0  |  1  |  1  |
                                                         (In this case, R0B = 3H)
```

Accumulator values

```
KL_CALC:
        ANL     A,R0A
```

(t) For K_I input, the low-order two bits of the key return counter (R_{0A}) is stored in the accumulator. For S_0 or S_1 input, the LSB is stored in the accumulator.

K_I input

0	0	KR1	KR0

S_0 or S_1 input

0	0	0	KR0

```
        ORL     A,R09
```

(u) OR processing of accumulator values and compare key data L (R_{09}).

K_I input

KS1	KS0	KR1	KR0

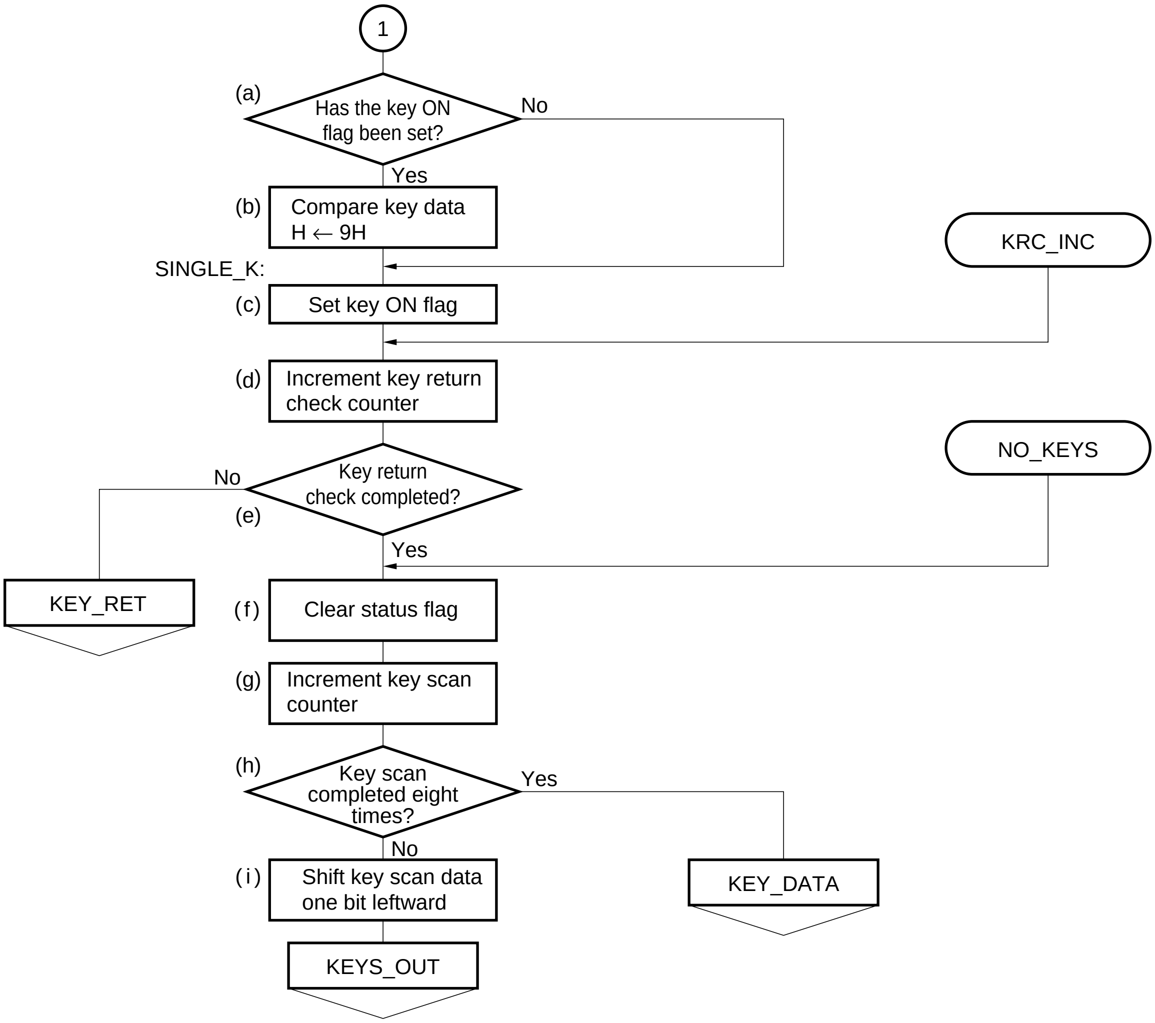
S_0 or S_1 input

KS2	KS1	KS0	KR0

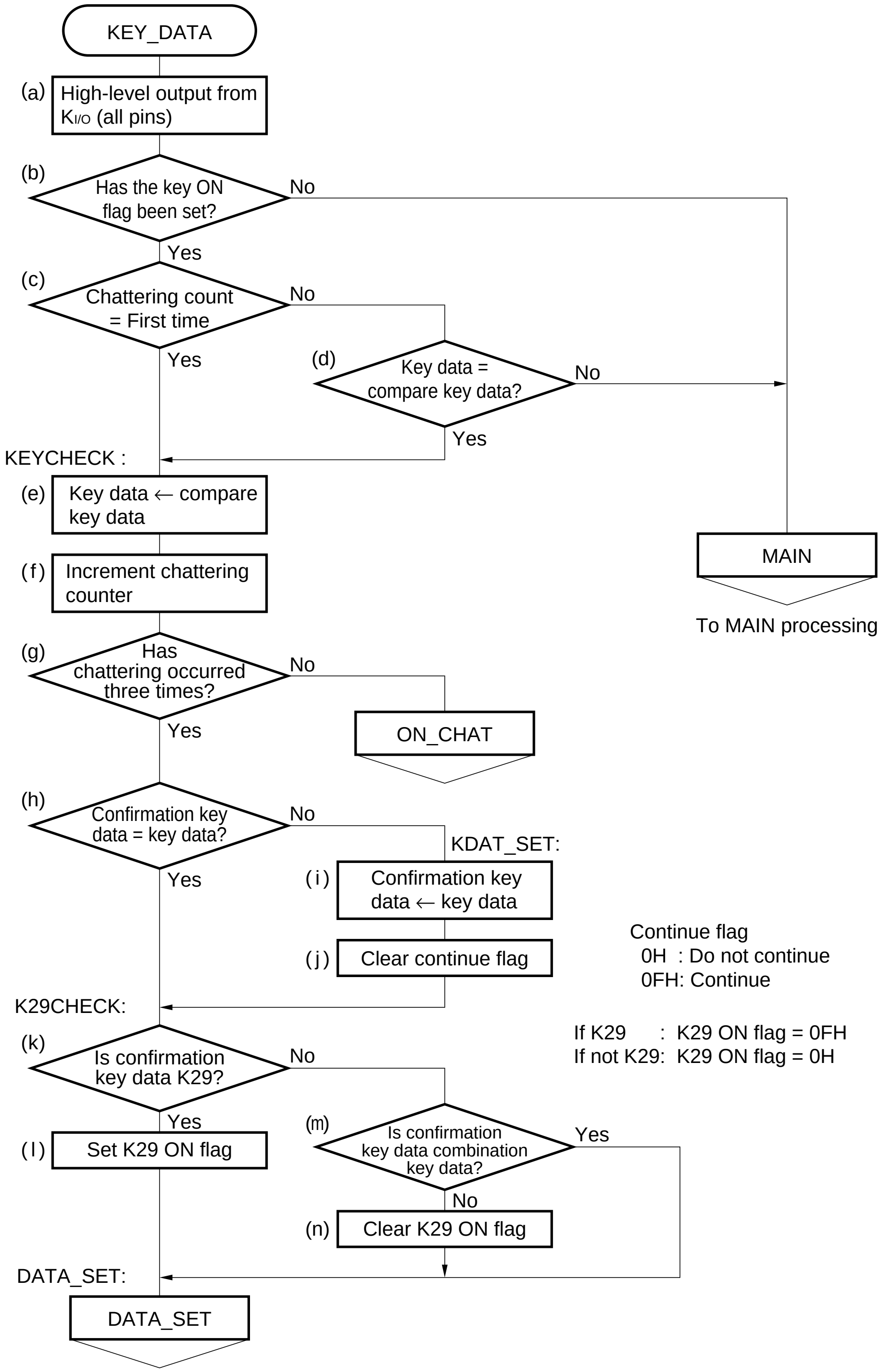
```
        MOV     R09,A
```

(v) Accumulator data is stored in compare key data L (R_{09}).

R_{09}

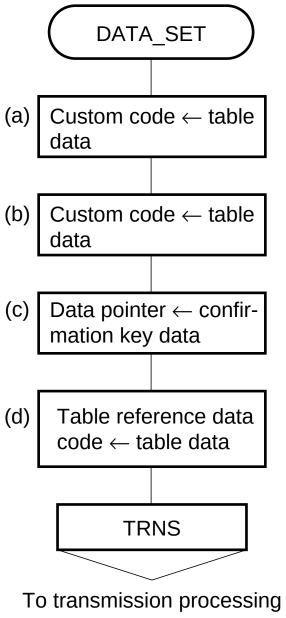

1
(a)
Has the key ON flag been set?
No
Yes
(b)
Compare key data
H ← 9H
SINGLE_K:
(c)
Set key ON flag
KRC_INC
(d)
Increment key return check counter
NO_KEYS
No
Key return check completed?
(e)
Yes
KEY_RET
(f)
Clear status flag
(g)
Increment key scan counter
(h)
Key scan completed eight times?
Yes
No
(i)
Shift key scan data one bit leftward
KEY_DATA
KEYS_OUT

```
   ;**  Key combination check  **
        MOV     A,R16     |
        SCAF              }........................ (a) Determines whether or not the key ON flag (R16) has been set.
        JNC     SINGLE_K  |                             If it has been cleared, processing branches to SINGLE_K.
        MOV     A,#09H    |
                          }........................ (b) Sets 9H to compare key data H (R19).
        MOV     R19,A     |
SINGLE_K:
        MOV     A,#0FH    |
                          }........................ (c) Sets the key ON flag (R16)
        MOV     R16,A     |                             (at this point, the accumulator value is 0FH).
KRC_INC:
        MOV     A,R0A     |
        INC     A         }........................ (d) Increments the key return check counter (R0A).
        MOV     R0A,A     |
        JNC     KEY_RET                             (e) Determines whether or not the key return check has been
                                                        completed four times.
                                                        If it has not, processing branches to KEY_RET.
NO_KEYS:
        STTS    #0101B                              (f) Clears status flag.
        MOV     A,R06     |
        INC     A         }........................ (g) Increments key scan counter (R06).
        MOV     R06,A     |
        JC      KEY_DATA                            (h) Determines whether or not the key scan has been
        MOV     A,R18     |                             completed eight times.
        RL      A         |                             If it has been completed, processing branches to KEY_DATA.
        MOV     R18,A     |
        MOV     A,R08     |
        RL      A         }........................ (i) Shifts the key scan data (R8) one bit leftward.
        MOV     R08,A     |
        JNC     KEYS_OUT  |
        MOV     R8,#010H  |
        JMP     KEYS_OUT
```


KEY_DATA
(a) High-level output from KI/O (all pins)
(b) Has the key ON flag been set?
No
Yes
(c) Chattering count = First time
No
Yes
(d) Key data = compare key data?
No
Yes
KEYCHECK :
(e) Key data ← compare key data
(f) Increment chattering counter
MAIN
To MAIN processing
(g) Has chattering occurred three times?
No
Yes
ON_CHAT
(h) Confirmation key data = key data?
No
Yes
KDAT_SET:
(i) Confirmation key data ← key data
(j) Clear continue flag
Continue flag
0H : Do not continue
0FH: Continue
K29CHECK:
If K29 : K29 ON flag = 0FH
If not K29: K29 ON flag = 0H
(k) Is confirmation key data K29?
No
Yes
(l) Set K29 ON flag
(m) Is confirmation key data combination key data?
Yes
No
(n) Clear K29 ON flag
DATA_SET:
DATA_SET

```
;***********************************************
; Transmit Key Data = ON Chattering Key Data ??
;***********************************************
KEY_DATA:
        OUT     P0,#0FFH                                  (a) Sets all key scan output (via KI/O) as high-level output.
        MOV     R8,#0F0H                                  Remark R8 is used for setting immediate data (R16: 0FH, R08: 0H).
        MOV     A,R16          ┐
        SCAF                   ├ ........................ (b) Determines whether or not the key ON flag (R16) has been set.
        JNC     MAIN           ┘                              If it has been cleared, processing branches to MAIN.
        MOV     A,R00          ┐
        XRL     A,R0B          ├ ........................ (c) Determines whether or not chattering processing is occurring
       SCAF                    │                              for the first time (0CH).  (In this case, R00 = #0011B.)
        JC      KEYCHECK       ┘                              If it is the first time, processing branches to KEYCHECK.
        MOV     A,R14          ┐
        XRL     A,R18          │                          Remark In this case, R16 = 0FH.
        XRL     A,R19          │
        SCAF                   │
        JNC     MAIN           ├ ........................ (d) Determines whether or not the key data (R4) matches the
        XRL     A,R04          │                              compare key data (R9).
        XRL     A,R09          │                              (In this case, accumulator value = 0FH.)
        SCAF                   │                              If they do not match, processing branches to MAIN.
        JNC     MAIN           ┘
KEYCHECK:
        MOV     A,R19          ┐
        MOV     R14,A          ├ ........................ (e) Stores compare key data (R9) in key data (R4).
        MOV     A,R09          │
        MOV     R04,A          ┘
        MOV     A,R00          ┐
        INC     A              ├ ........................ (f) Increments the chattering counter (R00).
        MOV     R00,A          ┘
        SCAF
        JNC     ON_CHAT                                   (g) Determines whether or not chattering processing has
                                                              been completed three times.
                                                              If it has not been completed three times, processing
;**  Key High Check  **                                       branches to ON_CHAT.
        XRL     A,R11          ┐
        XRL     A,R14          │                          Remark In this case, accumulator value = 0FH.
        SCAF                   │
        JNC     KDAT_SET       │
;**  Key Low Check  **         ├ ........................ (h) Determines whether or not the confirmation key data (R1)
        XRL     A,R01          │                              matches the key data (R4).
        XRL     A,R04          │                              (In this case, accumulator value = 0FH.)
        SCAF                   │                              If they do not match, processing branches to KDAT_SET.
        JC      K29CHECK       ┘                              If they match, processing branches to K29CHECK.
KDAT_SET:
        MOV     A,R14          ┐
        MOV     R11,A          ├ ........................ (i) Stores key data (R4) in confirmation key data (R1).
        MOV     A,R04          │
        MOV     R01,A          ┘
        MOV     A,R08                                     (j) Clears continue flag (R03).  (In this case, R08 = 0H).
        MOV     R03,A

K29CHECK:
        MOV     A,#0100B       ┐
        XRL     A,R11          │
        SCAF                   │
        JNC     DATA_K_CHK     ├ ........................ (k) Determines whether or not K29 has been pressed.
        MOV     A,R01          │                              (In this case, R0B = 3H).
        XRL     A,R0B          │                              If a key other than K29 has been pressed, processing
        SCAF                   │                              branches to K29_FLG.
        JNC     K29_FLG        ┘
```

```
DBL_K_CHK:
        MOV     A,#06H  ┐
        XRL     A,R11   ├······················ (l)  Determines whether or not the confirmation key data is
        SCAF            │                            combination key data.
        JC      DATA_SET┘                            If it is combination key data, processing branches to DATA_SET.

        MOV     A,R08                           (m)  Sets 0H to the accumulator.  (In this case, R08 = 0H.)
K29_FLG:
        MOV     R13,A                           (n)  Clears the K29 ON flag
                                                     (in this case, the accumulator value = 0H).
                                                (o)  Sets the K29 ON flag
                                                     (in this case, the accumulator value = 0FH).
```


DATA_SET
(a)
Custom code ← table data
(b)
Custom code ← table data
(c)
Data pointer ← confirmation key data
(d)
Table reference data code ← table data
TRNS
To transmission processing

```
;***********************************************
;               Transmit Data Set
;***********************************************
DATA_SET:
    ;**  Custom Code  **
        MOV     R4,#CUSTM1                            (a)  Stores custom code (R4) in table data.

    ;**  Custom Code'  **
        MOV     R5,#CUSTM2                            (b)  Stores custom code (R5) in table data.

    ;**  Data Code **
        MOV     A,R11   
        MOV     R10,A    } ·························  (c)  Stores confirmation key data (R1) in data pointer (R0).
        MOV     A,R01   
        MOV     R00,A   
        MOV     R6,@R0                                (d)  Stores data code (R6) in table data.
END
```

6.3 Transmission Processing

Transmission processing transmits custom code, custom code', data code, and $\overline{\text{data code}}$ stored in the data memory, using the NEC-R format that was described in **CHAPTER 2 TRANSMISSION WAVEFORM**.

The transmission method uses a data table to store transmission times sets the timer counter via a table reference operation. Afterward, it enters HALT mode (cancellation condition: TIMER) to enable transmission.

A method such as that shown in Figure 6-10 is used to ensure correct and simple time management. Immediately after cancellation of HALT mode, a similar method is used to set the transmission time data for the next transmission to the timer counter so that the transmission operation can be performed during a set time between HALT modes.

When carrier output is set to ON (hereafter, this status is called "H"), the timer counter's MSB (output control bit) is set. When carrier output is set to OFF (hereafter, this status is called "L"), the timer counter's MSB (output control bit) is cleared.

Figure 6-9 shows the timer counter's configuration.

Figure 6-9. Configuration of Timer Counter

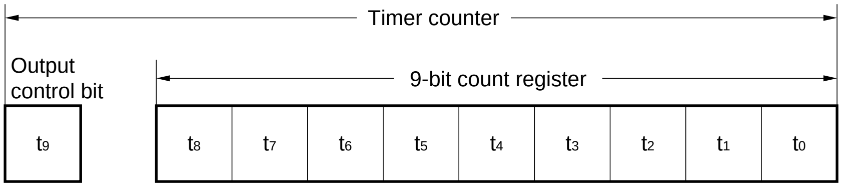
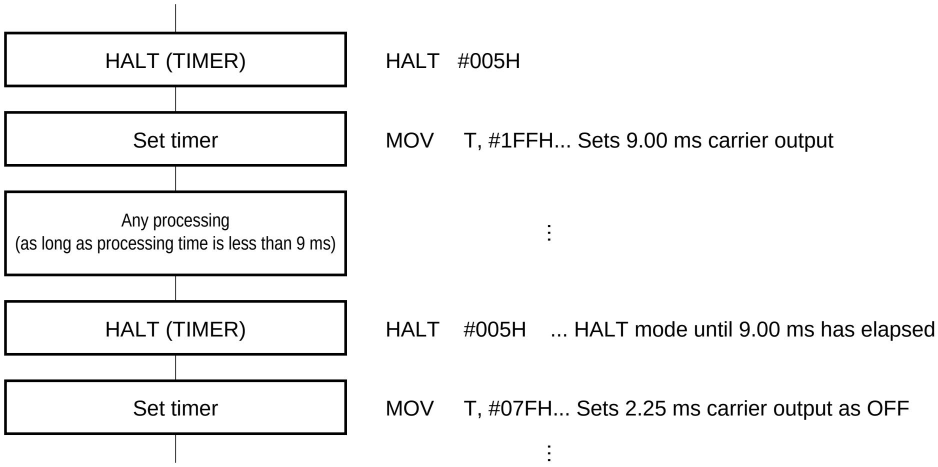


Figure 6-10. Time Management of Timer Counter



```
HALT   #005H
MOV    T, #1FFH... Sets 9.00 ms carrier output
                 ⋮
HALT   #005H   ... HALT mode until 9.00 ms has elapsed
MOV    T, #07FH... Sets 2.25 ms carrier output as OFF
                 ⋮
```

The timer's operation time can be calculated as $[(\text{Setting} + 1) \times 8/f_X]$.

When operating at 455 kHz, "MOV T,#1FFH" indicates that $[(1FFH + 1) \times 8/(455 \times 10^3) \div 9.00\text{ ms}]$ has been set.

6.3.1 Description of processing

Transmission processing includes the following seven types of processing.

- Initialization
- Leader code transmission
- Code transmission (custom code, custom code’, data code, and $\overline{\text{data code}}$)
- Stop bit transmission
- Frame space transmission
- Transmission of second and subsequent frames
- OFF chattering elimination processing

Each of these types of processing is described below.

(1) Initialization

The initialization procedure for transmission processing is described in (a) and (b) below.

(a) The key OFF check counter (R07) is set to 10 times (= 05H).
(b) The data 1 transmit counter is set to 0DFH (for 24 bits).

Table 6-5 describes the counter contents.
The data 1 transmit counter is used to count the number of times “bit data 1” is transmitted within custom code or custom code’ that can affect the frame space transmission time.
The number of “data 1” occurrences in the data code and $\overline{\text{data code}}$ is fixed (at eight) and therefore does not affect the frame space transmission time.

Table 6-5. Description of Data 1 Transmit Counter (R9)

Counter value	Data 1 transmit counter description
D0H | E6H	Number of “data 1” occurrences: 0 to 7 This status does not exist since the number of “data 1” occurrences in data code and $\overline{\text{data code}}$ is fixed (at eight).
E7H	Number of “data 1” occurrences: 8 Number of “data 1” occurrences in custom code and custom code’: 0
E8H | EFH	Number of “data 1” occurrences: 9 to 16 Number of “data 1” occurrences in custom code and custom code’: 1 to 8
F0H | F7H	Number of “data 1” occurrences: 17 to 24 Number of “data 1” occurrences in custom code and custom code’: 9 to 16

(2) Leader code transmission

The transmission method for the leader code is described in (a) to (f) below.

(a) The table reference address (1D8H) for the leader code is set as follows.
- High-order address (0DH) is set to data pointer H (R10)
- Low-order address (8H) of “data 0” in bit data is set to data 0 (R08)

(b) Output data 0H for leader code is set to work area 2 (R1A).

(c) A subroutine (BITOUT0F) is called to transmit the leader code as a one-bit transmission.

(d) Work area 2 (R1A) is shifted leftward and a bit judgment is performed.

(e) The table reference address L for the judged bit is set to data pointer L to enable the transmission time to be set to the timer counter.

(f) The leader code (H: 9.00 ms, L: 4.50 ms) is transmitted.

(3) Code transmission (custom code, custom code’, data code, and $\overline{\text{data code}}$)

The code transmission method is described in (a) to (h) below.

(a) The table reference addresses (1DAH and 1DCH) for bit data used for code transmission are set as follows.
- High-order address (0DH) is set to data pointer H (R10)
- Low-order address (0AH) of “data 0” in bit data is set to data 0 (R08)
- Low-order address (0CH) of “data 1” in bit data is set to data 1 (R18)

(b) The following data is set to work area 2 (R1A) when each type of code is transmitted.
- Custom code H (R14)
- Custom code H’ (R15)
- Data code H (R16)
- $\overline{\text{Data code H}}$ (R16 is fully inverted)
- Custom code L (R04)
- Custom code L’ (R05)
- Data code L (R06)
- $\overline{\text{Data code L}}$ (R06 is fully inverted)

(c) A subroutine is called to transmit the each type of code as a four-bit transmission.

(d) Work area 2 (R1A) is shifted leftward and a bit judgment is performed. If “data 1” is judged, the number of “data 1” occurrences is counted to enable frame space transmission.

(e) The table reference address L for the judged bit is set to data pointer L to enable the transmission time to be set to the timer counter.

(f) Each type of code is transmitted.
- “Data 0” ... H: 0.56 ms, L: 0.56 ms
- “Data 1” ... H: 0.56 ms, L: 1.69 ms

A key OFF check is performed during low level transmission of each bit.
For a description of the key OFF check, see **(7) OFF chattering elimination processing** below.

(g) Steps (d) to (f) above are repeated until four bits have been transmitted.

(h) Steps (b) to (g) above are repeated until all transmissions from custom code H to $\overline{\text{data code L}}$ are completed.

(4) Transmission of stop bit

The method for transmitting stop bits is described in (a) to (f) below.

(a) The table reference address (1DEH) for the stop bit is set as follows.
- High-order address (0DH) is set to data pointer H (R10)
- Low-order address (0EH) of "data 0" in bit data is set to data 0 (R08)

(b) Output data 0H for stop bit is set to work area 2 (R1A).

(c) A subroutine is called to transmit the stop bit as a one-bit transmission.

(d) Work area 2 (R1A) is shifted leftward and a bit judgment is performed.

(e) The table reference address L for the judged bit is set to data pointer L to enable the transmission time to be set to the timer counter.

(f) The stop bit (H: 0.56 ms, L: 3.00 ms) is transmitted.

(5) Frame space transmission

The code transmission time differs between transmission of the "data 1" and "data 0" bit data, with variation in the range of 59.06 ms to 77.06 ms. Therefore, when transmitting code, the frame space transmission time varies according to the number of "data 1" occurrences.

However, since the number of "data 1" occurrences is fixed (at eight) within the data code and $\overline{\text{data code}}$, the frame space transmission times are not affected when these types of code are transmitted. The number of "data 1" occurrences in the custom code and custom code' do affect frame space transmission times.

The number of "data 1" occurrences are counted during transmission to enable responses to changes in transmission times. The frame space transmission times can be adjusted based on the count values. For details of "data 1" counts, see **(3) Code transmission** above.

Table 6-6 lists frame space times correspond to the number of "data 1" occurrences.

Since the maximum value that can be set to the timer counter is 9.00 ms, the frame space is divided into the three patterns shown in Figure 6-11 before being transmitted.

Differences between times shown in Table 6-6 and Figure 6-11 are margins of error during code transmission (such as when the "data 0" low level time of 0.565 is set as 0.56) which are absorbed by frame spaces.

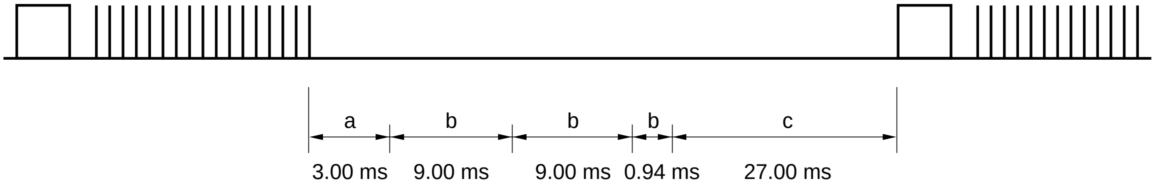
Table 6-6. Frame Space Times Corresponding to “Data 1” Occurrences

No. of “Data 1” occurrences	Transmission time (ms)	No. of “Data 1” occurrences	Transmission time (ms)	No. of “Data 1” occurrences	Transmission time (ms)	No. of “Data 1” occurrences	Transmission time (ms)
8	48.940	13	43.315	18	37.690	23	32.065
9	47.815	14	42.190	19	36.565	24	30.940
10	46.690	15	41.065	20	35.440		
11	45.565	16	39.940	21	34.315		
12	44.440	17	38.815	22	33.190		

Remark Since the number of “data 1” occurrences in data code and $\overline{\text{data code}}$ is fixed (at eight), the “Data 1” values are never between 0 and 7.

Figure 6-11. Method for Dividing Frame Space Transmission Times (1/2)

(a) When there are eight “data 1” occurrences (data 1 transmission counter value: 0E7H)

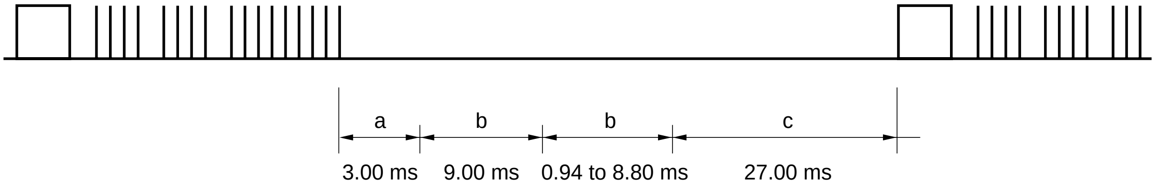


a: After transmitting the stop bit, the frame space is transmitted for 3.00 ms.

b: A frame space for the time period corresponding to the number of “data 1” occurrences (18.94 ms) is transmitted.

c: If a key has been pressed and held, ON chattering elimination processing (9.00 ms × 3 times) is performed.
In other cases, a frame space is transmitted for 27.00 ms.

(b) When there are from 9 to 16 “data 1” occurrences (data 1 transmission counter value range: 0E8H to 0EFH)



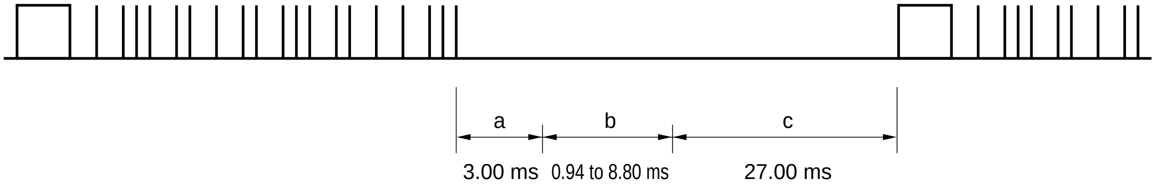
a: After transmitting the stop bit, the frame space is transmitted for 3.00 ms.

b: A frame space for the time period corresponding to the number of “data 1” occurrences (9.94 to 17.80 ms) is transmitted.

c: If a key has been pressed and held, ON chattering elimination processing (9.00 ms × 3 times) is performed.
In other cases, a frame space is transmitted for 27.00 ms.

Figure 6-11. Method for Dividing Frame Space Transmission Times (2/2)

(c) When there are from 17 to 24 “data 1” occurrences (data 1 transmission counter value range: 0F0H to 0F7H)



a: After transmitting the stop bit, the frame space is transmitted for 3.00 ms.

b: A frame space for the time period corresponding to the number of “data 1” occurrences (0.94 to 8.80 ms) is transmitted.

c: If a key has been pressed and held, ON chattering elimination processing (9.00 ms × 3 times) is performed.
In other cases, a frame space is transmitted for 27.00 ms.

(6) Transmission of second and subsequent frames

Until the current key is released or changed, code transmission of second and subsequent frames is repeated starting with the leader code in the same manner as for the first frame. This transmission method is described in (2) to (5) above.

For a description of the key OFF check, see **(7) OFF chattering elimination processing** below.

(7) OFF chattering elimination processing

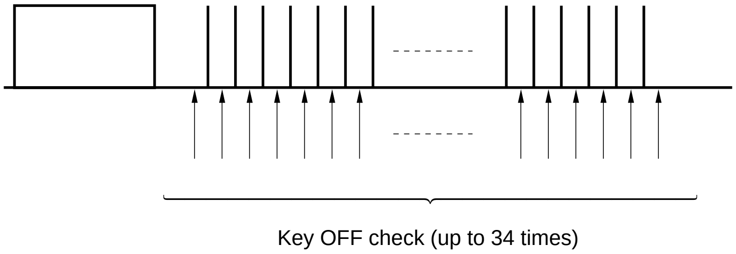
A key OFF check is performed during low level output of bit data.

Key OFF status is confirmed only when absence of key input (i.e., key OFF) has been judged for ten consecutive times during 34 times of low level output that includes the leader code and stop bit. If even one key input (key ON) is detected during the ten consecutive times being checked, the ten-time counter is cleared and the count is restarted.

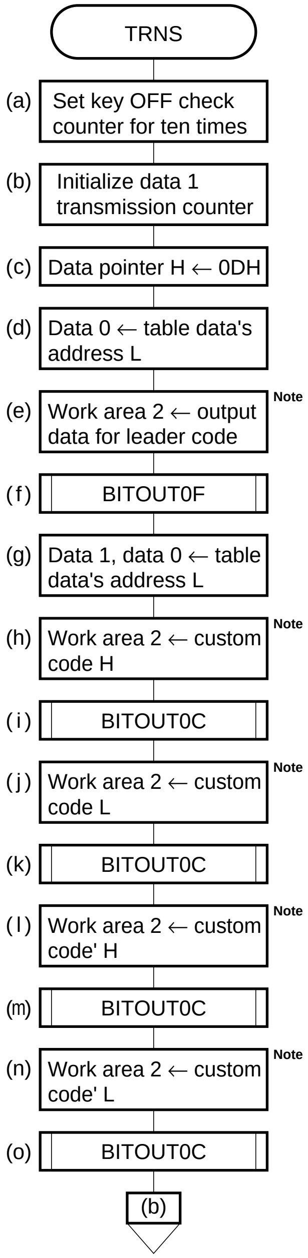
If key OFF status is not confirmed during the entire 34-time check, the key is judged as being pressed and held.

Even if key OFF status is confirmed during transmission of one frame, initialization processing is not performed until at least two frames have been transmitted.

Figure 6-12. Key OFF Check during Bit Data Transmission

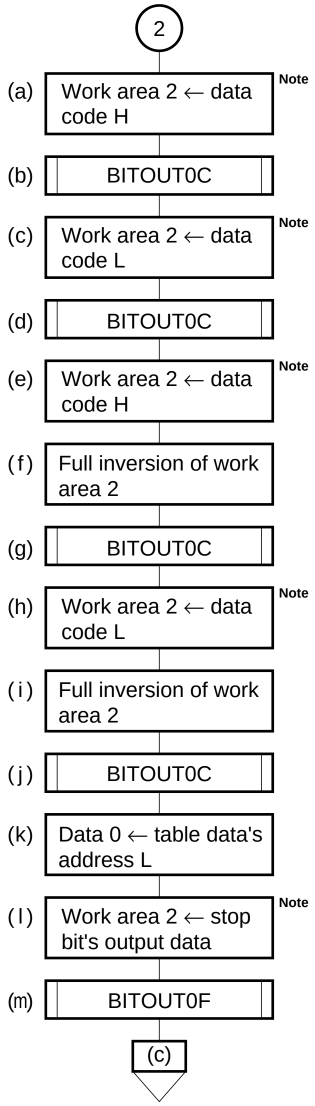


6.3.2 Detailed flow chart


TRNS
(a) Set key OFF check counter for ten times
(b) Initialize data 1 transmission counter
(c) Data pointer H ← 0DH
(d) Data 0 ← table data's address L
(e) Work area 2 ← output data for leader code
Note
(f) BITOUT0F
(g) Data 1, data 0 ← table data's address L
(h) Work area 2 ← custom code H
Note
(i) BITOUT0C
(j) Work area 2 ← custom code L
Note
(k) BITOUT0C
(l) Work area 2 ← custom code' H
Note
(m) BITOUT0C
(n) Work area 2 ← custom code' L
Note
(o) BITOUT0C
(b)

Note As part of the main routine, the output data for the leader code is set to the accumulator (A) and the same data is set to work area 2 (R1A) via the called subroutine (BITOUT0C).

```
******** TRANSMISSION PROCESSING ********

        TIME9M  EQU     1FFH                    ;9.00ms(9.002ms)

;##########   P   U   B   L   I   C   ##########
        ;

;##########   E   X   T   E   R   N   ##########
        EXTRN           MAIN                    ;MAIN Routine
        EXTRN           KEY48                   ;Key Check Routine
        ;

;##########   S    T    A    R    T   ##########
TRNS:
        MOV     R7,#0F5H                        (a) Sets the key OFF check counter (R07) for ten times (= 5H).
                                                    R17 is used for immediate setting (R17 = 0FH).
        MOV     R9,#0DFH                        (b) Initializes (sets DFH to) the data 1 transmission counter (R9).

;************************************************
;               Leader  Code                    Remark  Transmission of leader code (H:  9.00 ms, L:  4.50 ms)
;************************************************
        MOV     R0,#0D0H                        (c) Sets the 0DH (high-order address of the table data for the
                                                    bit data transmission time) to data pointer H.
        MOV     R8,#008H                        (d) Sets the low-order address of the table data for the bit
                                                    data transmission time to data 1 (R18) and data 0 (R08)
                                                    (R18 ← 0H, R08 ← 8H).
        MOV     A,R18                           (e) Sets the output data (0H) for leader code to the accumulator.
                                                    (The accumulator’s value is transferred to work area 2 as
                                                    part of the bit output subroutine.)
        CALL    BITOUT0F                        (f) Calls the bit output subroutine
                                                    (for one-bit transmission:  counter value = 0FH).
;************************************************
;               Custom  Code                    Remark  Transmission of custom code (H:  0.56 ms, L:  0.56 ms or 1.69 ms)
;************************************************

        MOV     R8,#0CAH                        (g) Sets the low-order address of the table data for the custom
                                                    code to data 1 (R18) and data 0 (R08) (R18 ← CH, R08 ← AH).
        MOV     A,R14                           (h) Sets output data H (R14) for custom code to accumulator.
                                                    (The accumulator’s value is transferred to work area 2 as
                                                    part of the bit output subroutine.)
        CALL    BITOUT0C                        (i) Calls the bit output subroutine
                                                    (for four-bit transmission:  counter value = 0CH).
        MOV     A,R04                           (j) Sets output data L (R04) for custom code to accumulator.
                                                    (The accumulator’s value is transferred to work area 2 as
                                                    part of the bit output subroutine.)
        CALL    BITOUT0C                        (k) Calls the bit output subroutine
                                                    (for four-bit transmission:  counter value = 0CH).
;************************************************
;               Custom  Code’                   Remark  Transmission of custom code’ (H:  0.56 ms, L:  0.56 ms or 1.69 ms)
;************************************************

        MOV     A,R15                           (l) Sets output data H (R15) for custom code’ to accumulator.
                                                    (The accumulator’s value is transferred to work area 2 as
                                                    part of the bit output subroutine.)
        CALL    BITOUT0C                        (m) Calls the bit output subroutine (for four-bit transmission:
                                                    counter value = 0CH).
        MOV     A,R05                           (n) Sets output data L (R05) for custom code’ to accumulator.
                                                    (The accumulator’s value is transferred to work area 2 as
                                                    part of the bit output subroutine.)
        CALL    BITOUT0C                        (o) Calls the bit output subroutine
                                                    (for four-bit transmission:  counter value = 0CH).
```


2
(a) Work area 2 ← data code H Note
(b) BITOUT0C
(c) Work area 2 ← data code L Note
(d) BITOUT0C
(e) Work area 2 ← data code H Note
(f) Full inversion of work area 2
(g) BITOUT0C
(h) Work area 2 ← data code L Note
(i) Full inversion of work area 2
(j) BITOUT0C
(k) Data 0 ← table data's address L
(l) Work area 2 ← stop bit's output data Note
(m) BITOUT0F
(c)

Note As part of the main routine, the output data for the leader code is set to the accumulator (A) and the same data is set to work area 2 (R1A) by the called subroutine (BITOUT0C).

