To our customers,

Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)
Send any inquiries to http://www.renesas.com/inquiry.
Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.

2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.

4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.

5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.

6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.

7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

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

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

   “Specific”:
   Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.

8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.

9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.

10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.

12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.
To all our customers

Regarding the change of names mentioned in the document, such as Hitachi Electric and Hitachi XX, to Renesas Technology Corp.

The semiconductor operations of Mitsubishi Electric and Hitachi were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Renesas Technology Home Page: http://www.renesas.com

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003
Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

---

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.

2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.

3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.

The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.

Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).

4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.

5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.

6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.

7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination. Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.

8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.
H8/3867 Series
Application Note
Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi’s or any third party’s patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party’s rights, including intellectual property rights, in connection with use of the information contained in this document.

2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.

3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi’s sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.

4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.

5. This product is not designed to be radiation resistant.

6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.

7. Contact Hitachi’s sales office for any questions regarding this document or Hitachi semiconductor products.
Preface

The H8/300L Series of single-chip microcomputers are based on the high-speed H8/300L CPU, and integrate all peripheral functions necessary for system configuration.

The H8/300L CPU uses an instruction set which is compatible with the H8/300 CPU.

The H8/3867 Series and H8/3827 Series are provided with such peripheral functions for system configuration as an LCD controller/driver, six different timers, a 14-bit pulse width modulator (PWM), a two-channel serial communication interface, and an A/D converter. These models can be used as microcomputers for embedded systems where LCD display is required.

The H8/3867 Series models are equipped with a booster constant-voltage (5 V) power supply as an LCD driver power supply, providing a constant 5 V regardless of \( V_{cc} \).

These H8/3867 Series application notes include a “Basic Operation” section with operation examples when using the built-in peripheral functions of the H8/3867 Series independently. They are provided in the hope that they will be of use for software and hardware design.

*Operation of the programs and circuits described in these application notes has been verified, but their operation should be confirmed by the user as well before actually being used.*
Contents

Section 1   Guide to Using the H8/3867 Series Application Notes ......................... 1
   1.1   Contents of Basic Operation.............................................................................. 2

Section 2   Basic Operation ....................................................................................... 5
   2.1   Internal Power Supply Step-Down Circuit Settings ........................................... 5
   2.2   Asynchronous Event Counter Operation............................................................... 10
   2.3   LCD Display with Static Duty .............................................................................. 35
   2.4   LCD Display with 1/4 Duty .................................................................................. 58
   2.5   LCD Display with Segment External Expansion .................................................. 83
   2.6   Oscillation Stabilization Time Settings ............................................................... 108
   2.7   Module Standby Mode Settings .......................................................................... 117
   2.8   Clock Operation Using Timer F .......................................................................... 124
Section 1  Guide to Using the H8/3867 Series Application Notes

These application notes consist of two sections, as follows.

![Diagram of Application Notes](image)

**Figure 1  Contents of these Application Notes**

**Guide to Using the H8/3867 Series Application Notes**

Explains how to use the H8/3867 Series application notes.

**Basic Operation**

Explains how to use the built-in peripheral functions of the H8/3867 Series through simple task examples.
1.1 Contents of Basic Operation

Basic Operation includes the sections shown below, explaining use of the built-in peripheral functions.

![Diagram of Basic Operation Contents]

**Specifications**
Explains system specifications for task examples.

**Explanation of Functions Used**
Explains the features of peripheral functions used in task examples, and allocation of the peripheral functions.

**Explanation of Operation**
Explains operation of task examples using timing charts.
Explanation of Software

1. Explanation of Modules
   Explains the software modules used for operation in task examples.

2. Explanation of Arguments
   Explains input arguments necessary for module execution, and arguments output following execution.

3. Explanation of Internal Registers Used
   Explains internal registers such as a timer control register and serial mode register of peripheral functions used in modules.

4. Explanation of RAM Usage
   Explains RAM label names and functions used in modules.

Flowcharts

Uses flowcharts to explain the software executed in task examples.

Program Lists

Gives program lists for software executed in task examples.
Section 2  Basic Operation

2.1  Internal Power Supply Step-Down Circuit Settings

|-------------------------------------------------|---------------------|------------------------------------------------------|

Usage

The H8/3867 Series incorporates an internal power supply step-down circuit. Below the features and usage of the internal power supply step-down circuit are explained, together with important notes and the power supply voltage and operating range.

Features of the internal power supply step-down circuit

1. By using the internal power supply step-down circuit, the internal power supply voltage can be held constant at approximately 1.5 V without depending on the voltage of the power supply connected to the external \( V_{cc} \) pin.

2. Current consumed when an external power supply at 1.8 V or higher is used can be held to approximately the same low current as at 1.5 V.

3. It is also possible to use the same level of an external power supply voltage and internal power supply voltage, without using the internal power supply step-down circuit.
Power supply connection when using the internal power supply step-down circuit

An external power supply is connected to the V_{cc} pin as shown in figure 1, and a capacitance of approximately 1 \mu F is inserted between CV_{cc} and V_{ss}. By adding this external circuit, the internal step-down circuit becomes operative.

![Figure 1  Power Supply Connection When Using the Internal Power Supply Step-down Circuit](Figure)

Notes on operation using the internal power supply step-down circuit

1. The interface to the external circuit uses as reference levels the voltage of the power supply connected to the V_{cc} pin and the level of the ground connected to the V_{ss} pin. For example, the high and low port input/output levels become the V_{cc} level and the V_{ss} level, respectively.

2. When the internal power supply step-down circuit is used, the operating frequency f_{osc} range is, for a V_{cc} of 2.2 to 5.5 V, f_{osc} = 0.4 MHz to 2 MHz; otherwise, it is f_{osc} = 0.4 MHz to 1 MHz.

3. The LCD power supply and A/D converter analog power supply are not affected by internal step-down processing.
Power supply connection when not using the internal power supply step-down circuit

The external power supply is connected across the \( V_{cc} \) and \( CV_{cc} \) pins, as shown in figure 2. The external power supply is input directly to the internal power supply circuit.

![Diagram](image)

**Figure 2  Power Supply Connection When Not Using the Internal Power Supply Step-down Circuit**

Note on operation not using the internal power supply step-down circuit

Power supply voltages between 1.8 V and 5.5 V can be used. Operation cannot be guaranteed if a voltage outside this range (less than 1.8 V or more than 5.5 V) is input.
Power supply voltage and oscillator frequency ranges

Figure 3 shows the ranges of the power supply voltage and the oscillator frequency (shaded regions).

(a) Internal power supply step-down circuit not used
Active (high-speed) mode
Sleep (high-speed) mode

(b) Internal power supply step-down circuit used
Active (high-speed) mode
Sleep (high-speed) mode

Figure 3  Power Supply Voltage and Oscillator Frequency Ranges
Power supply voltage and operating frequency ranges

Figure 4 shows the ranges of the power supply voltage and operating frequency (shaded regions).

(a) Internal power supply step-down circuit
not used
Active (high-speed) mode
Sleep (high-speed) mode (except CPU)

(b) Internal power supply step-down circuit
used
Active (high-speed) mode
Sleep (high-speed) mode (except CPU)

(c) Internal power supply step-down circuit
not used
Active (medium-speed) mode
(except A/D converter and PWM)
Sleep (medium-speed) mode
(except A/D converter and PWM)

(d) Internal power supply step-down circuit
used
Active (medium-speed) mode
(except A/D converter and PWM)
Sleep (medium-speed) mode
(except A/D converter and PWM)

Figure 4  Power Supply Voltage and Operating Frequency Ranges
2.2 Asynchronous Event Counter Operation

<table>
<thead>
<tr>
<th>Asynchronous Event Counter Operation</th>
<th>MCU: H8/3867 Series</th>
<th>Functions Used: Asynchronous Event Counter (AEC)</th>
</tr>
</thead>
</table>

Specifications

1. Using an asynchronous event counter, once every 524.288 ms there is a transition from subactive mode to active (high-speed) mode, reversal of the port output in active (high-speed) mode, and a transition back to subactive mode.

2. The 2-MHz event input is applied to the asynchronous event input L pin (AEVL).

3. In this task example, the circuit is used as a 16-bit asynchronous event counter.
Explanation of Functions Used

1. In this task example, an asynchronous event counter (AEC) is used to induce transitions between subactive and active modes and to invert the port output. The features of the AEC are as follows.

   - Input external events can be counted asynchronously, independently of basic clock operation.
   - The counter has a 16-bit configuration, and can count up to 65,536 events.
   - The circuit can also be used as two independent 8-bit event counter channels.
   - The counter can be reset or halted under software control.
   - Event counter overflow can be detected to automatically generate an interrupt.
   - A module standby mode can be employed to set standby mode in module units when not in use.

2. Figure 1 is a block diagram of the 16-bit asynchronous event counter used in this task example.

![Figure 1 Block Diagram of Asynchronous Event Counter](image)

- ECCSR: Event counter control/status register
- ECH: Event counter H
- ECL: Event counter L
- AEVH: Asynchronous event input H
- AEVL: Asynchronous event input L
- IRREC: Event counter overflow interrupt request flag

Figure 1  Block Diagram of Asynchronous Event Counter
3. Functions of the 16-bit asynchronous event counter are explained in table 1 below.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Functions of the 16-bit Asynchronous Event Counter</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Event counter control/status register (ECCSR)</strong></td>
<td><strong>Function</strong></td>
</tr>
<tr>
<td><strong>Event counter H (ECH)</strong></td>
<td><strong>Function</strong></td>
</tr>
<tr>
<td><strong>Event counter L (ECL)</strong></td>
<td><strong>Function</strong></td>
</tr>
<tr>
<td><strong>Asynchronous event input H (AEVH)</strong></td>
<td><strong>Function</strong></td>
</tr>
<tr>
<td><strong>Asynchronous event input L (AEVL)</strong></td>
<td><strong>Function</strong></td>
</tr>
<tr>
<td><strong>Asynchronous event counter interrupt request flag (IRREC)</strong></td>
<td><strong>Function</strong></td>
</tr>
<tr>
<td><strong>Asynchronous event counter interrupt enable (IENEC)</strong></td>
<td><strong>Function</strong></td>
</tr>
</tbody>
</table>
4. Figure 2 shows an example of settings when using the circuit as a 16-bit asynchronous event counter.

![Figure 2 Example of Settings for 16-bit Asynchronous Event Counter](image)

Upon reset, CH2 is cleared to 0, so that after reset ECH and ECL operate as a 16-bit event counter. In addition, the circuit will also operate as a 16-bit event counter by using the settings shown in figure 2. The operating clock source is the asynchronous event input from the AEVL pin. When the next clock pulse is input after the count values for both ECH and ECL reach H'FF, ECH and ECL overflow, the OVH flag of ESSR is set to 1, the count values of ECH and ECL are both returned to H'00, and counting-up is restarted. Upon occurrence of overflow, the IRREC bit of IRR2 is set to 1. At this time, if the IENEC bit of IENR2 is 1, an interrupt request is sent to the CPU.
5. Asynchronous event counter operating modes are indicated in table 2.

Table 2  Asynchronous Event Counter Operating Modes

<table>
<thead>
<tr>
<th>Operating Mode</th>
<th>Reset</th>
<th>Active</th>
<th>Sleep</th>
<th>Watch</th>
<th>Subactive</th>
<th>Subsleep</th>
<th>Standby</th>
<th>Module Standby</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECCSR</td>
<td>Reset</td>
<td>Functions</td>
<td>Functions</td>
<td>Held*</td>
<td>Functions</td>
<td>Functions</td>
<td>Held*</td>
<td>Held</td>
</tr>
<tr>
<td>ECH</td>
<td>Reset</td>
<td>Functions</td>
<td>Functions*</td>
<td>Functions</td>
<td>Functions</td>
<td>Functions</td>
<td>Functions*</td>
<td>Halted</td>
</tr>
<tr>
<td>ECL</td>
<td>Reset</td>
<td>Functions</td>
<td>Functions*</td>
<td>Functions</td>
<td>Functions</td>
<td>Functions</td>
<td>Functions*</td>
<td>Halted</td>
</tr>
</tbody>
</table>

Note: * When an asynchronous external event is input, the counter is incremented, but the count overflow H/L flags are not affected.

6. Notes on the 16-bit asynchronous event counter

a. Before reading the values of ECH and ECL, the CUEH and CUEL bits of ECCSR are cleared to 0, to prevent asynchronous events from being input to the counter. If the counter is incremented during reading, the correct value cannot be read. When clearing the CUEH and CUEL bits of ECCSR to 0, ECH and ECL may each be incremented by one.

b. When the internal power supply step-down circuit is not being used, the maximum clock frequency for input to the AEVH and AEVL pins is 6 MHz when \( V_{cc} \) is 4.5 to 5.5 V, is 4 MHz when \( V_{cc} \) is 3.0 to 5.5 V, and is 3.2 MHz when \( V_{cc} \) is 2.6 to 5.5 V. When the internal power supply step-down circuit is being used or not being used, the maximum clock frequency is 2 MHz when \( V_{cc} \) is 2.2 to 5.5 V, and otherwise is 1 MHz. In addition, the clock high and low widths should be a minimum of 83 ns.

c. When the AEC is used in 16-bit mode, either the CUEH bit in ECCSR should be set to 1 and then CRCH set to 1, or else after CUEH and CRCH are set simultaneously the clock pulse should be input. Thereafter, the value of CUEH should not be modified during use in 16-bit mode. If, while in 16-bit mode, CUEH is changed, ECH may be erroneously incremented.

d. Table 3 shows operating modes and event input frequencies.
### Table 3  Relation between Operating Modes and AEVH/AEVL Pin Event Input Frequencies

<table>
<thead>
<tr>
<th>Mode</th>
<th>Maximum AEVH/AEVL Pin Input Clock Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-bit mode</td>
<td>Internal step-down circuit not used:</td>
</tr>
<tr>
<td></td>
<td>$V_{CC} = 4.5 \text{ to } 5.5 \text{ V/6 MHz}$</td>
</tr>
<tr>
<td></td>
<td>$V_{CC} = 3.0 \text{ to } 5.5 \text{ V/8 MHz}$</td>
</tr>
<tr>
<td></td>
<td>$V_{CC} = 2.6 \text{ to } 5.5 \text{ V/3.2 MHz}$</td>
</tr>
<tr>
<td></td>
<td>$V_{CC} = 2.2 \text{ to } 5.5 \text{ V/2 MHz}$</td>
</tr>
<tr>
<td></td>
<td>Other than above/1 MHz</td>
</tr>
<tr>
<td>8-bit mode</td>
<td>Internal step-down circuit used:</td>
</tr>
<tr>
<td></td>
<td>$V_{CC} = 2.2 \text{ to } 5.5 \text{ V/2 MHz}$</td>
</tr>
<tr>
<td></td>
<td>Other than above/1 MHz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8-bit mode</th>
<th>Active (high-speed), Sleep (high-speed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{osc} = 400 \text{ kHz to } 4 \text{ MHz}$</td>
<td>$2 \cdot f_{osc}$</td>
</tr>
<tr>
<td>$f_{osc}$</td>
<td>$f_{osc}$</td>
</tr>
<tr>
<td>$f_{osc}$</td>
<td>$f_{osc}$</td>
</tr>
<tr>
<td>$f_{osc}$</td>
<td>$f_{osc}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8-bit mode</th>
<th>Active (medium-speed), Sleep (medium-speed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{osc} = 32.768 \text{ kHz or } 38.4 \text{ kHz}$</td>
<td>$1000 \text{ kHz}$</td>
</tr>
<tr>
<td>$f_{osc}$</td>
<td>$500 \text{ kHz}$</td>
</tr>
<tr>
<td>$f_{osc}$</td>
<td>$250 \text{ kHz}$</td>
</tr>
</tbody>
</table>

### Table 4  Function Allocation

<table>
<thead>
<tr>
<th>Function</th>
<th>Function Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECCSR</td>
<td>Sets 16-bit asynchronous event counter functions, detects counter overflow, enables/disables input to ECH, ECL of the event clock.</td>
</tr>
<tr>
<td>ECH</td>
<td>Functions as the upper 8-bit up-counter of a 16-bit event counter, taking the ECL overflow signal as the input clock.</td>
</tr>
<tr>
<td>ECL</td>
<td>Functions as the lower 8-bit up-counter of a 16-bit event counter, taking the external asynchronous event AEVL pin as the input clock.</td>
</tr>
<tr>
<td>AVEL</td>
<td>Functions as the input pin for 2-MHz external asynchronous event input.</td>
</tr>
<tr>
<td>IRREC</td>
<td>Indicates whether there has been an asynchronous event counter interrupt request.</td>
</tr>
<tr>
<td>IENEC</td>
<td>Enables/disables asynchronous event counter interrupt requests.</td>
</tr>
</tbody>
</table>
Explanation of Operation

1. Figure 3 illustrates the principle of operation. Asynchronous event counter operation is based on the hardware and software processing indicated in the figure.
Explanation of Software

1. Explanation of Modules

Table 5 explains the modules in this task example.

Table 5   Explanation of Modules

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Label Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main routine</td>
<td>MAIN</td>
<td>Initializes the stack pointer, RAM, port 4, asynchronous event counter, and system control register; enables interrupts; executes direct transitions to subactive mode; after 524.288 ms, controls port output and executes direct transitions to active (high-speed) mode and to subactive mode.</td>
</tr>
<tr>
<td>Asynchronous event counter interrupt processing routine</td>
<td>AECINT</td>
<td>By routine for processing asynchronous event counter interrupts, clears an interrupt request flag, increments and initializes an 8-bit counter, and after 524.288 ms, sets a flag in RAM.</td>
</tr>
<tr>
<td>Direct transition interrupt processing routine</td>
<td>DTINT</td>
<td>By routine for processing direct transition interrupts, clears the interrupt request flag</td>
</tr>
</tbody>
</table>

2. Explanation of Arguments

In this task example, no arguments are used.
3. Explanation of Internal Registers Used

