使用说明

78K0S/Kx1+

举例程序（低电压检测）

电压低于 2.7 V 时的复位

本文档介绍了举例程序的操作概述，并介绍了如何运用举例程序以及如何设置和应用低电压检测功能。本举例程序中，通过检测 V_{DD} 低于 V_{LVI}（V_{LVI} = 2.85 V ±0.15 V）来产生内部复位（LVI 复位）信号。对于 LVI 复位，复位存留 LED 当前显示数据，复位解除后恢复 LED 输出。

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**
1.1 初始设置的主要内容 3
1.2 主循环的后续内容 4
1.3 低电压检测(LVI)功能 5
第二章 电路图 **6**
2.1 电路图 6
2.2 外围硬件 6
第三章 软件 **7**
3.1 文件的组成 7
3.2 所使用的内部外设功能 8
3.3 初始设置和操作概要 8
3.4 流程图 10
第四章 设置方法 **11**
4.1 低电压检测(LVI)功能设置 11
第五章 使用该器件进行操作检验 **18**
5.1 连编举例程序 18
5.1.1 汇编语言版(源文件 + 工程文件) 18
5.1.2 C 语言版(仅源文件) 20
5.2 该器件的操作 25
第六章 相关文档 **27**
附录 A 程序清单 **28**
附录 B 版本修订历史 **38**

文档编号 U18821CA1V0AN00（第一版）
发布日期 2008 年 03 月 N

于日本印刷

- 本文档信息发布于2008年03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），交通用信号控制设备，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，提供了一个低电压检测（LVI）功能的应用实例。

设置 LVI 电路，以便通过检测 V_{DD} 低于 V_{LVI}（V_{LVI} = 2.85 V ±0.15 V）来产生内部复位（LVI 复位）信号。

初始设置完成后，经检测开关输入的下降沿并执行中断服务程序，根据开关输入次数来显示 LED 灯模式。（此过程同本举例程序（中断）中的描述一致。）

当复位由非 LVI 产生时，用程序初始化开关输入次数。当产生 LVI 复位时，立即保留复位产生之前的开关输入次数，并且 LVI 复位解除后据此显示相应的 LED 灯模式，这是因为 RAM 保留了复位之前的瞬间数据，除非它（复位前的电压）下降到 POC 电压之下。

1.1 初始设置的主要内容

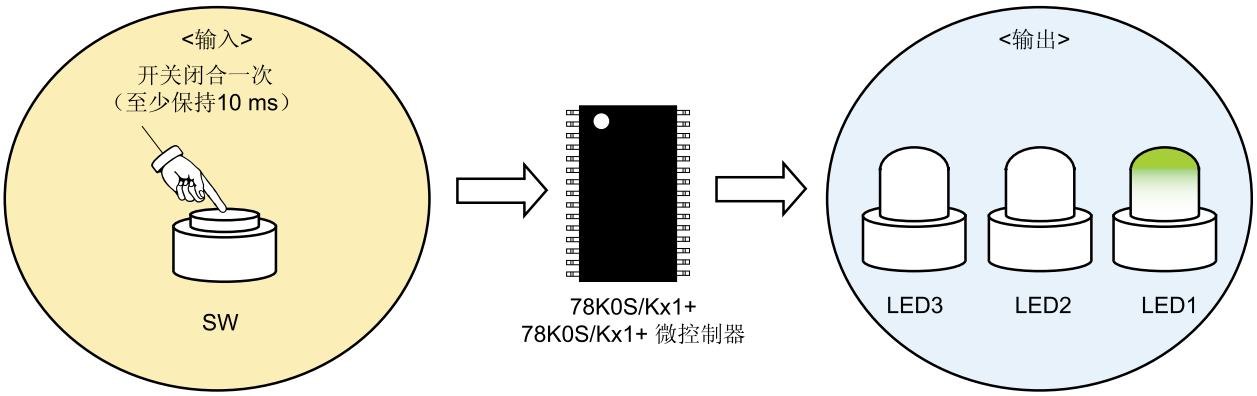
初始设置的内容如下：

选择内部高速振荡器作为系统时钟信号源[注]。
停止看门狗定时器的运行。
将 V_{LVI}（低电压检测电压）设置为 2.85 V ±0.15 V。
V_{DD}（电源电压）高于或等于 V_{LVI} 后，当检测到 V_{DD} 低于 V_{LVI} 时，产生内部复位（LVI 复位）信号。
将 CPU 的时钟频率设置为 4 MHz。
设置 I/O 端口。
将 INTP1（外部中断）的有效沿设置为下降沿。
中断允许。

注 使用选项字节来设置。

1.2 主循环的后续内容

通过检测由开关输入引起的 INTP1 引脚的下降沿来执行中断服务程序。在中断服务中，检测到 INTP1 引脚的下降沿之后且此下降沿能持续约 10ms，则可确认开关接通，LED 灯模式随之发生变化。如果大约 10 ms 之后，确认开关断开，那么本次处理被视为一次抖动，LED 亮灭模式不会改变。



开关输入次数[注]	LED输出		
	LED3	LED2	LED1
0	熄灭	熄灭	熄灭
1	熄灭	熄灭	点亮
2	熄灭	点亮	熄灭
3	熄灭	点亮	点亮
4	点亮	熄灭	熄灭
5	点亮	熄灭	点亮
6	点亮	点亮	熄灭
7	点亮	点亮	点亮

注 第八次开关输入后，灯的模式开始从第零次开关输入时的模式进行重复。

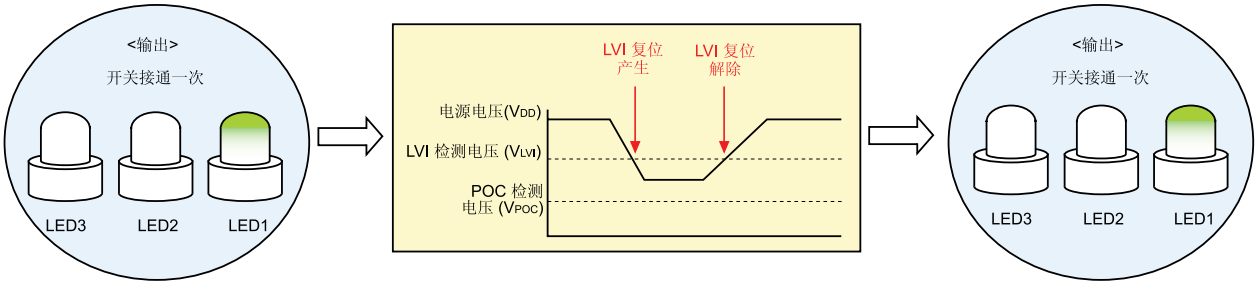
注意事项 关于使用该器件时的注意事项，参见各产品（78K0S/KU1+、78K0S/KY1+、78K0S/KA1+、78K0S/KB1+）的用户手册。

[专栏] 抖动
抖动是一种开关按下之后由于机械触点的弹跳而引起电信号瞬间反复接通和断开的现象。

1.3 低电压检测(LVI)功能

本举例程序中，当 V_{DD} 变得低于 V_{LVI} 时，会由低电压检测（LVI）功能产生内部复位（LVI 复位）信号。此时，寄存器的值被初始化，但 RAM 保留复位前的瞬间值，除非 V_{DD} 下降到 POC 电压之下。复位前所保留的开关输入数量将在复位后恢复，并且当产生 LVI 复位[注]时，据此来显示相应的 LED 灯模式。当复位由非 LVI 产生时，用程序初始化开关输入次数并且所有 LED 灯都熄灭。

注 正如下述[专栏]中所述的那样，当 C 语言程序中使用标准启动例程时，则在主函数之前会初始化 RAM 数据。为了避免这种情况，在本 C 语言版举例程序启动例程中的一部分被'注释'掉了，这样就不会将先前的 RAM 数据初始化（清为 0）。



[专栏] 启动例程的处理

标准启动例程主要完成下列处理：

- 堆栈指针的设置。
- 硬件初始化（需在早期阶段完成）。
- 程序库中所用变量的初始化。
- 将带有初始值的外部变量的初始值和 sreg 变量的初始值从 ROM 传输到 RAM 中。
- 将 0 赋值给 RAM 中不带初始值的的外部变量和 sreg 变量[注]。

注 此处理在包含于本 C 语言版举例程序内的启动例程源文件（cstart.asm）中被‘注释’掉了。

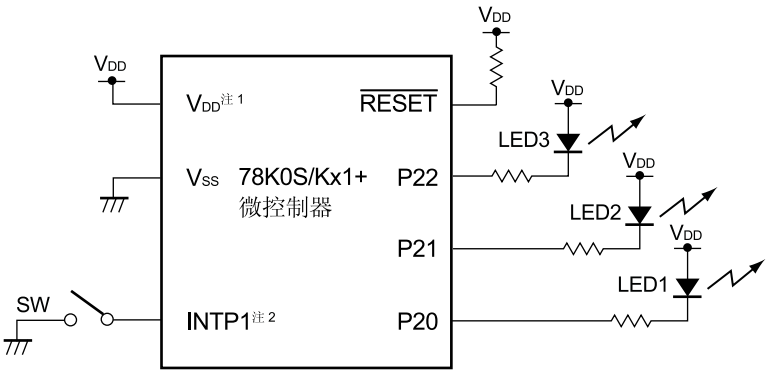
详情参见关于 **CC78K0S C 编译器操作用户手册**启动例程的相关章节。

第二章　电路图

本章介绍了该举例程序的电路图和所用的外围硬件。

2.1　电路图

电路图显示如下：



注　**1.** 适用电压范围为 3.0 V ≤ V_{DD} ≤ 5.5 V。

2. INTP1/P43：78K0S/KA1+微控制器和 78K0S/KB1+微控制器。
INTP1/P32：78K0S/KY1+微控制器和 78K0S/KU1+微控制器。

注意事项　1. 将 AV_{REF} 引脚直接连接至 V_{DD}（仅适用于 78K0S/KA1+微控制器和 78K0S/KB1+微控制器）。

2. 将 AV_{SS} 引脚直接连接至 GND（仅适用于 78K0S/KB1+微控制器）。

3. 除电路图中所示引脚和 AV_{REF} 引脚、AV_{SS} 引脚之外，保留所有未用引脚开路（未连接）。

2.2　外围硬件

所使用的外围硬件如下所示：

(1) 开关(SW)

开关用作控制 LED 灯的输入。

(2) LED 灯(LED1, LED2, LED3)

LED 灯用作开关输入的相应输出显示。

第三章　软件

本章介绍了所下载压缩文件的组成、所用微控制器的内部外设功能、初始设置和本举例程序的操作概要，并且显示了流程图。

3.1　文件的组成

下表显示了所下载压缩文件的组成：

(1)　汇编语言版本

文件名称	说明	包含的压缩 (*.zip)文件	
			
main.asm	用于硬件初始化处理的源文件和微控制器的主处理程序。	●	●
op.asm	用于设置选项字节的汇编语言源文件（设置系统时钟信号源）。	●	●
lvi.prw	用于集成开发环境PM+的工作空间文件。		●
lvi.prj	用于集成开发环境PM+的工程文件。		●