```
;***********************************************
;              Data  Code
;***********************************************
        MOV     A,R16
        CALL    BITOUT0C
        MOV     A,R06
        CALL    BITOUT0C
```

Remark Transmission of data code (H: 0.56 ms, L: 0.56 or 1.69 ms)

(a) Sets output data H (R_{16}) for the data code to accumulator. (The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(b) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

(c) Sets output data L (R_{06}) for the data code to accumulator. (The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(d) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

```
;***********************************************
;              Data  Code’
;***********************************************
        MOV     A,R16
        XRL     A,#0FH
        CALL    BITOUT0C
        MOV     A,R06
        XRL     A,#0FH
        CALL    BITOUT0C
```

Remark Transmission of $\overline{\text{data code}}$ (H: 0.56 ms, L: 0.56 or 1.69 ms)

(e) Sets output data H (R_{16}) for the data code to accumulator.

(f) The accumulator’s value is inverted to create output data for $\overline{\text{data code H}}$.
(The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(g) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

(h) Sets output data L (R_{06}) for the data code to accumulator.

(i) The accumulator’s value is inverted to create output data for $\overline{\text{data code L}}$.
(The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(j) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

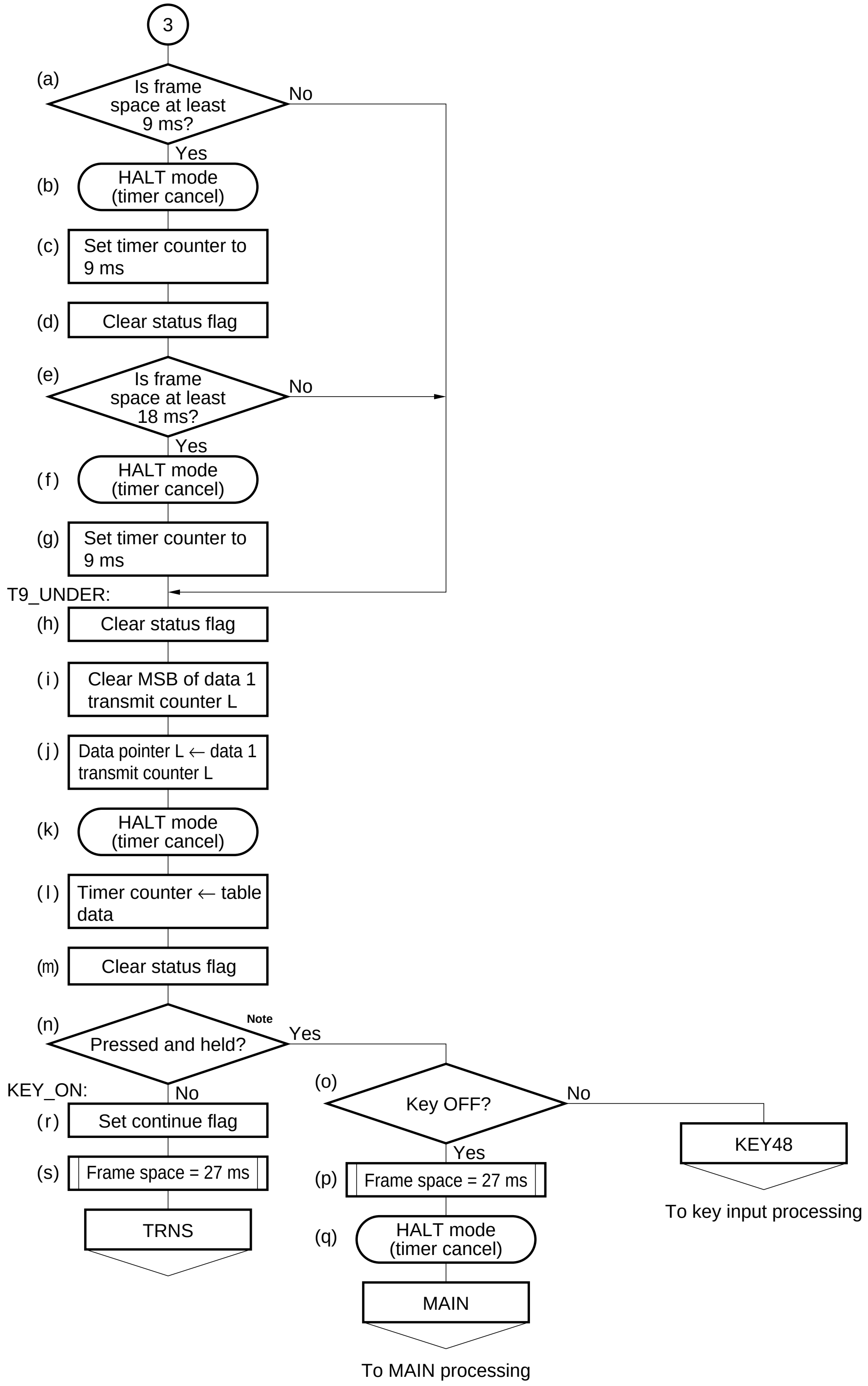
```
;***********************************************
;              Stop  Bit
;***********************************************
        MOV     R8,#00EH
        MOV     A,R18
        CALL    BITOUT0F
```

Remark Transmits stop bit (H: 0.56 ms, L: 3.00 ms)

(k) Sets the low-order address of table data having a stop bit carrier to data 0 (R_{08}).

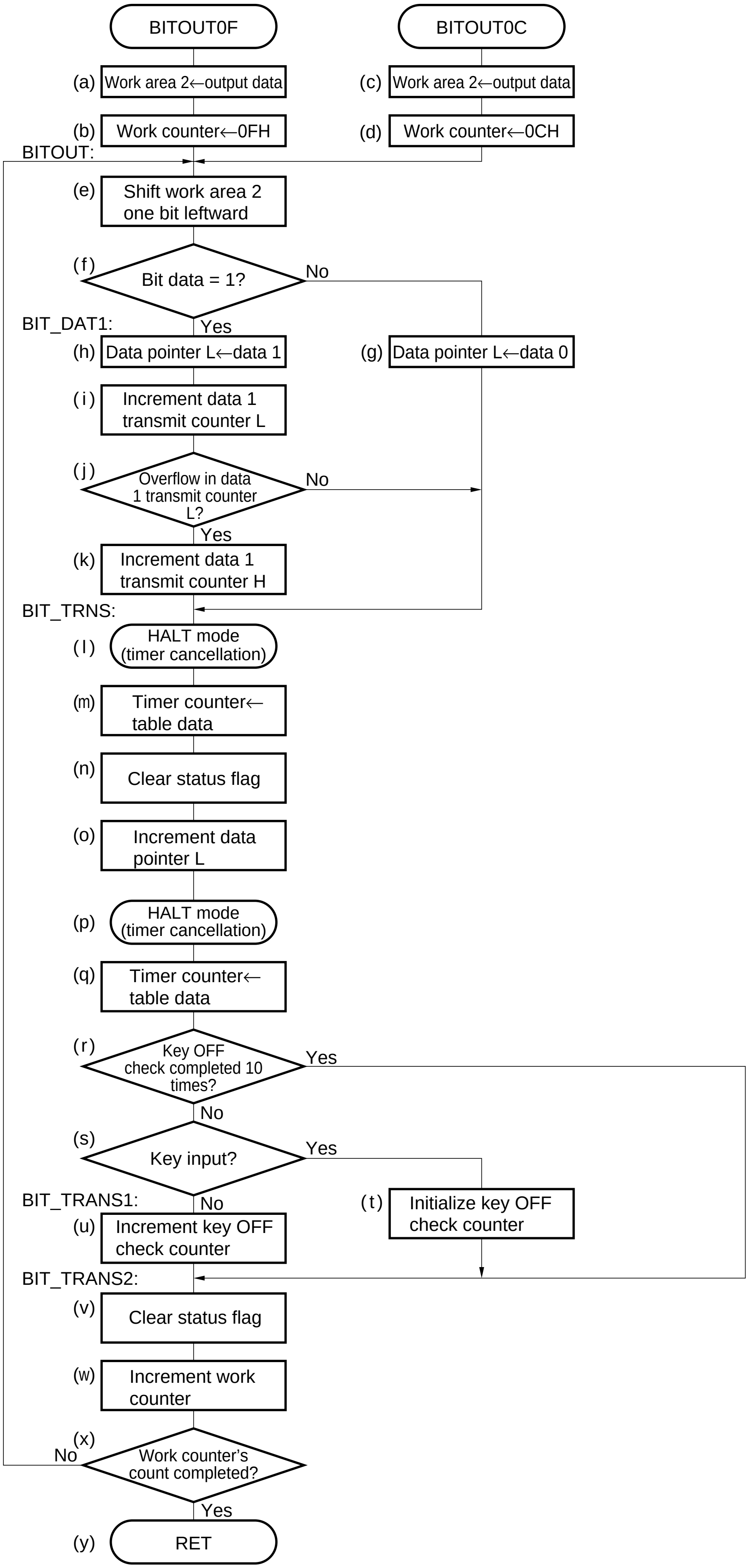
(l) Sets the output data (0H) for the stop bit to the accumulator. (The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(m) Calls the bit output subroutine (for one-bit transmission: counter value = 0FH).



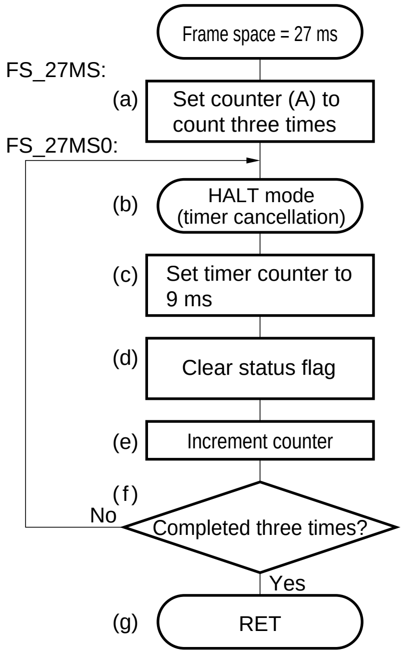
Note If pressed and held, continue flag = 0FH.

```
;************************************************
;               Frame  Space                       Remark  Frame space transmission (L:  30.94 ms to 48.94 ms)
;************************************************
        MOV     A,R19                              (a)  Judges the number of bit data 1 occurrences for custom
        SCAF           ........................         code (16 bits) as eight or less.
        JC      T9_UNDER                                If nine or above, processing branches to T9_UNDER.
        HALT    #005H                              (b)  HALT mode:  canceled when time counter reaches 00H.
        MOV     T,#TIME9M                          (c)  If eight or less, sets timer counter to 9 ms (= 1FFH).
        STTS    #0101B                             (d)  Clears status flag.
        MOV     A,R09                              (e)  Judges the number of bit data 1 occurrences for custom
        RL      A      ........................         code (16 bits) as zero or not zero.
        JC      T9_UNDER                                If 1 to 8, processing branches to T9_UNDER.
        HALT    #005H                              (f)  HALT mode:  canceled when time counter reaches 00H.
        MOV     T,#TIME9M                          (g)  If zero, sets timer counter to 9 ms (= 1FFH) again for a
                                                        total transmission time of 18 ms.
T9_UNDER:
        STTS    #0101B                             (h)  Clears status flag.
        MOV     A,R09
        ANL     A,#0111B ........................  (i)  Clears the MSB of the data 1 transmit counter L (R09).
        MOV     R00,A                              (j)  Sets value of data 1 transmit counter L (R09) to data
                                                        pointer L (R00).
        HALT    #005H                              (k)  HALT mode:  canceled when time counter reaches 00H.
        MOV     T,@R0                              (l)  Performs table lookup to set a value from 0.940 ms to
                                                        8.815 ms according to the number of bit data 1 occur-
                                                        rences counted by the timer counter.
        STTS    #0101B                             (m)  Clears status flag.
        MOV     A,R03
        SCAF           ........................    (n)  Judges transmission of first frame (to check for continue status).
        JNC     KEY_ON                                  If transmitting the first frame, processing branches to KEY_ON.
        MOV     A,R07
        SCAF           ........................    (o)  If transmitting the second or subsequent frame (continue
        JNC     KEY48                                   status), it judges whether a key OFF status occurs during
                                                        the transmission.  If continue status is detected, process-
                                                        ing branches to KEY_48.
;**  Frame Space  = 27ms  **
        CALL    FS_27MS                            (p)  If a key OFF status is detected during bit data transmission,
                                                        the remaining 27 ms of the frame space is output.
        HALT    #005H                              (q)  HALT mode:  canceled when time counter reaches 00H.
        JMP     MAIN                                    Processing branches to initialization processing and MAIN.
        ;
KEY_ON:
        MOV     A,R17  ........................    (r)  Sets continue flag (R03) (in this case, R17 = 0FH).
        MOV     R03,A
  ;**  Frame Space  = 27ms  **
        CALL    FS_27MS                            (s)  Calls subroutine for 27-ms output.
        JMP     TRNS                                    Processing branches to TRNS.
```


BITOUT0F
BITOUT0C
(a) Work area 2←output data
(c) Work area 2←output data
(b) Work counter←0FH
(d) Work counter←0CH
BITOUT:
(e) Shift work area 2 one bit leftward
(f) Bit data = 1?
No
BIT_DAT1:
Yes
(h) Data pointer L←data 1
(g) Data pointer L←data 0
(i) Increment data 1 transmit counter L
(j) Overflow in data 1 transmit counter L?
No
Yes
(k) Increment data 1 transmit counter H
BIT_TRNS:
(l) HALT mode (timer cancellation)
(m) Timer counter← table data
(n) Clear status flag
(o) Increment data pointer L
(p) HALT mode (timer cancellation)
(q) Timer counter← table data
(r) Key OFF check completed 10 times?
Yes
No
(s) Key input?
Yes
BIT_TRANS1:
No
(t) Initialize key OFF check counter
(u) Increment key OFF check counter
BIT_TRANS2:
(v) Clear status flag
(w) Increment work counter
(x) Work counter's count completed?
No
Yes
(y) RET

```
;**************************************************
;*****                                       *****
;*****             Transmit                  *****
;*****        Subroutine : Bit Out           *****
;*****                                       *****
;**************************************************
BITOUT0F:
        MOV     R1A,A                              (a) Sets output data to work area 2 (R1A).
        MOV     A,#0FH                             (b) Sets 0FH to accumulator to set the transmit bit counter to
                                                       count one time.
        JMP     BITOUT                                 Processing branches to BITOUT.
        ;
BITOUT0C:
        MOV     R1A,A                              (c) Sets output data to work area 2 (R1A).
        MOV     A,#0CH                             (d) Sets 0CH to accumulator to set the transmit bit counter to
                                                       count four times.
BITOUT:
        MOV     R0A,A                              Remark  Either 0FH or 0CH can be set to the transmit bit counter (R0A).

        MOV     A,R1A       }
        RL      A           }......................(e) Shifts the output data one bit leftward.
        MOV     R1A,A       }
        JC      BIT_DAT1                           (f) Judges whether bit data value is "0" or "1".
;**  Bit Data = 0  **                                  If it is "1", processing branches to BIT_DAT1.
        MOV     A,R08       }
        MOV     R00,A       }......................(g) Sets data 0 (R08) to data pointer L (R00).
        JMP     BIT_TRNS                               Processing branches to BIT_TRNS.
        ;
;**  Bit Data = 1  **;
BIT_DAT1:
        MOV     A,R18       }
        MOV     R00,A       }......................(h) Sets data 1 (R18) to data pointer L (R00).
        MOV     A,R09       }
        INC     A           }......................(i) Increments data 1 transmit counter L (R09).
        MOV     R09,A       }
        JNC     BIT_TRNS                           (j) If there is no overflow in data 1 transmit counter L (R09),
        MOV     A,R19       }                          processing branches to BIT_TRNS.
        INC     A           }......................(k) Increments data 1 transmit counter H (R19).
        MOV     R19,A       }
BIT_TRNS:
        HALT    #005H                              (l) HALT mode:  canceled when time counter reaches 00H.
        MOV     T,@R0                              (m) Performs table lookup to set transmit time data to timer counter.
        STTS    #0101B                             (n) Clears status flag.
        MOV     A,R00       }
        INC     A           }......................(o) Increments data pointer L (R00).
        MOV     R00,A       }
        HALT    #005H                              (p) HALT mode:  canceled when time counter reaches 00H.

        MOV     T,@R0                              (q) Performs table lookup to set transmit time data to timer counter.
        MOV     T,R07       }
        INC     A           }......................(r) Judges whether key OFF status is maintained during ten consecutive times.
        JC      BIT_TRANS2  }                          If key OFF status is confirmed, processing branches to BIT_TRANS2.
        STTS    #1110B      }.....................{(s) Determines whether or not key input exists.
        JNF     BIT_TRANS1  }                      {   If a key has been pressed and held, processing branches to BIT_TRANS1.
    ;**  Key input exists  **
        MOV     R7,#0F5H                           (t) Initializes (= 5H) the key OFF check counter (R07) if key
                                                       input exists even once during the (10-time) key OFF check.
        JMP     BIT_TRANS2                             Processing branches to BIT_TRANS2.
        ;
```

