NEC

应用注释

78K0S/Kx1+

示例程序 (8-位定时器 H1)

计时器

本文件包含示例程序操作概述，使用方法以及怎样设置和使用 8-位定时器 H1 的计时器功能。在示例程序中，使用 8-位定时器 H1 计时器功能可实现 LED 在固定周期闪亮。另外，闪亮周期随开关输入次数变化。

目标设备

78K0S/KA1+ 微处理器
78K0S/KB1+微处理器
78K0S/KU1+微处理器
78K0S/KY1+微处理器

文件号： U18862CA1V0AN00 （第一版）
出版日期： 2008 年 03 月 N

日文版

目录

第一章 概述…… 3
1.1 初始设置的主要内容……..3
1.2 主循环之后的内容…… 4
第二章 电路图…… 5
2.1 电路图……..5
2.2 外围硬件…… 5
第三章 软件…… 7
3.1 文件配置……. 7
3.2 所用内部外围功能…… 8
3.3 初始设置及运行概览……. 8
3.4 流程图…… 10
第四章 设置方法…… 11
4.1 设置 8 位定时器H1 的间隔定时器功能…… 11
4.2 设置LED闪烁周期和颤动检测时间…….... 18
第五章 用系统仿真器 SM+进行运行检查…….. 22
5.1 构建示例程序…… 22
5.2 随SM+运行…… 23
第六章 相关文件…… 28
附件 A 程序列表…… 29
附件B 修订记录…… 42

· 本文档中的信息于 2008 年 3 月开始使用。文档内容可能会不作通知进行修改。实际设计请参阅日电电子最新发布的数据表或数据册等，查看日电电子产品的最新指标。并非所有产品和/或类型在每个国家都能使用。请联系日电电子销售代表，了解可用性信息及其他信息。
· 未经日电电子书面许可，不能以任何形式或方式对本文档的任何部分进行复制或重现。本文档出现的任何错误，日电电子不承担责任。
· 对于在使用本文档列出的日电电子产品时产生的侵犯专利、版权以及其他第三方知识产权的行为，以及对于其他使用这些产品产生的责任，日电电子不承担责任。对于日电电子及其他子公司的任何专利、版权以及其他知识产权，日电电子没有以许可、明示、暗示以及其他任何方式授权。
· 本文档中对电路、软件及其他相关信息的描述旨在说明半导体产品的操作及应用举例。这些电路、软件和信息在客户设备设计中的使用应由客户承担全部责任。如果这些电路、软件和信息导致客户或第三方遭受损失，日电电子不承担责任。
· 日电电子尽力提升日电电子产品的质量、可靠性和安全性，但请客户同意并理解这些产品的瑕疵无法完全消除。为了尽量减少由于日电电子产品导致的财产损失或人身伤害（包括死亡），客户必须在其设计中采取足够的安全措施，如冗余、防火、防故障等特性。
· 日电电子产品分为下列三种质量等级：“标准”、“特别”及“专用”。
“专用”质量等级只适用于基于用户设计的“质量保证项目”的特定应用开发的日电电子产品。日电电子的建议应用由其质量级别决定，如下所示。客户在将日电电子产品用于特别用途之前必须检查各产品的质量等级。
“标准”：计算机、办公设备、通信设备、测试测量设备、音频视频设备、家用电子产品、机械工具、个人电子设备、工业机器人。
“特别”：运输设备（汽车、火车、轮船等）、交通控制系统、抗灾系统、防犯罪系统、安全设备、医疗设备（不专为生命救护而设计）。
“专用”：飞机、航空设备、水下中继器、核反应堆控制系统、生命救护系统、用于生命救护的医疗设备等。

除非在日电电子的数据表或数据册等当中有明确说明，日电电子产品的质量级别均为“标准”。客户如果希望日电电子产品实现日电电子未预定的应用，必须提前联系日电电子的销售代表以确定日电电子愿意支持给定应用。