(2)　C 语言版本

文件名称	说明	包含的压缩 (*.zip)文件	
			
main.c	用于硬件初始化处理的源文件和微控制器的主处理程序。	●	●
op.asm	用于设置选项字节的汇编语言源文件（设置系统时钟信号源）。	●	●
cstart.asm	启动例程源文件（‘注释’掉ROM处理程序中的一部分）。	●	●
def.inc	程序库型设置文件（包含“cstart.asm”的文件）。	●	●
macro.inc	针对不同模板类型的宏定义文件（包含“cstart.asm”的文件）。	●	●
lvi.prw	用于集成开发环境PM+的工作空间文件。		●
lvi.prj	用于集成开发环境PM+的工程文件。		●

备注　  　：　仅包含源文件。

 　：　包含用于集成开发环境 PM+的文件。

3.2 所使用的内部外设功能

本举例程序中使用了微控制器的下列内部外设功能：

- V_{DD} < V_{LVI} 检测： 低电压检测器(LVI)
- 开关输入： INTP1 [注](外部中断)
- LED 输出： P20、P21、P22 (输出端口)

注 INTP1/P43：78K0S/KA1+微控制器和 78K0S/KB1+微控制器。
INTP1/P32：78K0S/KY1+微控制器和 78K0S/KU1+微控制器。

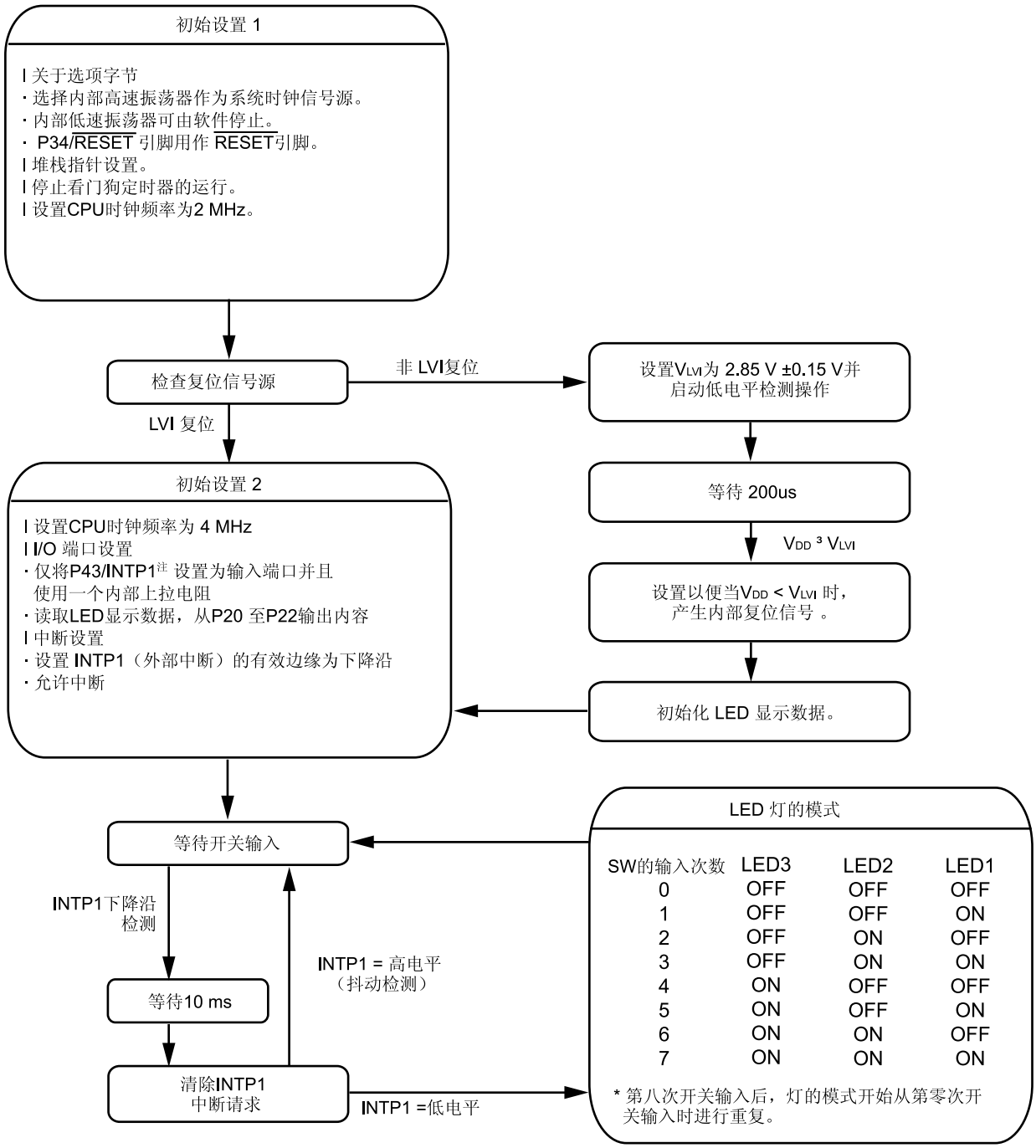
3.3 初始设置和操作概要

本举例程序中，初始设置时进行时钟频率的选择、I/O 端口的设置、中断的设置、LVI 的设置以及类似的操作。

初始设置完成后，通过检测开关输入(SW)的下降沿来执行中断服务程序并且根据开关输入次数对三个 LED 灯（LED1、LED2、和 LED3）进行控制。（该处理与举例程序（中断）中的描述一致。）

复位由非 LVI 产生时，用程序来初始化开关输入次数。当产生 LVI 复位时，立即恢复复位产生之前的开关输入次数，并且复位解除后据此显示相应的 LED 灯模式，这是因为 RAM 保留了复位前的瞬间值，除非它（复位前的电压）下降到 POC 电压之下。

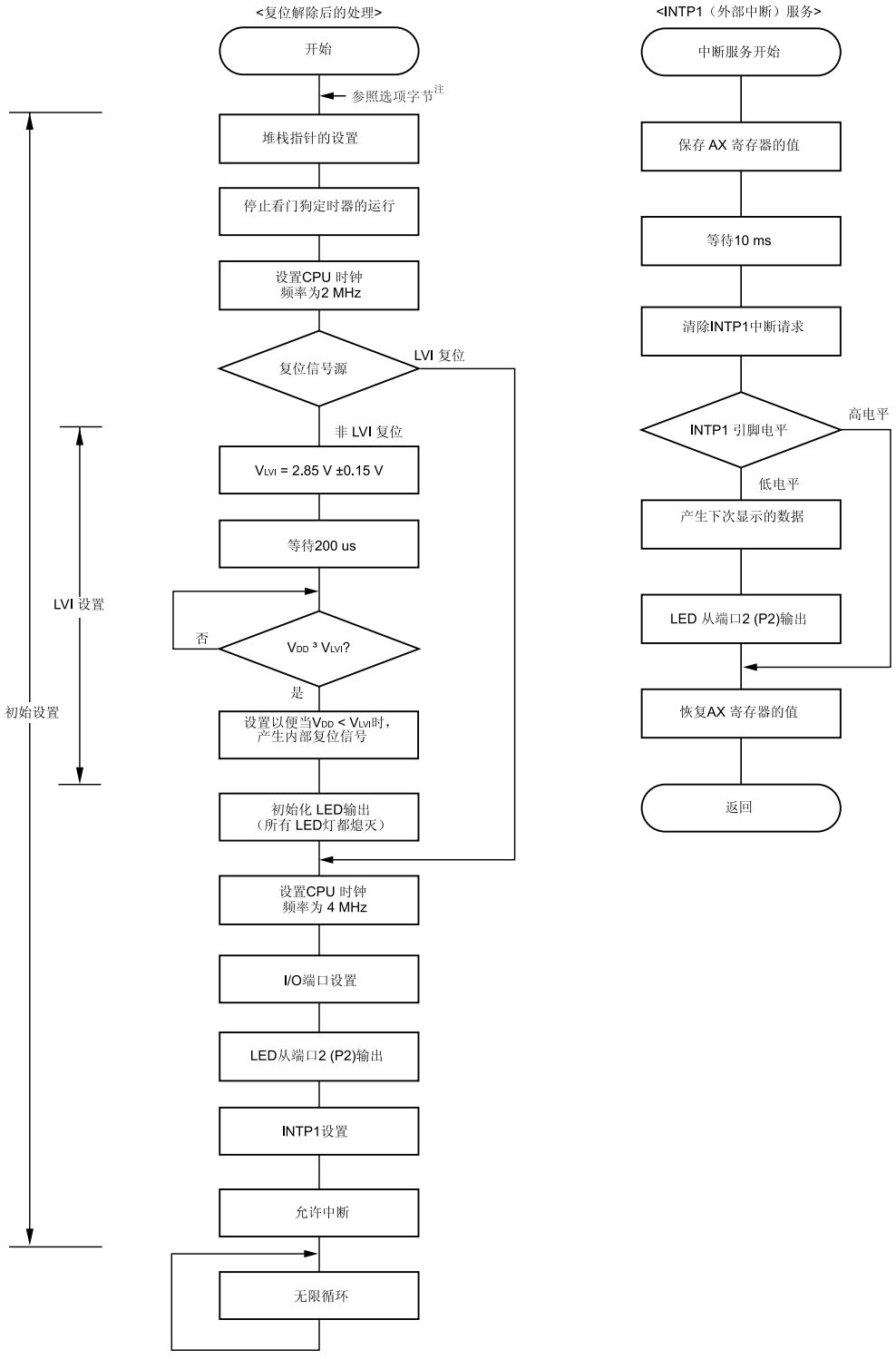
如下所示的状态转换图对此进行了详细描述：



注 INTP1/P43：78K0S/KA1+微控制器和 78K0S/KB1+微控制器。
INTP1/P32：78K0S/KY1+微控制器和 78K0S/KU1+微控制器。

3.4 流程图

举例程序的流程图如下所示：



注 复位解除后，微控制器自动援引选项字节。本举例程序中，援引选项字节设置下列内容：

- 将内部高速振荡时钟（8 MHz（TYP.））用作系统时钟信号源。
- 内部低速振荡器可利用软件停止。
- 将 P34/$\overline{\text{RESET}}$ 引脚用作 $\overline{\text{RESET}}$ 引脚。

第四章　设置方法

本章介绍低电压检测功能。

关于其它的初始设置，参见 **78K0S/Kx1+举例程序（初始设置）LED 灯开关控制的使用说明**。关于中断，参见 **78K0S/Kx1+举例程序（中断）由开关输入产生的外部中断使用说明**。

关于如何设置寄存器，参见各产品（**78K0S/KU1+**、**78K0S/KY1+**、**78K0S/KA1+**、**78K0S/KB1+**)的用户手册。