```
   ;**  No key input  **
BIT_TRANS1:
        INC     A        }
        MOV     R07,A    }······················ (u)  Increments key OFF check counter.
BIT_TRANS2:
        STTS    #0101B                           (v)  Clears status flag.
        MOV     A,R0A    }
        INC     A        }······················ (w)  Increments transmit bit counter (R0A).
        JNC     BITOUT                           (x)  Determines whether or not the transmit bit count has been completed.
                                                      If not completed, processing branches to BITOUT.
        RET                                      (y)  End of processing
        ;
END
```


Frame space = 27 ms
FS_27MS:
(a)
Set counter (A) to count three times
FS_27MS0:
(b)
HALT mode (timer cancellation)
(c)
Set timer counter to 9 ms
(d)
Clear status flag
(e)
Increment counter
(f)
No
Completed three times?
Yes
(g)
RET

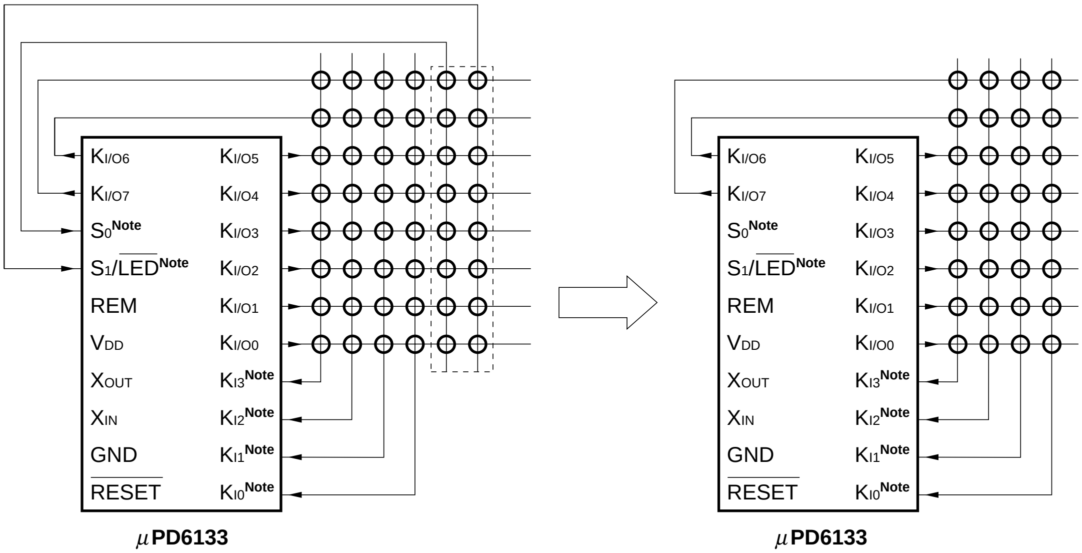
```
;*************************************************
;*****                                       *****
;*****           FRAME SPACE = 27MS          ***** Remark  Subroutine for outputting remaining 27 ms of frame space.
;*****          Subroutine : FS 27ms         *****
;*****                                       *****
;*************************************************
FS_27MS:
        MOV     A,#0DH                             (a) Sets counter (accumulator) to count three times (= 0DH).
FS_27MS0:
        HALT    #005H                              (b) HALT mode:  canceled when time counter reaches 00H.
        MOV     T,#TIME9M                          (c) Sets timer counter to 9 ms (= 1FFH).
        STTS    #0101B                             (d) Clears status flag.
        INC     A                                  (e) Increments counter (accumulator).
        JNC     FS_27MS0                           (f) Determines whether or not three times have been counted.
                                                       If they have not been counted, processing branches to
                                                       FS_27MS0.
        RET                                        (g) End of processing
```

[MEMO]

Note the following caution points when modifying the key matrix or the number of keys.

(1) When changing the number of keys from 48 to 40 or 32

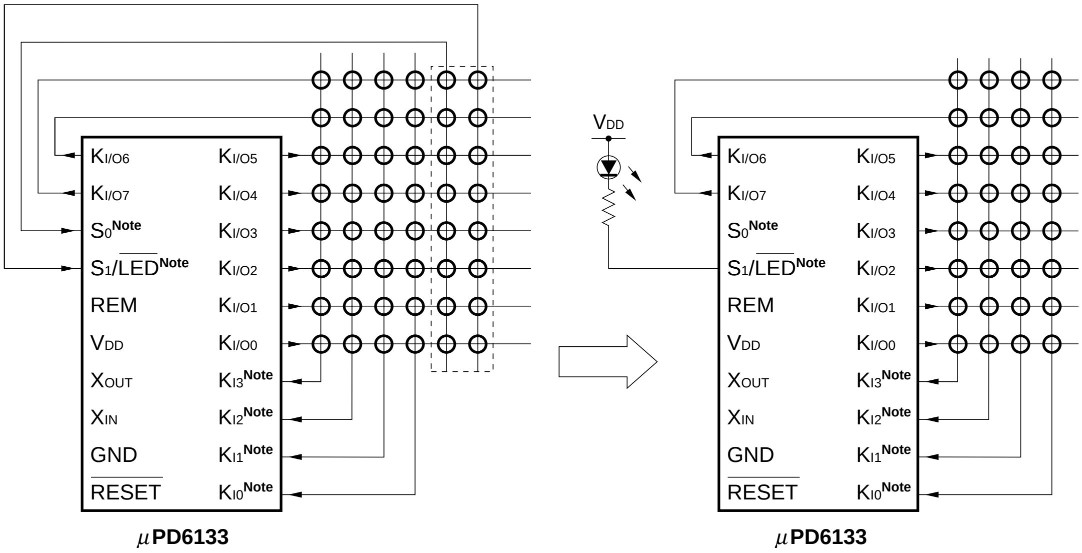
- Delete the keys that are enclosed in broken lines in the following diagram (when doing so, if the S_0 and $S_1/\overline{LED}$ pins are left unconnected, the number of keys can be changed without modifying the main program itself.



Note   Set by the program for “internal pull-down resistor”.

(2) In addition to the modification described in (1) above, the $S_1/\overline{LED}$ pin can be used as an LED pin:

- Delete the keys that are enclosed in broken lines in the following diagram.
- Leave the S_0 pin unconnected.
- Change the $S_1/\overline{LED}$ pin so that b2 (the bit that sets the I/O mode for the $S_1/\overline{LED}$ port) in the main program’s control register 1 (P4) remains in output mode (bit value = 1). (In μPD6133 Series products, setting the $S_1/\overline{LED}$ pin to output mode automatically eliminates internal pull-down resistance.)



Note Set by the program for “internal pull-down resistor”.

CHAPTER 8 PROGRAM LIST

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 01-002



SOURCE = MAIN48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
     1                                  ;*********************************************************************
     2                                  ;***                                                               ***
     3                                  ;***     Multi - Purpose Remote Control Transmitter System         ***
     4                                  ;***                                                               ***
     5                                  ;***     CPU        : uPD6133 Series                               ***
     6                                  ;***     CPU Clock  : 455kHz                                       ***
     7                                  ;***     Trans. CODE: NEC-R  Format (48Key)                        ***
     8                                  ;***     Version    : 2.0                                          ***
     9                                  ;***     Programmer : NEC IC Microcomputer Systems Corporation     ***
    10                                  ;***                                                               ***
    11                                  ;***                    Copyright(c)   NEC Corporation   1995      ***
    12                                  ;***                    Copyright(c)   NIMS Corporation  1995      ***
    13                                  ;*********************************************************************
    14                                  EJECT
```

```
AS6133 V1.01    << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 01-003



SOURCE = MAIN48.ASM

E STNO LOC. OBJ.                M  I   SOURCE STATEMENT
    15      001C                               TIME05M         EQU     01CH                   ;0.5ms (0.510ms)
    16      01FF                               DamyTime        EQU     512-1                  ;9.00ms (= 17.582us * 512)
    17
    18                                 ;##########   P   U   B   L   I   C   ##########
    19                                         PUBLIC          MAIN
    20                                         ;
    21                                 ;##########   E   X   T   E   R   N   ##########
    22                                         ;
    23
    24                                 ;##########   S    T    A    R    T   ##########
    25
    26                                         ; Control Register (P3)     Reset:03H
    27                                         ;==============================================================;
    28                                         ;  D9   D8 !  D7 !  D6 !  D5 !  D4 !  D3 !  D2 !  D1 !  D0 !    ;
    29                                         ;          !     ! D.P.! D.P.! D.P.! TCTL! CARY! MOD1! MOD0!    ;
    30                                         ; Test Mode!     ! AD10! AD9 ! AD8 !     !     !     !     !    ;
    31                                         ;--------------------------------------------------------------;
    32                                         ;          !  0  !  *  !  *  !  *  ! 1/1 !  ON ! fx,fx/8   ! 0  ;
    33                                         ; Set ″0″  !-----------------------------------!.fx/12(1/2)!----;
    34                                         ;          !  0  !  *  !  *  !  *  ! 1/2 ! OFF !,fx/12(1/3)! 1  ;
    35                                         ;==============================================================;
    36
    37                                         ; Control Register (P4)      Reset:26H
    38                                         ;==============================================================;
    39                                         ;  D9   D8 !  D7 !  D6 !  D5 !  D4 !  D3 !  D2   !  D1 !  D0 !   ;
    40                                         ;          !     !     !  KI !S0/S1!     !S1/LED! KI/0!  S0 !   ;
    41                                         ;          !     !     ! Pull! Pull!     ! MODE ! MODE! MODE!   ;
    42                                         ;--------------------------------------------------------------;
    43                                         ;          !  0  !  0  ! OFF ! OFF !  0  !  IN  !  IN ! OFF ! 0 ;
    44                                         ;    X     !-----------------------------------------------!---;
    45                                         ;          !  0  !  0  !  ON !  ON !  0  ! OUT  ! OUT !  IN ! 1 ;
    46                                         ;==============================================================;
    47
    48                                 ;***********************************************
    49                                 ;       Control Register Initialize
    50                                 ;***********************************************
    51                                 MAIN:
    52 0000 E6F8 EFEF                          OUT     P0,#0FFH                               ;KI/O All High
    53 0002 E6FC E3E3                          OUT     P4,#033H
    54 0004 E6FB E1E3                          OUT     P3,#013H                               ;Set Data Pointer (P3:D8,D9)
    55 0006 E6FF F7FF                          MOV     T,#DamyTime
    56 0008 E3F1 E0E5                          STTS    #0101B                                 ;Clear Status Flag
    57
    58                                 ;***********************************************
    59                                 ;       RAM Initialize Routine
    60                                 ;***********************************************
    61 000A E6E1 E0E0                          MOV     R1,#000H                               ;Final Key Data (R1) = 00H
    62 000C E6E3 E0E0                          MOV     R3,#000H                               ;K29 ON Flag (R13) , Continuance Flag (R03) = 00H
    63
    64 000E E2F1 E0E8                          HALT    #008H                                  ;STOP mode (KI = High)
    65 0010 E0E0                               NOP                                            ;No operation command
    66 0011 E6FF E0E7                          MOV     T,#TIME05M                             ;Set Timer : 0.5ms
    67                                         ;
    68                                 END
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 02-002



SOURCE = KEY48.ASM

E STNO LOC. OBJ.                M  I   SOURCE STATEMENT
     1                                 ;*************************************************
     2                                 ;*****                                       *****
     3                                 ;*****       uPD6133 Series  Key Check       *****
     4                                 ;*****   Trans. CODE: NEC-R  Format (48Key)  *****
     5                                 ;*****                                       *****
     6                                 ;*************************************************
     7      01FF                               TIME9M  EQU     1FFH                           ;9.00ms (9.002ms)
     8      0050                               CUSTM1  EQU     050H                           ;Custom Code  = 0AH
     9      00AF                               CUSTM2  EQU     0AFH                           ;Custom Code' = F5H
    10
    11                                 ;##########   P   U   B   L   I   C   ##########
    12                                         PUBLIC          KEY48
    13                                         ;
    14
    15                                 ;##########   E   X   T   E   R   N   ##########
    16                                         EXTRN           MAIN                           ;MAIN Routine
    17                                         ;
    18
    19                                 ;##########   S    T    A    R    T   ##########
    20                                 KEY48:
    21 0013 E6E0 E0EC                          MOV     R0,#00CH                               ;Chattering Counter (R00) = 3 Times
    22                                 ;***********************************************
    23                                 ;               ON Chattering
    24                                 ;***********************************************
    25                                 ON_CHAT:
    26 0015 E3F1 E0E5                          STTS    #0101B                                 ;Clear Status Flag
    27 0017 E6F8 EFEF                          OUT     P0,#0FFH                               ;KI/O All High
    28 0019 E6FC E3E3                          OUT     P4,#033H
    29 001B E2F1 E0E5                          HALT    #005H                                  ;HALT mode (Timer = 00H)
    30 001D E6FF F7FF                          MOV     T,#TIME9M                              ;Timer = 9.00ms
    31 001F E6E9 E0E0                          MOV     R9,#000H                               ;Comparative Key Data (R9) = 00H
    32 0021 E6E6 E0E8                          MOV     R6,#008H                               ;Key Flag (R16) = 00H , Key Scan Counter (R06) = 08H
    33 0023 E3F1 E0EE                          STTS    #1110B                                 ;With input Key (KI or S0 or S1) ?
    34 0025 EFF1 E0E0                                                  JNF     MAIN           ;       else  MAIN:     (Without input Key)
    35                                                                                        ;       then            (With input Key)
    36 0027 E6E8 E0E1                          MOV     R8,#001H                               ;Key Scan Data (R8) Initialize : 001H
    37
    38                                 ;***********************************************
    39                                 ;                  Key Scan
    40                                 ;***********************************************
    41                                 KEYS_OUT:
    42 0029 FFE8                               MOV     A,R08
    43 002A E5F8                               OUT     P00,A                                  ;Key Scan Data L (P00) Output
    44 002B FEE8                               MOV     A,R18
    45 002C E4F8                               OUT     P10,A                                  ;Key Scan Data H (P10) Output
    46                                     ;+++++++++++++++++++++++++++++
    47                                     ;++      100us Wait       ++
    48                                     ;+++++++++++++++++++++++++++++
    49 002D E6E2 EEE1                          MOV     R2,#0E1H                               ;Set Immediate Data : R2
    50 002F E6EB ECE3                          MOV     RB,#0C3H                               ;Set Immediate Data : RB
    51 0031 E0E0                               NOP
    52 0032 E0E0                               NOP
    53 0033 E0E0                               NOP
    54 0034 E0E0                               NOP
    55                                 ;***********************************************
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 02-003



SOURCE = KEY48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
    56                                  :               Key Return Check
    57                                  ;***********************************************
    58 0035 E3F1 E0EB                           STTS    #1011B                                  ;With input Key (KI or S0 or S1) ?
    59 0037 EFF1 E9E3                                                   JNF     NO_KEYS         ;       else  NO_KEYS:  (Without input Key)
    60                                                                                          ;       then            (With input Key)
    61 0039 E3F1 E0E3                           STTS    #0011B                                  ;With input KI (P11) ?
    62 003B EFF1 E4E9                           JNF     S0S1CALC                                ;       else S0S1CALC: (With input KI)
    63                                                                                          ;       then           (Without input KI)
    64                                                          ;*******************************
    65                                                          ;       KI Data Calculate
    66                                                          ;*******************************
    67 003D FFF9                                                        IN      A,P01           ;With input KI , Does it have S0 or S1 ?
    68 003E FCF3                                                        RL      A               ;With input S0 and S1 , it judges ″MULTIPLE KEYS ERROR ″ ,
    69 003F ECF1 E0E0                                                   JC      MAIN            ; and then goes to ″ MAIN: ″ !
    70 0041 FCF3                                                        RL      A
    71 0042 ECF1 E0E0                                                   JC      MAIN
    72
    73 0044 FEEB                                                        MOV     A,R1B           ;R1B = 0CH
    74 0045 E5EA                                                        MOV     R0A,A           ;Key Return Check Counter (R0A) = 4 Times (=0CH)
    75 0046 FEF9                                                        IN      A,P11
    76 0047 E8F1 E4EE                                                   JMP     KEY_RET0
    77                                                                  ;
    78
    79                                          ;*******************************
    80                                          ;       S0,S1 Data Calculate
    81                                          ;*******************************
    82                                  S0S1CALC:
    83 0049 E6E9 ECE0                           MOV     R9,#0C0H
    84 004B FEE2                                MOV     A,R12                                   ;R12 = 0EH
    85 004C E5EA                                MOV     R0A,A                                   ;Set Key Return Check Counter (R0A) = 2 Times (=0EH)
    86 004D FFF9                                IN      A,P01
    87
    88                                  KEY_RET0:
    89 004E E4E0                                MOV     R10,A                                   ;Work (R10) = P11 or P01
    90 004F E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
    91                                  KEY_RET:
    92 0051 FEE0                                MOV     A,R10
    93 0052 FEF3                                RLZ     A
    94 0053 E4E0                                MOV     R10,A
    95 0054 EDF1 E8EE                                                   JNC     KRC_INC         ;With input Key ?
    96                                      ;**  With input Key  **                                 ;       else KRC_INC: (Without input Key)
    97 0056 FEE6                                MOV     A,R16                                   ;       then          (With input Key)
    98 0057 FAF3                                SCAF                                            ;Key Flag (R16) = 0FH ?
    99 0058 EDF1 E6E9                           JNC     KEY_CALC                                ;       else KEY_CALC: (Except for 0FH)
   100 005A FEE3                                                        MOV     A,R13           ;       then           (= 0FH)
   101 005B FAF3                                                        SCAF                    ;K29 ON Flag (R13) = 0FH ?
   102 005C EDF1 E0E0                                                   JNC     MAIN            ;       else  MAIN:     (= Except for 0FH)
   103                                                                                          ;       then            (= 0FH)
   104 005E FFF1 F4E0                                                   MOV     A,#0100B
   105 0060 F4E9                                                        XRL     A,R19
   106 0061 FAF3                                                        SCAF                    ;Comparative Key Data = K29 (R19=0BH) ?
   107 0062 EDF1 E0E0                                                   JNC     MAIN            ;       then  MAIN:     (Key = Except for K29)
   108 0064 FFE9                                                        MOV     A,R09           ;       else            (Key = K29)
   109 0065 F5EB                                                        XRL     A,R0B           ;R0B = 03H
   110 0066 FAF3                                                        SCAF                    ;Comparative Key Data = K29 (R09=0CH) ?
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 02-004



SOURCE = KEY48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   111 0067 EDF1 E0E0                                                   JNC     MAIN            ;       then DBL_K_CHK:   (Key = Except for K29)
   112
   113                                          ;*******************************
   114                                          ;       Key Data Calculate
   115                                          ;*******************************
   116                                  KEY_CALC:
   117 0069 FFEB                                MOV     A,R0B                                   ;R0B = 03H
   118 006A F4E9                                XRL     A,R19
   119 006B FAF3                                SCAF                                            ;Comparative Key Data H (R19) = 0CH ?
   120 006C EDF1 E7E5                           JNC     KI_CALC                                 ;       else  KI_CALC:  (KI Data)
   121                                                        ;**  S0 or S1 data Calculation  **;       then            (S0 or S1 Data)
   122 006E FFE6                                                        MOV     A,R06           ;Acc <-- Key Scan Counter (R06)
   123 006F FCF3                                                        RL      A
   124 0070 FAE2                                                        ANL     A,R12           ;R12 = 1110B
   125 0071 E5E9                                                        MOV     R09,A
   126 0072 FFE2                                                        MOV     A,R02           ;R02 = 01H
   127 0073 E8F1 E8E1                                                   JMP     KL_CALC
   128                                                                  ;
   129                                      ;**  KI data Calculation  **
   130                                  KI_CALC:
   131 0075 FFE6                                MOV     A,R06                                   ;Acc <-- Key Scan Counter(R06)
   132 0076 FCF3                                RL      A
   133 0077 FCF3                                RL      A                                       ;Left shift (2 Times)
   134 0078 E5E9                                MOV     R09,A                                   ;Comparative Key Data L (R09) <-- Acc
   135 0079 FBE2                                ANL     A,R02                                   ;Acc AND 0001B (R02=01H)
   136 007A FDF1 FAE0                           ORL     A,#1010B                                ;Acc OR 1010B
   137 007C E4E9                                MOV     R19,A                                   ;Comparative Key Data H (R19) <-- Acc
   138 007D FFE9                                MOV     A,R09                                   ;Acc <-- Comparative Key Data L (R09)
   139 007E FAEB                                ANL     A,R1B                                   ;Acc AND 1100B (R1B=0CH)
   140 007F E5E9                                MOV     R09,A                                   ;Comparative Key Data L (R09) <-- Acc
   141 0080 FFEB                                MOV     A,R0B                                   ;Acc <-- 0011B (R0B=03H)
   142
   143                                  KL_CALC:
   144 0081 FBEA                                ANL     A,R0A                                   ;Acc  AND  Key Return Check Counter (R0A)
   145 0082 FDE9                                ORL     A,R09                                   ;Acc OR Comparative Key Data L (R09)
   146 0083 E5E9                                MOV     R09,A                                   ;Comparative Key Data L (R09) <-- Acc
   147
   148                                      ;**  Double Key Check  **
   149 0084 FEE6                                MOV     A,R16
   150 0085 FAF3                                SCAF                                            ;Key Flag (R16) = 0FH ?
   151 0086 EDF1 E8EB                           JNC     SINGLE_K                                ;       else  SINGLE_K  (Single Key)
   152 0088 FFF1 F9E0                                                   MOV     A,#09H          ;       then            (Double Key)
   153 008A E4E9                                                        MOV     R19,A
   154                                  SINGLE_K:
   155 008B FFF1 FFE0                           MOV     A,#0FH
   156 008D E4E6                                MOV     R16,A
   157
   158                                  KRC_INC:
   159 008E FFEA                                MOV     A,R0A
   160 008F F4F3                                INC     A                                       ;Key Return Check Counter (R0A) increment
   161 0090 E5EA                                MOV     R0A,A                                   ;End of Key Return Check ?
   162 0091 EDF1 E5E1                                                   JNC     KEY_RET         ;       else  KEY_RET:  (Continuous)
   163                                                                                          ;       then            (End)
   164
   165                                  NO_KEYS:
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 02-005



SOURCE = KEY48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   166 0093 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
   167 0095 FFE6                                MOV     A,R06
   168 0096 F4F3                                INC     A                                       ;Key Scan Counter (R06) increment
   169 0097 E5E6                                MOV     R06,A                                   ;End of Key Scan number of 8 Time ?
   170 0098 ECF1 EAE6                                                   JC      KEY_DATA        ;       then  KEY_DATA: (End)
   171 009A FEE8                                MOV     A,R18                                   ;       else           (Continuous)
   172 009B FCF3                                RL      A                                       ;Key Scan Data H (R18) Shift
   173 009C E4E8                                MOV     R18,A
   174 009D FFE8                                MOV     A,R08
   175 009E FCF3                                RL      A
   176 009F E5E8                                MOV     R08,A                                   ;Key Scan Data L (R08) Shift --> CY ?
   177 00A0 EDF1 E2E9                           JNC     KEYS_OUT                                ;       else  KEYS_OUT: (Key Scan Data a low rank)
   178 00A2 E6E8 E1E0                                                   MOV     R8,#010H        ;       then           (Key Scan Data a high rank)
   179 00A4 E8F1 E2E9                                                   JMP     KEYS_OUT
   180                                                                  ;
   181
   182                                   ;***********************************************
   183                                   ; Transmit Key Data = ON Chattering Key Data ?
   184                                   ;***********************************************
   185                                   KEY_DATA:
   186 00A6 E6F8 EFEF                           OUT     P0,#0FFH                                ;KI/O All High
   187 00A8 E6E8 EFE0                           MOV     R8,#0F0H                                ;Set Immediate Data : R8
   188 00AA FEE6                                MOV     A,R16
   189 00AB FAF3                                SCAF                                            ;Key Flag (R16) = 0FH ?
   190 00AC EDF1 E0E0                                                   JNC     MAIN            ;       else  MAIN:    (= Except for 0FH)
   191 00AE FFE0                                MOV     A,R00                                   ;       then           (= 0FH)
   192 00AF F5EB                                XRL     A,R0B                                   ;R0B = #0011B
   193 00B0 FAF3                                SCAF                                            ;Chattering counter (R00) = 1 Times (=0DH) ?
   194 00B1 ECF1 EBEE                                                   JC      KEYCHECK        ;       then  KEYCHECK: (= 0DH)
   195                                                                                          ;       else           (= 0E-0FH)
   196 00B3 FEE4                                MOV     A,R14
   197 00B4 F4E8                                XRL     A,R18                                   ;R18 = 0FH
   198 00B5 F4E9                                XRL     A,R19                                   ;Acc = 0FH
   199 00B6 FAF3                                SCAF                                            ;Key Data H (R14) = Comparative Key Data H (R19)
   200 00B7 EDF1 E0E0                                                   JNC     MAIN            ;       else  MAIN:    (Unmatch)
   201                                                                                          ;       then           (Match)
   202 00B9 F5E4                                XRL     A,R04                                   ;Acc = 0FH
   203 00BA F5E9                                XRL     A,R09
   204 00BB FAF3                                SCAF                                            ;Key Data L (R04) = Comparative Key Data L (R09)
   205 00BC EDF1 E0E0                                                   JNC     MAIN            ;       else  MAIN:    (Unmatch)
   206                                   KEYCHECK:                                              ;       then           (Match)
   207 00BE FEE9                                MOV     A,R19
   208 00BF E4E4                                MOV     R14,A                                   ;Key Data H (R14) <-- Comparative Key Data H (R19)
   209 00C0 FFE9                                MOV     A,R09
   210 00C1 E5E4                                MOV     R04,A                                   ;Key Data L (R04) <-- Comparative Key Data L (R09)
   211 00C2 FFE0                                MOV     A,R00
   212 00C3 F4F3                                INC     A                                       ;Chattering Counter(R00) increment
   213 00C4 E5E0                                MOV     R00,A
   214 00C5 FAF3                                SCAF                                            ;End of Chattering Routine of 3 Time ?
   215 00C6 EDF1 E1E5                                                   JNC     ON_CHAT         ;       else  ON_CHAT:  (= 0DH)
   216                                          ;**  Key High Check **                          ;       then           (= 0E-0FH)
   217 00C8 F4E1                                XRL     A,R11                                   ;Acc = 0FH
   218 00C9 F4E4                                XRL     A,R14
   219 00CA FAF3                                SCAF                                            ;Final Key Data H (R11) = Key Data H (R14) ?
   220 00CB EDF1 EDE2                           JNC     KDAT_SET                                ;       else  KDAT_SET: (Unmacth)
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 02-006



SOURCE = KEY48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   221                                                                 ;**  Key Low Check  **  ;       then            (Match)
   222 00CD F5E1                                                       XRL     A,R01           ;Acc = 0FH
   223 00CE F5E4                                                       XRL     A,R04
   224 00CF FAF3                                                       SCAF                    ;Final Key Data L(R01) = Key Data L (R04) ?
   225 00D0 ECF1 EDE8                                                  JC      K29CHECK        ;       then  K29CHECK: (Match)
   226                                  KDAT_SET:                                              ;       else            (Unmatch)
   227 00D2 FEE4                                MOV     A,R14
   228 00D3 E4E1                                MOV     R11,A                                  ;Final Key Data H (R11) <-- Key Data H (R14)
   229 00D4 FFE4                                MOV     A,R04
   230 00D5 E5E1                                MOV     R01,A                                  ;Final Key Data L (R01) <-- Key Data L (R04)
   231 00D6 FFE8                                MOV     A,R08
   232 00D7 E5E3                                MOV     R03,A                                  ;Continuance Flag = 00H (R08=00H)
   233
   234                                  K29CHECK:
   235 00D8 FFF1 F4E0                           MOV     A,#0100B
   236 00DA F4E1                                XRL     A,R11
   237 00DB FAF3                                SCAF                                           ;Final Key Data = K29 (R11=0BH) ?
   238 00DC EDF1 EEE3                           JNC     DBL_K_CHK                              ;       then  DBL_K_CHK:  (Key = Except for K29)
   239 00DE FFE1                                                       MOV     A,R01           ;       else              (Key = K29)
   240 00DF F5EB                                                       XRL     A,R0B           ;R0B = 03H
   241 00E0 FAF3                                                       SCAF                    ;Final Key Data = K29 (R01=0CH) ?
   242 00E1 ECF1 EEEA                                                  JC      K29_FLG         ;       then  K29_FLG:  (Key = Except for K29)
   243                                                                                         ;       else            (Key = K29)
   244                                  DBL_K_CHK:
   245 00E3 FFF1 F6E0                           MOV     A,#06H
   246 00E5 F4E1                                XRL     A,R11
   247 00E6 FAF3                                SCAF                                           ;Final Key Data = K29+ K30 or K31 or K32 (R11=09H) ?
   248 00E7 ECF1 EEEB                           JC      DATA_SET                               ;       then  DATA_SET: (Key = Except for Double Key)
   249 00E9 FFE8                                                       MOV     A,R08
   250                                  K29_FLG:
   251 00EA E4E3                                                       MOV     R13,A           ;K29 ON Flag = 00H (R08=00H)
   252                                  ;***********************************************
   253                                  ;               Transmit Data Set
   254                                  ;***********************************************
   255                                  DATA_SET:
   256                                      ;**  Custom Code  **
   257 00EB E6E4 E5E0                           MOV     R4,#CUSTM1                             ;Custom Code(R4) <-- Custom Code
   258
   259                                      ;**  Custom Code’ **
   260 00ED E6E5 EAEF                           MOV     R5,#CUSTM2                             ;Custom Code’(R5) <-- Custom Code’
   261
   262                                      ;**   Data Code   **
   263 00EF FEE1                                MOV     A,R11
   264 00F0 E4E0                                MOV     R10,A                                  ;Data Pointer H(R10) <-- Final Key Data H(R11)
   265 00F1 FFE1                                MOV     A,R01
   266 00F2 E5E0                                MOV     R00,A                                  ;Data Pointer L(R00) <-- Final Key Data L(R01)
   267 00F3 E7E6                                MOV     R6,@R0
   268
   269                                  END


TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-002



SOURCE = TRANS48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
     1                                  ;*************************************************
     2                                  ;*****                                       *****
     3                                  ;*****       uPD6133 Series  Transmit        *****
     4                                  ;*****   Trans. CODE: NEC-R  Format (48Key)  *****
     5                                  ;*****                                       *****
     6                                  ;*************************************************
     7      01FF                                TIME9M          EQU     1FFH                    ;9.00ms (9.002ms)
     8
     9                                  ;##########   P   U   B   L   I   C   ##########
    10                                          ;
    11
    12                                  ;##########   E   X   T   E   R   N   ##########
    13                                          EXTRN           MAIN                            ;MAIN Routine
    14                                          EXTRN           KEY48                           ;Key Check Routine
    15                                          EXTRN           FS_27MS                         ;Frame Space Subroutine
    16                                          ;
    17
    18                                  ;##########   S    T    A    R    T   ##########
    19                                  TRNS:
    20 00F4 E6E7 EFE5                           MOV     R7,#0F5H                                ;R17 : Set Immediate Data (=0FH)
    21                                                                                          ;Key OFF Check Counter (R07) = 10 Times (=05H)
    22 00F6 E6E9 EDEF                           MOV     R9,#0DFH                                ;Data1 Trans Counter (R9) = DFH
    23                                  ;***********************************************
    24                                  ;               Leader  Code
    25                                  ;***********************************************
    26 00F8 E6E0 EDE0                           MOV     R0,#0D0H                                ;Data Pointer H (R10) = 0DH
    27 00FA E6E8 E0E8                           MOV     R8,#008H                                ;Set Table Address L
    28 00FC FEE8                                MOV     A,R18                                   ;Output Data (Work) <-- 00H (R18)
    29 00FD E6F2 E8F1 E5FF                      CALL    BITOUT0F
    30
    31                                  ;***********************************************
    32                                  ;               Custom  Code
    33                                  ;***********************************************
    34 0100 E6E8 ECEA                           MOV     R8,#0CAH                                ;Set Table Address L
    35 0102 FEE4                                MOV     A,R14                                   ;Output Data (Work) <-- Custom Code H (R14)
    36 0103 E6F2 E8F1 E6F4                      CALL    BITOUT0C
    37
    38 0106 FFE4                                MOV     A,R04                                   ;Output Data (Work) <-- Custom Code L (R04)
    39 0107 E6F2 E8F1 E6F4                      CALL    BITOUT0C
    40
    41                                  ;***********************************************
    42                                  ;               Custom  Code’
    43                                  ;***********************************************
    44 010A FEE5                                MOV     A,R15                                   ;Output Data (Work) <-- Custom Code’ H (R15)
    45 010B E6F2 E8F1 E6F4                      CALL    BITOUT0C
    46
    47 010E FFE5                                MOV     A,R05                                   ;Output Data (Work) <-- Custom Code’ L (R05)
    48 010F E6F2 E8F1 E6F4                      CALL    BITOUT0C
    49
    50                                  ;***********************************************
    51                                  ;               Data  Code
    52                                  ;***********************************************
    53 0112 FEE6                                MOV     A,R16                                   ;Output Data (Work) <-- Data Code H (R16)
    54 0113 E6F2 E8F1 E6F4                      CALL    BITOUT0C
    55
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-003



SOURCE = TRANS48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   56 0116 FFE6                                 MOV     A,R06                                   ;Output Data (Work) <-- Data Code L (R06)
   57 0117 E6F2 E8F1 E6F4                       CALL    BITOUT0C
   58
   59                                   ;***********************************************
   60                                   ;              Data  Code’
   61                                   ;***********************************************
   62 011A FEE6                                 MOV     A,R16                                   ;Output Data (Work) <-- Data Code H (R16)
   63 011B F5F1 FFE0                            XRL     A,#0FH
   64 011D E6F2 E8F1 E6F4                       CALL    BITOUT0C
   65
   66 0120 FFE6                                 MOV     A,R06                                   ;Output Data (Work) <-- Data Code L (R06)
   67 0121 F5F1 FEE0                            XRL     A,#0FH
   68 0123 E6F2 E8F1 E6F4                       CALL    BITOUT0C
   69
   70                                   ;***********************************************
   71                                   ;              Stop  Bit
   72                                   ;***********************************************
   73 0126 E6E8 E0EE                            MOV     R8,#00EH                                ;Set Table Address L
   74 0128 FEE8                                 MOV     A,R18                                   ;Output Data (Work) <-- 00H (R18)
   75 0129 E6F2 E8F1 E5FF                       CALL    BITOUT0F
   76
   77                                   ;***********************************************
   78                                   ;              Frame  Space
   79                                   ;***********************************************
   80 012C FEE9                                 MOV     A,R19
   81 012D FAF3                                 SCAF                                            ;Frame Space > 9.00ms ?
   82 012E ECF1 E3FE                            JC      T9_UNDER                                ;       then  T9_UNDER: (More than 9.00ms)
   83                                                                                           ;       else            (Less than 9.00ms)
   84 0130 E2F1 E0E5                                                    HALT    #005H           ;HALT mode (Timer = 00H)
   85 0132 E6FF F7FF                                                    MOV     T,#TIME9M
   86 0134 E3F1 E0E5                                                    STTS    #0101B          ;Clear Status Flag
   87
   88 0136 FFE9                                                         MOV     A,R09
   89 0137 FCF3                                                         RL      A               ;Frame Space > 18.00ms ?
   90 0138 ECF1 E3FE                            JC      T9_UNDER                                ;       then T9_UNDER:  (Less than 18.00ms)
   91                                                                                           ;       else            (More than 18.00ms)
   92 013A E2F1 E0E5                                                    HALT    #005H           ;HALT mode (Timer = 00H)
   93 013C E6FF F7FF                                                    MOV     T,#TIME9M
   94
   95                                   T9_UNDER:
   96 013E E3F1 E0E5                            STTS    #0101B                                  ;Clear Status Flag
   97 0140 FEE9                                 MOV     A,R09
   98 0141 FBF1 F7E0                            ANL     A,#0111B
   99 0143 E5E0                                 MOV     R00,A
  100 0144 E2F1 E0E5                            HALT    #005H                                   ;HALT mode (Timer = 00H)
  101 0146 E7FF                                 MOV     T,@R0
  102 0147 E3F1 E0E5                            STTS    #0101B                                  ;Clear Status Flag
  103
  104 0149 FFE3                                 MOV     A,R03
  105 014A FAF3                                 SCAF                                            ;Continuously pressed key ?
  106 014B EDF1 E5F8                            JNC     KEY_ON                                  ;       else  KEY_ON:   (1st. Frame)
  107 014D FFE7                                                         MOV     A,R07           ;       then            (Since 2nd. Frame)
  108 014E FAF3                                                         SCAF                    ;Without input Key ?
  109 014F EDF1 E1E3                                                    JNC     KEY48           ;       then  KEY48:    (With input Key)
  110                                                           ;**  Frame Space  = 27ms  **    ;       else            (Without input Key)
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-004



SOURCE = TRANS48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   111 0151 EXTRN                                                       CALL    FS_27MS
   112 0154 E2F1 E0E5                                                   HALT    #005H           ;HALT mode (Timer = 00H)
   113 0156 E8F1 E0E0                                                   JMP     MAIN
   114                                                                  ;
   115                                  KEY_ON:
   116 0158 FEE7                                MOV     A,R17
   117 0159 E5E3                                MOV     R03,A                                   ;Continuance Flag (R03) = 0FH (R17=0FH)
   118                                    ;**  Frame Space  = 27ms  **
   119 015A EXTRN                               CALL    FS_27MS
   120 015D E8F1 EFE4                           JMP     TRNS
   121                                          ;
   122
   123
   124                                  ;*************************************************
   125                                  ;*****                                       *****
   126                                  ;*****               Transmit                *****
   127                                  ;*****         Subroutine : Bit Out          *****
   128                                  ;*****                                       *****
   129                                  ;*************************************************
   130                                  BITOUT0F:
   131 015F E4EA                                MOV     R1A,A                                   ;Set Output Data (1 Bit Output)
   132 0160 FFF1 FFE0                           MOV     A,#0FH                                  ;Send Bit Counter = 1 Times (=0FH)
   133 0162 E8F1 E6F7                                                   JMP     BITOUT
   134                                                                  ;
   135                                  BITOUT0C:
   136 0164 E4EA                                MOV     R1A,A                                   ;Set Output Data (4 Bit Output)
   137 0165 FFF1 FCE0                           MOV     A,#0CH                                  ;Send Bit Counter = 4 Times (=0CH)
   138
   139                                  BITOUT:
   140 0167 E5EA                                MOV     R0A,A                                   ;Send Bit Counter (R0A) = 0CH or 0EH or 0FH
   141
   142 0168 FEEA                                MOV     A,R1A
   143 0169 FCF3                                RL      A
   144 016A E4EA                                MOV     R1A,A                                   ;Bit Data = 1 ?
   145 016B ECF1 E7F1                           JC      BIT_DAT1                                ;       then  BIT_DAT1: (Bit Data = 1)
   146                                                                  ;**  Bit Data = 0  **   ;       else            (Bit Data = 0)
   147 016D FFE8                                                        MOV     A,R08           ;Set Data Pointer L (R00) <-- Data0 (R08)
   148 016E E5E0                                                        MOV     R00,A
   149 016F E8F1 E7FB                                                   JMP     BIT_TRNS
   150                                                                  ;
   151                                      ;**  Bit Data = 1  **;