Table 6 gives explanations of the internal registers used in this task example.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECCSR OVH</td>
<td>Event counter control/status register (Counter overflow H) A status flag indicating overflow of ECH.  • When OVH = 0, indicates no overflow of ECH  • When OVH = 1, indicates ECH overflow</td>
<td>H'FF95 Bit 7</td>
<td>0</td>
</tr>
<tr>
<td>ECCSR OVL</td>
<td>Event counter control/status register (Counter overflow L) A status flag indicating overflow of ECL.  • When OVL = 0, indicates no overflow of ECL  • When OVL = 1, indicates ECL overflow</td>
<td>H'FF95 Bit 6</td>
<td>0</td>
</tr>
<tr>
<td>ECCSR CH2</td>
<td>Event counter control/status register (Channel selection) Selects whether to use ECH and ECL as a single-channel 16-bit event counter, or as two independent 8-bit event counter channels.  • When CH2 = 0, ECH and ECL function as a single concatenated 16-bit event counter  • When CH2 = 1, ECH and ECL function as two independent 8-bit event counter channels</td>
<td>H'FF95 Bit 4</td>
<td>0</td>
</tr>
<tr>
<td>ECCSR CUEH</td>
<td>Event counter control/status register (Count-up enable H) Enables or disables the event clock input to ECH.  • When CUEH = 0, disables the event clock input to ECH  • When CUEH = 1, enables the event clock input to ECH</td>
<td>H'FF95 Bit 3</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 6  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECCSR CUEL</td>
<td>Event counter control/status register (Count-up enable L) Enables or disables the event clock input to ECL. • When CUEL = 0, disables the event clock input to ECL • When CUEL = 1, enables the event clock input to ECL</td>
<td>H'FF95 Bit 2</td>
<td>0</td>
</tr>
<tr>
<td>ECCSR CRCH</td>
<td>Event counter control/status register (Counter reset control H) Controls ECH reset. • When CRCH = 0, ECH is reset • When CRCH = 1, ECH reset is canceled and count-up function is enabled</td>
<td>H'FF95 Bit 1</td>
<td>0</td>
</tr>
<tr>
<td>ECCSR CRCL</td>
<td>Event counter control/status register (Counter reset control L) Controls ECL reset. • When CRCL = 0, ECL is reset • When CRCL = 1, ECL reset is canceled and count-up function is enabled</td>
<td>H'FF95 Bit 0</td>
<td>0</td>
</tr>
<tr>
<td>ECH</td>
<td>Event counter H An 8-bit readable up-counter; by combining it with ECL, it can operate as the upper 8 bits of a 16-bit event counter.</td>
<td>H'FF96</td>
<td>H'00</td>
</tr>
<tr>
<td>ECL</td>
<td>Event counter L An 8-bit readable up-counter; by combining it with ECH, it can operate as the lower 8 bits of a 16-bit event counter.</td>
<td>H'FF97</td>
<td>H'00</td>
</tr>
</tbody>
</table>
Table 6  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMA</td>
<td>Timer mode register A (Internal clock selector 3)</td>
<td>H'FFB0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Selects the clock input to TCA.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When TMA3 = 0, PSS is selected as the TCA input clock source, and an interval timer function is selected for timer A</td>
<td>Bit 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When TMA3 = 1, PSW is selected as the TCA input clock source, and a clock time base function is selected for timer A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PMR3</td>
<td>Port mode register 3 (P3_/AEVL pin function switch)</td>
<td>H'FFCA</td>
<td>1</td>
</tr>
<tr>
<td>AVEL</td>
<td>Determines whether the P3_/AEVL pin is to be used as the P3, pin, or as the AEVL pin.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When AEVL = 0, the P3_/AEVL pin functions as the P3, pin</td>
<td>Bit 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When AEVL = 1, the P3_/AEVL pin functions as the AEVL pin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDR4</td>
<td>Port data register 4 (P4_0)</td>
<td>H'FFD7</td>
<td>0</td>
</tr>
<tr>
<td>P4_0</td>
<td>Stores the P4_0 pin data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When P4_0 = 0, the P4_0 pin output level is low</td>
<td>Bit 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When P4_0 = 1, the P4_0 pin output level is high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCR4</td>
<td>Port control register 4 (Port control register 4_)</td>
<td>H'FFE7</td>
<td>1</td>
</tr>
<tr>
<td>PCR4_0</td>
<td>Controls the P4_0 pin input/output.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When PCR4_0 = 0, the P4_0 pin functions as an input pin</td>
<td>Bit 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When PCR4_0 = 1, the P4_0 pin functions as an output pin</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 6  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
</table>
| SYSCR1 SSBY   | System control register 1  
(Software standby)  
Carries out transition to standby mode or watch mode.  
• When SSBY = 0, after executing a SLEEP instruction in active mode, causes a transition to sleep mode, or after executing a SLEEP instruction in subactive mode, causes a transition to subsleep mode  
• When SSBY = 1, after executing a SLEEP instruction in active mode, causes a transition to standby mode or to watch mode, or after executing a SLEEP instruction in subactive mode, causes a transition to watch mode | H'FFF0  
Bit 7 | 1 |
| SYSCR1 STS2 STS1 STS0 | System control register 1  
(Standby timer select 2 to 0)  
Specify the time for the CPU and peripheral functions to wait until the clock stabilizes when standby mode or watch mode is canceled and a transition is made to active mode due to a specific interrupt.  
• When STS2 to STS1 = 000, standby time is 8,192 states  
• When STS2 to STS1 = 001, standby time is 16,384 states  
• When STS2 to STS1 = 010, standby time is 32,768 states  
• When STS2 to STS1 = 011, standby time is 65,536 states  
• When STS2 to STS1 = 100, standby time is 131,072 states  
• When STS2 to STS1 = 101, standby time is 2 states  
• When STS2 to STS1 = 110, standby time is 8 states  
• When STS2 to STS1 = 111, standby time is 16 states | H'FFF0  
Bit 6 to bit 4 | STS2 = 0  
STS1 = 0  
STS0 = 0 |
<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSCR1 LSON</td>
<td>System control register 1 (Low speed on flag)</td>
<td>H'FFF0 Bit 3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>When watch mode is canceled, selects either the system clock ( \phi ) or the subclock ( \phi_{sub} ) as the CPU operating clock.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When LSON = 0, selects the system clock ( \phi ) as the CPU operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When LSON = 1, selects the subclock ( \phi_{sub} ) as the CPU operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSCR2 NESEL</td>
<td>System control register 2 (Noise elimination sampling frequency selection)</td>
<td>H'FFF1 Bit 4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Selects the frequency at which the watch clock signal ( \phi_w ) generated by the subclock oscillator is sampled relative to the oscillator clock ( \phi_{osc} ) generated by the system clock oscillator.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When NESEL = 0, sampling rate is ( \phi_{osc} /16 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When NESEL = 1, sampling rate is ( \phi_{osc} /4 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register Name</td>
<td>Description</td>
<td>RAM Address</td>
<td>Setting</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>-------------</td>
<td>---------</td>
</tr>
</tbody>
</table>
| SYSCR2 DTON   | System control register 2 (Direct transfer on flag) Specifies whether or not to make direct transitions among active (high-speed) mode, active (medium-speed) mode, and subactive mode when a SLEEP instruction is executed.  
- When DTON = 0, if a SLEEP instruction is executed in active mode, a transition to standby mode, watch mode or sleep mode occurs; if a SLEEP instruction is executed in subactive mode, a transition to watch mode or subsleep mode occurs  
- When DTON = 1, if a SLEEP instruction is executed in active (high-speed) mode, a direct transition occurs to active (medium-speed) mode (when SSBY = 1, MSON = 1, LSON = 0) or to subactive mode (when SSBY = 1, TMA3 = 1, LSON = 1); if a SLEEP instruction is executed in active (medium-speed) mode, a direct transition occurs to active (high-speed) mode (when SSBY = 0, MSON = 0, LSON = 0) or to subactive mode (when SSBY = 1, TMA3 = 1, LSON = 1); and if a SLEEP instruction is executed in subactive mode, a direct transition occurs to active (high-speed) mode (when SSBY = 1, TMA3 = 1, LSON = 0, MSON = 0) or to active (medium-speed) mode (when SSBY = 1, TMA3 = 1, LSON = 0, MSON = 1) | H'FFF1 Bit 3 | 1 |
<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSCR2 MSON</td>
<td>System control register 2 (Medium speed on flag) Selects whether to operate in active (high-speed) mode or in active (medium-speed) mode after cancellation of standby mode, watch mode, or sleep mode. • When MSON = 0, operates in active (high-speed) mode • When MSON = 1, operates in active (medium-speed) mode</td>
<td>H'FFF1 Bit 2</td>
<td>0</td>
</tr>
<tr>
<td>SYSCR2 SA1 SA0</td>
<td>System control register 1 (Subactive mode clock select 1, 0) Select the CPU clock rate ($\phi_w/8$, $\phi_w/4$, $\phi_w/2$) in subactive mode. • When SA1 = 0 and SA0 = 0, $\phi_w/8$ is selected • When SA1 = 0 and SA0 = 1, $\phi_w/4$ is selected • When SA1 = 1 and SA0 = *, $\phi_w/2$ is selected *: Don't care</td>
<td>H'FFF0 Bit 1, bit 0</td>
<td>1</td>
</tr>
<tr>
<td>IRR2 IRRDT</td>
<td>Interrupt request register 2 (Direct transition interrupt request flag) Indicates whether there has been a direct transition interrupt request. • When IRRDT = 0, indicates that no direct transition interrupt has been requested • When IRRDT = 1, indicates that a direct transition interrupt has been requested</td>
<td>H'FFF7 Bit 7</td>
<td>0</td>
</tr>
</tbody>
</table>
Table 6   Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRR2 IRREC</td>
<td>Interrupt request register 2 (Asynchronous event counter interrupt request flag)</td>
<td>H'FFF7</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Indicates whether there has been an asynchronous event counter interrupt request.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When IRREC = 0, indicates that no asynchronous event counter interrupt has been requested.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When IRREC = 1, indicates that an asynchronous event counter interrupt has been requested</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IENR2 IENDT</td>
<td>Interrupt enable register 2 (Direct transition interrupt enable)</td>
<td>H'FFF4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Enables or disables direct transition interrupt requests.</td>
<td>Bit 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When IENDT = 0, disables direct transition interrupt requests.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When IENDT = 1, enables direct transition interrupt requests</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IENR2 IENEC</td>
<td>Interrupt enable register 2 (Asynchronous event counter interrupt enable)</td>
<td>H'FFF4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Enables or disables asynchronous event counter interrupt requests.</td>
<td>Bit 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When IENEC = 0, disables asynchronous event counter interrupt requests</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When IENEC = 1, enables asynchronous event counter interrupt request</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. Explanation of RAM Usage

Table 7 explains RAM usage for this task example.

<table>
<thead>
<tr>
<th>Label Name</th>
<th>Function</th>
<th>RAM Address</th>
<th>Modules Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLAG</td>
<td>Flag indicating 524.288 ms have elapsed.</td>
<td>H'F780</td>
<td>MAIN, AECINT</td>
</tr>
<tr>
<td>CNT</td>
<td>8-bit counter to count the number of occurrences of a timer F interrupt request.</td>
<td>H'F781</td>
<td>MAIN, AECINT</td>
</tr>
</tbody>
</table>
Flowchart

1. Main routine

- Initialize the stack pointer to H'FF80.
- Mask the I bit of CCR to disable an interrupt.
- Initialize the RAM.
- Initialize the P4_0 pin; set PDR and PCR for the P4_0 pin to 0 and 1, respectively, to initialize the P4_0 pin output to low level.
- Reset the 16-bit event counter.
- Initialize the AEVL pin; specify the P3_7/AEVL pin function as the AEVL input pin.
- Specify SSBY = 1, LSON = 1, DTON = 1, and TMA3 = 1 to make a direct transition from active (high-speed) mode to subactive mode.
- Clear the direct transition interrupt request flag to 0.
- Set the IENDT bit to 1 to enable direct transition interrupt requests.
- Clear the I bit of CCR to enable an interrupt.
- Execute a SLEEP instruction to make a direct transition to subactive mode.
1. Main routine (cont)

0 → IRREC

H'81 → IENR2

H'0F → ECCSR

Specify SSBY = 1, LSON = 0, MSON = 0, DTON = 1, and TMA3 = 1 to make a direct transition from subactive mode to active (high-speed) mode.

Set the iENEC bit to 1 to enable asynchronous event counter interrupt requests.

Start the counting-up of the 16-bit event counter.

H'00 → FLAG

H'E7 → SYSCR1

H'F8 → SYSCR2

H'18 → TMA3

SLEEP

NOP

Invert P4 0

H'8F → SYSCR1

H'F8 → SYSCR2

H'18 → TMA3

SLEEP

NOP

Check whether or not 524.288 ms have elapsed.

Clear the asynchronous event counter interrupt request flag to 0.

Specify SSBY = 1, LSON = 1, DTON = 1, and TMA3 = 1 to make a direct transition from active (high-speed) mode to subactive mode.

Clear FLAG to H'00.

Check whether or not 524.288 ms have elapsed.

Start the counting-up of the 16-bit event counter.
2. Asynchronous event counter interrupt processing routine

- **AECINT**
  - **PUSH R0** — Save the R0 register data onto the stack.
  - **0 → IRREC** — Clear the asynchronous event counter interrupt request flag to 0.
  - **0 → OVH** — Clear the counter overflow H bit to 0.
  - **0 → OVL** — Clear the counter overflow L bit to 0.
  - **CNT + 1 → CNT** — Increment the 8-bit counter (CNT) specified in the RAM.

  - **No** — Check whether or not the 8-bit counter (CNT) has been incremented to 10.
    - **CNT = H'10?** — Specify FLAG to H'01.
    - **H'00 → CNT** — Initialize CNT to H'00.
  - **Yes** — Save the R0 register data onto the stack.

- **POP R0** — Restore R0 from the stack.

- **RTE**

3. Direct transition interrupt processing routine

- **AECINT**
  - **0 → IRRDT** — Clear the direct transition interrupt request flag to 0.

- **RTE**
Program Lists

;***********************************************************
;     'Asynchronous Event Counter Control'
;***********************************************************
; Function : AEC(Asynvhronous Event Counter) *
;***********************************************************
; External Clock : 6MHz
; Internal Clock : 3MHz
; Sub Clock      : 32.768kHz
;***********************************************************

.cpu  3001

;***********************************************************
; Symbol Definition
;***********************************************************

ECCSR .equ h'ff95 ;Event Counter Control/Status Register
ECH    .equ h'ff96 ;Event Counter H
ECL    .equ h'ff97 ;Event Counter L
TMA    .equ h'ffb0 ;Timer Mode Register A
PMR3   .equ h'ffca ;Port Mode Register 3
PDR4   .equ h'ffd7 ;Port Data Register 4
PCR4   .equ h'ffe7 ;Port Control Register 4
SYSCR1 .equ h'fff0 ;System Control Register 1
SYSCR2 .equ h'fff1 ;System Control Register 2
IENR2  .equ h'fff4 ;Interrupt Enable Register 2
IRR2   .equ h'fff7 ;Interrupt Request Register 2

;***********************************************************
; RAM Allocation
;***********************************************************

FLAG  .equ h'f780 ;Bit0 : Event Flag
CNT    .equ h'f781 ;8-bit Counter

Rev. 1.0, 08/99, page 30 of 151
Vector Address

MAIN: .equ $
mov.w #h'ff80,sp ;Initialize Stack Pointer
orc #h'80,ccr ;Interrupt Disable
; mov.b #h'00,r0l
mov.b r0l,@CNT ;Initialize 8-bit Counter
mov.b r0l,@FLAG ;Initialize Event Flag
;
mov.w #h'f8f9,r0
mov.b r0h,@PDR4 ;Initialize P40 PDR
mov.b r0l,@PCR4 ;Initialize P40 Terminal Function
;
mov.b #h'00,r0l
mov.b r0l,@ECCSR ;Reset 16-bit Event Counter
mov.b #h'84,r0l
mov.b r0l,@PMR3 ;Initialize AEVL Terminal Function
;
mov.b #h'8f,r0l ;SSBY="1", LSON="1"
mov.b r0l,@SYSCR1 ;DTON="1", TMA3="1"
mov.b #h'F8,r0l
mov.b r0l,@SYSCR2
mov.b #h'18,r0l
mov.b r0l,@TMA
;
bclr #7,@IRR2 ;Clear IRRDT
mov.b #h'80,r0l
mov.b r0l,@IENR2 ;Direct Transfer Interrupt Enable
;
andc #h'7f,ccr ;Interrupt Enable
;
sleep ;Direct Transfer to Subactive Mode
nop
;
bclr #0,@IRR2 ;Clear IRREC
mov.b #h'81,r0l
mov.b r0l,@IENR2 ;Asynchronous Event Counter Interrupt Enable
;
mov.b #h'0f,r0l
mov.b r0l,@ECCSR ;16-bit Event counter count-up start

Rev. 1.0, 08/99, page 32 of 151

Renesas
EVTMN:  
    mov.b @FLAG, r0l  
    btst #0, r0l  ; Event Flag = "1" ?  
    beq EVTNN  ; No.  
    
    mov.b #h'00, r0l  ; Clear Event Flag  
    mov.w #h'e7f8, r0  ; SSBY="1", LSON="0"  
    mov.b r0h, @SYSCR1  ; MSON="0", DTON="1"  
    mov.b r0l, @SYSCR2  ; TMA3="1"  
    mov.b #h'18, r0l  ; STS2-0="000"  
    mov.b r0l, @TMA  
    
    sleep  ; Direct Transfer to Active Mode  
    nop  
    
    mov.b @PDR4, r0l  ; Load PDR4  
    bnot #0, r0l  ; Invert P40 PDR  
    mov.b r0l, @PDR4  ; Store PDR4  
    
    mov.w #h'8ff8, r0  ; SSBY="1", LSON="1"  
    mov.b r0h, @SYSCR1  ; DTON="1", TMA3="1"  
    mov.b r0l, @SYSCR2  
    mov.b #h'18, r0l  
    mov.b r0l, @TMA  
    
    sleep  ; Direct Transfer to Subactive Mode  
    nop  
    
    bra EVTNN  
    
;*****************************************************************  
;*  AECINT : AEC Interrupt Routine  
;*****************************************************************  

AECINT: .equ $  

push r0  ; Store r0

Revis 1.0, 08/99, page 33 of 151
; bclr  0@IRR2 ;Clear IRREC
bclr  7@ECCSR ;Clear OVH
bclr  6@ECCSR ;Clear OVL
;
mov.b @CNT,r0l ;Load CNT
inc  r0l  ;Increment CNT
cmp.b #h'10,r0l ;CNT = h'10 ?
beq  EVNT ;Yes. CNT Initialize
mov.b r0l,@CNT ;Store CNT
bra  RNFI
;
EVNT:  mov.b #h'01,r0l
mov.b r0l,@FLAG ;Set Event Flag
mov.b #h'00,r0l
mov.b r0l,@CNT ;Initialize 8-bit Counter
;
RNFI  pop  r0 ;Restore r0
rte
;
;******************************************************************************
;* DTINT : Direct Transfer Interrupt Routine
;******************************************************************************
;
DTINT:   .equ  $  
bclr  7@IRR2 ;Clear IRRDT
rte
;
.end
2.3 LCD Display with Static Duty

| LCD Display with Static Duty | MCU: H8/3867 Series | Functions Used: LCD Controller/Driver |

Specifications

1. LCD display is performed using the segment-type LCD controller circuit, LCD driver, and power supply circuit of the H8/3867 Series.

2. A single common signal and 32 segment signals are used for LCD display with static duty.

3. As the power supply driving the LCD, a step-up constant-voltage power supply (5 V) is used.

4. An example of LCD module connection and an LCD display example for this task example appear in figure 1.

Figure 1   LCD Display Example
Explanation of Functions Used

1. In this task example, the LCD controller/driver is used for LCD display. The features of the LCD controller/driver are described below.

   - Display capacity
     a. Duty cycle: static
        Internal driver: 32 segments
        Segment external-expansion driver: 256 segments
     b. Duty cycle: 1/2
        Internal driver: 32 segment
        Segment external-expansion driver: 128 segments
     c. Duty cycle: 1/3
        Internal driver: 32 segment
        Segment external-expansion driver: 64 segments
     d. Duty cycle: 1/4
        Internal driver: 32 segment
        Segment external-expansion driver: 64 segments
   - LCD RAM capacity: 8 bits × 32 bytes (256 bits)
   - LCD RAM is word-accessible.
   - All segment output pins can be used as port pins in eight-pin units.
   - Depending on the duty cycle, the common output pins not used can be used for a common double-buffer (parallel connection).
   - Display is possible in all operating modes other than standby mode.
   - Frame frequency can be selected from among 11 values.
   - A power supply split-resistance is built-in, for supply of LCD driver power.
   - Use of module standby mode enables a module to be placed in standby mode independently when not used.
   - An internal step-up constant-voltage (5 V) power supply enables LCD display even at low voltages.
   - A or B waveform can be selected by software.
2. Figure 2 is a block diagram of the LCD controller/driver used in this task examples.

Figure 2  Block Diagram of LCD Controller/Driver (LCD Display with Static Duty)
3. Functions of the LCD controller/driver are explained in table 1.

**Table 1  LCD Controller/Driver Functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LCD port control register (LPCR)</strong></td>
<td>LPCR is an 8-bit read/write register which selects the duty cycle, the LCD driver and pin functions. LPCR is initialized to H'00 upon reset.</td>
</tr>
<tr>
<td><strong>LCD control register (LCR)</strong></td>
<td>LCR is an 8-bit read/write register which turns the LCD drive power supply on and off, controls display data, and selects the frame frequency. LCR is initialized to H'80 upon reset.</td>
</tr>
<tr>
<td><strong>LCD control register 2 (LCR2)</strong></td>
<td>LCR2 is an 8-bit read/write register which controls switching between A and B waveforms, selects the driver power supply, controls the step-up constant-voltage (5 V) power supply, and selects the duty cycle for charge/discharge pulses controlling disconnection of the power supply split-resistance from power supply circuit. LCR2 is initialized to H'60 upon reset.</td>
</tr>
<tr>
<td><strong>Segment output pins (SEG_{32} to SEG_{1})</strong></td>
<td>These are pins used for LCD segment driving; all these pins are multiplexed as port pins, and their functions can be selected programmably.</td>
</tr>
<tr>
<td><strong>Common output pins (COM_{4} to COM_{1})</strong></td>
<td>These are LCD common driving output pins; under static or 1/2-duty driving, they can be configured in parallel.</td>
</tr>
<tr>
<td><strong>Segment external expansion signal pin (CL_{1})</strong></td>
<td>This is a display data latch clock pin which is multiplexed as SEG_{32}.</td>
</tr>
<tr>
<td><strong>Segment external expansion signal pin (CL_{2})</strong></td>
<td>This is a display data shift clock pin which is multiplexed as SEG_{31}.</td>
</tr>
<tr>
<td><strong>Segment external expansion signal pin (M)</strong></td>
<td>This is an LCD alternation signal pin which is multiplexed as SEG_{29}.</td>
</tr>
<tr>
<td><strong>Segment external expansion signal pin (DO)</strong></td>
<td>This is a serial display data signal pin which is multiplexed as SEG_{30}.</td>
</tr>
<tr>
<td><strong>LCD power supply pins (V_{0} to V_{3})</strong></td>
<td>These pins are used when connecting an external bypass capacitor or when using an external power supply circuit.</td>
</tr>
<tr>
<td><strong>LCD RAM</strong></td>
<td>Sets the display data. The relation between the LCD RAM and the display segments differs depending on the duty cycle. After the registers necessary for display have been set, instructions similar to the instructions for normal RAM are used to write data corresponding to the duty, and when the display is turned on, display is started automatically. Word/byte access instructions can be used to set data in the LCD RAM.</td>
</tr>
</tbody>
</table>
4. In this task example, a 16-line 8-character segment LCD is used for display under static driving. Figure 3 is a diagram showing connections for segment signals and common signals of 16-line 8-character segment LCD used in this task example.

![Diagram of Segment Signals and Common Signals](image)

**Figure 3** Connections of Segment Signals and Common Signals of the 16-Line 8-Character Segment LCD Used in this Task Example
5. Figure 4 shows the LCD RAM mapping under static driving without segment external expansion.

![Figure 4 LCD RAM Mapping under Static Driving without Segment External Expansion](image)

Rev. 1.0, 08/99, page 40 of 151
6. Figure 5 shows the relation between the 16-line 8-character segment LCD display and LCD RAM settings used in this task example. In this example, the LCD display is cycled through the series pattern 1 → pattern 2 → pattern 3 → pattern 4 → pattern 1 →.

1. Pattern 1

2. Pattern 2

3. Pattern 3

4. Pattern 4

Figure 5  Relation between LCD Display and LCD RAM Settings
7. Figure 6 shows the relation between the LCD RAM addresses and the segments SEG, through SEG 8 of the 16-line 8-character segment LCD. As the figure indicates, when the LCD RAM bits corresponding to 0 through 7 are set to 1 the corresponding LCD areas are lit, and when cleared to 0 the corresponding areas are unlit.

![Figure 6](image)

Figure 6  Relation between LCD Lit/Unlit States and LCD RAM Settings

8. Table 2 indicates function allocations in this task example.

<table>
<thead>
<tr>
<th>Function</th>
<th>Function Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR</td>
<td>Selects duty cycle, LCD driver, and pin functions.</td>
</tr>
<tr>
<td>LCR</td>
<td>Turns LCD drive power supply on and off, controls display data, and selects frame frequency.</td>
</tr>
<tr>
<td>LCR2</td>
<td>Switches between A and B waveforms, selects drive power supply, controls step-up constant-voltage (5 V) power supply, selects duty cycle for charge/discharge pulses to control disconnection of power supply split-resistance from power supply circuit.</td>
</tr>
<tr>
<td>SEG 0 to SEG 8</td>
<td>Used as segment drivers.</td>
</tr>
<tr>
<td>COM 1</td>
<td>Used as a common driver.</td>
</tr>
<tr>
<td>V 0, V 1</td>
<td>The V 0 and V 1 pins are shorted in order to use the step-up constant-voltage (5 V) power supply as the LCD drive power supply.</td>
</tr>
<tr>
<td>LCD RAM</td>
<td>Sets the LCD display data.</td>
</tr>
</tbody>
</table>

Table 2  Function Allocations

Rev. 1.0, 08/99, page 42 of 151
Explanation of Functions Used

1. Hardware settings for LCD display are explained below.

   a. LCD drive power supply settings
      The H8/3867 Series can use either the internal power supply circuit or an external power supply circuit as the LCD drive power supply. In addition, either the power supply voltage \( V_{cc} \) or the step-up constant-voltage (5 V) can be selected for the internal power supply circuit.

      When using the internal power supply circuit to drive the LCD, the \( V_i \) and \( V_t \) pins are connected externally, as illustrated in figure 7.

      In this task example, the step-up constant-voltage power supply is used as the LCD drive power supply.