关于汇编语言指令，参见 **78K/0S 系列指令用户手册**。

4.1　低电压检测(LVI)功能设置

低电压检测功能具有下列两种操作模式：

- 用作复位（参见**[实例 1]**）。
- 用作中断（参见**[实例 2]**）。

低电压检测功能主要由下列两种寄存器进行控制：

- 低电压检测寄存器（LVIM）。
- 低电压检测电平选择寄存器（LVIS）。

(1) 关于低电压检测操作的设置

低电压检测寄存器（LVIM）用于设置低电压检测操作模式并控制其操作。

图 4-1. 低电压检测寄存器(LVIM)的格式

LVIM

LVION	0	0	0	0	0	LVIMD	LVIF

低电压检测标志(只读)

0	当$V_{DD} \geq V_{LVI}$时或禁止操作。
1	$V_{DD} < V_{LVI}$

低电压检测操作模式的选择

0	当$V_{DD} < V_{LVI}$时，产生中断信号。
1	当$V_{DD} < V_{LVI}$时，产生内部复位信号。

低电压检测操作的允许或禁止

0	禁止操作。
1	允许操作。

(2) 关于低电压检测电平的设置

低电压检测电平选择寄存器（LVIS）用来设置低电压检测的电平。

图 4-2. 低电压检测电平选择寄存器(LVIS)的格式

LVIS

0	0	0	0	LVIS3	LVIS2	LVIS1	LVIS0

低电压检测电平

0	0	0	0	V_{LVI0} (4.3 V ±0.2 V)
0	0	0	1	V_{LVI1} (4.1 V ±0.2 V)
0	0	1	0	V_{LVI2} (3.9 V ±0.2 V)
0	0	1	1	V_{LVI3} (3.7 V ±0.2 V)
0	1	0	0	V_{LVI4} (3.5 V ±0.2 V)
0	1	0	1	V_{LVI5} (3.3 V ±0.15 V)
0	1	1	0	V_{LVI6} (3.1 V ±0.15 V)
0	1	1	1	V_{LVI7} (2.85 V ±0.15 V)
1	0	0	0	V_{LVI8} (2.6 V ±0.1 V)
1	0	0	1	V_{LVI9} (2.35 V ±0.1 V)
除上述之外				设置禁止

[实例 1]　通过设置低电压检测电平（V_{LVI}）为 2.85 V ± 0.15 V，将低电压检测功能用作复位（与本举例程序的内容一致）。

- 设置步骤：

<1>　屏蔽 LVI 中断（LVIMK = 1）[注]。
<2>　由 LVIS 寄存器的位 3 至位 0（LVIS3 至 LVIS0）来设置检测电平。
<3>　将 LVIM 寄存器的位 7（LVION）设置为 1 以允许 LVI 操作。
<4>　利用软件等待至少 200 μs。
<5>　利用 LVIM 寄存器的位 0（LVIF）等待直到确认“$V_{DD} \geq V_{LVI}$（LVIF = 0）”。
<6>　将 LVIM 寄存器的位 1（LVIMD）设置为 1，以便当检测 LVI 时，产生内部复位信号。

注　省略本举例程序和下页程序实例中的此项设置，这是因为复位后已屏蔽了 LVI 中断。

备注　以上所述的步骤<2>至步骤<6>与下页中的步骤<2>至<6>相符。

MK0

1	1	1	1	1	1	1	1

中断服务控制

1	禁止中断服务。

LVIS

0	0	0	0	0	1	1	1

低电压检测电平

0	1	1	1	V_{LVI7} (2.85 V ±0.15 V)

LVIM

1	0	0	0	0	0	1	1

低电压检测标志（只读）

1	$V_{DD} < V_{LVI}$

低电压检测操作模式的选择

1	当$V_{DD} < V_{LVI}$时，产生内部复位信号。

低电压检测操作的允许或禁止

1	允许操作。

• 汇编语言程序实例（与本举例程序的内容一致）

```
XMAIN   CSEG    UNIT
RESET_START：
<2>     MOV     LVIS，  #00000111B      ；设置低电压检测电平(VLVI)为 2.85 V ± 0.15 V。
<3>     SET1    LVION                   ；允许低电压检测器操作。

        MOV     A，     #40             ；指定 200 us 等待的计数值。
WAIT_200US：
<4>     DEC     A
        BNZ     $WAIT_200US             ；  0.5[us/clk] × 10[clk] × 40[计数值] = 200[us]。
WAIT_LVI：
        NOP
<5>     BT      LVIF，  $WAIT_LVI       ；如果 VDD < VLVI，则进行转移。

<6>     SET1    LVIMD                   ；设置以便当 VDD < VLVI 时，产生内部复位信号。
```

• C 语言程序实例（同本举例程序的内容一致）

```
void hdwinit(void){
        unsigned char ucCnt200us；      /*用于 200 us 等待的 8 位变量 */
<2>             LVIS  = 0b00000111；    /* 设置低电压检测电平(VLVI)为 2.85 V ± 0.15 V */
                LVION = 1；             /* 允许低电压检测器操作 */
<3>
                for (ucCnt200us = 0；ucCnt200us < 9; ucCnt200us++){ /*约 200 us 的等待 */
<4>                     NOP()；
                }

                while (LVIF){           /* 等待 VDD >= VLVI */
<5>                     NOP()；
                }

                LVIMD = 1；             /* 设置以便当 VDD < VLVI 时，产生内部复位信号 */
<6>     }
```

备注　**1.** 同本举例程序中一样，上述所涉及的等待时间（200 μs）由 2 MHz 的 f_{CPU} （CPU 时钟频率）计算得到。

2. 以上所述的步骤<2>至步骤<6>与上页的步骤<2>至步骤<6>相对应。

[实例 2]　　通过设置低电压检测电平（V_{LVI}）为 4.3 V ±0.2 V，将低电压检测功能用作中断。

- 设置步骤：

<1>　屏蔽 LVI 中断（LVIMK = 1）[注]。
<2>　由 LVIS 寄存器的位 3 至位 0（LVIS3 至 LVIS0）设置检测电平。
<3>　将 LVIM 寄存器的位 7（LVION）设置为 1 以允许 LVI 操作。.
<4>　利用软件等待至少 200μs。
<5>　利用 LVIM 寄存器的位 0（LVIF）等待直到确认“$V_{DD} \geq V_{LVI}$ (LVIF = 0)”。
<6>　清除 LVI（LVIIF = 0）的中断请求标志。
<7>　释放 LVI（LVIMK = 0）的中断屏蔽标志。
<8>　执行 EI 指令（当使用向量中断时）。

注　　省略以上所示程序实例中的此项设置，这是因为复位后已屏蔽了 LVI 中断。

备注　　以上所述的步骤<2>至步骤<8>与下页及再下页的步骤<2>至步骤<8>相对应。

MK0

1	1	1	1	1	1	0/1	1

中断服务控制

0	允许中断服务。
1	禁止中断服务。

IF0

0	0	0	0	0	0	0	0

中断请求标志

0	无中断请求信号产生。

PSW

1	0	0	0	0	0	1	0

中断响应的允许或禁止

1	允许

LVIS

0	0	0	0	0	0	0	0

低电压检测电平

0	0	0	0	V_{LVI0} (4.3 V ±0.2 V)

LVIM

1	0	0	0	0	0	0	1

低电压检测标志（只读）

1	$V_{DD} < V_{LVI}$

低电压检测操作模式的选择

0	当$V_{DD} < V_{LVI}$时，产生中断信号。

* 因为缺省值为 0，所以无需设置。

低电压检测操作的允许或禁止

1	允许操作。

• 汇编语言程序实例（使用低电压检测中断将 CPU 时钟频率设置为低速）。

```
HIGH_SPEED    EQU     00H
LOW_SPEED     EQU     02H

XVCT    CSEG    AT      0000H
        DW      RESET_START   ; (00)   RESET
        DW      RESET_START   ; (02)   --
        DW      RESET_START   ; (04)   --
        DW      INTERRUPT_LVI ; (06)   INTLVI

XMAIN   CSEG    UNIT
RESET_START:
        MOV     PCC,    #00000000B      ;  提供至CPU的时钟 (fcpu) = fxp (= fx/4 = 2 MHz)。
<2>     MOV     LVIS,   #00000000B      ;  设置低电压检测电平(VLVI)为 4.3 V ± 0.2 V。
<3>     SET1    LVION                   ;  允许低电压检测器操作。

        MOV     A,      #40             ;  指定 200 us 等待的计数值。
WAIT_200US:
<4>     DEC     A
        BNZ     $WAIT_200US             ;  0.5[us/clk] × 10[clk] × 40[计数值] = 200[us]

WAIT_LVI1:
        NOP
<5>     BT      LVIF,   $WAIT_LVI1      ;  如果 VDD < VLVI，则进行转移。
        MOV     PPCC,   #HIGH_SPEED     ;  提供至外围硬件的时钟 (fxp) = fx (= 8 MHz)。
                                        ;  -> 提供至 CPU 的时钟 (fcpu) = fxp = 8 MHz。
<6>     CLR1    LVIIF                   ;  清除 LVI 中断请求标志。
<7>     CLR1    LVIMK                   ;  释放 LVI 中断请求屏蔽标志。
<8>     EI                              ;  允许向量中断。

MAIN_LOOP:
        MOV     A,      PPCC
        CMP     A,      #HIGH_SPEED
        BZ      $MAIN_LOOP              ;  如果 CPU 时钟(fcpu)为 8 MHz，则转到 MAIN_LOOP。

WAIT_LVI2:
        NOP
        BT      LVIF,   $WAIT_LVI2      ;  如果 VDD < VLVI，则进行转移。
        MOV     PPCC,   #HIGH_SPEED     ;  提供至外围硬件的时钟(fxp) = fx (= 8 MHz)。
                                        ;  -> 提供至 CPU 的时钟 (fcpu) = fxp = 8 MHz。
        BR      $MAIN_LOOP              ;  转到 MAIN_LOOP。

INTERRUPT_LVI:
        MOV     PPCC,   #LOW_SPEED      ;  提供至外围硬件的时钟 (fxp) = fx/4 (= 2 MHz)。
                                        ;  -> 提供至 CPU 的时钟 (fcpu) = fxp = 2 MHz。
        RETI                            ;  从中断服务程序中返回。
```