   152                                  BIT_DAT1:
   153 0171 FEE8                                MOV     A,R18                                   ;Set Data Pointer L(R00) <-- Data1 (R18)
   154 0172 E5E0                                MOV     R00,A
   155 0173 FFE9                                MOV     A,R09
   156 0174 F4F3                                INC     A                                       ;Data 1 Transmit Counter L (R09) increment
   157 0175 E5E9                                MOV     R09,A                                   ;Data 1 Transmit Counter L (R09) = Overflow ?
   158 0176 EDF1 E7FB                           JNC     BIT_TRNS                                ;       else  BIT_TRNS: (Data 1 Transmit Counter L <= 0FH)
   159 0178 FEE9                                                        MOV     A,R19            ;       then            (Data 1 Transmit Counter L > 0FH)
   160 0179 F4F3                                                        INC     A               ;Data 1 Transmit Counter H(R19) increment
   161 017A E4E9                                                        MOV     R19,A
   162                                  BIT_TRNS:
   163 017B E2F1 E0E5                           HALT    #005H                                   ;HALT mode (Timer = 00H)
   164 017D E7FF                                MOV     T,@R0
   165 017E E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-005



SOURCE = TRANS48.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   166 0180 FFE0                                MOV     A,R00
   167 0181 F4F3                                INC     A                                       ;Data Pointer L (R00) increment
   168 0182 E5E0                                MOV     R00,A
   169 0183 E2F1 E0E5                           HALT    #005H                                   ;HALT mode (Timer = 00H)
   170 0185 E7FF                                MOV     T,@R0
   171
   172 0186 FFE7                                MOV     A,R07
   173 0187 F4F3                                INC     A                                       ;Key Off Check Counter (R07) increment
   174 0188 ECF1 E9F3                                                   JC      BIT_TRANS2      ;Key Off Check Counter (R07) = End of 10 Times ?
   175                                                                                          ;       then  BIT_TRANS2: (End of 10 Times)
   176                                                                                          ;       else             (Less than 10 Times)
   177 018A E3F1 E0EE                           STTS    #1110B                                  ;With input key ?
   178 018C EFF1 E9F2                           JNF     BIT_TRANS1                              ;       else  BIT_TRANS1: (With input)
   179                                                                  ;**  With input key  ** ;       then             (Without input)
   180 018E E6E7 EFE5                                                   MOV     R7,#0F5H        ;Key off Check Counter (R07) = 05H
   181 0190 E8F1 E9F3                                                   JMP     BIT_TRANS2
   182                                                                  ;
   183                                      ;**  Without input key  **
   184                                  BIT_TRANS1:
   185 0192 E5E7                                MOV     R07,A
   186
   187                                  BIT_TRANS2:
   188 0193 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
   189 0195 FFEA                                MOV     A,R0A
   190 0196 F4F3                                INC     A                                       ;Send Bit Counter (R0A) increment
   191 0197 EDF1 E6F7                           JNC     BITOUT
   192 0199 E8F2                                RET
   193                                          ;
   194                                  END


TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 04-002



SOURCE = DTABLE48.TBL

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
    1                                   ;*************************************************
    2                                   ;*****                                       *****
    3                                   ;*****       uPD6133 Series  Data Table      *****
    4                                   ;*****   Trans. CODE: NEC-R  Format (48Key)  *****
    5                                   ;*****                                       *****
    6                                   ;*************************************************
    7      01FF                                 TIME9M          EQU     1FFH                    ;9.00ms (9.002ms)
    8
    9                                   ;##########     P  U  B  L  I  C      ##########
   10                                           PUBLIC          DATACODE
   11                                           PUBLIC          FS_27MS
   12                                           ;
   13
   14                                   DATACODE:
   15                                   ;**************************************
   16                                   ;*****         Key Data          *****
   17                                   ;*****        Double Key         *****
   18                                   ;**************************************
   19 019D                              ORG     19DH
   20                                     ;** K29 + K30 **
   21 019D E0E9                                 DW      009H            ;K29 + K30 : Key Data = 0DH
   22                                     ;** K29 + K31 **
   23 019E E8E9                                 DW      089H            ;K29 + K31 : Key Data = 0EH
   24                                     ;** K29 + K32 **
   25 019F E4E9                                 DW      049H            ;K29 + K32 : Key Data = 0FH
   26                                           ;
   27
   28                                   ;**************************************
   29                                   ;*****         Key Data          *****
   30                                   ;*****        Single Key         *****
   31                                   ;**************************************
   32 01A0                              ORG     1A0H
   33                                     ;** K 1 - K16 **
   34 01A0 E0E0                                 DW      000H            ;K 1 : Key Data = 20H
   35 01A1 E8E0                                 DW      080H            ;K 2 : Key Data = 21H
   36 01A2 E4E0                                 DW      040H            ;K 3 : Key Data = 22H
   37 01A3 ECE0                                 DW      0C0H            ;K 4 : Key Data = 23H
   38 01A4 E2E0                                 DW      020H            ;K 5 : Key Data = 24H
   39 01A5 EAE0                                 DW      0A0H            ;K 6 : Key Data = 25H
   40 01A6 E6E0                                 DW      060H            ;K 7 : Key Data = 26H
   41 01A7 EEE0                                 DW      0E0H            ;K 8 : Key Data = 27H
   42 01A8 E1E0                                 DW      010H            ;K 9 : Key Data = 28H
   43 01A9 E9E0                                 DW      090H            ;K10 : Key Data = 29H
   44 01AA E5E0                                 DW      050H            ;K11 : Key Data = 2AH
   45 01AB EDE0                                 DW      0D0H            ;K12 : Key Data = 2BH
   46 01AC E3E0                                 DW      030H            ;K13 : Key Data = 2CH
   47 01AD EBE0                                 DW      0B0H            ;K14 : Key Data = 2DH
   48 01AE E7E0                                 DW      070H            ;K15 : Key Data = 2EH
   49 01AF EFE0                                 DW      0F0H            ;K16 : Key Data = 2FH
   50                                           ;
   51
   52                                     ;** K17 - K32 **
   53 01B0 E0E8                                 DW      008H            ;K17 : Key Data = 30H
   54 01B1 E8E8                                 DW      088H            ;K18 : Key Data = 31H
   55 01B2 E4E8                                 DW      048H            ;K19 : Key Data = 32H
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 04-003



SOURCE = DTABLE48.TBL

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
    56 01B3 ECE8                                DW      0C8H            ;K20 : Key Data = 33H
    57 01B4 E2E8                                DW      028H            ;K21 : Key Data = 34H
    58 01B5 EAE8                                DW      0A8H            ;K22 : Key Data = 35H
    59 01B6 E6E8                                DW      068H            ;K23 : Key Data = 36H
    60 01B7 EEE8                                DW      0E8H            ;K24 : Key Data = 37H
    61 01B8 E1E8                                DW      018H            ;K25 : Key Data = 38H
    62 01B9 E9E8                                DW      098H            ;K26 : Key Data = 39H
    63 01BA E5E8                                DW      058H            ;K27 : Key Data = 3AH
    64 01BB EDE8                                DW      0D8H            ;K28 : Key Data = 3BH
    65 01BC E3E8                                DW      038H            ;K29 : Key Data = 3CH
    66 01BD EBE8                                DW      0B8H            ;K30 : Key Data = 3DH
    67 01BE E7E8                                DW      078H            ;K31 : Key Data = 3EH
    68 01BF EFE8                                DW      0F8H            ;K32 : Key Data = 3FH
    69                                          ;
    70                                    ;** K33 - K48 **
    71 01C0 E0E4                                DW      004H            ;K33 : Key Data = 40H
    72 01C1 E8E4                                DW      084H            ;K34 : Key Data = 41H
    73 01C2 E4E4                                DW      044H            ;K35 : Key Data = 42H
    74 01C3 ECE4                                DW      0C4H            ;K36 : Key Data = 43H
    75 01C4 E2E4                                DW      024H            ;K37 : Key Data = 44H
    76 01C5 EAE4                                DW      0A4H            ;K38 : Key Data = 45H
    77 01C6 E6E4                                DW      064H            ;K39 : Key Data = 46H
    78 01C7 EEE4                                DW      0E4H            ;K40 : Key Data = 47H
    79 01C8 E1E4                                DW      014H            ;K41 : Key Data = 48H
    80 01C9 E9E4                                DW      094H            ;K42 : Key Data = 49H
    81 01CA E5E4                                DW      054H            ;K43 : Key Data = 4AH
    82 01CB EDE4                                DW      0D4H            ;K44 : Key Data = 4BH
    83 01CC E3E4                                DW      034H            ;K45 : Key Data = 4CH
    84 01CD EBE4                                DW      0B4H            ;K46 : Key Data = 4DH
    85 01CE E7E4                                DW      074H            ;K47 : Key Data = 4EH
    86 01CF EFE4                                DW      0F4H            ;K48 : Key Data = 4FH
    87                                          ;
    88
    89                                  ;***************************************
    90                                  ;*****        Frame Space         *****
    91                                  ;*****         Time  Data         *****
    92                                  ;ORG 1D0H ******************************
    93 01D0 E7ED                                DT      1F4H            ;Counter:000 = 8.815ms (8.809ms)
    94 01D1 E6ED                                DT      1B4H            ;Counter:001 = 7.690ms (7.683ms)
    95 01D2 E5ED                                DT      174H            ;Counter:010 = 6.565ms (6.558ms)
    96 01D3 E4ED                                DT      134H            ;Counter:011 = 5.440ms (5.433ms)
    97 01D4 E3ED                                DT      0F4H            ;Counter:100 = 4.315ms (4.308ms)
    98 01D5 E2ED                                DT      0B4H            ;Counter:101 = 3.190ms (3.182ms)
    99 01D6 E1ED                                DT      074H            ;Counter:110 = 2.065ms (2.057ms)
   100 01D7 E0ED                                DT      034H            ;Counter:111 = 0.940ms (0.932ms)
   101                                          ;
   102
   103                                  ;***************************************
   104                                  ;*****         Time  Data         *****
   105                                  ;ORG 1D8H ******************************
   106                                    ;** Leader Code **
   107 01D8 FFFF                                DT      3FFH            ;Carrier ON  : 9.000ms (9.002ms)
   108 01D9 F3FF                                DT      0FFH            ;Carrier OFF : 4.500ms (4.501ms)
   109                                    ;** Bit data 0 **
   110 01DA F8F7                                DT      21FH            ;Carrier ON  : 0.560ms (0.563ms)
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 04-004



SOURCE = DTABLE48.TBL

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   111 01DB F0F7                                DT      01FH            ;Carrier OFF : 0.565ms (0.563ms)
   112                                    ;** Bit data 1 **
   113 01DC F8F7                                DT      21FH            ;Carrier ON  : 0.560ms (0.563ms)
   114 01DD F1F7                                DT      05FH            ;Carrier OFF : 1.690ms (1.687ms)
   115                                    ;** Stop Bit **
   116 01DE F8F7                                DT      21FH            ;Carrier ON  : 0.560ms (0.563ms)
   117 01DF E2FA                                DT      0A9H            ;Carrier OFF : 3.000ms (2.989ms)
   118                                          ;
   119
   120
   121
   122                                  ;*************************************************
   123                                  ;*****                                       *****
   124                                  ;*****           Frame Space = 27ms          *****
   125                                  ;*****          Subroutine : FS 27ms         *****
   126                                  ;*****                                       *****
   127                                  ;*************************************************
   128
   129                                  FS_27MS:
   130 01E0 FFF1 FDE0                           MOV     A,#0DH                                  ;Counter(Acc) = 3 Times (=0DH)
   131                                  FS_27MS0:
   132 01E2 E2F1 E0E5                           HALT    #005H                                   ;HALT mode (Timer = 00H)
   133 01E4 E6FF F7FF                           MOV     T,#TIME9M                               ;Timer = 9.00ms
   134 01E6 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
   135 01E8 F4F3                                INC     A                                       ;More than 27ms ?
   136 01E9 EDF1 EEF2                           JNC     FS_27MS0                                ;       else  FS_27MS0: (Frame Space < 27ms)
   137 01EB E8F2                                RET                                             ;       then  RET       (Frame Space = 27ms)
   138                                          ;
   139                                  END


TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     21:25:49 06/03/96 PAGE 05-002



SOURCE = OPTION.ASM

E STNO LOC. OBJ.                M  I   SOURCE STATEMENT
    1                                  ;**************************************
    2                                  ;*****         Set Option         *****
    3                                  ;**************************************
    4                                  OPTION
    5 0000 0001                                USEP0C
    6                                  ENDOP
    7                                  END


TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

[MEMO]

PART 3

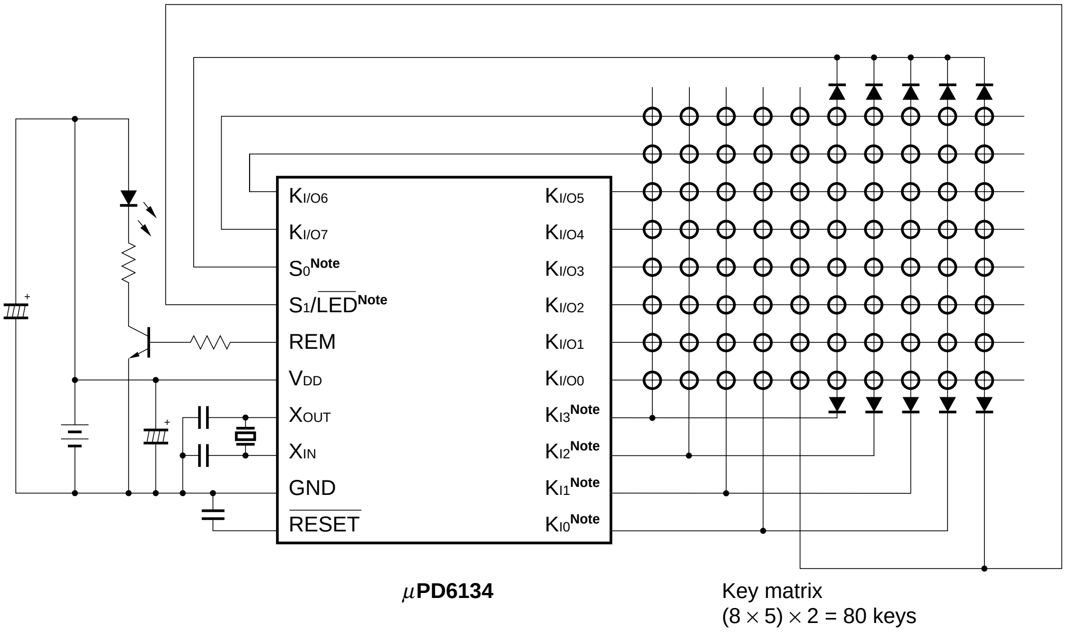
80-KEY PROGRAM

[MEMO]

1.1 Application Circuit Example

Figure 1-1 shows an application circuit example.

Figure 1-1. Application Circuit Example



Note The program is set for “on-chip pull-down resistors”.

The K_I, $K_{I/O}$, S_0, and $S_1/\overline{LED}$ pins are used to configure an 80-key key matrix. The transmission code is output via the REM pin with a carrier signal.

Table 1-1 lists the various pin functions.

Table 1-1. Pin Functions

Pin name	I/O	Function
$K_{I/O0}$ to K_{IO7}	Output	Key source (active high)
K_{I0} to K_{I3}	Input	Key return (active high)
S_0	Input	Key expansion
$S_1/\overline{LED}$	Input	Key return (active high)
REM	Output	Infrared remote control signal (with carrier)

1.2 Key Matrix

Figure 1-2 illustrates the key matrix.

The “Kn” symbol (in which n = 1 to 80) indicates each key’s position.

Figure 1-2. Key Matrix

$K_{I/O7}$	K29	K30	K31	K32	K40	K69	K70	K71	K72	K80
$K_{I/O6}$	K25	K26	K27	K28	K39	K65	K66	K67	K68	K79
$K_{I/O5}$	K21	K22	K23	K24	K38	K61	K62	K63	K64	K78
$K_{I/O4}$	K17	K18	K19	K20	K37	K57	K58	K59	K60	K77
$K_{I/O3}$	K13	K14	K15	K16	K36	K53	K54	K55	K56	K76
$K_{I/O2}$	K9	K10	K11	K12	K35	K49	K50	K51	K52	K75
$K_{I/O1}$	K5	K6	K7	K8	K34	K45	K46	K47	K48	K74
$K_{I/O0}$	K1	K2	K3	K4	K33	K41	K42	K43	K44	K73
	K_{I3}	K_{I2}	K_{I1}	K_{I0}	S_1	K_{I3}	K_{I2}	K_{I1}	K_{I0}	S_1
	$S_0 = 0$					$S_0 = 1$ (expansion)				

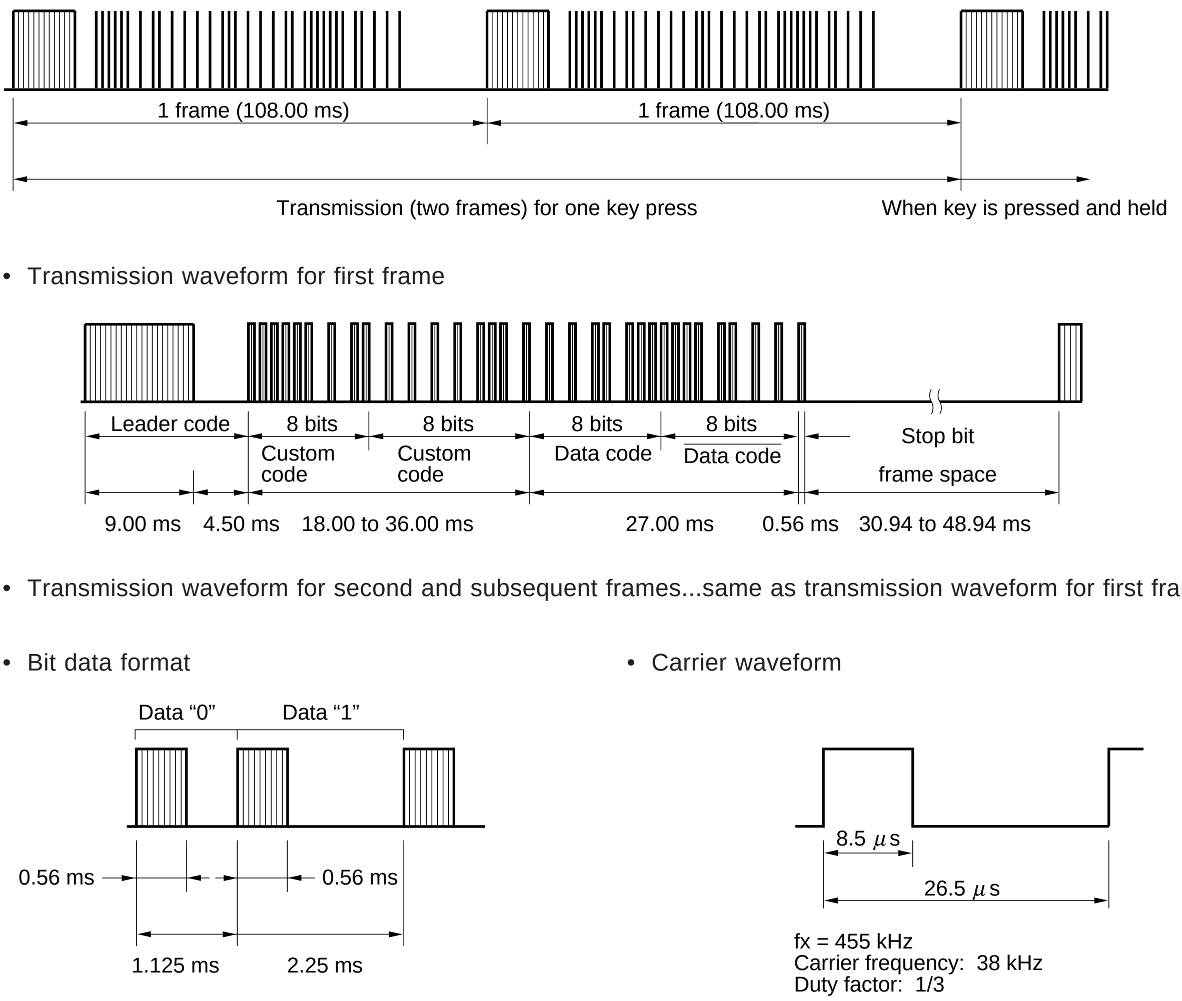
CHAPTER 2 TRANSMISSION WAVEFORM

The transmission waveform that is output from the REM pin uses the NEC-R (NEC continuous) format. For description of the output data code, see **CHAPTER 4 OUTPUT CODES**.

2.1 NEC-R Format

Figure 2-1 illustrates the NEC-R format.

Figure 2-1. NEC-R Format



- Transmission waveform

1 frame (108.00 ms)
1 frame (108.00 ms)
Transmission (two frames) for one key press
When key is pressed and held

- Transmission waveform for first frame

- Transmission waveform for second and subsequent frames...same as transmission waveform for first frame

- Bit data format

- Carrier waveform

fx = 455 kHz
Carrier frequency: 38 kHz
Duty factor: 1/3

[MEMO]

CHAPTER 3 TIMING CHARTS

3.1 Timing Charts for Key Input to REM Output

ON chattering elimination processing checks for key input every 9.00 ms and if it detects ON status three consecutive times, it determines that the key is ON. “ON chattering” elimination processing also checks for key status changes (between ON and OFF or when the key is pressed and held).

The OFF chattering elimination processing is described below.

Key OFF status is checked during low-level output of the bit data.

During transmission of one frame (108 ms), key input is checked ten times with reference to the timing (34 times) of the low-level output from the REM pin.

(1) If key OFF status is detected all ten times
... Key OFF status is determined.

(2) If key ON status is detected during at least one of the ten times
... When key ON status is detected, the check counter is cleared and key input is checked another ten times.

(3) If (1) and (2) above do not determine key OFF status
... Key press and hold status (key ON status) is determined.

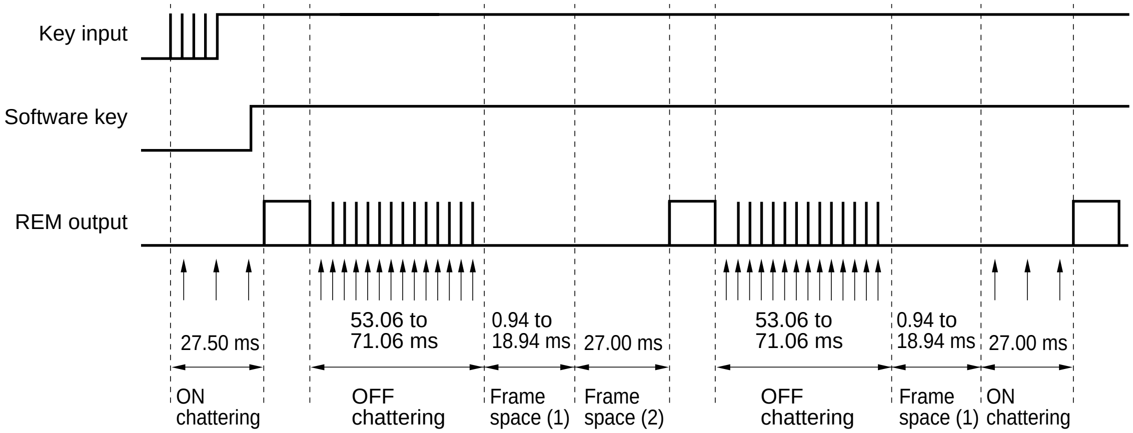
Even if key OFF status is determined during transmission of one frame, initialization processing does not begin until after the second or a subsequent frame is transmitted.

3.1.1 Timing when a key is pressed and held

Figure 3-1 shows a timing chart for when a key is pressed and held.

For details of ON chattering elimination, see **6.2.1 (1) Chattering elimination processing**.

Figure 3-1. Timing Chart when Key is Pressed and Held



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1) : This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

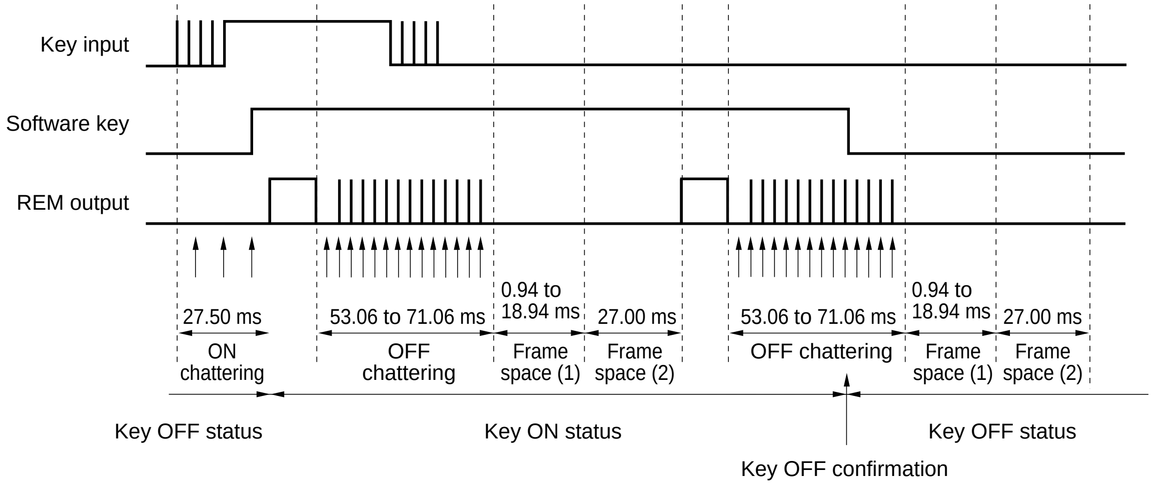
Frame space (2) : When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

3.1.2 Timing during key OFF status

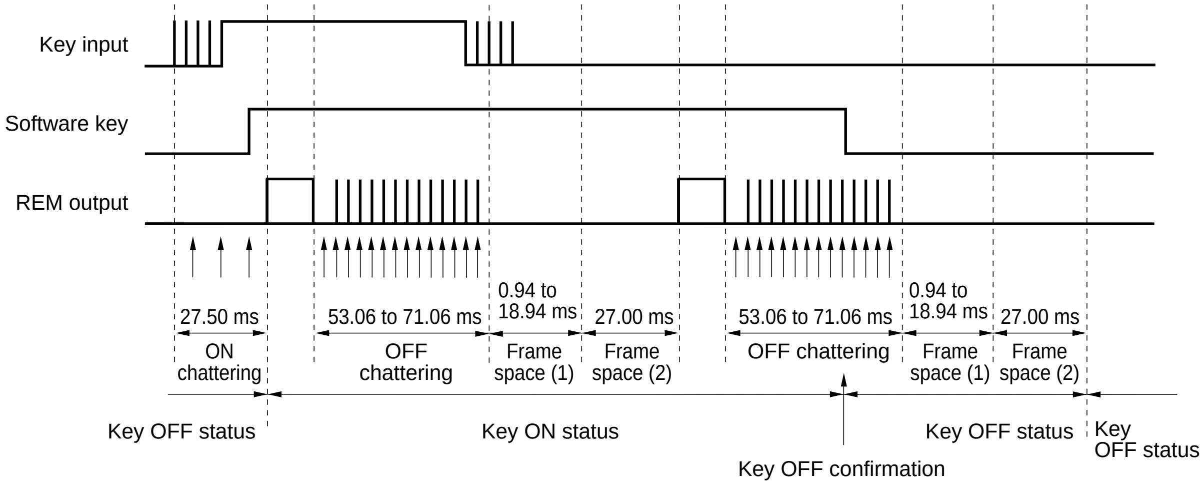
Figure 3-2 shows timing charts for when a key is released during a frame transmission.

Figure 3-2. Timing Chart for Key OFF Status (1/3)

(1) When key is released during transmission of the first frame (key OFF status has been determined ten times consecutively during transmission of the first frame)
... Key OFF status is confirmed during transmission of second frame



(2) When key is released during transmission of the first frame (key OFF status has not been determined ten times consecutively during transmission of the first frame)
... Key OFF status is confirmed during transmission of second frame



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

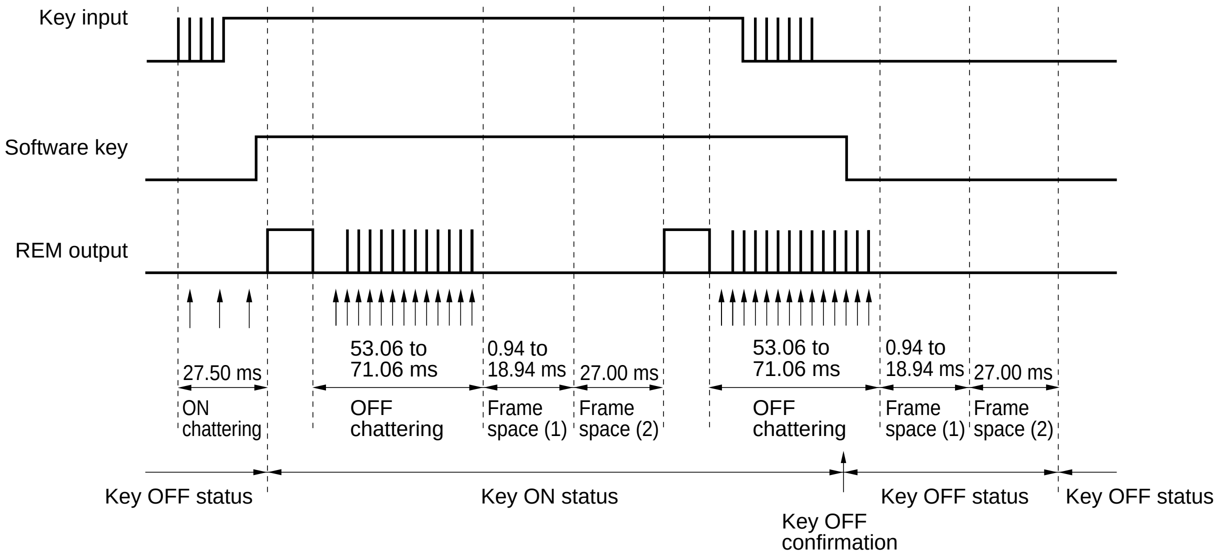
OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1) : This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

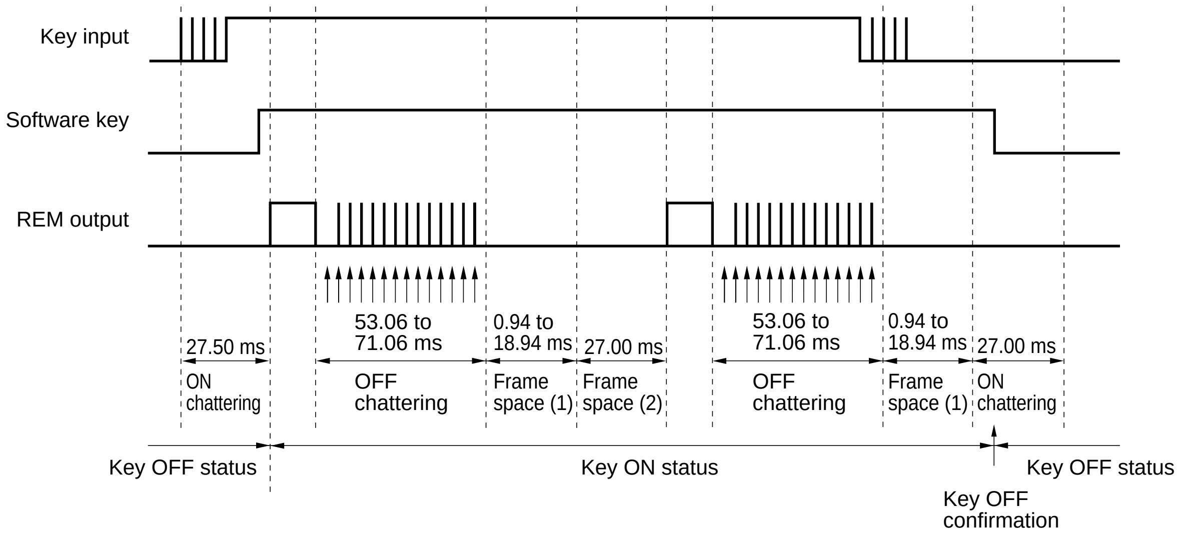
Frame space (2) : When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

Figure 3-2. Timing Chart for Key OFF Status (2/3)

(3) When key is released during transmission of the second frame (key OFF status has been determined ten times consecutively during transmission of the second frame)
... Key OFF status is confirmed during transmission of second frame



(4) When key is released during transmission of the second frame (key OFF status has not been determined ten times consecutively during transmission of the second frame)
... Key OFF status is confirmed by the key's ON chattering



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

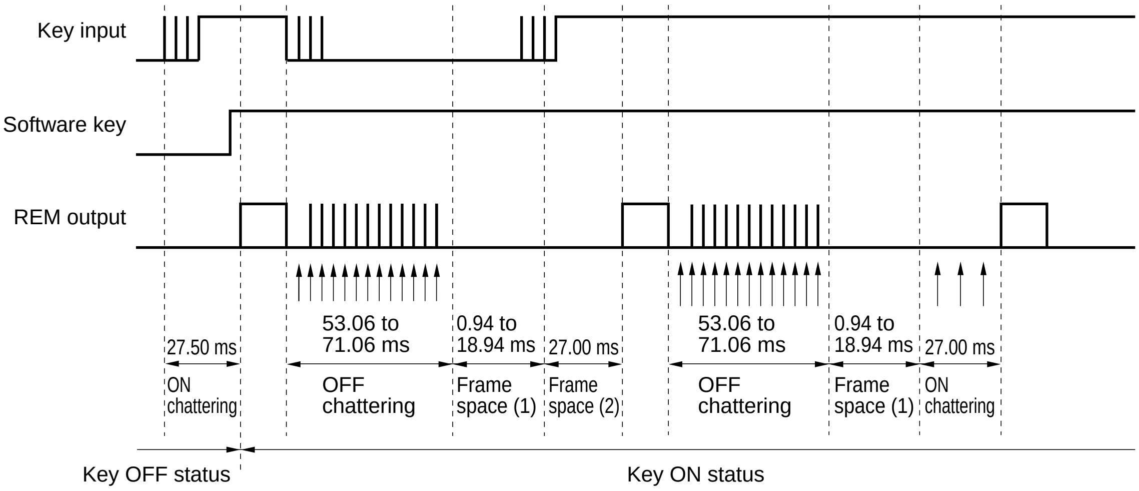
OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1) : This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

Frame space (2) : When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

Figure 3-2. Timing Chart for Key OFF Status (3/3)

(5) When key is released during transmission of the first frame and is pressed again later
... Key OFF cannot be confirmed since a key ON status is detected during the second frame



Key input : Actual key operation (manually pressing or releasing a key)

Software key : Software-based key operation (key input judgment by program)

REM output : Transmission waveform output via REM pin

ON chattering : This refers to ON chattering elimination processing. Key input is checked three times at 9.00-ms intervals.

OFF chattering : This refers to OFF chattering elimination processing. Key input is checked throughout the (34-bit) bit space that includes the leader code and stop bit.

Frame space (1) : This frame space is used to maintain a one-frame time of 108.00 ms to modify the transmission time for custom codes.

Frame space (2) : When it is determined that the current frame is the first frame or that a key OFF status is in effect, frame space (2) is transmitted (during 27.00 ms) without any ON chattering elimination processing.

3.2 Timing Charts of Key Operations

3.2.1 Output patterns prior to key confirmation

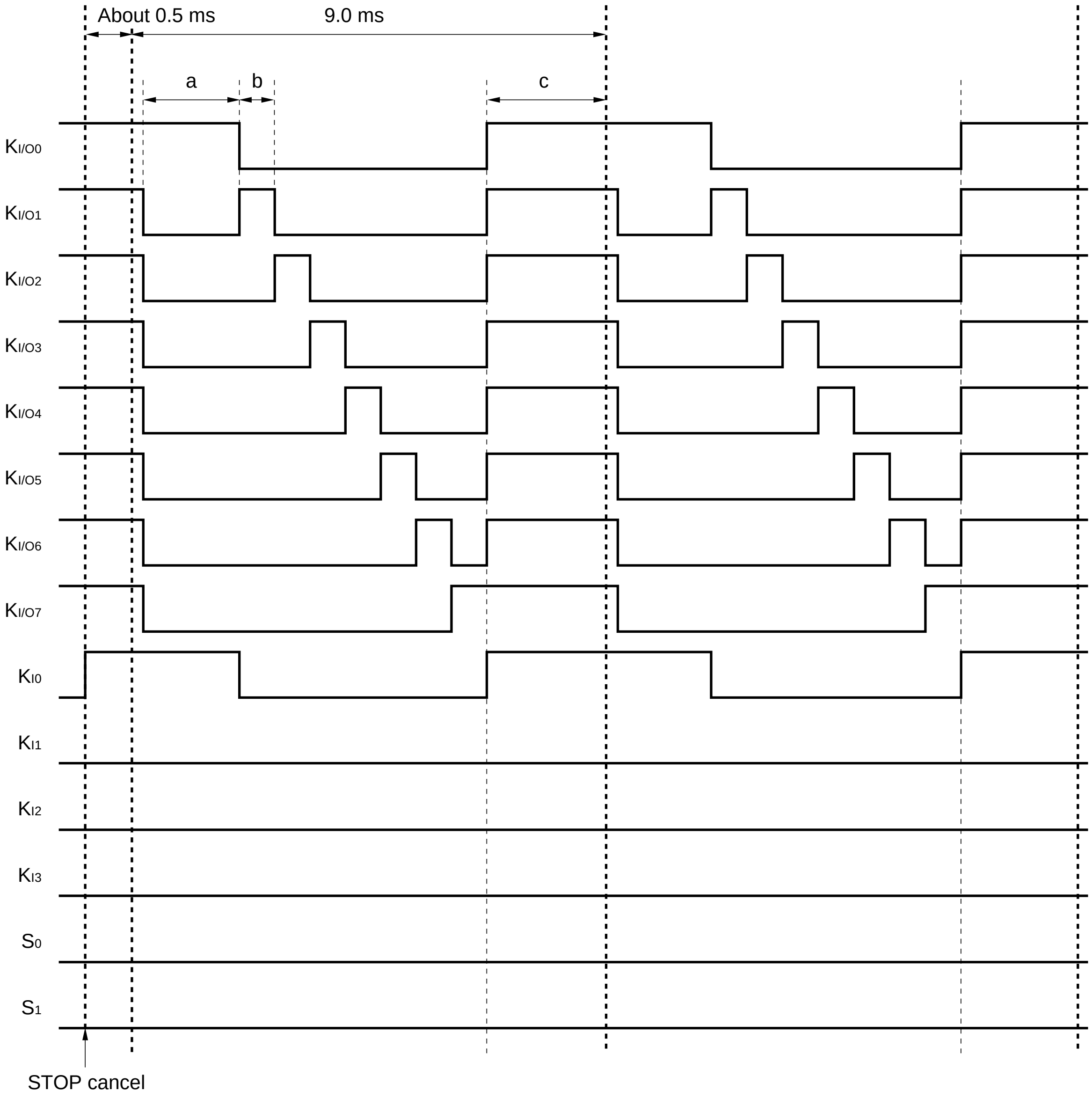
Prior to key confirmation, ON chattering elimination processing (9.00 ms × 3 times = 27.00 ms) is performed and a key is confirmed when it has the same status all three times.

Key input confirmation is determined after at least 100 μs (at least six instructions when operating at 455 kHz) has elapsed after the key source output pin goes to high level.

Figure 3-3 shows the detailed output patterns prior to key confirmation.

Figure 3-3. Output Patterns Prior to Key Confirmation (1/6)

(1) When K4 is pressed



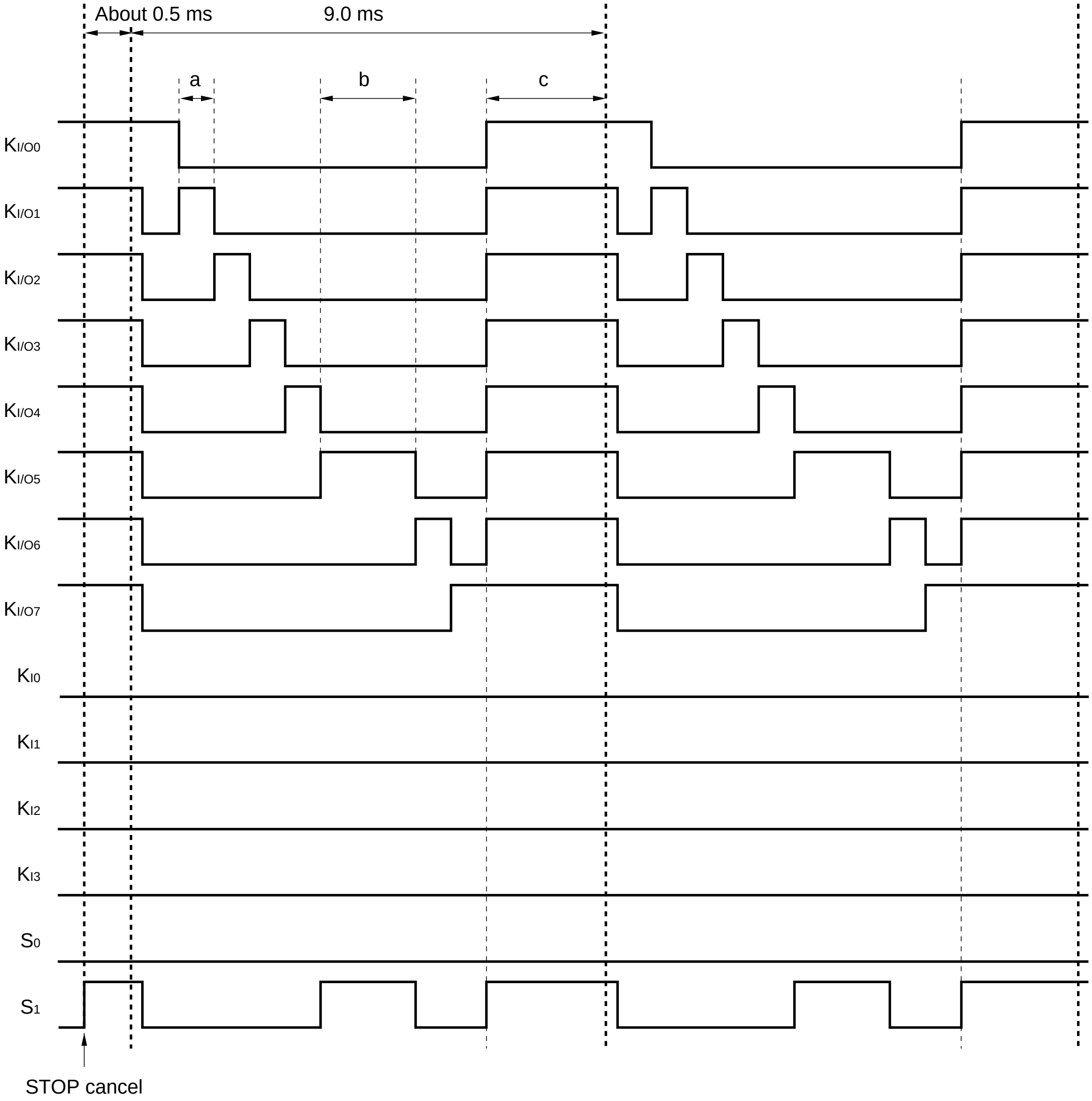
a : When a key ON status has been determined, the key return is checked and the key data is calculated.
Execution time ranges from approximately 1.32 to 2.02 ms.

b : Key input is checked.
Execution time is approximately 0.41 ms.

c : This is the total output time for key scanning. This time varies according to the value of “a”.
Execution time ranges from approximately 4.18 to 4.88 ms.

Figure 3-3. Output Patterns prior to Key Confirmation (2/6)

(2) When K38 is pressed



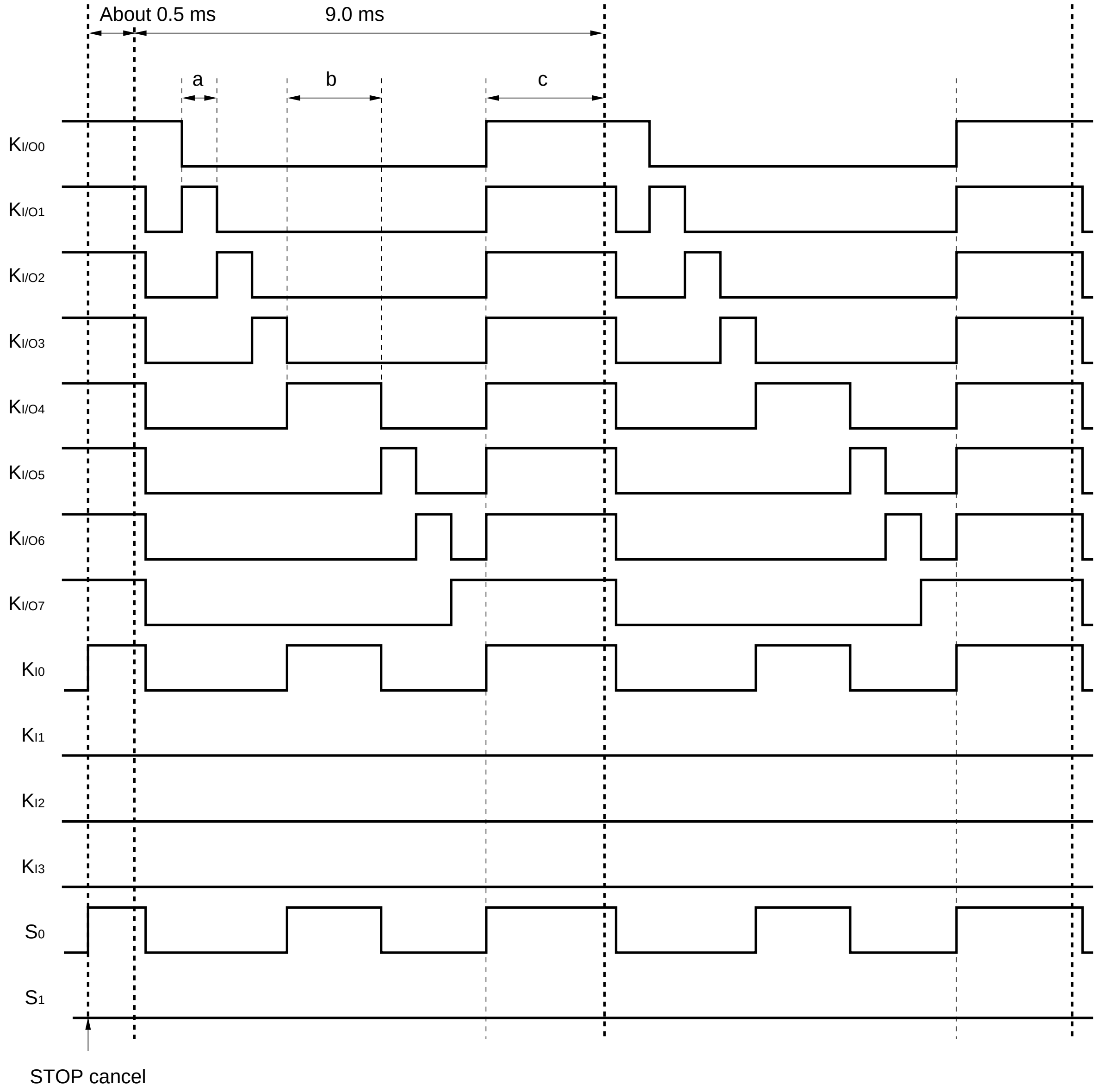
a : When a key ON status has been determined, the key return is checked and the key data is calculated.
Execution time ranges from approximately 1.32 to 2.02 ms.

b : Key input is checked.
Execution time is approximately 0.41 ms.

c : This is the total output time for key scanning. This time varies according to the value of “a”.
Execution time ranges from approximately 4.18 to 4.88 ms.

Figure 3-3. Output Patterns prior to Key Confirmation (3/6)

(3) When K60 is pressed



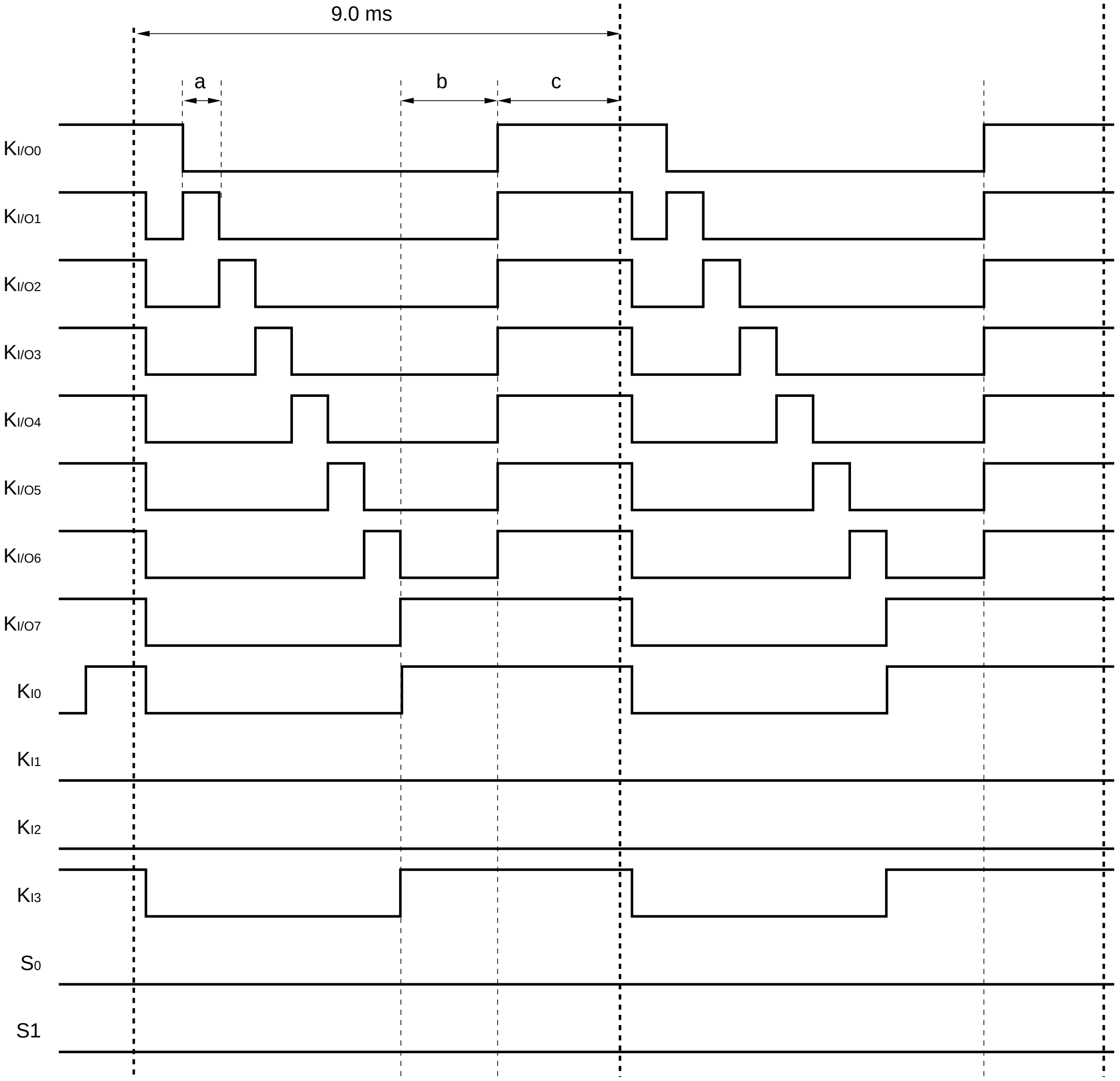
a : When a key ON status has been determined, the key return is checked and the key data is calculated.
Execution time ranges from approximately 1.32 to 2.02 ms.

b : Key input is checked.
Execution time is approximately 0.41 ms.

c : This is the total output time for key scanning. This time varies according to the value of “a”.
Execution time ranges from approximately 4.18 to 4.88 ms.

Figure 3-3. Output Patterns prior to Key Confirmation (4/6)

(4) When K29 is pressed along with valid combination key K32 (K29 + K32)



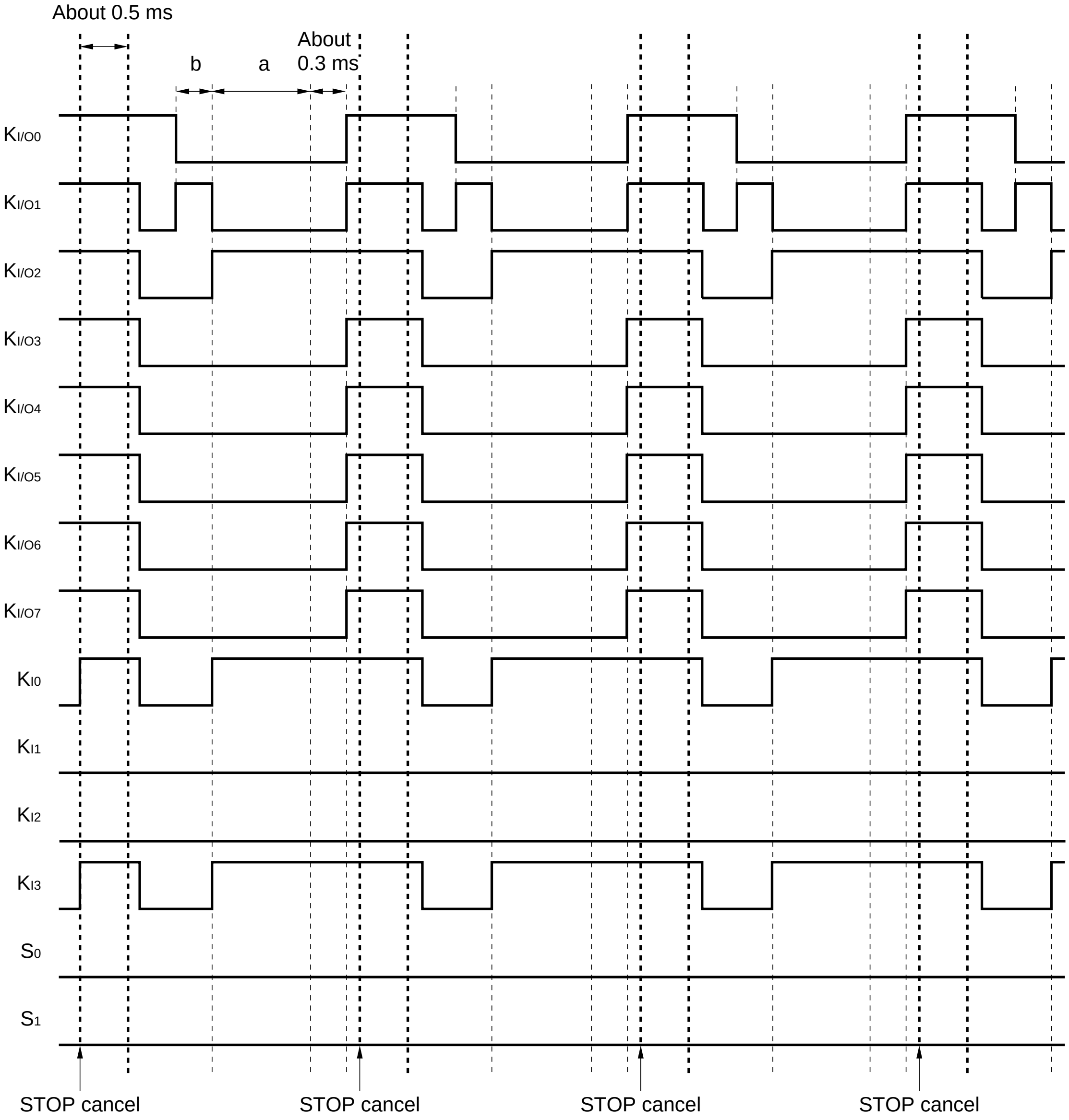
a : When a key ON status has been determined, the key return is checked and the key data is calculated. Execution time ranges from approximately 1.32 to 2.02 ms.

b : Key input is checked. Execution time is approximately 0.41 ms.

c : This is the total output time for key scanning. This time varies according to the value of “a”. Execution time ranges from approximately 4.18 to 4.88 ms.

Figure 3-3. Output Patterns prior to Key Confirmation (5/6)

(5) When K9 and K12 are pressed but are invalid due to an identical key source

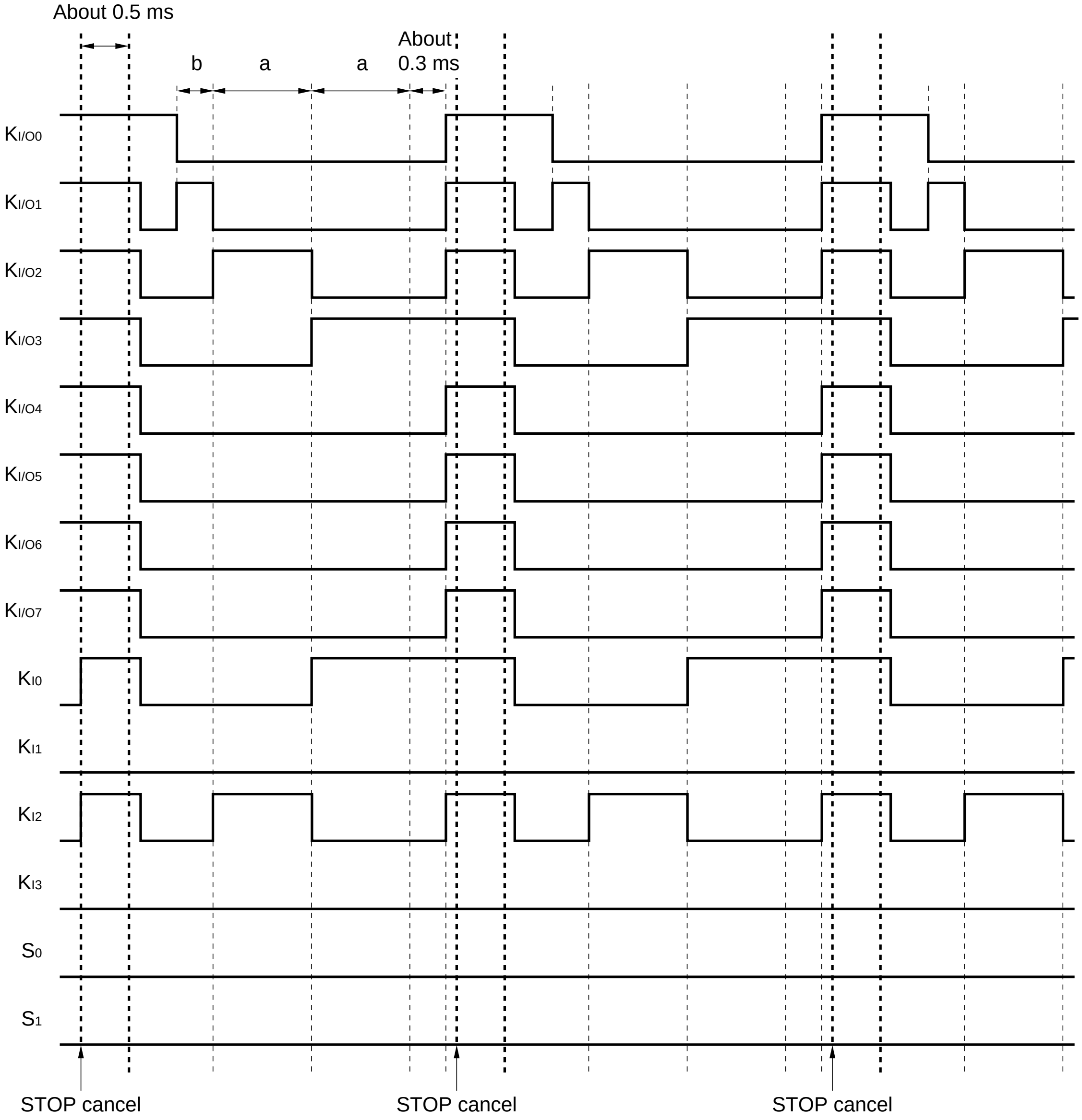


a : When a key ON status has been determined, the key return is checked and the key data is calculated.
Execution time ranges from approximately 1.32 to 2.02 ms.

b : Key input is checked.
Execution time is approximately 0.41 ms.

Figure 3-3. Output Patterns prior to Key Confirmation (6/6)

(6) When K10 and K16 are pressed but are invalid due to an identical key source



a : When a key ON status has been determined, the key return is checked and the key data is calculated.
Execution time ranges from approximately 1.32 to 2.02 ms.

b : Key input is checked.
Execution time is approximately 0.41 ms.

3.2.2 Operation of combination key

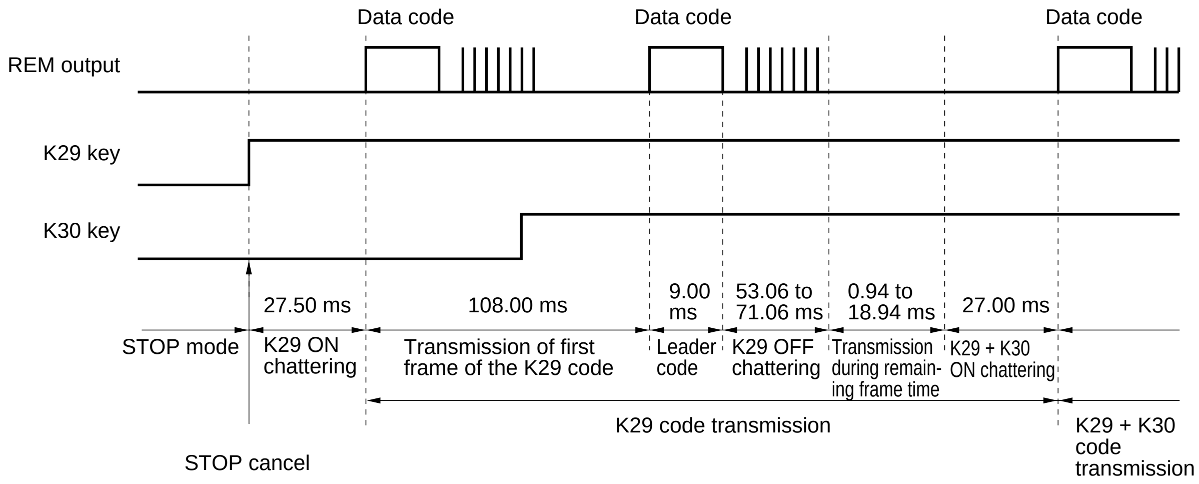
K29 is the combination key. There are three valid key combination pairs: K29 + K30, K29 + K31, and K29 + K32.

For any pattern, a key combination is valid only when K29 is pressed first. In other words, if another key is pressed before or at the same time as K29, the key combination is not valid.

Figure 3-4 shows examples of key operations when a key combination is entered.

Figure 3-4. Timing Chart of Key Combination Operation (1/2)

(1) If another valid key (from K30 to K32) is pressed during transmission of the first frame of the K29 code, after the second frame of the K29 code is transmitted, the key combination becomes valid and the combination's code is transmitted.



(2) If another valid key (from K30 to K32) is pressed during transmission of the third frame of the K29 code, after the K29 code and the third frame are transmitted, the key combination becomes valid and the combination's code is transmitted.

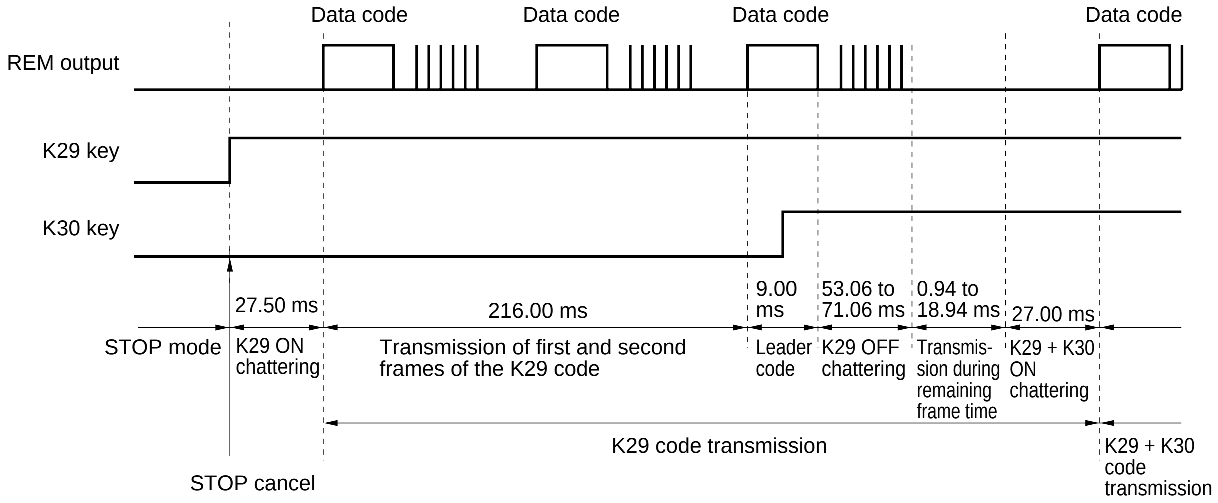
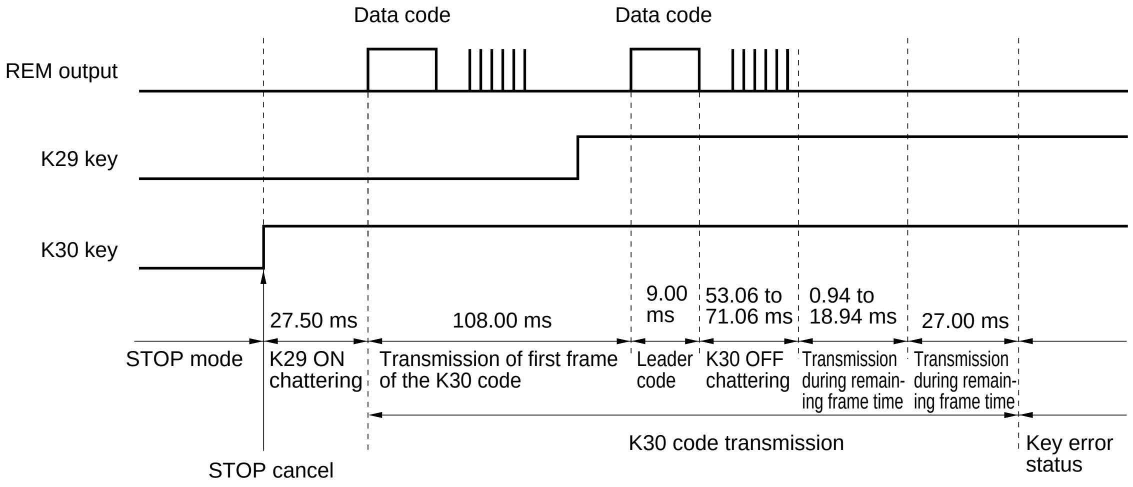
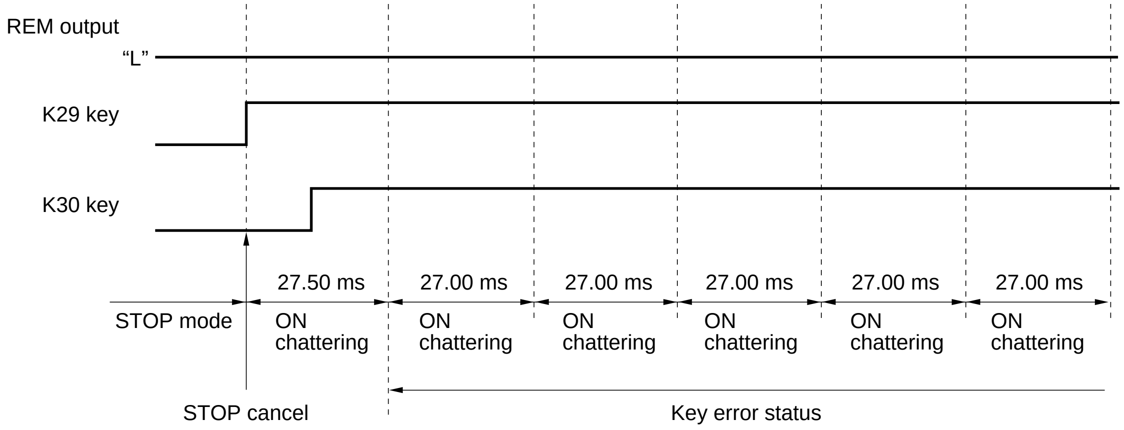


Figure 3-4. Timing Chart of Key Combination Operation (2/2)

(3) If K29 is pressed after another valid key (from K30 to K32) has been confirmed, a key check produces a key error and no code is transmitted (the key combination is not valid).



(4) If another valid key (from K30 to K32) is pressed before K29 has been confirmed, a key check produces a key error and no code is transmitted (the key combination is not valid).



3.2.3 Key transfer operation

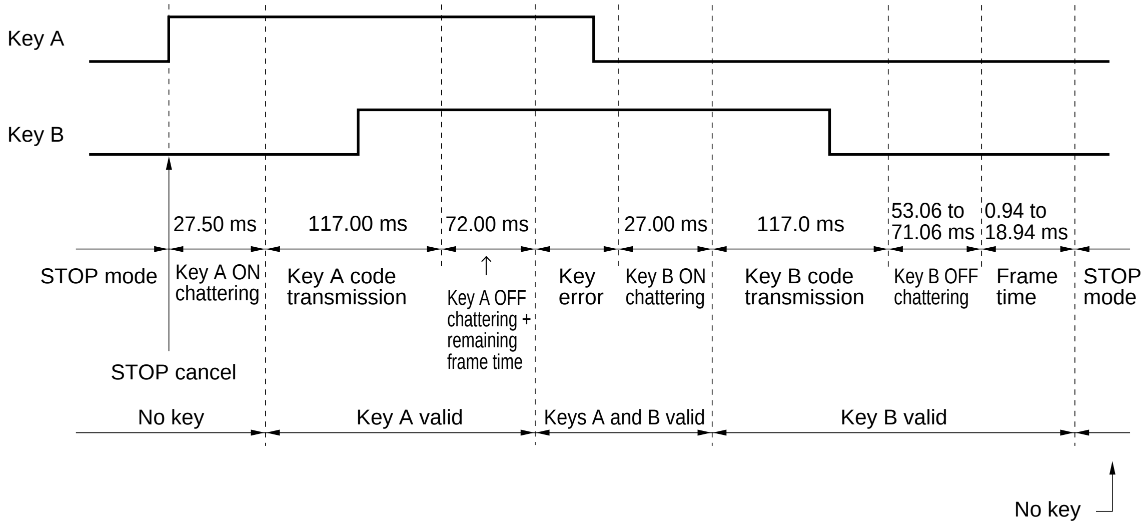
A key transfer operation can be performed when a valid combination key has been pressed.

The key transfer operation is an operation that occurs when a second key is pressed before a previously pressed key is released. The operation for the second key is performed when the previous key is released.

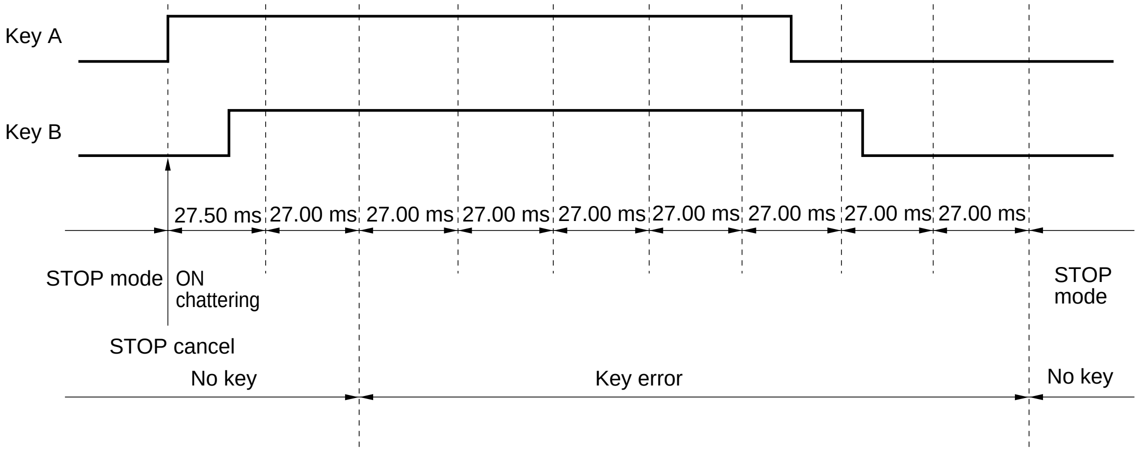
Figure 3-5 shows the operation timing of a key transfer operation.

Figure 3-5. Operation Timing of Key Transfer Operation

(1) When a key transfer operation is performed to transfer to key B during transmission of the code for key A, a key check is performed after all of the code has been transmitted. A key error occurs if it is during a key combination period.
When key A is released, key B becomes valid and its code is transmitted.



(2) If the transfer to key B occurs before key A is confirmed, a key error occurs during the key check. A key error also occurs if it is during a key combination period.
After key A is released, a key check is performed for key B. However, key B was released before it could be checked, so the check result is "no key".



CHAPTER 4 OUTPUT CODES

Table 4-1 lists the output codes.

For output using the NEC format, NEC provides each customer with a custom code to avoid the risk of interfering with output from another remote control unit that outputs using the NEC format. This program is set to output 0AH as the custom code and F5H as the custom code’.

Contact your NEC sales representative for information on obtaining a custom code.

Table 4-1. Output Codes (1/2)

Key No.	Custom code	Custom code’	Data	Key No.	Custom code	Custom code’	Data	Key No.	Custom code	Custom code’	Data
K1	0AH	F5H	00H	K28	0AH	F5H	1BH	K55	0AH	F5H	36H
K2	0AH	F5H	01H	K29	0AH	F5H	1CH	K56	0AH	F5H	37H
K3	0AH	F5H	02H	K30	0AH	F5H	1DH	K57	0AH	F5H	38H
K4	0AH	F5H	03H	K31	0AH	F5H	1EH	K58	0AH	F5H	39H
K5	0AH	F5H	04H	K32	0AH	F5H	1FH	K59	0AH	F5H	3AH
K6	0AH	F5H	05H	K33	0AH	F5H	20H	K60	0AH	F5H	3BH
K7	0AH	F5H	06H	K34	0AH	F5H	21H	K61	0AH	F5H	3CH
K8	0AH	F5H	07H	K35	0AH	F5H	22H	K62	0AH	F5H	3DH
K9	0AH	F5H	08H	K36	0AH	F5H	23H	K63	0AH	F5H	3EH
K10	0AH	F5H	09H	K37	0AH	F5H	24H	K64	0AH	F5H	3FH
K11	0AH	F5H	0AH	K38	0AH	F5H	25H	K65	0AH	F5H	40H
K12	0AH	F5H	0BH	K39	0AH	F5H	26H	K66	0AH	F5H	41H
K13	0AH	F5H	0CH	K40	0AH	F5H	27H	K67	0AH	F5H	42H
K14	0AH	F5H	0DH	K41	0AH	F5H	28H	K68	0AH	F5H	43H
K15	0AH	F5H	0EH	K42	0AH	F5H	29H	K69	0AH	F5H	44H
K16	0AH	F5H	0FH	K43	0AH	F5H	2AH	K70	0AH	F5H	45H
K17	0AH	F5H	10H	K44	0AH	F5H	2BH	K71	0AH	F5H	46H
K18	0AH	F5H	11H	K45	0AH	F5H	2CH	K72	0AH	F5H	47H
K19	0AH	F5H	12H	K46	0AH	F5H	2DH	K73	0AH	F5H	48H
K20	0AH	F5H	13H	K47	0AH	F5H	2EH	K74	0AH	F5H	49H
K21	0AH	F5H	14H	K48	0AH	F5H	2FH	K75	0AH	F5H	4AH
K22	0AH	F5H	15H	K49	0AH	F5H	30H	K76	0AH	F5H	48H
K23	0AH	F5H	16H	K50	0AH	F5H	31H	K77	0AH	F5H	4CH
K24	0AH	F5H	17H	K51	0AH	F5H	32H	K78	0AH	F5H	4DH
K25	0AH	F5H	18H	K52	0AH	F5H	33H	K79	0AH	F5H	4EH
K26	0AH	F5H	19H	K53	0AH	F5H	34H	K80	0AH	F5H	4FH
K27	0AH	F5H	1AH	K54	0AH	F5H	35H				

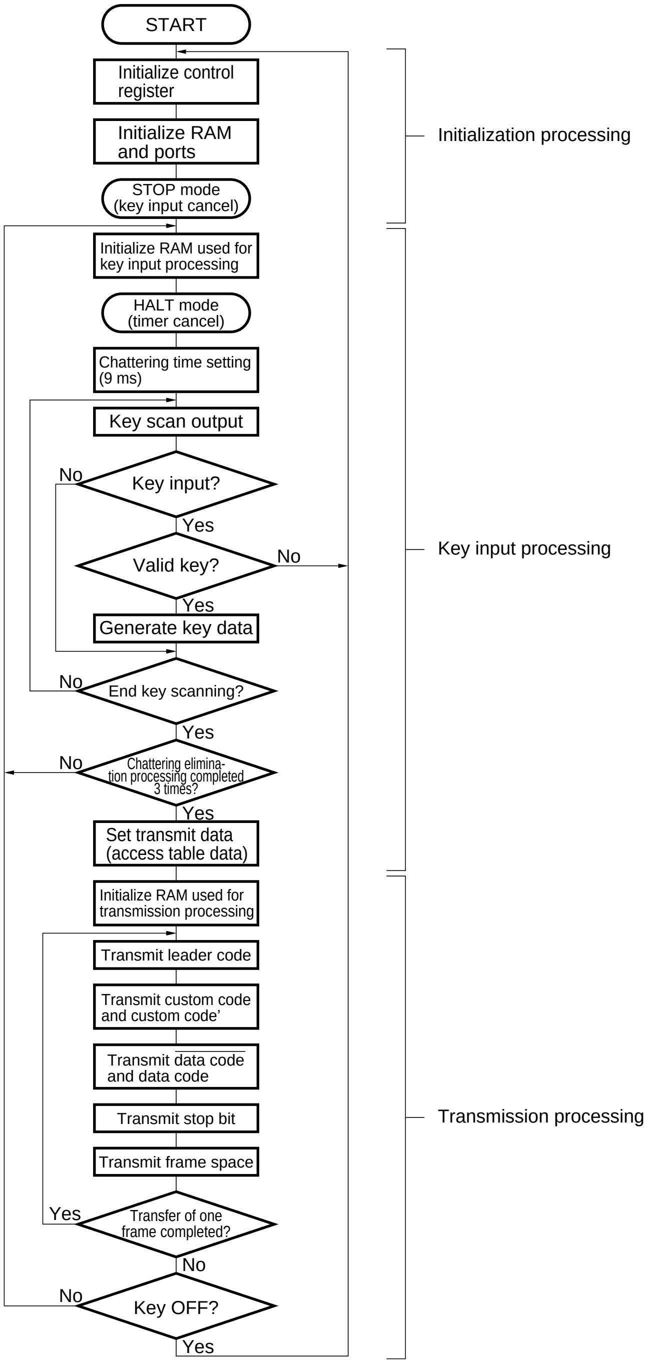
Table 4-1. Output Codes (2/2)

Key No.	Custom code	Custom code’	Data
K29+K30	0AH	F5H	90H
K29+K31	0AH	F5H	91H
K29+K32	0AH	F5H	92H

5.1 General Flow Chart

Figure 5-1 shows a general flow chart of this program.

Figure 5-1. General Flow Chart


START
Initialize control register
Initialize RAM and ports
STOP mode (key input cancel)
Initialization processing
Initialize RAM used for key input processing
HALT mode (timer cancel)
Chattering time setting (9 ms)
Key scan output
No
Key input?
Yes
Valid key?
No
Key input processing
Yes
Generate key data
No
End key scanning?
Yes
No
Chattering elimination processing completed 3 times?
Yes
Set transmit data (access table data)
Initialize RAM used for transmission processing
Transmit leader code
Transmit custom code and custom code'
Transmit data code and data code
Transmit stop bit
Transmission processing
Transmit frame space
Yes
Transfer of one frame completed?
No
No
Key OFF?
Yes

5.2 Program Memory (ROM) Configuration

The μPD6134's program memory (ROM) consists of 1002 steps x 10 bits (the 22 steps from 3EAH to 3FFH comprise the test program area).

Figure 5-2 shows a ROM map for this program and Figure 5-3 shows a data table map.

Figure 5-2. ROM Map

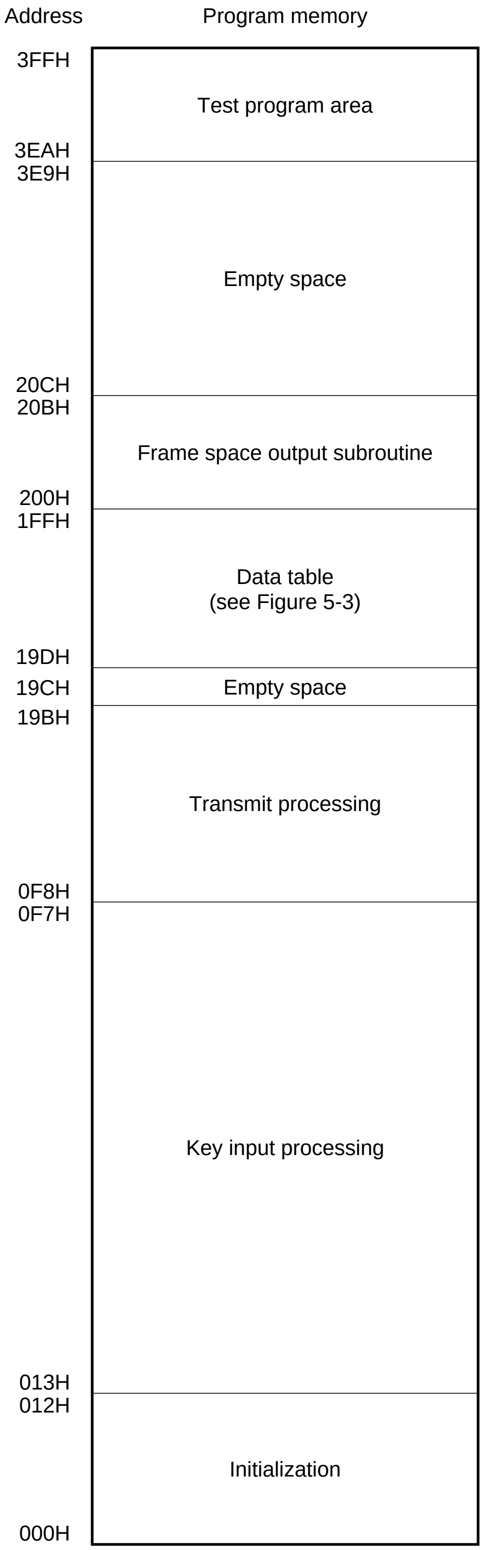
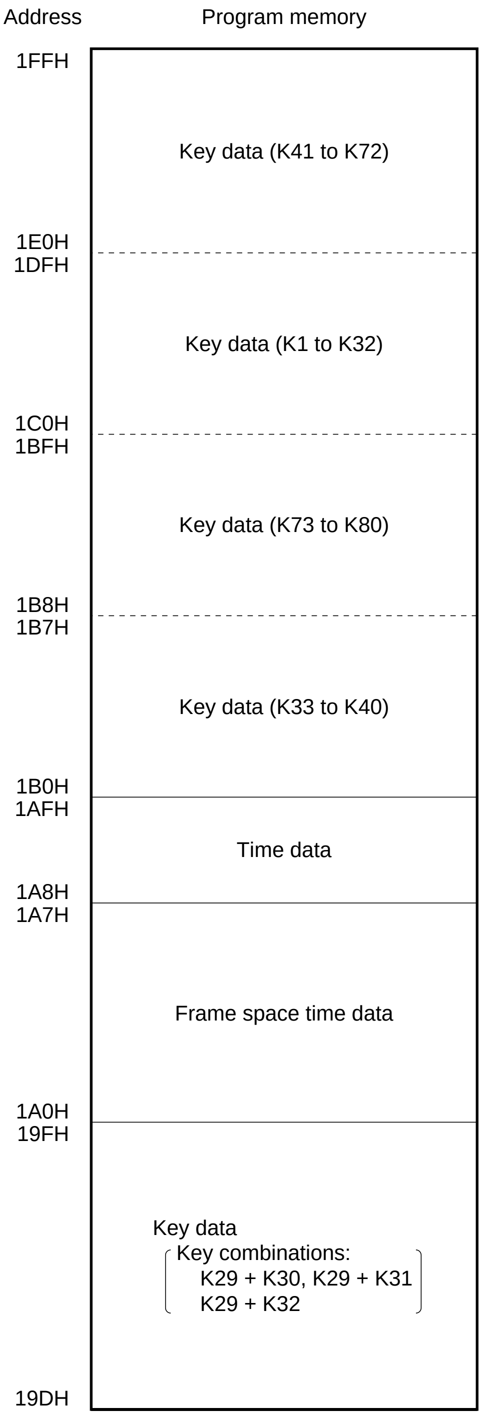


Figure 5-3. Data Table Map



5.3 Data Memory (RAM) Configuration

The data memory (RAM) consists of 32 × 4-bit static RAM, which is used to store processing data. Some instructions enable RAM contents to be manipulated in 8-bit units.

R0 can function as a data pointer for ROM addresses. ROM contents can be accessed once a ROM address is set to this data pointer. This is called a table reference for ROM data. After reset, the value of R0 becomes 00H.

RF can also be used as an address stack register. After reset, the values of R1 to RF are undefined.

Tables 5-1 to 5-3 show RAM maps for the entire program (key input processing and transmission processing), for key input processing alone, and for transmission processing alone. Table 5-4 describes RAM usage.

Table 5-1. RAM Map

	H (1)	L (0)
R0	Work area 1 Data pointer H	Chattering counter Data pointer L
R1	Confirmation key data H	Confirmation key data L
R2	For immediate setting (0FH)	For immediate setting (1H)
R3	K29 ON flag	Continue flag
R4	Key data H Custom code H	Key data L Custom code L
R5	Custom code’ H	Custom code’ L
R6	Key ON flag Data code H	Key scan counter Data code L
R7	For immediate setting (0FH)	Key OFF check counter
R8	Key scan data H For immediate setting (0FH) Data 1	Key scan data L For immediate setting (0H) Data 0
R9	Compare key data H Data 1 transmit counter H	Compare key data L Data 1 transmit counter L
RA	Work area 2	Key return check counter Transmit bit counter
RB	For immediate setting (0CH)	For immediate setting (3H)
RC	For immediate setting (08H)	For immediate setting (3H)
RD	Not used	Not used
RE	Not used	Not used
RF	Address stack register	

Table 5-2. Map of RAM used for Key Input Processing

	H (1)	L (0)
R0	Work area 1 Data pointer H	Chattering counter Data pointer L
R1	Confirmation key data H	Confirmation key data L
R2	For immediate setting (0FH)	For immediate setting (1H)
R3	K29 ON flag	Continue flag
R4	Key data H Custom code H	Key data L Custom code L
R5	Custom code' H	Custom code' L
R6	Key ON flag Data code H	Key scan counter Data code L
R7	▨	▨
R8	Key scan data H For immediate setting (0FH)	Key scan data L For immediate setting (0H)
R9	Compare key data H	Compare key data L
RA	▨	Key return check counter
RB	For immediate setting (0CH)	For immediate setting (3H)
RC	For immediate setting (08H)	For immediate setting (2H)
RD	Not used	Not used
RE	Not used	Not used
RF	Address stack register	

▨ : Used for transmission processing

Table 5-3. RAM Map Used for Transmission Processing

	H (1)	L (0)
R0	Data pointer H Work area 1	Data pointer L
R1	▨	▨
R2	▨	▨
R3	▨	Continue flag
R4	Custom code H	Custom code L
R5	Custom code' H	Custom code' L
R6	Data code H	Data code L
R7	For immediate setting (0FH)	Key OFF check counter
R8	Data 1	Data 0
R9	Data 1 transmit counter H	Data 1 transmit counter L
RA	Work area 2	Transmit bit counter
RC	▨	▨
RC	Not used	Not used
RD	Not used	Not used
RE	Not used	Not used
RF	Address stack register	

▨ : Used for key input processing

Table 5-4. Description of RAM Usage (1/3)

<table>
<tr><th>Name</th><th>RAM</th><th>Description</th></tr>
<tr><td>Data pointer</td><td>R0</td><td>This pointer is used for indicating ROM addresses. It is used when performing a table reference for ROM data.</td></tr>
<tr><td>Chattering counter</td><td>R00</td><td>This counter is used to count the number of times chattering occurs during key input processing. 0CH is set to this counter as the initial value to count three times. 0FH indicates completion of three times.</td></tr>
<tr><td>Work area 1</td><td>R10</td><td>This is a work area that is used to temporarily store data.</td></tr>
<tr><td>Confirmation key data</td><td>R1</td><td>This is used to store confirmation key data after ON chattering elimination processing has been completed.
If other confirmation key data has already been stored, it is compared with the key data when chattering elimination processing is completed (R4). If the two sets of data match, the previous data is retained. If they do not match, the new key data (R4) is stored.</td></tr>
<tr><td>For immediate setting (1H)</td><td>R02</td><td>This is used to clear the high-order three bits.</td></tr>
<tr><td>For immediate setting (0FH)</td><td>R12</td><td>This is used to set 0FH.</td></tr>
<tr><td>Continue flag</td><td>R03</td><td>During transmission processing, this flag is used to determine whether it is the first frame or a second or subsequent frame that is currently being transmitted.<table><tr><th>Value</th><th>Description</th></tr><tr><td>0FH</td><td>Second or subsequent frame is being transmitted</td></tr><tr><td>0H</td><td>First frame is being transmitted</td></tr></table>0FH: Set, 0H: Clear</td></tr>
<tr><td>K29 ON flag</td><td>R13</td><td>This flag is used to determine whether or not the K29 key (combination key) has been pressed.<table><tr><th>Value</th><th>Description</th></tr><tr><td>0FH</td><td>K29 key has been pressed</td></tr><tr><td>0H</td><td>Key other than K29 key has been pressed</td></tr></table>0FH: Set, 0H: Clear</td></tr>
<tr><td>Key data</td><td>R4</td><td>This is used to store the previous key data during ON chattering. This data is compared with the compare key data (R9). If they match, the previous data is retained. If they do not match, the new compare key data (R9) is stored.</td></tr>
<tr><td>Custom code</td><td>R4</td><td>This is used to store the custom code.</td></tr>
<tr><td>Custom code’</td><td>R5</td><td>This is used to store the custom code’.</td></tr>
<tr><td>Data code</td><td>R6</td><td>This is used to store the data code.</td></tr>
<tr><td>Key scan counter</td><td>R06</td><td>This counter is used to count the number of key scans. Since a count of eight is required, 8H (8 times) is set as the counter’s initial value.</td></tr>
</table>

Table 5-4. Description of RAM Usage (2/3)

<table>
<tr><th>Name</th><th>RAM</th><th>Description</th></tr>
<tr><td>Key ON flag</td><td>R16</td><td>This flag is used to determine whether or not a key has been pressed.<table><tr><th>Value</th><th>Description</th></tr><tr><td>0FH</td><td>Key has been pressed</td></tr><tr><td>0H</td><td>Key has been released</td></tr></table>0FH: Set, 0H: Clear</td></tr>
<tr><td>Key OFF check counter</td><td>R07</td><td>This counter is used to count the number of times key OFF status occurs during bit data transmission.
Since a count of ten is required, 5H is set as the initial value. When the value becomes 0FH (meaning ten continuous times of key OFF status), key OFF status is confirmed. If key OFF status is not determined at this time, the counter value is initialized.</td></tr>
<tr><td>For immediate setting (0FH)</td><td>R17</td><td>This is used to set various flags.</td></tr>
<tr><td>Key scan data</td><td>R18</td><td>This is used to store output data for key scanning.</td></tr>
<tr><td>Data 0
Data 1</td><td>R08
R18</td><td>This is used to store the row addresses of time data (table data) for transmitting a bit data value of "0" or "1".
Data 0: Bit data "0"
Data 1: Bit data "1"</td></tr>
<tr><td>For immediate setting (0H)</td><td>R08</td><td>This is used for flag setting.</td></tr>
<tr><td>For immediate setting (0FH)</td><td>R18</td><td>This is used for data inversion.
The setting is "0FH".</td></tr>
<tr><td>Compare key data</td><td>R9</td><td>This is used to store key data during ON chattering.</td></tr>
<tr><td>Data 1 transmit counter</td><td>R9</td><td>This counter is used to count the number of times bit data "1" is transmitted.</td></tr>
<tr><td>Key return check counter</td><td>R0A</td><td>This counter is used to calculate the key data based on the key return data.
Since a count of four is required for K$_I$ input and a count of one is required for input of S$_1$, a initial value of 0CH (for four times) or 0EH (for one time) is set.</td></tr>
<tr><td>Transmit bit counter</td><td>R0A</td><td>This counter is used to set the required number of bits stored in work 2 (R1A) for bit transmissions. The counter counts from one to four times.<table><tr><th>Setting</th><th>Description</th></tr><tr><td>0FH</td><td>1-bit transmission (for transmitting leader code and stop bit)</td></tr><tr><td>0EH</td><td>2-bit transmission (not used)</td></tr><tr><td>0DH</td><td>3-bit transmission (not used)</td></tr><tr><td>0CH</td><td>4-bit transmission (for transmitting custom code, custom code', data code, and $\overline{\text{data code}}$)</td></tr></table></td></tr>
</table>

Table 5-4. Description of RAM Usage (3/3)

Name	RAM	Description
Work area 2	R1A	This is used to store output data for bit transmissions.
For immediate setting (3H)	R0B	This is used for data judgments and for setting 3H.
For immediate setting (0CH)	R1B	This is used for data judgments, for setting 0CH, for setting the high-order two bits, and for clearing the low-order two bits.
For immediate setting (2H)	R0C	This is used for bit 1 judgments.
For immediate setting (08H)	R1C	This is used for clearing the low-order three bits.

5.4 Flag Maps

Table 5-5 describes flag operations during various types of processing.

Table 5-5. Flag Map

(1) Continue flag (R03)

Processing		R03
Initialization processing		Clear (0H)
Key input processing	When the key data used for ON chattering elimination processing differs from the confirmation key data from the previous ON chattering elimination processing.	Clear (0H)
Transmission processing	During frame space transmission of all frames	Judgment
	During frame space transmission of only one frame	Set (0FH)

Judgment: A judgment is made during this processing.

(2) K29 ON flag (R13)

Processing		R13
Initialization processing		Clear (0H)
Key input processing	When key ON flag has been set	Judgment
	When chattering has been completed three times and the confirmation key data is K29	Set (0FH)
	When chattering has been completed three times and the confirmation key data is not K29, nor is there a valid key combination	Clear (0H)
Transmission processing		–

Judgment: A judgment is made during this processing.

—: Not used

(3) Key ON flag (R16)

Processing		R16
Initialization processing		–
Key input processing	During RAM initialization	Clear (0H)
	Start of key return check	Judgment
	After key data calculation	Judgment
	During key return check or when there is key input	Set (0FH)
	After completion of key scanning (8 times)	Judgment
Transmission processing		×

Judgment: A judgment is made during this processing.

—: Not used

×: Used by other application

CHAPTER 6 PROGRAM DESCRIPTION

6.1 Initial Settings

Microcontrollers used in infrared remote control transmitters generally use batteries as their power source. However, the lighting of infrared LEDs requires a large current consumption. The abrupt change in the power supply voltage when infrared LEDs are being lit can cause sudden changes in the contents of RAM, ports, etc., which must be taken into consideration.

To prevent operation faults that may occur as a result of sudden changes in the contents of RAM, ports, etc., the program should be designed to reset data in RAM, ports, etc., to the initial value after each transmission.

6.1.1 Description of processing

(1) Port settings and control register initialization

(a) $K_{I/O}$ port (P0)

This is an 8-bit I/O port that is used for key scan output.
This port's initial setting is FFH. All of the port's bits ($K_{I/O1}$ to $K_{I/O7}$) are set to high-level output.

(b) Control register 0 (P3)

Tables 6-1 and 6-2 list the contents of control register 0. The initial setting is 13H. The initial settings are shown in shaded areas in Tables 6-1 and 6-2.

Table 6-1. Control Register 0 (P3)

Bit		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
Name		–	–	DP (data pointer)		TCTL	CARY	MOD_1	MOD_0
				DP_9	DP_8				
Setting	0	Fixed to “0”	Fixed to “0”	0	0	1/1	ON	See Table 6-2.	
	1			1	1	1/2	OFF		
After reset		0	0	0	0	0	0	1	1

b_0 and b_1 .... Specify the REM output’s carrier frequency and duty factor.

b_2 .... Indicates presence/absence of carrier for frequency specified by b_0 and b_1.
“0” = ON (with carrier), “1” = OFF (no carrier, high level)

b_3 .... Changes the carrier frequency and the timer clock division ratio.
“0” = 1/1 (carrier frequency: values set to b_0 and b_1, timer clock: $f_X/8$)
“1” = 1/2 (carrier frequency: one half of values set to b_0 and b_1, timer clock: $f_X/16$)

Table 6-2. Time Clock and Carrier Frequency Settings

b3	b2	b1	b0	Timer clock	Carrier frequency (duty factor)
0	0	0	0	fx/8	fx (Duty 1/2)
		0	1		fx/8 (Duty 1/2)
		1	0		fx/12 (Duty 1/2)
		1	1		fx/12 (Duty 1/3)
	1	×	×		No carrier (high level)
1	0	0	0	fx/16	fx/2 (Duty 1/2)
		0	1		fx/16 (Duty 1/2)
		1	0		fx/24 (Duty 1/2)
		1	1		fx/24 (Duty 1/3)
	1	×	×		No carrier (high level)

b_4 and b_5 .... Specify the high-order two bits (DP_8 and DP_9) of the ROM data pointer.

Remarks 1. × : Don’t care
2. ▨ : Initial setting (13H)
3. fx : System clock frequency

(c) Control register 1 (P4)

Table 6-3 lists the contents of control register 1.

The initial setting is 32H, which is shown in the shaded areas of the table.

Table 6-3. Control Register 1 (P4)

Bit		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
Name		–	–	K_I pull-down	S_0/S_1 pull-down	–	$S_1/\overline{LED}$ mode	$K_{I/O}$ mode	S_0 mode
Setting	0	Fixed to "0"	Fixed to "0"	OFF	OFF	Fixed to "0"	S_1	IN	OFF
	1			ON	ON		$\overline{LED}$	OUT	IN
When reset		0	0	1	0	0	1	1	0

b_0 .... Specifies the S_0 port's input mode. "0" = OFF mode (high impedance), "1" = IN (input mode).

b_1 .... Specifies the $K_{I/O}$ port's I/O mode. "0" = IN (input mode), "1" = OUT (output mode).

b_2 .... Specifies the $S_1/\overline{LED}$ port's I/O mode. "0" = S_1 (input mode), "1" = $\overline{LED}$ (output mode)

b_4 .... Specifies presence/absence of pull-down resistor when S_0/S_1 port is in input mode. "0" = OFF (no pull-down), "1" = ON (pull-down).

b_5 .... Specifies presence/absence of pull-down resistor when K_I port is in input mode. "0" = OFF (no pull-down), "1" = ON (pull-down).

Remarks 1. All pull-down resistors are automatically switched off during output mode and OFF mode.

2. ▨ : Initial setting (32H)

(2) Initialize RAM

The following RAM contents are cleared to "0".

- Confirmation key data (R1)
- Continue flag (R03)
- K29 ON flag (R13)

(3) Set STOP mode

Table 6-4 lists the cancellation conditions for the HALT instruction.

The initial setting is 8H. When initialized, the (STOP mode) cancellation conditions are set as shown in the shaded areas of the table.

Table 6-4. Cancellation Conditions for HALT Instruction

HALT instruction operand value				Mode setting	Precondition for settings	Cancellation condition
b3	b2	b1	b0			
0	0	0	0	STOP	High-level output from all $K_{I/O}$ pins	High-level input via at least one K_I pin
	0	1	1	STOP	High-level output from all $K_{I/O}$ pins	High-level input via at least one K_I pin
	1	1	0	STOP$^{\text{Note 1}}$	High-level output from $K_{I/O0}$ pin	High-level input via at least one K_I pin
1	Any combination of b_2, b_1, and b_0 above			STOP	[The following conditions in addition to the above conditions]	
					–	High-level input via at least one pin between S_0 and $S_1$$^{\text{Note 2}}$
0/1	1	0	1	HALT	–	When timer’s down counter reaches 0

Notes 1. When HALT#×110B is set, use the $K_{I/O0}$ pin and the K_I pin to configure a key matrix so that an internal reset is executed whenever a runaway (control loss) condition occurs.

2. S_0 and/or S_1 (at least one of these pins that are used to cancel standby mode) must be set to input mode (an internal reset will not be executed if both are set to output mode).

Cautions 1. An internal reset is executed if the HALT instruction is executed using operand values other than those specified above or when the precondition for HALT instruction execution has not been met.

2. If STOP mode is set when the timer’s down counter has not yet reached “0” (i.e., when the timer is operating), all 10 bits of the timer’s down counter and the timer output enable flags are cleared to zero and STOP mode is set.

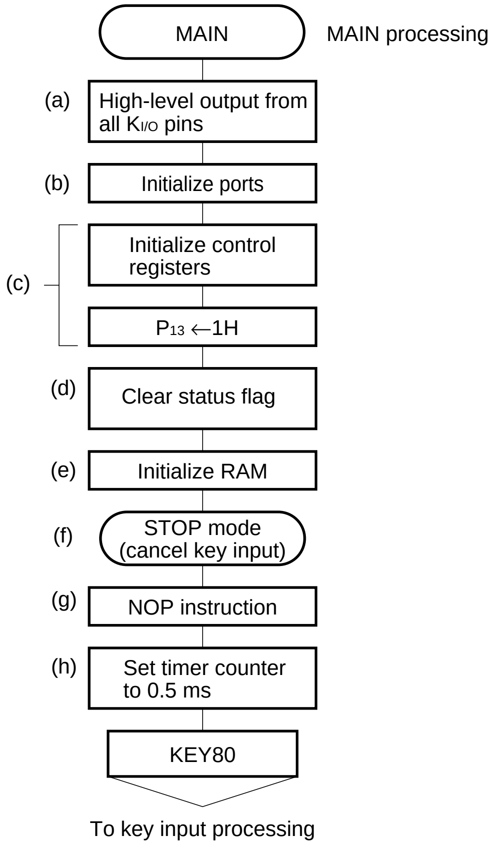
3. Specify a NOP instruction as the first instruction following cancellation of STOP mode.

Remark ▨: Initial setting (8H)

(4) After cancellation of STOP mode, execute a NOP instruction, then initialize the timer.

The timer’s initial setting is 1CH (= 0.5 ms).

6.1.2 Detailed flow chart



