

To our customers,

---

## Old Company Name in Catalogs and Other Documents

---

On April 1<sup>st</sup>, 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 1<sup>st</sup>, 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 them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

# 使用说明

## 78K0S/Kx1+

### 举例程序(初始设置)

### LED 灯的开关控制

本文档概述了举例程序中的初始设置并描述了微控制器的初始设置。该举例程序中，微控制器完成诸如时钟频率或 I/O 端口选择此类基本初始设置后，可控制两个开关输入和三个 LED 灯。

#### 目标器件：

78K0S/KA1+微控制器  
78K0S/KB1+微控制器  
78K0S/KU1+微控制器  
78K0S/KY1+微控制器

#### 目录

第一章 概要.....	3
第二章 电路图 .....	4
2.1 电路图 .....	4
2.2 外围硬件 .....	4
第三章 软件.....	5
3.1 文件配置.....	5
3.2 所使用的内部外设功能 .....	6
3.3 初始设置和工作概要 .....	6
3.4 流程图 .....	7
第四章 设置方法 .....	8
4.1 选项字节设置 .....	8
4.2 向量表设置 .....	12
4.3 堆栈指针设置 .....	13
4.4 看门狗定时器设置 .....	14
4.5 时钟设置.....	17
4.6 端口设置.....	23
4.7 主处理程序.....	27
第五章 使用系统仿真器 SM+进行操作检查.....	29
5.1 创建举例程序 .....	29
5.2 SM+的操作.....	30
第六章 相关文档 .....	34
附录 A 程序清单 .....	35
附录 B 版本修订历史 .....	42

文档编号 U18752CA1V0AN00(第一版)  
发布日期 2008 年 02 月 N CP(K)

© NEC Electronics Corporation 2008  
于日本印刷

- 本文档信息发布于2008年02月。未来可能未经预先通知而进行更改。在实际进行生产设计时，请参阅各产品最新的数据规格书或数据手册等相关资料，以获取本公司产品的最新规格。并非所有的产品和/或型号都向每个国家供应。请向本公司销售代表查询产品供货及其他信息。
- 未经本公司事先书面许可，禁止采用任何方式复制或转载本文件中的内容。本文件所登载内容的错误，本公司概不负责。
- 本公司对于因使用本文件中列明的本公司产品而引起的，对第三者的专利、版权以及其它知识产权的侵权行为概不负责。本文件登载的内容不应视为本公司对本公司或其他人所有的专利、版权以及其它知识产权做出任何明示或默示的许可及授权。
- 本文件中的电路、软件以及相关信息仅用以说明半导体产品的运作和应用实例。用户如在设备设计中应用本文件中的电路、软件以及相关信息，应自行负责。对于用户或其他人因使用了上述电路、软件以及相关信息而引起的任何损失，本公司概不负责。
- 虽然本公司致力于提高半导体产品的质量及可靠性，但用户应同意并知晓，我们仍然无法完全消除出现产品缺陷的可能。为了最大限度地减少因本公司半导体产品故障而引起的对人身、财产造成损害（包括死亡）的危险，用户务必在其设计中采用必要的安全措施，如冗余度、防火和防故障等安全设计。
- 本公司产品质量分为：“标准等级”、“专业等级”以及“特殊等级”三种质量等级。

“特殊等级”仅适用于为特定用途而根据用户指定的质量保证程序所开发的日电电子产品。另外，各种日电电子产品的推荐用途取决于其质量等级，详见如下。用户在选用本公司的产品时，请事先确认产品的质量等级。

“标准等级”：计算机，办公自动化设备，通信设备，测试和测量设备，视音频设备，家电，加工机械，个人电气设备以及产业用机器人。

“专业等级”：运输设备（汽车、火车、船舶等），交通信号控制设备，防灾装置，防止犯罪装置，各种安全装置以及医疗设备（不包括专门为维持生命而设计的设备）。

“特殊等级”：航空器械，宇航设备，海底中继设备，原子能控制系统，为了维持生命的医疗设备和用于维持生命的装置或系统等。

除在本公司半导体产品的数据表或数据手册等资料中另有特别规定以外，本公司半导体产品的质量等级均为“标准等级”。如果用户希望在本公司设计意图以外使用本公司半导体产品，务必事先与本公司销售代表联系以确认本公司是否同意为该项应用提供支持。

（注）

（1）本声明中的“本公司”是指日本电气电子株式会社（NEC Electronics Corporation）及其控股公司。

（2）本声明中的“本公司产品”是指所有由日本电气电子株式会社或为日本电气电子株式会社（如上定义）开发或制造的产品。

M8E 02.11-1

第一章 概要

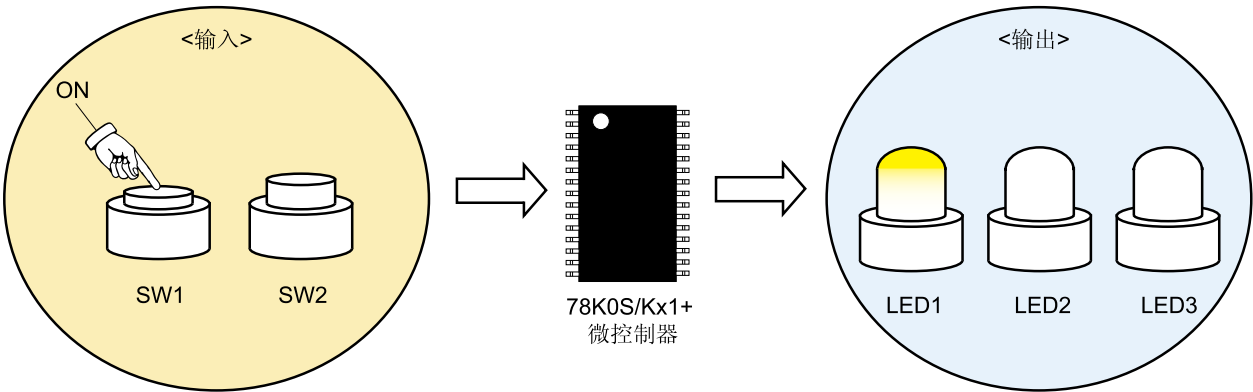
本举例程序中，完成了 78K0S/Kx1+微控制器基本的初始设置，如设置选项字节、选择时钟频率、以及设置 I/O 端口。完成初始设置后，在主处理操作中利用两个开关输入来控制三个 LED 灯。

(1) 初始设置的主要内容

- 选择内部高速振荡器作为系统时钟信号源。
- 停止看门狗定时器的运行。
- 将 CPU 和外围硬件的时钟频率都设为 2 MHz。
- 设置 I/O 端口。

(2) 主处理操作的内容

由 78K0S/Kx1+微控制器检测开关输入来控制 LED 灯（LED1、LED2、LED3）。



开关输入		LED输出		
SW1	SW2	LED1	LED2	LED3
OFF	OFF	OFF	OFF	OFF
ON	OFF	ON	OFF	OFF
OFF	ON	OFF	ON	OFF
ON	ON	OFF	OFF	ON

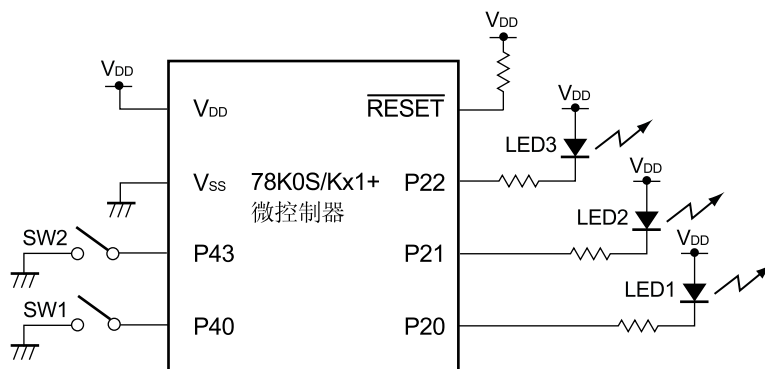
注意事项 有关使用器件时的注意事项，参见各产品的用户手册（78K0S/KU1+，78K0S/KY1+，78K0S/KA1+，78K0S/KB1+）。

## 第二章 电路图

本章介绍了该举例程序中所使用的电路图及外围硬件。

### 2.1 电路图

电路图显示如下：



- 注意事项
1. 直接将 **AVREF** 引脚连接至  $V_{DD}$ （仅用于 **78K0S/KA1+** 和 **78K0S/KB1+** 微控制器）。
  2. 直接将 **AVSS** 引脚连接至  $GND$ （仅用于 **78K0S/KB1+** 微控制器）。
  3. 除电路图中所示的引脚及 **AVREF** 引脚和 **AVSS** 引脚外，保留所有未用引脚为开路状态（未连接）。

### 2.2 外围硬件

所使用的外围硬件如下所示：

#### (1) 开关（SW1、SW2）

这些开关用于控制 LED 灯的输入。

#### (2) LED（LED1、LED2、LED3）




这些 LED 用作对应于开关输入的输出显示。

第三章 软件




本章介绍了所下载压缩文件的配置、所用微控制器的内部外设功能、以及该举例程序中的初始设置和操作概要，并显示了流程图。

3.1 文件组成

下表显示了所下载压缩文件的组成：

文件名	说明	包含的压缩文件 (*.zip)		
				
main.asm (汇编语言版)	有关微控制器硬件初始处理和主处理程序的源文件。	● <sup>注1</sup>	● <sup>注1</sup>	
main.c (C语言版)				
op.asm	设置选项字节(设置系统时钟信号源)的汇编程序源文件。	●	●	
initial.prw	集成开发环境PM+的Workspace文件。		●	
initial.prj	集成开发环境PM+的项目文件。		●	
initial.pri initial.prs initial.prm	78K0S/Kx1+系统仿真器SM+的项目文件。		● <sup>注2</sup>	
initial0.pnl	78K0S/Kx1+系统仿真器SM+的I/O面板文件(用于检查外围硬件的工作)。		● <sup>注2</sup>	●
initial0.wvo	78K0S/Kx1+系统仿真器SM+的时序图文件 (用于检查波形)。			●

- 注    1. 汇编版本包含“main.asm”文件， C 语言版本包含“main.c”文件。  
      2. 78K0S/KU1+微控制器的文件中不包含这些文件。

- 备注       :    仅包含源文件。
-     :    包含用于集成开发环境 PM+和 78K0S/Kx1+系统仿真器 SM+的文件。
-     :    包含用于 78K0S/Kx1+系统仿真器的微控制器工作仿真文件。