![Figure 7 Example of Connection of LCD Power Supply Pins
When Using Internal Power Supply Circuit](image-url)
b. Contrast control function

A block diagram of the LCD drive power supply circuit appears in figure 8. Either \( V_{cc} \) or a 5 V output from the step-up constant-voltage power supply circuit is output to pin \( V_0 \). When these voltages are used directly to drive the LCD, the \( V_0 \) and \( V_1 \) pins should be shorted. By inserting a variable resistance \( R \) between the \( V_0 \) and \( V_1 \) pins, the voltage applied to the \( V_1 \) pin can be adjusted, and the LCD panel contrast can be controlled.

![Block Diagram of LCD Drive Power Supply Circuit](image)

**Figure 8** Block Diagram of LCD Drive Power Supply Circuit

c. Step-up constant-voltage (5 V) power supply

The H8/3867 Series has an internal step-up constant-voltage (5 V) power supply, supplying a constant 5 V independent of \( V_{cc} \).

By setting SUPS of the LCD control register 2 (LCR2) to 1, the step-up constant-voltage (5 V) power supply is activated, and a constant 5 V is output to the \( V_0 \) pin. This can be used either with pins \( V_0 \) and \( V_1 \) short-circuited, or with a resistance inserted to divide the voltage.

Note: The step-up constant-voltage (5 V) power supply must not be used for purposes other than to drive the LCD. In addition, when driving a large panel, the power supply capacity may be insufficient. In such cases, either \( V_{cc} \) or an external power supply circuit can be used as the power supply.

Rev. 1.0, 08/99, page 44 of 151
2. Software settings for LCD display are explained below.
   
a. Duty selection
   DTS1 and DTS0 are used to select from among static, 1/2 duty, 1/3 duty, and 1/4 duty.

b. Segment driver selection
   SGS3 to SGS0 are used to select the segment drivers to be used.

c. Frame frequency selection
   By setting CKS3 to CKS0, the frame frequency can be selected. The frame frequency should be selected according to the LCD panel.

d. Selection of A and B waveforms
   LCDAB can be used to select either the A or the B waveform for use as the LCD waveform.

e. Selection of LCD drive power supply
   When using the internal power supply circuit, SUPS can be used to select the power supply to be used. When using an external power supply circuit, SUPS is used to select Vcc, and PSW should be used to turn off the LCD drive power supply.
3. Figure 9 shows the operation principle of this task example.

| SEG 1 | SEG 2 | SEG 3 | SEG 4 | SEG 5 | SEG 6 | SEG 7 | SEG 8 | SEG 9 | SEG 10 | SEG 11 | SEG 12 | SEG 13 | SEG 14 | SEG 15 | SEG 16 | SEG 17 | SEG 18 | SEG 19 | SEG 20 | SEG 21 | SEG 22 | SEG 23 | SEG 24 | SEG 25 | SEG 26 | SEG 27 | SEG 28 | SEG 29 | SEG 30 | SEG 31 | SEG 32 |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
|       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |
|       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |
|       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |       |

**Figure 9  Operation Principle**
Explanation of Software

1. Explanation of Modules

   Table 3 explains the modules in this task example.

   **Table 3   Module Explanation**

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Label Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main routine</td>
<td>MAIN</td>
<td>Initializes the stack pointer, LCD RAM and LCD controller/driver, enables interrupts, and controls LCD display.</td>
</tr>
</tbody>
</table>

2. Explanation of Arguments

   In this task example, no arguments are used.

3. Explanation of Internal Registers Used

   Table 4 gives explanation of the internal registers used in this task example.

   **Table 4   Explanation of Internal Registers Used**

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR, DTS1, DTS0</td>
<td>LCD port control register (Duty cycle selection 1, 0)</td>
<td>H'FFC0</td>
<td>DTS1 = 0</td>
</tr>
<tr>
<td></td>
<td>Select duty from among static, 1/2 duty, 1/3 duty, and 1/4 duty.</td>
<td>Bit 7, bit 6</td>
<td>DTS0 = 0</td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 0 and DTS0 = 0, static duty is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 0 and DTS0 = 1, 1/2 duty is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 1 and DTS0 = 0, 1/3 duty is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 1 and DTS0 = 1, 1/4 duty is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register Name</td>
<td>Description</td>
<td>RAM Address</td>
<td>Settings</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>LPCR</td>
<td>LCD port control register (Common function selection)</td>
<td>H'FFC0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether the same waveform is output from several pins in order to increase the common driving capacity, when common pins are not selected for a given duty cycle.</td>
<td>Bit 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMX = 0, the same waveform is not output from multiple common pins not used at that duty cycle.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMX = 1, the same waveform is output from multiple common pins not used at that duty cycle.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPCR</td>
<td>LCD port control register (Expansion signal select)</td>
<td>H'FFC0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether the SEG&lt;sub&gt;32&lt;/sub&gt;/CL&lt;sub&gt;1&lt;/sub&gt;, SEG&lt;sub&gt;31&lt;/sub&gt;/CL&lt;sub&gt;2&lt;/sub&gt;, SEG&lt;sub&gt;30&lt;/sub&gt;/D0, and SEG&lt;sub&gt;29&lt;/sub&gt;/M pins are used as segment pins (SEG&lt;sub&gt;32&lt;/sub&gt; through SEG&lt;sub&gt;29&lt;/sub&gt;), or as segment external expansion signal pins (CL&lt;sub&gt;1&lt;/sub&gt;, CL&lt;sub&gt;2&lt;/sub&gt;, D0, M).</td>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, they are used as segment pins (SEG&lt;sub&gt;32&lt;/sub&gt; through SEG&lt;sub&gt;29&lt;/sub&gt;).</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 1, they are used as segment external expansion signal pins (CL&lt;sub&gt;1&lt;/sub&gt;, CL&lt;sub&gt;2&lt;/sub&gt;, D0, M).</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4   Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR SGS3 to SGS0</td>
<td>LCD port control register (Segment driver selection)</td>
<td>H'FFC0</td>
<td>SGS3 = 1</td>
</tr>
<tr>
<td></td>
<td>Select the segment driver to be used.</td>
<td>Bit 3 to bit 0</td>
<td>SGS2 = 0</td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 0, pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG, function as ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 1, pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG, function as ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 1 and SGS0 = *, pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG&lt;sub&gt;m&lt;/sub&gt;, function as segment drivers and pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG, function as ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, SGS3 = 0, SGS2 = 1, SGS1 = 0 and SGS0 = *, pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG&lt;sub&gt;m&lt;/sub&gt;, function as segment drivers and pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG, function as ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, SGS3 = 1, SGS2 = *, SGS1 = * and SGS0 = *, pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG, function as segment drivers</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 1, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 0, pins SEG&lt;sub&gt;n&lt;/sub&gt; through SEG&lt;sub&gt;m&lt;/sub&gt;, function as external expansion pins and pins SEG&lt;sub&gt;m&lt;/sub&gt; through SEG, function as ports</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• SGX = 1, SGS3 = *, SGS2 = *, SGS1 = * and SGS0 = * cannot be specified</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*: Don’t care
## Table 4  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR PSW</td>
<td>LCD control register (LCD drive power supply on/off control)</td>
<td>H'FFC1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Turns the LCD drive power supply off when LCD display is not used in power-down mode, or when an external power supply is used. When the ACT bit is cleared to 0, or when in standby mode, the LCD drive power supply is turned off regardless of this bit setting.</td>
<td>Bit 6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When PSW = 0, the LCD drive power supply is turned off</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When PSW = 1, the LCD drive power supply is turned on</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCR ACT</td>
<td>LCD control register (Display function activate)</td>
<td>H'FFC1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Selects whether the LCD controller/driver is to be used or not. By clearing this bit to 0, LCD controller/driver operation is halted. Also, regardless of the value of PSW, the LCD drive power supply is turned off. However, the register contents are maintained.</td>
<td>Bit 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When ACT = 0, LCD controller/driver operation is halted</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When ACT = 1, LCD controller/driver operations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCR DISP</td>
<td>LCD control register (Display data control)</td>
<td>H'FFC1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>DISP selects whether the LCD RAM contents or blank data are to be displayed.</td>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DISP = 0, blank data is displayed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DISP = 1, LCD RAM data is displayed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR</td>
<td>LCD control register (Frame frequency select 3 to 0)</td>
<td>H'FFC1</td>
<td>CKS3 = 1</td>
</tr>
<tr>
<td></td>
<td>Select the operating clock and the frame frequency.</td>
<td>Bit 3 to bit 0</td>
<td>CKS2 = 1</td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 0, CKS2 = *, CKS1 = 0 and CKS0 = 0, φw is selected as operating clock</td>
<td></td>
<td>CKS1 = 1</td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 0, CKS2 = *, CKS1 = 0 and CKS0 = 1, φw/2 is selected as operating clock</td>
<td></td>
<td>CKS0 = 0</td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 0, CKS2 = *, CKS1 = 1 and CKS0 = *, φw/4 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 0, CKS1 = 0 and CKS0 = 0, φ2 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 0, CKS1 = 1 and CKS0 = 1, φ4 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 0, CKS1 = 1 and CKS0 = 0, φ8 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 0, CKS1 = 1 and CKS0 = 1, φ16 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 1, CKS1 = 0 and CKS0 = 0, φ32 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 1, CKS1 = 0 and CKS0 = 1, φ64 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 1, CKS1 = 1 and CKS0 = 0, φ128 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKS3 = 1, CKS2 = 1, CKS1 = 1 and CKS0 = 1, φ256 is selected as operating clock</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*: Don't care
Table 4  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR2</td>
<td>LCDAB (A waveform/B waveform switching control)</td>
<td>H'FFC2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether the A or B waveform is to be used for LCD driving.</td>
<td>Bit 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When LCDAB = 0, the LCD is driven using the A waveform</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When LCDAB = 1, the LCD is driven using the B waveform</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCR2</td>
<td>SUPS (Drive power supply select, step-up constant-voltage (5 V) power supply control)</td>
<td>H'FFC2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Selects whether the A or B waveform is to be used for LCD driving.</td>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SUPS = 0, the drive power supply is V_{CC}, and the step-up constant-voltage (5 V) power supply operation is halted</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SUPS = 1, the drive power supply is 5 V, and the step-up constant-voltage (5 V) power supply operates</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 4  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR2</td>
<td>LCD control register 2 (Charge/discharge pulse duty cycle select 3 to 0) Select the duty cycle while the power supply split-resistance is connected to the power supply circuit. When CDS3 = 0, CDS2 = 0, CDS1 = 0 and CDS0 = 0, the duty cycle is 1.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>H'FFC2</td>
<td>CDS3 = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bit 3 to bit 0</td>
<td>CDS2 = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CDS1 = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CDS0 = 0</td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 0, CDS1 = 0 and CDS0 = 1, the duty cycle is 1/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 0, CDS1 = 1 and CDS0 = 0, the duty cycle is 2/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 0, CDS1 = 1 and CDS0 = 1, the duty cycle is 3/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 1, CDS1 = 0 and CDS0 = 0, the duty cycle is 4/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 1, CDS1 = 0 and CDS0 = 1, the duty cycle is 5/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 1, CDS1 = 1 and CDS0 = 0, the duty cycle is 6/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 1, CDS2 = 1, CDS1 = 1 and CDS0 = 1, the duty cycle is 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 1, CDS2 = 0, CDS1 = * and CDS0 = *, the duty cycle is 1/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 1, CDS2 = 1, CDS1 = * and CDS0 = *, the duty cycle is 1/32</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>*: Don’t care</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. Explanation of RAM Usage

In this task example, RAM is not used.
Flowchart

1. Main routine

```
Flowchart
1. Main routine

MAIN
H'FF80 → SP
1 → CCR I-bit
H'00 → H'F740 to H'F743
H'08 → LPCR
H'FE → LCR
H'70 → LCR2
0 → CCR I-bit
H'1111 → R0

R0 → H'F740
R0 → H'F742
H'0000 → R1
H'0000 → R2
H'0005 → R3

R1 + 1 → R1
Yes
R1 = H'0000?
No
R2 + 1 → R2
Yes
R2 = R3?
No

ROH rotate left
R0L rotate left

---

Set display data to LCD RAM.
Initialize general registers.
R1 = H'0000
R2 = H'0000
R3 = H'0005

---

Set display data to R0.
Set display data to LCD RAM.

---

R1 ≠ H'0000?
Increment the R1 register.

---

R2 ≠ R3?
Increment the R2 register.

---

Software timer

---

Rotate ROH, and R0L registers to the left.
Display data: H'1111 → H'2222 → H'4444 →
H'8888 → H'1111 → ....
```

Rev. 1.0, 08/99, page 54 of 151
Program Lists

;***********************************************************************
;*  H8/3867 Application Note
;*
;*  'Liquid Crystal Display
;*  -Static Drive, Internal Driver-
;*
;*  Function : LCD Controller / Driver
;*
;*  External Clock : 6MHz
;*  Internal Clock : 3MHz
;*  Sub Clock      : 32.768kHz
;***********************************************************************
;
.cpu 3001
;
;***********************************************************************
;*  Symbol Definition
;***********************************************************************
;
LPCR .equ h'ffc0 ;LCD Port Control Register
LCR  .equ h'ffc1 ;LCD Control Register
LCR2 .equ h'ffc2 ;LCD Control Register 2
;
;***********************************************************************
;*  Vector Address
;***********************************************************************
;
.org h'0000
.data.w MAIN ;No.0 Reset Interrupt(H'0000-H'0001)
;
.org h'0008
.data.w MAIN ;No.4 _IRQ0 Interrupt(H'0008-H'0009)
.data.w MAIN ;No.5 _IRQ1 Interrupt(H'000A-H'000B)
.data.w MAIN ;No.6 _IRQ2 Interrupt(H'000C-H'000D)
.data.w MAIN ;No.7 _IRQ3 Interrupt(H'000E-H'000F)
.data.w MAIN ;No.8 _IRQ4 Interrupt(H'0010-H'0011)

Revis. 1.0, 08/99, page 55 of 151
.data.w MAIN ;No.9  _WKP0-_WK7 Interrupt(H'0012-H'0013)
;
.org  h'0016
.data.w MAIN ;No.11 Timer A Interrupt(H'0016-H'0017)
.data.w MAIN ;No.12 AEC Interrupt(H'0018-H'0019)
.data.w MAIN ;No.13 Timer C Interrupt(H'001A-H'001B)
.data.w MAIN ;No.14 Timer FL Interrupt(H'001C-H'001D)
.data.w MAIN ;No.15 Timer FH Interrupt(H'001E-H'001F)
.data.w MAIN ;No.16 Timer G Interrupt(H'0020-H'0021)
.data.w MAIN ;No.17 SCI31 Interrupt(H'0022-H'0023)
.data.w MAIN ;No.18 SCI32 Interrupt(H'0024-H'0025)
.data.w MAIN ;No.19 A/D Converter Interrupt(H'0026-H'0028)
.data.w MAIN ;No.20 Direct Transfer Interrupt(H'0028-H'0029)
;
;=========================================================================================
;* MAIN : Main Routine
;=========================================================================================
;
.org  h'1000
;
MAIN: .equ  $
    mov.w  #h'ff80,sp  ;Initialize Stack Pointer
    orc  #h'80,ccr  ;Interrupt Disable
;
    mov.w  #h'0000,r0  ;Initialize LCD RAM
    mov.w  r0,@(h'f740)
    mov.w  r0,@(h'f742)
;
    mov.b  #h'08,r01  ;Initialize LCD Controller/Driver
    mov.b  r01,0LPCR
    mov.b  #h'fe,r01
    mov.b  r01,0LCR
    mov.b  #h'70,r01
    mov.b  r01,0LCR2
;
    andc  #h'7f,ccr  ;Interrupt REnable
;
Rev. 1.0, 08/99, page 56 of 151
mov.w #h'1111,r0 ;Set LCD RAM

MAIN99: mov.w r0,@(h'f740)
mov.w r0,@(h'f742)

; sub.w r1,r1 ;Set Software Timer
sub.w r2,r2
mov.w #h'0005,r3

INC: adds #1,r1
mov.w r1,r1
bne INC

; adds #1,r2
cmp.w r2,r3
bne INC

; rotl r0h ;Display Data Rotate Left
rotl r0l

bra MAIN99

; .end
2.4 LCD Display with 1/4 Duty

| LCD Display with 1/4 Duty | MCU: H8/3867 Series | Functions Used: LCD Controller/Driver |

Specifications

1. LCD display is performed using the segment-type LCD controller circuit, LCD driver, and power supply circuit of the H8/3867 Series.

2. Four common signals and 32 segment signals are used for LCD display with 1/4 duty.

3. As the power supply driving the LCD, a step-up constant-voltage power supply (5 V) is used.

4. An example of LCD module connection and an LCD display example for this task example appear in figure 1.

![Diagram of LCD Display Example](image)

Figure 1 LCD Display Example
Explanation of Functions Used

1. In this task example, the LCD controller/driver is used for LCD display. The features of the LCD controller/driver are described below.

   • Display capacity
     a. Duty cycle: static
        Internal driver: 32 segments
        Segment external-expansion driver: 256 segments
     b. Duty cycle: 1/2
        Internal driver: 32 segment
        Segment external-expansion driver: 128 segments
     c. Duty cycle: 1/3
        Internal driver: 32 segment
        Segment external-expansion driver: 64 segments
     d. Duty cycle: 1/4
        Internal driver: 32 segment
        Segment external-expansion driver: 64 segments
   • LCD RAM capacity: 8 bits × 32 bytes (256 bits)
   • LCD RAM is word-accessible.
   • All segment output pins can be used as port pins in eight-pin units.
   • Depending on the duty cycle, the common output pins not used can be used for a common double-buffer (parallel connection).
   • Display is possible in all operating modes other than standby mode.
   • Frame frequency can be selected from among 11 values.
   • A power supply split-resistance is built-in, for supply of LCD driver power.
   • Use of module standby mode enables a module to be placed in standby mode independently when not used.
   • An internal step-up constant-voltage (5 V) power supply enables LCD display even at low voltages.
   • A or B waveform can be selected by software.
2. Figure 2 is a block diagram of the LCD controller/driver used in this task examples.

![Block Diagram of LCD Controller/Driver](image)

---

**Figure 2** Block Diagram of LCD Controller/Driver (LCD Display with 1/4 Duty)
Functions of the LCD controller/driver are explained in table 1.

<table>
<thead>
<tr>
<th>LCD port control register (LPCR)</th>
<th>Function</th>
<th>LPCR is an 8-bit read/write register which selects the duty cycle, the LCD driver and pin functions. LPCR is initialized to H'00 upon reset.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD control register (LCR)</td>
<td>Function</td>
<td>LCR is an 8-bit read/write register which turns the LCD drive power supply on and off, controls display data, and selects the frame frequency. LCR is initialized to H'80 upon reset.</td>
</tr>
<tr>
<td>LCD control register 2 (LCR2)</td>
<td>Function</td>
<td>LCR2 is an 8-bit read/write register which controls switching between A and B waveforms, selects the driver power supply, controls the step-up constant-voltage (5 V) power supply, and selects the duty cycle for charge/discharge pulses controlling disconnection of the power supply split-resistance from power supply circuit. LCR2 is initialized to H'60 upon reset.</td>
</tr>
<tr>
<td>Segment output pins (SEG₃₂ to SEG₁)</td>
<td>Function</td>
<td>These are pins used for LCD segment driving; all these pins are multiplexed as port pins, and their functions can be selected programmably.</td>
</tr>
<tr>
<td>Common output pins (COM₄ to COM₁)</td>
<td>Function</td>
<td>These are LCD common driving output pins; under static or 1/2-duty driving, they can be configured in parallel.</td>
</tr>
<tr>
<td>Segment external expansion signal pin (CL₃₂)</td>
<td>Function</td>
<td>This is a display data latch clock pin which is multiplexed as SEG₃₂.</td>
</tr>
<tr>
<td>Segment external expansion signal pin (CL₃₁)</td>
<td>Function</td>
<td>This is a display data shift clock pin which is multiplexed as SEG₃₁.</td>
</tr>
<tr>
<td>Segment external expansion signal pin (M)</td>
<td>Function</td>
<td>This is an LCD alternation signal pin which is multiplexed as SEG₃₀.</td>
</tr>
<tr>
<td>Segment external expansion signal pin (DO)</td>
<td>Function</td>
<td>This is a serial display data signal pin which is multiplexed as SEG₃₉.</td>
</tr>
<tr>
<td>LCD power supply pins (V₃ to V₀)</td>
<td>Function</td>
<td>These pins are used when connecting an external bypass capacitor or when using an external power supply circuit.</td>
</tr>
<tr>
<td>LCD RAM</td>
<td>Function</td>
<td>Sets the display data. The relation between the LCD RAM and the display segments differs depending on the duty cycle. After the registers necessary for display have been set, instructions similar to the instructions for normal RAM are used to write data corresponding to the duty, and when the display is turned on, display is started automatically. Word/byte access instructions can be used to set data in the LCD RAM.</td>
</tr>
</tbody>
</table>
4. In this task example, a 16-line 8-character segment LCD is used for display with 1/4 duty driving. Figure 3 is a diagram showing connections for segment signals and common signals of the 16-line 8-character segment LCD used in this task example.

![Figure 3 Connections of Segment Signals and Common Signals of the 16-Line 8-Character Segment LCD Used in this Task Example](image-url)
5. Figure 4 shows the LCD RAM mapping under 1/4 duty driving without segment external expansion.