```
******** INITIALIZATION PROCESSING  ********
    TIME05M EQU     01CH                    ; 0.5ms(0.510ms)
    DamyTime        EQU     512-1           ; 9.00ms

;########## P U B L I C ##########
    PUBLIC          MAIN
    ;
;##########  S T A R T  ##########
;*********************************************
;   Control Register Initialize
;*********************************************
MAIN:
    OUT     P0,#0FFH                        (a) Sets all key scan outputs (KI/O) to high level
    OUT     P4,#032H                        (b) Initializes ports
                                                S0:  High impedance, S1:  Input mode,
                                                KI/O:  Output mode
    OUT     P3,#013H                        (c) Initializes control registers 0 and 1 (P3 and P4)
                                                With carrier, frequency: fX/12, duty factor: 1/3
                                                TCTL:  1/1, KI:  with pull-down resistor
                                                S0 and S1: with pull-down resistor
                                                Data pointer: DP8 and DP9 = 01H
                                                (Most significant position of table reference
                                                address)
    MOV     T,#DamyTime  }
    STTS    #0101B       }----------------  (d) Clears status flags
;*********************************************
;   RAM Initialize Routine
;*********************************************  (e) Initializes RAM
    MOV     R1,#000H                            Confirmation key data (R1): 00H
    MOV     R3,#000H                            K29 ON flag (R13) and continue flag (R03):  0H

    HALT    #008H                           (f) STOP mode:  Canceled by high-level input via
                                                KI, S0, and S1
    NOP                                     (g) NOP instruction
    MOV     T,#TIME05M                      (h) Initializes timer value
    ;                                           ICH = 0.5 ms
END
```

6.2 Key Input Processing

6.2.1 Description of processing

Key input processing includes chattering elimination processing, key scan processing, custom code generation processing, and key data generation processing.

(1) Chattering elimination processing

When switching to key ON or key OFF status, an unstable condition called chattering (ON chattering or OFF chattering) exists until the key signal is stabilized as the high-level or low-level signal (see Figure 6-1). Since key input during chattering is also unstable, the program must provide a means of eliminating chattering.

Figure 6-1. Chattering of Key Input Signal

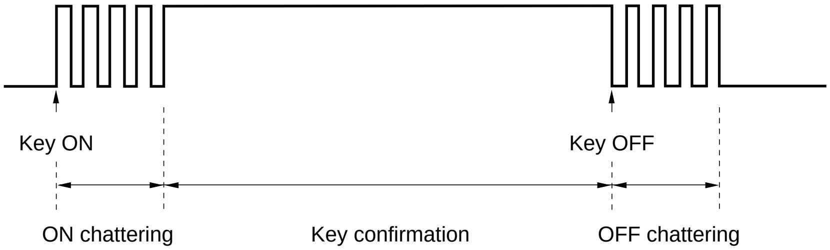


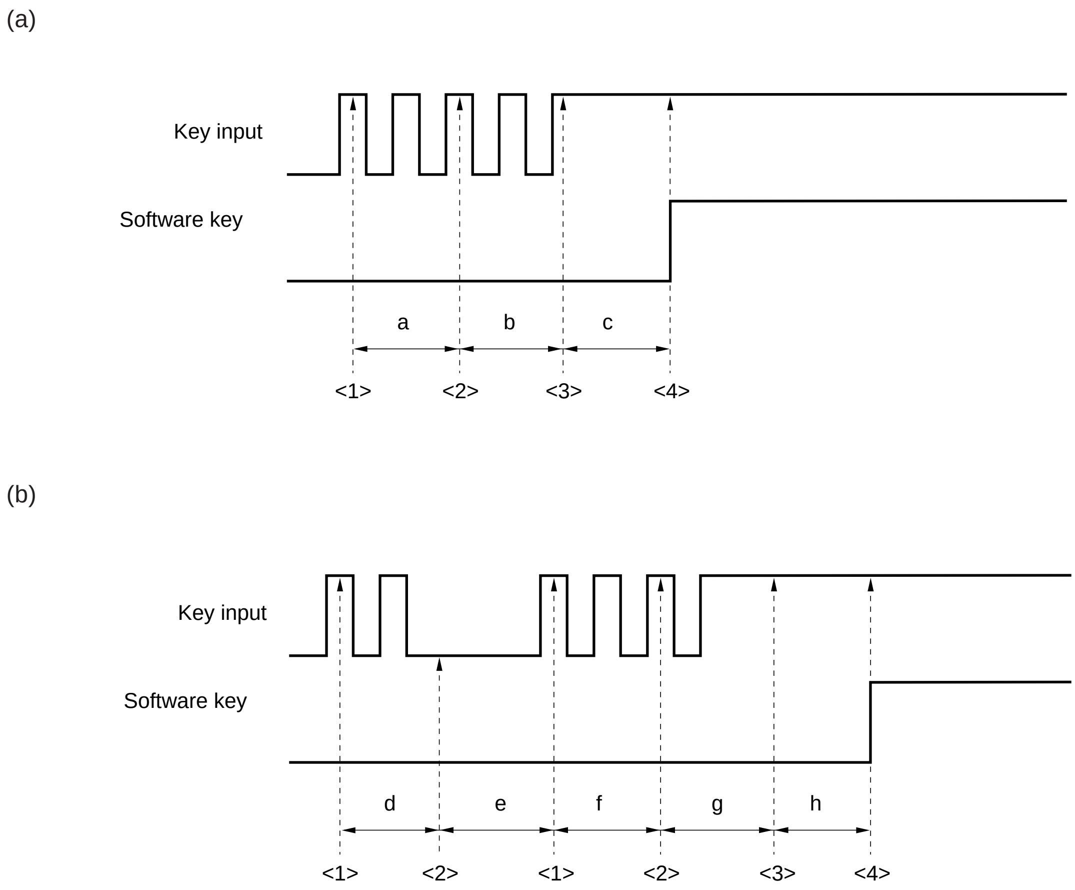
Figure 6-2 illustrates an example of chattering elimination processing (key ON judgment example).

At point <1>, key ON status is detected when STOP mode is canceled, and key input is checked during a set time period (from points <1> to <4>: 9.00 ms × 3).

In part (a) of the figure, key ON status is detected at all check points, so the key signal is judged to be at high level at the software key's point <4>.

The wait period (a + b + c) after key ON status is detected at point <1> and before the software judges the signal's high level status is called the chattering elimination period. In part (a), the chattering elimination period is "a + b + c" since the key input status is always ON when checked. In part (b), the chattering elimination period is "d + e + f + g + h".

Figure 6-2. Key ON Judgment when Chattering Occurs



(2) Key scan processing

Key scan processing is described in sections (a) and (b) below.

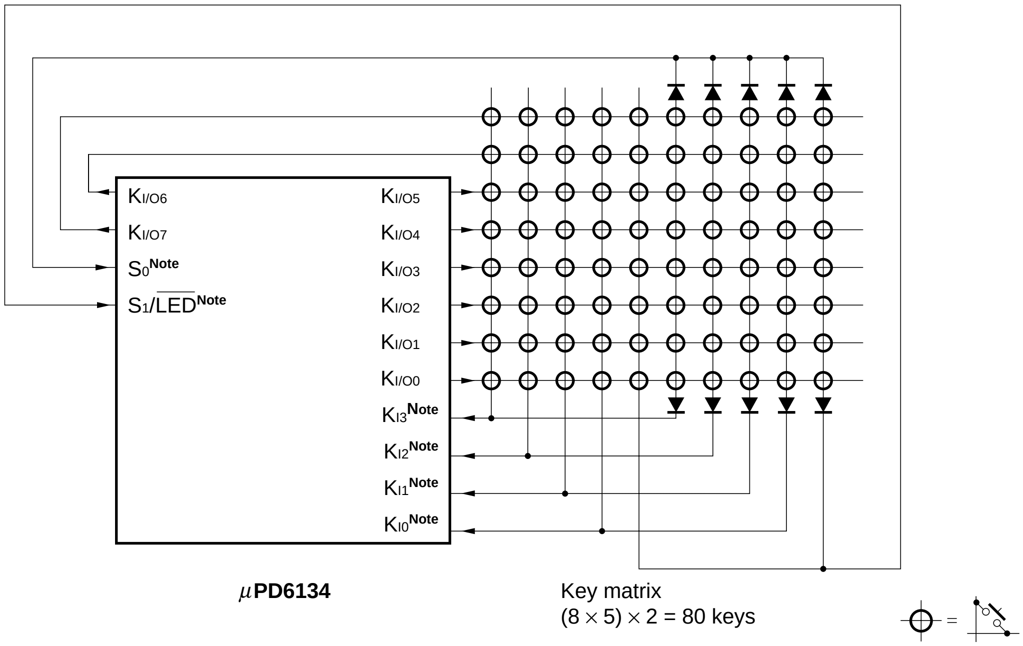
(a) Key matrix

Figure 6-3 shows an example of an 80-key key matrix.

In this example, $K_{I/O0}$ to $K_{I/O7}$ are output ports that output key scan signals. These signals are captured (as key return signals) via input ports comprised of K_{I0} to K_{I3}, S_0, and S_1.

Since each of these key return signal input ports (K_{I0} to K_{I3}, S_0 and S_1) is connected to an internal pull-down resistor, low-level signals are input when no keys are being pressed.

Figure 6-3. 80-key Key Matrix



Note The program is set for “on-chip pull-down resistor”.

(b) Key scan

To judge which of the keys in the 80-key key matrix is being pressed, the STTS instruction is used to check for high-level signal input via K_{I0} to K_{I3}, S_0, and S_1.

Next, the key scan signal’s output ports ($K_{I/O0}$ to $K_{I/O7}$) are set to high level one at a time starting from $K_{I/O0}$ to determine which output among $K_{I/O0}$ to $K_{I/O7}$ corresponds to the detected input.

During the key scan processing part of the program, the key scan counter (key source position) and key return check counter (key return position) are used to detect the key data (key position). Figure 6-5 shows the correspondence between key data and the values of these two counters.

For description of the key data's bit configuration in data memory, see **(4) (a) Bit configuration of key data**. When performing a key scan, factors such as stray capacitance in the keyboard and key source delay due to line impedance must be taken into consideration. Therefore, this program waits for about 100 μs (six steps when at 455 kHz) following high-level output before capturing the key input.

Figure 6-4. Key Matrix

$K_{I/O7}$	K29	K30	K31	K32	K40	K69	K70	K71	K72	K80
$K_{I/O6}$	K25	K26	K27	K28	K39	K65	K66	K67	K68	K79
$K_{I/O5}$	K21	K22	K23	K24	K38	K61	K62	K63	K64	K78
$K_{I/O4}$	K17	K18	K19	K20	K37	K57	K58	K59	K60	K77
$K_{I/O3}$	K13	K14	K15	K16	K36	K53	K54	K55	K56	K76
$K_{I/O2}$	K9	K10	K11	K12	K35	K49	K50	K51	K52	K75
$K_{I/O1}$	K5	K6	K7	K8	K34	K45	K46	K47	K48	K74
$K_{I/O0}$	K1	K2	K3	K4	K33	K41	K42	K43	K44	K73
	K_{I3}	K_{I2}	K_{I1}	K_{I0}	S_1	K_{I3}	K_{I2}	K_{I1}	K_{I0}	S_1
	S_0 = 0					S_0 = 1 (expansion)				

Figure 6-5. Key Data (Key Position)

Key scan counter (R06) ↓										
F	K29	K30	K31	K32	K40	K69	K70	K71	K72	K80
E	K25	K26	K27	K28	K39	K65	K66	K67	K68	K79
D	K21	K22	K23	K24	K38	K61	K62	K63	K64	K78
C	K17	K18	K19	K20	K37	K57	K58	K59	K60	K77
B	K13	K14	K15	K16	K36	K53	K54	K55	K56	K76
A	K9	K10	K11	K12	K35	K49	K50	K51	K52	K75
9	K5	K6	K7	K8	K34	K45	K46	K47	K48	K74
8	K1	K2	K3	K4	K33	K41	K42	K43	K44	K73
Key return → check counter (R0A)	C	D	E	F	F	C	D	E	F	F

(3) Custom code generation processing

This program is set to output 0AH as the custom code and F5H as the custom code'.

Specifically, 50H is set to data memory R4 as the custom code and AFH is set to R5 as the custom code'.

Caution In the NEC format, the LSB is transmitted first, so values are set in opposite order to the bit string.

(4) Key data generation processing

Key data processing is described in (a) and (b) below.

(a) Bit configuration of key data

Key data consists of eight bits. Each bit indicates a key source or key return status.

To accommodate the eight types from $K_{I/O0}$ to $K_{I/O7}$, the key scan counter uses the LSB of key data H and the high-order two bits of key data L when input is via K_{I0} to K_{I3}, and uses the low-order three bits of key data L when input is via S_1.

The key return check counter accommodates the four types from K_{I0} to K_{I3} by using the low-order two bits of the key data. For input via S_1, there is only one type, so the key return check counter is not used.

The data from key expansion pin S_0 uses bit 1 of the key data H if input is via K_{I0} to K_{I3} and uses the MSB of key data L if input is via S_1.

When the first key input is detected, 0BH is set to key data H if the input is from other than K_I (such as input via S_1). This key data H is also used to determine the format used to generate the key data, as shown in Figures 6-6 and 6-7.

As is described in "(b) Data code" below, this key data is also used as an address for table reference. Therefore, the key data is configured as shown in Figures 6-6 and 6-7.

Figure 6-6. Bit Configuration of Key Data when Input is via K_{I0} to K_{I3}

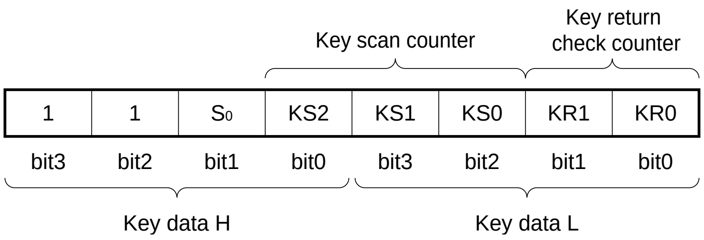


Figure 6-7. Bit Configuration of Key Data when Input is via S_1

					Key scan counter		
1	0	1	1	S_0	KS2	KS1	KS0
bit3	bit2	bit1	bit0	bit3	bit2	bit1	bit0
Key data H				Key data L			

(b) Data code

The data code can be calculated by using the key data obtained via key input processing (as shown in Figures 6-6 and 6-7) as an address for table reference.

The table reference method uses the contents of ROM as a transmit code by setting 1H to the high-order four bits of control register 0 (P3) and by setting the key data to the data pointer (see Figure 6-8).

Figure 6-8. Configuration of Data Pointer

Control register 0 (P3)

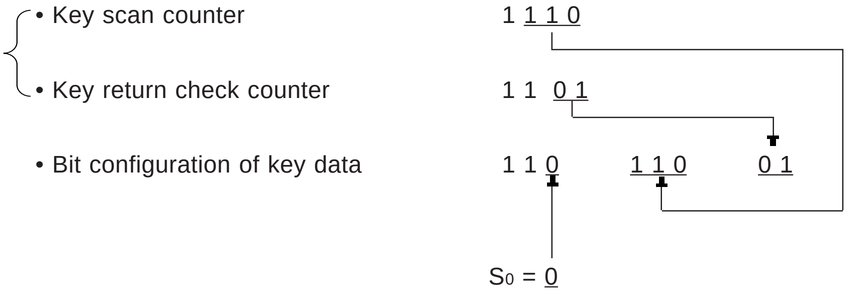
b7	b6	b5	b4
Fixed to "0"	Fixed to "0"	DP_9	DP_8

Data pointer (DP)

R_{10}				R_{00}			
DP_7	DP_6	DP_5	DP_4	DP_3	DP_2	DP_1	DP_0

Example When key position is K26 (key input K_{I2})

1. Key scan counter (R06) becomes EH key return check counter (R0A) becomes 0H (see Figure 6-5).
2. The low-order three bits of the key scan counter and the low-order two bits of the key return check counter are used to configure the key data, as was shown in Figure 6-6.



- Key scan counter 1 1 1 0
- Key return check counter 1 1 0 1
- Bit configuration of key data 1 1 0 1 1 0 0 1

S_0 = 0

3. Set key data to data pointer.
   - Configuration of data pointer 0 0 0 1 1 1 0 1 1 0 0 1 → 1 D 9 H

The following table reference addresses are generated when a key is generated using one of the bit configurations shown in Figures 6-6 and 6-7. As mentioned above, 1H is set to the high-order four bits of control register 0 (P3).

- Key input: K_{I0} to K_{I3}, key position: K1 to K32 (when S_0 = 0)
  ... reference addresses = 1C0H to 1DFH

- Key input: K_{I0} to K_{I3}, key position: K41 to K72 (when S_0 = 1)
  ... reference addresses = 1E0H to 1FFH

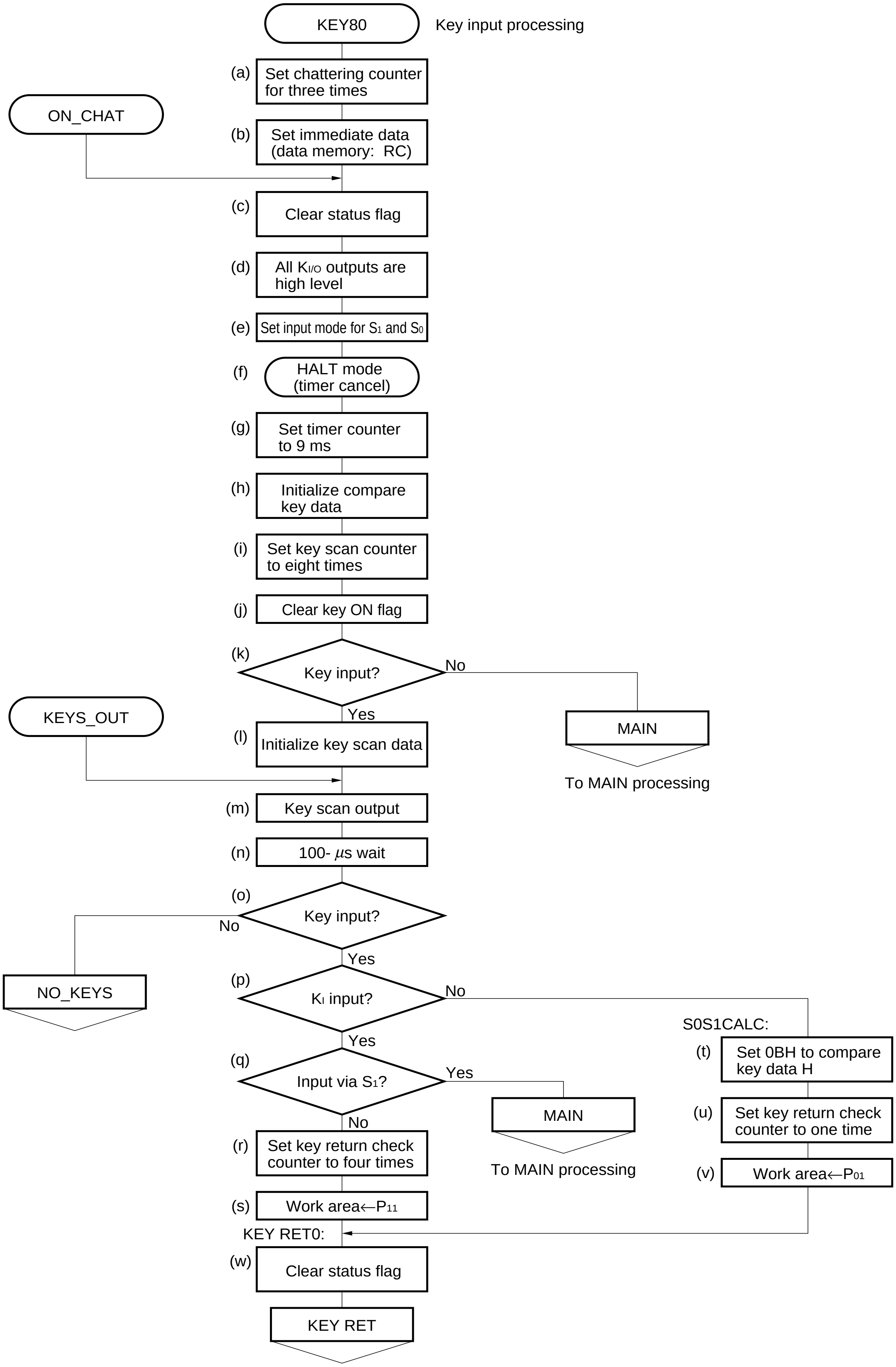
- Key input: S_1, key position: K33 to K40 (when S_0 = 0)
  ... reference addresses = 1B0H to 1B7H

- Key input: S_1, key position: K73 to K80 (when S_0 = 1)
  ... reference addresses = 1B8H to 1BFH

- Key combination (K29 + K30, K29 + K31, K29 + K32)
  When a key combination has been confirmed, 9H is set to key data H.

  K29 + K30 ... reference address = 19DH
  K29 + K31 ... reference address = 19EH
  K29 + K32 ... reference address = 19FH

6.2.2 Detailed flow chart


KEY80
Key input processing
(a) Set chattering counter for three times
ON_CHAT
(b) Set immediate data (data memory: RC)
(c) Clear status flag
(d) All KI/O outputs are high level
(e) Set input mode for S1 and S0
(f) HALT mode (timer cancel)
(g) Set timer counter to 9 ms
(h) Initialize compare key data
(i) Set key scan counter to eight times
(j) Clear key ON flag
(k) Key input?
No
MAIN
To MAIN processing
Yes
KEYS_OUT
(l) Initialize key scan data
(m) Key scan output
(n) 100- µs wait
(o) Key input?
No
NO_KEYS
Yes
(p) KI input?
No
S0S1CALC:
(t) Set 0BH to compare key data H
Yes
(q) Input via S1?
Yes
MAIN
To MAIN processing
(u) Set key return check counter to one time
No
(r) Set key return check counter to four times
(v) Work area←P01
(s) Work area←P11
KEY RET0:
(w) Clear status flag
KEY RET