### 3.2 所使用的内部外设功能

该举例程序中，使用了微控制器的下列内部外设功能：

输入端口（用于开关输入）： P40、P43。

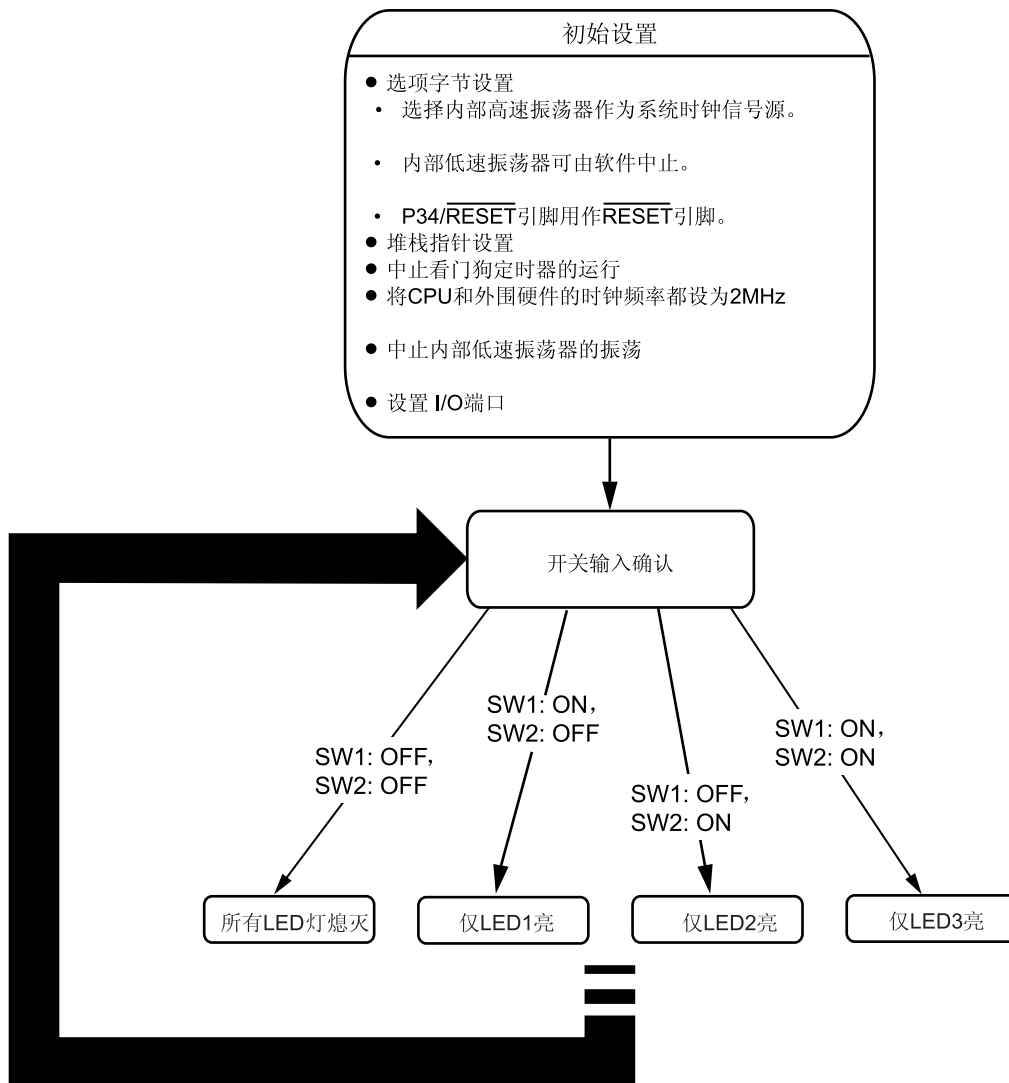
输出端口（用于 LED 灯）： P20、P21、P22。

### 3.3 初始设置和工作概要

该举例程序中，时钟频率选择、以及 I/O 端口的设置之类的操作在初始设置中完成。

初始设置完成后，三个 LED 灯（LED1、LED2、LED3）由两个开关输入（SW1、SW2）联合控制。

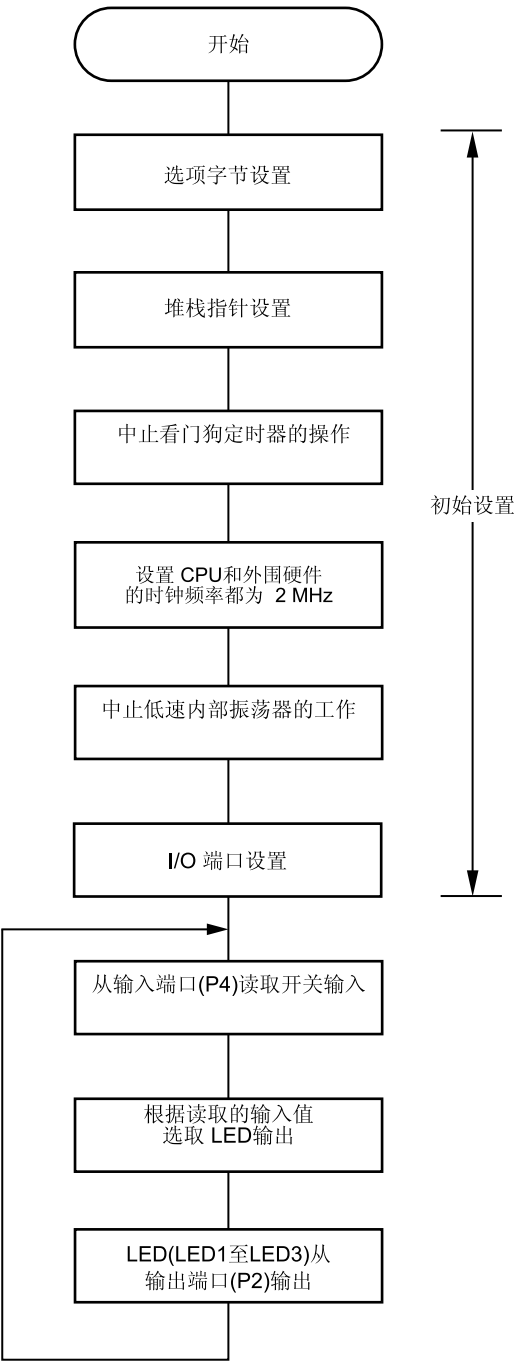
详情如下列状态转换图所示：





3.4 流程图

本举例程序的流程图如下所示：



第四章 设置方法

本章介绍了如何进行选项字节、向量表、堆栈指针、看门狗定时器、时钟频率、以及端口的设置，对主处理程序也进行了介绍。

关于如何设置寄存器，参见各产品的用户手册（78K0S/KU1+、78K0S/KY1+、78K0S/KA1+、78K0S/KB1+）。  
有关汇编语言指令，参见 78K/0S 系列指令用户手册。

4.1 选项字节设置

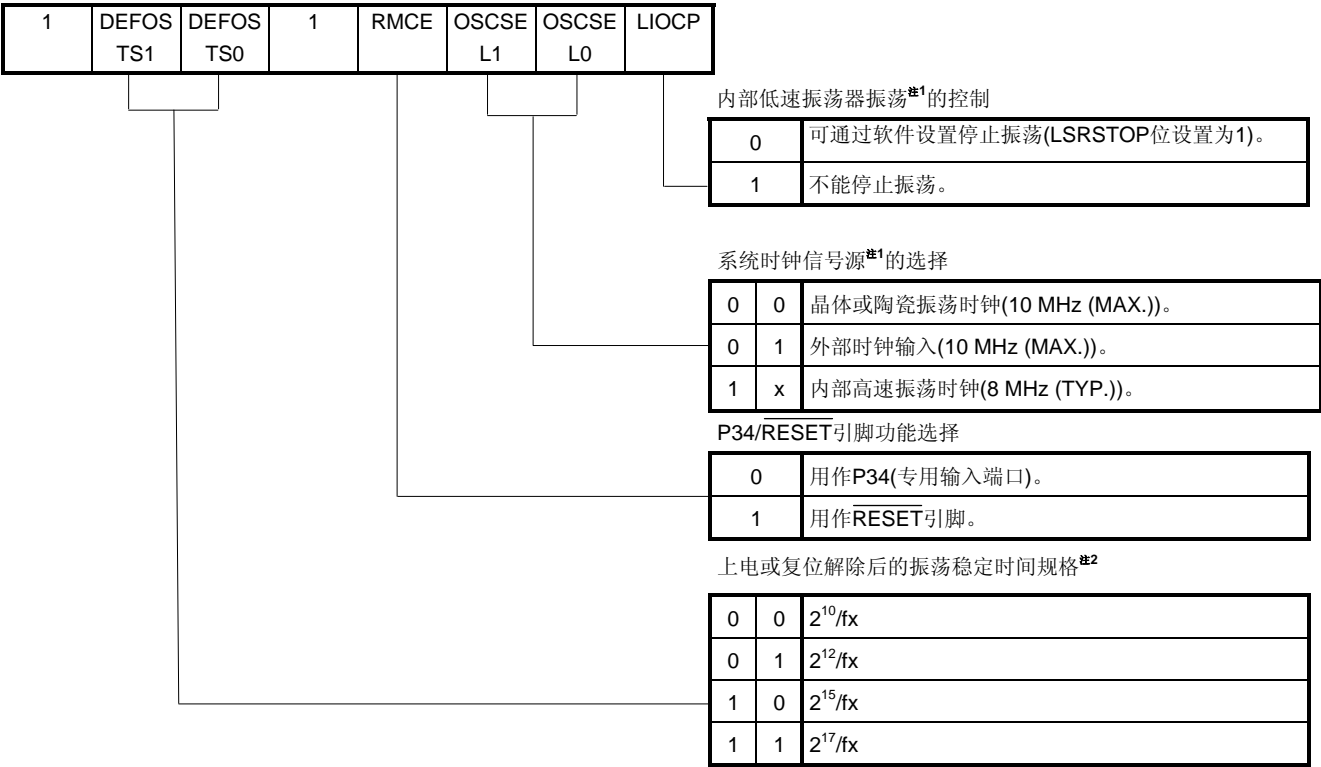
必须设置选项字节，由选项字节设置下列选项：

- (1) 选择系统时钟信号源。
- (2) 控制内部低速振荡器的振荡。
- (3) P34/RESET 引脚功能选择。
- (4) 上电或复位解除后的振荡稳定时间规格（仅当使用晶体振荡器或陶瓷振荡器时）。

该举例程序中，选项字节按照[实例 1]（随后涉及）中的说明进行设置。

图 4-1. 选项字节格式

地址：0080H



注 1. 复位解除后，停止内部低速振荡器的振荡、停止看门狗定时器的的工作、以及设置时钟频率。详情参见 4.4 看门狗定时器的设置和 4.5 时钟设置。  
(注 2、注意事项及备注在下页)。

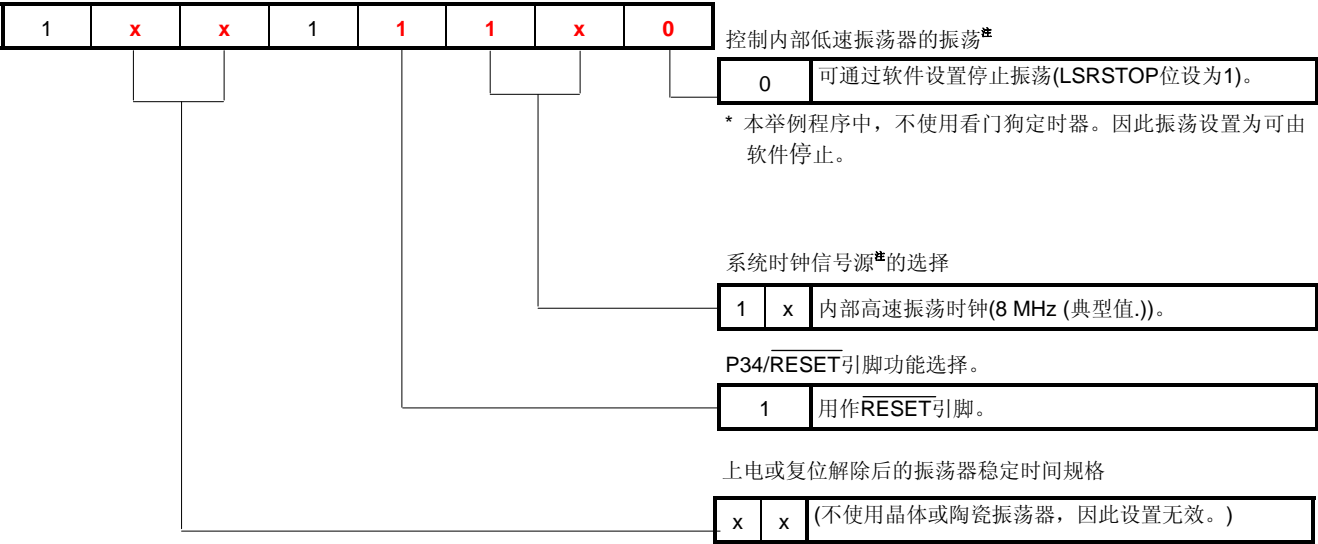
注 2. 如果将内部高速振荡时钟或外部时钟输入选择为系统时钟，则不需要振荡稳定时间，因此设置无效（无需理会）。

注意事项 位 4 和位 7 必须设为 1。

备注 x: 不必理会。

[实例 1] 使用内部高速振荡时钟，用作  $\overline{\text{RESET}}$  引脚，可以停止内部低速振荡器（与举例程序中的设置相同）。

地址：0080H



注 复位解除后，停止内部低速振荡器的振荡，停止看门狗定时器的的工作以及设置时钟频率。有关细节，参见 4.4 看门狗定时器的设置和 4.5 时钟设置。

选项字节设置值为“1xx11x0（x: 不必理会，位 4 和位 7 必须设为 1）”。当软件一并描述保护字节设置（在下文中，所有块允许写入）时，程序设计如下（下例中‘x’设置为 0）：

```
OPBT  CSEG  AT      0080H
      DB    10011100B
      DB    11111111B
```

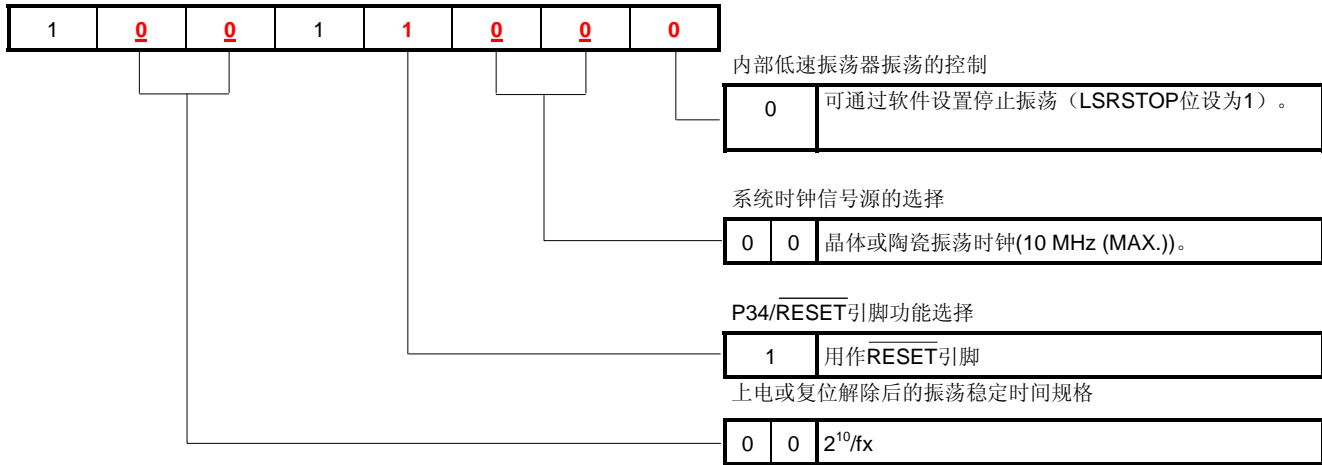
为使用 C 语言编程，安排如下所示的汇编语言源文件（文件名：“\*.asm（\*: 任意）”），指定其为工程源文件，并与其它源文件（main.c）一同编译之。

```
OPBT  CSEG  AT      0080H
      DB    10011100B
      DB    11111111B
      end
```

备注 保护字节（地址：0081H）用于设置禁止写入和块擦除的区域，其设置内容仅在自编程模式期间有效。详情参见各产品的用户手册（78K0S/KU1+、78K0S/KA1+、78K0S/KB1+）。

**[实例 2]** 使用晶体或陶瓷振荡时钟，用作  $\overline{\text{RESET}}$  引脚，可停止内部低速振荡器，并且振荡稳定时间最小化 ( $2^{10}/f_x$ )。  
(除带下划线的部分外，其余设置同实例 1。)

地址：0080H



选项字节设置值为“10011000 (位 4 和位 7 必须设为 1)”。当软件一并描述保护字节设置时，程序设计如下：

OPBT	CSEG	AT	0080H
	DB		10011000B
	DB		11111111B



**[专栏]** 什么是 CSEG (代码段)、DSEG (数据段)、以及 BSEG (位段)？

CSEG、DSEG、及 BSEG 伪指令指示了将要为指令代码或数据之类的代码分配的地址。这样的伪指令导致将所描述的数据和指令分别定位到相应的地址中去 (CSEG 伪指令定位于 ROM 区域，DSEG 伪指令定位于 RAM 区域，BSEG 伪指令定位于 RAM 区域中的 saddr 地址区域)。

例如：要将选项字节的设置内容分配到内部 ROM (flash 存储器) 中从 0080H 开始的地址，首先，CSEG 伪指令和 AT 标志用于将 0080H 规定为地址。其次，DB 伪指令用于定义从 0080H 地址开始的若干字节的存储数据值，此后在汇编语言程序代码中加以描述。

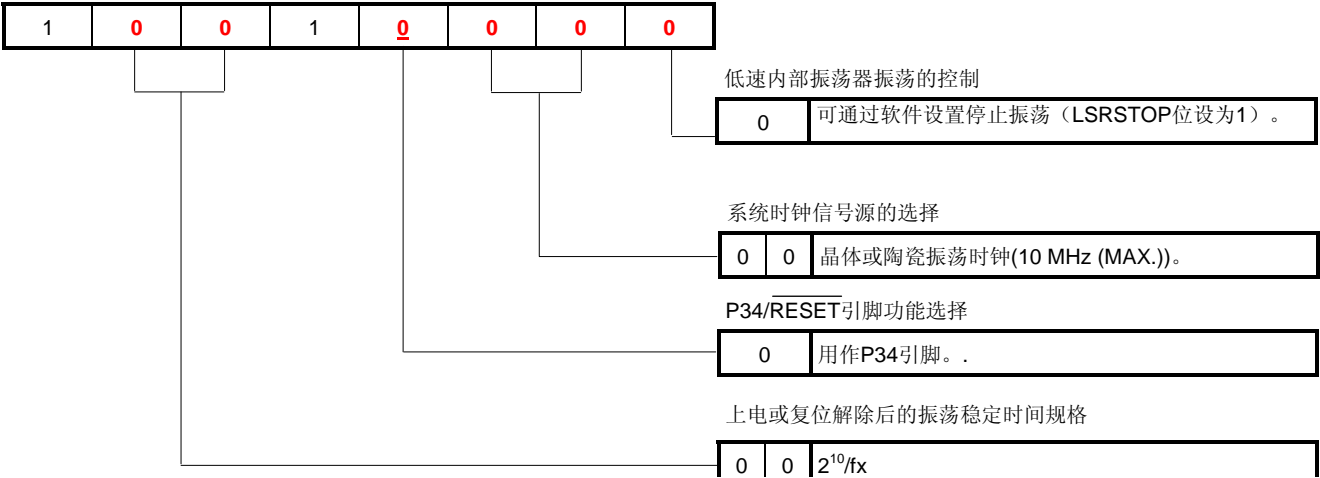
DB 伪指令和 DW 伪指令仅能够用于由 CSEG 伪指令指定的 ROM 区域。在由 DSEG 伪指令或 BSEG 伪指令所指定的 RAM 区域中，虽然声明 DB 伪指令或 DW 伪指令不会导致出错，但是禁止使用。这种情况下，使用 MINICUBE2 (片上调试仿真器) 或 SM+ (系统仿真器) 时，目标程序可以生成并得以执行，这是因为指令代码和数据已被扩展到了 RAM 区域。然而，对于实际器件来说，因为这些无法扩展至 RAM 区域，所以操作禁止。