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEG₂</td>
<td>SEG₂</td>
<td>SEG₂</td>
<td>SEG₂</td>
<td>SEG₁</td>
<td>SEG₁</td>
<td>SEG₁</td>
<td>SEG₁</td>
</tr>
<tr>
<td>SEG₄</td>
<td>SEG₄</td>
<td>SEG₄</td>
<td>SEG₄</td>
<td>SEG₃</td>
<td>SEG₃</td>
<td>SEG₃</td>
<td>SEG₃</td>
</tr>
<tr>
<td>SEG₆</td>
<td>SEG₆</td>
<td>SEG₆</td>
<td>SEG₆</td>
<td>SEG₅</td>
<td>SEG₅</td>
<td>SEG₅</td>
<td>SEG₅</td>
</tr>
<tr>
<td>SEG₈</td>
<td>SEG₈</td>
<td>SEG₈</td>
<td>SEG₈</td>
<td>SEG₇</td>
<td>SEG₇</td>
<td>SEG₇</td>
<td>SEG₇</td>
</tr>
<tr>
<td>SEG₁₀</td>
<td>SEG₁₀</td>
<td>SEG₁₀</td>
<td>SEG₁₀</td>
<td>SEG₉</td>
<td>SEG₉</td>
<td>SEG₉</td>
<td>SEG₉</td>
</tr>
<tr>
<td>SEG₁₂</td>
<td>SEG₁₂</td>
<td>SEG₁₂</td>
<td>SEG₁₂</td>
<td>SEG₁₁</td>
<td>SEG₁₁</td>
<td>SEG₁₁</td>
<td>SEG₁₁</td>
</tr>
<tr>
<td>SEG₁₄</td>
<td>SEG₁₄</td>
<td>SEG₁₄</td>
<td>SEG₁₄</td>
<td>SEG₁₃</td>
<td>SEG₁₃</td>
<td>SEG₁₃</td>
<td>SEG₁₃</td>
</tr>
<tr>
<td>SEG₁₆</td>
<td>SEG₁₆</td>
<td>SEG₁₆</td>
<td>SEG₁₆</td>
<td>SEG₁₅</td>
<td>SEG₁₅</td>
<td>SEG₁₅</td>
<td>SEG₁₅</td>
</tr>
<tr>
<td>SEG₁₈</td>
<td>SEG₁₈</td>
<td>SEG₁₈</td>
<td>SEG₁₈</td>
<td>SEG₁₇</td>
<td>SEG₁₇</td>
<td>SEG₁₇</td>
<td>SEG₁₇</td>
</tr>
<tr>
<td>SEG₂₀</td>
<td>SEG₂₀</td>
<td>SEG₂₀</td>
<td>SEG₂₀</td>
<td>SEG₁₉</td>
<td>SEG₁₉</td>
<td>SEG₁₉</td>
<td>SEG₁₉</td>
</tr>
<tr>
<td>SEG₂₂</td>
<td>SEG₂₂</td>
<td>SEG₂₂</td>
<td>SEG₂₂</td>
<td>SEG₂₁</td>
<td>SEG₂₁</td>
<td>SEG₂₁</td>
<td>SEG₂₁</td>
</tr>
<tr>
<td>SEG₂₄</td>
<td>SEG₂₄</td>
<td>SEG₂₄</td>
<td>SEG₂₄</td>
<td>SEG₂₃</td>
<td>SEG₂₃</td>
<td>SEG₂₃</td>
<td>SEG₂₃</td>
</tr>
<tr>
<td>SEG₂₆</td>
<td>SEG₂₆</td>
<td>SEG₂₆</td>
<td>SEG₂₆</td>
<td>SEG₂₅</td>
<td>SEG₂₅</td>
<td>SEG₂₅</td>
<td>SEG₂₅</td>
</tr>
<tr>
<td>SEG₂₈</td>
<td>SEG₂₈</td>
<td>SEG₂₈</td>
<td>SEG₂₈</td>
<td>SEG₂₇</td>
<td>SEG₂₇</td>
<td>SEG₂₇</td>
<td>SEG₂₇</td>
</tr>
<tr>
<td>SEG₃₀</td>
<td>SEG₃₀</td>
<td>SEG₃₀</td>
<td>SEG₃₀</td>
<td>SEG₂₉</td>
<td>SEG₂₉</td>
<td>SEG₂₉</td>
<td>SEG₂₉</td>
</tr>
<tr>
<td>SEG₃₂</td>
<td>SEG₃₂</td>
<td>SEG₃₂</td>
<td>SEG₃₂</td>
<td>SEG₃₁</td>
<td>SEG₃₁</td>
<td>SEG₃₁</td>
<td>SEG₃₁</td>
</tr>
</tbody>
</table>

Figure 4  LCD RAM Mapping under 1/4 Duty Driving
without Segment External Expansion
6. Figure 5 shows the relation between the 16-line 8-character segment LCD display and LCD RAM settings used in this task example. As shown in the figure, by setting the LCD RAM appropriately, “H8/3867” is displayed on the 16-line 8-character segment LCD.

![Figure 5](relation_between_lcd_display_and_lcd_ram_settings.png)

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Display data for “H”</th>
<th>Display data for “8”</th>
<th>Display data for “/”</th>
<th>Display data for “3”</th>
<th>Display data for “8”</th>
<th>Display data for “6”</th>
<th>Display data for “7”</th>
<th>Display data for “ “</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F740</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F741</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F742</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F743</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F744</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F745</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F746</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F747</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F748</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F749</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F74A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F74B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F74C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F74D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F74E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H’F74F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: ‘ ‘ denotes a blank (nothing displayed)
7. Figure 6 shows the relation between the display of the eighth column from the right in the 16-line 8-character segment LCD and the LCD RAM corresponding to SEG₁ through SEG₄. As indicated in figure 6, when the LCD RAM bits corresponding to 0 through f are set to 1, the LCD regions are lit; when they are cleared to 0, the LCD regions are unlit.

Figure 6  Relation between LCD Lit/Unlit States and Corresponding LCD RAM Settings
8. Table 2 shows examples of SEG, through SEG, display and display data for a 16-line 8-character segment LCD.

Table 2  Display Data Examples

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Display</th>
<th>Address</th>
<th>Display Data</th>
<th>Symbol</th>
<th>Display</th>
<th>Address</th>
<th>Display Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HF740</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td>G</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>HF740</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td>H</td>
<td>HF740</td>
<td>0 0 1 1 0 0 0 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>HF740</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td>I</td>
<td>HF740</td>
<td>0 0 0 0 1 1 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/</td>
<td>HF740</td>
<td>0 0 1 0 0 0 0 0 0 0</td>
<td>J</td>
<td>HF740</td>
<td>0 0 0 1 1 1 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>HF740</td>
<td>0 1 1 1 0 1 0 0 0 0</td>
<td>K</td>
<td>HF740</td>
<td>0 1 1 1 0 0 1 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HF740</td>
<td>0 0 0 0 0 0 0 0 0 0</td>
<td>L</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>HF740</td>
<td>0 0 1 1 0 0 1 0 0 0</td>
<td>M</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>HF740</td>
<td>0 0 0 1 0 0 1 0 0 0</td>
<td>N</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>HF740</td>
<td>0 1 0 0 0 0 1 0 0 0</td>
<td>O</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>HF740</td>
<td>0 1 0 1 0 0 1 0 0 0</td>
<td>P</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 1 0 0</td>
<td>Q</td>
<td>HF740</td>
<td>0 0 0 0 0 0 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>HF740</td>
<td>0 1 0 0 0 0 0 0 0 0</td>
<td>R</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 1 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 1 0 0</td>
<td>S</td>
<td>HF740</td>
<td>0 0 0 1 0 0 0 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>HF740</td>
<td>0 1 0 0 0 0 0 0 0 0</td>
<td>T</td>
<td>HF740</td>
<td>0 0 0 0 1 0 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>HF740</td>
<td>0 1 0 0 0 0 1 0 0 0</td>
<td>U</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>HF740</td>
<td>0 0 0 0 1 1 0 0 0 0</td>
<td>V</td>
<td>HF740</td>
<td>0 0 0 0 1 1 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>HF740</td>
<td>0 0 0 0 1 1 0 0 0 0</td>
<td>W</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>HF740</td>
<td>0 0 0 0 1 1 1 0 0 0</td>
<td>X</td>
<td>HF740</td>
<td>0 0 0 0 1 0 0 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>HF740</td>
<td>0 1 1 1 0 0 0 1 1 0</td>
<td>Y</td>
<td>HF740</td>
<td>0 0 0 0 0 0 1 0 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>HF740</td>
<td>0 1 0 0 0 0 1 0 0 0</td>
<td>Z</td>
<td>HF740</td>
<td>0 0 0 1 0 0 0 0 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Rev. 1.0, 08/99, page 66 of 151
9. Table 3 indicates function allocations in this task example.

**Table 3  Function Allocations**

<table>
<thead>
<tr>
<th>Function</th>
<th>Function Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR</td>
<td>Selects duty cycle, LCD driver, and pin functions.</td>
</tr>
<tr>
<td>LCR</td>
<td>Turns LCD drive power supply on and off, controls display data, and selects frame frequency.</td>
</tr>
<tr>
<td>LCR2</td>
<td>Switches between A and B waveforms, selects drive power supply, controls step-up constant-voltage (5 V) power supply, selects duty cycle for charge/discharge pulses to control disconnection of power supply split-resistance from power supply circuit.</td>
</tr>
<tr>
<td>SEGₐ to SEG₁</td>
<td>Used as segment drivers.</td>
</tr>
<tr>
<td>COMₐ to COM₁</td>
<td>Used as common drivers.</td>
</tr>
<tr>
<td>V₀, V₁</td>
<td>The V₀ and V₁ pins are shorted in order to use the step-up constant-voltage (5 V) power supply as the LCD drive power supply.</td>
</tr>
<tr>
<td>LCD RAM</td>
<td>Sets the LCD display data.</td>
</tr>
</tbody>
</table>
Explanation of Functions Used

1. Hardware settings for LCD display are explained below.

   a. LCD drive power supply settings

      The H8/3867 Series can use either the internal power supply circuit or can use an external power supply circuit as the LCD drive power supply. In addition, either the power supply voltage (V<sub>cc</sub>) or the step-up constant-voltage (5 V) can be selected for the internal power supply circuit.

      When using the internal power supply circuit to drive the LCD, the V<sub>0</sub> and V<sub>1</sub> pins are connected externally, as illustrated in figure 7.

      In this task example, the step-up constant-voltage power supply is used as the LCD drive power supply.

---

**Figure 7  Example of Connection of LCD Power Supply Pins When Using Internal Power Supply Circuit**
b. Contrast control function

A block diagram of the LCD drive power supply circuit appears in figure 8. Either $V_{cc}$ or a 5 V output from the step-up constant-voltage power supply circuit is output to pin $V_0$. When these voltages are used directly to drive the LCD, the $V_0$ and $V_1$ pins should be shorted. By inserting a variable resistance $R$ between the $V_0$ and $V_1$ pins, the voltage applied to the $V_1$ pin can be adjusted, and the LCD panel contrast can be controlled.

![Figure 8 Block Diagram of LCD Drive Power Supply Circuit](image)

c. Step-up constant-voltage (5 V) power supply

The H8/3867 Series has an internal step-up constant-voltage (5 V) power supply, supplying a constant 5 V independent of $V_{cc}$.

By setting SUPS of the LCD control register 2 (LCR2) to 1, the step-up constant-voltage (5 V) power supply is activated, and a constant 5 V is output to the $V_0$ pin. This can be used either with pins $V_0$ and $V_1$ short-circuited, or with a resistance inserted to divide the voltage.

**Note:** The step-up constant-voltage (5 V) power supply must not be used for purposes other than to drive the LCD. In addition, when driving a large panel, the power supply capacity may be insufficient. In such cases, either $V_{cc}$ or an external power supply circuit can be used as the power supply.
2. Software settings for LCD display are explained below.

   a. Duty selection
      DTS1 and DTS0 are used to select from among static, 1/2 duty, 1/3 duty, and 1/4 duty.

   b. Segment driver selection
      SGS3 to SGS0 are used to select the segment drivers to be used.

   c. Frame frequency selection
      By setting CKS3 to CKS0, the frame frequency can be selected. The frame frequency
      should be selected according to the LCD panel.

   d. Selection of A and B waveforms
      LCDAB can be used to select either the A or the B waveform for use as the LCD
      waveform.

   e. Selection of LCD drive power supply
      When using the internal power supply circuit, SUPS can be used to select the power supply
      to be used. When using an external power supply circuit, SUPS is used to select Vcc, and
      PSW should be used to turn off the LCD drive power supply.
3. Figure 9 shows the operation principle of this task example.
Explanation of Software

1. Explanation of Modules

Table 4 explains the modules in this task example.

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Label Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main routine</td>
<td>MAIN</td>
<td>Initializes the stack pointer, LCD RAM, and LCD controller/driver, and enables interrupts.</td>
</tr>
</tbody>
</table>

2. Explanation of Arguments

In this task example, no arguments are used.

3. Explanation of Internal Registers Used

Table 5 gives explanation of the internal registers used in this task example.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR, DTS1, DTS0</td>
<td>LCD port control register (Duty cycle selection 1, 0) Select duty from among static, 1/2 duty, 1/3 duty, and 1/4 duty.  • When DTS1 = 0 and DTS0 = 0, static duty is selected  • When DTS1 = 0 and DTS0 = 1, 1/2 duty is selected  • When DTS1 = 1 and DTS0 = 0, 1/3 duty is selected  • When DTS1 = 1 and DTS0 = 1, 1/4 duty is selected</td>
<td>H'FFC0  Bit 7 and bit 6</td>
<td>DTS1 = 1  DTS0 = 1</td>
</tr>
<tr>
<td>Register Name</td>
<td>Description</td>
<td>RAM Address</td>
<td>Settings</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>LPCR CMX</td>
<td>LCD port control register (Common function selection)</td>
<td>H'FFC0 Bit 5</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether the same waveform is output from several pins in order to increase the common driving capacity, when common pins are not selected for a given duty cycle.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMX = 0, the same waveform is not output from multiple common pins not used at that duty cycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMX = 1, the same waveform is output from multiple common pins not used at that duty cycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPCR SGX</td>
<td>LCD port control register (Expansion signal select)</td>
<td>H'FFC0 Bit 4</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether the SEG&lt;sub&gt;10&lt;/sub&gt;/CL&lt;sub&gt;1&lt;/sub&gt;, SEG&lt;sub&gt;9&lt;/sub&gt;/CL&lt;sub&gt;2&lt;/sub&gt;, SEG&lt;sub&gt;8&lt;/sub&gt;/D0, and SEG&lt;sub&gt;7&lt;/sub&gt;/M pins are used as segment pins (SEG&lt;sub&gt;10&lt;/sub&gt; through SEG&lt;sub&gt;7&lt;/sub&gt;), or as segment external expansion signal pins (CL&lt;sub&gt;1&lt;/sub&gt;, CL&lt;sub&gt;2&lt;/sub&gt;, D0, M).</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, they are used as segment pins (SEG&lt;sub&gt;10&lt;/sub&gt; through SEG&lt;sub&gt;7&lt;/sub&gt;)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 1, they are used as segment external expansion signal pins (CL&lt;sub&gt;1&lt;/sub&gt;, CL&lt;sub&gt;2&lt;/sub&gt;, D0, M)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register Name</td>
<td>Description</td>
<td>RAM Address</td>
<td>Settings</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>LPCR</td>
<td>LCD port control register (Segment driver selection)</td>
<td>H'FFC0 Bit 3 to bit 0</td>
<td>SGS3 = 1 SGS2 = 0 SGS1 = 0 SGS0 = 0</td>
</tr>
</tbody>
</table>

Select the segment driver to be used.

- When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 0, pins SEG32 through SEG1 function as ports
- When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 1, pins SEG32 through SEG1 function as ports
- When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 1 and SGS0 = *, pins SEG32 through SEG1 function as segment drivers and pins SEG32 through SEG1 function as ports
- When SGX = 0, SGS3 = 0, SGS2 = 1, SGS1 = 0 and SGS0 = *, pins SEG32 through SEG1 function as segment drivers and pins SEG32 through SEG1 function as ports
- When SGX = 0, SGS3 = 1, SGS2 = 1, SGS1 = 1 and SGS0 = *, pins SEG32 through SEG1 function as segment drivers
- When SGX = 1, SGS3 = 0, SGS2 = 1, SGS1 = 0 and SGS0 = 0, pins SEG32 through SEG1 function as external expansion pins and pins SEG32 through SEG1 function as ports
- SGX = 1, SGS3 = *, SGS2 = *, SGS1 = * and SGS0 = * cannot be specified

*: Don't care
<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR PSW</td>
<td>LCD control register (LCD drive power supply on/off control) Turns the LCD drive power supply off when LCD display is not used in power-down mode, or when an external power supply is used. When the ACT bit is cleared to 0, or when in standby mode, the LCD drive power supply is turned off regardless of this bit setting. • When PSW = 0, the LCD drive power supply is turned off • When PSW = 1, the LCD drive power supply is turned on</td>
<td>H'FFC1 Bit 6</td>
<td>1</td>
</tr>
<tr>
<td>LCR ACT</td>
<td>LCD control register (Display function activate) Selects whether the LCD controller/driver is to be used or not. By clearing this bit to 0, LCD controller/driver operation is halted. Also, regardless of the value of PSW, the LCD drive power supply is turned off. However, the register contents are maintained. • When ACT = 0, LCD controller/driver operation is halted • When ACT = 1, LCD controller/driver operations</td>
<td>H'FFC1 Bit 5</td>
<td>1</td>
</tr>
<tr>
<td>LCR DISP</td>
<td>LCD control register (Display data control) DISP selects whether the LCD RAM contents or blank data are to be displayed. • When DISP = 0, blank data is displayed • When DISP = 1, LCD RAM data is displayed</td>
<td>H'FFC1 Bit 4</td>
<td>1</td>
</tr>
</tbody>
</table>
Table 5  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR</td>
<td>LCD control register (Frame frequency select 3 to 0)</td>
<td>H'FFC1</td>
<td>CKS3 = 1</td>
</tr>
<tr>
<td></td>
<td>Select the operating clock and the frame frequency.</td>
<td>Bit 3 to bit 0</td>
<td>CKS2 = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CKS1 = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CKS0 = 0</td>
</tr>
<tr>
<td>LCR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- When CKS3 = 0, CKS2 = *, CKS1 = 0 and CKS0 = 0, \(\phi\) is selected as operating clock
- When CKS3 = 0, CKS2 = *, CKS1 = 0 and CKS0 = 1, \(\phi/2\) is selected as operating clock
- When CKS3 = 0, CKS2 = *, CKS1 = 1 and CKS0 = *, \(\phi/4\) is selected as operating clock
- When CKS3 = 1, CKS2 = 0, CKS1 = 0 and CKS0 = 0, \(\phi/2\) is selected as operating clock
- When CKS3 = 1, CKS2 = 0, CKS1 = 1 and CKS0 = 1, \(\phi/4\) is selected as operating clock
- When CKS3 = 1, CKS2 = 0, CKS1 = 1 and CKS0 = 1, \(\phi/8\) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 0 and CKS0 = 0, \(\phi/32\) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 0 and CKS0 = 1, \(\phi/64\) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 1 and CKS0 = 0, \(\phi/128\) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 1 and CKS0 = 1, \(\phi/256\) is selected as operating clock

*: Don't care
### Table 5  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR2</td>
<td>LCD control register 2 (A waveform/B waveform switching control)</td>
<td>H'FFC2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether the A or B waveform is to be used for LCD driving.</td>
<td>Bit 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When LCDAB = 0, the LCD is driven using the A waveform</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When LCDAB = 1, the LCD is driven using the B waveform</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCR2</td>
<td>SUPS (Drive power supply select, step-up constant-voltage (5 V) power supply control)</td>
<td>H'FFC2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>When V_{cc} is selected as the drive power supply, the step-up constant-voltage (5 V) power supply operation is halted; when 5 V is selected as the drive power supply, the step-up constant-voltage (5 V) power supply operates.</td>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SUPS = 0, the drive power supply is V_{cc}, and the step-up constant-voltage (5 V) power supply operation is halted</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SUPS = 1, the drive power supply is 5 V, and the step-up constant-voltage (5 V) power supply operates.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 5  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR2</td>
<td>LCD control register 2 (Charge/discharge pulse duty cycle selection 3 to 0)</td>
<td>H'FFC2</td>
<td>CDS3 = 0</td>
</tr>
<tr>
<td></td>
<td>Select the duty cycle while the power supply split-resistance is connected</td>
<td>Bit 3 to bit 0</td>
<td>CDS2 = 0</td>
</tr>
<tr>
<td></td>
<td>to the power supply circuit.</td>
<td></td>
<td>CDS1 = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CDS0 = 0</td>
</tr>
</tbody>
</table>

- When CDS3 = 0, CDS2 = 0, CDS1 = 0 and CDS0 = 0, the duty cycle is 1
- When CDS3 = 0, CDS2 = 0, CDS1 = 0 and CDS0 = 1, the duty cycle is 1/8
- When CDS3 = 0, CDS2 = 0, CDS1 = 1 and CDS0 = 0, the duty cycle is 2/8
- When CDS3 = 0, CDS2 = 0, CDS1 = 1 and CDS0 = 1, the duty cycle is 3/8
- When CDS3 = 0, CDS2 = 1, CDS1 = 0 and CDS0 = 0, the duty cycle is 4/8
- When CDS3 = 0, CDS2 = 1, CDS1 = 0 and CDS0 = 1, the duty cycle is 5/8
- When CDS3 = 0, CDS2 = 1, CDS1 = 1 and CDS0 = 0, the duty cycle is 6/8
- When CDS3 = 0, CDS2 = 1, CDS1 = 1 and CDS0 = 1, the duty cycle is 0
- When CDS3 = 1, CDS2 = 0, CDS1 = * and CDS0 = *, the duty cycle is 1/16
- When CDS3 = 1, CDS2 = 1, CDS1 = * and CDS0 = *, the duty cycle is 1/32

*: Don’t care

4. Explanation of RAM Usage

In this task example, RAM is not used.
Flowchart

1. Main routine

- **MAIN**
  - **H’FF80 → SP**
    - Initialize the stack pointer.
  - **1 → CCR I-bit**
    - Set the I bit of CCR to 1 to mask an interrupt.
  - **H’00 → H’F740 to H’F74F**
    - Initialize the LCD RAM.
  - **H’C8 → LPCR**
    - Set duty cycle to 1/4, set SEG₃₂ through SEG₁ as segment drivers.
  - Turn on the LCD drive power supply, and set LCD controller/driver operation, LCD RAM data display, clock, and frame frequency.
  - **H’FE → LCR**
  - **H’70 → LCR2**
  - **H’6246 → H’F740**
    - Set LCD drive power supply waveform to A, LCD drive power supply to step-up constant-voltage (5 V) power supply, and charge/discharge pulse duty cycle to 1.
    - Set display data for “H” to LCD RAM.
  - **H’724E → H’F742**
    - Set display data for “8” to LCD RAM.
  - **H’0420 → H’F744**
    - Set display data for “/” to LCD RAM.
  - **H’124E → H’F746**
    - Set display data for “3” to LCD RAM.
  - **H’724E → H’F748**
    - Set display data for “8” to LCD RAM.
  - **H’724A → H’F74A**
    - Set display data for “6” to LCD RAM.
  - **H’400E → H’F74C**
    - Set display data for “7” to LCD RAM.
  - **H’0000 → H’F74E**
    - Set display data for “0” to LCD RAM.
  - **0 → CCR I-bit**
    - Clear the I bit of CCR to 0 to enable an interrupt.
Program Lists