（注）
（1）本声明中所用的“日电电子”表示日电电子公司，包括其控股的子公司。
（2）“日电电子产品”表示由日电电子（如上所规定）开发或制造的任何产品。

M&E 02 11-1

第一章 概述

本示例程序展示使用 8 位定时器 H1 的间隔定时器功能的示例。LED 以固定周期闪烁，LED 的闪烁周期根据开关输入的次数进行改变。

1.1 初始设置的主要内容

初始设置的主要内容如下。

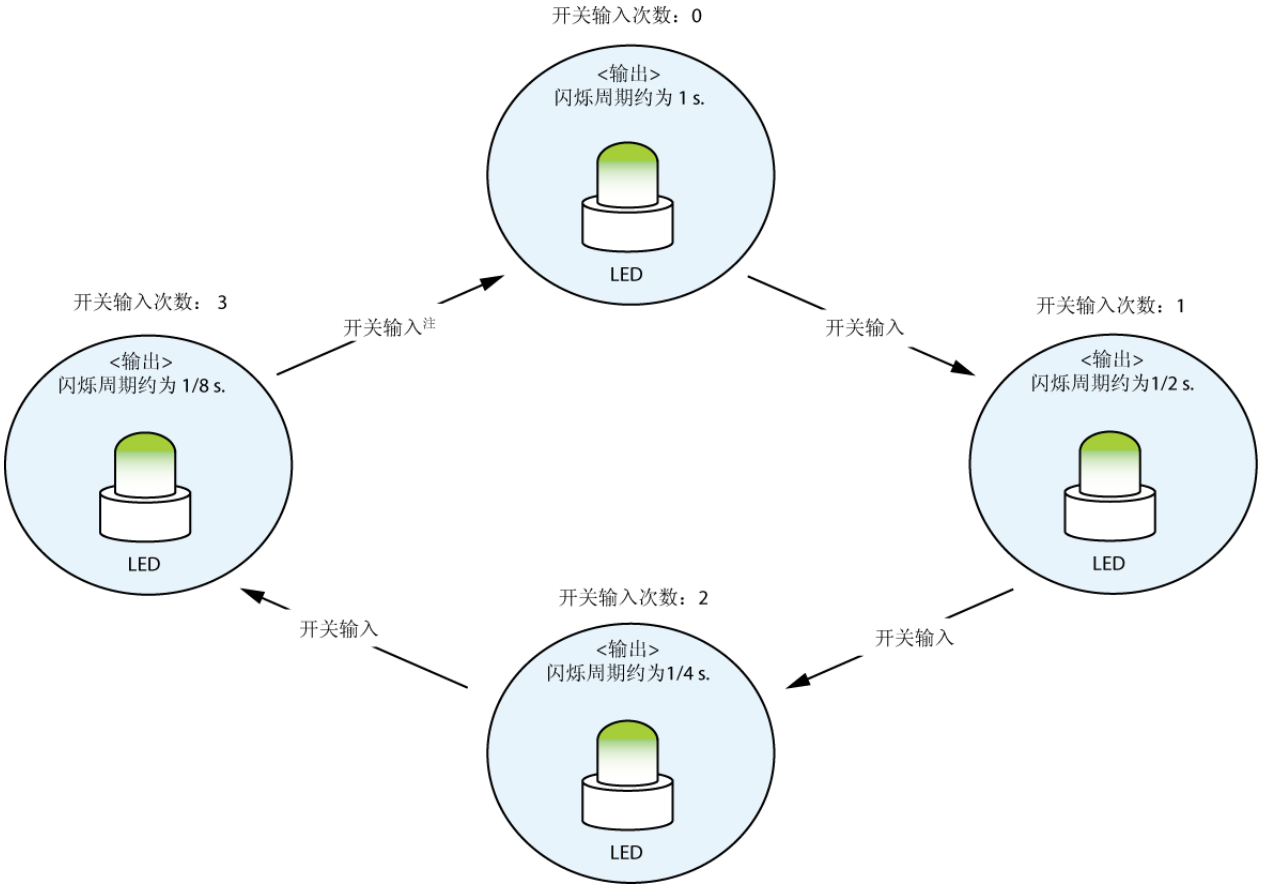
- 选择高速内部振荡器作为系统时钟源[注]
- 停止看门狗计时器的运行
- 将 VLVI（低压检测电压）设置为 4.3V±0.2V
- 在 VDD（供电电压）大于等于 VLVI 后，当检测到 VDD 小于 VLVI 时，生成内部复位（LVI 复位）信号
- 将 CPU 时钟频率设置为 8MHz
- 设置 I/O 端口
- 设置 8 位定时器 H1
  - 将计数时钟设置为 fXP/2^6（125kHz），设置工作模式为间隔定时器模式，禁用 TOH1 的定时器输出
  - 设置时间间隔周期为 2ms（8μs×250）
- 将 INTP1（外部中断）的有效沿设置为下降沿
- 启用 INTP1 和 INTTMH1 中断