关于 CSEG 伪指令、DSEG 伪指令、以及 BSEG 伪指令之详情，参见 RA78K0S 语言用户手册。

**[实例 3]**      使用晶体或陶瓷振荡时钟，用作 P34 引脚，可停止内部低速振荡器，并且振荡稳定时间最小化（ $2^{10}/f_x$ ）。

（除带下划线的部分外，其余设置同实例 2。）

地址：0080H



选项字节设置值为“10010000（位 4 和位 7 必须设为 1）”。当软件一并描述保护字节设置时，程序设计如下：

OPBT	CSEG	AT	0080H
	DB		10010000B
	DB		11111111B

4.2 向量表设置

因复位和各种中断请求而引起的程序转移，其起始地址存储在在向量表区域中。

本举例程序中，不执行中断服务，因此仅设置复位启动期间所使用的复位向量。用汇编语言编码时需要进行此项设置。用 C 语言编码时，因为启动例程会自动设置复位向量，所以不需要此项设置。

关于如何设置向量表和中断以及设置实例之详情，参见本举例程序中的“中断（preliminary name, under preparation）”。

**[实例 1]** 仅设置复位启动期间所使用的复位向量（同本举例程序中的设置一致）。

XVCT	CSEG	AT	0000H	地址	功能名称
<1>	DW	RESET_START		; (00)	RESET
	DW	RESET_START		; (02)	--
	DW	RESET_START		; (04)	--
	DW	RESET_START		; (06)	INTLVI
	DW	RESET_START		; (08)	INTP0
	DW	RESET_START		; (0A)	INTP1
	DW	RESET_START		; (0C)	INTTMH1
	DW	RESET_START		; (0E)	INTTM000
	DW	RESET_START		; (10)	INTTM010
	DW	RESET_START		; (12)	INTAD
	DW	RESET_START		; (14)	--
	DW	RESET_START		; (16)	INTP2
	DW	RESET_START		; (18)	INTP3
	DW	RESET_START		; (1A)	INTTM80
	DW	RESET_START		; (1C)	INTSRE6
	DW	RESET_START		; (1E)	INTSR6
	DW	RESET_START		; (20)	INTST6

复位解除后，程序从复位向量所指定的地址处（上述<1>RESET\_START 处）开始执行。

本举例程序中，不使用 0000H 之外的向量表地址，如同 0000H 一样，RESET\_START 被设置到其余所有的向量表地址。如此设置后，即使产生中断，程序也能跳转到 RESET\_START，并且执行与复位解除后相同的处理。

**[专栏]** 什么是#pragma 指示符？

#pragma 指示符是用于 C 语言中的预处理指令并且编写在源程序的开头。



主要的#pragma 命令如下：

- #pragma sfr: 关于 SFR 区域的相关操作可以在 C 语言层面上进行描述。
- #pragma ei: EI 指令能够在 C 语言层面上进行描述。
- #pragma di: DI 指令能够在 C 语言层面上进行描述。
- #pragma nop: NOP 指令能够在 C 语言层面上进行描述。(不进行 CPU 操作而时钟继续递进。)
- #pragma interrupt: 中断功能能够在 C 语言层面上进行描述。

#pragma 指示符之详情参见 [CC78K0S 语言用户手册](#)中关于扩展功能的章节。

4.3 堆栈指针设置

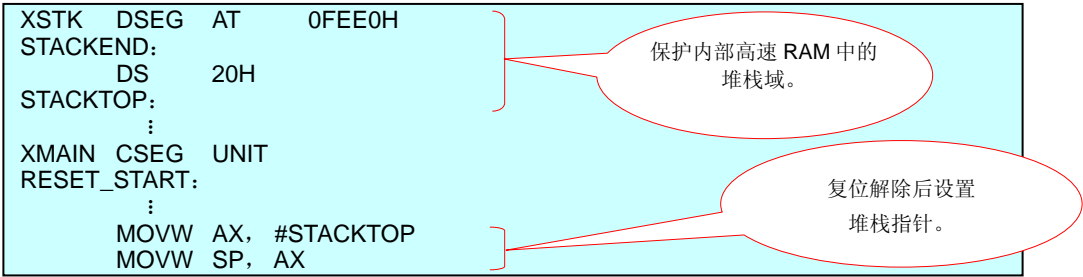
堆栈域是一块内存区域，用于临时存放诸如程序计数器、寄存器值、及 PSW（程序状态字）之类的数据。堆栈域只能规定在内部高速 RAM 区域内。由堆栈指针设置堆栈起始地址以保护堆栈域。

当执行下列指令或中断发生时使用堆栈域：

- PUSH、CALL、CALLT、中断：将数据保存到堆栈域中。
- POP、RET、RETI：从堆栈域中恢复数据。

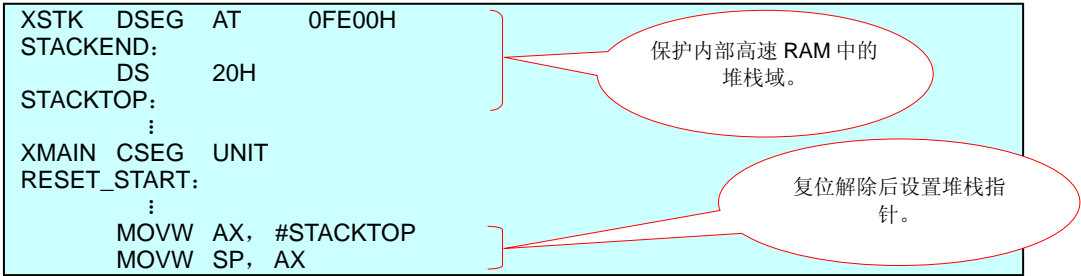
用汇编语言编码时，需设置堆栈域的安全性。用 C 语言编码时，由于启动例程会自动保护堆栈，因此不需要此项设置。

**[实例 1]** 将内部高速 RAM 区域中的 FEE0H 至 FEFFH（32 字节）用作堆栈域。  
（同本举例程序中的设置一致）



该实例中，将地址 FF00H (= FEE0H + 20H) 规定为堆栈指针。FF00H 不仅是一块高速 RAM 区域，同时也作为 SFR 区域，因此当其变换为高速 RAM 区域时，堆栈指针会变成 FB00H。实际上将数据存储至堆栈时，由于 FB00H 的递减（-1），使得堆栈指针成为 FAFFH，FF00H 之内容就这样被修改了。然而，此处并不是高速 RAM 区域，因此要将其转换到高速 RAM 区域中，变为 FEFFH。该值与指定地址 FF00H 为堆栈指针时的值相符，于是数据从地址 FEFFH 处开始存储。根据上述说明，内部高速 RAM 区域中最末 32 字节（FEE0H 至 FEFFH）也能够作为堆栈域而受到保护。

**[实例 2]** 将内部高速 RAM 区域中 FE00H 至 FE1FH（32 字节）用作堆栈域。



该实例中，指定地址 FE20H (= FE00H + 20H) 为堆栈指针。此项设置避开了 saddr 区域并保护了堆栈域。

**注意事项** 上述实例 2 的设置，仅可用于具备 256 字节内部高速 RAM 的产品。

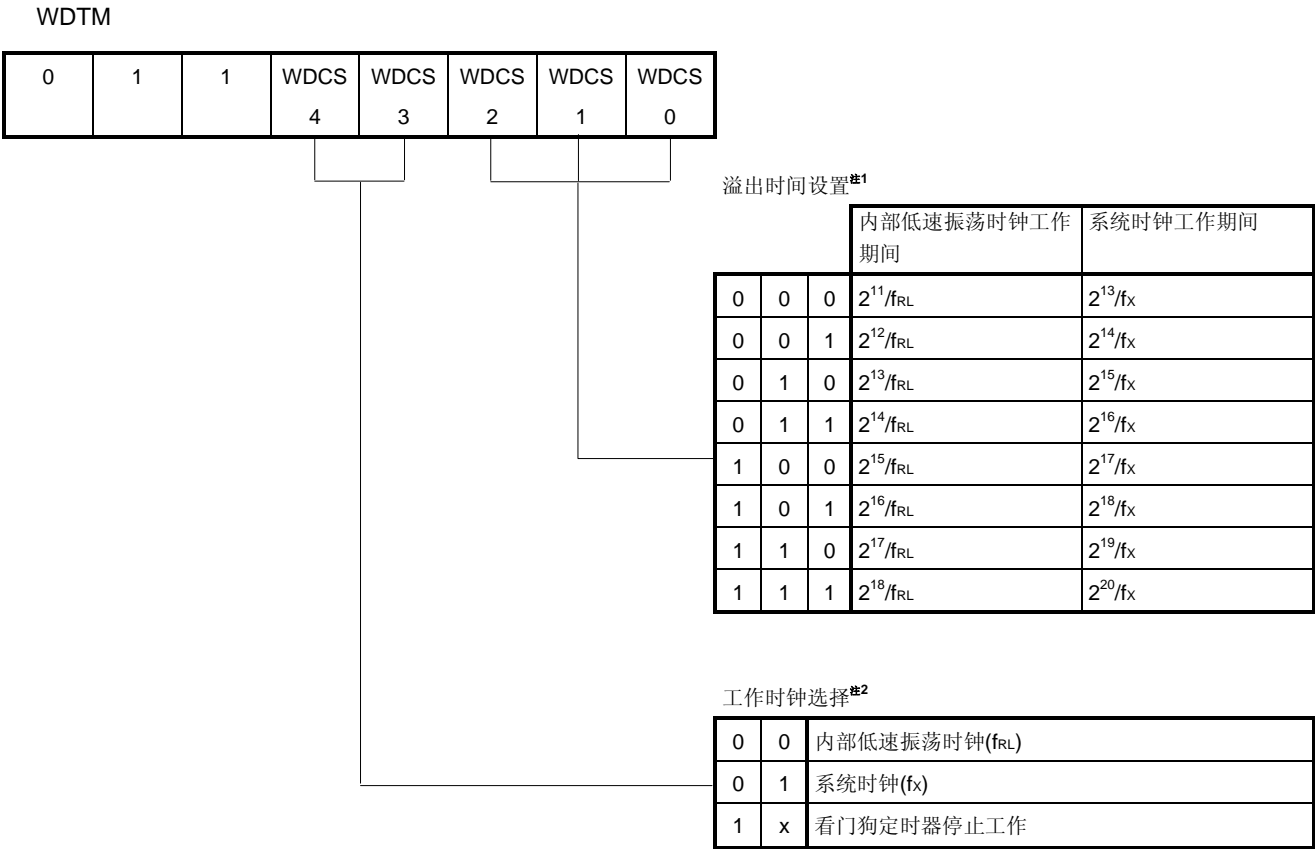
4.4 看门狗定时器的设置

WDTM 寄存器用于选择看门狗定时器的工作时钟和溢出时间。

本举例程序中，看门狗定时器没有用于程序循环检测，因此 WDTM 寄存器按照[\[实例 1\]](#)（随后涉及）中的说明进行设置。

**注意事项** 初始设置期间必须设置看门狗定时器的工作时钟和溢出时间。

图 4-2. 看门狗定时器模式寄存器格式(WDTM)



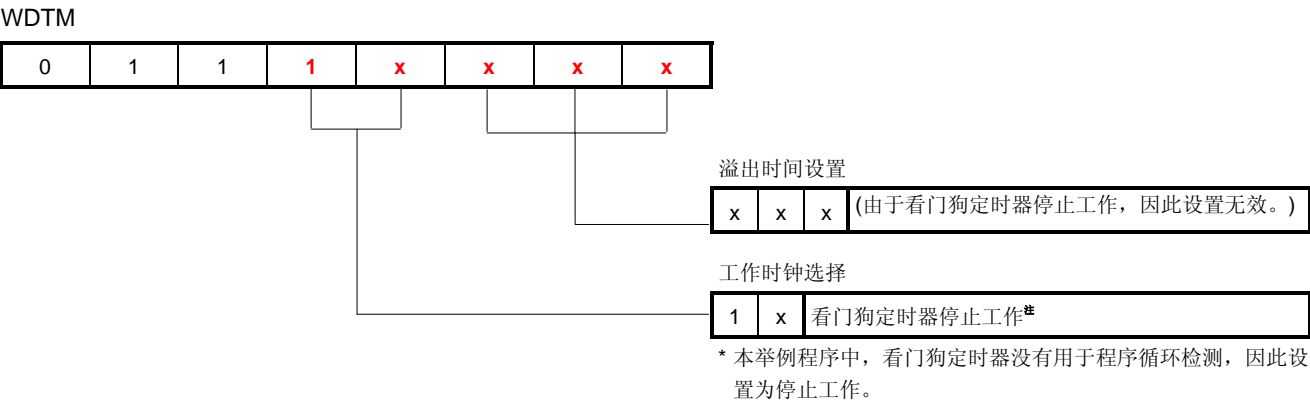
- 注**
1. 当选择“看门狗定时器停止工作”时，溢出时间设置无效（无需理会）。
  2. 为了停止看门狗定时器的工作或将系统时钟作为看门狗定时器的时钟，选项字节必须设为“可通过软件设置停止振荡（LSRSTOP 位设为 1）”。详情参见 [4.1 选项字节设置](#)。

**注意事项** 位 7、位 6、及位 5 必须分别设为 0、1、和 1。

**备注** x: 不必理会。



**[实例 1]** 看门狗定时器停止工作（同本举例程序中的设置一致）。



**注** 为停止看门狗定时器的工作，选项字节必须设为“可通过软件设置停止振荡（LSRSTOP 位设为 1）”。详情参见 [4.1 选项字节设置](#)。

WDTM 寄存器设置值为“0111xxxx（x：不必理会，位 7、位 6、及位 5 必须分别设为 0、1、和 1）”。（下列实例中，位 3 的“x”设为 0，位 2 的“x”设为 1。）