;***********************************************************
;*   H8/3867 Application Note
;*
;*   'Liquid Crystal Display
;*   -1/4 Duty Drive, Internal Driver-'
;*
;*   Function : LCD Controller / Driver
;*
;*   External Clock : 6MHz
;*   Internal Clock : 3MHz
;*   Sub Clock      : 32.768kHz
;***********************************************************

; .cpu 3001
;
;***********************************************************
;* Symbol Defnition
;***********************************************************
;
LPCR .equ h'ffc0 ;LCD Port Control Register
LCR .equ h'ffc1 ;LCD Control Register
LCR2 .equ h'ffc2 ;LCD Control Register 2
;
;***********************************************************
;* Vector Address
;***********************************************************
;
.org h'0000
.data.w MAIN ;No.0 Reset Interrupt(H'0000-H'0001)
;
.org h'0008
.data.w MAIN ;No.4 _IRQ0 Interrupt(H'0008-H'0009)
.data.w MAIN ;No.5 _IRQ1 Interrupt(H'000A-H'000B)
.data.w MAIN ;No.6 _IRQ2 Interrupt(H'000C-H'000D)
.data.w MAIN ;No.7 _IRQ3 Interrupt(H'000E-H'000F)
.data.w MAIN ;No.8 _IRQ4 Interrupt(H'0010-H'0011)

Rev. 1.0, 08/99, page 80 of 151
; No.9 _WKP0—_WKP7 Interrupt (H'0012—H'0013)
; org h'0016
.data.w MAIN ; No.11 Timer A Interrupt (H'0016—H'0017)
.data.w MAIN ; No.12 AEC Interrupt (H'0018—H'0019)
.data.w MAIN ; No.13 Timer C Interrupt (H'001A—H'001B)
.data.w MAIN ; No.14 Timer FL Interrupt (H'001C—H'001D)
.data.w MAIN ; No.15 Timer FH Interrupt (H'001E—H'001F)
.data.w MAIN ; No.16 Timer G Interrupt (H'0020—H'0021)
.data.w MAIN ; No.17 SCI31 Interrupt (H'0022—H'0023)
.data.w MAIN ; No.18 SCI32 Interrupt (H'0024—H'0025)
.data.w MAIN ; No.19 A/D Converter Interrupt (H'0026—H'0028)
.data.w MAIN ; No.20 Direct Transfer Interrupt (H'0028—H'0029)

***********************************************************
* MAIN : Main Routine
***********************************************************

.org h'1000

MAIN: .equ $
mov.w #h'ff80,sp ; Initialize Stack Pointer
orc #h'80,ccr ; Interrupt Disable

sub.b r0l,r0l ; Initialize LCD RAM
mov.w #h'f740,r1
mov.w #h'f750,r2

INIT: mov.b r0l,@r1
addu #1,r1
cmp.w r2,r1
bne INIT

mov.b #h'c8,r0l
mov.b r0l,@LPCR ; Initialize LCD Port Control
mov.b #h'fe,r0l
mov.b r0l,@LCR ; Initialize LCD Control
mov.b #h'70,r0l

Revisão: 1.0, 08/99, página 81 de 151
mov.b  r0l,0LC2 ;Initialize LCD Control 2

mov.w  #h'f740,r1 ;Set LCD RAM Start Address
mov.w  #h'f750,r2 ;Set LCD RAM End Address
mov.w  #h'1500,r3 ;Set LCD Data Address

DISP: mov.w  0r3,r0 ;Load LCD Data
mov.w  r0,r1 ;Store LCD Data to LCD RAM
adds  #2,r3 ;Increment LCD Data Address
adds  #2,r1 ;Increment LCD RAM Address
cmp.w  r2,r1 ;LCD RAM Address = LCD RAM End Address ?
bne  DISP ;No.

andc  #h'7f,ccr ;Interrupt Enable

EXIT: bra  EXIT ;Yes.

;*****************************************************************
;* LCD Data Table
;*****************************************************************

.org  h'1500

.data.w  h'6246 ;"H"
data.w  h'724e ;"8"
data.w  h'0420 ;="/"
data.w  h'124e ;"3"
data.w  h'724e ;"8"
data.w  h'724a ;"6"
data.w  h'400e ;"7"
data.w  h'0000 ;"*"
2.5 LCD Display with Segment External Expansion

<table>
<thead>
<tr>
<th>LCD Display with Segment External Expansion</th>
<th>MCU: H8/3867 Series</th>
<th>Functions Used: LCD Controller/Driver</th>
</tr>
</thead>
</table>

Specifications

1. LCD display is performed using the segment-type LCD controller circuit, LCD driver, and power supply circuit of the H8/3867 Series.

2. By connecting an HD66100 to an H8/3867 Series for segment external expansion, data is displayed on the LCD.

3. Data is displayed on the 16-line 8-character segment LCD with 1/4 duty.

4. As the power supply driving the LCD, Vcc is used.

5. An example of LCD module to the HD66100 connection and an LCD display example for this task example appear in figure 1.

![Figure 1 LCD Module to HD66100 Connection and LCD Display Example](image-url)
Explanation of Functions Used

1. In this task example, the LCD controller/driver is used for LCD display. The features of the LCD controller/driver are described below.

- Display capacity
  a. Duty cycle: static
     - Internal driver: 32 segments
     - Segment external-expansion driver: 256 segments
  b. Duty cycle: 1/2
     - Internal driver: 32 segment
     - Segment external-expansion driver: 128 segments
  c. Duty cycle: 1/3
     - Internal driver: 32 segment
     - Segment external-expansion driver: 64 segments
  d. Duty cycle: 1/4
     - Internal driver: 32 segment
     - Segment external-expansion driver: 64 segments

- LCD RAM capacity: 8 bits \( \times \) 32 bytes (256 bits)
- LCD RAM is word-accessible.
- All segment output pins can be used as port pins in eight-pin units.
- Depending on the duty cycle, the common output pins not used can be used for a common double-buffer (parallel connection).
- Display is possible in all operating modes other than standby mode.
- Frame frequency can be selected from among 11 values.
- A power supply split-resistance is built-in, for supply of LCD driver power.
- Use of module standby mode enables a module to be placed in standby mode independently when not used.
- An internal step-up constant-voltage (5 V) power supply enables LCD display even at low voltages.
- A or B waveform can be selected by software.
2. Figure 2 is a block diagram of the LCD controller/driver used in this task examples.

Figure 2  Block Diagram of LCD Controller/Driver
(LCD Display with Segment External Expansion)
3. Functions of the LCD controller/driver are explained in table 1.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>LCD Controller/Driver Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LCD port control register (LPCR)</strong></td>
<td>Function</td>
</tr>
<tr>
<td></td>
<td><strong>LCD control register (LCR)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>LCD control register 2 (LCR2)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Common output pins (COM₃ to COM₆)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Segment external expansion signal pin (CL₃)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Segment external expansion signal pin (CL₄)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Segment external expansion signal pin (M)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Segment external expansion signal pin (DO)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>LCD power supply pins (V₀ to V₃)</strong></td>
</tr>
<tr>
<td></td>
<td><strong>LCD RAM</strong></td>
</tr>
</tbody>
</table>
4. In this task example, a 16-line 8-character segment LCD is used for display with segment external expansion with 1/4 duty. Figure 3 is a diagram showing connections for segment signals and common signals of the 16-line 8-character segment LCD used in this task example.

Figure 3  Connections of Segment Signals and Common Signals of the 16-Line 8-Character Segment LCD Used in this Task Example
5. Figure 4 shows the LCD RAM mapping under 1/4 duty driving when using segment external expansion.

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>H'F740</td>
<td>SEG2</td>
<td>SEG2</td>
<td>SEG2</td>
<td>SEG2</td>
<td>SEG1</td>
<td>SEG1</td>
<td>SEG1</td>
</tr>
<tr>
<td>H'F741</td>
<td>SEG4</td>
<td>SEG4</td>
<td>SEG4</td>
<td>SEG4</td>
<td>SEG3</td>
<td>SEG3</td>
<td>SEG3</td>
</tr>
<tr>
<td>H'F742</td>
<td>SEG8</td>
<td>SEG8</td>
<td>SEG8</td>
<td>SEG8</td>
<td>SEG5</td>
<td>SEG5</td>
<td>SEG5</td>
</tr>
<tr>
<td>H'F743</td>
<td>SEG9</td>
<td>SEG9</td>
<td>SEG9</td>
<td>SEG9</td>
<td>SEG7</td>
<td>SEG7</td>
<td>SEG7</td>
</tr>
<tr>
<td>H'F744</td>
<td>SEG10</td>
<td>SEG10</td>
<td>SEG10</td>
<td>SEG10</td>
<td>SEG9</td>
<td>SEG9</td>
<td>SEG9</td>
</tr>
<tr>
<td>H'F745</td>
<td>SEG12</td>
<td>SEG12</td>
<td>SEG12</td>
<td>SEG12</td>
<td>SEG11</td>
<td>SEG11</td>
<td>SEG11</td>
</tr>
<tr>
<td>H'F746</td>
<td>SEG14</td>
<td>SEG14</td>
<td>SEG14</td>
<td>SEG14</td>
<td>SEG13</td>
<td>SEG13</td>
<td>SEG13</td>
</tr>
<tr>
<td>H'F747</td>
<td>SEG16</td>
<td>SEG16</td>
<td>SEG16</td>
<td>SEG16</td>
<td>SEG15</td>
<td>SEG15</td>
<td>SEG15</td>
</tr>
<tr>
<td>H'F748</td>
<td>SEG18</td>
<td>SEG18</td>
<td>SEG18</td>
<td>SEG18</td>
<td>SEG17</td>
<td>SEG17</td>
<td>SEG17</td>
</tr>
<tr>
<td>H'F749</td>
<td>SEG20</td>
<td>SEG20</td>
<td>SEG20</td>
<td>SEG20</td>
<td>SEG19</td>
<td>SEG19</td>
<td>SEG19</td>
</tr>
<tr>
<td>H'F74A</td>
<td>SEG22</td>
<td>SEG22</td>
<td>SEG22</td>
<td>SEG22</td>
<td>SEG21</td>
<td>SEG21</td>
<td>SEG21</td>
</tr>
<tr>
<td>H'F74B</td>
<td>SEG24</td>
<td>SEG24</td>
<td>SEG24</td>
<td>SEG24</td>
<td>SEG23</td>
<td>SEG23</td>
<td>SEG23</td>
</tr>
<tr>
<td>H'F74C</td>
<td>SEG26</td>
<td>SEG26</td>
<td>SEG26</td>
<td>SEG26</td>
<td>SEG25</td>
<td>SEG25</td>
<td>SEG25</td>
</tr>
<tr>
<td>H'F74D</td>
<td>SEG28</td>
<td>SEG28</td>
<td>SEG28</td>
<td>SEG28</td>
<td>SEG27</td>
<td>SEG27</td>
<td>SEG27</td>
</tr>
<tr>
<td>H'F74E</td>
<td>SEG30</td>
<td>SEG30</td>
<td>SEG30</td>
<td>SEG30</td>
<td>SEG29</td>
<td>SEG29</td>
<td>SEG29</td>
</tr>
<tr>
<td>H'F74F</td>
<td>SEG32</td>
<td>SEG32</td>
<td>SEG32</td>
<td>SEG32</td>
<td>SEG31</td>
<td>SEG31</td>
<td>SEG31</td>
</tr>
<tr>
<td>H'F75F</td>
<td>SEG64</td>
<td>SEG64</td>
<td>SEG64</td>
<td>SEG64</td>
<td>SEG63</td>
<td>SEG63</td>
<td>SEG63</td>
</tr>
</tbody>
</table>

Figure 4  LCD RAM Mapping under 1/4 Duty Driving with Segment External Expansion
6. Figure 5 shows the relation between the 16-line 8-character segment LCD display and LCD RAM settings used in this task example. As shown in the figure, by setting the LCD RAM appropriately, “H8/3867” is displayed on the 16-line 8-character segment LCD.

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>H'F740</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>H'F741</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>H'F742</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>H'F743</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>H'F744</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>H'F745</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>H'F746</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>H'F747</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>H'F748</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>H'F749</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>H'F74A</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>H'F74B</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>H'F74C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>H'F74D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>H'F74E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>H'F74F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: "·" denotes a blank (nothing displayed)
7. Figure 6 shows the relation between the display of the eighth column from the right in the 16-line 8-character segment LCD and the LCD RAM corresponding to SEG_1 through SEG_4. As indicated in figure 6, when the LCD RAM bits corresponding to 0 through f are set to 1, the LCD regions are lit; when they are cleared to 0, the LCD regions are unlit.
8. Table 2 shows examples of SEG₁ through SEG₄ display and display data for a 16-line 8-character segment LCD.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Display</th>
<th>Address</th>
<th>Display Data</th>
<th>Symbol</th>
<th>Display</th>
<th>Address</th>
<th>Display Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>H'F740</td>
<td>0</td>
<td>0 0 0 0 0 0 0</td>
<td>G</td>
<td>H'F740</td>
<td>0 1 1 1 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>H'F741</td>
<td>0</td>
<td>0 0 0 0 0 0 0</td>
<td>H</td>
<td>H'F741</td>
<td>0 0 0 0 1 0 1 0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>H'F740</td>
<td>0</td>
<td>0 0 0 0 1 1 1</td>
<td>I</td>
<td>H'F740</td>
<td>0 1 1 0 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>H'F741</td>
<td>0</td>
<td>1 1 1 1 0 0 0</td>
<td>J</td>
<td>H'F741</td>
<td>0 1 1 1 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>H'F740</td>
<td>0</td>
<td>1 1 1 1 0 1 0</td>
<td>K</td>
<td>H'F740</td>
<td>0 1 1 1 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>H'F741</td>
<td>0</td>
<td>1 1 1 0 0 1 0</td>
<td>L</td>
<td>H'F741</td>
<td>0 1 0 0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>H'F740</td>
<td>0</td>
<td>0 0 0 0 0 0 0</td>
<td>M</td>
<td>H'F740</td>
<td>0 1 1 1 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>H'F741</td>
<td>0</td>
<td>0 0 0 0 0 1 1</td>
<td>N</td>
<td>H'F741</td>
<td>0 1 1 1 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>H'F740</td>
<td>0</td>
<td>0 0 0 0 1 1 1</td>
<td>O</td>
<td>H'F740</td>
<td>0 1 1 1 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>H'F741</td>
<td>0</td>
<td>1 1 1 0 0 0 0</td>
<td>P</td>
<td>H'F741</td>
<td>0 1 0 0 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>H'F740</td>
<td>0</td>
<td>1 1 1 1 0 1 0</td>
<td>Q</td>
<td>H'F740</td>
<td>0 1 1 1 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>H'F741</td>
<td>0</td>
<td>0 0 0 0 0 0 0</td>
<td>R</td>
<td>H'F741</td>
<td>0 1 0 0 0 0 1 1</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>H'F740</td>
<td>0</td>
<td>0 0 0 0 1 1 1</td>
<td>S</td>
<td>H'F740</td>
<td>0 1 0 0 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>H'F741</td>
<td>0</td>
<td>1 1 1 0 0 0 0</td>
<td>T</td>
<td>H'F471</td>
<td>0 1 0 0 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>H'F740</td>
<td>0</td>
<td>1 1 1 0 0 0 0</td>
<td>U</td>
<td>H'F740</td>
<td>0 1 1 1 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>H'F741</td>
<td>0</td>
<td>1 1 1 0 0 0 0</td>
<td>V</td>
<td>H'F741</td>
<td>0 1 1 0 0 0 0 0</td>
<td></td>
</tr>
</tbody>
</table>

---

Rev. 1.0, 08/99, page 91 of 151
9. Table 3 indicates function allocations in this task example.

<table>
<thead>
<tr>
<th>Function</th>
<th>Function Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR</td>
<td>Selects duty cycle, LCD driver, and pin functions.</td>
</tr>
<tr>
<td>LCR</td>
<td>Turns LCD drive power supply on and off, controls display data, and selects frame frequency.</td>
</tr>
<tr>
<td>LCR2</td>
<td>Switches between A and B waveforms, selects drive power supply, controls step-up constant-voltage (5 V) power supply, selects duty cycle for charge/discharge pulses to control disconnection of power supply split-resistance from power supply circuit.</td>
</tr>
<tr>
<td>COM₁ to COM₅</td>
<td>Used as common drivers.</td>
</tr>
<tr>
<td>V₀ to V₃</td>
<td>Used as the LCD drive power supply pins to be connected to the HD66100.</td>
</tr>
<tr>
<td>CL₁</td>
<td>Functions as a display data latch clock output to be connected to the HD66100.</td>
</tr>
<tr>
<td>CL₂</td>
<td>Functions as a display data latch clock output to be connected to the HD66100.</td>
</tr>
<tr>
<td>M</td>
<td>Functions as an LCD alternation signal output to be connected to the HD66100.</td>
</tr>
<tr>
<td>DO</td>
<td>Functions as a serial display data output to be connected to the HD66100.</td>
</tr>
<tr>
<td>LCD RAM</td>
<td>Sets the LCD display data.</td>
</tr>
</tbody>
</table>
Explanation of Functions Used

1. Hardware settings for LCD display are explained below.

   a. LCD drive power supply settings
      The H8/3867 Series can use either the internal power supply circuit or an external power supply circuit as the LCD drive power supply. In addition, either the power supply voltage (V_{cc}) or the step-up constant-voltage (5 V) can be selected for the internal power supply circuit.

      When using the internal power supply circuit to drive the LCD, the V_o and V_i pins are connected externally, as illustrated in figure 7.

      In this task example, the step-up constant-voltage power supply is used as the LCD drive power supply.

![Figure 7](image)

   Figure 7  Example of Connection of LCD Power Supply Pins
   When Using Internal Power Supply Circuit
b. Contrast control function

A block diagram of the LCD drive power supply circuit appears in figure 8. Either $V_{CC}$ or a 5 V output from the step-up constant-voltage power supply circuit is output to pin $V_0$. When these voltages are used directly to drive the LCD, the $V_0$ and $V_1$ pins should be shorted. By inserting a variable resistance $R$ between the $V_0$ and $V_1$ pins, the voltage applied to the $V_1$ pin can be adjusted, and the LCD panel contrast can be controlled.

Figure 8  Block Diagram of LCD Drive Power Supply Circuit
2. Software settings for LCD display are explained below.

a. Duty selection
   DTS1 and DTS0 are used to select from among static, 1/2 duty, 1/3 duty, and 1/4 duty.

b. Segment driver selection
   SGS3 to SGS0 are used to select the segment drivers to be used.

c. Frame frequency selection
   By setting CKS3 to CKS0, the frame frequency can be selected. The frame frequency
   should be selected according to the LCD panel.

d. Selection of A and B waveforms
   LCDAB can be used to select either the A or the B waveform for use as the LCD
   waveform.

e. Selection of LCD drive power supply
   When using the internal power supply circuit, SUPS can be used to select the power supply
   to be used. When using an external power supply circuit, SUPS is used to select V_{cc} and
   PSW should be used to turn off the LCD drive power supply.
3. Figure 9 shows the operation principle of this task example.
Explanation of Software

1. Explanation of Modules

    Table 4 explains the modules in this task example.

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Label Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main routine</td>
<td>MAIN</td>
<td>Initializes the stack pointer, LCD RAM, and LCD controller/driver, and enables interrupts.</td>
</tr>
</tbody>
</table>

2. Explanation of Arguments

    In this task example, no arguments are used.

3. Explanation of Internal Registers Used

    Table 5 gives explanation of the internal registers used in this task example.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR, DTS1, DTS0</td>
<td>LCD port control register (Duty cycle selection 1, 0)</td>
<td>H'FFC0</td>
<td>DTS1 = 1</td>
</tr>
<tr>
<td></td>
<td>Select duty from among static, 1/2 duty, 1/3 duty, and 1/4 duty.</td>
<td>Bit 7 and bit 6</td>
<td>DTS0 = 1</td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 0 and DTS0 = 0, static duty is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 0 and DTS0 = 1, 1/2 duty is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 1 and DTS0 = 0, 1/3 duty is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTS1 = 1 and DTS0 = 1, 1/4 duty is selected</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 5  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR CMX</td>
<td>LCD port control register (Common function selection)</td>
<td>H'FFC0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether the same waveform is output from several pins in order to increase the common driving capacity, when common pins are not selected for a given duty cycle.</td>
<td>Bit 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMX = 0, the same waveform is not output from multiple common pins not used at that duty cycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMX = 1, the same waveform is output from multiple common pins not used at that duty cycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPCR SGX</td>
<td>LCD port control register (Expansion signal select)</td>
<td>H'FFC0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Selects whether the SEG₃₂/CL₁, SEG₃₁/CL₂, SEG₃₀/D₀, and SEG₃₉/M pins are used as segment pins (SEG₃₂ through SEG₃₉), or as segment external expansion signal pins (CL₁, CL₂, D₀, M).</td>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 0, they are used as segment pins (SEG₃₂ through SEG₃₉)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When SGX = 1, they are used as segment external expansion signal pins (CL₁, CL₂, D₀, M)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 5  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPCR SGS3 to SGS0</td>
<td>LCD port control register (Segment driver selection)</td>
<td>H'FFC0</td>
<td>SGS3 = 0, SGS2 = 0, SGS1 = 0, SGS0 = 0</td>
</tr>
</tbody>
</table>

Select the segment driver to be used.