注 通过选项字节进行设置。

1.2 主循环之后的内容

初始设置完成后，利用 8 位定时器 H1 中断（INTTMH1）的生成，LED 以固定周期闪烁。

在检测到 INTP1 引脚的下降沿时处理 INTP1 中断，该沿是由开关输入生成的。在检测到 INTP1 引脚的下降沿后经过 10ms，当 INTP1 处于高电平（开关关闭）时发现抖动。检测到沿后经过 10ms，当 INTP1 处于低电平（开关打开）时，LED 的闪烁周期根据开关输入的次数进行改变。



注 在第四次开关输入后，重复第零次开关输入时的闪烁周期。

注意事项 关于使用设备的注意事项，请参见各产品（78K0S/KU1+、78K0S/KY1+、78K0S/KA1+、78K0S/KB1+）的用户手册。

[列]抖动

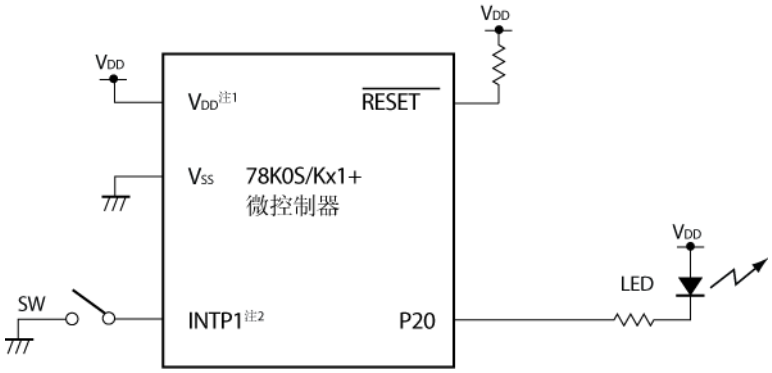
抖动，是在开关按下后瞬间由于接触点的机械开合而导致电信号反复开关的现象。

第二章 电路图

本章描述本示例程序中所用的电路图和外围硬件。

2.1 电路图

电路图如下所示。



注 **1.** 工作电压范围为 4.5V≤V_{DD}≤5.5V。

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

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

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

3. 除电路图中所示引脚及 AV_{REF}、AV_{SS} 外，其他所有不用的引脚保留开路（不连接）。

2.2 外围硬件

所用外围硬件如下所示。

（1）开关（SW）

开关用来作为控制 LED 发光的输入信号。

（2）LED

LED 用来作为对应 8 位定时器 H1 的间隔定时器功能和开关输入的输出。

第三章　软件

本章描述所下载的压缩文件的文件配置、所用微控制器的内部外围功能、示例程序的初始设置和运行概览，并展示流程图。

3.1　文件配置

下表展示所下载的压缩文件的文件配置。

文件名	描述	包含压缩（*.zip）文件		
				
main.asm （汇编语言版） --- main.c （C语言版）	用于微控制器的硬件初始化处理及主处理的源文件	●注1	●注1	
op.asm	用于设置选项字节（设置系统时钟源）的汇编器源文件	●	●	
tmh1.prw	用于集成开发环境PM+的工作区文件		●	
tmh1.prj	用于集成开发环境PM+的项目文件		●	
tmh1.pri tmh1.prs tmh1.prm	用于适用78K0S/Kx1+的系统仿真器SM+的项目文件		●注2	
tmh10.pnl	适用78K0S/Kx1+的系统仿真器SM+的I/O面板文件（用于检查外围硬件的工作）		●注2	●
tmh10.wvo	适用78K0S/Kx1+的系统仿真器SM+的时间图文件（用于检查波形）			●