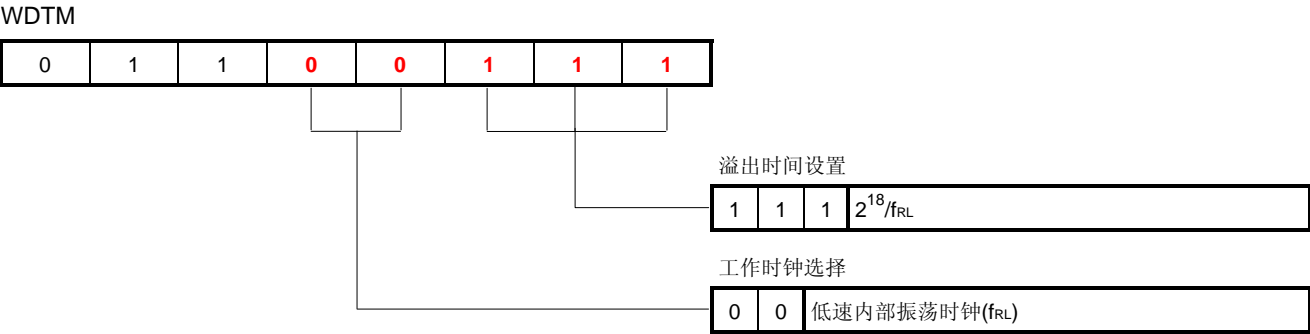
• 汇编语言

```
MOV WDTM, #01110111B
```


• C 语言

```
WDTM = 0b01110111;
```

**[实例 2]** 内部低速振荡时钟 (f<sub>RL</sub>)用作看门狗定时器的工作时钟，溢出时间设为最大周期(2<sup>18</sup>/f<sub>RL</sub>)（同复位解除后的 WDTM 值一致）。



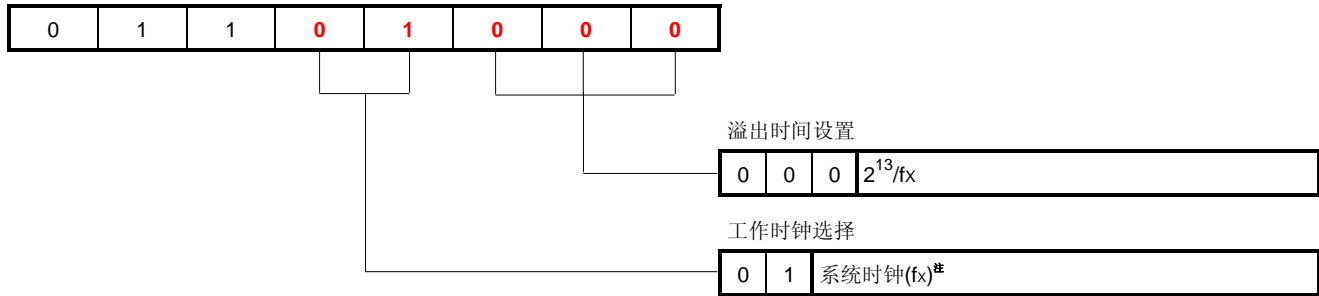
为依照上述说明使用 WDTM 寄存器，由于其设置值与复位解除后的值相同，故不需要通过程序来设置。

**[专栏]** 二进制值的描述

为了描述二进制值，汇编语言中在二进制值后缀“B”或“Y”，而 C 语言中二进制值前缀“0b”或“0B”。

**[实例 3]** 系统时钟( $f_x$ )用作看门狗定时器的工作时钟，溢出时间设为最小周期( $2^{13}/f_x$ )。

WDTM



**注** 为了将系统时钟用作看门狗定时器的工作时钟，选项字节必须设为“可通过软件设置停止振荡（LSRSTOP 位设为 1）”。详情参见 [4.1 选项字节设置](#)。

WDTM 寄存器设置值为“01101000（位 7、位 6、及位 5 必须分别设为 0、1、和 1）”。

• 汇编语言

```
MOV WDTM, #01101000B
```

• C 语言

```
WDTM = 0b01101000;
```



**[专栏]** 硬件初始化函数和 main 函数

为了用 C 语言创建程序，CPU 复位后，立即调用硬件初始化函数对外设进行初始化。因此硬件初始化函数主要说明了诸如看门狗定时器设置和时钟频率选择之类的初始设置。

调用硬件初始化函数之后则调用 main 函数，主处理程序在 main 函数中进行描述。

不要由 main 函数调用硬件初始化函数。如果这样的话，硬件初始化函数就会执行两次，只允许设置一次的看门狗定时器设置也会执行两次。结果，在第二次执行期间会产生内部复位信号，使得程序初始化设置不起作用。

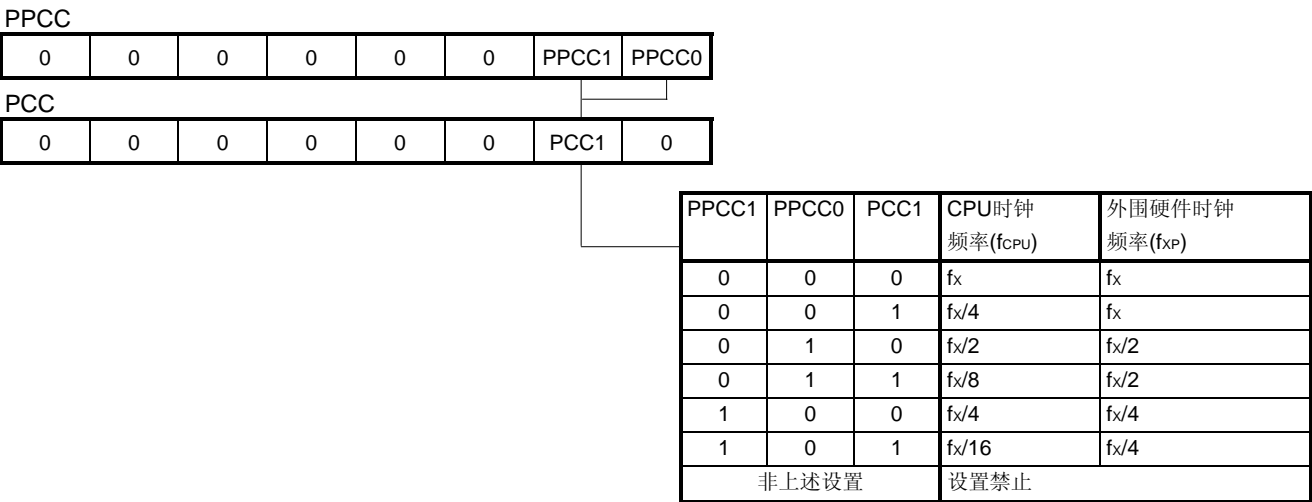
详情参见 NEC Electronics FAQ 网页上 [CC78K0S 语言用户手册](#)和[编程时最先执行的处理](#)。

4.5 时钟设置

(1) 时钟频率设置

由选项字节划分系统时钟设置频率以产生 CPU 时钟频率( $f_{CPU}$ )和提供至外围硬件的时钟频率( $f_{XP}$ )。  
PPCC 寄存器和 PCC 寄存器用来选择 CPU 时钟频率( $f_{CPU}$ )与提供至外围硬件的时钟频率( $f_{XP}$ )。  
本举例程序中 PPCC 寄存器和 PCC 寄存器按照[\[实例 1\]](#)（随后涉及）中的说明进行设置，因此  $f_{CPU} = f_{XP} = 2$  MHz。

图 4-3. 处理器时钟控制寄存器 (PCC)格式和预处理器时钟控制寄存器 (PPCC)格式



- 注意事项 1. PPCC 位 2 至位 7 和 PCC 的位 0 与位 2 至位 7 必须设为 0。  
2. 可用的时钟频率变化范围随电源电压的不同而异。

谐振器	状态	CPU时钟频率( $f_{CPU}$ )	外围硬件的时钟频率( $f_{XP}$ )
陶瓷谐振器、晶体谐振器、外部时钟	4.0至5.5 V	125 kHz ≤ $f_{CPU}$ ≤ 10 MHz	500 kHz ≤ $f_{XP}$ ≤ 10 MHz
	3.0至4.0 V	125 kHz ≤ $f_{CPU}$ ≤ 6 MHz	
	2.7至3.0 V	125 kHz ≤ $f_{CPU}$ ≤ 5 MHz	
	2.0至2.7 V <sup>注</sup>	125 kHz ≤ $f_{CPU}$ ≤ 2 MHz	500 kHz ≤ $f_{XP}$ ≤ 5 MHz
内部高速振荡器	4.0至5.5 V	500 kHz (TYP.) ≤ $f_{CPU}$ ≤ 8 MHz (TYP.)	2 MHz (TYP.) ≤ $f_{XP}$ ≤ 8 MHz (TYP.)
	2.7至4.0 V	500 kHz (TYP.) ≤ $f_{CPU}$ ≤ 4 MHz (TYP.)	
	2.0至2.7 V <sup>注</sup>	500 kHz (TYP.) ≤ $f_{CPU}$ ≤ 2 MHz (TYP.)	2 MHz (TYP.) ≤ $f_{XP}$ ≤ 4 MHz (TYP.)

注 因为加电清除(POC)电路的检测电压( $V_{POC}$ )为 2.1 V ±0.1 V，故在 2.2 至 5.5 V 的电压范围内使用标准型产品和(A)级产品。

因为加电清除(POC)电路的检测电压为 2.26 V(MAX.)，故在 2.26 至 5.5 V 的电压范围内使用(A2)级产品。

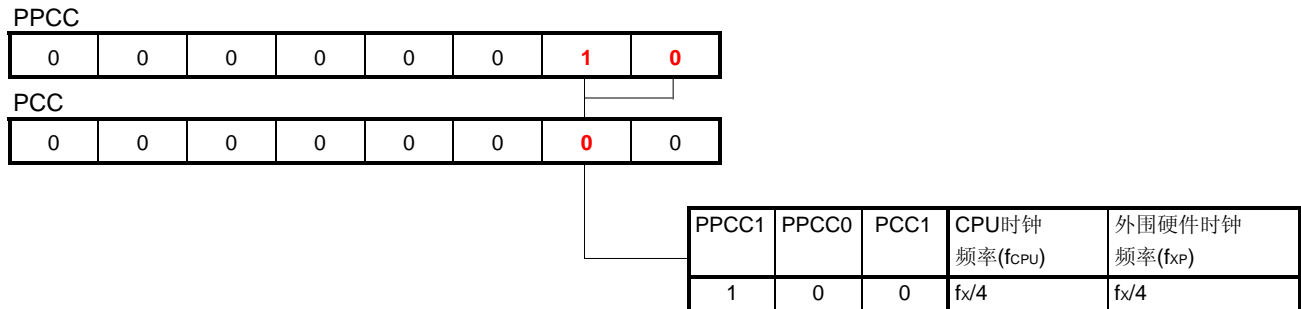
备注  $f_x$ ：系统时钟频率。

所用系统时钟由选项字节来设置。详情参见 [4.1 选项字节设置](#)。

### [实例 1]

$$f_{CPU} = f_{XP} = f_X/4$$

(该值与本举例程序的设置值相同。本举例程序中,“内部高速振荡时钟(8 MHz (TYP.))”通过此选项字节设置选择为系统时钟(fx),因此 CPU 时钟频率(fCPU)和外围硬件时钟频率(fXP)都为 2 MHz (= 8 MHz/4。)



备注

$f_x$ : 系统时钟频率。

所用系统时钟频率由选项字节来设置。详情参见 [4.1 选项字节设置](#)。

PPCC 寄存器设置值为“000000**10**（位 2 至位 7 必须设为 0）”，PCC 寄存器设置值为“000000**00**（位 0 与位 2 至位 7 必须设为 0）”。

对于标准型产品和(A)级产品，加电清除(POC)电路的检测电压( $V_{POC}$ )为  $2.1\text{ V} \pm 0.1\text{ V}$ 。为了符合在“电源电压 $\geq 2.0\text{ V}$ ”的条件（本举例程序中 CPU 时钟频率的设置条件）下会释放 POC，因此，运行本举例程序时，CPU 工作正常。这点与(A2)级产品是相同的。

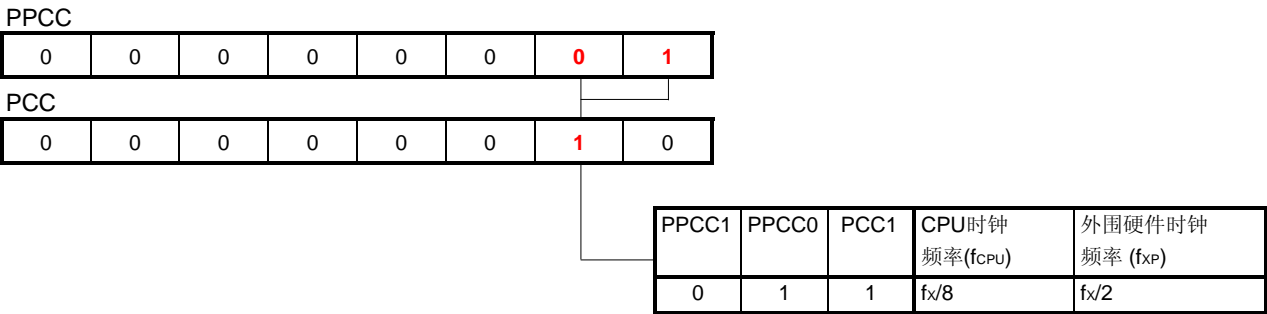
## 汇编语言

```
MOV    PPCC,    #00000010B
MOV    PCC,     #00000000B
```

## C 语言

```
PPCC = 0b00000010;  
PCC = 0b00000000;
```

**[实例 2]**       $f_{CPU} = f_x/8$ ,  $f_{XP} = f_x/2$   
（当“内部高速振荡时钟(8 MHz (TYP.))”由此选项字节设置选择为系统时钟( $f_x$ )时，CPU 时钟频率( $f_{CPU}$ )为 1 MHz (= 8 MHz/8)，而外围硬件时钟频率( $f_{XP}$ )为 4 MHz (= 8 MHz/2)。）



**备注**       $f_x$ : 系统时钟频率。  
所用系统时钟由选项字节来设置。详情参见 [4.1 选项字节设置](#)。

PPCC 寄存器设置值为“0000000**1**（位 2 至位 7 必须设为 0）”，PCC 寄存器设置值为“000000**1**0（位 0 与位 2 至位 7 必须设为 0）”。

可用的时钟频率变化范围随电源电压的不同而异。因此设置 PPCC 和 PCC 时需特别注意。（详情参见[图 4-3.处理器时钟控制寄存器 \(PCC\)格式和预处理器时钟控制寄存器 \(PPCC\)格式](#)。）