```
******** KEY INPUT PROCESSING ********
        TIME9M  EQU     1FFH                    ; 9.00ms(9.002ms)
        CUSTM1  EQU     050H                    ; Custom Code = 0AH
        CUSTM2  EQU     0AFH                    ; Custom Code' = F5H

;########## P U B L I C ##########
        PUBLIC          KEY80
        ;

;########## E X T E R N ##########
        EXTRN           MAIN                    ; MAIN Routine
        ;

;##########  S T A R T  ##########
KEY80:
        MOV     R0,#00CH                        (a)  Sets (0CH to) chattering counter (R00)
                                                     (Three times count ends at 0FH.)
        MOV     RC,#082H                        (b)  Use RC to set immediate data (R1C = 8H, R0C
                                                      = 2H)
;***********************************************
;               ON Chattering
;***********************************************
ON_CHAT:
        STTS    #0101B                          (c)  Clears status flag
        OUT     P0,#0FFH                        (d)  Sets all KI/O outputs to high level
        OUT     P4,#033H                        (e)  Sets input mode for S1 and S0
        HALT    #005H                           (f)  HALT mode:  timer is canceled when count
                                                     reaches 00H
        MOV     T,#TIME9M                       (g)  Sets timer counter.  (9 ms = 1FFH)
        MOV     R9,#000H                        (h)  Initializes (sets to "00H") compare key data (R9)
        MOV     R6,#008H                        (i) (j) Sets 8H to key scan counter (R06) and sets 0H
                                                      to clear key ON flag (R16)
        STTS    #1110B                          (k)  Determines when there is key input (via KI, S0, or
                                                     S1)
        JNF     MAIN                                 If no key input, processing branches to MAIN.
        MOV     R8,#001H                        (l)  Initializes (sets 01H to) key scan data (R8)

;***********************************************
;                 Key Scan
;***********************************************
KEYS_OUT:
        MOV     A,R08
        OUT     P00,A     -----------------     (m)  Outputs key scan data (P0)
        MOV     A,R18
        OUT     P10,A
;+++++++++++++++++++++++++++++
;++ 100-μs wait before key input ++
;+++++++++++++++++++++++++++++
        MOV     R2,#0F1H                        Remark  R2 is used to set immediate data
                                                        (R12: 0FH, R02: 1H).
        MOV     RB,#0C3H                        Remark  R8 is used to set immediate data
                                                        (R1B: 0CH, R0B: 3H).
        NOP                -----------------    (n) 100-μs wait before performing key input check
        NOP
        NOP
        NOP
```

```
;***********************************************
;               Key Return Check
;***********************************************
        STTS    #1011B
        JNF     NO_KEYS
```

(o) Determines whether or not there is key input.
If no key input, processing branches to NO_KEYS.

```
        STTS    #0011B
        JNF     S0S1CALC
```

(p) Determines whether input is via K_I or S_0 and S_1.
If via S_0 and S_1, processing branches to S0S1CALC.

```
;*******************************
;       KI Data Calculate
;*******************************
        IN      A,P01
        RL      A

        JC      MAIN
```

(q) When there is input via K_I, it determines whether there is also input via S_1.
If there is also input via S_1, a multiple combination key error is detected and it goes to MAIN processing.

```
        MOV     A,R1B
        MOV     R0A,A
```

(r) Initializes the key return check counter (R_{0A}).
0CH = 4 times (In this case, R_{1B} = 0CH).

```
        IN      A,P11
        JMP     KEY_RET0
```

(s) The pin status (P_{11}) of K_I is transferred to the accumulator.

```
;*******************************
;       S0,S1 Data Calculate
;*******************************
S0S1CALC:
        MOV     R9,#0B0H
```

(t) A judgment value (0BH) for K33 to K40 and K73 to K80 (for input via S_0 and S_1) is set to the compare key data H (R_{19}).

```
        MOV     A,R12
        MOV     R0A,A
```

(u) Initializes the key return check counter (R_{0A}).
0FH = 1 time (in this case, R_{12} = 0FH).

```
        IN      A,P01
```

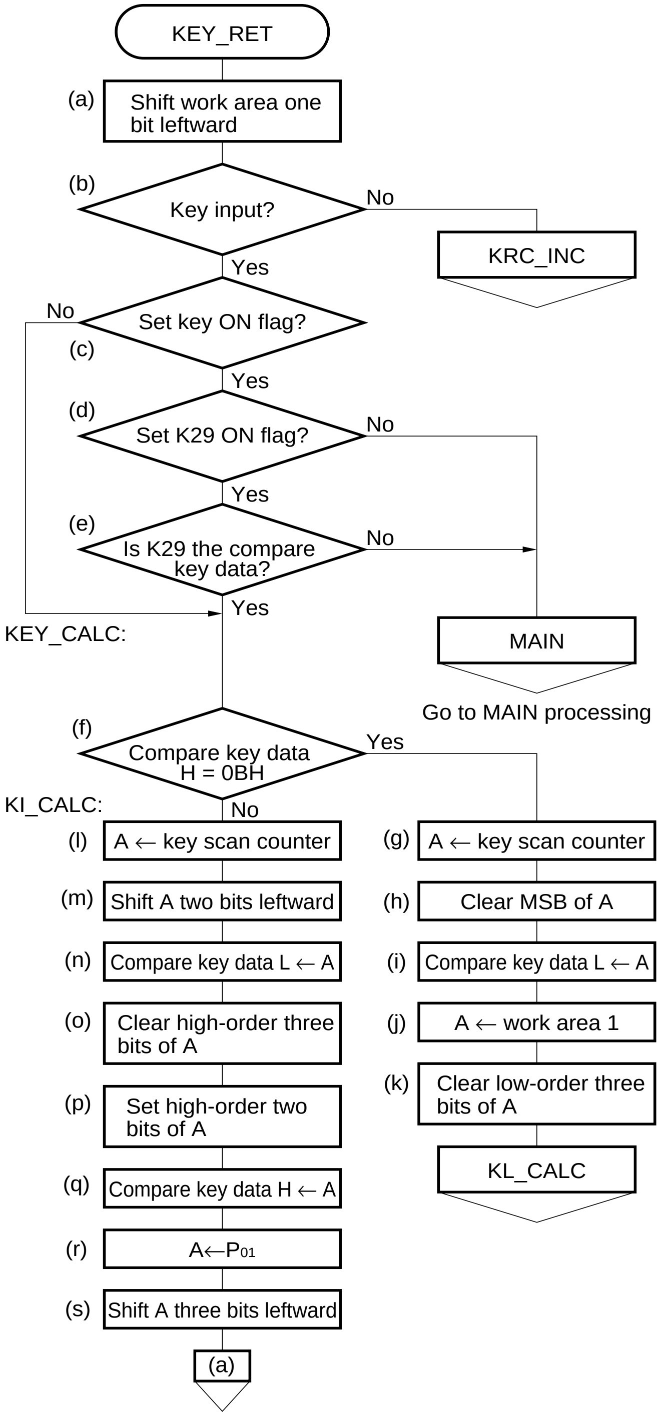
(v) The pin status (P_{01}) of S_0 or S_1 is transferred to the accumulator.

```
KEY_RET0:
        MOV     R10,A
```

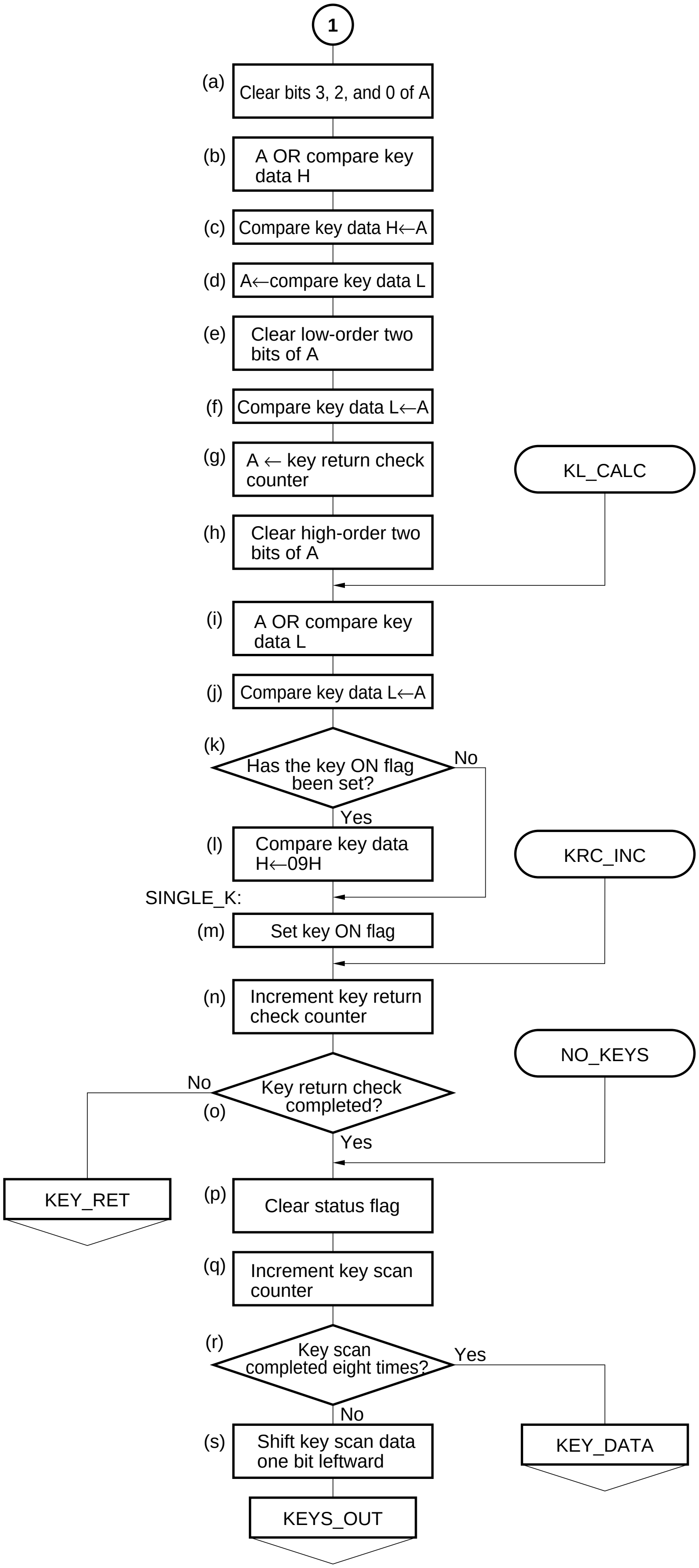
Remark The accumulator's value (P_{00} or P_{01}) is stored in the work area.

```
        STTS    #0101B
```

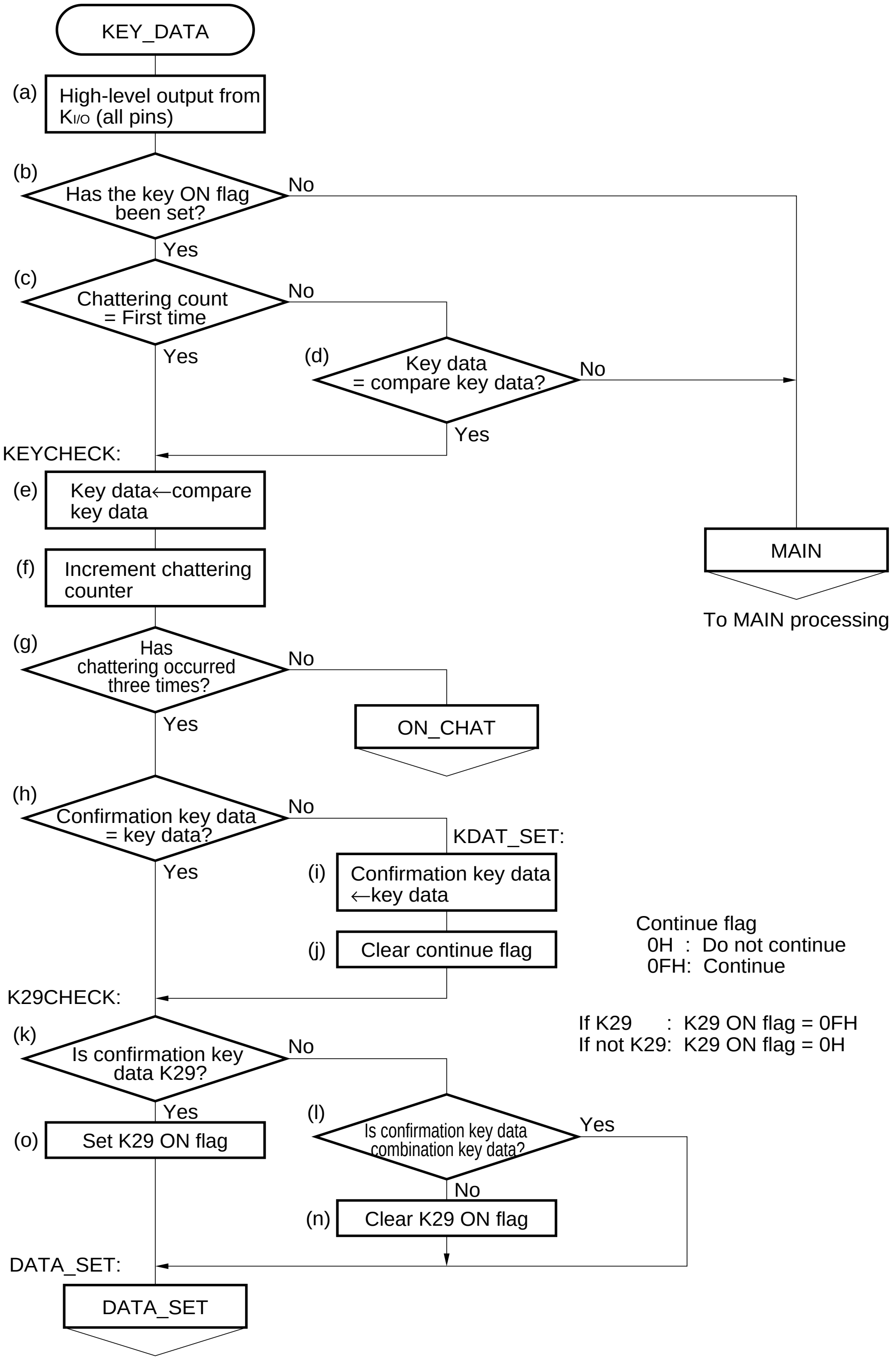
(w) Clears status flag.


KEY_RET
(a)
Shift work area one bit leftward
(b)
Key input?
No
KRC_INC
Yes
No
Set key ON flag?
(c)
Yes
(d)
Set K29 ON flag?
No
Yes
(e)
Is K29 the compare key data?
No
Yes
KEY_CALC:
MAIN
Go to MAIN processing
(f)
Compare key data H = 0BH
Yes
KI_CALC:
No
(l)
A ← key scan counter
(g)
A ← key scan counter
(m)
Shift A two bits leftward
(h)
Clear MSB of A
(n)
Compare key data L ← A
(i)
Compare key data L ← A
(o)
Clear high-order three bits of A
(j)
A ← work area 1
(k)
Clear low-order three bits of A
(p)
Set high-order two bits of A
KL_CALC
(q)
Compare key data H ← A
(r)
A←P01
(s)
Shift A three bits leftward
(a)

```
KEY_RET:
        MOV     A,R10
        RLZ     A        ------------ (a)  Shifts work area 1 (R10) one bit leftward
        MOV     R10,A
        JNC     KRC_INC               (b)  Determines whether or not key input exists (CY = 1).
;** Key input exists **                    If there is no key input, processing branches to KRC_INC.
        MOV     A,R16
        SCAF             ------------ (c)  Determines whether or not key has been pressed (combination key).
        JNC     KEY_CALC                   If no combination key (only one key) has been pressed, processing
                                           branches to KEY_CALC.
        MOV     A,R13
        SCAF             ------------ (d)  Determines whether or not the K29 ON flag has been set.
        JNC     MAIN                       If it has been cleared, processing branches to MAIN.
        MOV     A,R0C                      Remark  In this case, R0C = 2H.
        XRL     A,R19
        SCAF
        JNC     MAIN     ------------ (e)  Determines whether or not the compare key data (R9) matches K29's
                                           key data.
        MOV     A,R09                      If they do not match, processing branches to MAIN.
        XRL     A,R0B
        SCAF
        JNC     MAIN
;*******************************
;       Key Data Calculate
;*******************************
KEY_CALC:
        MOV     A,#04H
        XRL     A,R19                 (f)  Determines whether or not the pressed key is between K33 and K40
        SCAF                               or K73 and K80 (input via S0 or S1).
        JNC     KI_CALC                    If a key between K1 and K32 and or K41 and K72 has been pressed,
;** S1 data calculation **                 processing branches to KI_CALC.      Accumulator values
        MOV     A,R06                 (g)  Transfers key scan counter (R06)     | KS3 | KS2 | KS1 | KS0 |
                                           to accumulator.
        ANL     A,#0111B              (h)  Clears the accumulator's MSB.        |  0  | KS2 | KS1 | KS0 |
        MOV     R09,A                 (i)  Stores accumulator values in                    ACC → R09
                                           compare key data L (R09).
        MOV     A,R10                 (j)  Transfers work area 1 (R10) to       | S0  |  1  |  1  | S1  |
                                           accumulator.
        ANL     A,R1C                 (k)  Clears the accumulator's low-order   | S0  |  0  |  0  |  0  |
                                           three bits.
                                           (In this case, R1C = 8H).
        JMP     KL_CALC                    To KL_CALC.
        ;
;** KI data calculation **
KI_CALC:
        MOV     A,R06                 (l)  Transfers key scan counter (R06) to  | KS3 | KS2 | KS1 | KS0 |
                                           accumulator.
        RL      A
        RL      A        ------------ (m)  Shifts accumulator two bits leftward | KS1 | KS0 | KS3 | KS2 |
        MOV     R09,A                 (n)  Stores accumulator values in                    ACC → R09
                                           compare key data L (R09).
        ANL     A,R02                 (o)  Clears the high-order three bits of the |  1  |  0  |  0  | KS2 |
                                           accumulator.  (In this case, R02 = 1H)
        ORL     A,R1B                 (p)  Sets the high-order two bits of      |  1  |  1  |  0  | KS2 |
                                           accumulator (In this case, R1B = 0CH.)          ACC → R09
        MOV     R19,A                 (q)  Stores accumulator values in
                                           compare key data H (R19).
        IN      A,P01                 (r)  Transfers S1 and S0 data to          | S1  | S0  |  1  |  1  |
                                           accumulator.
        RL      A
        RL      A        ------------ (s)  Shifts accumulator three bits leftward. |  1  | S1  | S0  |  1  |
        RL      A
```


1
(a) Clear bits 3, 2, and 0 of A
(b) A OR compare key data H
(c) Compare key data H←A
(d) A←compare key data L
(e) Clear low-order two bits of A
(f) Compare key data L←A
(g) A ← key return check counter
KL_CALC
(h) Clear high-order two bits of A
(i) A OR compare key data L
(j) Compare key data L←A
(k) Has the key ON flag been set?
No
Yes
(l) Compare key data H←09H
KRC_INC
SINGLE_K:
(m) Set key ON flag
(n) Increment key return check counter
NO_KEYS
No
(o) Key return check completed?
Yes
KEY_RET
(p) Clear status flag
(q) Increment key scan counter
(r) Key scan completed eight times?
Yes
No
(s) Shift key scan data one bit leftward
KEY_DATA
KEYS_OUT

```
                                                                              Accumulator values
        ANL     A,R0C               (a) Clears bits 3, 2, and 0 of accumulator | 0   | 0   | S0  | 0   |
                                        (In this case, R0C = 2H)
        ORL     A,R19               (b) Performs OR processing of              | 1   | 1   | S0  | KS2 |
                                        accumulator values and compare key
                                        data H (R19).
        MOV     R19,A               (c) Stores accumulator values in                   ACC → R19
                                        compare key data H (R19).

        MOV     A,R09               (d) Transfers compare key data L (R09)     | KS1 | KS0 | KS3 | KS2 |
                                        to accumulator.
        ANL     A,R1B               (e) Clears the low-order two bits of the   | KS1 | KS0 | 0   | 0   |
                                        accumulator.  (In this case, R1B =1100B)

        MOV     R09,A               (f) Stores accumulator values in                   ACC → R09
                                        compare key data L (R09).
        MOV     A,R0A               (g) Transfers key return check counter     | KR3 | KR2 | KR1 | KR0 |
                                        (R0A) to accumulator.
        ANL     A,R08               (h) Clears the accumulator's high-order    | 0   | 0   | KS1 | KS0 |
                                        two bits.  (In this case, R0B =0011B).
KL_CALC:                                                                        KI input
        CRL     A,R09               (i) Performs OR processing of              | KS1 | KS0 | KR1 | KR0 |
                                        accumulator values and compare         S1 input
                                        key data L (R09).                      | S0  | KS2 | KS1 | KS0 |

        MOV     R09,A               (j) Stores accumulator values to                   ACC → R09
                                        compare key data L (R09).
;** Key combination check **
        MOV     A,R16     }
        SCAF              }-------- (k) Determines whether or not the key ON flag (R16) has been set.
        JNC     SINGLE_K  }             If it has been cleared, processing branches to SINGLE_K.
        MOV     A,#09H    }
        MOV     R19,A     }-------- (l) Sets 9H to compare key data H (R19).
SINGLE_K:
        MOV     A,R12     }
                          }-------- (m) Sets the key ON flag (R16) (In this case, R12 = 0FH).
        MOV     R16,A     }
KRC_INC:
        MOV     A,R0A     }
        INC     A         }-------- (n) Increments the key return check counter (R0A).
        MOV     R0A,A     }
        JNC     KEY_RET             (o) Determines whether or not the key return check has been completed
                                        four times (for KI input) or one time (for S1 input).
                                        If it has not been completed the required number of times, processing
                                        branches to KEY_RET.
NO_KEY:
        STTS    #0101B              (p) Clears status flags.
        MOV     A,R06     }
        INC     A         }-------- (q) Increments key scan counter (R06).
        MOV     R06,A     }
        JC      KEY_DATA            (r) Determines whether or not the key scan has been completed eight
                                        times.
        MOV     A,R18     }             If it has been completed, processing branches to KEY_DATA.
        RL      A         }
        MOV     R18,A     }
        MOV     A,R08     }
        RL      A         }-------- (s) Shifts the key scan data (R8) one bit leftward.
        MOV     R08,A     }
        JNC     KEYS_OUT  }
        MOV     R8,#010H  }
        JMP     KEYS_OUT  }
```


KEY_DATA
(a)
High-level output from
KI/O (all pins)
(b)
Has the key ON flag been set?
No
Yes
(c)
Chattering count = First time
No
Yes
(d)
Key data = compare key data?
No
Yes
KEYCHECK:
(e)
Key data←compare key data
(f)
Increment chattering counter
MAIN
To MAIN processing
(g)
Has chattering occurred three times?
No
Yes
ON_CHAT
(h)
Confirmation key data = key data?
No
KDAT_SET:
Yes
(i)
Confirmation key data ←key data
(j)
Clear continue flag
Continue flag
0H : Do not continue
0FH: Continue
K29CHECK:
If K29 : K29 ON flag = 0FH
If not K29: K29 ON flag = 0H
(k)
Is confirmation key data K29?
No
Yes
(o)
Set K29 ON flag
(l)
Is confirmation key data combination key data?
Yes
No
(n)
Clear K29 ON flag
DATA_SET:
DATA_SET

```
;**********************************************
; Transmit Key Data = ON Chattering Key Data??
;**********************************************
KEY_DATA:
        OUT     P0,#0FFH
```

(a) Sets all key scan output (via $K_{I/O}$) as high-level output.

```
        MOV     R8,#0F0H
```

Remark R8 is used for setting immediate data (R_{16}: 0FH, R_{08}: 0H).

```
        MOV     A,R16
        SCAF
        JNC     MAIN
```

(b) Determines whether or not the key ON flag (R_{16}) has been set.
If it has been cleared, processing branches to MAIN.

```
        MOV     A,R00
        XRL     A,R0B
        SCAF
        JC      KEYCHECK
```

(c) Determines whether or not chattering processing is occurring for the first time.
(In this case, R_{0B} = #0011B.)
If it is the first time, processing branches to KEYCHECK.

Remark In this case, R_{16} = 0FH.

```
        MOV     A,R14
        XRL     A,R18
        XRL     A,R19
        SCAF
        JNC     MAIN
        XRL     A,R04
        XRL     A,R09
        SCAF
        JNC     MAIN
```

(d) Determines whether or not the key data (R4) matches the compare key data (R9).
(In this case, accumulator value = 0FH.)
If they do not match, processing branches to MAIN.

```
KEYCHECK:
        MOV     A,R19
        MOV     R14,A
        MOV     A,R09
        MOV     R04,A
```

(e) Stores compare key data (R9) in key data (R4).

```
        MOV     A,R00
        INC     A
        MOV     R00,A
```

(f) Increments the chattering counter (R_{00}).

```
        SCAF
        JNC     ON_CHAT
```

(g) Determines whether or not chattering processing has been completed three times.
If it has not been completed three times, processing branches to ON_CHAT.

```
;** Key High Check **
        XRL     A,R11
        XRL     A,R14
        SCAF
        JNC     KDAT_SET
;** Key Low Check **
        XRL     A,R01
        XRL     A,R04
        SCAF
        JC      K29CHECK
```

Remark In this case, accumulator value = 0FH.

(h) Determines whether or not the confirmation key data (R1) matches the key data (R4).
(In this case, accumulator value = 0FH.)
If they do not match, processing branches to KDAT_SET.
If they match, processing branches to K29CHECK.

```
KSAT_SET:
        MOV     A,R14
        MOV     R11,A
        MOV     A,R04
        MOV     R01,A
```

(i) Stores key data (R4) in confirmation key data (R1).

```
        MOV     A,R08
        MOV     R03,A
```

(j) Clears continue flag (R_{03}). (In this case, R_{08} = 0H).

```
K29CHECK:
        MOV     A,R0C
        XRL     A,R11
        SCAF
        JNC     DBL_K_CHK

        MOV     A,R01
        XRL     A,R0B
        SCAF
        JC      K29_FLG
```

Remark In this case, R_{0C} = 2H.

(k) Determines whether or not K29 has been pressed.
(In this case, R_{0B} = 3H).
If a key other than K29 has been pressed, processing branches to K29_FLG.

```
SBL_K_CHK:
        MOV     A,#06H
        XRL     A,R11
        SCAF
        JC      DATA_SET
```

(l) Determines whether or not the confirmation key data is combination key data.
If it is combination key data, processing branches to DATA_SET.

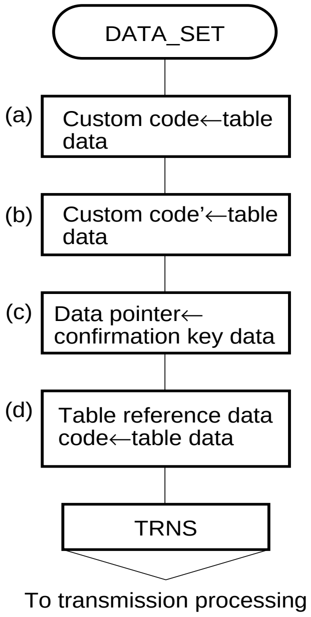
```
        MOV     A,R08
```

(m) Sets 0H to the accumulator.
(In this case, R_{08} = 0H).

```
K29_FLG:
        MOV     R13,A
```

(n) Clears the K29 ON flag (in this case, the accumulator value = 0H).

(o) Sets the K29 ON flag (in this case, the accumulator value = 0FH).


DATA_SET
(a)
Custom code←table data
(b)
Custom code'←table data
(c)
Data pointer← confirmation key data
(d)
Table reference data code←table data
TRNS
To transmission processing

```
;***********************************************
;               Transmit Data Set
;***********************************************
DATA_SET:
    ;**  Custom Code  **
       MOV      R4,#CUSTM1                          (a) Stores table data in custom code (R4).
    ;**  Custom Code' **
       MOV      R5,#CUSTM2                          (b) Stores table data in custom code' (R5).

    ;**  Custom Code  **
       MOV      A,R11       }
       MOV      R10,A       }-----------------      (c) Sets confirmation key data (R1) in data pointer
       MOV      A,R01       }                           (R0).
       MOV      R00,A       }
       MOV      R6,@R0                              (d) Stores table data in data code (R6).

SET_END:
       END
```

6.3 Transmission Processing

Transmission processing transmits custom code, custom code', data code, and $\overline{\text{data code}}$ stored in the data memory, using the NEC-R format that was described in **CHAPTER 2 TRANSMISSION WAVEFORM**.

The transmission method uses a data table to store transmission times and sets the timer counter via a table reference operation. Afterward, it enters HALT mode (cancellation condition: TIMER) to enable transmission.

A method such as that shown in Figure 6-10 is used to ensure correct and simple time management. Immediately after cancellation of HALT mode, a similar method is used to set the transmission time data for the next transmission to the timer counter so that the transmission operation can be performed during a set time between HALT modes.

When carrier output is set to ON (hereafter, this status is called “H”), the timer counter’s MSB (output control bit) is set. When carrier output is set to OFF (hereafter, this status is called “L”), the timer counter’s MSB (output control bit) is cleared.

Figure 6-9 shows the timer counter’s configuration.

Figure 6-9. Configuration of Timer Counter

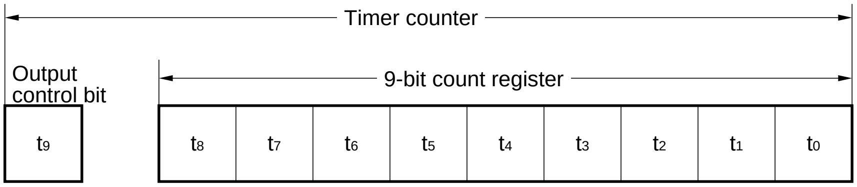
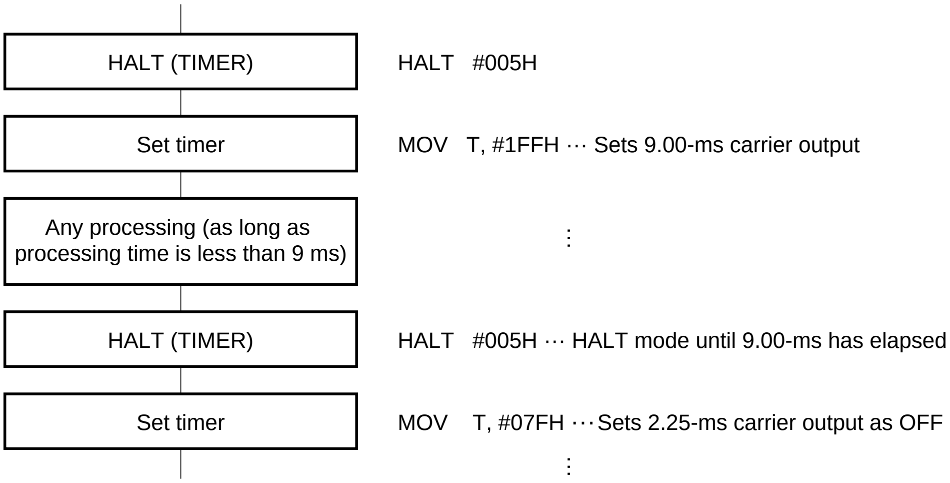


Figure 6-10. Time Management of Timer Counter


HALT (TIMER)
Set timer
Any processing (as long as processing time is less than 9 ms)
HALT (TIMER)
Set timer