按照使用 DW 伪指令所声明初始值（“INTERRUPT_LVI”），当中断由 LVI 检测产生时，中断服务从“INTERRUPT_LVI”标号处开始执行。

设置 CPU 的时钟频率。

加电引导后，当 $V_{DD} \geq V_{LVI}$ 时，设置 CPU 的时钟频率为高速。

LVI 检测后，当 $V_{DD} \geq V_{LVI}$ 时，设置 CPU 的时钟频率为高速。

根据由 LVI 检测产生的中断，中断服务从“INTERRUPT_LVI”处开始执行。

当 $V_{DD} < V_{LVI}$ 时，设置 CPU 的时钟频率为低速。

备注　**1.** 以上所涉及的等待时间（200 μs）和 CPU 时钟频率由 8 MHz 的 fx（系统时钟振荡频率）计算得到。

2. 以上所述的步骤<2>至步骤<8>与上页的步骤<2>至步骤<8>相对应。

• C 语言举例程序（通过使用低电平检测中断将 CPU 时钟频率设置为低速）。

```
#pragma SFR                                  /* SFR 名能够在 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 NOP                                  /* NOP 指令能够在 C 语言层面上进行描述 */
#pragma interrupt INTLVI fn_intlvi   /* 中断函数声明：INTLVI */

#define  HighSpeed 0x00
#define  LowSpeed 0x02

void hdwinit(void){
        PCC   = 0b00000000;          /* 提供至CPU的时钟 (fcpu) = fxp (= fx/4 = 2 MHz) */

<2>     LVIS  = 0b00000000;          /* 设置低电压检测电平(VLVI)为 4.3 V ± 0.2 V */
<3>     LVION = 1;                   /* 允许低电压检测器操作 */

<4>     for (ucCnt200us = 0; ucCnt200us < 9; ucCnt200us++){          /* 等待约 200 us */
                NOP();
        }

<5>     while (LVIF){                /* 等待 VDD >= VLVI */
                NOP();
        }

        PPCC  = HighSpeed;           /* 提供至外围硬件的时钟 (fxp) = fx (= 8 MHz)
                                        -> 提供至 CPU 的时钟 (fcpu) = fxp = 8 MHz */

<6>     LVIIF = 0;                   /* 清除 LVI 中断请求标志 */
<7>     LVIMK = 0;                   /* 释放 LVI 中断请求屏蔽标志 */
<8>     EI();                        /* 允许向量表中断 */

        return;
}

void main(void){
        while (1){
                while (PPCC == HighSpeed){          /* 等待 LVI 中断 */
                        NOP();
                }

                while (LVIF){                       /* 等待 VDD >= VLVI */
                        NOP();
                }

                PPCC  = HighSpeed;           /* 提供至外围硬件的时钟 (fxp) = fx (= 8 MHz)
                                                -> 提供至 CPU 的时钟 (fcpu) = fxp = 8 MHz */
        }
}

__interrupt void fn_intlvi(){
        PPCC  = LowSpeed;            /* 提供至外围硬件的时钟 (fxp) = fx/4 (= 2 MHz)
                                        -> 提供至 CPU 的时钟 (fcpu) = fxp = 2 MHz */

        return;
}
```

按照在预处理指示符 (#pragma 指示符)中声明的 INTLVI 中断函数(本实例中的“fn_intlvi”)及使用_interrupt 修饰符声明的中断函数，当产生中断时，中断服务程序从使用_interrupt 修饰符所声明的中断函数处开始执行。

设置 CPU 时钟频率。

加电引导后，当 $V_{DD} \geq V_{LVI}$ 时，设置 CPU 时钟频率为高速。

LVI 检测后，当 $V_{DD} \geq V_{LVI}$ 时，设置 CPU 时钟频率为高速。

根据由 LVI 检测所产生的中断，中断服务程序从“fn_intlvi”处开始执行。

当 $V_{DD} < V_{LVI}$ 时，设置 CPU 时钟频率为低速。

备注 1. 以上所涉及的等待时间（200 μs）和 CPU 时钟频率由 8 MHz 的 fx（系统时钟振荡频率）计算得到。

2. 以上所述的步骤<2>至步骤<8>与 15 页的步骤<2>至步骤<8>相对应。

第五章　使用该器件进行操作检验

利用下载的举例程序，本章介绍了从连编到使用该器件进行操作检验的流程。

5.1　连编举例程序

利用由单击 图标所下载的汇编语言版举例程序（源文件+ 工程文件）和单击 图标所下载的 C 语言版举例程序（仅源文件），本节介绍了如何连编举例程序。关于如何编译其它下载的程序，参见 **78K0S/Kx1+举例程序启动指南使用说明中的第三章 注册集成开发环境 PM+ 的工程并执行编译**

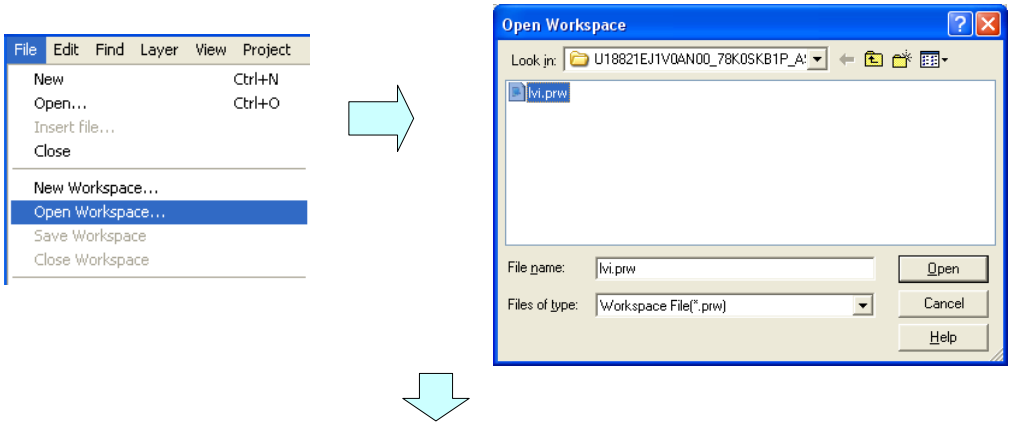
如何操作 PM+之详情，参见 **PM+工程管理器用户手册**。

5.1.1　汇编语言版（源文件 + 工程文件）

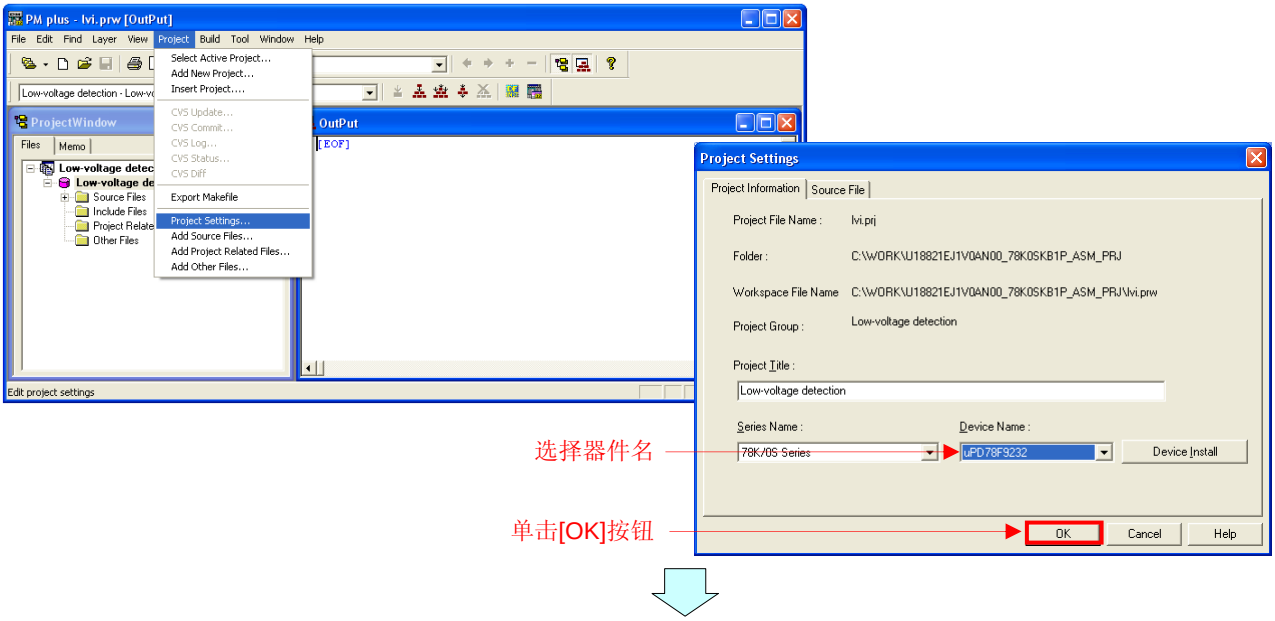
利用通过单击 图标所下载的 78K0S/KB1+微控制器举例程序（低电压检测）中的汇编语言版文件，本节介绍了如何连编举例程序。

(1) 启动 PM+。

(2) 从[File]菜单中单击[Open Workspace]选项，选择“lvi.prw”并单击[Open]按钮。将创建一个可自动读取源文件的工作空间。

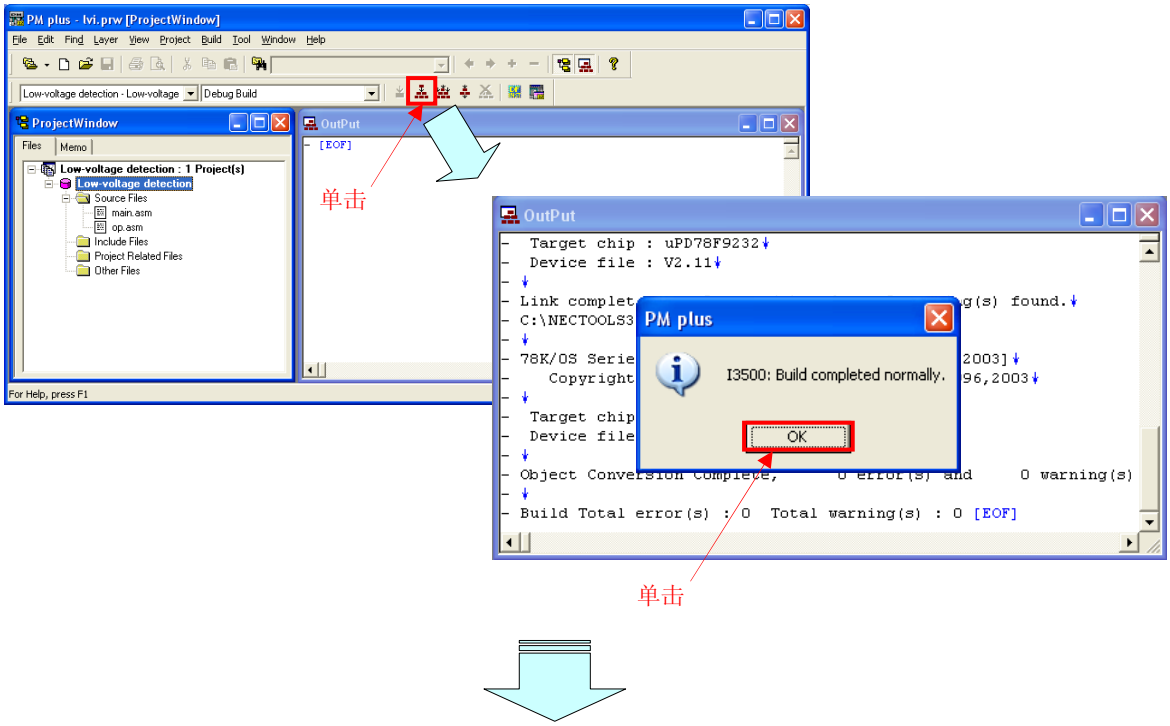


(3) 从[Project]菜单中选择[Project Settings]选项。当[Project Settings]对话框打开时，选择所用的器件名（按照缺省值，将会选择为具备最大容量 ROM 或 RAM 的器件），然后单击[OK]。