汇编语言

```
MOV  PPCC,  #00000001B
MOV  PCC,   #00000010B
```

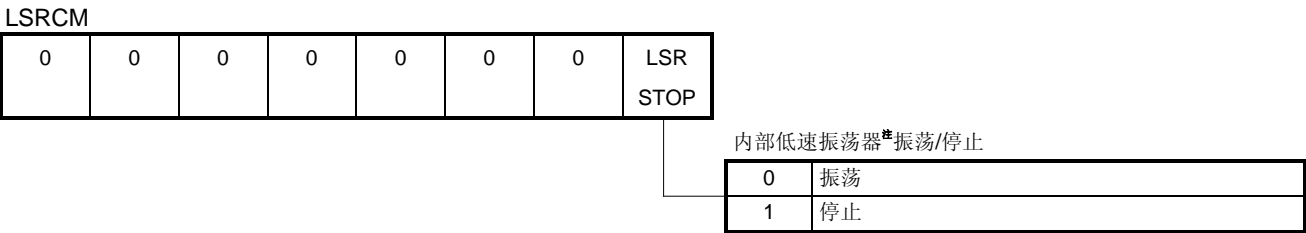
C 语言

```
PPCC = 0b00000001;
PCC = 0b00000010;
```

(2) 内部低速振荡器设置

内部低速振荡时钟用作看门狗定时器和 8 位定时器 H1 的时钟信号源。  
LSRCM 寄存器用来振荡或停止内部低速振荡器。  
本举例程序中，内部低速振荡时钟未用，因此 LSRCM 寄存器按照[\[实例 1\]](#)（随后涉及）中的说明进行设置。

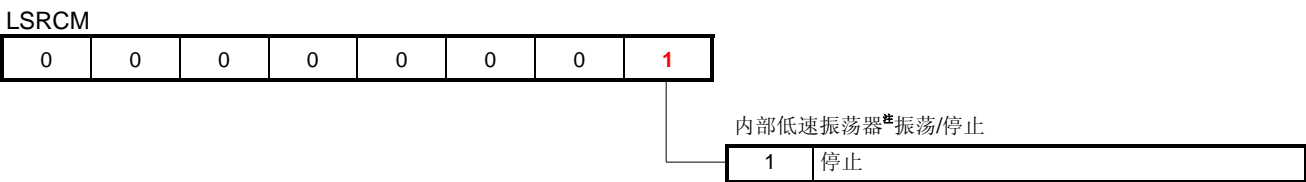
图 4-4. 内部低速振荡模式寄存器(LSRCM)的格式



**注** 为了中止内部低速振荡器，选项字节必须设置为“可通过软件设置停止振荡（LSRSTOP 位设为 1）”。详情参见 [4.1 选项字节设置](#)。

**注意事项** LSRCM 的位 1 至位 7 必须设为 0。

**[实例 1]** 内部低速振荡器振荡停止（同本举例程序中的设置一致）。



**注** 为了停止内部低速振荡器，选项字节必须设置为“可通过软件设置停止振荡（LSRSTOP 位设为 1）”。详情参见 [4.1 选项字节设置](#)。

LSRCM 寄存器设置值为“00000001（位 7 至位 1 必须设为 0）”。

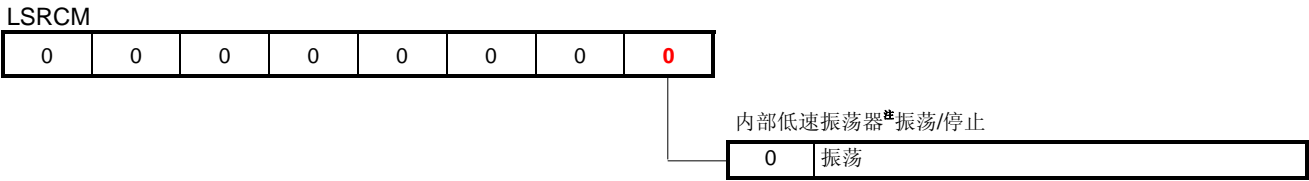
汇编语言

```
MOV  LSRCM,  #00000001B
```

C 语言

```
LSRCM = 0b00000001;
```

[实例 2] 内部低速振荡器振荡



注 当内部低速振荡器的振荡由选项字节设置为“不能停止”时，LSRCM 设置无效（不必理会）。详情参见 [4.1 选项字节设置](#)。

LSRCM 寄存器设置值为“00000000（位 7 至位 1 必须设为 0）”。

• 汇编语言

```
MOV LSRCM, #00000000B
```

• C 语言

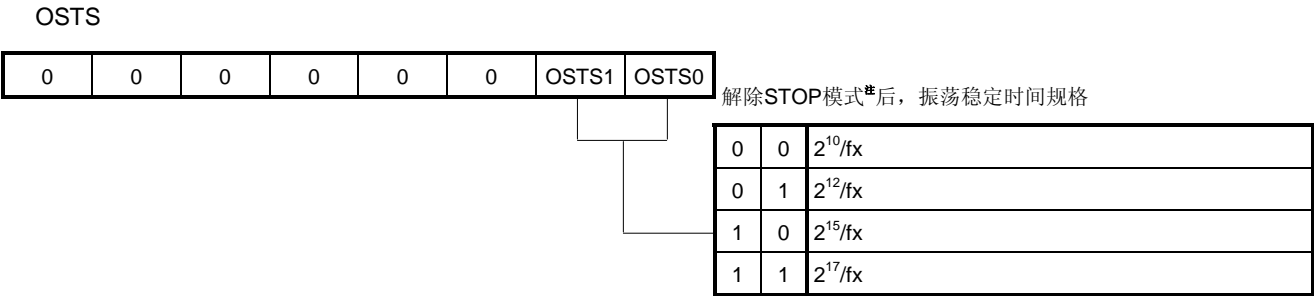
```
LSRCM = 0b00000000;
```

(3) 解除 STOP 模式后振荡稳定时间的设置（当使用晶体振荡或陶瓷振荡作为系统时钟信号源时）

OSTS 寄存器用于设置解除 STOP 模式后的振荡稳定时间。仅当使用晶体振荡或陶瓷振荡作为系统时钟信号源时，OSTS 设置有效。

本举例程序中，使用内部高速振荡时钟，因此不必设置 OSTS 寄存器。

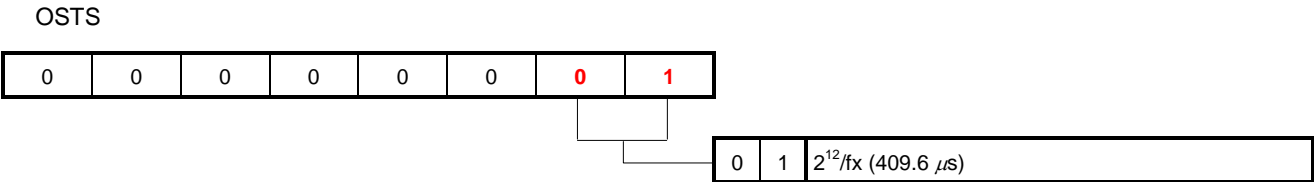
图 4-5. 振荡稳定时间选择寄存器(OSTS)的格式



注 当内部高速振荡时钟或外部时钟输入选择为系统时钟时，无需设置 OSTS 寄存器。

注意事项 位 7 至位 2 必须设为 0。

【实例 1】 当使用晶体振荡时钟或陶瓷振荡时钟时( $f_x = 10\text{ MHz}$ )，将振荡稳定时间设置为  $409.6\text{ }\mu\text{s}$ 。



OSTS 寄存器设置值为“00000001（位 7 至位 2 必须设为 0）”。

• 汇编语言

```
MOV OSTS, #00000001B
```

• C 语言

```
OSTS = 0b00000001;
```



4.6 端口设置

注意事项 片上端口随不同的产品而异，因此端口的设置也会有所不同。

	78K0S/KA1+	78K0S/KB1+	78K0S/KU1+	78K0S/KY1+
端口0	—	P00至P03	—	—
端口2	P20至P23	P20至P23	P20至P23	P20至P23
端口3	P30、P31、P34 <sup>※</sup>	P30至P33、P34 <sup>※</sup>	P32、P34 <sup>※</sup>	P32、P34 <sup>※</sup>
端口4	P40至P45	P40至P47	P40、P43	P40至P47
端口12	P121至P123	P120至P123	—	—
端口13	P130	P130	—	—

注 P34 与 RESET 共用一个端口。由选项字节设置来选择所使用的功能。详情参见 4.1 选项字节设置。

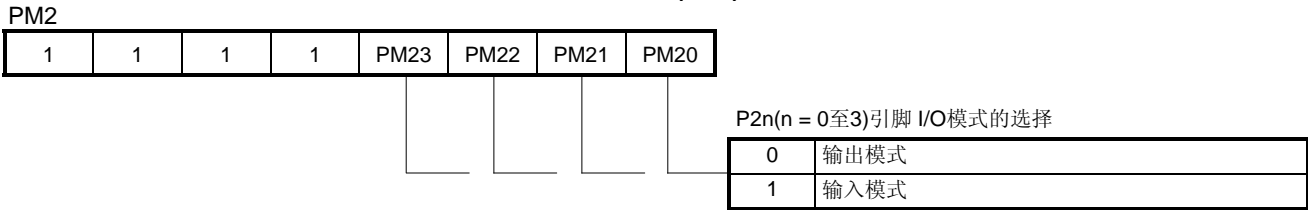
(1) I/O 端口设置

PMxx 寄存器用来设置 I/O 端口是用作输入端口还是输出端口。复位解除后，端口设置为输入端口。

用 PM2 寄存器举例来介绍 PMxx 格式。

本举例程序中，端口 2 按照[实例 1]（随后涉及）中的说明进行设置，而端口 4 按照[实例 2]。

图 4-6. 端口模式寄存器 2(PM2)的格式



注意事项 位 7 至位 4 必须设为 1。

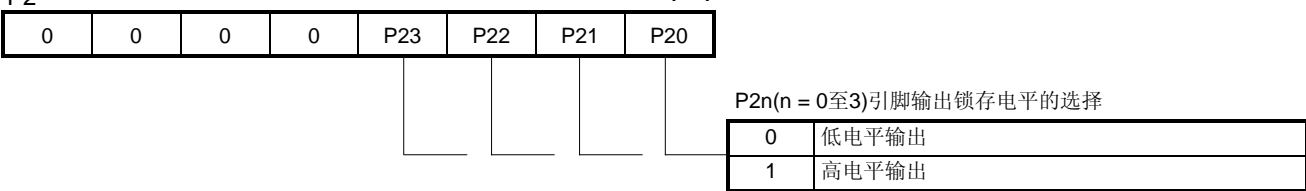
(2) 输出端口输出锁存设置

Pxx 寄存器用于将输出端口的输出锁存设置为高电平或低电平。复位解除后，输出端口的输出锁存设置为低电平输出。

用 P2 寄存器举例来介绍 Pxx 格式。

本举例程序中，端口 2 按照[实例 1]（随后涉及）中的说明进行设置。

图 4-7. 端口寄存器 2(P2)的格式



注意事项 位 7 至位 4 必须设为 0。

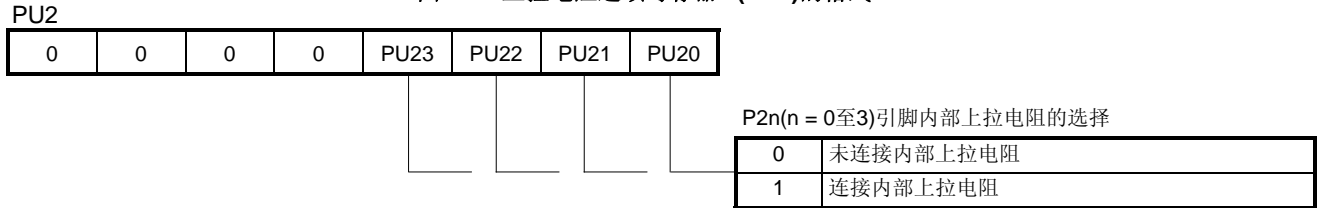
**(3) 设置内部上拉电阻至输入端口的连接**

PUxx 寄存器用来设置内部上拉电阻是否连接至输入端口。复位解除后内部上拉电阻未连接。

用 PU2 寄存器举例来说明 PUxx 格式。

本举例程序中，端口 4 按照[\[实例 2\]](#)（随后涉及）中的说明进行设置。

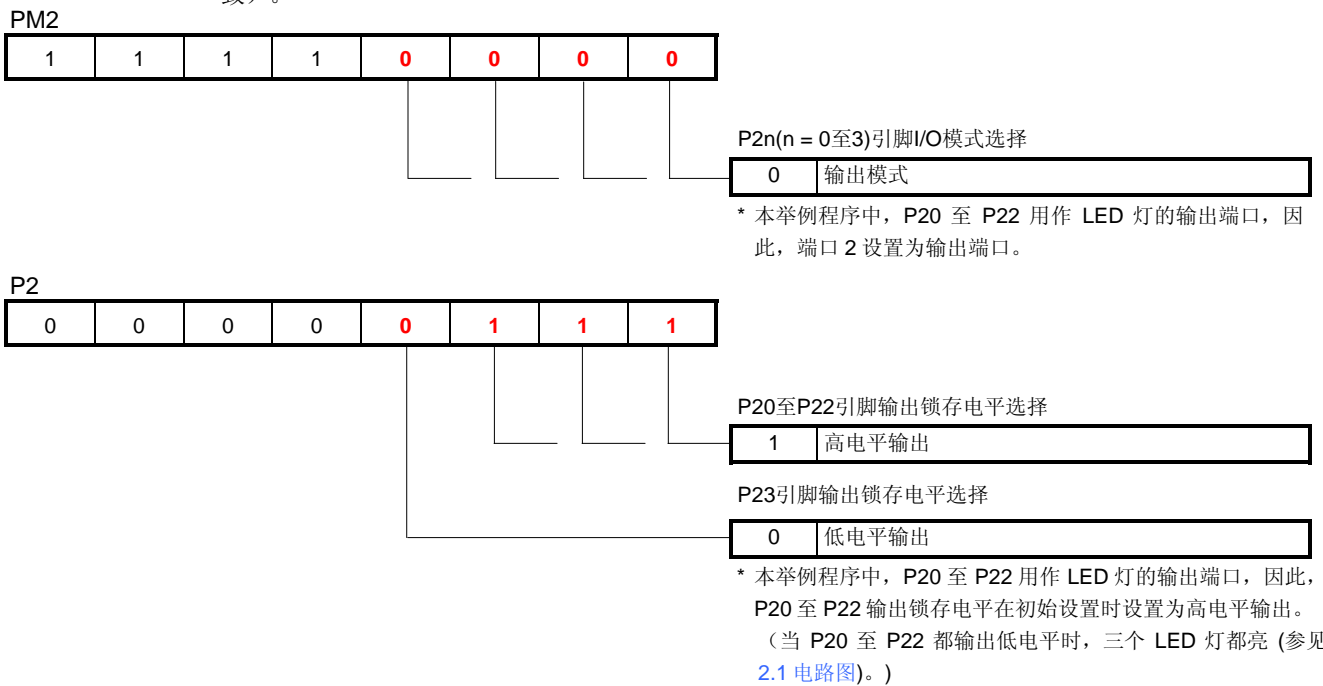
图 4-8. 上拉电阻选项寄存器 2(PU2)的格式



**注意事项** 位 7 至位 4 必须设为 0。

**[实例 1]**

- 将 P20 至 P23 设置为输出端口。
- 将 P20 至 P22 输出锁存设置为高电平输出，设置 P23 输出锁存为低电平输出（同本举例程序中的设置一致）。