- When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 0, pins SEG_0 through SEG_7 function as ports
- When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 1, pins SEG_0 through SEG_7 function as ports
- When SGX = 0, SGS3 = 0, SGS2 = 0, SGS1 = 1 and SGS0 = *, pins SEG_8 through SEG_25 function as segment drivers and pins SEG_0 through SEG_7 function as ports
- When SGX = 0, SGS3 = 0, SGS2 = 1, SGS1 = 0 and SGS0 = *, pins SEG_0 through SEG_17 function as segment drivers and pins SEG_8 through SEG_16 function as ports
- When SGX = 0, SGS3 = 0, SGS2 = 1, SGS1 = 1 and SGS0 = *, pins SEG_0 through SEG_9 function as segment drivers and pins SEG_8 through SEG_1 function as ports
- When SGX = 0, SGS3 = 1, SGS2 = *, SGS1 = * and SGS0 = *, pins SEG_0 through SEG_1 function as segment drivers
- When SGX = 1, SGS3 = 0, SGS2 = 0, SGS1 = 0 and SGS0 = 0, pins SEG_0 through SEG_29 function as external expansion pins and pins SEG_28 through SEG_1 function as ports
- SGX = 1, SGS3 = *, SGS2 = *, SGS1 = * and SGS0 = * cannot be specified

*: Don't care
### Table 5  
**Explanation of Internal Registers Used (cont)**

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
</table>
| LCR PSW       | LCD control register (LCD drive power supply on/off control) Turns the LCD drive power supply off when LCD display is not used in power-down mode, or when an external power supply is used. When the ACT bit is cleared to 0, or when in standby mode, the LCD drive power supply is turned off regardless of this bit setting.  
- When PSW = 0, the LCD drive power supply is turned off  
- When PSW = 1, the LCD drive power supply is turned on | H'FFC1 Bit 6 | 1 |
| LCR ACT       | LCD control register (Display function activate) Selects whether the LCD controller/driver is to be used or not. By clearing this bit to 0, LCD controller/driver operation is halted. Also, regardless of the value of PSW, the LCD drive power supply is turned off. However, the register contents are maintained.  
- When ACT = 0, LCD controller/driver operation is halted  
- When ACT = 1, LCD controller/driver operations | H'FFC1 Bit 5 | 1 |
| LCR DISP      | LCD control register (Display data control) DISP selects whether the LCD RAM contents or blank data are to be displayed.  
- When DISP = 0, blank data is displayed  
- When DISP = 1, LCD RAM data is displayed | H'FFC1 Bit 4 | 1 |
<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
</table>
| LCR CKS3 to CKS0 | LCD control register (Frame frequency select 3 to 0) Select the operating clock and the frame frequency. | H'FFC1 | CKS3 = 1
Bit 3 to bit 0 | CKS2 = 1
CKS1 = 1
CKS0 = 0 |
- When CKS3 = 0, CKS2 = *, CKS1 = 0 and CKS0 = 0, \( \phi \) is selected as operating clock
- When CKS3 = 0, CKS2 = *, CKS1 = 0 and CKS0 = 1, \( \phi/2 \) is selected as operating clock
- When CKS3 = 0, CKS2 = *, CKS1 = 1 and CKS0 = *, \( \phi/4 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 0, CKS1 = 1 and CKS0 = 0, \( \phi/2 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 0, CKS1 = 0 and CKS0 = 1, \( \phi/4 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 0, CKS1 = 0 and CKS0 = 1, \( \phi/8 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 0, CKS1 = 1 and CKS0 = 1, \( \phi/16 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 0 and CKS0 = 0, \( \phi/32 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 0 and CKS0 = 1, \( \phi/64 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 0 and CKS0 = 1, \( \phi/128 \) is selected as operating clock
- When CKS3 = 1, CKS2 = 1, CKS1 = 1 and CKS0 = 1, \( \phi/256 \) is selected as operating clock

*: Don't care
<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR2</td>
<td>LCDAB</td>
<td>H'FFC2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>LCD control register 2 (A waveform/B waveform switching control) Selects whether the A or B waveform is to be used for LCD driving.</td>
<td>Bit 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- When LCDAB = 0, the LCD is driven using the A waveform
- When LCDAB = 1, the LCD is driven using the B waveform

<table>
<thead>
<tr>
<th>LCR2</th>
<th>SUPS</th>
<th>LCD control register 2 (Drive power supply select, step-up constant-voltage (5 V) power supply control) When ( V_{cc} ) is selected as the drive power supply, the step-up constant-voltage (5 V) power supply operation is halted; when 5 V is selected as the drive power supply, the step-up constant-voltage (5 V) power supply operates.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Bit 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- When SUPS = 0, the drive power supply is \( V_{cc} \), and the step-up constant-voltage (5 V) power supply operation is halted
- When SUPS = 1, the drive power supply is 5 V, and the step-up constant-voltage (5 V) power supply operates


<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCR2</td>
<td>LCD control register 2</td>
<td>H'FFC2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Charge/discharge pulse duty cycle selection 3 to 0)</td>
<td>Bit 3 to 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Select the duty cycle while the power supply split-resistance is connected to</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>the power supply circuit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 0, CDS1 = 0 and CDS0 = 0, the duty cycle is 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 0, CDS1 = 0 and CDS0 = 1, the duty cycle is 1/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 0, CDS1 = 1 and CDS0 = 0, the duty cycle is 2/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 0, CDS1 = 1 and CDS0 = 1, the duty cycle is 3/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 1, CDS1 = 0 and CDS0 = 0, the duty cycle is 4/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 1, CDS1 = 0 and CDS0 = 1, the duty cycle is 5/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 1, CDS1 = 1 and CDS0 = 0, the duty cycle is 6/8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 0, CDS2 = 1, CDS1 = 1 and CDS0 = 1, the duty cycle is 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 1, CDS2 = 0, CDS1 = * and CDS0 = *, the duty cycle is 1/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CDS3 = 1, CDS2 = 1, CDS1 = * and CDS0 = *, the duty cycle is 1/32</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*: Don’t care

4. Explanation of RAM Usage

In this task example, RAM is not used.
Flowchart

1. Main routine

```
MAIN

H'FF80 → SP  ----------------- Initialize the stack pointer.

1 → CCR I-bit  ----------------- Set the I bit of CCR to 1 to mask an interrupt.

H'00 → H'7F40 to H'F74F  ----------------- Initialize the LCD RAM.

H'D0 → LPCR  ----------------- Set duty cycle to 1/4, and set SEG32/CL1 through SEG29/M as segment drivers.

H'FE → LCR  ----------------- Turn on LCD drive power supply, and set LCD controller/driver operation, LCD RAM data display, clock, and frame frequency.

H'60 → LCR2  ----------------- Set LCD drive power supply waveform to A, LCD drive power supply to VCC, and charge/discharge pulse duty cycle to 1.

H'6246 → H'F740  ----------------- Set display data for “H” to LCD RAM.

H'724E → H'F742  ----------------- Set display data for “8” to LCD RAM.

H'0420 → H'F744  ----------------- Set display data for “/” to LCD RAM.

H'124E → H'F746  ----------------- Set display data for “3” to LCD RAM.

H'724E → H'F748  ----------------- Set display data for “8” to LCD RAM.

H'724A → H'F74A  ----------------- Set display data for “6” to LCD RAM.

H'400E → H'F74C  ----------------- Set display data for “7” to LCD RAM.

H'0000 → H'F74E  ----------------- Set display data for “0” to LCD RAM.

0 → CCR I-bit  ----------------- Clear the I bit of CCR to 0 to enable an interrupt.
```

Rev. 1.0, 08/99, page 104 of 151
Program Lists

;****************************************************************************
;*    H8/3867 Application Note
;*
;*     'Liquid Crystal Display
;*      -Using Segment External Expansion,
;*     Using HD66100, 1/4 Duty Drive'
;*
;*      Function : LCD Controller / Driver
;*
;*     External Clock : 6MHz
;*     Internal Clock : 3MHz
;*     Sub Clock      : 32.768kHz
;*****************************************************************************

.cpu 300

;*****************************************************************************
;* Symbol Definition
;*****************************************************************************

.LPCR  .equ h'ffc0   ;LCD Port Control Register
.LCR   .equ h'ffc1   ;LCD Control Register
.LCR2  .equ h'ffc2   ;LCD Control Register 2

;*****************************************************************************
;* Vector Address
;*****************************************************************************

.org h'0000
.data.w MAIN   ;No.0 Reset Interrupt(H'0000-H'0001)

.org h'0008
.data.w MAIN   ;No.4 _IRQ0 Interrupt(H'0008-H'0009)
.data.w MAIN   ;No.5 _IRQ1 Interrupt(H'000A-H'000B)
.data.w MAIN   ;No.6 _IRQ2 Interrupt(H'000C-H'000D)
.data.w MAIN   ;No.7 _IRQ3 Interrupt(H'000E-H'000F)
.data.w MAIN ;No.8 _IRQ4 Interrupt(H'0010-H'0011)
.data.w MAIN ;No.9 _WKPO--WKP7 Interrupt(H'0012-H'0013)

; .org  h'0016
.data.w MAIN ;No.11 Timer A Interrupt(H'0016-H'0017)
.data.w MAIN ;No.12 AEC Interrupt(H'0018-H'0019)
.data.w MAIN ;No.13 Timer C Interrupt(H'001A-H'001B)
.data.w MAIN ;No.14 Timer FL Interrupt(H'001C-H'001D)
.data.w MAIN ;No.15 Timer FH Interrupt(H'001E-H'001F)
.data.w MAIN ;No.16 Timer G Interrupt(H'0020-H'0021)
.data.w MAIN ;No.17 SCI31 Interrupt(H'0022-H'0023)
.data.w MAIN ;No.18 SCI32 Interrupt(H'0024-H'0025)
.data.w MAIN ;No.19 A/D Converter Interrupt(H'0026-H'0028)
.data.w MAIN ;No.20 Direct Transfer Interrupt(H'0028-H'0029)

;***********************************************************
;* MAIN : Main Routine
;***********************************************************
.org  h'1000

; MAIN: .equ $
   mov.w #h'ff80,sp ;Initialize Stack Pointer
   orc #h'80,ccr ;Interrupt Disable
   sub.b r0l,r0l ;Initialize LCD RAM
   mov.w #h'f740,r1
   mov.w #h'f750,r2

INIT:  mov.b r0l,0rl
   adds $1,rl
   cmp.w r2,rl
   bne INIT

   mov.b #h'd0,r0l ;Initialize LCD Port Control
   mov.w r0l,0LPCR
   mov.b #h'fe,r0l ;Initialize LCD Control
   mov.w r0l,0LCR
   mov.b #h'60,r0l ;Initialize LCD Control 2
mov.b r0l,@LCR2
;
mov.w #h'f740,r1 ;Set LCD RAM Start Address
mov.w #h'f750,r2 ;Set LCD RAM End Address
mov.w #h'1500,r3 ;Set LCD Data Address

DISP: mov.w 0,r3,r0 ;Load LCD Data
mov.w r0,@r1 ;Store LCD Data to LCD RAM
adds #2,r3 ;Increment LCD Data Address
adds #2,r1 ;Increment LCD RAM Address
cmp.w r2,r1 ;LCD RAM Address = LCD RAM End Address ?
bne DISP ;No.
;
EXIT: bra EXIT ;Yes.
;
;***********************************************************
;* LCD Data Table
;***********************************************************
.org h'1500
;
.data.w h'6246 ;"H"
data.w h'724e ;"8"
data.w h'0420 ;"/"
data.w h'124e ;"3"
data.w h'724e ;"8"
data.w h'724a ;"6"
data.w h'400e ;"7"
data.w h'0000 ;" "
;
.end
2.6 Oscillation Stabilization Time Settings

Oscillation Stabilization Time Settings

<table>
<thead>
<tr>
<th>Oscillation Stabilization Time Settings</th>
<th>MCU: H8/3867 Series</th>
<th>Functions Used: Power-Down Mode</th>
</tr>
</thead>
</table>

Settings

The time for which the CPU and peripheral functions must wait until the clock stabilizes when, by means of specific interrupts, standby or watch mode is canceled and there is a transition to active mode, is set. This standby time must be set to be longer than the time for oscillation stabilization, in accordance with the operating frequency.

Setting the standby time

The standby time is set by setting the standby timer selects 2 to 0 bits (STS2 to STS0) of the system control register 1 (SYSCR1).

Explanation of the STS2 to STS0 bits

Table 1 explains the STS2 to STS0 bits of the SYSCR1 register.

<table>
<thead>
<tr>
<th>STS2</th>
<th>STS1</th>
<th>STS0</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Standby time = 8,192 states (Initial value)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Standby time = 16,384 states</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Standby time = 32,768 states</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Standby time = 65,536 states</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Standby time = 131,072 states</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Standby time = 2 states (External clock mode)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Standby time = 8 states</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Standby time = 16 states</td>
</tr>
</tbody>
</table>

Note: When an external clock signal is to be input, the standby timer select pins should be set to external clock mode prior to execution of the mode transition. When an external clock is not used, the external clock input mode should not be set.
Operating frequency and oscillation stabilization time when a crystal oscillator is used

Table 2 shows the standby times resulting for different operating frequencies and STS2 to STS0 settings when a crystal oscillator is used. STS2 to STS0 are set so that the standby time is longer than the time required for oscillation stabilization.

Table 2 Operating Frequency and Oscillation Stabilization Times for Crystal Oscillators

<table>
<thead>
<tr>
<th>Standby Timer Select Bit Settings</th>
<th>Operating Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 MHz</td>
</tr>
<tr>
<td>STS2 STS1 STS0 Standby Time</td>
<td></td>
</tr>
<tr>
<td>0 0 0 8,192 states</td>
<td>4.1</td>
</tr>
<tr>
<td>1 0 16,384 states</td>
<td>8.2</td>
</tr>
<tr>
<td>1 0 32,768 states</td>
<td>16.4</td>
</tr>
<tr>
<td>1 1 65,536 states</td>
<td>32.8</td>
</tr>
<tr>
<td>1 0 131,072 states</td>
<td>65.5</td>
</tr>
<tr>
<td>1 1 2 states (Use prohibited)</td>
<td>0.001</td>
</tr>
<tr>
<td>1 0 8 states</td>
<td>0.004</td>
</tr>
<tr>
<td>1 1 16 states</td>
<td>0.008</td>
</tr>
</tbody>
</table>

Unit: ms

When an external clock is used

It is recommended that the circuit be used with STS2 = 1, STS1 = 0, and STS0 = 1. Use at other settings is also possible, but operation may begin before the standby time has completed.
Oscillation stabilization times

Table 3 shows the AC characteristics of oscillation stabilization times.

### Table 3  AC Characteristics of Oscillation Stabilization Time

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Applicable Pins</th>
<th>Test Conditions</th>
<th>Values</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscillation stabilization time</td>
<td>( t_{oc} )</td>
<td>OSC(_1), OSC(_2)</td>
<td>( V_{cc} = 2.2 \text{ V to } 5.5 \text{ V} ) (as shown in figure 1)</td>
<td>Min Typ Max Unit</td>
<td>Figure 1*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( V_{cc} = 2.2 \text{ V to } 5.5 \text{ V} ) (as shown in figure 1)</td>
<td>0.1 8 ms</td>
<td>Figure 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Other than the above</td>
<td>— — 50 ms</td>
<td>Figure 1</td>
</tr>
</tbody>
</table>

\( V_{cc} = 1.8 \text{ to } 5.5 \text{ V}, AV_{cc} = 1.8 \text{ to } 5.5 \text{ V}, V_{ss} = AV_{ss} = 0.0 \text{ V}, T_a = -20 \text{ to } +75^\circ \text{C}, \) including subactive mode

Note: *: Internal power supply step-down circuit not used.

- **Crystal oscillator parameters**
  - Frequency (MHz): 1, 4.193
  - \( R_S \) (max): 40 \( \Omega \), 100 \( \Omega \)
  - \( C_0 \) (max): 3.5 pF, 16 pF

- **Ceramic oscillator parameters**
  - Frequency (MHz): 0.4, 4
  - \( R_S \) (max): 8.6 \( \Omega \), 8.8 \( \Omega \)
  - \( C_0 \) (max): 326 pF, 36 pF

**Figure 1  Oscillator Equivalent Circuit**
Example of oscillation stabilization time settings

1. Functions
   A transition from active (high-speed) to watch mode is induced, and after 250 ms watch mode is canceled by a timer A interrupt, with a transition back to active (high-speed) mode. In making the transition from watch mode to active (high-speed) mode, the time for the CPU and peripheral functions to wait until the clock stabilizes is set to eight states.

2. Notes
   In these settings, when the watch mode is canceled by a timer A interrupt, part of the timer A interrupt processing includes prohibition of timer A interrupt requests. Hence when there is a transition from active (high-speed) mode to watch mode, and watch mode is then canceled by a timer A interrupt with a transition to active (high-speed) mode, processing is completed.

3. Watch mode
   a. Transition to watch mode
      When, in active mode or subactive mode, the software standby bit (SSBY) of the system control register 1 (SYSCR1) is 1 and the internal clock selector 3 bit (TMA3) of the timer mode register A (TMA) is 1, if a sleep instruction is executed, there is a transition to watch mode. In watch mode, operation of all built-in peripheral functions other than timer A, timer F, timer G, the asynchronous event counter, and the LCD (operation/halted selectable), is halted. So long as the standard voltage is supplied, the CPU, the internal registers for part of the built-in peripheral functions, and internal RAM are maintained, and the I/O ports are held at their states prior to transition.

   b. Watch mode cancellation
      Watch mode is canceled by an interrupt (IRQ, WKP, to WKP, timer A, timer F, timer G) or by RES pin input.
      In cancellation by an interrupt, when the interrupt occurs the watch mode is canceled, and if the low-speed on-flag (LSON) of SYSCR1 is 0 and the medium-speed on-flag (MSON) of the system control register 2 (SYSCR2) is 0, there is a transition to active (high-speed) mode. If LSON = 0 and MSON = 1, there is a transition to active (medium-speed) mode, and if LSON = 1 there is a transition to subactive mode. On transitions to active modes, after the time set by the STS2 to STS0 bits of SYSCR1 has elapsed, the stabilized clock pulse is supplied to the entire LSI, and interrupt exception processing begins. When the I bit of CCR is 1 or when acceptance of the relevant interrupt by the interrupt enable register is disabled, watch mode is not canceled.
      In cancellation by input to the RES pin, if the RES pin is forced low, system clock oscillation is started. After the time for oscillation stabilization has elapsed, if the RES pin is forced high, the CPU initiates reset exception processing. The system clock is supplied to the entire LSI at the time, system clock oscillation is started. The RES pin should always be held low until the system clock oscillation stabilizes.
## Flowchart

1. **Main routine**

   - Set oscillation stabilization time
     - H'FF80 → SP
       - Initialize the stack pointer.
     - 1 → CCR I-bit
       - Set the I bit of CCR to 1 to mask an interrupt.
     - H'E7 → SYSCR1
       - Set the SSBY bit to 1, and set the standby time for oscillation stabilization to 8 states by setting STS2 = 1, STS1 = 1, and STS0 = 0, to make a transition to watch mode.
     - H'F0 → SYSCR2
     - 0 → IRRTA
       - Clear the timer A interrupt request flag to 0.
     - H'80 → IENR1
       - Enable timer A interrupt requests.
     - H'FF → TMA
       - Reset the prescaler W (PSW) and the timer counter A (TCA).
     - H'1A → TMA
       - Specify PSW as the TCA input clock source, and set the TCA overflow frequency to 0.25 sec.
     - 0 → CCR I-bit
       - Clear the I bit of CCR to 0 to enable an interrupt.
     - Execute a SLEEP instruction
       - Execute a SLEEP instruction to make a transition from active (high-speed) mode to watch mode.
2. Timer A interrupt processing routine

- Timer A interrupt
  - $0 \rightarrow \text{IRRTA}$
    - Clear the timer A interrupt request flag.
  - $\text{H'00} \rightarrow \text{IENR1}$
    - Disable timer A interrupt requests.
- RTE
Program Lists

;***********************************************************
;*   H8/3867 Application Note
;*
;* 'Oscillator Settling Time - 8 States'
;*
;* Function: Oscillator Settling Time
;*
;* External Clock: 6MHz
;* Internal Clock: 3MHz
;* Sub Clock: 32.768kHz
;***********************************************************
;
.cpu 3001
;
;***********************************************************
;* Symbol Definition
;***********************************************************
;
TMA .equ h'ffb0 ;Timer Mode Register A
SYSCR1 .equ h'fff0 ;System Control Register 1
SYSCR2 .equ h'fff1 ;System Control Register 2
IENR1 .equ h'fff3 ;Interrupt Enable Register 1
IRR1 .equ h'fff6 ;Interrupt Request Register 1
;
;***********************************************************
;* Vector Address
;***********************************************************
;
.org h'0000
.data.w MAIN ;No.0 Reset Interrupt (H'0000-H'0001)
;
.org h'0008
.data.w MAIN ;No.4 _IRQ0 Interrupt (H'0008-H'0009)
data.w MAIN ;No.5 _IRQ1 Interrupt (H'000A-H'000B)
data.w MAIN ;No.6 _IRQ2 Interrupt (H'000C-H'000D)
data.w MAIN ;No.7 _IRQ3 Interrupt (H'000E-H'000F)
.data.w MAIN ;No.8 _IRQ4 Interrupt(H'0010-H'0011)
.data.w MAIN ;No.9 _WKP0-_WKP7 Interrupt(H'0012-H'0013)

.org h'0016
.data.w TAINT ;No.11 Timer A Interrupt(H'0016-H'0017)
.data.w MAIN ;No.12 AEC Interrupt(H'0018-H'0019)
.data.w MAIN ;No.13 Timer C Interrupt(H'001A-H'001B)
.data.w MAIN ;No.14 Timer FL Interrupt(H'001C-H'001D)
.data.w MAIN ;No.15 Timer FH Interrupt(H'001E-H'001F)
.data.w MAIN ;No.16 Timer G Interrupt(H'0020-H'0021)
.data.w MAIN ;No.17 SCI31 Interrupt(H'0022-H'0023)
.data.w MAIN ;No.18 SCI32 Interrupt(H'0024-H'0025)
.data.w MAIN ;No.19 A/D Converter Interrupt(H'0026-H'0028)
.data.w MAIN ;No.20 Direct Transfer Interrupt(H'0028-H'0029)

;**********************************************************************
; MAIN : Main Routine
;**********************************************************************

.org h'1000

MAIN: .equ $

mov.w #h'ff80,sp ;Initialize Stack Pointer
orc #h'80,ccr ;Interrupt Disable

mov.b #h'e7,r0l ;Initialize System Control Regsiter
mov.b r0l,@SYSCR1
mov.b #h'f0,r0l
mov.b r0l,@SYSCR2

bclr #7,@IRR1
mov.b #h'80,r0l
mov.b r0l,@IENR1

mov.b #h'ff,r0l
mov.b r0l,@TMA
mov.b  #h'1a,r0l
mov.b  r0l,0TMA
;
andc  #h'7f,ccr
;
sleep
;
nop
;
EXIT:  bra  EXIT
;
;*******************************************************************************
;*  TMAINT : Timer A Interrupt Routine
;*******************************************************************************
;
TAINT:  .equ  $0
        bclr  #7,@IRRI
;
        mov.b  #h'00,r0l
        mov.b  r0l,@IENR1
;
        rte
;
        .end
### 2.7 Module Standby Mode Settings

<table>
<thead>
<tr>
<th>Module Standby Mode Settings</th>
<th>MCU: H8/3867 Series</th>
<th>Functions Used: Module Standby Mode</th>
</tr>
</thead>
</table>

**Settings**

Module standby halts the supply of the system clock to the module and stops module functions. Module standby can be set for individual peripheral functions. All built-in peripheral modules can be set to module standby mode.