(4) 单击  （[Build]按钮）。当源文件正常编译后，将显示“I3500:Build completed normally.”信息。

(5) 单击信息框中的[OK]按钮。将会创建一个用于 flash 存储器写入的 HEX 文件。



将产生一个用于 flash 存储器写入的 HEX 文件。→ 转到 **5.2**。

5.1.2　C语言版（仅源文件）

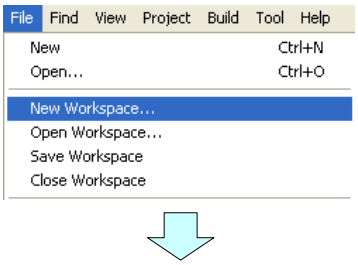
使用通过单击 图标所下载的 78K0S/KB1+微控制器举例程序（低电压检测）中的 C 语言版文件，本节介绍了如何编译链接示例程序。

本示例程序的 C 语言版本除了包含“main.c”和“op.asm”之外，还包含启动例程源文件“cstart.asm”、程序库型设置文件“def.inc”、及宏定义型文件“macro.inc”。这些文件相应的部分已从标准启动例程中‘注释’掉了。为了利用这些文件创建 PM+工程，需要执行的操作不同于其它 C 语言版举例程序的那些操作。步骤描述如下：

- 工程注册

(1)　启动 PM+。

(2)　从[File]菜单中选择[New Workspace]选项。



(3)　显示[New WorkSpace - Step 1/8 [Workspace Information]]对话框。设置下列选项：

(a)　Workspace File Name（本实例中，键入“lvi”作为文件名）。

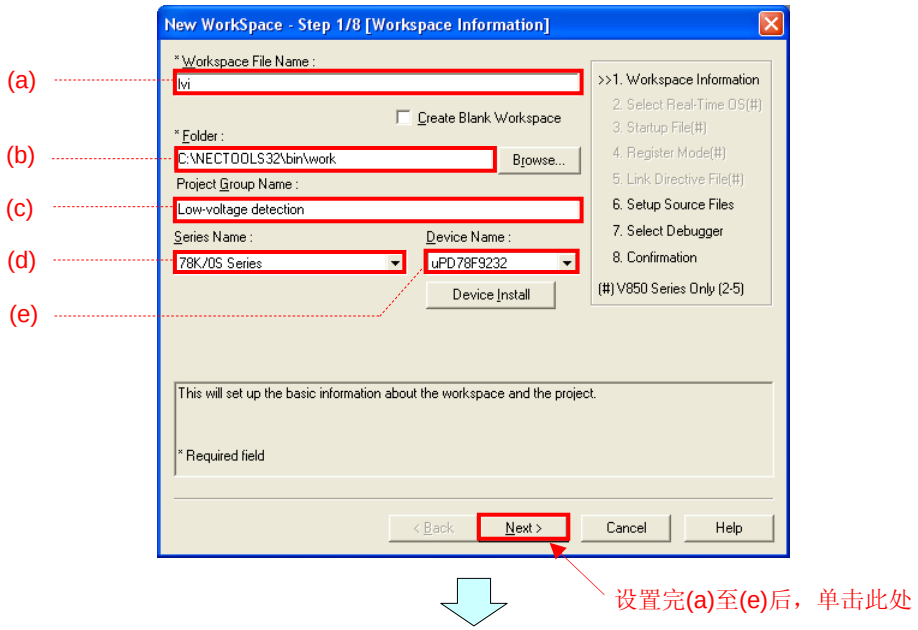
(b)　Folder（本实例中，指定了一个定位于缺省文件夹（“bin”文件夹，其中已存在 PM+）下随意创建的“work”文件夹。）

(c)　Project Group Name（本实例中，键入“Low-voltage detection”作为工程组名。）

(d)　Series Name（本实例中，选择“78K/0S Series”作为系列名。）

(e)　Device Name（本实例中，设值为 78K0S/KB1+微控制器的“uPD78F9232”产品。）

设置完选项(a)至选项(e)后，单击[Next]按钮。



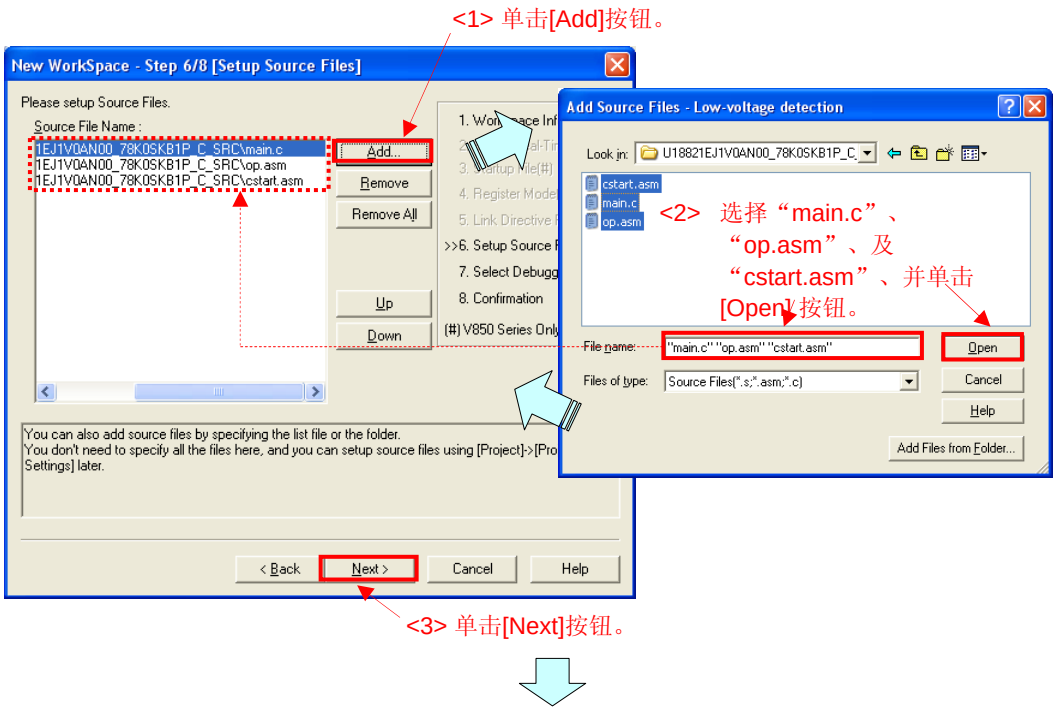
(4)　显示[New WorkSpace - Step 6/8 [Setup Source Files]]对话框。按照下列步骤设置源文件。

<1>　单击[Add]按钮。

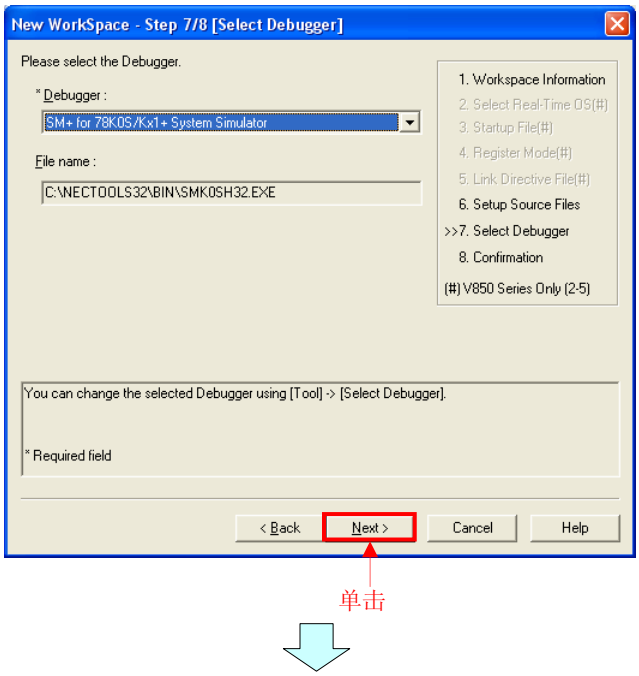
<2>　显示[Add Source Files]对话框。选择下列源文件并单击[Open]按钮。

- main.c
- op.asm
- cstart.asm

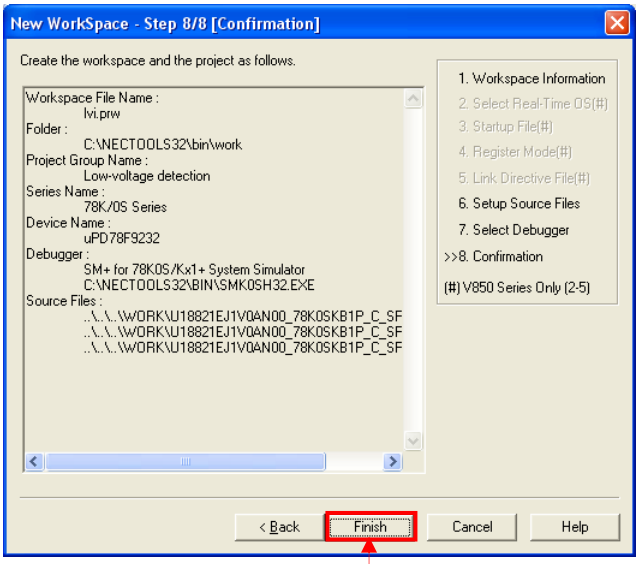
<3>　设置步骤<2>中所选择的源文件。单击[Next]按钮。



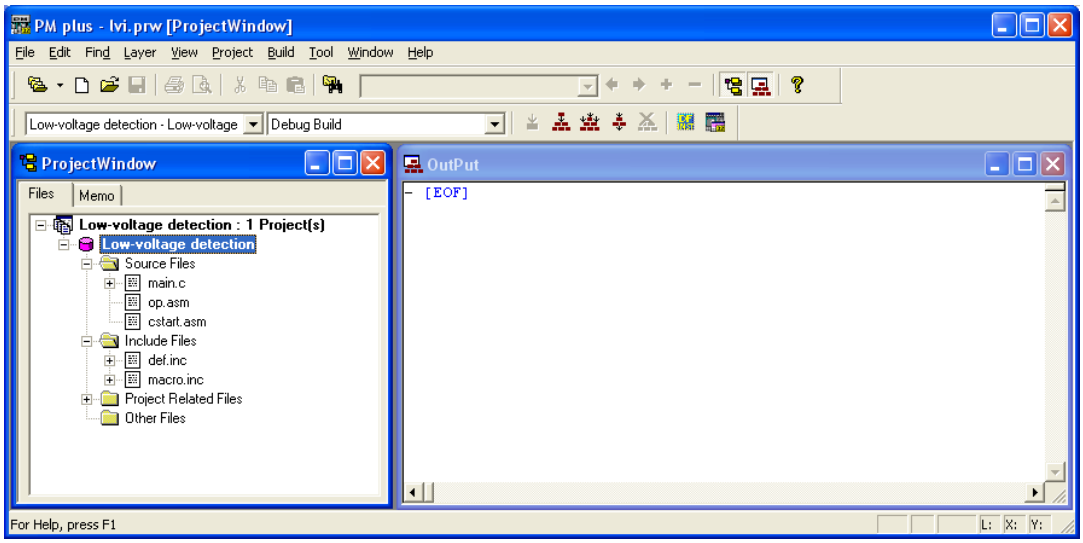
(5)　显示[New WorkSpace - Step 7/8 [Select Debugger]]对话框。单击[Next]按钮。