PM2 寄存器设置值为“11110000（位 7 至位 4 必须设为 1）”，P2 寄存器设置值为“00001111（位 7 至位 4 必须设为 0）”。

## • 汇编语言

```
MOV PM2, #11110000B
MOV P2, #00001111B
```

## • C 语言

```
PM2 = 0b11110000;
P2 = 0b00001111;
```

- [实例 2]
- 将 P40 和 P43 设置为输入端口。
  - 设置内部上拉电阻连接至 P40 和 P43。
- (同本举例程序中的设置一致)

PM4

• 78K0S/KB1+, 78K0S/KY1+

x	x	x	x	1	x	x	1
---	---	---	---	---	---	---	---

P40和P43引脚I/O模式选择

1	输入模式
---	------

• 78K0S/KA1+

0 <sup>※</sup>	0 <sup>※</sup>	x	x	1	x	x	1
----------------	----------------	---	---	---	---	---	---

P40和P43引脚I/O模式选择

1	输入模式
---	------

注 复位解除后，位 7 和位 6 设置为 1。在初始设置中，它们必须设为 0。

• 78K0S/KU1+

0 <sup>※</sup>	0 <sup>※</sup>	0 <sup>※</sup>	0 <sup>※</sup>	1	0 <sup>※</sup>	0 <sup>※</sup>	1
----------------	----------------	----------------	----------------	---	----------------	----------------	---

P40和P43引脚I/O模式选择

1	输入模式
---	------

\* 本举例程序中，P40 和 P43 用作开关输入端口，因此，P40 和 P43 设置为输入端口。

注 复位后，位 7 至位 4、位 2、及位 1 设为 1。在初始设置中，它们必须设为 0。

PU4

• 78K0S/KB1+, 78K0S/KY1+

x	x	x	x	1	x	x	1
---	---	---	---	---	---	---	---

内部上拉电阻至P40引脚和P43引脚的连接选择

1	连接内部上拉电阻。
---	-----------

• 78K0S/KA1+

0	0	x	x	1	x	x	1
---	---	---	---	---	---	---	---

内部上拉电阻至P40引脚和P43引脚的连接选择

1	连接内部上拉电阻。
---	-----------

注意事项 位 7 和位 6 必须设为 0。

78K0S/KU1+

0	0	0	0	1	0	0	1
---	---	---	---	---	---	---	---

内部上拉电阻至P40引脚和P43引脚的连接选择

1	连接内部上拉电阻。
---	-----------

注意事项 位 7 至位 4、位 2、及位 1 必须设为 0。

\* 本举例程序中，为了控制 P40 和 P43 的输出电平，将内部上拉电阻连接至 P40 和 P43。（当开关断开时选择为高电平输入，当开关闭合时选择为低电平输入（参见 2.1 电路图）。）

78K0S/KB1+和 78K0S/KY1+的 PM4 寄存器设置值为“xxxx1xx1（x：不必理会）”，对于 78K0S/KA1+的 PM4 寄存器设置值为“00xx1xx1（x：不必理会）”，而对于 78K0S/KU1+的 PM4 寄存器设置值为则为“00001001”。78K0S/KB1+ 和 78K0S/KY1+的 PU4 寄存器设置值为“xxxx1xx1（x：不必理会）”，对于 78K0S/KA1+的 PU4 寄存器设置值为“00xx1xx1（x：不必理会）”，而对于 78K0S/KU1+的 PU4 寄存器设置值为则为“00001001”。（上述实例中，PM4 寄存器和 PU4 寄存器中“x”的值设为 0。）

- 汇编语言

```
MOV PM4, #00001001B  
MOV PU4, #00001001B
```

- C 语言

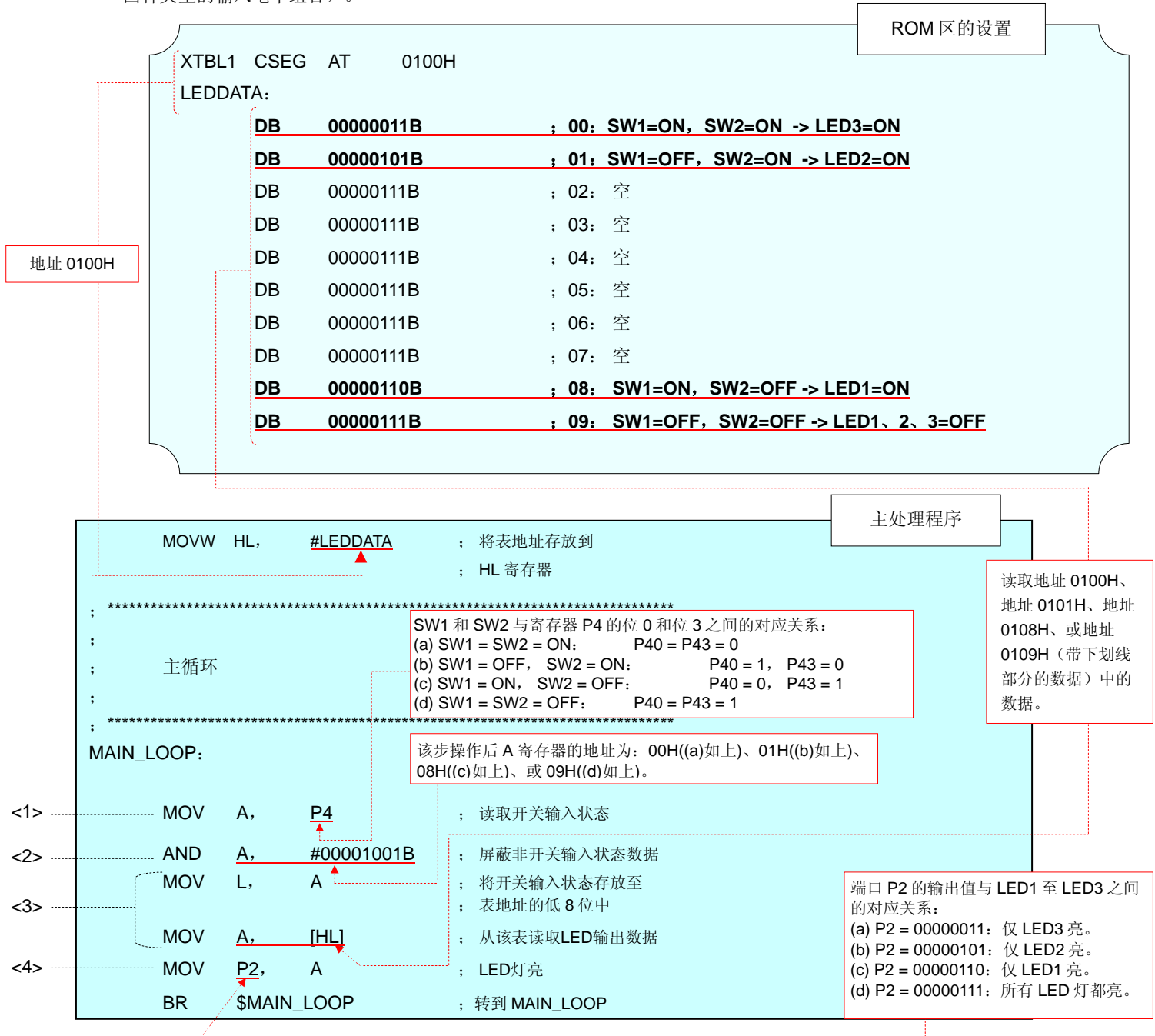
```
PM4 = 0b00001001;  
PU4 = 0b00001001;
```

## 4.7 主处理程序

在汇编语言中，由主处理程序来执行下列操作：

- <1> 读取 P4 寄存器的数据。
- <2> 在读取的 8 位数据之中，除输入端口位（P40、P43），将其余位都清为 0。
- <3> 输出数据根据 P40 和 P43 的组合从地址 0100H 至地址 0109H（“LEDDATA”表）中选取。
- <4> 将所选取的数据输出至 P2 寄存器。

经过操作<1>和操作<2>，仅能够识别连接至开关（SW1 和 SW2）的 P40 引脚和 P43 引脚的输入电平组合。在操作<3>的地址 0100H 至地址 0109H 之中，仅使用 0100H、0101H、0108H、及 0109H 四个地址中数据。（因为只有四种类型的输入电平组合）。



C 语言中主处理程序与汇编语言中主处理程序的操作相似。

在 C 语言中，输入数据和输出数据对应存放于同一数组中。

```
/* *****
```

Main loop

```
*****
```

```
void main(void){
```

```
    const unsigned char OUTDATA[10] =  
    {0x03, 0x05, 0x07, 0x07, 0x07, 0x07, 0x07, 0x07, 0x06, 0x07};
```

```
                                /*LED输出型数组*/
```

```
    unsigned char INDATA;          /*开关输入变量*/
```

```
    while(1){
```

```
        INDATA = P4 & 0b00001001;    /*合法开关输入状态*/
```

```
        P2 = OUTDATA[INDATA];      /*从表中读取LED输出数据*/
```


```
    }
```

花括号中定义了用于放置输出数据的十个数据单元。但是，该主处理程序中，仅将带下划线的部分用作输出数据。（非下划线的部分为数据哑元。）

输入数据和输出数据之间的对应关系如下：


开关输入	P40, P43	INDATA	OUTDATA	LED 灯
SW1=ON, SW2=ON	P40=0, P43=0	0b00000000	0x03	仅 LED3 亮。
SW1=OFF, SW2=ON	P40=1, P43=0	0b00000001	0x05	仅 LED2 亮。
SW1=ON, SW2=OFF	P40=0, P43=1	0b00001000	0x06	仅 LED1 亮。
SW1=OFF, SW2=OFF	P40=1, P43=1	0b00001001	0x07	所有 LED 灯都亮。

## 第五章 使用系统仿真器SM+进行操作检查

选择 图标  所下载的汇编语言文件，本章介绍了使用 78K0S/Kx1+的系统仿真器 SM+，举例程序如何运行。

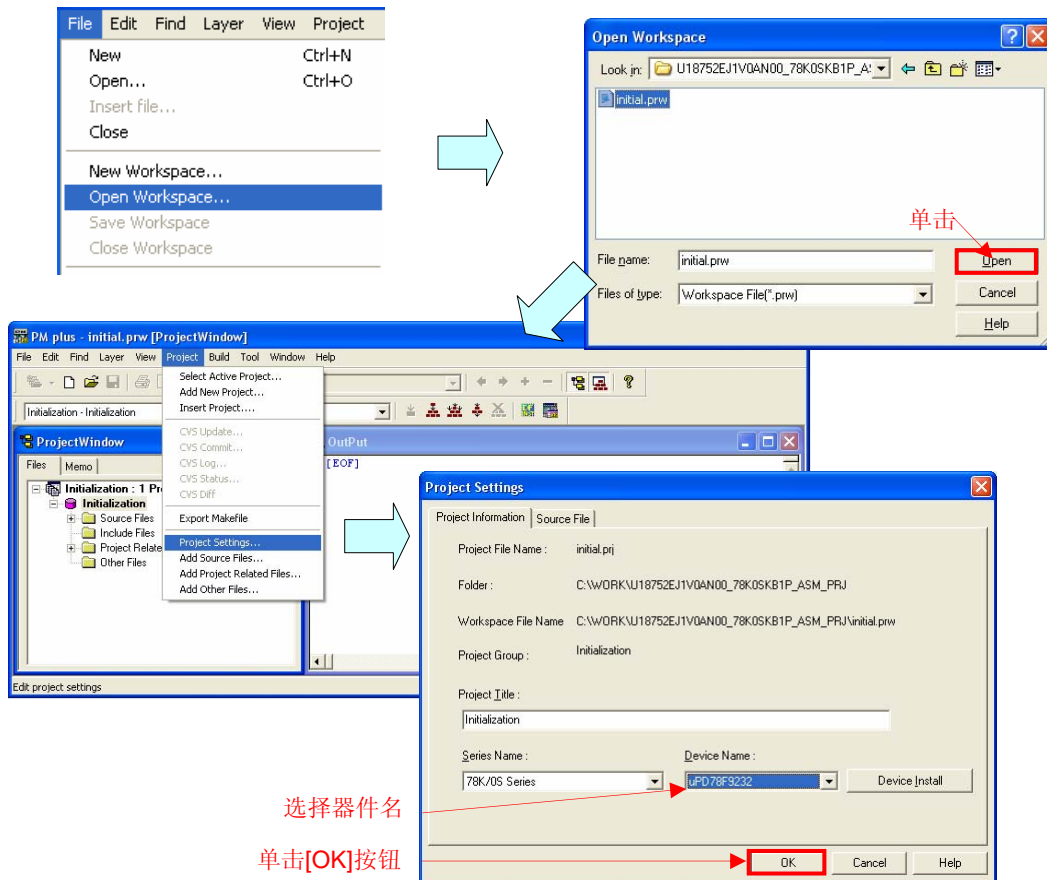
**注意事项** 78K0S/Kx1+的系统仿真器 SM+不支持 78K0S/KU1+微控制器（同 2007 年 04 月）。因此 78K0S/KU1+微控制器的操作不能由 78K0S/Kx1+的系统仿真器 SM+来检查。