注　**1.** “main.asm”包含在汇编语言版中，“main.c”包含在 C 语言版中。

2. 这些文件不包含在 78K0S/KU1+微控制器的文件中。

备注　  ：　仅包含源文件。

 ：包含与集成开发环境 PM+和 78K0S/Kx1+系统仿真器 SM+一起使用的文件。

 ：包含与适用 78K0S/Kx1+的系统仿真器 SM+一起使用的微控制器运行仿真文件。

3.2 所用内部外围功能

本示例程序使用微控制器的下列内部外围功能。

- 间隔定时器功能： 8 位定时器 H1
- V_{DD}<V_{LVI} 检测： 低压检测器（LVI）
- 开关输入： INTP1[注]（外部中断）
- LED 输出： P20（输出端口）

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

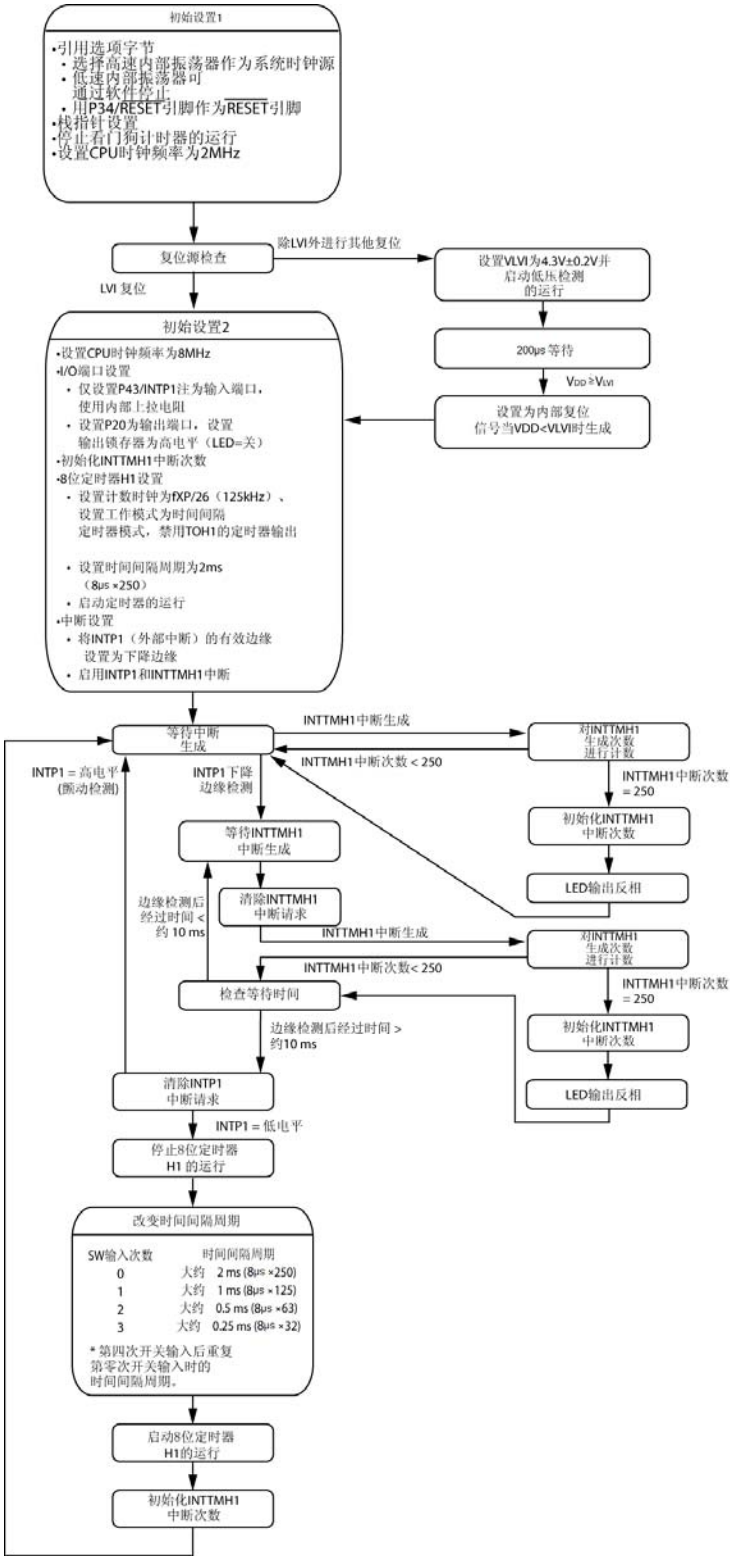
3.3 初始设置及运行概览

本示例程序进行的操作有：包括低压检测功能在内的初始设置、时钟频率的选择、I/O 端口的设置、8 位定时器 H1（间隔定时器）的设置、中断设置。

初始设置完成后，利用 8 位定时器 H1 中断（INTTMH1）的生成，LED 以固定周期闪烁。

在检测到 INTP1 引脚的下降沿时处理 INTP1 中断，该沿是由开关输入生成的。在检测到 INTP1 引脚的下降沿后经过 10ms，当 INTP1 处于高电平（开关关闭）时发现抖动。检测到沿后经过 10ms，当 INTP1 处于低电平（开关打开）时，LED 的闪烁周期根据开关输入的次数进行改变。