(6)　显示[New WorkSpace - Step 8/8 [Confirmation]]对话框。确认这些设置并单击[Finish]按钮。



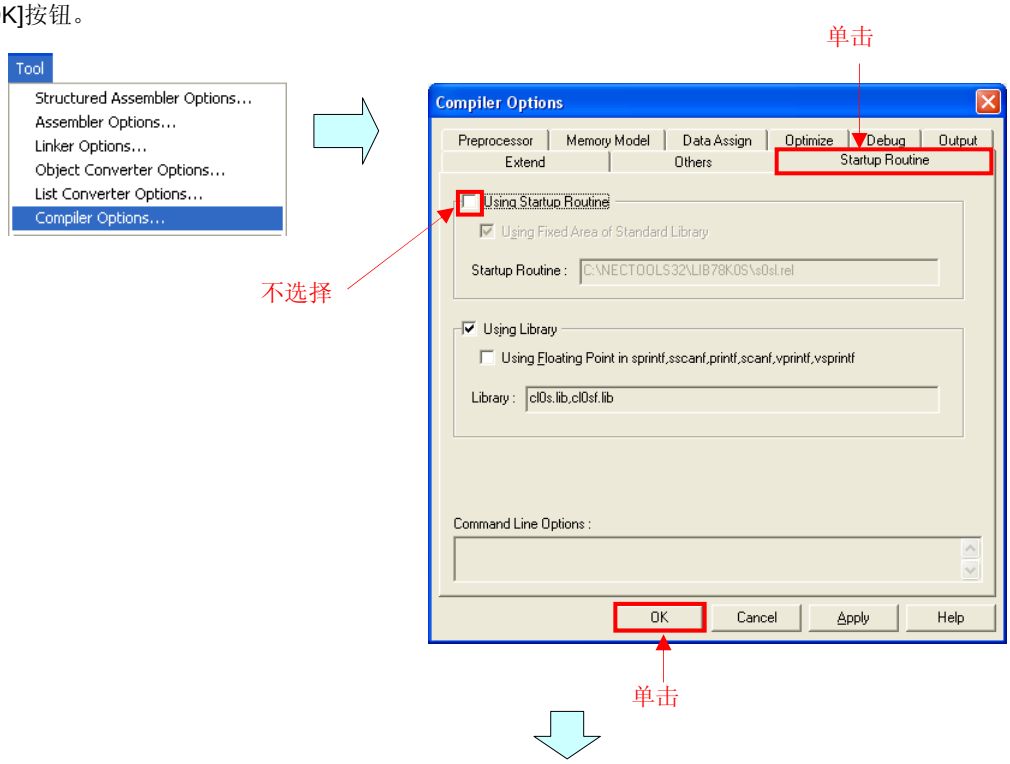
(7)　一个工作区创建完成，并注册了此项工程。



- 编译器选项设置

(8)　从[Tool]菜单中选择[Compiler Options]选项。

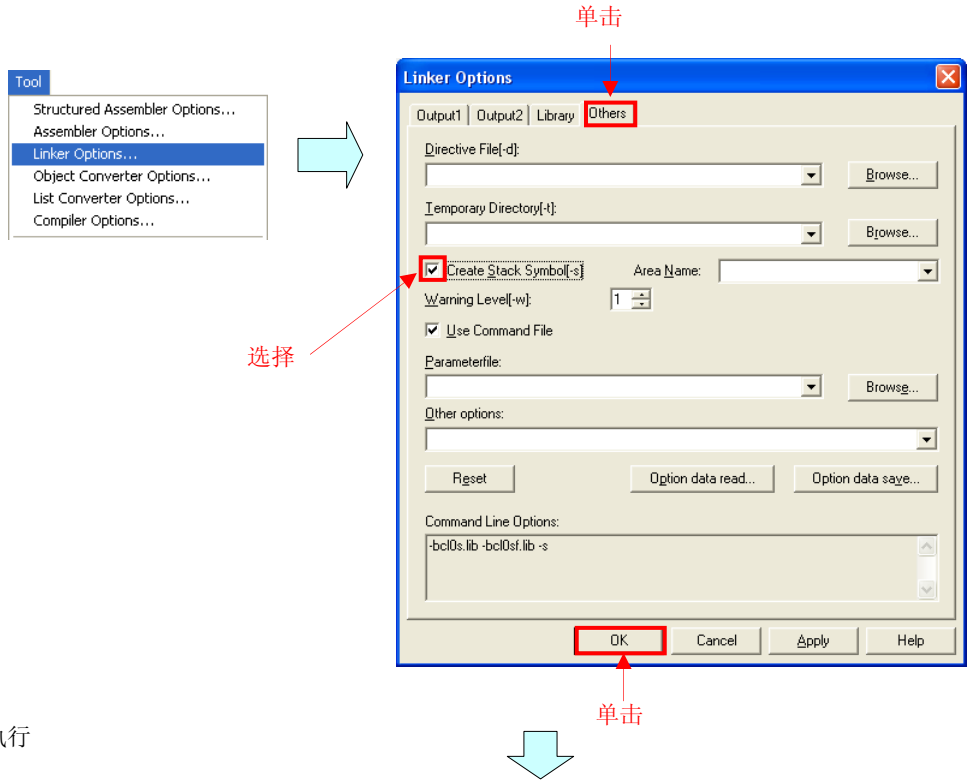
(9)　显示[Compiler Options]对话框。单击[Startup Routine]选项卡，不选择[Using Startup Routine]复选框并单击[OK]按钮。



- 链接器选项设置

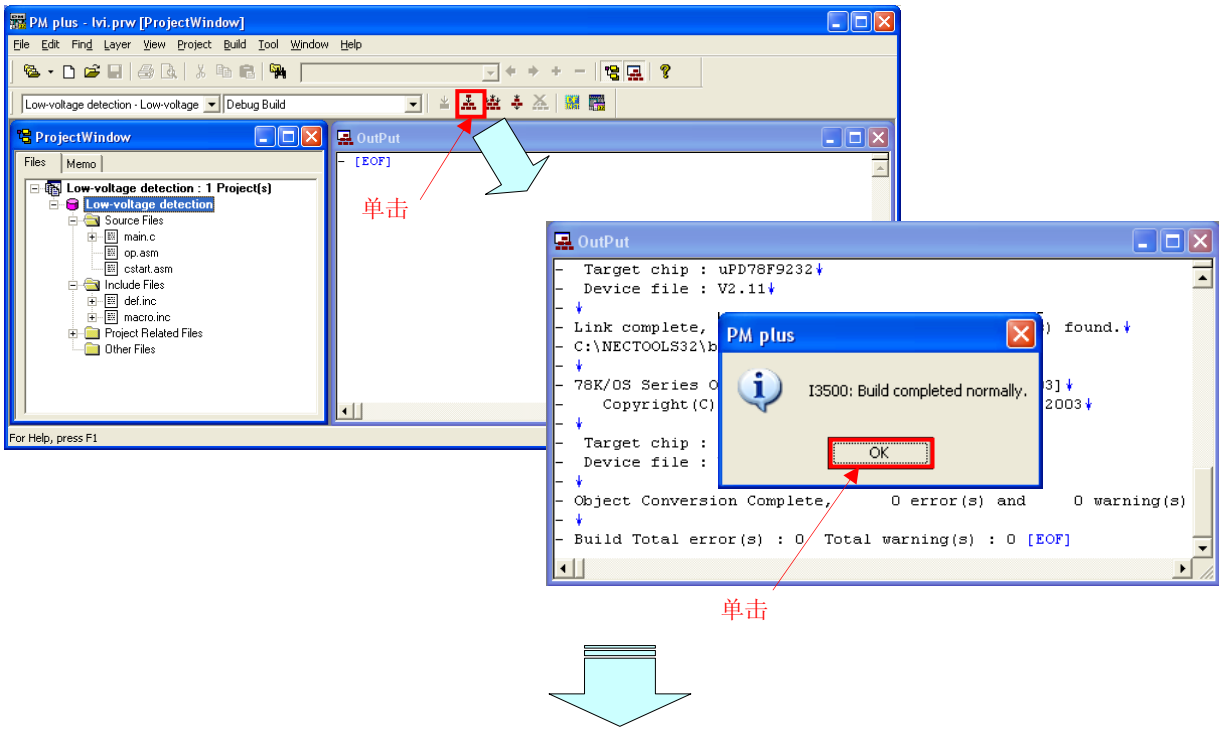
(10) 从[Tool]菜单中选择[Linker Options]选项。

(11) 显示[Linker Options]对话框。单击[Others]选项卡，选择[Create Stack Symbol[-s]]复选框，并单击[OK]按钮。



- 编译的执行

(12) 单击  （[Build]按钮）。当源文件正常编译后，将显示“I3500:Build completed normally.”信息。
(13) 单击信息框中的[OK]按钮。将会创建一个用于 flash 存储器写入的 HEX 文件。

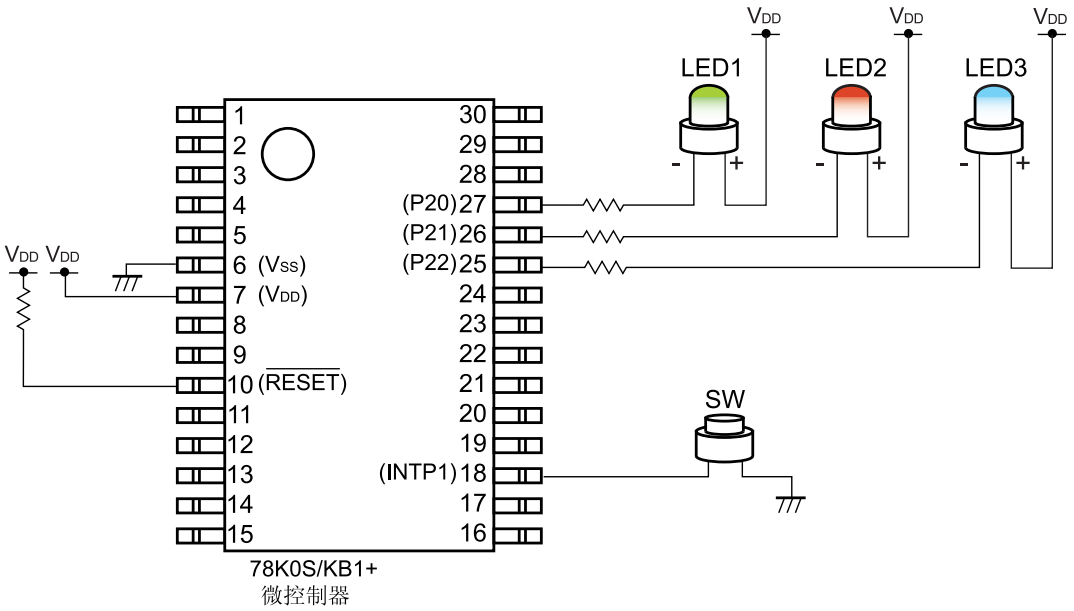


将产生一个用于 flash 存储器写入的 HEX 文件。→ 转到 **5.2**。

5.2　该器件的操作

本节介绍了使用该器件进行检查操作的一个实例。
由执行编译所产生的 HEX 文件能够写入到该器件的 flash 存储器中。
关于如何写入到该器件的 flash 存储器中，参见 **78K0S/Kx1+简化的 Flash 写入手册信息**（已配备）。