### 5.1 创建举例程序

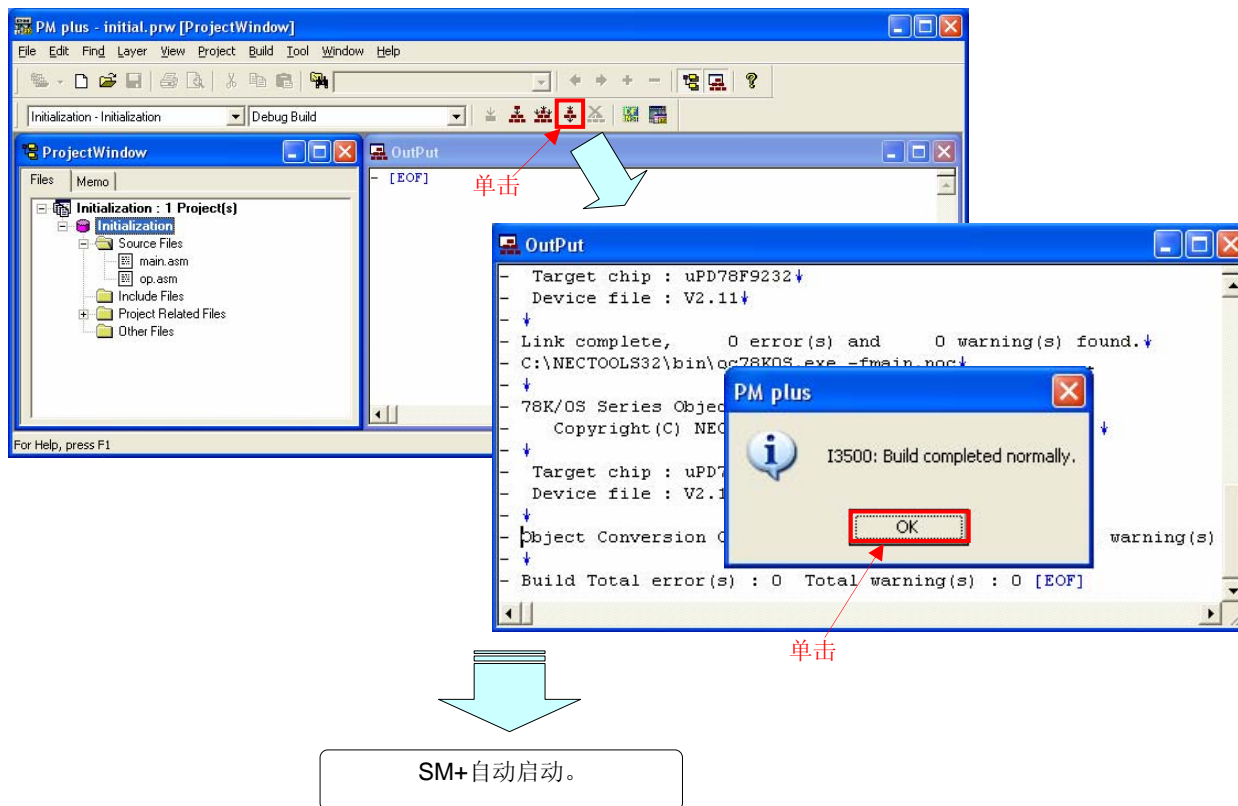
为利用 78K0S/Kx1+的系统仿真器 SM+来检查举例程序的操作（以下简称“SM+”），SM+必须在创建举例程序之后启动。本节介绍了操作顺序实例：从由集成开发环境 PM+创建举例程序开始，然后使用由选择  图标所下载的汇编语言文件，直到启动 SM+。

如何操作 PM+之详情，参见 [PM+工程管理用户手册](#)。

- (1) 启动 PM+。
- (2) 从[File]菜单中单击[Open Workspace]选项，选择“initial.prw”并单击[Open]按钮，于是就在自动读取的源文件内创建了一个工作区。
- (3) 从[Project]菜单中选择[Project Settings]选项。当打开[Project Settings]选项窗口时，选择所用的器件名（器件 ROM 或 RAM 的最大容量选用缺省值），然后单击[OK]按钮。



- (4) 单击  ([Build → Debug]按钮)。当“main.asm”和“op.asm”源文件正常编译完成后，将显示“I3500: Build completed normally”的信息。
- (5) 单击信息框中的[OK]按钮以自动启动 SM+。

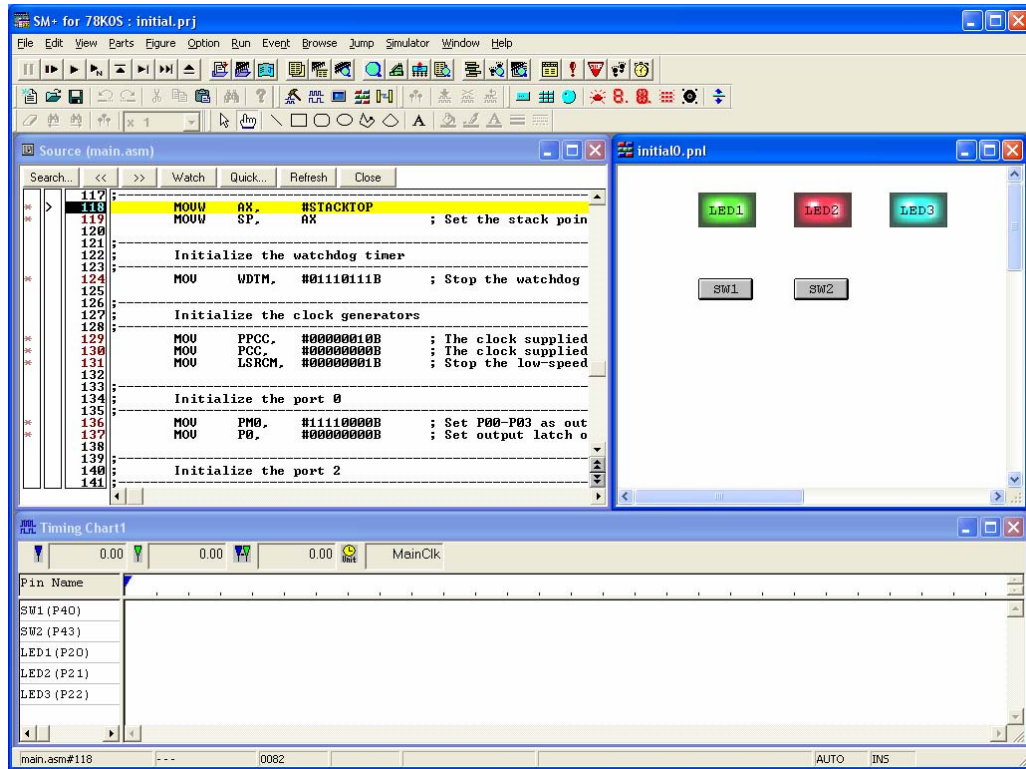


## 5.2 SM+的操作

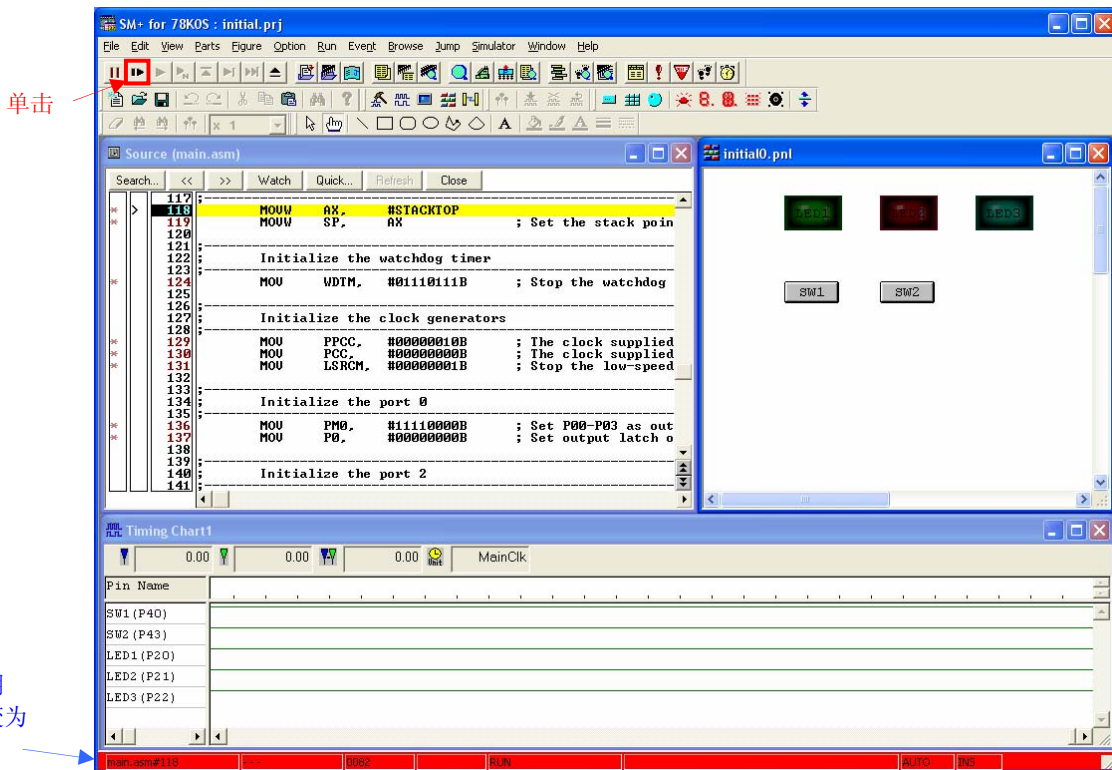
本节介绍了检查关于 SM+ 之 I/O 面板窗口或时序图窗口操作的实例。  
如何操作 SM+ 之详情，参见 [SM+系统仿真器操作用户手册](#)。



- (1) 在 PM+ 窗口中单击 [Build → Debug] 启动 SM+ 后（参见 5.1），将显示以下界面：（此界面为使用汇编语言源文件时的界面。）

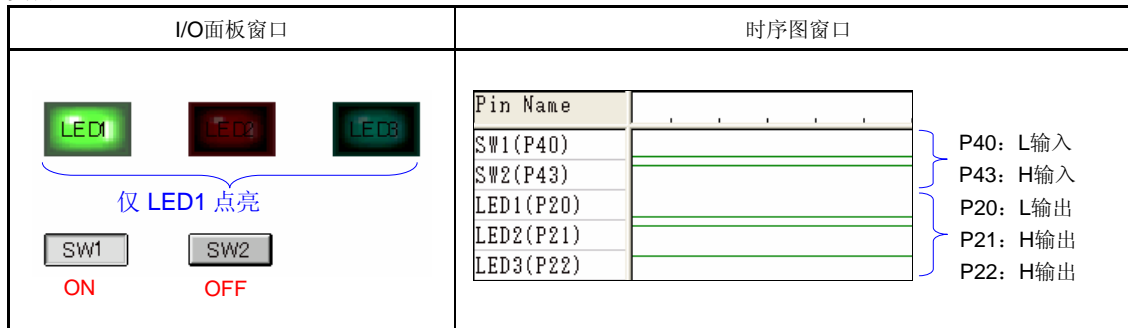


- (2) 单击  ([Restart]按钮)。CPU 复位后将执行程序同时显示如下界面：

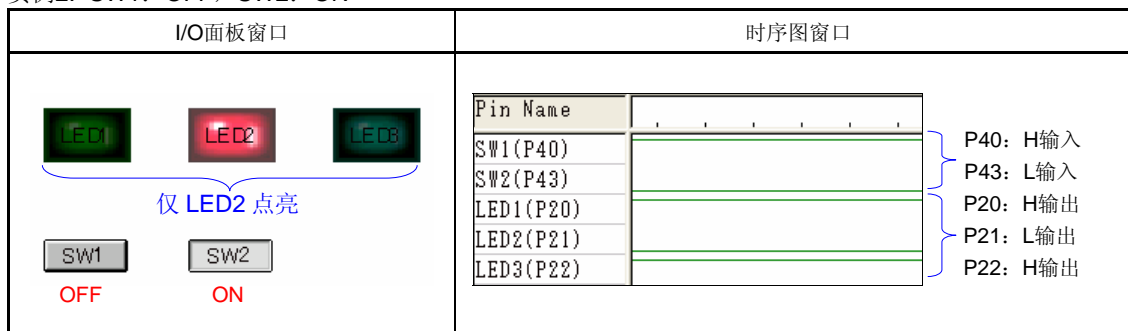


(3) 程序执行期间，在 I/O 面板窗口中单击[SW1]按钮和[SW2]按钮。根据[SW1]按钮和[SW2]按钮的组合，在 I/O 面板窗口中观察[LED1]灯至[LED3]灯的变化，同时观测时序图窗口中的波形变动。

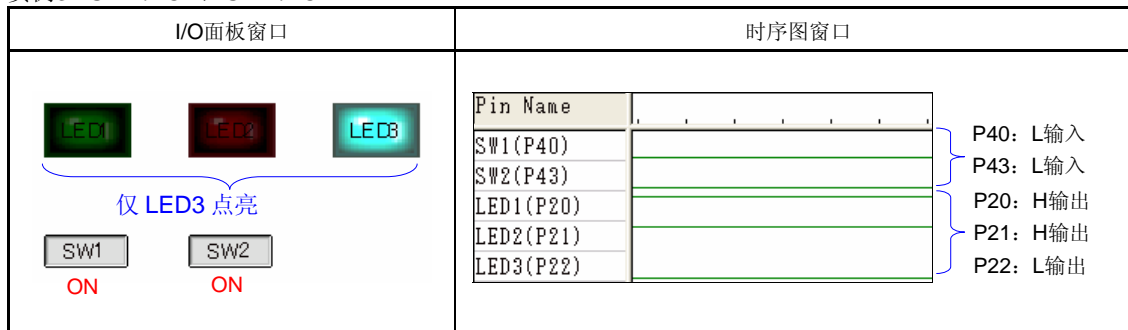
实例1. SW1: ON, SW2: OFF



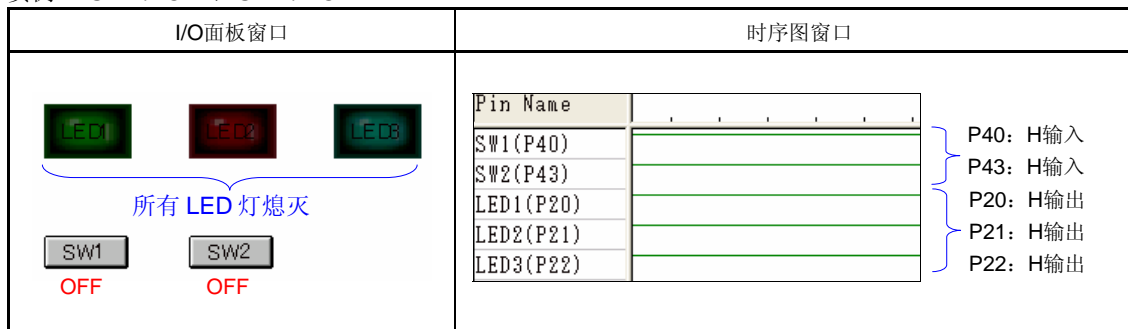
实例2. SW1: OFF, SW2: ON



实例3. SW1: ON, SW2: ON



实例4. SW1: OFF, SW2: OFF



备注 H: 高电平, L: 低电平。

[补充] 利用 SM+ 的监控功能可检查端口 2 和端口 4 的数值变动。

- <1> 从[Browse]菜单中选择[Watch]选项以打开[Watch]窗口。
- <2> 单击[Add]以打开[Add Watch]对话框。(此时, [Watch]窗口仍然处于打开状态。)
- <3> 在[Name]域中键入“P4”, 然后单击[Add]按钮以将寄存器“P4”添加到[Watch]窗口中。(此时, [Add Watch]窗口仍然保持打开状态。)
- <4> 接着, 在[Name]域中键入“P2”, 然后单击[OK]按钮以将寄存器“P2”添加到[Watch]窗口中, 同时[Add Watch]窗口会随之关闭。