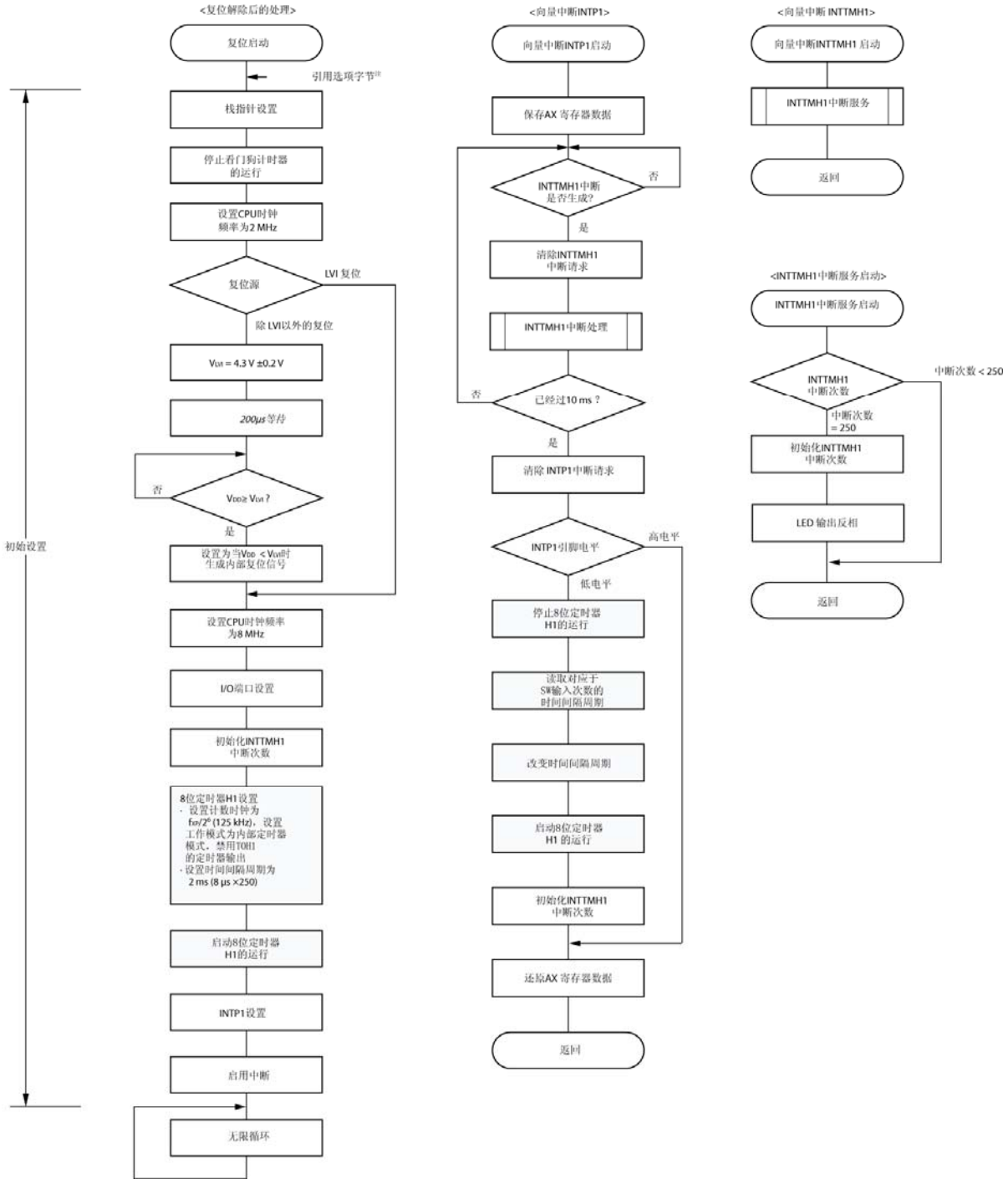
详情在如下所示的状态转换图中描述。



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

3.4 流程图

示例程序的流程图如下所示。



注 对选项字节的引用由微控制器在复位解除后自动进行。在本示例程序中，下面的内容通过引用选项字节进行设置。

- 用高速内部时钟（8MHz（典型））作为系统时钟源
- 低速内部振荡器可用软件停止
- 用 P34/$\overline{\text{RESET}}$ 引脚作为 $\overline{\text{RESET}}$ 引脚

第四章 设置方法

本章描述 8 位定时器 H1 的间隔定时器功能。

关于其他初始设置，请参见**78K0S/Kx1+示例程序（初始设置）LED发光开关控制的应用注释**。关于中断，请参见**78K0S/Kx1+示例程序（中断）由开关输入生成的外部中断的应用注释**。关于低压检测（LVI），请参见**78K0S/Kx1+示例程序（低压检测）在小于 2.7V检测过程中的复位生成的应用注释**。

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

关于汇编器指令，请参见**78K/0S系列指令用户手册**。

4.1 设置 8 位定时器H1 的间隔定时器功能

当使用 8 位定时器 H1 时，将设置下面的五类寄存器。

- 8 位定时器 H 模式寄存器 1（TMHMD1）