如何连接该器件和所用外围硬件（开关和 LED 灯）的实例显示如下：



已写有本举例程序的该器件操作实例介绍如下：

(1)　当 $V_{DD} \geq 3.0$ V (正常操作中)[注]

LED 灯的模式随开关输入次数的变化而改变。

注　因为低电平检测电压（V_{LVI}）设定为 2.85 V ± 0.15 V，故当 $V_{DD} \geq 3.0$ V 时，执行开关输入中断服务的操作。

开关输入次数[注]	LED 输出		
	LED3	LED2	LED1
0	熄灭	熄灭	熄灭
1	熄灭	熄灭	点亮
2	熄灭	点亮	熄灭
3	熄灭	点亮	点亮
4	点亮	熄灭	熄灭
5	点亮	熄灭	点亮
6	点亮	点亮	熄灭
7	点亮	点亮	点亮

注　第八次开关输入后，灯的模式开始从第零次开关输入时进行重复。

如果按下开关的时间少于 10 ms，则将本次开关输入识别为抖动且 LED 显示模式不会改变（与本次开关按下之前的模式相同）。

(2) $V_{DD} \geq 3.0$ V → $V_{DD} = 2.5$ V (低电压检测) → $V_{DD} \geq 3.0$ V

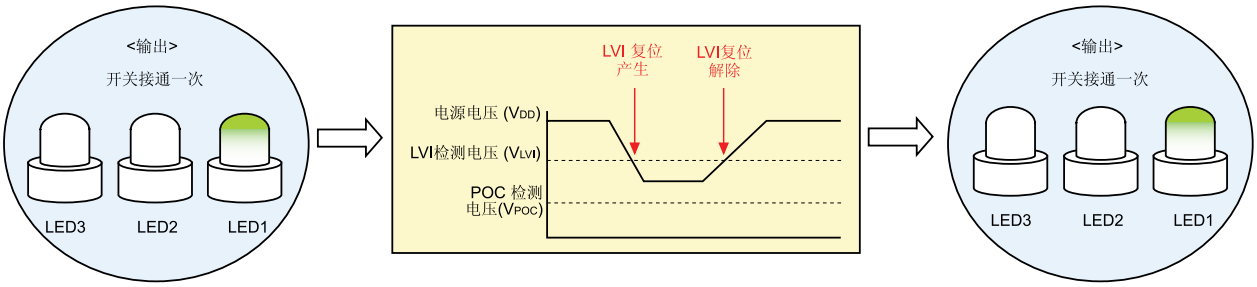
根据电源电压的改变，LED 灯变化如下：

<1> $V_{DD} \geq 3.0$ V → $V_{DD} = 2.5$ V

将产生 LVI 复位，这是因为低电压检测电平（V_{LVI}）设为 2.85 V ±0.15 V 并且低电压检测功能设置为用于复位。此时，所有 LED 灯都熄灭，但是 RAM 保留了复位前 LED 显示数据的瞬间值。

<2> $V_{DD} = 2.5$ V → $V_{DD} \geq 3.0$ V

操作返回到正常模式。这时，立即恢复复位前的亮灭模式，这是因为确认复位信号源是一个 LVI 复位，并且读取了保留在 RAM 中的 LED 显示数据。



(3) $V_{DD} \geq 3.0$ V → 2.0 V > V_{DD} (低于数据保留的电源电压) → $V_{DD} \geq 3.0$ V

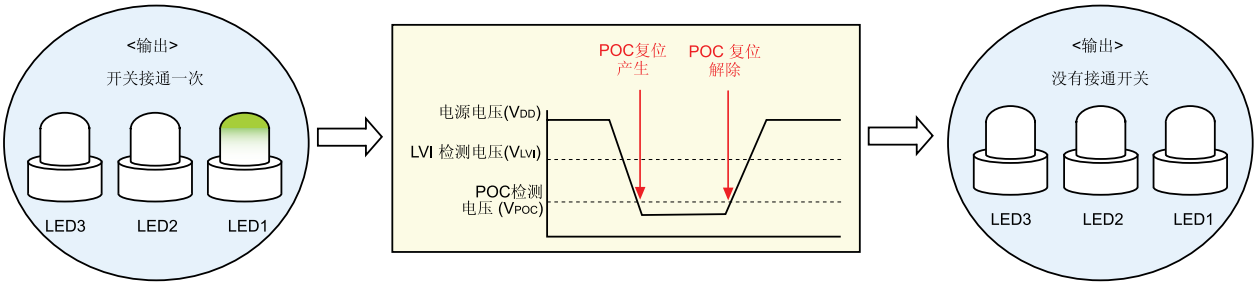
根据电源电压的改变，LED 灯变化如下：

<1> $V_{DD} \geq 3.0$ V → 2.0 V > V_{DD}

将产生由加电清除（POC）引起的复位。这时，所有 LED 灯都熄灭且 RAM 中的数据不明确。

<2> 2.0 V > V_{DD} → $V_{DD} \geq 3.0$ V

操作返回到正常模式。此时，所有 LED 灯都不亮（开关输入次数 = 0），这是因为确认复位信号源非 LVI 复位且已经初始化 RAM 中的 LED 灯模式。



第六章　相关文档

文档名称		日语/英语
78K0S/KU1+用户手册		PDF
78K0S/KY1+用户手册		PDF
78K0S/KA1+用户手册		PDF
78K0S/KB1+用户手册		PDF
78K/0S系列指令用户手册		PDF
RA78K0S汇编语言程序包用户手册	语言	PDF
	操作	PDF
CC78K0S C编译器用户手册	语言	PDF
	操作	PDF
PM+工程管理器用户手册		PDF
SM+ 系统仿真器操作用户手册		PDF
78K0S/KA1+简化的Flash 写入手册 MINICUBE2 信息		PDF
78K0S/Kx1+举例程序(初始设置) LED灯开关控制的使用说明		PDF
78K0S/Kx1+举例程序启动指南使用说明		PDF
78K0S/Kx1+举例程序(中断)由开关输入产生的外部中断使用说明		PDF

附录A 程序清单

78K0S/KB1+ 微控制器的源文件作为一个程序清单实例显示如下：

● main.asm（汇编语言版）

```
; ********************************************************************************
;
;       NEC Electronics    78K0S/KB1+
;
; ********************************************************************************
;       78K0S/KB1+    举例程序
; ********************************************************************************
;       低电压检测
; ********************************************************************************
; <<历史记录>>
;       2007.6.--         发布
; ********************************************************************************
;
; <<概要>>
;
; 本举例程序提供了一个低电压检测（LVI）功能的应用实例。
;
; 低电压检测器(LVI)用于设置以便 VDD 低于 VLVI（2.85 V ± 0.15 V）时，
; 产生内部复位信号。
; 初始设置完成后， LED 灯的亮灭模式随开关输入次数而改变。
; （这个同本举例程序中断中的描述一致。）
; 此时，如果复位由非 LVI 产生，则初始化开关输入次数，
; 但是，当复位由 LVI 产生时，保留复位之前的开关输入次数，
; 然后据此显示 LED 灯的亮灭模式，
; 这是因为已经保留了 RAM 数据。
;
;
;   <主要设置内容>
;
;   - 停止看门狗定时器的运行。
;   - 将低电压检测电压设置（VLVI）为 2.85 V ± 0.15 V。
;   - VDD >= VLVI 之后，当 VDD < VLVI 时，产生内部复位信号（低电压检测器）。
;   - 将 CPU 时钟频率设置为 4 MHz。
;   - 将外部中断 INTP1 的有效沿设置为下降沿。
;   - 设置开关输入期间的消抖动时间为 10 ms。
;
;
;   <低电压检测和复位解除后 LED 灯的亮灭模式>
;
;   -复位由非低电压检测器产生... 所有 LED 灯都熄灭。
;   - 复位由低电压检测器产生 ... 保持复位前 LED 灯的亮灭模式。
;
;
;   <开关输入次数和 LED 灯的亮灭模式>
;
```

```
;    +-------------------------------------+
;    | SW 输入 | LED3 | LED2 | LED1 |
;    |  (P43)    | (P22) | (P21) | (P20) |
;    |-----------|-----------------------|
;    |  0 次 | 关 | 关| 关 |
;    |  1 次  | 关 | 关 | 开  |
;    |  2 次 | 关 | 开  | 关 |
;    |  3 次 | 关 | 开  | 开 |
;    |  4 次 | 开  |关 | 关 |
;    |  5 次 | 开 | 关 | ON  |
;    |  6 次 | 开  | 开  | 关 |
;    |  7 次 | 开 | 开  | 开  |
;    +-------------------------------------+
;    #第八次开关输入后，灯的模式开始从第零次开关输入时的模式进行重复。
;
;
; <<I/O 端口设置>>
;
;  输入：P43
;  输出：P00-P03、P20-P23、P30-P33、P40-P42、P44-P47、P120-P123、P130
;  # 所用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     INTERRUPT_P1       ; (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 RAM
;
; ===========================================================================
```

```
XRAM  DSEG  SADDR
CNT_1：      DS    1              ；用于外层循环。
CNT_2：      DS    1              ；用于内层循环。
LEDDATA：    DS    1              ；LED 显示模式变量。

；=============================================================================
；
；       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。

；*****************************************************************************
；
；       RESET 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。

；-----------------------------------------------------------------------------
；       检测低电压 + 初始化看门狗定时器 + 设置时钟
；-----------------------------------------------------------------------------
；----- 初始化看门狗定时器 -----
       MOV   WDTM，       #01110111B    ；停止看门狗定时器的操作。

；----- 设置时钟 <1> -----
       MOV   PCC，  #00000000B    ；提供至 CPU 的时钟(fcpu) = fxp (= fx/4 = 2 MHz)。
       MOV   LSRCM，      #00000001B    ；停止内部低速振荡器的振荡。

；----- 检查复位信号源 -----
       MOV   A，    RESF          ；读取复位信号源。
       BT    A.0，  $SET_CLOCK    ；复位期间省略后继 LVI 相关处理，并转到 SET_CLOCK 处。

；----- 设置低电压检测 -----
       MOV   LVIS，  #00000111B   ；将低电压检测电压(VLVI)设置为 2.85 V ± 0.15 V。
       SET1  LVION                ；允许低电压检测器操作。

       MOV   A，    #40           ；指定 200 us 的等待计数值。
；----- 等待 200 us -----
WAIT_200US：
       DEC   A
       BNZ   $WAIT_200US          ；0.5[us/clk] x 10[clk] x 40[计数值] = 200[us]。

；----- VDD >= VLVI 等待处理 -----
```

```
WAIT_LVI：
        NOP
        BT      LVIF，  $WAIT_LVI       ；如果 VDD < VLVI，则发生转移。

        SET1    LVIMD                   ；如此设置以便当 VDD < VLVI 时产生内部复位信号。

        MOV     LEDDATA，  #00000111B       ；初始化 LED 显示数据。

；----- 设置时钟 <2> -----
SET_CLOCK：
        MOV     PPCC，          #00000001B      ；提供至外围硬件的时钟(fxp) = fx/2 (= 4 MHz)。
                                                ；  ->提供至 CPU 的时钟(fcpu) = fxp = 4 MHz。

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

；-------------------------------------------------------------------------
；       初始化端口 2
；-------------------------------------------------------------------------
        MOV     A，     LEDDATA         ；读取 LED 显示数据。
        MOV     P2，    A               ；设置 LED 输出(P20-P22)和 P23 的输出锁存为低电平。
        MOV     PM2，   #11110000B      ；设置 P20-P23 为输出模式。

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

；-------------------------------------------------------------------------
；       初始化端口 4
；-------------------------------------------------------------------------
        MOV     P4，    #00000000B      ；设置 P40-P47 的输出锁存为低电平。
        MOV     PU4，   #00001000B      ；连接片上上拉电阻至 P43。
        MOV     PM4，   #00001000B      ；设置 P43 为输入模式，P40-P42 和 P44-P47 为输出模式。

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

；-------------------------------------------------------------------------
；       初始化端口 13
；-------------------------------------------------------------------------
        MOV     P13，   #00000001B      ；设置 P130 的输出锁存为高电平。
；-------------------------------------------------------------------------
；       设置中断
；-------------------------------------------------------------------------
        MOV     INTM0，         #00000000B      ；  设置 INTP1 下降沿有效。
```

```
        CLR1   PIF1                    ；预先清除无效中断请求。
        CLR1   PMK1                    ；解除 INTP1 中断屏蔽。

        EI                             ；允许向量中断。

；*****************************************************************************
；
；       主循环
；
；*****************************************************************************
MAIN_LOOP：
        NOP
        BR     $MAIN_LOOP              ；转到 MAIN_LOOP。

；*****************************************************************************
；
；       外部中断 INTP1
；
；*****************************************************************************
INTERRUPT_P1：
        PUSH   AX                      ；将 AX 的值保存到堆栈。

；----- 10 ms 等待以处理抖动 -----
        MOV    CNT_1，         #215    ；为外层循环指定计数值。
        NOP
        NOP
LOOP_1：
        MOV    CNT_2，         #17     ；为内层循环指定计数值。
        NOP
LOOP_2：
        NOP
        DBNZ   CNT_2，         $LOOP_2             ；内层循环。
        DBNZ   CNT_1，         $LOOP_1             ；外层循环。

        CLR1   PIF1                    ；清除 INTP1 中断请求。

；----- 抖动检测的识别 -----
        BT     P4.3，  $END_INTP1      ；如果没有开关输入，则进行转移。

；----- LED 灯的处理 -----
        DEC    LEDDATA                 ；当前 LED 的显示数据减 1。
        AND    LEDDATA，  #00000111B       ；屏蔽非位 0 至位 2。
        MOV    A，     LEDDATA         ；读取 LED 显示数据。
        MOV    P2，    A               ；输出 LED 灯。

END_INTP1：
        POP    AX                      ；恢复 AX 寄存器的值。
        RETI                           ；从中断服务程序中返回。

end
```