```
HALT   #005H
MOV   T, #1FFH ··· Sets 9.00-ms carrier output
                 ⋮
HALT   #005H ··· HALT mode until 9.00-ms has elapsed
MOV    T, #07FH ···Sets 2.25-ms carrier output as OFF
                 ⋮
```

The timer’s operation time can be calculated as [(Setting + 1) X 8/fx].

When operating at 455 kHz, “MOV T,#1FFH” indicates that [(1FFH + 1) × 8/(455 × 10^3) ≒ 9.00 ms] has been set.

6.3.1 Description of processing

Transmission processing includes the following seven types of processing.

- Initialization
- Leader code transmission
- Code transmission (custom code, custom code’, data code, and $\overline{\text{data code}}$)
- Stop bit transmission
- Frame space transmission
- Transmission of second and subsequent frames
- OFF chattering elimination processing

Each of these types of processing is described below.

(1) Initialization

The initialization procedure for transmission processing is described in (a) and (b) below.

(a) The key OFF check counter (R07) is set to 10 times (= 05H).
(b) The data 1 transmit counter is set to 0DFH (for 24 bits).

Table 6-5 describes the counter contents.
The data 1 transmit counter is used to count the number of times bit data “1” is transmitted within custom code or custom code’ that can affect the frame space transmission time.
The number of “data 1” occurrences in the data code and $\overline{\text{data code}}$ is fixed (at eight) and therefore does not affect the frame space transmission time.

Table 6-5. Description of Data 1 Transmit Counter (R9)

Counter value	Data 1 transmit counter description
D0H to E6H	Number of “data 1” occurrences: 0 to 7 This status does not exist since the number of “data 1” occurrences in data code and $\overline{\text{data code}}$ is fixed (at eight).
E7H	Number of “data 1” occurrences: 8 Number of “data 1” occurrences in custom code and custom code’: 0
E8H to EFH	Number of “data 1” occurrences: 9 to 16 Number of “data 1” occurrences in custom code and custom code’: 1 to 8
F0H to F7H	Number of “data 1” occurrences: 17 to 24 Number of “data 1” occurrences in custom code and custom code’: 9 to 16

(2) Leader code transmission

The transmission method for the leader code is described in (a) to (f) below.

(a) The table reference address (1A8H) for the leader code is set as follows.
- High-order address (0AH) is set to data pointer H (R10)
- Low-order address (8H) of “data 0” in bit data is set to data 0 (R08)

(b) Output data 0H for leader code is set to work area 2 (R1A).

(c) A subroutine (BITOUT0F) is called to transmit the leader code as a one-bit transmission.

(d) Work area 2 (R1A) is shifted leftward and a bit judgment is performed.

(e) The table reference address L for the judged bit is set to data pointer L to enable the transmission time to be set to the timer counter.

(f) The leader code (H: 9.00 ms, L: 4.50 ms) is transmitted.

(3) Code transmission (custom code, custom code’, data code, and $\overline{\text{data code}}$)

The code transmission method is described in (a) to (h) below.

(a) The table reference addresses (1AAH and 1ACH) for bit data used for code transmission are set as follows.
- High-order address (0AH) is set to data pointer H (R10)
- Low-order address (0AH) of “data 0” in bit data is set to data 0 (R08)
- Low-order address (0CH) of “data 1” in bit data is set to data 1 (R18)

(b) The following data is set to work area 2 (R1A) when each type of code is transmitted.
- Custom code H (R14)
- Custom code L (R04)
- Custom code H’ (R15)
- Custom code L’ (R05)
- Data code H (R16)
- Data code L (R06)
- $\overline{\text{Data code H}}$ (R16 is fully inverted)
- $\overline{\text{Data code L}}$ (R06 is fully inverted)

(c) A subroutine is called to transmit the each type of code as a four-bit transmission.

(d) Work area 2 (R1A) is shifted leftward and a bit judgment is performed. If “data 1” is judged, the number of “data 1” occurrences is counted to enable frame space transmission.

(e) The table reference address L for the judged bit is set to data pointer L to enable the transmission time to be set to the timer counter.

(f) Each type of code is transmitted.
- “Data 0” ... H: 0.56 ms, L: 0.56 ms
- “Data 1” ... H: 0.56 ms, L: 1.69 ms

A key OFF check is performed during low level transmission of each bit.

For a description of the key OFF check, see **(7) OFF chattering elimination processing** below.

(g) Steps (d) to (f) above are repeated until four bits have been transmitted.

(h) Steps (b) to (g) above are repeated until all transmissions from custom code H to $\overline{\text{data code L}}$ are completed.

(4) Transmission of stop bit

The method for transmitting stop bits is described in (a) to (f) below.

(a) The table reference address (1AEH) for the stop bit is set as follows.
- High-order address (0AH) is set to data pointer H (R10)
- Low-order address (0EH) of “data 0” in bit data is set to data 0 (R08)

(b) Output data 0H for stop bit is set to work area 2 (R1A).

(c) A subroutine is called to transmit the stop bit as a one-bit transmission.

(d) Work area 2 (R1A) is shifted leftward and a bit judgment is performed.

(e) The table reference address L for the judged bit is set to data pointer L to enable the transmission time to be set to the timer counter.

(f) The stop bit (H: 0.56 ms, L: 3.00 ms) is transmitted.

(5) Frame space transmission

The code transmission time differs between transmission of the “data 1” and “data 0” bit data, with variation in the range of 59.06 ms to 77.06 ms. Therefore, when transmitting code, the frame space transmission time varies according to the number of “data 1” occurrences.

However, since the number of “data 1” occurrences is fixed (at eight) within the data code and $\overline{\text{data code}}$, the frame space transmission times are not affected when these types of code are transmitted. The number of “data 1” occurrences in the custom code and custom code’ do affect frame space transmission times.

The number of “data 1” occurrences are counted during transmission to enable responses to changes in transmission times. The frame space transmission times can be adjusted based on the count values. For details of “data 1” counts, see **(3) Code transmission** above.

Table 6-6 lists frame space times correspond to the number of “data 1” occurrences.

Since the maximum value that can be set to the timer counter is 9.00 ms, the frame space is divided into the three patterns shown in Figure 6-11 before being transmitted.

Differences between times shown in Table 6-6 and Figure 6-11 are margins of error during code transmission (such as when the “data 0” low level time of 0.565 ms is set as 0.56 ms) which are absorbed by frame spaces.

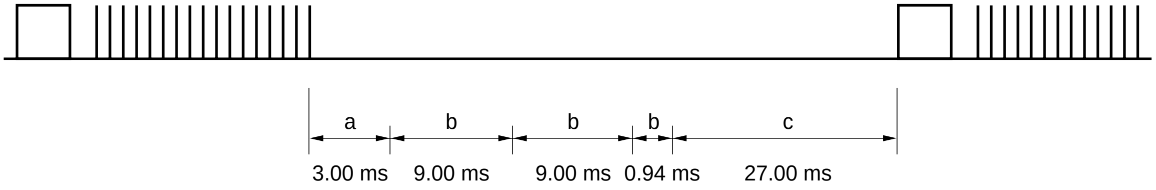
Table 6-6. Frame Space Times Corresponding to the Number of "Data 1" Occurrences

No. of "Data 1" occurrences	Transmission time (ms)	No. of "Data 1" occurrences	Transmission time (ms)	No. of "Data 1" occurrences	Transmission time (ms)	No. of "Data 1" occurrences	Transmission time (ms)
8	48.940	13	43.315	18	37.690	23	32.065
9	47.815	14	42.190	19	36.565	24	30.940
10	46.690	15	41.065	20	35.440		
11	45.565	16	39.940	21	34.315		
12	44.440	17	38.815	22	33.190		

Remark Since the number of "data 1" occurrences in data code and $\overline{\text{data code}}$ is fixed (at eight), the "data 1" values are never between 0 and 7.

Figure 6-11. Method for Dividing Frame Space Transmission Times (1/2)

(a) When there are eight "data 1" occurrences (data 1 transmission counter value: 0E7H)

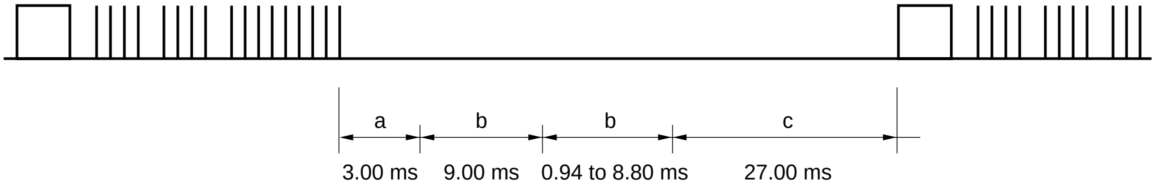


a: After transmitting the stop bit, the frame space is transmitted for 3.00 ms.

b: A frame space is transmitted for the period corresponding to the number of "data 1" occurrences (18.94 ms).

c: If a key has been pressed and held, ON chattering elimination processing (9.00 ms × 3 times) is performed. In other cases, a frame space is transmitted for 27.00 ms.

(b) When there are from 9 to 16 "data 1" occurrences (data 1 transmission counter value range: 0E8H to 0EFH)



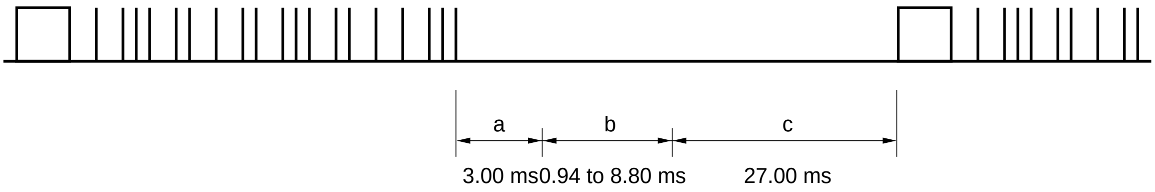
a: After transmitting the stop bit, the frame space is transmitted for 3.00 ms.

b: A frame space is transmitted for the period corresponding to the number of "data 1" occurrences (9.94 ms to 17.80 ms).

c: If a key has been pressed and held, ON chattering elimination processing (9.00 ms × 3 times) is performed. In other cases, a frame space is transmitted for 27.00 ms.

Figure 6-11. Method for Dividing Frame Space Transmission Times (2/2)

(c) When there are from 17 to 24 “data 1” occurrences (data 1 transmission counter value range: 0F0H to 0F7H)



a: After transmitting the stop bit, the frame space is transmitted for 3.00 ms.

b: A frame space is transmitted for the period corresponding to the number of “data 1” occurrences (0.94 ms to 8.80 ms).

c: If a key has been pressed and held, ON chattering elimination processing (9.00 ms × 3 times) is performed. In other cases, a frame space is transmitted for 27.00 ms.

(6) Transmission of second and subsequent frames

Until the current key is released or changed, code transmission of second and subsequent frames is repeated starting with the leader code in the same manner as for the first frame. This transmission method is described in (2) to (5) above.

For a description of the key OFF check, see **(7) OFF chattering elimination processing** below.

(7) OFF chattering elimination processing

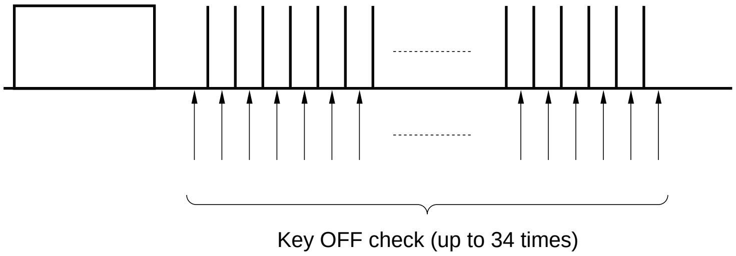
A key OFF check is performed during low level output of bit data.

Key OFF status is confirmed only when absence of key input (i.e., key OFF) has been judged for ten consecutive times during 34 times of low level output that includes the leader code and stop bit. If even one key input (key ON) is detected during the ten consecutive times being checked, the ten-time counter is cleared and the count is restarted.

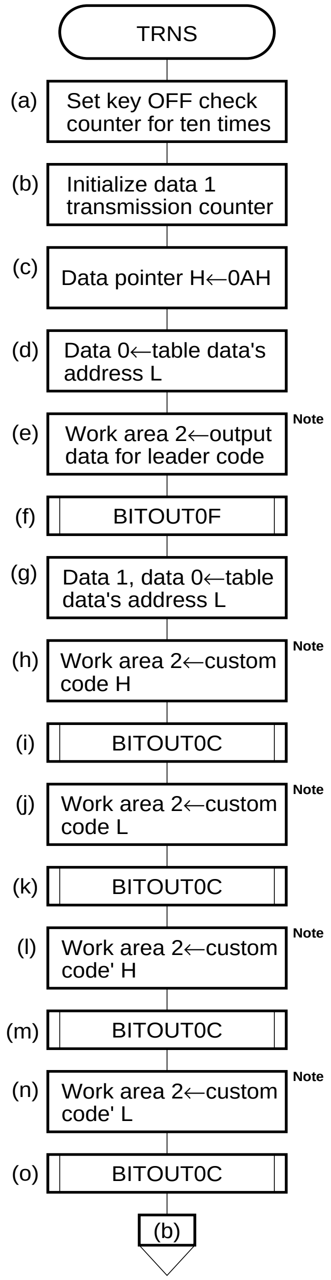
If key OFF status is not confirmed during the entire 34-time check, the key is judged as being pressed and held.

Even if key OFF status is confirmed during transmission of one frame, initialization processing is not performed until at least two frames have been transmitted.

Figure 6-12. Key OFF Check during Bit Data Transmission

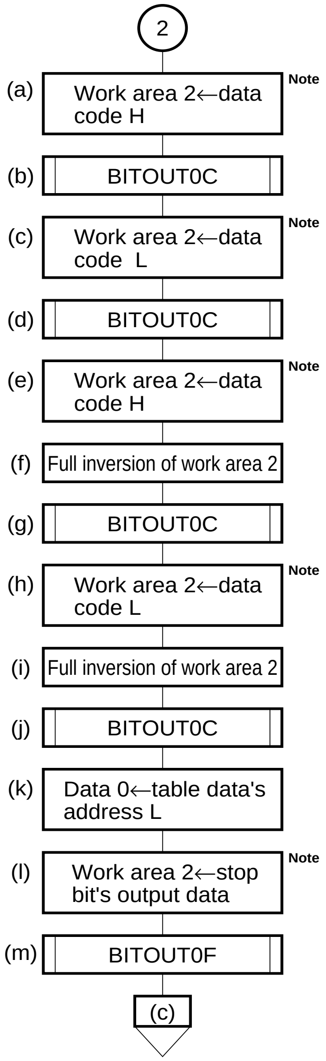


6.3.2 Detailed flow chart


TRNS
(a) Set key OFF check counter for ten times
(b) Initialize data 1 transmission counter
(c) Data pointer H←0AH
(d) Data 0←table data's address L
(e) Work area 2←output data for leader code
Note
(f) BITOUT0F
(g) Data 1, data 0←table data's address L
(h) Work area 2←custom code H
Note
(i) BITOUT0C
(j) Work area 2←custom code L
Note
(k) BITOUT0C
(l) Work area 2←custom code' H
Note
(m) BITOUT0C
(n) Work area 2←custom code' L
Note
(o) BITOUT0C
(b)

Note As part of the main routine, the output data for the leader code is set to the accumulator (A) and the same data is set to work area 2 (R1A) via the called subroutine (BITOUT0C).

```
******** TRANSMISSION PROCESSING ********

        TIME9M  EQU     1FFH                            ;   9.00ms(9.002ms)

;##########  P  U  B  L  I  C  ##########
        ;

;##########  E  X  T  E  R  N  ##########
        EXTRN           MAIN                            ;   MAIN Routine
        EXTRN           KEY80                           ;   Key Check Routine
        EXTRN           FS_27MS                         ;   Frame Space Subroutine

;##########   S  T  A  R  T   ##########
TRNS:
        MOV     R7,#0F5H                                (a) Sets the key OFF check counter (R07) for ten
                                                            times ( = 5H).
                                                            R17 is used for immediate setting (R17 = 0FH).
        MOV     R9,#0DFH                                (b) Initializes (sets 0FH to) the data 1 transmission
                                                            counter (R09).
;***********************************************
;               Leader Code                             Remark Transmission of leader code (H: 9.00 ms,
;***********************************************               L: 4.50 ms)
        MOV     R0,#0A0H                                (c) Sets the 0AH (high-order address of the table
                                                            data for the bit data transmission time) to data
                                                            pointer H (R10).
        MOV     R8,#008H                                (d) Sets the low-order address of the table data for
                                                            the bit data transmission time to data 1 (R18)
                                                            and data 0 (R08) (R18 ← 0H, R08 ← 8H).
        MOV     A,R18                                   (e) Sets the output data (0H) for leader code to the
                                                             accumulator.  (The accumulator's value is
                                                            transferred to work area 2 as part of the bit
                                                            output subroutine.)
        CALL    BITOUT0F                                (f) Calls the bit output subroutine (for one-bit
                                                            transmission:  counter value = 0FH).
;***********************************************
;               Custom Code                             Remark Transmission of custom code
;***********************************************               (H: 0.56 ms, L: 0.56 or 1.69 ms)
        MOV     R8,#0CAH                                (g) Sets the low-order address of the table data for
                                                            the custom code to data 1 (R18) and data 0
                                                            (R08) (R18 ← CH, R08 ← AH).
        MOV     A,R14                                   (h) Sets output data H (R14) for custom code to the
                                                            accumulator. (The accumulator's value is
                                                            transferred to work area 2 as part of the bit
                                                            output subroutine.)
        CALL    BITOUT0C                                (i) Calls the bit output subroutine (for four-bit
                                                            transmission:  counter value = 0CH).
        MOV     A,R04                                   (j) Sets output data L (R04) for custom code to the
                                                            accumulator. (The accumulator's value is
                                                            transferred to work area 2 as part of the bit
                                                            output subroutine.)
        CALL    BITOUT0C                                (k) Calls the bit output subroutine (for four-bit
                                                            transmission:  counter value = 0CH).

;***********************************************
;               Custom Code'                            Remark Transmission of custom code'
;***********************************************               (H: 0.56 ms, L: 0.56 or 1.69 ms)
        MOV     A,R15                                   (l) Sets output data H (R15) for custom code' to
                                                            the accumulator. (The accumulator's value is
                                                            transferred to work area 2 as part of the bit
                                                            output subroutine.)
        CALL    BITOUT0C                                (m) Calls the bit output subroutine (for four-bit
                                                            transmission:  counter value = 0CH).
        MOV     A,R05                                   (n) Sets output data L (R05) for custom code' to
                                                            accumulator. (The accumulator's value is
                                                            transferred to work area 2 as part of the bit
                                                            output subroutine.)
        CALL    BITOUT0C                                (o) Calls the bit output subroutine (for four-bit
                                                            transmission:  counter value = 0CH).
```



Note As part of the main routine, the output data for the leader code is set to the accumulator (A) and the same data is set to work area 2 (R1A) by the called subroutine (BITOUT0C).

```
;***********************************************
;               Data Code
;***********************************************
        MOV     A,R16
        CALL    BITOUT0C
        MOV     A,R06
        CALL    BITOUT0C
```

Remark Transmission of data code
(H: 0.56 ms, L: 0.56 or 1.69 ms)

(a) Sets output data H (R_{16}) for the data code to the accumulator.
(The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(b) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

(c) Sets output data L (R_{06}) for the data code to the accumulator.
(The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(d) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

```
;***********************************************
;               Data Code’
;***********************************************
        MOV     A,R16
        XRL     A,#0FH
        CALL    BITOUT0C
        MOV     A,R06
        XRL     A,#0FH
        CALL    BITOUT0C
```

Remark Transmission of $\overline{\text{data code}}$
(H: 0.56 ms, L: 0.56 or 1.69 ms)

(e) Sets output data H (R_{16}) for the data code to the accumulator.

(f) The accumulator’s value is inverted to create output data for $\overline{\text{data code}}$ H.
(The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(g) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

(h) Sets output data L (R_{06}) for the data code to the accumulator.

(i) The accumulator’s value is inverted to create output data for $\overline{\text{data code}}$ L.
(The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(j) Calls the bit output subroutine (for four-bit transmission: counter value = 0CH).

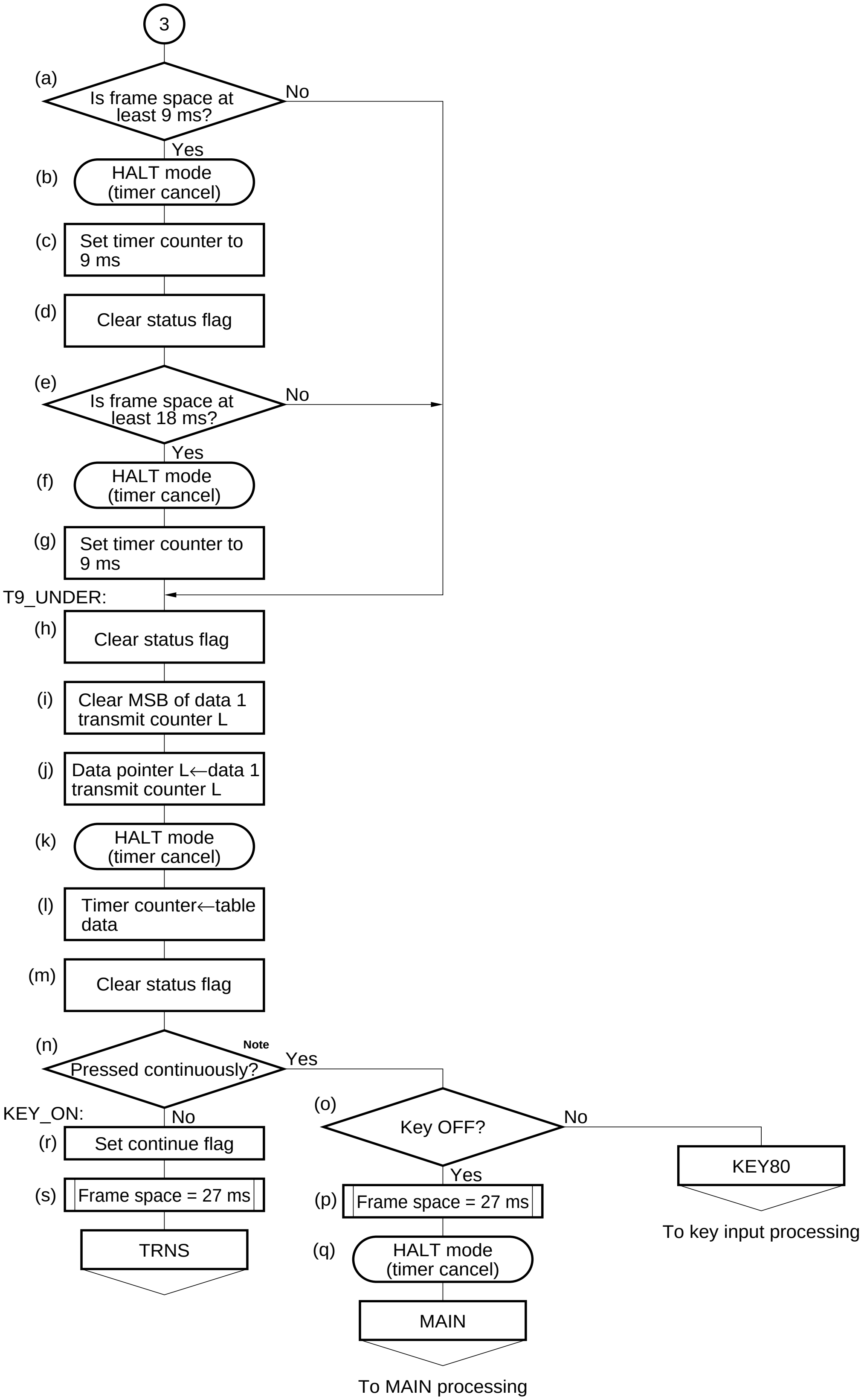
```
;***********************************************
;               Stop Bit
;***********************************************
        MOV     R8,#00EH
        MOV     A,R18
        CALL    BITOUT0F
```

Remark Transmits stop bit
(H: 0.56 ms, L: 3 ms)

(k) Sets the low-order address of table data having a stop bit carrier to data 0 (R_{08}).

(l) Sets the output data (0H) for the stop bit to the accumulator.
(The accumulator’s value is transferred to work area 2 as part of the bit output subroutine.)

(m) Calls the bit output subroutine (for one-bit transmission: counter value = 0FH).



Note If pressed continuously, continue flag = 0FH.

```
;***********************************************
;               Frame Space
;***********************************************
        MOV     A,R19
        SCAF
        JC      T9_UNDER

        HALT    #005H

        MOV     T,#TIME9M

        STTS    #0101B

        MOV     A,R09
        RL      A
        JC      T9_UNDER

        HALT    #005H

        MOV     T,#TIME9M

T9_UNDER:
        STTS    #0101B
        MOV     A,R09
        ANL     A,#0111B
        MOV     R00,A

        HALT    #005H

        MOV     T,@R0



        STTS    #0101B

        MOV     A,R03
        SCAF
        JNC     KEY_ON
        MOV     A,R07
        SCAF
        JNC     KEY80

;**  Frame Space = 27ms  **
        CALL    FS_27MS


        HALT    #005H

        JMP     MAIN
        ;
KEY_ON:
        MOV     A,R17
        MOV     R03,A
  ;**  Frame Space = 27ms  **
        CALL    FS_27MS
        JMP     TRNS
        ;
```

Remark Frame space transmission (L: 30.94 ms to 48.94 ms)

(a) Judges the number of bit data 1 occurrences for custom code (16 bits) as eight or less.
If nine or above, processing branches to T9_UNDER.

(b) HALT mode: canceled when time counter reaches 00H.

(c) If eight or less, sets timer counter to 9 ms (= 1FFH).

(d) Clears status flag.

(e) Judges the number of bit data 1 occurrences for custom code (16 bits) as zero or not zero.
If 1 to 8, processing branches to T9_UNDER.

(f) HALT mode: canceled when time counter reaches 00H.

(g) If zero, sets timer counter to 9 ms (= 1FFH) again for a total transmission time of 18 ms.

(h) Clears status flag.

(i) Clears the MSB of the data 1 transmit counter L (R$_{09}$).

(j) Sets value of data 1 transmit counter L (R$_{09}$) to data pointer L (R$_{00}$).

(k) HALT mode: canceled when time counter reaches 00H.

(l) Performs table reference to set a value from 0.940 ms to 8.815 ms to the timer counter according to the number of bit data 1 occurrences.

(m) Clears status flag.

(n) Judges transmission of first frame (to check for continue status). If transmitting the first frame, processing branches to KEY_ON.

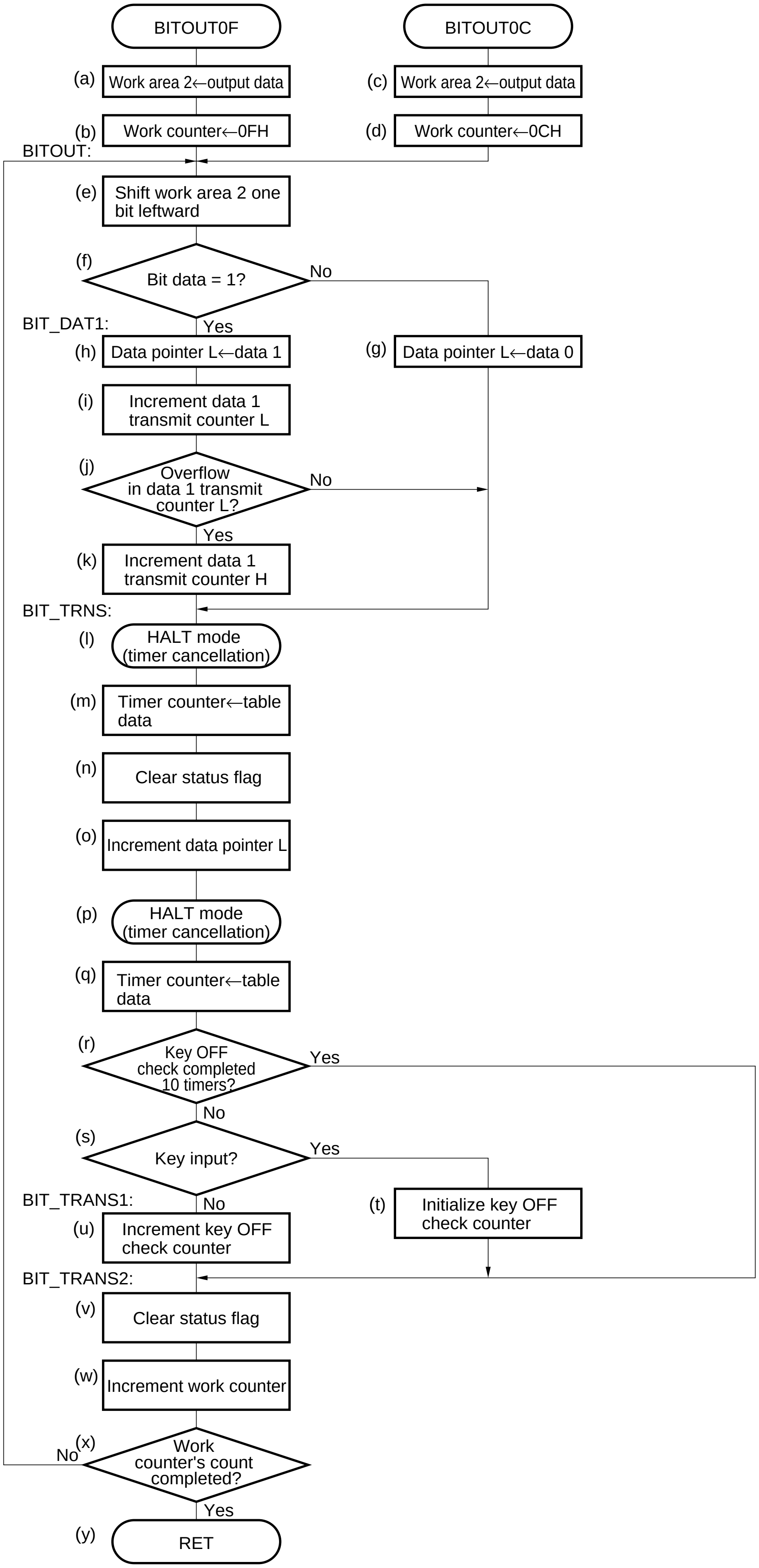
(o) If transmitting the second or subsequent frame (continue status), it judges whether a key OFF status occurs during the transmission. If continue status is detected, processing branches to KEY80.

(p) If a key OFF status is detected during bit data transmission, the remaining 27 ms of the frame space is output.

(q) HALT mode: canceled when time counter reaches 00H.
Processing branches to initialization processing and MAIN.

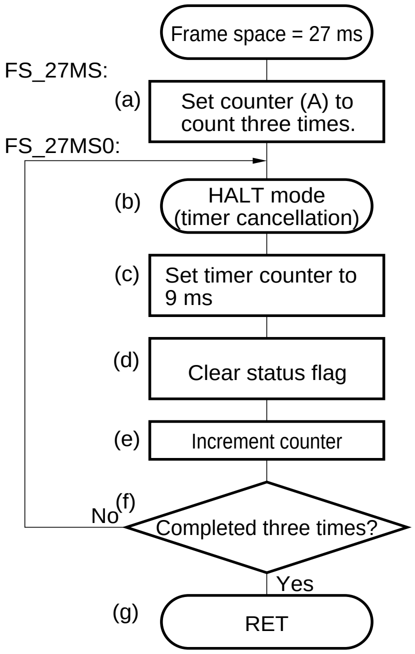
(r) Sets continue flag (R$_{03}$) (in this case, R$_{17}$ = OFH).

(s) Calls subroutine for 27-ms output.
Processing branches to TRNS.


BITOUT0F
BITOUT0C
(a) Work area 2←output data
(c) Work area 2←output data
(b) Work counter←0FH
(d) Work counter←0CH
BITOUT:
(e) Shift work area 2 one bit leftward
(f) Bit data = 1?
No
Yes
BIT_DAT1:
(h) Data pointer L←data 1
(g) Data pointer L←data 0
(i) Increment data 1 transmit counter L
(j) Overflow in data 1 transmit counter L?
No
Yes
(k) Increment data 1 transmit counter H
BIT_TRNS:
(l) HALT mode (timer cancellation)
(m) Timer counter←table data
(n) Clear status flag
(o) Increment data pointer L
(p) HALT mode (timer cancellation)
(q) Timer counter←table data
(r) Key OFF check completed 10 timers?
Yes
No
(s) Key input?
Yes
BIT_TRANS1:
No
(t) Initialize key OFF check counter
(u) Increment key OFF check counter
BIT_TRANS2:
(v) Clear status flag
(w) Increment work counter
(x) Work counter's count completed?
No
Yes
(y) RET

```
;*************************************************
;*****                                       *****
;*****             Transmit                  *****
;*****         Subroutine : Bit Out          *****
;*************************************************
BITOUTOF:
        MOV     R1A,A                                 (a) Sets output data to work area 2 (R1A).
        MOV     A,#0FH                                (b) Sets 0FH to accumulator to set the transmit bit
                                                          counter to count one time.
        JMP     BITOUT                                    Processing branches to BITOUT.
;
BITOUTOC:
        MOV     R1A,A                                 (c) Sets output data to work area 2 (R1A).
        MOV     A,#0CH                                (d) Sets 0CH to accumulator to set the transmit bit
                                                          counter to count four times.
BITOUT:
        MOV     R0A,A                                 Remark 0FH, 0EH, or 0CH can be set to the transmit
                                                             bit counter (R0A).

        MOV     A,R1A      }
        RL      A          }------------------------  (e) Shifts the output data one bit leftward.
        MOV     R1A,A      }
        JC      BIT_DAT1                              (f) Judges whether bit data value is "0" or "1".
                                                          If it is "1", processing branches to BIT_DAT1.
;**  Bit Data = 0  **
        MOV     A,R08      }
        MOV     R00,A      }------------------------  (g) Sets data 0 (R08) to data pointer L (R00).
        JMP     BIT_TRN    }                              Processing branches to BIT_TRNS.
        ;
;**  Bit Data = 1  **;
BIT_DAT1:
        MOV     A,R18      }------------------------  (h) Sets data 1 (R18) to data pointer L (R00).
        MOV     R00,A      }
        MOV     A,R09      }
        INC     A          }------------------------  (i) Increments data 1 transmit counter L (R09).
        MOV     R09,A      }
        JNC     BIT_TRANS                             (j) If there is no overflow in data 1 transmit counter
                                                          L (R09), processing branches to BIT_TRNS.
        MOV     A,R19      }
        INC     A          }------------------------  (k) Increments data 1 transmit counter H (R19).
        MOV     R19,A      }
BIT_TRNS:
        HALT    #005H                                 (l) HALT mode:  canceled when time counter
                                                          reaches 00H.
        MOV     T,@R0                                 (m) Performs table reference to set transmit time data
                                                          to timer counter.
        STTS    #0101B                                (n) Clears status flag.
        MOV     A,R00      }
        INC     A          }------------------------  (o) Increments data pointer L (R00).
        MOV     R00,A      }
        MOV     R00,A
        HALT    #005H                                 (p) HALT mode: canceled when time counter reaches
                                                          00H.
        MOV     T,@R0                                 (q) Performs table reference to set transmit time data
                                                          to timer counter.
        MOV     A,R07      }
        INC     A          }------------------------  (r) Judges whether key OFF status is maintained
        JC      BIT_TRANS2 }                              during ten consecutive times.
                                                          If key OFF status is confirmed, processing
                                                          branches to BIT_TRANS2.
        STTS    #1110B     }------------------------  (s) Determines whether or not key input exists.
        JNF     BIT_TRANS1 }                              If a key has been pressed and held, processing
                                                          branches to BIT_TRANS1.
;**  Key input exists  **
        MOV     R7,#0F5H                              (t) Initializes (= 5H) the key OFF check counter
                                                          (R07) if key input exists even once during the
                                                          (10-time) key OFF check.
        JMP     BIT_TRANS2                                Processing branches to BIT_TRANS2.
        ;
```

```
    ;**  No key input  **
BIT_TRANS1:
        MOV     R07,A                                       (u)  Increments key OFF check counter.
BIT_TRANS2:
        STTS    #0101B                                      (v)  Clears status flag.
        MOV     A,R0A      }
        INC     A          }---------------------------     (w) Increments transmit bit counter (R0A).
        JNC     BITOUT                                      (x)  Determines whether or not the transmit bit count
                                                                 has been completed. If not completed, processing
                                                                 branches to BITOUT.
        RET                                                 (y)  End of processing
        ;
END
```


Frame space = 27 ms
FS_27MS:
(a)
Set counter (A) to count three times.
FS_27MS0:
(b)
HALT mode (timer cancellation)
(c)
Set timer counter to 9 ms
(d)
Clear status flag
(e)
Increment counter
(f)
No
Completed three times?
Yes
(g)
RET

```
;*************************************************
;*****                                       *****
;*****          FRAME SPACE = 27MS           *****
;*****         Subroutine : FS 27ms          *****
;*****                                       *****
;*************************************************
FS_27MS:
        MOV     A,#0DH