**Setting module standby mode**

Module standby mode can be set for a particular module by clearing the corresponding bits of the clock stop register 1 (CKSTPR1) and clock stop register 2 (CKSTPR2) to 0.

**Canceling module standby mode**

Module standby mode can be cancelled for a particular module by setting the corresponding bits of the clock stop register 1 (CKSTPR1) and clock stop register 2 (CKSTPR2) to 1.

After reset, CKSTPR1 and CKSTPR2 are both initialized to H'FF.
Explanation of CKSTPR1 and CKSTPR2 registers

Table 1 gives explanations of the CKSTPR1 and CKSTPR2 registers.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Bit Number</th>
<th>Bit Name</th>
<th>Set Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKSTPR1</td>
<td>Bit 6</td>
<td>S31CKSTP</td>
<td>0</td>
<td>Sets SCI3-1 to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels SCI3-1 module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 5</td>
<td>S32CKSTP</td>
<td>0</td>
<td>Sets SCI3-2 to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels SCI3-2 module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 4</td>
<td>ADCKSTP</td>
<td>0</td>
<td>Sets A/D converter to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels A/D converter module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 3</td>
<td>TGCKSTP</td>
<td>0</td>
<td>Sets timer G to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels timer G module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 2</td>
<td>TFCKSTP</td>
<td>0</td>
<td>Sets timer F to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels timer F module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 1</td>
<td>TCCKSTP</td>
<td>0</td>
<td>Sets timer C to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels timer C module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 0</td>
<td>TACCKSTP</td>
<td>0</td>
<td>Sets timer A to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels timer A module standby mode</td>
</tr>
<tr>
<td>CKSTPR2</td>
<td>Bit 3</td>
<td>AECKSTP</td>
<td>0</td>
<td>Sets AEC to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels AEC module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 2</td>
<td>WDCKSTP</td>
<td>0</td>
<td>Sets WDT to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels WDT module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 1</td>
<td>PWCKSTP</td>
<td>0</td>
<td>Sets PWM to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels PWM module standby mode</td>
</tr>
<tr>
<td></td>
<td>Bit 0</td>
<td>LDCKSTP</td>
<td>0</td>
<td>Sets LCD to module standby mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>Cancels LCD module standby mode</td>
</tr>
</tbody>
</table>
Example of module standby mode settings

1. Function

In this example, while in active (high-speed) mode, timer F and timer G are set to module standby mode, and then make a transition to watch mode.

2. Notes

a. Operation continues in watch mode only when an external clock is input as the timer F and timer G input clock, or when \( \phi/4 \) is selected as the internal clock. If any other clock is selected, operation is stopped while in watch mode. Hence under these settings, after setting the timer F, timer G input clock to \( \phi/4 \), timer F and timer G should be set to module standby mode before the transition to watch mode.

b. For the settings of this example, all interrupt requests are disabled, so that if there is a transition to watch mode, watch mode can only be canceled by input from the RES pin.
Flowchart

Module standby mode

1 → CCR I-bit

H'8F → SYSCR1

H'F0 → SYSCR2

H'18 → TMA

H'00 → IRR2

H'00 → IENR2

H'FF → OCRFH

H'FF → OCRFL

H'10 → TCSRF

H'07 → TCRF

H'03 → TMG

H'F3 → CKSTPR1

H'FF → CKSTPR2

0 → CCR I-bit

Execute a SLEEP instruction

Set the I bit of CCR to 1 to mask an interrupt.

Set the SSBY, LSON, and TMA3 bits to 1, to make a transition to watch mode.

Clear the IRRTG and IRRTFH bits to 0, and initialize the interrupt request flag.

Clear the IENTG and IENTFH bits to 0, and disable timer G and timer FL interrupts.

Initialize both OCRFH and OCRFL to H'FF.

Disable TCFH and TCFL overflow interrupt requests, and enable TCF clear by a compare match.

Set timer F to 16-bit mode, and TCFL input clock to $\phi w/4$.

Disable TCG overflow interrupt requests and TCG clear by the input capture input signal, and set the TCG input clock to $\phi w/4$.

Clear the TFCKSTP and TGCKSTP bits to 0, and set timer F and timer G to module standby mode.

Clear the I bit of CCR to 0 to enable an interrupt.

Execute a SLEEP instruction to make a transition from active (high-speed) mode to watch mode.
Program Lists

;******************************************************
;* H8/3867 Application Note
;*
;* 'Module Standby Mode
;* - In Watch Mode, Timer F&G Module Standby Mode Set
;*
;* Function : Module Standby Mode
;*
;* External Clock : 6MHz
;* Internal Clock : 3MHz
;* Sub Clock : 32.768kHz
;******************************************************
;
.cpu 3001
;
;******************************************************
;* Symbol Definition
;******************************************************
;
TMA .equ h'fb0 ;Timer Mode Register A
TCRF .equ h'fb6 ;Timer Control Register F
TCSR .equ h'fb7 ;Timer Control/Status Register F
OCRFH .equ h'fb9 ;Output Compare Register FH
OCRFL .equ h'fbb ;Output Compare Register FL
TMG .equ h'fc0 ;Timer Mode Register G
SYSCR1 .equ h'ff0 ;System Control Register 1
SYSCR2 .equ h'ff1 ;System Control Register 2
IENR2 .equ h'ff4 ;Interrupt Enable Register 2
IR2 .equ h'ff7 ;Interrupt Request Register 2
CKSTPR1 .equ h'f'f ;Clock Stop Register 1
CKSTPR2 .equ h'f'f ;Clock Stop Register 2
;
;******************************************************
;* Vector Address
;******************************************************
;
.org h'0000
.data.w MAIN ;No.0 Reset Interrupt (H'0000-H'0001)
;
.org h'0008
.data.w MAIN ;No.4 _IRQ0 Interrupt (H'0008-H'0009)
.data.w MAIN ;No.5 _IRQ1 Interrupt (H'000A-H'000B)
.data.w MAIN ;No.6 _IRQ2 Interrupt (H'000C-H'000D)
.data.w MAIN ;No.7 _IRQ3 Interrupt (H'000E-H'000F)
.data.w MAIN ;No.8 _IRQ4 Interrupt (H'0010-H'0011)
.data.w MAIN ;No.9 _WKP0-_WKP7 Interrupt (H'0012-H'0013)
;
.org h'0016
.data.w MAIN ;No.11 Timer A Interrupt (H'0016-H'0017)
.data.w MAIN ;No.12 ABC Interrupt (H'0018-H'0019)
.data.w MAIN ;No.13 Timer C Interrupt (H'001A-H'001B)
.data.w MAIN ;No.14 Timer FL Interrupt (H'001C-H'001D)
.data.w MAIN ;No.15 Timer FH Interrupt (H'001E-H'001F)
.data.w MAIN ;No.16 Timer G Interrupt (H'0020-H'0021)
.data.w MAIN ;No.17 SCI31 Interrupt (H'0022-H'0023)
.data.w MAIN ;No.18 SCI32 Interrupt (H'0024-H'0025)
.data.w MAIN ;No.19 A/D Converter Interrupt (H'0026-H'0028)
.data.w MAIN ;No.20 Direct Transfer Interrupt (H'0028-H'0029)
;
;******************************************************************************
*
MAIN : Main Routine
;******************************************************************************
;
.org h'1000
;
MAIN: .equ $   
   mov.w #h'ff80,sp ;Initialize Stack Pointer
   orc #h'80,ccr ;Interrupt Disable
;
   mov.w #h'8ff0,r0 ;Initialize System Control Regsiter
   mov.b r0h,@SYSCR1
   mov.b r0l,@SYSCR2
   mov.b #h'18,r0l ;Initialize Timer Mode Register

Rev. 1.0, 08/99, page 122 of 151
mov.b r0, @TMA

; sub.b r0, r0 ; Initialize Timer F
mov.b r0, @IRR2
mov.b r0, @IENR2
mov.b #h'ff, r0
mov.b r0, @OCR2H
mov.b r0, @OCR2L
mov.b #h'10, r0
mov.b r0, @TCRF

; mov.b #h'03, r0 ; Initialize Timer G
mov.b r0, @TMG

; mov.w #h'f3ff, r0 ; Timer F & G Module Standby Mode ON
mov.b r0h, @CKSTPR1
mov.b r0l, @CKSTPR2

andc #h'7f, ccr ; Interrupt Enable

; sleep ; Transfer to Watch Mode
nop

; EXIT: bra EXIT

.end


2.8 Clock Operation Using Timer F

<table>
<thead>
<tr>
<th>Clock Operation Using Timer F</th>
<th>MCU: H8/3867 Series</th>
<th>Functions Used: Timer F</th>
</tr>
</thead>
</table>

Specifications

1. A 38.4-kHz subclock is used for clock operation employing timer F.

2. Timer F interrupts are issued every 1 sec, and a counter provided for clock use in RAM is incremented.

3. The clock counter provided in RAM has eight bits for counting seconds and eight bits for counting minutes; it begins counting from 00 min, 00 sec, and after counting up to 59 min, 59 sec, in the next cycle it is initialized to 00 min, 00 sec and continues counting.

4. After completion of initialization, there is a transition from active (high-speed) mode to watch mode, a timer F interrupt request causes a transition to subactive mode, the counter provided in RAM is incremented, and there is another transition to watch mode.

5. The mode transition diagram for this task example is shown in figure 1.

![Mode Transition Diagram](image-url)

**Figure 1** Diagram of Mode Transitions for this Task Example
Explanation of Functions Used

1. In this task example, clock operation is performed in which timer F is used to increment a counter provided in RAM every second. The features of timer F are as follows.

   • Four different internal clocks (φ/32, φ/16, φ/4, φ/4) or an external clock can be selected (external event counting is possible).
   • A single compare-match signal can be used for a toggle output to the TMOFH pin (the initial value of the toggle output can be set).
   • The counter can be reset by a compare-match signal.
   • There are a total of two interrupt factors: one compare-match, and one overflow.
   • Operation as two independent 8-bit timers (timer FH and timer FL) is also possible (in 8-bit mode).
   • When φ/4 is selected as the internal clock, operation in watch mode, subactive mode, and sleep mode is possible.
   • Using the module standby mode, it is possible to set standby mode in module units when not in use.
2. Figure 2 shows a block diagram of the timer F 16-bit compare match function used in this task example.

```
TCRF: Timer control register F
TCSRF: Timer control status register F
TCFH: 8-bit timer counter FH
TCFL: 8-bit timer counter FL
OCRFH: Output compare register FH
OCRFL: Output compare register FL
IRRTFH: Timer FH interrupt request flag
PSS: Prescaler S
```

Figure 2  Block Diagram of Timer F 16-Bit Compare Match Function
3. Table 1 explains timer F functions.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Timer F Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Timer control register F (TCRF)</strong></td>
<td>Functions TCRF is an 8-bit write-only register. It switches between 16-bit mode and 8-bit mode, selects between four types of internal clocks and an external event, and sets the output levels of the TMOFH and TMOFL pins. On reset, TCRF is initialized to H’00.</td>
</tr>
<tr>
<td><strong>Timer control status register F (TCSRF)</strong></td>
<td>Functions TCSRF is an 8-bit read/write register. It selects counter clear, sets the overflow flag, sets the compare match flag, and enables interrupt requests due to overflows.</td>
</tr>
<tr>
<td><strong>16-bit timer counter F (TCF)</strong></td>
<td>Functions TCF is a 16-bit read/write up-counter. It consists of cascade connections of the 8-bit timer counters (TCFH and TCFL). In addition to use as a 16-bit counter employing TCF for the upper eight bits and TCFL for the lower eight bits, TCFH and TCFL can also be used as independent 8-bit counters. TCFH and TCFL can be read and written from the CPU, but when used in 16-bit mode, data transfer with the CPU is via a temporary register (TEMP). On reset, TCFH and TCFL are both initialized to H’00. If the CKSH2 bit of TCRF is clear to 0, then TCF functions as a 16-bit counter. The TCF input clock can be selected using the CKSL2 to CKSL0 bits of TCRF. The CCLRH bit of TCSRF can be used to clear TCF on compare match. When TCF overflows, the OVFH bit of TCSRF is set to 1, and if the IENTFH bit of IENR2 is 1, an interrupt request is sent to the CPU.</td>
</tr>
<tr>
<td><strong>8-bit timer counter FH (TCFH)</strong></td>
<td>Functions TCFH can be made to operate as an independent 8-bit counter by setting the CKSH2 bit of TCRF to 1. The TCFH input clock is selected using the CKSH2 to CKSH0 bits of TCRF. The CCLRH bit of TCSRF can be used to clear TCFH on a compare match. When there is an overflow of TCFH, the OVFH bit of TCSRF is set to 1. At this time, if the OVFIE bit of TCSRF is 1, the IRRTFH bit of IRR2 is set to 1, and if the IENTFH bit of IENR2 is 1, an interrupt request is sent to the CPU.</td>
</tr>
<tr>
<td><strong>8-bit timer counter FL (TCFL)</strong></td>
<td>Functions TCFL can be made to operate as an independent 8-bit counter by setting the CKSH2 bit of TCRF to 1. The TCFL input clock is selected using the CKSL2 to CKSL0 bits of TCRF. The CCLRL bit of TCSRF can be used to clear TCFL on a compare match. When there is an overflow of TCFL, the OVFL bit of TCSRF is set to 1. At this time, if the OVIIE bit of TCSRF is 1, the IRRTFL bit of IRR2 is set to 1, and if the IENTFL bit of IENR2 is 1, an interrupt request is sent to the CPU.</td>
</tr>
</tbody>
</table>
### Table 1  Timer F Functions (cont)

#### 16-bit output compare register F (OCRF)

| Functions | OCRF consists of two 8-bit read/write registers (OCRFH and OCRFL). In addition to being used as a 16-bit register of which OCRFH is the upper eight bits and OCRFL is the lower eight bits, OCRFH and OCRFL can also be used as independent 8-bit registers. OCRFH and OCRFL can be read and written from the CPU, but when used in 16-bit mode, data transfer with the CPU is via TEMP. On reset, OCRFH and OCRFL are both initialized to H'FF.

On clearing the CKSH2 bit of TCRF to 0, OCRF operates as a 16-bit register. The contents of OCRF are constantly compared with TCF, and if the values of the two match, the CMFH bit of TCSRF is set to 1. At the same time, the IRRTFH bit of IRR2 is also set to 1. At this time if the IENTFH bit of IENR2 is 1, an interrupt request is sent to the CPU. The toggle output of a compare match can be output from the TMOFH pin. In addition, the TOLH bit of TCRF can be used to select the output level (high/low). |

#### 8-bit output compare register FH (OCRFH)

| Functions | When the CKSH2 bit of TCRF is set to 1, OCRF operates as two 8-bit registers (OCRFH and OCRFL). The contents of OCRFH are compared with TCFH, and the contents of OCRFL are compared with TCFL. If the values of OCRFH and TCFH match, the CMFH bit of TCSRF is set to 1. At the same time, the IRRTFH bit of IRR2 is also set to 1. At this time if the IENTFH bit of IENR2 is 1, an interrupt request is sent to the CPU. The toggle output of a compare match can be output from the TMOFH pin. In addition, the TOLH pin of TCRF can be used to select the output level (high/low). |

#### 8-bit output compare register FL (OCRFL)

| Functions | When the CKSH2 bit of TCRF is set to 1, OCRF operates as two 8-bit registers (OCRFH and OCRFL). The contents of OCRFH are compared with TCFH, and the contents of OCRFL are compared with TCFL. If the values of OCRFL and TCFL match, the CMFL bit of TCSRF is set to 1. At the same time, the IRRTFL bit of IRR2 is also set to 1. At this time if the IENTFL bit of IENR2 is 1, an interrupt request is sent to the CPU. The toggle output of a compare match can be output from the TMOFL pin. In addition, the TOLL pin of TCRF can be used to select the output level (high/low). |

#### Timer FH interrupt request flag (IRRTFH)

| Functions | IRRTFH of the OCRF register is set to 1, if TCF matches OCRF in 16-bit mode, if TCFH matches OCRFH in 8-bit mode, or if TCF and TCFH overflow when IENTFH is set to 1. IRRTFH is cleared to 0, if IRRTFH is written to 1 when IRRTFH is set to 1. |

#### Timer FL interrupt request flag (IRRTFL)

| Functions | IRRTFL of the OCRF register is set to 1, if TCFL matches OCRFL in 8-bit mode or if TCFL overflows when IENTFL is set to 1. IRRTFL is cleared to 0, if IRRTFL is written to 1 when IRRTFL is set to 1. |
Table 1  Timer F Functions (cont)

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer FH interrupt enable (IENTFH)</td>
<td>Functions IENTFH enables or disables interrupt requests caused by timer FH compare matches or overflows.</td>
</tr>
<tr>
<td>Timer FL interrupt enable (IENTFL)</td>
<td>Functions IENTFL enables or disables interrupt requests caused by timer FL compare matches or overflows.</td>
</tr>
<tr>
<td>Timer F event input (TMIF)</td>
<td>Functions TIMF is used as an event input pin to be input to the TCFL.</td>
</tr>
<tr>
<td>Timer FH output (TMOFH)</td>
<td>Functions TMOFH is a timer FH toggle output pin.</td>
</tr>
<tr>
<td>Timer FL output (TMOFL)</td>
<td>Functions TMOFL is a timer FL toggle output pin.</td>
</tr>
</tbody>
</table>

4. The method for setting the timer FH interrupt cycle is explained below.

In this task example, 38.4 kHz is used as the subclock, and the timer F operates as a clock time base.
By setting TCRF CKSL2 to 1, CKSL1 to 1 and CKSL0 to 1, the TCF input clock is set to φ_w/4. Here φ_w/4 is given by

φ_w/4 = 38.4 kHz/4 = 9.6 kHz

Hence the TCF input clock cycle is

1/9.6 kHz ≅ 104.167 µs

Here if OCRF is set to H'2580, then the time until the values of TCF and OCRF match is calculated as

H'2580 × (1/9.6 kHz) = 9600 × 104.167 µs = 1 sec

Hence the settings for OCRF used to set the timer FH interrupt cycle T_m are calculated by the following equation.

OCRF setting = T_m/(1/9.6 kHz) = T_m × 9.6 kHz

Table 2 shows timer FH interrupt cycle T_m values and OCRA setting examples.
5. The interface with the CPU is explained below.

TCF and OCRF are 16-bit read/write registers. On the other hand, the data bus between the CPU and internal peripheral modules has an 8-bit data width. Hence when the CPU accesses TCF or OCRF, it must do so via the 8-bit temporary register (TEMP).

When reading or writing TCF or writing OCRF in 16-bit mode, operations must always be performed in 16-bit units (with byte-size MOV instructions executed twice in succession), in the order of the upper byte and lower byte. If only the upper byte or only the lower byte is accessed, the data is not transferred correctly.

In 8-bit mode, there are no restraints on access order.

### Table 2 Examples of Timer FH Interrupt Cycles and OCRF Settings

<table>
<thead>
<tr>
<th>$T_{	ext{FH}}$ (sec)</th>
<th>Calculation</th>
<th>OCRF Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.125</td>
<td>$0.125 \text{ sec} \times 9.6 \text{ kHz} = 1200$</td>
<td>H'04B0</td>
</tr>
<tr>
<td>0.25</td>
<td>$0.25 \text{ sec} \times 9.6 \text{ kHz} = 2400$</td>
<td>H'0960</td>
</tr>
<tr>
<td>0.5</td>
<td>$0.5 \text{ sec} \times 9.6 \text{ kHz} = 4800$</td>
<td>H'12C0</td>
</tr>
<tr>
<td>1</td>
<td>$1 \text{ sec} \times 9.6 \text{ kHz} = 9600$</td>
<td>H'2580</td>
</tr>
<tr>
<td>0.125</td>
<td>$2 \text{ sec} \times 9.6 \text{ kHz} = 19200$</td>
<td>H'4B00</td>
</tr>
</tbody>
</table>
a. Write operation

By writing the upper byte, the upper byte data is transferred to TEMP. Next the lower byte is written; the data in TEMP is written to the upper byte register, and the lower-byte data is written directly to the lower byte register.

The TCF write operation when H'AA55 is written to TCF is illustrated in figure 3.

Figure 3  TCF Write Operation
b. **Read operation**

In the case of TCF, when the upper byte is read, the upper byte data is transferred directly to the CPU, and the lower byte data is transferred to TEMP. Next the lower byte data is read; the lower byte data in TEMP is transferred to the CPU.

In the case of OCRF, in upper byte reading the upper byte data is transferred directly to the CPU. In lower byte reading the lower byte data is also transferred directly to the CPU.

Figure 4 shows a TCF read operation when TCF contains H'AAFF.