键入“P4”，然后单击[Add]按钮。

单击

键入“P2”，然后单击[OK]按钮。

P2 和 P4 已注册。

此为 CPU 复位后的界面实例。CPU 复位后, P2 和 P4 的数值如下:  
P2: 0x00  
P4: 0x10

- <5> 执行程序, 并单击 I/O 面板窗口中的[SW1]按钮和[SW2]按钮。根据[SW1]按钮和[SW2]按钮的组合方式, 观察[Watch]窗口中 P2 和 P4 数值的变化。

SW1和SW2的组合	[Watch]窗口中的数值
SW1: ON, SW2: OFF	P2: 0x06, P4: 0x08
SW1: OFF, SW2: ON	P2: 0x05, P4: 0x01
SW1: ON, SW2: ON	P2: 0x03, P4: 0x00
SW1: OFF, SW2: OFF	P2: 0x07, P4: 0x09

## 第六章 相关文档

文档名		日语/英语
78K0S/KU1+用户手册		<a href="#">PDF</a>
78K0S/KY1+用户手册		<a href="#">PDF</a>
78K0S/KA1+用户手册		<a href="#">PDF</a>
78K0S/KB1+用户手册		<a href="#">PDF</a>
78K/0S系列指令用户手册		<a href="#">PDF</a>
RA78K0S汇编语言程序包用户手册	语言	<a href="#">PDF</a>
	操作	<a href="#">PDF</a>
CC78K0S C 编译器用户手册	语言	<a href="#">PDF</a>
	操作	<a href="#">PDF</a>
PM+工程管理用户手册		<a href="#">PDF</a>
SM+ 系统仿真器操作用户手册		<a href="#">PDF</a>
举例程序启动指南（预备名）		准备中

## 附录A 程序清单

78K0S/KB1+微控制器源程序作为程序实例如下所示：

### ● main.asm（汇编语言版本）

```
*****
;
;
;       NEC Electronics   78K0S/KB1+
;
; *****
;       78K0S/KB1+   举例程序
; *****
;       初始化
; *****
; <<历史记录>>
;       2007.4.--   发布
; *****
;
; <<概要>>
;
; 本举例程序由诸如时钟频率和输入或输出端口之类的设置功能来初始化微控制器。
;
; 初始化之后，三个 LED 灯由两个开关来控制。
;
; <初始化中的主要设置内容>
; - 设置向量表
; - 设置堆栈指针
; - 停止看门狗定时器的操作
; - 设置 CPU 时钟频率为 2 MHz
; - 停止内部低速振荡器
; - 设置端口为输入或输出模式。
; - 设置片上上拉电阻的连接（仅输入端口）。
; - 设置输出端口的输出锁存
; <开关输入和 LED 输出>
;
; +-----+
; | SW1 | SW2 | LED1 | LED2 | LED3 |
; | (P40) | (P43) | (P20) | (P21) | (P22) |
; |-----|-----|
; | OFF | OFF | OFF | OFF | OFF |
; | ON  | OFF | ON  | OFF | OFF |
; | OFF | ON  | OFF | ON  | OFF |
; | ON  | ON  | OFF | OFF | ON  |
; +-----+
; <<I/O 端口设置>>
; 输入： P40、P43
; 输出： P00-03、P20-23、P30-33、P41、P42、P44-P47、P120-P123、P130
; # 将所有未用端口设置为输出模式。
;
; *****
```

```

; =====
;
;   向量表
;
; =====
XVCT CSEG AT 0000H
    DW RESET_START      ; (00) RESET
    DW RESET_START      ; (02) --
    DW RESET_START      ; (04) --
    DW RESET_START      ; (06) INTLVI
    DW RESET_START      ; (08) INTP0
    DW RESET_START      ; (0A) INTP1
    DW RESET_START      ; (0C) INTTMH1
    DW RESET_START      ; (0E) INTTM000
    DW RESET_START      ; (10) INTTM010
    DW RESET_START      ; (12) INTAD
    DW RESET_START      ; (14) --
    DW RESET_START      ; (16) INTP2
    DW RESET_START      ; (18) INTP3
    DW RESET_START      ; (1A) INTTM80
    DW RESET_START      ; (1C) INTSRE6
    DW RESET_START      ; (1E) INTSR6
    DW RESET_START      ; (20) INTST6

; =====
;
;   定义 ROM 数据表
;
; =====
XTBL1 CSEG AT 0100H
LEDDATA:
    DB 00000011B        ; 00: SW1=ON, SW2=ON -> LED3=ON
    DB 00000101B        ; 01: SW1=OFF, SW2=ON -> LED2=ON
    DB 00000111B        ; 02: 空
    DB 00000111B        ; 03: 空
    DB 00000111B        ; 04: 空
    DB 00000111B        ; 05: 空
    DB 00000111B        ; 06: 空
    DB 00000111B        ; 07: 空
    DB 00000110B        ; 08: SW1=ON, SW2=OFF -> LED1=ON
    DB 00000111B        ; 09: SW1=OFF, SW2=OFF -> LED1、2、3=OFF

; =====
;
;   定义内存堆栈区
;
; =====
XSTK DSEG AT 0FEE0H
STACKEND:
    DS 20H                ; 内存堆栈区 = 32 字节
STACKTOP:                 ; 内存堆栈区的起始地址 = FF00H

```

```

; *****
;
;      复位后的初始化
;
; *****
XMAIN CSEG UNIT
RESET_START:

; -----
;      初始化堆栈指针
; -----
      MOVW AX,    #STACKTOP
      MOVW SP,    AX          ; 设置堆栈指针为 FF00H

; -----
;      初始化看门狗定时器
; -----
      MOV  WDTM,    #01110111B  ; 停止看门狗定时器的操作

; -----
;      初始化时钟发生器
; -----
      MOV  PPCC,    #00000010B  ; 供应至外围硬件的时钟
                                   ; (fxp) = fx/4 (= 2MHz)
      MOV  PCC,    #00000000B  ; 供应至 CPU 的时钟(fcpu) = fxp
                                   ; (= 2MHz)
      MOV  LSRCM, #00000001B  ; 停止低速内部振荡器

; -----
;      初始化端口 0
; -----
      MOV  PM0,    #11110000B  ; 将 P00-P03 设为输出模式
      MOV  P0,     #00000000B  ; 设置 P00-P03 的输出锁存为低电平

; -----
;      初始化端口 2
; -----
      MOV  PM2,    #11110000B  ; 将 P20-P23 设为输出模式
      MOV  P2,     #00000111B  ; 设置 P20-P22 的输出锁存为高电平, P23 为
                                   ; 低电平

; -----
;      初始化端口 3
; -----
      MOV  PM3,    #11110000B  ; 将 P30-P33 设为输出模式
      MOV  P3,     #00000000B  ; 设置 P30-P33 的输出锁存为低电平

; -----
;      初始化端口 4
; -----
      MOV  PM4,    #00001001B  ; 将 P40 和 P43 设置为输入模式, 将 P41、P42 和

```

```

; -----
MOV  PU4,  #00001001B ; P44-P4 设置为输出模式
; 将片上上拉电阻连接至 P40 和
; P43
MOV  P4,   #00000000B ; 设置 P41、P42 及 P44-P47 的输出锁存为
; 低电平

; -----
; 初始化端口 12
; -----
MOV  PM12,  #11110000B ; 将 P120-P123 设置为输出模式
MOV  P12,   #00000000B ; 设置 P120-P123 的输出锁存为低电平

; -----
; 初始化端口 13
; -----
MOV  P13,   #00000001B ; 设置 P130 的输出锁存为高电平

; -----
; 初始化通用寄存器
; -----
MOVW HL,    #LEDDATA ; 将表地址存放到 HL 寄存器

; *****
;
; 主循环
;
; *****
MAIN_LOOP:
MOV  A,  P4 ; 读取开关输入状态
AND  A,  #00001001B ; 屏蔽非开关输入的状态
MOV  L,  A ; 将开关输入状态存放到表地址的
; 低 8 位
MOV  A,  [HL] ; 从表中读取 LED 输出数据
MOV  P2,  A ; 输出 LED 灯
BR   $MAIN_LOOP ; 转到 MAIN_LOOP

end

```



## ● main.c (C 语言版本)

```
/******
```

NEC Electronics 78K0S/KB1+

```
*****
```

78K0S/KB1+ 举例程序

```
*****
```

初始化

```
*****
```

<<历史记录>>

2007.4.-- 发布

```
*****
```

<<概要>>

本举例程序通过诸如时钟频率和输入或输出之类的设置功能来初始化微控制器。

初始化后，三个 LED 灯由两个开关来控制。

<初始化中的主要设置内容>

- 停止开门狗定时器的操作。
- 将 CPU 时钟频率设置为 2 MHz。
- 停止内部低速振荡器。
- 将端口设置为输入或输出模式。
- 设置片上上拉电阻的连接（仅输入端口）。
- 设置输出端口的输出锁存。

<开关输入和 LED 输出>

```
+-----+
| SW1 | SW2 | LED1 | LED2 | LED3 |
| (P40) | (P43) | (P20) | (P21) | (P22) |
|-----|
| OFF | OFF | OFF | OFF | OFF |
| ON  | OFF | ON  | OFF | OFF |
| OFF | ON  | OFF | ON  | OFF |
| ON  | ON  | OFF | OFF | ON  |
+-----+
```

<<I/O 端口设置>>

输入： P40、P43

输出： P00-03、P20-23、P30-33、P41、P42、P44-P47、P120-P123、P130

#将所有未用端口设置为输出模式。

```
*****/
```

```
/*=====
```

预处理指示符(#pragma)

```

=====*/
#pragma      SFR                      /* SFR 名称可在 C 源程序层面上进行描述 */

/*****

RESET 后的初始化

*****/
void hdwinit(void){
/*-----
    初始化看门狗定时器
-----*/
    WDTM = 0b01110111;          /* 停止看门狗定时器的操作 */

/*-----
    初始化时钟发生器
-----*/
    PPCC = 0b00000010;          /* 提供至外围 */
                                /* 硬件的时钟(fxp) = fx/4 (= 2MHz) */
    PCC  = 0b00000000;          /* 提供至 CPU 的时钟 (fcpu) = */
                                /* fxp (= 2MHz) */
    LSRCM = 0b00000001;         /* 停止内部低速振荡器 */

/*-----
    初始化端口 0
-----*/
    PM0 = 0b11110000;           /* 将 P00-P03 设置为输出模式 */
    P0  = 0b00000000;           /* 设置 P00-P03 的输出锁存为低电平 */

/*-----
    初始化端口 2
-----*/
    PM2 = 0b11110000;           /* 将 P20-P23 设置为输出模式 */
    P2  = 0b00000111;           /* 设置 P20-P22 的输出锁存为高电平, */
                                /* P23 的从输出锁存为低电平 */

/*-----
    初始化端口 3
-----*/
    PM3 = 0b11110000;           /* 将 P30-P33 设置为输出模式 */
    P3  = 0b00000000;           /* 将 P30-P33 的输出锁存设置为低电平 */

/*-----
    初始化端口 4
-----*/
    PM4 = 0b00001001;           /* 设置 P40 和 P43 为输入模式, P41、 */
                                /* P42 和 P44-P47 为输出模式 */
    PU4 = 0b00001001;           /* 连接片上上拉电阻至 */
                                /* P40 和 P43 */
    P4  = 0b00000000;           /* 设置 P41、P42 和 */
                                /* P44-P47 的输出锁存为低电平 */

/*-----
    初始化端口 12
-----*/

```

```

PM12 = 0b11110000;          /* 将 P120-P123 设置为输出模式 */
P12  = 0b00000000;          /* 设置 P120-P123 的输出锁存为 */
                              /* 低电平 */

/*-----
   初始化端口 13
-----*/
P13  = 0b00000001;          /* 设置 P130 的输出锁存为 */
                              /* 高电平 */
}

/*****

Main loop

*****/
void main(void){

    const unsigned char OUTDATA[10] = {0x03, 0x05, 0x07, 0x07, 0x07, 0x07, 0x07,
                                         0x07, 0x06, 0x07};

    /* Array of LED output pattern */
    unsigned char INDATA;          /* 开关输入变量 */

    while(1){
        INDATA = P4 & 0b00001001;    /* 有效输入状态 */
        P2 = OUTDATA[INDATA];        /* 从表中读取 LED 输出数据 */
    }
}

```

● op.asm （汇编语言和 C 语言通用）

```

; =====
;
;      选项字节
;
; =====
OPBT  CSEG  AT      0080H
      DB    10011100B    ; 选项字节区域。
;
;      |||
;      |||+----- 内部低速振荡器
;                  可由软件停止。
;      |++----- 内部高速振荡时钟(8MHz)
;                  选择为系统时钟信号源。
;      +----- P34/RESET 引脚用作 RESET 引脚。

      DB    11111111B    ; 保护字节区域 （用于自编程
;                        ; 模式）
;
;      |||||
;      ++++++----- 所有模块都能被写入或擦除。

end

```

附录B 版本修订历史

版本	发布日期	页码	修订
第一版	2008年02月	-	-

详细信息请联系:

**中国区**

**MCU 技术支持热线:**

电话: +86-400-700-0606 (普通话)

服务时间: 9:00-12:00, 13:00-17:00 (不含法定节假日)

**网址:**

<http://www.cn.necel.com/> (中文)

<http://www.necel.com/> (英文)

**[北京]**

**日电电子(中国)有限公司**

中国北京市海淀区知春路 27 号

量子芯座 7, 8, 9, 15 层

电话: (+86) 10-8235-1155

传真: (+86) 10-8235-7679

**[深圳]**

**日电电子(中国)有限公司深圳分公司**

深圳市福田区益田路卓越时代广场大厦 39 楼

3901, 3902, 3909 室

电话: (+86) 755-8282-9800

传真: (+86) 755-8282-9899

**[上海]**

**日电电子(中国)有限公司上海分公司**

中国上海市浦东新区银城中路 200 号

中银大厦 2409-2412 和 2509-2510 室

电话: (+86) 21-5888-5400

传真: (+86) 21-5888-5230

**[香港]**

**香港日电电子有限公司**

香港九龙旺角太子道西 193 号新世纪广场

第 2 座 16 楼 1601-1613 室

电话: (+852) 2886-9318

传真: (+852) 2886-9022

2886-9044

**上海恩益禧电子国际贸易有限公司**

中国上海市浦东新区银城中路 200 号

中银大厦 2511-2512 室

电话: (+86) 21-5888-5400

传真: (+86) 21-5888-5230

**[成都]**

**日电电子(中国)有限公司成都分公司**

成都市二环路南三段 15 号天华大厦 7 楼 703 室

电话: (+86)28-8512-5224

传真: (+86)28-8512-5334