● main.c（C 语言版）

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

        NEC Electronics     78K0S/KB1+

************************************************************************************
        78K0S/KB1+     举例程序
************************************************************************************
        低电压检测
************************************************************************************
<<历史记录>>
        2007.6.--          发布
************************************************************************************

<<概要>>

本举例程序介绍了使用低电压检测(LVI)功能的一个实例。
低电压检测器(LVI)用来设置以便当 VDD 低于 VLVI (2.85 V ± 0.15 V)时，
产生内部复位信号。
初始设置完成后，LED 灯的亮灭模式随开关输入的次数而改变。
（这个与本举例程序中断中的描述一致。）
此时，如果复位由非 LVI 产生，则初始化开关输入次数，
但是，当复位由 LVI 产生时，保留复位之前的开关输入次数，并且据此显示 LED 灯的亮灭模式
这是因为已经保留了 RAM 中的数值。


 <主要设置内容>

 - 声明一个通过中断运行的函数：INTP1 -> fn_intp1()。
 - 停止看门狗定时器的运行。
 - 设置低电压检测电压(VLVI)为 2.85 V ± 0.15 V。
 - VDD >= VLVI 之后，当 VDD < VLVI 时，产生内部复位信号（低电压检测器）。
 - 设置 CPU 时钟频率为 4 MHz。
 - 设置外部中断 INTP1 的有效沿为下降沿。
 - 设置开关输入期间的抖动检测时间为 10 ms。


 <低电压检测和复位解除后 LED 灯的亮灭模式>

 -复位由非低电压检测器产生... 所有 LED 灯都熄灭。
 - 复位由低电压检测器产生... 保持复位前 LED 灯的亮灭模式。


 <开关输入次数和 LED 灯模式>

  +-------------------------------------+
  | SW 输入 |  LED3 |  LED2 |  LED1 |
  |   (P43)     |  (P22) | (P21)   |  (P20) |
  |-------------|-----------------------|
  |   0 次  |  关 |  关|  关|
  |   1 次   |  关  |  关  |  开    |
  |   2 次  |  关  |开   |  关  |
  |   3 次  |  关  |  开    |开   |
  |   4 次  |开   |  关  |  关  |
```

```
  |  5 次 | 开  | 关 | 开  |
  |  6 次 | 开  | 开  | 关 |
  |  7 次 |开  | 开  | 开  |
  +-----------------------------------+
  #第八次开关输入后，灯的模式开始从第零次开关输入时的模式进行重复。


<<I/O 端口设置>>

 输入： P43
 输出： P00-P03、P20-P23、P30-P33、P40-P42、P44-P47、P120-P123、P130
 # 所用未用端口设置为输出模式。

*****************************************************************************/


/*===========================================================================

        预处理指示符（#pragma）

===========================================================================*/
#pragma         SFR                             /* SFR 名称能够在 C 语言层面上进行描述 */
#pragma         EI                              /* EI 中断能够在 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 */
#pragma interrupt INTP1 fn_intp1        /* 中断函数声明：INTP1 */

/*****************************************************************************

        RESET 后的初始化

*****************************************************************************/
        sreg unsigned char g_ucLED;   /* 用于 LED 显示数据的 8 位变量（内部高速 RAM 区） */

void hdwinit(void){
        unsigned char ucCnt200us;       /* 用于 200 us 等待的 8 位变量 */

/*---------------------------------------------------------------------------
        初始化看门狗定时器 + 检测低电压 + 设置时钟
---------------------------------------------------------------------------*/
        /* 初始化看门狗定时器 */
        WDTM  = 0b01110111;                     /* 停止看门狗定时器的运行 */

        /* 设置时钟 <1> */
        PCC   = 0b00000000;             /* 提供至 CPU 的时钟 (fcpu) = fxp (= fx/4 = 2 MHz) */
        LSRCM = 0b00000001;                     /* 停止内部低速振荡器的振荡 */

        /* 检查复位信号源 */
        if (!(RESF & 0b00000001)){      /* LVI 复位期间，省略后继 LVI 相关处理 */

                /* 设置低电压检测 */
                LVIS  = 0b00000111;   /* 设置低电压检测电压为 2.85 V ± 0.15 V */
                LVION = 1;                      /*低电压检测器操作允许*/

                for (ucCnt200us = 0;  ucCnt200us < 9;  ucCnt200us++){           /* 等待约 200 us */
                        NOP();
```

```
            }

            while (LVIF){           /* 等待 VDD >= VLVI */
                NOP();
            }

            LVIMD = 1;                      /* 如此设置以便当 VDD < VLVI 时，产生内部复位信号 */

            g_ucLED = 0b00000111;           /* 初始化 LED 显示数据 */
        }

        /* 设置时钟 <2> */
        PPCC  = 0b00000001;         /* 提供至外围硬件的时钟(fxp) = fx/2 (= 4 MHz)
                                            -> 提供至 CPU 的时钟(fcpu) = fxp = 4 MHz */

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

/*------------------------------------------------------------------------
        初始化端口 2
------------------------------------------------------------------------*/
        P2    = g_ucLED;                    /* 输出 LED 显示数据 */
        PM2   = 0b11110000;         /* 设置 P20-P23 为输出模式 */

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

/*------------------------------------------------------------------------
        初始化端口 4
------------------------------------------------------------------------*/
        P4    = 0b00000000;         /* 设置 P40-P47 的输出锁存为低电平 */
        PU4   = 0b00001000;         /* 连接片上上拉电阻至 P43 */
        PM4   = 0b00001000;         /* 设置 P43 为输入模式， P40-P42 和 P44-P47 为输出模式 */

/*------------------------------------------------------------------------
        初始化端口 12
------------------------------------------------------------------------*/
        P12   = 0b00000000;         /* 设置 P120-P123 的输出锁存为低电平 */
        PM12  = 0b11110000;         /* 设置 P120-P123 为输出模式 */

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

/*------------------------------------------------------------------------
        设置中断
------------------------------------------------------------------------*/
        INTM0 = 0b00000000;         /* 设置 INTP1 下降沿有效*/
```

```
        PIF1  = 0;                          /* 预先清除无效中断请求 */
        PMK1  = 0;                          /* 解除 INTP1 中断屏蔽 */

        return;
}

/***************************************************************************

        Main loop

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

        EI();                               /* 允许向量中断 */

        while (1){
                NOP();
                NOP();
        }
}

/***************************************************************************

        外部中断 INTP1

***************************************************************************/
__interrupt void fn_intp1(){
        unsigned int unChat;    /* 用于消抖动定时器的 16 位变量 */

        for (unChat = 0;  unChat < 555;  unChat++){    /* 等待约 10 ms（用于消抖动） */
                NOP();
        }

        PIF1  = 0;                          /* 清除 INTP1 中断请求*/

        if (!P4.3){                 /* 如果 SW 接通达 10 ms 或更长时间，则执行处理 */
                g_ucLED -= 1;               /* 当前 LED 显示数据减 1 */
                g_ucLED &= 0b00000111;      /* 屏蔽除位 0 至位 2 外的其它位 */
                P2 = g_ucLED;               /* 输出 LED 灯 */
        }

        return;
}
```

● op.asm（汇编语言版和 C 语言版共用）

```
;  ============================================================================
;
;       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 (8 MHz)选择位系统时钟信号源。
```

```
;                +---------- P34/RESET 引脚用作 RESET 引脚。

      DB     11111111B      ;  保护字节区域（用于自编程模式）。
;              ||||||||
;              ++++++++---------- 所用块可进行写入或擦除。

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

[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

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

[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

[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

[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