![Figure 4  TCF Read Operation](image)
6. Notes on use of the timer F

While the timer F is operating in 16-bit timer mode, the following conflicts and operations may occur.

a. When all bits match and a compare match signal is generated, the TMOFH pin output is toggled. When TCRF writing by a MOV instruction and a compare match signal occur simultaneously, TOLH data resulting from TCRF writing is output to pin TMOFH. In 16-bit mode, the TMOFL pin output is undefined, and should not be used. Use it as a port instead.

b. When OCRFL writing and compare match signal generation occur simultaneously, the compare match signal is invalid. However, when the data to be written matches the counter value, at that point a compare match signal is generated. The compare match signal is output in synchronization with the TCFL clock, so that if the clock is stopped, no compare match signal is generated even if a compare match occurs.

c. When all 16 bits match and a compare match signal is generated, the compare match flag CMFH is set. Similarly, if conditions for the lower 8 bits are satisfied, CMFL is set.

d. When there is a TCF overflow, OVFH is set; but when the lower 8 bits of OVFL overflow, if set conditions are satisfied, OVFL is set. If TCFL writing and overflow signal output occur simultaneously, the overflow signal is not output.

e. When, in active mode and sleep mode, \( \phi /4 \) is selected as the TCF internal clock, the system clock and internal clock are out of synchronization, and so synchronization is secured by an internal synchronization circuit. This results in a maximum count cycle error of \( 1/\phi \) (sec). In order to prevent this error from occurring, the system must be operated in subactive mode, subsleep mode, or watch mode.
Table 3  Function Allocation

<table>
<thead>
<tr>
<th>Function</th>
<th>Function Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCRF</td>
<td>Sets 16-bit mode and selects TCFL input clock.</td>
</tr>
<tr>
<td>TCSRIF</td>
<td>Selects counter clear, sets the overflow flag, sets the compare match flag, and enables/disables interrupt requests due to overflows.</td>
</tr>
<tr>
<td>TCF</td>
<td>Functions as a 16-bit up-counter configured by the connection of 8-bit timer counters TCFH and TCFL. TCF counts internal clock ($\phi_W/4$), and sets the IRTFH and CMFH bits to 1 if a compare match occurs.</td>
</tr>
<tr>
<td>OCRF</td>
<td>16-bit register configured by the connection of 8-bit registers OCRFH and OCRFL. If OCRF matches TCF, a compare match signal is generated.</td>
</tr>
<tr>
<td>IRRTFH</td>
<td>Indicates if a timer FH interrupt is requested or not.</td>
</tr>
<tr>
<td>IENTFH</td>
<td>Enables or disables timer FH interrupt requests.</td>
</tr>
<tr>
<td>$\phi_W$</td>
<td>Subclock frequency, 38.4 kHz in this task example.</td>
</tr>
</tbody>
</table>
Explanation of Operation

1. Operation of the timer F 16-bit timer mode is explained below.

   Timer F is a 16-bit counter which is incremented each time a clock pulse is input. The value of the timer counter F is continuously compared with the value set in the output compare register F; when they match, the counter is cleared, an interrupt request is issued, and port toggle output is possible. The timer can also operate as two independent 8-bit timers.
   
   When the CKSH2 bit of the timer control register F (TCRF) is set to 0, timer F operates as a 16-bit timer.
   
   Immediately after reset, the timer counter F (TCF) is set to H'0000, the output compare register F (OCRF) is set to H'FFFF, and the timer control register F (TCRF) and timer control status register F (TCSRF) are both initialized to H'00. The counter begins to be incremented by input from an external event (TMIF). The external event edge is selected through the IEG3 bit of the IRQ edge select register (IEGR).
   
   As the operating clock for timer F, the CKSL2 through CKSL0 bits of TCRF can be used to select from three kinds of internal clock output by prescaler S, an internal clock which is 1/4 the subclock, or an external clock.
   
   The contents of TCF and OCRF are continuously compared; when the two match, the CMFH bit of TCSRF is set to 1. At this time if the IENTFH bit of IENR2 is 1 an interrupt request is sent to the CPU, and at the same time the TMOFH pin output is toggled. Also, if the CCLRH bit of TCSRF is 1, TCF is cleared. The output from pin TMOFH can be set by the TOLH bit of TCRF.
   
   When TCF overflows (H'FFFF → H'0000), the OVFH bit of TCSRF is set to 1. At the time, if both the OVFIEH bit of TCSRF and the IENTFH bit of IENR2 are 1, an interrupt request is sent to the CPU.
2. Timer F operating modes are indicated in table 4.

**Table 4  Timer F Operating Modes**

<table>
<thead>
<tr>
<th>Operating Mode</th>
<th>TCF</th>
<th>OCRF</th>
<th>TCRF</th>
<th>TCSRF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>Reset</td>
<td>Reset</td>
<td>Reset</td>
<td>Reset</td>
</tr>
<tr>
<td>Active</td>
<td>Functions</td>
<td>Functions</td>
<td>Functions</td>
<td>Functions</td>
</tr>
<tr>
<td>Sleep</td>
<td>Functions</td>
<td>Held</td>
<td>Held</td>
<td>Held</td>
</tr>
<tr>
<td>Watch</td>
<td>Functions/</td>
<td>Held</td>
<td>Held</td>
<td>Held</td>
</tr>
<tr>
<td></td>
<td>Halted*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subactive</td>
<td>Functions/</td>
<td>Functions</td>
<td>Functions</td>
<td>Functions</td>
</tr>
<tr>
<td></td>
<td>Halted*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Subsleep</td>
<td>Functions/</td>
<td>Held</td>
<td>Held</td>
<td>Held</td>
</tr>
<tr>
<td></td>
<td>Halted*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standby</td>
<td>Halted</td>
<td>Held</td>
<td>Held</td>
<td>Held</td>
</tr>
<tr>
<td>Module standby</td>
<td>Halted</td>
<td>Held</td>
<td>Held</td>
<td>Held</td>
</tr>
</tbody>
</table>

Note: * If $\phi/4$ is selected as the TCF’s internal clock in active mode or sleep mode, the system clock and internal clock are mutually asynchronous, and so synchronization is established by a synchronization circuit. This results in a maximum error of $1/\phi(s)$ in the count period.

When the counter is operated in subactive mode, watch mode, or subsleep mode, $\phi/4$ must always be selected as the internal clock. The counter will not operate if any other internal clock is selected.
3. Figure 5 illustrates the principle of operation in this task example.
Explanation of Software

1. Explanation of Modules

Table 5 explains the modules in this task example.

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Label Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main routine</td>
<td>MAIN</td>
<td>Initializes the stack pointer, RAM and timer F, enables interrupts, and executes a transition to watch mode.</td>
</tr>
<tr>
<td>Timer F interrupt processing routine</td>
<td>TFINT</td>
<td>Clears IRRTFH and CMFH to 0, save register data, increment and clear SEC and MIN defined in RAM, restores register data.</td>
</tr>
</tbody>
</table>

2. Explanation of Arguments

In this task example, no arguments are used.
3. Explanation of Internal Registers Used

Table 6 gives explanations of the internal registers used in this task example.

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCRF</td>
<td>CKSH2 to CKSH0</td>
<td>H'FFB6 Bit 6 to bit 4</td>
<td>CKSH2 = 0 CKSH1 = 0 CKSH0 = 0</td>
</tr>
<tr>
<td></td>
<td>Timer control register F (Clock select H 2 to 0)</td>
<td></td>
<td>Select the clock input to TCFH from among four internal clock source or TCFL overflow.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• When CKSH2 = 0, CKSH1 = 0 and CKSH0 = 0, TCFL overflow is selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• When CKSH2 = 0, CKSH1 = 0 and CKSH0 = 1, TCFL overflow is selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• When CKSH2 = 0, CKSH1 = 1 and CKSH0 = 0, TCFL overflow is selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• When CKSH2 = 1, CKSH1 = 0 and CKSH0 = 0, internal clock ( \phi /32 ) is selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• When CKSH2 = 1, CKSH1 = 0 and CKSH0 = 1, internal clock ( \phi /16 ) is selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• When CKSH2 = 1, CKSH1 = 1 and CKSH0 = 0, internal clock ( \phi /4 ) is selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• When CKSH2 = 1, CKSH1 = 1 and CKSH0 = 1, internal clock ( \phi w/4 ) is selected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Note that CKSH2 = 0, CKSH1 = 1 and CKSH0 = 1 cannot be specified</td>
</tr>
<tr>
<td>Register Name</td>
<td>Description</td>
<td>RAM Address</td>
<td>Setting</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>TCRF</td>
<td>Timer control register F (Clock select L 2 to 0)</td>
<td>H’FFB6 Bit 2 to bit 0</td>
<td>CKSL2 = 1 CKSL1 = 1 CKSL0 = 1</td>
</tr>
<tr>
<td></td>
<td>Select the clock input to TCFL from among four internal clock source or an external event.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKSL2 = 0, CKSL1 = 0 and CKSL0 = 0, an external event is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKSL2 = 0, CKSL1 = 0 and CKSL0 = 1, an external event is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKSL2 = 0, CKSL1 = 1 and CKSL0 = 0, an external event is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKSL2 = 1, CKSL1 = 0 and CKSL0 = 0, internal clock φ/32 is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKSL2 = 1, CKSL1 = 0 and CKSL0 = 1, internal clock φ/16 is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKSL2 = 1, CKSL1 = 1 and CKSL0 = 0, internal clock φ/4 is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CKSL2 = 1, CKSL1 = 1 and CKSL0 = 1, internal clock φw/4 is selected</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Note that CKSL2 = 0, CKSL1 = 1 and CKSL0 = 1 cannot be specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCSRF</td>
<td>Timer control/status register F (Timer overflow flag H)</td>
<td>H’FFB7 Bit 7</td>
<td>0</td>
</tr>
<tr>
<td>OVFH</td>
<td>A status flag indicating overflow of TCF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When OVFH = 0, indicates no overflow of TCF</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When OVFH = 1, indicates TCF overflow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCSRF</td>
<td>Timer control/status register F (Compare match flag H)</td>
<td>H’FFB7 Bit 6</td>
<td>0</td>
</tr>
<tr>
<td>CMFH</td>
<td>A status flag indicating that TCF has matched OCRF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMFH = 0, indicates no compare match between TCF and OCRF</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When CMFH = 1, indicates TCF has matched OCRF</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 6  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCSR F OVIEH</td>
<td>Timer control/status register F (Timer overflow interrupt enable H) Enables or disables interrupt generation when TCF overflows. • When OVIEH = 0, disables TCF overflow interrupt requests • When OVIEH = 1, enables TCF overflow interrupt requests</td>
<td>H'FFB7 Bit 5</td>
<td>1</td>
</tr>
<tr>
<td>TCSR F CCLR H</td>
<td>Timer control/status register F (Counter clear H) Selects whether or not TCF is cleared when TCF has matched OCRF. • When CCLR H = 0, disables TCF clear by compare match • When CCLR H = 1, enables TCF clear by compare match</td>
<td>H'FFB7 Bit 4</td>
<td>1</td>
</tr>
<tr>
<td>TCF H</td>
<td>Timer counter FH Upper 8 bits of 16-bit timer counter F (TCF); functions as an 8-bit up-counter using a TCFL overflow signal as an input clock.</td>
<td>H'FFB8</td>
<td>H'00</td>
</tr>
<tr>
<td>TCF L</td>
<td>Timer counter FL Lower 8 bits of 16-bit timer counter F (TCF); functions as an 8-bit up-counter using φ/4 of internal clock as an input clock.</td>
<td>H'FFB9</td>
<td>H'00</td>
</tr>
<tr>
<td>OCRFH</td>
<td>Output compare register FH Upper 8 bits of 16-bit output compare register (OCRF); generates a compare match signal when OCRF has matched TCF.</td>
<td>H'FFBA</td>
<td>H'25</td>
</tr>
<tr>
<td>OCRFL</td>
<td>Output compare register FL Lower 8 bits of 16-bit output compare register (OCRF); generates a compare match signal when OCRF has matched TCF.</td>
<td>H'FFBB</td>
<td>H'80</td>
</tr>
</tbody>
</table>
Table 6  
Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
</table>
| IENR2 IENTFH  | Interrupt enable register 2 (Timer FH interrupt enable) Enables or disables timer FH interrupt requests.  
• When IENTFH = 0, disables timer FH interrupt requests  
• When IENTFH = 1, enables timer FH interrupt requests | H'FFF4  
Bit 3 | 1 |
| IRR2 IRRTFH   | Interrupt request register 2 (Timer FH interrupt request flag) Indicates whether there has been a timer FH interrupt request.  
• When IRRTFH = 0, indicates that no timer FH interrupt has been requested  
• When IRRTFH = 1, indicates that a timer FH interrupt has been requested | H'FFF7  
Bit 3 | 0 |
| SYSCR1 SSBY   | System control register 1 (Software standby) Carries out transitions to standby mode or watch mode.  
• When SSBY = 0, after executing a SLEEP instruction in active mode, causes a transition to sleep mode, or after executing a SLEEP instruction in subactive mode, causes a transition to subsleep mode.  
• When SSBY = 1, after executing a SLEEP instruction in active mode, causes a transition to standby mode or to watch mode, or after executing a SLEEP instruction in subactive mode, causes a transition to watch mode. | H'FFF0  
Bit 7 | 1 |
Table 6  Explanation of Internal Registers Used (cont)

<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSCR1</td>
<td>System control register 1 (Standby timer select 2 to 0) Specify the time for the CPU and peripheral functions to wait until the clock stabilizes when standby mode or watch mode is canceled and a transition is made to active mode due to a specific interrupt. Note that the standby time must be specified to be equal to or longer than the oscillation stabilization time according to the operating frequency.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>H'FFF0</td>
<td>STS2 = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bit 6 to bit 4</td>
<td>STS1 = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>STS0 = 0</td>
</tr>
<tr>
<td>STS2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STS1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STS0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- When STS2 to STS1 = 000, standby time is 8,192 states
- When STS2 to STS1 = 001, standby time is 16,384 states
- When STS2 to STS1 = 010, standby time is 32,768 states
- When STS2 to STS1 = 011, standby time is 65,536 states
- When STS2 to STS1 = 100, standby time is 131,072 states
- When STS2 to STS1 = 101, standby time is 2 states
- When STS2 to STS1 = 110, standby time is 8 states
- When STS2 to STS1 = 111, standby time is 16 states

<table>
<thead>
<tr>
<th>SYSCR1</th>
<th>LSON</th>
<th>System control register 1 (Low speed on flag) When watch mode is canceled, selects either the system clock (φ) or the subclock (φ_{sub}) as the CPU operating clock.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>H'FFF0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bit 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

- When LSON = 0, selects the system clock (φ) as the CPU operating clock
- When LSON = 1, selects the subclock (φ_{sub}) as the CPU operating clock
<table>
<thead>
<tr>
<th>Register Name</th>
<th>Description</th>
<th>RAM Address</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSCR2 DTON</td>
<td>System control register 2 (Direct transfer on flag)</td>
<td>H'FFF1 Bit 3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Specifies whether or not to make direct transitions among active (high-speed) mode, active (medium-speed) mode, and subactive mode when a SLEEP instruction is executed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTON = 0, if a SLEEP instruction is executed in active mode, a transition to standby mode, watch mode or sleep mode occurs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When DTON = 1, if a SLEEP instruction is executed in active (high-speed) mode, a direct transition occurs to active (medium-speed) mode (when SSBY = 1, MSON = 1, LSON = 0) or to subactive mode (when SSBY = 1, TMA = 1, LSON = 1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSCR2 MSON</td>
<td>System control register 2 (Medium speed on flag)</td>
<td>H'FFF1 Bit 2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Selects whether to operate in active (high-speed) mode or in active (medium-speed) mode after cancellation of standby mode, watch mode, or sleep mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When MSON = 0, operates in active (high-speed) mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When MSON = 1, operates in active (medium-speed) mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMA TMA3</td>
<td>Timer mode register A (Internal clock select 3)</td>
<td>H'FFB0 Bit 3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Selects the clock input to TCA.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When TMA3 = 0, PSS is selected as the TCA input clock source</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• When TMA3 = 1, PSW is selected as the TCA input clock source</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. Explanation of RAM Usage

Table 7 explains RAM usage for this task example.

<table>
<thead>
<tr>
<th>Label Name</th>
<th>Function</th>
<th>RAM Address</th>
<th>Modules Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEC</td>
<td>Counter used as a clock, counting the minutes</td>
<td>H'F780</td>
<td>MAIN, TFINT</td>
</tr>
<tr>
<td>MIN</td>
<td>Counter used as a clock, counting the seconds</td>
<td>H'F781</td>
<td>MAIN, TFINT</td>
</tr>
</tbody>
</table>
Flowchart

1. Main routine

```
  MAIN
  H'FF80 → SP  ------------------------ Initialize the stack pointer to H'FF80.
  H'80 → CCR  ------------------------ Mask the I bit of CCR to disable an interrupt.
  H'00 → SEC  ------------------------ Initialize the RAM counter to measure time.
  H'00 → MIN

  H'8C → SYSCR1

  H'F0 → SYSCR2

  H'08 → TMA

  H'08 → IENR2  ------------------------ Set IENTFH to 1 to enable timer FH interrupts.

  H'25 → OCRFH

  H'80 → OCRFL

  H'30 → TCSRF

  1 → CCR I-bit  ------------------------ Clear the I bit of CCR to enable an interrupt.

  H'07 → TCRF  ------------------------ Execute a SLEEP instruction to make a transition to watch mode.

  SLEEP
  NOP
```

Specify SSBY = 1, LSON = 1, MSON = 0, and TMA3 = 1 to make a transition from active (high-speed) mode to watch mode and transitions between watch mode and subactive mode.

Set IENTFH to 1 to enable timer FH interrupts.

Set OCRF to H'2580 to set timer FH interrupt request cycle = 1sec.

Set OVIEH to 1 to enable TCF overflow interrupt requests, and set CCLRH to 1 to enable TCF clear by a compare match.

Specify CKSH2 = 0, CKSH1 = 0, and CKSH0 = 0 to set timer F in 16-bit timer mode, and then specify CKSL2 = 1, CKSL1 = 1, and CKSL0 = 1 to select 1/4 of internal clock as a TCF input clock.

Rev. 1.0, 08/99, page 146 of 151

RENEAS
2. Timer F interrupt processing routine

- **TFINT**
- \(0 \rightarrow \text{IRRTFH}\)  
  - Clear the timer FH interrupt request flag to 0.
- \(0 \rightarrow \text{CMFH}\)  
  - Clear the compare match flag H to 0.
- **PUSH R0**  
  - Save register data onto the stack.
- \(\text{SEC} + 1 \rightarrow \text{SEC}\)  
  - Increment the RAM counter measuring seconds (SEC).
- **Yes**  
  - \(\text{SEC} \neq \text{H'3C}\)  
    - Check whether or not SEC defined in RAM has been incremented to H'3C (60).
  - \(\text{H'00} \rightarrow \text{SEC}\)  
    - Clear SEC to H'00 because SEC has been incremented to H'3C (60).
  - **Yes**  
    - \(\text{MIN} + 1 \rightarrow \text{MIN}\)  
      - Increment the RAM counter measuring minutes (MIN).
    - **Yes**  
      - \(\text{MIN} \neq \text{H'3C}\)  
        - Check whether or not MIN defined in RAM has been incremented to H'3C (60).
      - \(\text{H'00} \rightarrow \text{MIN}\)  
        - Clear MIN to H'00 because MIN has been incremented to H'3C (60).
    - **No**  
      - \(\text{H'00} \rightarrow \text{MIN}\)  
        - Clear MIN to H'00 because MIN has been incremented to H'3C (60).
  - **No**  
    - Clear MIN to H'00 because MIN has been incremented to H'3C (60).
- **POP R0**  
  - Restore register data.
- **RTE**

*Clear the timer FH interrupt request flag to 0.
Clear the compare match flag H to 0.
Save register data onto the stack.
Increment the RAM counter measuring seconds (SEC).
Check whether or not SEC defined in RAM has been incremented to H'3C (60).
Clear SEC to H'00 because SEC has been incremented to H'3C (60).
Increment the RAM counter measuring minutes (MIN).
Check whether or not MIN defined in RAM has been incremented to H'3C (60).
Clear MIN to H'00 because MIN has been incremented to H'3C (60).*
Program Lists

;******************************************************************************
;*     H8/3867 Application Note
;*
;*     'Timer F -Clock Time Base-
;*    
;*      Function : Timer F
;*
;*      External Clock : 6MHz
;*      Internal Clock : 3MHz
;*      Sub Clock      : 38.4kHz
;******************************************************************************

; .cpu 3001

;******************************************************************************
;* Symbol Definition
;******************************************************************************

; TMA .equ h'ffb0  ;Timer Mode Register A
TCRF .equ h'ffb6  ;Timer Control Register F
TCSR .equ h'ffb7  ;Timer Control/Status Register F
TCFH .equ h'ffb8  ;8-bit Timer Counter FH
TCFL .equ h'ffb9  ;8-bit Timer Counter FL
OCR .equ h'ffba   ;Output Compare Register FH
OCRFL .equ h'ffbb ;Output Compare Register FL
SYSCR1 .equ h'fff0 ;System Control Register 1
SYSCR2 .equ h'fff1 ;System Control Register 2
IENR2 .equ h'fff4 ;Interrupt Enable Register 2
IRR2 .equ h'fff7 ;Interrupt Request Register 2

;******************************************************************************
;* RAM Allocation
;******************************************************************************

; SEC  .equ h'f80  ;Second Counter
MIN   .equ h'f81  ;Minute Counter

Rev. 1.0, 08/99, page 148 of 151

Renesas
Vector Address

MAIN: .equ $

mov.w #H'ff80,sp ;Initialize Stack Pointer
orc #h'80,ccr ;Interrupt Disable
 ;
 sub.b  r01,r01   ;Initialize RAM
 mov.b  r01,@SEC
 mov.b  r01,@MIN
 ;
 mov.b  #h'8c,r01   ;Initialize System Control
 mov.b  r01,@SYSCR1
 mov.b  #h'f0,r01
 mov.b  r01,@SYSCR2
 mov.b  #h'08,r01
 mov.b  r01,@TMA
 ;
 mov.b  #h'08,r01   ;Timer F Interrupt Enable
 mov.b  r01,@IENR2
 ;
 mov.b  #h'25,r0h   ;Initialize Timer F
 mov.b  #h'80,r0l
 mov.b  r0h,@OCRPH
 mov.b  r0l,@OCRFL
 mov.b  #h'30,r0l
 mov.b  r0l,@TCSRF
 ;
 andc  #h'7f,ccr   ;Interrupt Enable
 ;
 mov.b  #h'07,r0l   ;Initialize TCFL Input Clock
 mov.b  r0l,@TCRF
 ;
 LOOP:   sleep   ;Transfer to Watch Mode
         nop
         bra  LOOP
 ;
 ;***********************************************************
 ;*  TFINT : Timer F Interrupt Routine
 ;***********************************************************
 ;
 TFINT:  .equ   $  
 bclr    #3,@IRR2   ;Clear Timer F Interrupt Request Flag
bclr #6, @TCSRFF ; Clear Compare Match Flag H

; push r0 ; Store r0

; mov.b @SEC, r0l ; Load Second Counter
mov.b @MIN, r0h ; Load Minute Counter
inc r0l ; Increment Second Counter
cmp.b #h'3c, r0l ; @SEC = d'60 ?
bne INTEXT ; No. Exit
mov.b #h'00, r0l ; Yes. Initialize Second Counter
inc r0h ; Increment Minute Counter
cmp.b #h'3c, r0h ; @MIN = d'60 ?
bne INTEXT ; No. Exit
mov.b #h'00, r0h ; Yes. Initialize Minute Counter

INTEXT: mov.b r0h, @MIN ; Store Minute Counter
mov.b r0l, @SEC ; Store Second Counter

; pop r0 ; Restore r0

; rte
;
.end