FS_27MS0:
        HALT    #005H

        MOV     T,#TIME9M
        STTS    #0101B
        INC     A
        JNC     FS_27MS0



        RET

END
```

Remark Subroutine for outputting remaining 27 ms of frame space.

(a) Sets counter (accumulator) to count three times (= 0DH).

(b) HALT mode: canceled when time counter reaches 00H.

(c) Sets timer counter to 9 ms (= 1FFH).

(d) Clears status flag.

(e) Increments counter (accumulator).

(f) Determines whether or not three times have been counted.
If they have not been counted, processing branches to FS_27MS0.

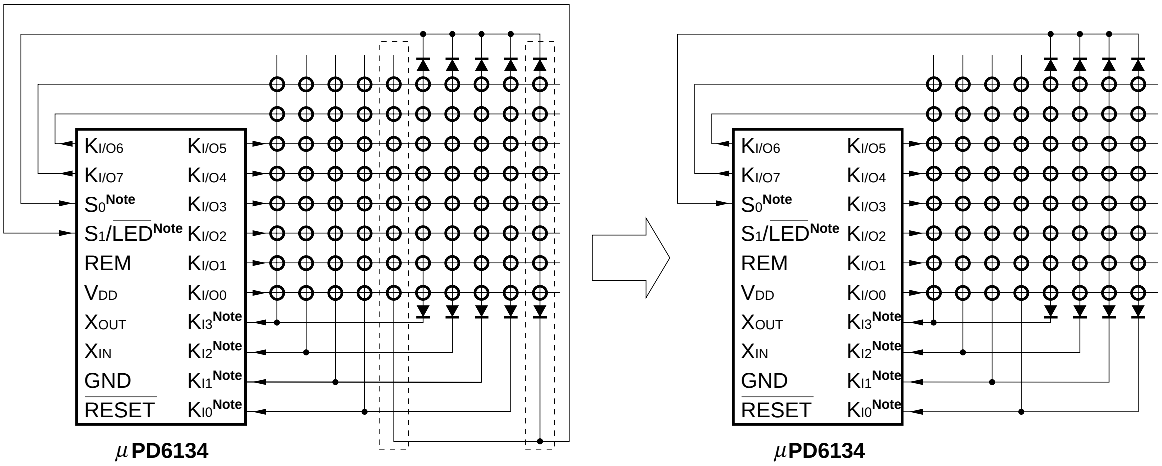
(g) End of processing

CHAPTER 7 CAUTIONS ON PROGRAM REVISIONS

Note the following caution points when modifying the key matrix or the number of keys.

(1) When changing the number of keys from 80 to 64

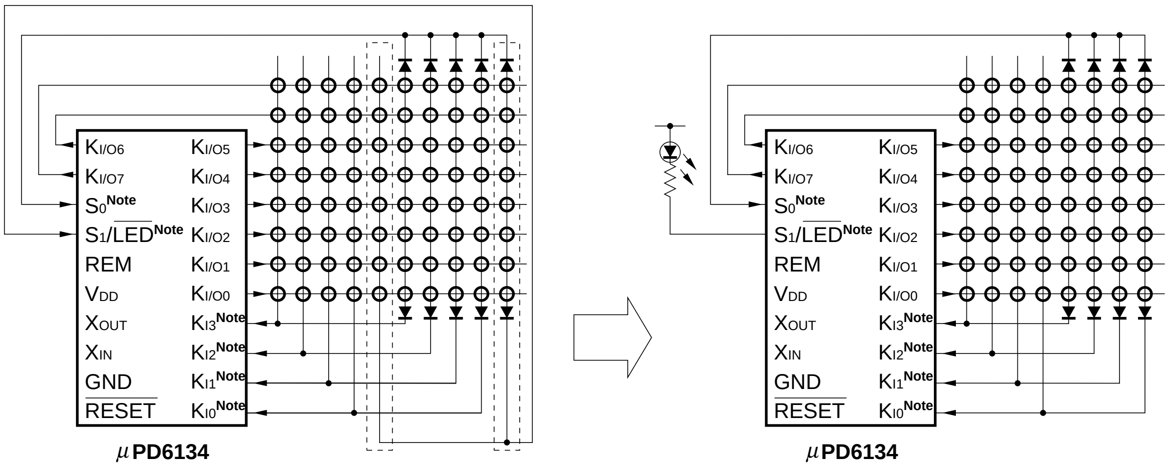
- Delete the keys that are enclosed in broken lines in the following diagram (when doing so, if the S1/$\overline{\text{LED}}$ pin is left unconnected, the number of keys can be changed without modifying the main program itself.



Note Set by the program for “internal pull-down resistor”.

(2) In addition to the modification described in (1) above, the S1/$\overline{\text{LED}}$ pin can be used as an LED pin:

- Delete the keys that are enclosed in broken lines in the following diagram.
- Change the S1/$\overline{\text{LED}}$ pin so that b2 (the bit that sets the I/O mode for the S1/$\overline{\text{LED}}$ port) in the main program’s control register 1 (P4) remains in output mode (bit value = 1). (In μPD6133 Series products, setting the S1/$\overline{\text{LED}}$ pin to output mode automatically eliminates internal pull-down resistance.)



Note Set by the program for “internal pull-down resistor”.

CHAPTER 8 CAUTIONS ON USE OF THIS PROGRAM

The key combination specifications in this program provide only three valid key combination patterns: K29 + K30, K29 + K31, and K29 + K32. For this key matrix, if an ordinary key (K1 to K40) is pressed in combination with an expansion key (K41 to K80) on the same key source line, the expansion key will always be the valid key.

For example, if K19 is pressed in combination with K59, then K59 is the valid key.

[MEMO]

CHAPTER 9 PROGRAM LIST

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>  01:00:00 08/29/96 PAGE 01-002


SOURCE = MAIN80.ASM

E STNO LOC. 0BJ.                M  I   SOURCE STATEMENT
     1                                  ;*********************************************************************
     2                                  ;***                                                               ***
     3                                  ;***     Multi-Purpose Remote Control Transmitter System           ***
     4                                  ;***                                                               ***
     5                                  ;***     CPU        : uPD6133 Series                               ***
     6                                  ;***     CPU Clock  : 455kHz                                       ***
     7                                  ;***     Trans. CODE: NEC-R Format (80Key)                         ***
     8                                  ;***     Version    : 2.0                                          ***
     9                                  ;***     Programmer : NEC IC Microcomputer Systems Corporation     ***
    10                                  ;***                                                               ***
    11                                  ;***                    Copyright(c)   NEC Corporation   1995      ***
    12                                  ;***                    Copyright(c)   NIMS Corporation  1995      ***
    13                                  ;*********************************************************************
    14                                  EJECT
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>  01:00:00 08/29/96 PAGE 01-003


SOURCE = MAIN80.ASM

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
    15      001C                                TIME05M      EQU     01CH                        ;0.5ms (0.510ms)
    16      01FF                                DamyTime     EQU     512-1                       ;9.00MS (= 17.582us * 512)
    17
    18                                  ;##########   P   U   B   L   I   C   ##########
    19                                          PUBLIC       MAIN
    20                                          ;
    21                                  ;##########   E   X   T   E   R   N   ##########
    22                                          ;
    23
    24                                  ;##########   S    T    A    R    T   ##########
    25
    26                                          ; Control Register (P3)   Reset:03H
    27                                          ;===============================================================;
    28                                          ;  D9   D8 !  D7 !  D6 !  D5 !  D4 !  D3 !  D2 !  D1 !  D0 !    ;
    29                                          ;          !     ! D.P.! D.P.! D.P.! TCTL! CARY! MOD1! MOD0!    ;
    30                                          ; Test Mode!     ! AD10! AD9 ! AD8 ! AD8 !     !     !     !    ;
    31                                          ;---------------------------------------------------------------;
    32                                          ;          !  0  !  *  !  *  ! 1/1 !  ON !      fx,fx/8    ! 0  ;
    33                                          ; Set “0”  !-----------------------------!   ,fx/12(1/2)   !----;
    34                                          ;          !  0  !  *  !  *  ! 1/2 ! OFF !   ,fx/12(1/3)   ! 1  ;
    35                                          ;===============================================================;
    36
    37                                          ; Control Register (P4)    Reset:26H
    38                                          ;===============================================================;
    39                                          ;  D9   D8 !  D7 !  D6 !  D5 !  D4 !  D3 !  D2  !  D1 !  D0 !   ;
    40                                          ;          !     !     !   KI!S0/S1!     !S1/LED! KI/0!  S0 !   ;
    41                                          ;          !     !     ! Pull! Pull!     ! MODE ! MODE! MODE!   ;
    42                                          ;---------------------------------------------------------------;
    43                                          ;          !  0  !  0  ! OFF ! OFF !  0  !  IN  !  IN ! OFF ! 0 ;
    44                                          ;      X   !------------------------------------------------!---;
    45                                          ;          !  0  !  0  !  ON !  0N !  0  ! OUT  ! OUT !  IN ! 1 ;
    46                                          ;===============================================================;
    47
    48                                  ;***********************************************
    49                                  ;       Control Register Initialize
    50                                  ;***********************************************
    51                                  MAIN:
    52 0000 E6F8 EFEF                           OUT     P0,#0FFH                                 ;KI/0 All High
    53 0002 E6FC E3E2                           OUT     P4,#032H
    54 0004 E6FB E1E3                           OUT     P3,#013H                                 ;Set Data Pointer (P3:D8,D9)
    55 0006 E6FF F7FF                           MOV     T,#DamyTime
    56 0008 E3F1 E0E5                           STTS    #0101B                                   ;Clear Status Flag
    57
    58                                  ;***********************************************
    59                                  ;       RAM Initialize Routine
    60                                  ;***********************************************
    61 000A E6E1 E0E0                           MOV     R1,#000H                                 ;Final Key Data (R1) = 00h
    62 000C E6E3 E0E0                           MOV     R3,#000H                                 ;K29 ON Flag (R13),Continuance Falg (R03) = 00H
    63
    64 000E E2F1 E0E8                           HALT    #008H                                    ;STOP mode (KI = High)
    65 0010 E0E0                                NOP                                              ;No operation command
    66 0011 E6FF E0E7                           MOV     T,#TIME05M                               ;Set Timer : 0.5ms
    67                                          ;
    68                                  END
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>  01:00:00 08/29/96 PAGE 02-002


SOURCE = KEY80.ASM

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
    1                                   ;*************************************************
    2                                   ;*****                                       *****
    3                                   ;*****       uPD6133 Series Key Check        *****
    4                                   ;*****   Trans. CODE: NEC-R Format (80Key)   *****
    5                                   ;*****                                       *****
    6                                   ;*************************************************
    7      01FF                                 TIME9M  EQU     1FFH                            ;9.00ms (9.002ms)
    8      0050                                 CUSTM1  EQU     050H                            ;Custom Code  = 0AH
    9      00AF                                 CUSTM1  EQU     0AFH                            ;Custom Code’ = F5H
   10
   11                                   ;##########   P   U   B   L   I   C   ##########
   12                                           PUBLIC          KEY80
   13                                           ;
   14
   15                                   ;##########   E   X   T   E   R   N   ##########
   16                                           EXTRN           MAIN                            ;MAIN Routine
   17                                           ;
   18
   19                                   ;##########   S    T    A    R    T   ##########
   20                                   KEY80:
   21 0013 E6E0 E0EC                            MOV     R0,#00CH                                ;Chattering Counter (R00) = 3 Times
   22 0015 E6EC E8E2                            MOV     RC,#082H                                ;Set Immediate Data : RC
   23                                   ;*************************************************
   24                                   ;               ON Chattering
   25                                   ;*************************************************
   26                                   ON_CHAT:
   27 0017 E3F1 E0E5                            STTS    #0101B                                  ;Clear Status Flag
   28 0019 E6F8 EFEF                            OUT     P0,#0FFH                                ;KI/0 All High
   29 001B E6FC E3E3                            OUT     P4,#033H
   30 001D E2F1 E0E5                            HALT    #005H                                   ;HALT mode (Timer = 00H)
   31 001F E6FF F7FF                            MOV     T,#TIME9M                               ;Timer = 9.00ms
   32 0021 E6E9 E0E0                            MOV     R9,#000H                                ;Comparative Key Data (R9) = 00H
   33 0023 E6E6 E0E8                            MOV     R6,#008H                                ;Key Flag (R16) = 00H, Key Scan Counter (R06) = 08H
   34 0025 E3F1 E0EE                            STTS    #1110B                                  ;With input Key (KI or S0 or S1) ?
   35 0027 EFF1 E0E0                                                  JNF   MAIN                ;       else MAIN:      (Without input Key)
   36                                                                                           ;       then            (With input Key)
   37 0029 E6E8 E0E1                            MOV     R8,#001H                                ;Key Scan Data (R8) Initialize : 001H
   38
   39                                   ;*************************************************
   40                                   ;                  Key Scan
   41                                   ;*************************************************
   42                                   KEYS_OUT:
   43 002B FFE8                                 MOV     A,R08
   44 002C E5F8                                 OUT     P00,A                                   ;Key Scan Data L (P00) Output
   45 OO2D FEE8                                 MOV     A,R18
   46 002E E4F8                                 OUT     P10,A                                   ;Key Scan Data H (P10) Output
   47                                        ;+++++++++++++++++++++++++++++
   48                                        ;++       100us Wait        ++
   49                                        ;+++++++++++++++++++++++++++++
   50 002F E6E2 EFE1                            MOV     R2,#0F1H                                ;Set Immediate Data : R2
   51 0031 E6EB ECE3                            MIV     RB,#0C3H                                ;Set Immediate Data : RB
   52 0033 E0E0                                 NOP
   53 0034 E0E0                                 NOP
   54 0035 E0E0                                 NOP
   55 0036 E0E0                                 NOP
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>  01:00:00 08/29/96 PAGE 02-003

SOURCE = KEY80.ASM

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
    56                                  ;***********************************************
    57                                  ;               Key Return Check
    58                                  ;***********************************************
    59 0037 E3F1 E0EB                           STTS    #1011B                                  ;With input Key (KI or S0 or S1) ?
    60 0039 EFF1 E9E8                                                   JNF     NO_KEYS         ;       else NO_KEYS:   (Without input Key)
    61                                                                                          ;       then            (With input Key)
    62 003B E3F1 E0E3                           STTS    #0011B                                  ;With input KI (P11)?
    63 003D EFF1 E4E8                           JNF     S0S1CALC                                ;       else S0S1CALC:  (With input KI)
    64                                                                                          ;       then            (Without input KI)
    65                                                          ;*******************************;       then            (KI(P11)Without input)
    66                                                          ;         KI Data Calculate
    67                                                          ;*******************************
    68 003F FFF9                                                          IN      A,P01
    69 0040 FCF3                                                          RL      A             ;With input S1(P01)?
    70 0041 ECF1 E0E0                                                     JC      MAIN          ;       else MAIN:      (With input S1)
    71                                                                                          ;       then            (Without input S1)
    72 0043 FEEB                                                          MOV     A,R1B         ;R1B = 0CH
    73 0044 E5EA                                                          MOV     R0A,A         ;Key Return Check Counter (R0A) = 4 Times (=0CH)
    74 0045 FEF9                                                          IN      A,P11
    75 0046 E8F1 E4ED                                                     JMP     KEY_RET0
    76                                                                    ;
    77

    78                                          ;*******************************
    79                                          ;       S0,S1 Data Calculate
    80                                          ;*******************************
    81                                  S0S1CALC:
    82 0048 E6E9 EBE0                           MOV     R9,#0B0H
    83 004A FEE2                                MOV     A,R12                                   ;R12 = 0FH
    84 004B E5EA                                MOV     R0A,A                                   ;Set Key Return Check Counter (R0A) = 1 Times (=0FH)
    85 004C FFF9                                IN      A,P01
    86
    87                                  KEY_RET0:
    88 004D E4E0                                MOV     R10,A                                   ;Work (R10) = P11 or P01
    89 004E E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
    90                                  KEY_RET:
    91 0050 FEE0                                MOV     A,R10
    92 0051 FEF3                                RIZ     A
    93 0052 E4E0                                MOV     R10,A
    94 0053 EDF1 E9E3                                                     JNC     KRC_INC       ;With input Key?
    95                                       ;** With input Key **                               ;       else KRC_INC:   (Without input Key)
    96 0055 FEE6                                MOV     A,R16                                   ;       then            (With input Key)
    97 0056 FAF3                                SCAF                                            ;Key Flag (R16) = 0FH?
    98 0057 EDF1 E6E7                           JNC     KEY_CALC                                ;       else KEY_CALC:  (Except for 0FH)
    99 0059 FEE3                                                          MOV     A,R13         ;       then            (= 0FH)
   100 005A FAF3                                                          SCAF                  ;K29 ON Flag (R13) = 0FH?
   101 005B EDF1 E0E0                                                     JNC     MAIN          ;       else MAIN:      (= Except for 0FH)
   102                                                                                          ;       then            (= 0FH)
   103 005D FFEC                                                          MOV     A,ROC         ;ROC = 02H
   104 005E F4E9                                                          XRL     A,R19
   105 005F FAF3                                                          SCAF                  ;Comparative Key Data = K29 (R19=ODH) ?
   106 0060 EDF1 E0E0                                                     JNC     MAIN          ;       then MAIN:      (Key = Except for K29)
   107 0062 FFE9                                                          MOV     A,R09         ;       else            (Key = K29)
   108 0063 F5EB                                                          XRL     A,R0B         ;R0B = 03H
   109 0064 FAF3                                                          SCAF                  ;Comparative Key Data = K29 (R09=0CH) ?
   110 0065 EDF1 E0E0                                                     JNC     MAIN          ;       then DBL_K_CHK: (Key = Except for K29)
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>  01:00:00 08/29/96 PAGE 02-004


SOURCE = KEY80.ASM

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
  111
  112                                           ;*******************************
  113                                           ;       Key Data Calculate
  114                                           ;*******************************
  115                                   KEY_CALC:
  116 0067 FFF1 F4E0                            MOV     A,#04H
  117 0069 F4E9                                 XRL     A,R19
  118 006A FAF3                                 SCAF                                            ;Comparative Key Data H (R19) = 0BH ?
  119 006B EDF1 E7E5                            JNC     KI_CALC                                 ;       else KI_CALC:   (KI Data)
  120                                                           ;** S1 data Calculation **      ;       then            (S1 Data)
  121 006D FFE6                                                        MOV     A,R06            ;Acc <-- Key Scan Counter (R06)
  122 006E FBF1 F7E0                                                   ANL     A,#0111B
  123 0070 E5E9                                                        MOV     R09,A
  124 0071 FEE0                                                        MOV     A,R10            ;Acc <-- Work 1 (R10)
  125 0072 FAEC                                                        ANL     A,R1C            ;R1C = 08H
  126 0073 E8F1 E8E8                                                   JMP     KL_CALC
  127                                                                  ;
  128                                       ;** KI data Calculation **
  129                                   KI_CALC:
  130 0075 FFE6                                 MOV     A,R06                                   ;Acc <-- Key Scan Counter (R06)
  131 0076 FCF3                                 RL      A
  132 0077 FCF3                                 RL      A                                       ;Left shift (2 Times)
  133 0078 E5E9                                 MOV     R09,A                                   ;Comparative Key Data L (R09) <-- Acc
  134 0079 FBE2                                 ANL     A,R02                                   ;Acc AND 0001B (R02=01H)
  135 007A FCEB                                 ORL     A,R1B                                   ;Acc OR 1100B (R1B=0CH)
  136 007B E4E9                                 MOV     R19,A                                   ;Comparative Key Data H (R19) <-- Acc
  137 007C FFF9                                 IN      A,P01                                   ;Acc <-- S1 and S0 Data
  138 007D FCF3                                 RL      A                                       ;Left shift (3 Times)
  139 007E FCF3                                 RL      A
  140 007F FCF3                                 RL      A
  141 0080 FBEC                                 ANL     A,ROC                                   ;Acc AND 0010B (R0C=02H)
  142 0081 FCE9                                 ORL     A,R19                                   ;Acc OR Comparative Key Data H (R19)
  143 0082 E4E9                                 MOV     R19,A                                   ;Comparative Key Data H (R19) <-- Acc
  144 0083 FFE9                                 MOV     A,R09                                   ;Acc <-- Comparative Key Data L (R09)
  145 0084 FAEB                                 ANL     A,R1B                                   ;Acc AND 1100B (R1B=0CH)
  146 0085 E5E9                                 MOV     R09,A                                   ;Comparative Key Data L (R09) <-- Acc
  147 0086 FFEA                                 MOV     A,R0A                                   ;Acc <-- Key Return Check Counter (R0A)
  148 0087 FBEB                                 ANL     A,R08                                   ;Acc AND 0011B (R08=03H)
  149                                   KL_CALC:
  150 0088 FDE9                                 ORL     A,R09                                   ;Acc OR Comparative Key Data L (R09)
  151 0089 E5E9                                 MOV     R09,A                                   ;Comparative Key Data L (R09) <-- Acc
  152
  153                                       ;** Double Key Check **
  154 008A FEE6                                 MOV     A,R16
  155 008B FAF3                                 SCAF                                            ;Key Flag (R16) = 0FH?
  156 008C EDF1 E9E1                            JNC     SINGLE_K                                ;       else SINGLE_K   (Single Key)
  157 008E FFF1 F9E0                                                   MOV     A,#09H           ;       then            (Double Key)
  158 0090 E4E9                                                        MOV     R19,A
  159                                   SINGLE_K:
  160 0091 FEE2                                 MOV     A,R12                                   ;R12 = 0FH
  161 0092 E4E6                                 MOV     R16,A
  162
  163                                   KRC_INC:
  164 0093 FFEA                                 MOV     A,R0A
  165 0094 F4F3                                 INC     A                                       ;Key Return Check Counter (R0A) increment
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>      01:00:00 08/29/96 PAGE 02-005


SOURCE = KEY80.ASM

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
   166 0095 E5EA                                MOV     R0A,A                                   ;End of Key Return Check ?
   167 0096 EDF1 E5E0                                                   JNC    KEY_RET         ;       else KEY_RET:   (Continuous)
   168                                                                                          ;       then            (End)
   169
   170                                  NO_KEYS:
   171 0098 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
   172 009A FFE6                                MOV     A,R06
   173 009B F4F3                                INC     A                                       ;Key Scan Counter(R06) increment
   174 009C E5E6                                MOV     R06.A                                   ;End of Key Scan number of 8 Time ?
   175 009D ECF1 EAEB                                                   JC     KEY_DATA        ;       then KEY_DATA:  (End)
   176 009F FEE8                                MOV     A,R18                                   ;       else            (Continuous)
   177 00A0 FCF3                                RL      A                                       ;Key Scan Data H (R18) Shift
   178 00A1 E4E8                                MOV     R18,A
   179 00A2 FFE8                                MOV     A,R08
   180 00A3 FCF3                                RL      A
   181 00A4 E5E8                                MOV     R08,A                                   ;Key Scan Data L (R08) Shift --> CY ?
   182 00A5 EDF1 E2EB                           JNC     KEYS_OUT                                ;       else KEYS_OUT:  (key Scan Data a low rank)
   183 00A7 E6E8 E1E0                                                   MOV    R8,#010H        ;       then            (key Scan Data a high rank)
   184 00A9 E8F1 E2EB                                                   JMP    KEYS_OUT
   185                                                                  ;
   186
   187                                  ;***********************************************
   188                                  ; Transmit Key Data = ON Chattering Key Data?
   189                                  ;***********************************************
   190                                  KEY_DATA:
   191 00AB E6F8 EFEF                           OUT     P0,#0FFH                                ;KI/0 All High
   192 00AD E6E8 EFE0                           MOV     R8,#0F0H                                ;Set Immediate Data : R8
   193 00AF FEE6                                MOV     A,R16
   194 00B0 FAF3                                SCAF                                            ;Key Flag (R16) = 0FH ?
   195 00B1 EDF1 E0E0                                                   JNC    MAIN            ;       else MAIN:      (=Except for 0FH)
   196 00B3 FFE0                                MOV     A,R00                                   ;       then            (= 0FH)
   197 00B4 F5EB                                XRL     A,R0B                                   ;R0B = #0011B
   198 00B5 FAF3                                SCAF                                            ;Chattering Counter (R00) = 1 Times (=0DH)?
   199 00B6 ECF1 ECE3                                                   JC     KEYCHECK        ;       then  KEYCHECK: (=0DH)
   200                                                                                          ;       else            (=0E-0FH)
   201 00B8 FEE4                                MOV     A,R14
   202 00B9 F4E8                                XRL     A,R18                                   ;R18 = 0FH
   203 00BA F4E9                                XRL     A,R19                                   :Acc = 0FH
   204 00BB FAF3                                SCAF                                            ;Key Data H (R14) = Comparative Key Data H (R19)
   205 00BC EDF1 E0E0                                                   JNC    MAIN            ;       else MAIN:      (Unmatch)
   206                                                                                          ;       then            (Match)
   207 00BE F5E4                                XRL     A,R04                                   ;Acc = 0FH
   208 00BF F5E9                                XRL     A,R09
   209 00C0 FAF3                                SCAF                                            ;Key Data L (R04) = Comparative Key Data L (R09)
   210 00C1 EDF1 E0E0                                                   JNC    MAIN            ;       else MAIN:      (Unmatch)
   211                                  KEYCHECK:                                               ;       then            (Match)
   212 00C3 FEE9                                MOV     A,R19
   213 00C4 E4E4                                MOV     R14,A                                   ;Key Data H (R14) ← Comparative Key Data H (R19)
   214 00C5 FFE9                                MOV     A,R09
   215 00C6 E5E4                                MOV     R04,A                                   ;Key Data L (R04) ← Comparative Key Data L (R09)
   216 00C7 FFE0                                MOV     A,R00
   217 00C8 F4F3                                INC     A                                       ;Chattering Counter (R00) increment
   218 00C9 E5E0                                MOV     R00,A
   219 00CA FAF3                                SCAF                                            ;End of Chattering Routine of 3 Time?
   220 00CB EDF1 E1E7                                                   JNC    ON_CHAT         ;       else ON_CHAT:   (= 00H)
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:00 08/29/96 PAGE 02-006


SOURCE = KEY80.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   221                                          ;** Key High Check **                           ;       then            (= 0E-0FH)
   222 00CD F4E1                                XRL    A,R11                                    ;Acc = 0FH
   223 00CE F4E4                                XRL    A,R14
   224 00CF FAF3                                SCAF                                            ;Final Key Data H (R11) = Key Data H (R14) ?
   225 00D0 EDF1 EDE7                           JNC    KDAT_SET                                 ;       else KDAT_SET:  (Unmacth)
   226                                                                ;** Key Low Check **      ;       then            (Match)
   227 00D2 F5E1                                                      XRL    A,R01              ;Acc = 0FH
   228 00D3 F5E4                                                      XRL    A,R04
   229 00D4 FAF3                                                      SCAF =                    ;Final Key Data L (R01) = Key Data L (R04) ?
   230 00D5 ECF1 EDED                                                 JC     K29CHECK           ;       then  K29CHECK: (Match)
   231                                  KDAT_SET:                                               ;       else            (Unmacth)
   232 00D7 FEE4                                MOV    A,R14
   233 00D8 E4E1                                MOV    R11,A                                    ;Final Key Data H (R11) <-- Key Data H (R14)
   234 00D9 FFE4                                MOV    A,R04
   235 00DA E5E1                                MOV    R01,A                                    ;Final Key Data L (R01) <-- Key Data L (R04)
   236 00DB FFE8                                MOV    A,R08
   237 00DC E5E3                                MOV    R03,A                                    ;Continuance Flag = 00H (R08=00H)
   238
   239                                  K29CHECK:
   240 00DD FFEC                                MOV    A,R0C                                    ;R0C = 02H (K29=1DCH)
   241 00DE F4E1                                XRL    A,R11
   242 00DF FAF3                                SCAF                                            ;Final Key Data = K29 (R11=0BH)?
   243 00E0 EDF1 EEE7                           JNC    DBL_K_CHK                                ;       then DBL_K_CHK: (Key = Except for K29)
   244 00E2 FFE1                                                      MOV    A,R01              ;       else            (Key = K29)
   245 00E3 F5EB                                                      XRL    A,R0B              ;R0B = 03H
   246 00E4 FAF3                                                      SCAF                      ;Final Key Data = K29 (R01=0CH)?
   247 00E5 ECF1 EEEE                                                 JC     K29_FLG            ;       then  K29_FLG:  (Key = Except for K29)
   248                                                                                          ;       else            (Key = K29)
   249                                  DBL_K_CHK:
   250 00E7 FFF1 F6E0                           MOV    A,#06H
   251 00E9 F4E1                                XRL    A,R11
   252 00EA FAF3                                SCAF                                            ;Final Key Data K29+ K30 or K31 or K32 (R11=09H) ?
   253 00EB ECF1 EEEF                           JC     DATA_SET                                 ;       then  DATA_SET: (Key = Except for Double Key)
   254 00ED FFE8                                                      MOV    A,R08
   255                                  K29_FLG:
   256 00EE E4E3                                                      MOV    R13,A              ;K29 ON Flag = 00H (R08=00H)
   257                                  ;***********************************************
   258                                  ;               Transmit Data Set
   259                                  ;***********************************************
   260                                  DATA_SET:
   261                                      ;** Custom Code **
   262 00EF E6E4 E5E0                           MOV    R4,#CUSTM1                               ;Custom Code(R4) <-- Custom Code
   263
   264                                      ;** Custom Code’ **
   265 00F1 E6E5 EAEF                           MOV    R5,#CUSTM2                               ;Custom Code’(R5) <-- Custom Code’
   266
   267                                      ;** Data Code **
   268 00F3 FEE1                                MOV    A,R11
   269 00F4 E4E0                                MOV    R10,A                                    ;Data Pointer H(R10) <-- Final Key Data H(R11)
   270 00F5 FFE1                                MOV    A,R01
   271 00F6 E5E0                                MOV    R00,A                                    ;Data Pointer L(R00) <-- Final Key Data L(R01)
   272 00F7 E7E6                                MOV    R6,@R0
   273
   274                                  END
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-002


SOURCE = TRANS80.ASM

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
     1                                  ;*************************************************
     2                                  ;*****                                       *****
     3                                  ;*****      uPD6133 Series Transmit          *****
     4                                  ;*****   Trans. CODE: NEC-R Format (80Key)   *****
     5                                  ;*****                                       *****
     6                                  ;*************************************************
     7      01FF                                TIME9M          EQU    1FFH                     ;9.00ms (9.002ms)
     8
     9                                  ;##########   P   U   B   L   I   C   ##########
    10                                          ;
    11
    12                                  ;##########   E   X   T   E   R   N   ##########
    13                                          EXTRN           MAIN                            ;MAIN Routine
    14                                          EXTRN           KEY80                           ;Key Check Routine
    15                                          EXTRN           FS_27MS                         ;Frame Space Subroutine
    16                                          ;
    17
    18                                  ;##########   S    T    A    R    T   ##########
    19                                  TRNS:
    20 00F8 E6E7 EFE5                           MOV      R7,#0F5H                               ;R17 : Set Immediate Data (=0FH)
    21                                                                                          ;Key OFF Check counter (R07) = 10 Times (=05H)
    22 00FA E6E9 EDEF                           MOV      R9,#00FH                               ;Data1 Trans counter (R9) = DFH
    23                                  ;*************************************************
    24                                  ;                Leader Code
    25                                  ;*************************************************
    26 00FC E6E0 EAE0                           MOV      R0,#0A0H                               ;Data Pointer H (R10) = 0AH
    27 00FE E6E8 E0E8                           MOV      R8,#008H                               ;Set Table Address L
    28 0100 FEE8                                MOV      A,R18                                  ;Output Data (Work) <-- 00H (R18)
    29 0101 E6F2 E8F1 E6F1                      CALL     BITOUTOF
    30
    31                                  ;*************************************************
    32                                  ;                Custom Code
    33                                  ;*************************************************
    34 0104 E6E8 ECEA                           MOV      R8,#0CAH                               ;Set Table Address L
    35 0106 FEE4                                MOV      A,R14                                  ;Output Data (Work) <-- Custom Code H (R14)
    36 0107 E6F2 E8F1 E6F6                      CALL     BITOUTOC
    37
    38 010A FFE4                                MOV      A,R04                                  ;Output Data (Work) <-- Custom Code H (R04)
    39 0108 E6F2 E8F1 E6F6                      CALL     BITOUTOC
    40
    41                                  ;*************************************************
    42                                  ;                Custom Code’
    43                                  ;*************************************************
    44 010E FEE5                                MOV      A,R15                                  ;Output Data (Work) <-- Custom Code’ H (R15)
    45 010F E6F2 E8F1 E6F6                      CALL     BITOUTOC
    46
    47 0112 FFE5                                MOV      A,R05                                  ;Output Data (Work) <-- Custom Code’ L (R05)
    48 0113 E6F2 E8F1 E6F6                      CALL     BITOUTOC
    49
    50                                  ;*************************************************
    51                                  ;                Custom Code
    52                                  ;*************************************************
    53 0116 FEE6                                MOV      A,R16                                  ;utput Data (Work) <-- Data Code H (R16)
    54 0117 E6F2 E8F1 E6F6                      CALL     BITOUTOC
    55
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-003


SOURCE = TRANS80.ASM

E STNO LOC. 0BJ.                M  I   SOURCE STATEMENT
   56 011A FFE6                                MOV     A,R06                                   ;Output Data (Work) <-- Data Code L (R06)
   57 011B E6F2 E8F1 E6F6                      CALL    BITOUTOC
   58
   59                                  ;***********************************************
   60                                  ;               Data Coce'
   61                                  ;***********************************************
   62 011E FEE6                                MOV     A,R16                                   ;Output Data (Work) <-- Data Code H (R16)
   63 011F F4E7                                XRL     A,R17                                   ;R17 = 0FH
   64 0120 E6F2 E8F1 E6F6                      CALL    BITOUTOC
   65
   66 0123 FFE6                                MOV     A,R06                                   ;Output Data (Work) <-- Data Code L (R06)
   67 0124 F4E7                                XRL     A,R17                                   ;R17 = 0FH
   68 0125 E6F2 E8F1 E6F6                      CALL    BITOUTOC
   69
   70                                  ;***********************************************
   71                                  ;               Stop Bit
   72                                  ;***********************************************
   73 0128 E6E6 E0EE                           MOV     R8,#00EH                                ;Set Table Address L
   74 012A FEE8                                MOV     A,R18                                   ;Output Data (Work) <-- 00H (R18)
   75 012B E6F2 E8F1 E6F1                      CALL    BITOUTOF
   76
   77                                  ;***********************************************
   78                                  ;               Frame Space
   79                                  ;***********************************************
   80 012E FEE9                                MOV     A,R19
   81 012F FAF3                                SCAF                                            ;Frame Space > 9.00ms ?
   82 0130 ECF1 E4F0                           JC      T9_UNDER                                ;       then  T9_UNDER: (More than 9.00ms)
   83                                                                                          ;       else            (Less than 9.00ms)
   84 0132 E2F1 E0E5                                                   HALT   #005H            ;HALT mode (Timer = 00H)
   85 0134 E6FF F7FF                                                   MOV    T,#TIME9M
   86 0136 E3F1 E0E5                                                   STTS   #0101B           ;Clear Status Flag
   87
   88 0138 FFE9                                                        MOV    A,R09
   89 0139 FCF3                                                        RL     A                ;Frame Space > 18.00ms ?
   90 013A ECF1 E4F0                           JC      T9_UNDER                                ;       then T9_UNDER:  (Less than 18.00ms)
   91                                                                                          ;       else            (More than 18.00ms)
   92 013C E2F1 E0E5                                                   HALT   #005H            ;HALT mode (Timer = 00H)
   93 013E E6FF F7FF                                                   MOV    T,#TIME9M
   94
   95                                  T9_UNDER:
   96 0140 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
   97 0142 FFE9                                MOV     A,R09
   98 0143 FBF1 F7E0                           ANL     A,#0111B
   99 0145 E5E0                                MOV     R00,A
  100 0146 E2F1 E0E5                           HALT    #005H                                   ;HALT mode (Timer = 00H)
  101 0148 E7FF                                MOV     T,@R0
  102 0149 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
  103
  104 014B FFE3                                MOV    A,R03
  105 014C FAF3                                SCAF                                            ;Continuously pressed key ?
  106 014D EDF1 E5FA                           JNC     KEY_ON                                  ;       else  KEY_ON:   (1st. Frame)
  107 014F FFE7                                                        MOV    A,R07            ;       then            (Since 2nd. Frame)
  108 0150 FAF3                                                        SCAF                    ;Without input Key ?
  109 0151 EDF1 E1E3                                                   JNC    KEY80            ;       then  KEY80:    (With input Key)
  110                                                         ;** Frame Space = 27ms **        ;       else           (Without input Key)
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-004


SOURCE = TRANS80.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
   111 0153 EXTRN                                                       CALL   FS_27MS
   112 0156 E2F1 E0E5                                                   HALT   #005H            ;HALT mode (Timer = 00H)
   113 0158 E8F1 E0E0                                                   JMP    MAIN
   114                                                                  ;
   115                                  KEY_ON:
   116 015A FEE7                                MOV     A,R17
   117 015B E5E3                                MOV     R03,A                                   ;Continuance Flag (R03) = 0FH (R17=0FH)
   118                                    ;** Frame Space  = 27ms **
   119 015C EXTRN                               CALL    FS_27MS
   120 015F E8F1 EFE8                           JMP     TRNS
   121                                          ;
   122
   123                                  ;*************************************************
   124                                  ;*****                                       *****
   125                                  ;*****               Transmit                *****
   126                                  ;*****         Subroutine : Bit Out          *****
   127                                  ;*****                                       *****
   128                                  ;*************************************************
   129                                  BITOUTOF:
   130 0161 E4EA                                MOV     R1A,A                                   ;Set Output Data (1 Bit Output)
   131 0162 FFF1 FFE0                           MOV     A,#0FH                                  ;Send Bit Counter = 1 Times (=0FH)
   132 0164 E8F1 E6F9                                                   JMP    BITOUT
   133                                                                  ;
   134                                  BITOUTOC:
   135 0166 E4EA                                MOV     R1A,A                                   ;Set Output Data (4 Bit Output)
   136 0167 FFF1 FCE0                           MOV     A,#0CH                                  ;Send Bit Counter = 4 Times (=0CH)
   137
   138                                  BITOUT:
   139 0169 E5EA                                MOV     R0A,A                                   ;Send Bit Counter (R0A) = 0CH or 0EH or 0FH
   140
   141 016A FEEA                                MOV     A,R1A
   142 016B FCF3                                RL      A
   143 016C E4EA                                MOV     R1A,A                                   ;Bit Data = 1 ?
   144 016D ECF1 E7F3                           JC      BIT_DAT1                                ;       then  BIT_DAT1: (Bit Data = 1)
   145                                                                  ;** Bit Data = 0 **     ;       else            (Bit Data = 0)
   146 016F FFE8                                                        MOV    A,R08            ;Set Data Pointer L (R00) <-- Data0 (R08)
   147 0170 E5E0                                                        MOV    R00,A
   148 0171 E8F1 E7FD                                                   JMP    BIT_TRNS
   149                                                                  ;
   150                                      ;** Bit Data = 1 **;
   151                                  BIT_DAT1:
   152 0173 FEE8                                MOV     A,R18                                   ;Set Data Pointer L(R00) <-- Data1 (R18)
   153 0174 E5E0                                MOV     R00,A
   154 0175 FFE9                                MOV     A,R09
   155 0176 F4F3                                INC     A                                       ;Data 1 Transmit Counter L (R09) increment
   156 0177 E5E9                                MOV     R09,A                                   ;Data 1 Transmit Counter L (R09) = Overflow ?
   157 0178 EDF1 E7FD                           JNC     BIT_TRNS                                ;       else BIT_TRNS:  (Data 1 Transmit Counter L <= 0FH)
   158 017A FEE9                                                        MOV    A,R19            ;       then            (Data 1 Transmit Counter L > 0FH)
   159 017B F4F3                                                        INC    A                ;Data 1 Transmit counter H(R19) increment
   160 017C E4E9                                                        MOV    R19,A
   161                                  BIT_TRNS:
   162 017D E2F1 E0E5                           HALT    #005H                                   ;HALT mode (Timer = 00H)
   163 017F E7FF                                MOV     T,@R0
   164 0180 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Flag
   165 0182 FFE0                                MOV     A,R00
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 03-005


SOURCE = TRANS80.ASM

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
   166 0183 F4F3                                INC     A                                       ;Data Pointer L (R00) increment
   167 0184 E5E0                                MOV     R00,A
   168 0185 E2F1 E0E5                           HALT    #005H                                   ;HALT mode (Timer = 00H)
   169 0187 E7FF                                MOV     T,@R0
   170
   171 0188 FFE7                                MOV     A,R07
   172 0189 F4F3                                INC     A                                       ;Key Off Check Counter (R07) increment
   173 018A ECF1 E9F5                                                   JC     BIT_TRANS2       ;Key Off Check Counter (R07) = End of 10 Times ?
   174                                                                                          ;       then  BIT_TRANS2: (End of 10 Times)
   175                                                                                          ;       else              (Less than 10 Times)
   176 018C E3F1 E0EE                           STTS    #1110B                                  ;With input key ?
   177 018E EFF1 E9F4                           JNF     BIT_TRANS1                              ;       else  BIT_TRANS1: (With input)
   178                                                                  ;** With input key **   ;       then              (Without input)
   179 0190 E6E7 EFE5                                                   MOV    R7, #0F5H        ;Key off Check Counter (R07) = 05H
   180 0192 E8F1 E9F5                                                   JMP    BIT_TRANS2
   181                                                                  ;
   182                                      ;** Without input Key **
   183                                  BIT_TRANS1:
   184 0194 E5E7                                MOV     R07,A
   185
   186                                  BIT_TRANS2:
   187 0195 E3F1 E0E5                           STTS    #0101B                                  ;Clear Status Falg
   188 0197 FFEA                                MOV     A,R0A
   189 0198 F4F3                                INC     A                                       ;Send Bit Counter (R0A) increment
   190 0199 EDF1 E6F9                           JNC     BITOUT
   191 019B E8F2                                RET
   192                                          ;
   193                                  END


TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 04-002


SOURCE = DTABLE80.TBL

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
     1                                  ;*************************************************
     2                                  ;*****                                      *****
     3                                  ;*****      uPD6133 Series Data Table        *****
     4                                  ;*****   Trans. CODE: NEC-R Format (80Key)   *****
     5                                  ;*****                                      *****
     6                                  ;*************************************************
     7     01FF                                 TIME9M          EQU    1FFH                     ;9.00ms (9.002ms)
     8
     9                                  ;##########   P   U   B   L   I   C   ##########
    10                                          PUBLIC          DATACODE
    11                                          PUBLIC          FS_27MS
    12                                          ;
    13
    14                                  DATACODE:
    15                                  ;**************************************
    16                                  ;*****         Data Code          *****
    17                                  ;*****         Double Key         *****
    18                                  ;**************************************
    19 019D                             ORG     19DH
    20                                    ;** K29 + K30 **
    21 019D E0E9                                DW      009H            ;K29 + K30 : Key Data = 70H
    22                                    ;** K29 + K31 **
    23 019E E8E9                                DW      089H            ;K29 + K31 : Key Data = 71H
    24                                    ;** K29 + K32 **
    25 019F E4E9                                DW      049H            ;K29 + K32 : Key Data = 72H
    26                                          ;
    27
    28                                  ;**************************************
    29                                  ;*****        Frame Space        *****
    30                                  ;*****         Time Data         *****
    31                                  ;ORG 1A0H *****************************
    32 01A0 E7ED                                DT      1F4H            ;Counter:000 = 8.815ms (8.809ms)
    33 01A1 E6ED                                DT      1B4H            ;Counter:001 = 7.690ms (7.683ms)
    34 01A2 E5ED                                DT      174H            ;Counter:010 = 6.565ms (6.558ms)
    35 01A3 E4ED                                DT      134H            ;Counter:011 = 5.440ms (5.433ms)
    36 01A4 E3ED                                DT      0F4H            ;Counter:100 = 4.315ms (4.308ms)
    37 01A5 E2ED                                DT      0B4H            ;Counter:101 = 3.190ms (3.182ms)
    38 01A6 E1ED                                DT      074H            ;Counter:110 = 2.065ms (2.057ms)
    39 01A7 E0ED                                DT      034H            ;Counter:111 = 0.940ms (0.932ms)
    40                                          ;
    41
    42                                  ;**************************************
    43                                  ;*****         Time Data          *****
    44                                  ;ORG 1A8H *****************************
    45                                    ;** Leader Code **
    46 01A8 FFFF                                DT      3FFH            ;Carrier ON  : 9.000ms (9.002ms)
    47 01A9 F3FF                                DT      0FFH            ;Carrier OFF : 4.500ms (4.501ms)
    48                                    ;** Bit data 0 **
    49 01AA F8F7                                DT      21FH            ;Carrier ON  : 0.560ms (0.563ms)
    50 01AB F0F7                                DT      01FH            ;Carrier OFF : 0.565ms (0.563ms)
    51                                    ;** Bit data 1 **
    52 01AC F8F7                                DT      21FH            ;Carrier ON  : 0.560ms (0.563ms)
    53 01AD F1F7                                DT      05FH            ;Carrier OFF : 1.690ms (1.687ms)
    54                                    ;** Stop Bit **
    55 01AE F8F7                                DT      21FH            ;Carrier ON  : 0.560ms (0.563ms)
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>      01:00:01 08/29/96 PAGE 04-003


SOURCE = DTABLE80.TBL

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
    56 01AF E2FA                                DT      0A9H            ;Carrier OFF : 3.00ms (2.989ms)
    57                                          ;
    58
    59                                  ;**************************************
    60                                  ;*****         Data Code          *****
    61                                  ;*****         Single Key         *****
    62                                  ;**************************************
    63 01B0                             ORG     1B0H
    64                                    ;** K33 - K40 **
    65 01B0 E0E4                                DW      004H            ;K33 : Data Code = 20H
    66 01B1 E8E4                                DW      084H            ;K34 : Data Code = 21H
    67 01B2 E4E4                                DW      044H            ;K35 : Data Code = 22H
    68 01B3 ECE4                                DW      0C4H            ;K36 : Data Code = 23H
    69 01B4 E2E4                                DW      024H            ;K37 : Data Code = 24H
    70 01B5 EAE4                                DW      0A4H            ;K38 : Data Code = 25H
    71 01B6 E6E4                                DW      064H            ;K39 : Data Code = 26H
    72 01B7 EEE4                                DW      0E4H            ;K40 : Data Code = 27H
    73                                          ;
    74                                    ;** K73 + K80 **
    75 01B8 E1E2                                DW      012H            ;K73 : Data Code = 48H
    76 01B9 E9E2                                DW      092H            ;K74 : Data Code = 49H
    77 01BA E5E2                                DW      052H            ;K75 : Data Code = 4AH
    78 01BB EDE2                                DW      0D2H            ;K76 : Data Code = 4BH
    79 01BC E3E2                                DW      032H            ;K77 : Data Code = 4CH
    80 01BD EBE2                                DW      0B2H            ;K78 : Data Code = 4DH
    81 01BE E7E2                                DW      072H            ;K79 : Data Code = 4EH
    82 01BF EFE2                                DW      0F2H            ;K80 : Data Code = 4FH
    83                                          ;
    84
    85 01C0                             ORG     1C0H
    86                                    ;** K1 - K16 **
    87 01C0 E0E0                                DW      000H            ;K 1 : Data Code = 00H
    88 01C1 E8E0                                DW      080H            ;K 2 : Data Code = 01H
    89 01C2 E4E0                                DW      040H            ;K 3 : Data Code = 02H
    90 01C3 ECE0                                DW      0C0H            ;K 4 : Data Code = 03H
    91 01C4 E2E0                                DW      020H            ;K 5 : Data Code = 04H
    92 01C5 EAE0                                DW      0A0H            ;K 6 : Data Code = 05H
    93 01C6 E6E0                                DW      060H            ;K 7 : Data Code = 06H
    94 01C7 EEE0                                DW      0E0H            ;K 8 : Data Code = 07H
    95 01C8 E1E0                                DW      010H            ;K 9 : Data Code = 08H
    96 01C9 E9E0                                DW      090H            ;K10 : Data Code = 09H
    97 01CA E5E0                                DW      050H            ;K11 : Data Code = 0AH
    98 01CB EDE0                                DW      0D0H            ;K12 : Data Code = 0BH
    99 01CC E3E0                                DW      030H            ;K13 : Data Code = 0CH
   100 01CD EBE0                                DW      0B0H            ;K14 : Data Code = 0DH
   101 01CE E7E0                                DW      070H            ;K15 : Data Code = 0EH
   102 01CF EFE0                                DW      0F0H            ;K16 : Data Code = 0FH
   103                                          ;
   104                                    ;** K17 - K32 **
   105 01D0 E0E8                                DW      008H            ;K17 : Data Code = 10H
   106 01D1 E8E8                                DW      088H            ;K18 : Data Code = 11H
   107 01D2 E4E8                                DW      048H            ;K19 : Data Code = 12H
   108 01D3 ECE8                                DW      0C8H            ;K20 : Data Code = 13H
   109 01D4 E2E8                                DW      028H            ;K21 : Data Code = 14H
   110 01D5 EAE8                                DW      0A8H            ;K22 : Data Code = 15H
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 04-004


SOURCE = DTABLE80.TBL

E STNO LOC. OBJ.                M  I   SOURCE STATEMENT
   111 01D6 E6E8                               DW      068H            ;K23 : Data Code = 16H
   112 01D7 EEE8                               DW      0E8H            ;K24 : Data Code = 17H
   113 01D8 E1E8                               DW      018H            ;K25 : Data Code = 18H
   114 01D9 E9E8                               DW      098H            ;K26 : Data Code = 19H
   115 01DA E5E8                               DW      058H            ;K27 : Data Code = 1AH
   116 01DB EDE8                               DW      0D8H            ;K28 : Data Code = 1BH
   117 01DC E3E8                               DW      038H            ;K29 : Data Code = 1CH
   118 01DD EBE8                               DW      0B8H            ;K30 : Data Code = 1DH
   119 01DE E7E8                               DW      078H            ;K31 : Data Code = 1EH
   120 01DF EFE8                               DW      0F8H            ;K32 : Data Code = 1FH
   121                                         ;
   122                                   ;** K41 - K56 **
   123 01E0 E1E4                               DW      014H            ;K41 : Data Code = 28H
   124 01E1 E9E4                               DW      094H            ;K42 : Data Code = 29H
   125 01E2 E5E4                               DW      054H            ;K43 : Data Code = 2AH
   126 01E3 EDE4                               DW      0D4H            ;K44 : Data Code = 2BH
   127 01E4 E3E4                               DW      034H            ;K45 : Data Code = 2CH
   128 01E5 EBE4                               DW      0B4H            ;K46 : Data Code = 2DH
   129 01E6 E7E4                               DW      074H            ;K47 : Data Code = 2EH
   130 01E7 EFE4                               DW      0F4H            ;K48 : Data Code = 2FH
   131 01E8 E0EC                               DW      00CH            ;K49 : Data Code = 30H
   132 01E9 E8EC                               DW      08CH            ;K50 : Data Code = 31H
   133 01EA E4EC                               DW      04CH            ;K51 : Data Code = 32H
   134 01EB ECEC                               DW      0CCH            ;K52 : Data Code = 33H
   135 01EC E2EC                               DW      02CH            ;K53 : Data Code = 34H
   136 01ED EAEC                               DW      0ACH            ;K54 : Data Code = 35H
   137 01EE E6EC                               DW      06CH            ;K55 : Data Code = 36H
   138 01EF EEEC                               DW      0ECH            ;K56 : Data Code = 37H
   139                                         ;
   140                                   ;** K57 - K72 **
   141 01F0 E1EC                               DW      01CH            ;K57 : Data Code = 38H
   142 01F1 E9EC                               DW      09CH            ;K58 : Data Code = 39H
   143 01F2 E5EC                               DW      05CH            ;K59 : Data Code = 3AH
   144 01F3 EDEC                               DW      0DCH            ;K60 : Data Code = 3BH
   145 01F4 E3EC                               DW      03CH            ;K61 : Data Code = 3CH
   146 01F5 EBEC                               DW      0BCH            ;K62 : Data Code = 3DH
   147 01F6 E7EC                               DW      07CH            ;K63 : Data Code = 3EH
   148 01F7 EFEC                               DW      0FCH            ;K64 : Data Code = 3FH
   149 01F8 E0E2                               DW      002H            ;K65 : Data Code = 40H
   150 01F9 E8E2                               DW      082H            ;K66 : Data Code = 41H
   151 01FA E4E2                               DW      042H            ;K67 : Data Code = 42H
   152 01FB ECE2                               DW      0C2H            ;K68 : Data Code = 43H
   153 01FC E2E2                               DW      022H            ;K69 : Data Code = 44H
   154 01FD EAE2                               DW      0A2H            ;K70 : Data Code = 45H
   155 01FE E6E2                               DW      062H            ;K71 : Data Code = 46H
   156 01FF EEE2                               DW      0E2H            ;K72 : Data Code = 47H
   157                                         ;
   158
   159
   160
   161                                 ;**************************************************
   162                                 ;*****                                        *****
   163                                 ;*****         Frame Space = 27ms             *****
   164                                 ;*****        Subroutine : FS 27ms            *****
   165                                 ;*****                                        *****
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 04-005


SOURCE = DTABLE80.TBL

E STNO LOC. 0BJ.                 M  I   SOURCE STATEMENT
   166                                  ;*************************************************
   167                                  FS_27MS:
   168 0200 FFF1 FDE0                           MOV     A,#00H                                  ;Counter(Acc) = 3 Times (=0DH)
   169                                  FS_27MS0:
   170 0202 E2F1 E0E5                           HALT    #005H                                   ;HALT mode (Timer = 00H)
   171 0204 E6FF F7FF                           MOV     T,#TIME9M                               ;Timer = 9.00ms
   172 0206 E3F1 E0E5                           STTS    #0101B                                  ;Status Flag Clear
   173 0208 F4F3                                INC     A                                       ;More than 27ms ?
   174 0209 EDF1 F0E2                           JNC     FS_27MS0                                ;       else  FS_27MS0: (Frame Space < 27ms)
   175 020B E8F2                                RET                                             ;       then  RET       (Frame Space = 27ms)
   176                                          ;
   177                                   END


TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

```
AS6133 V1.01     << D6134 ASSEMBLE LIST >>     01:00:01 08/29/96 PAGE 05-002


SOURCE = OPTION.ASM

E STNO LOC. OBJ.                 M  I   SOURCE STATEMENT
     1                                  ;**************************************
     2                                  ;*****         Set Option         *****
     3                                  ;**************************************
     4                                  OPTION
     5 0000 0001                                USEPOC
     6                                  ENDOP
     7                                  END



TOTAL ERRORS   = 0
TOTAL WARNINGS = 0

END OF LIST
```

Facsimile Message

From:

Name

Company

Tel. FAX

Address

Although NEC has taken all possible steps to ensure that the documentation supplied to our customers is complete, bug free and up-to-date, we readily accept that errors may occur. Despite all the care and precautions we've taken, you may encounter problems in the documentation. Please complete this form whenever you'd like to report errors or suggest improvements to us.

Thank you for your kind support.

North America
NEC Electronics Inc.
Corporate Communications Dept.
Fax: 1-800-729-9288
1-408-588-6130

Europe
NEC Electronics (Europe) GmbH
Technical Documentation Dept.
Fax: +49-211-6503-274

South America
NEC do Brasil S.A.
Fax: +55-11-6465-6829

Hong Kong, Philippines, Oceania
NEC Electronics Hong Kong Ltd.
Fax: +852-2886-9022/9044

Korea
NEC Electronics Hong Kong Ltd.
Seoul Branch
Fax: 02-528-4411

Taiwan
NEC Electronics Taiwan Ltd.
Fax: 02-719-5951

Asian Nations except Philippines
NEC Electronics Singapore Pte. Ltd.
Fax: +65-250-3583

Japan
NEC Corporation
Semiconductor Solution Engineering Division
Technical Information Support Dept.
Fax: 044-548-7900

I would like to report the following error/make the following suggestion:

Document title: ____________________

Document number: ____________________ Page number: __________

If possible, please fax the referenced page or drawing.

Document Rating	Excellent	Good	Acceptable	Poor
Clarity	□	□	□	□
Technical Accuracy	□	□	□	□
Organization	□	□	□	□