- 8 位定时器 H 比较寄存器 01（CMP01）
- 端口模式寄存器 x（PMx）注
- 端口寄存器 x（Px）注
- 端口模式控制寄存器 x（PMCx）注

注 要将 TOH1 引脚用于定时器输出，应照下面所示进行设置。

	Px 寄存器	PMx 寄存器	PMCx 寄存器
78K0S/KA1+和 78K0S/KB1+微控制器	P42=0	PM42=0	无需设置
78K0S/KY1+和 78K0S/KU1+微控制器	P20=0	PM20=0	PMC20=0

（1）8 位定时器 H1 的工作模式的相关设置

对于 8 位定时器 H1，利用 8 位定时器 H 模式寄存器 1（TMHMD1）设置工作模式、选择计数时钟、控制操作。

图 4-1　8 位定时器 H 模式寄存器 1（TMHMD1）的格式

TMHMD1

TMHE1	CKS12	CKS11	CKS10	TMMD11	TMMD10	TOLEV1	TOEN1

定时器输出 (TOH1) 控制

0	禁用输出。
1	启用输出。

定时器输出电平控制 (默认)

0	低电平
1	高电平

定时器工作模式

0	0	间隔定时器模式
1	0	PWM输出模式
除上述外		禁止设置

计数时钟的选择

0	0	0	f_{XP}
0	0	1	$f_{XP}/2^2$
0	1	0	$f_{XP}/2^4$
0	1	1	$f_{XP}/2^6$
1	0	0	$f_{XP}/2^{12}$
1	0	1	$f_{RL}/2^7$
除上述外			禁止设置

启用定时器的运行

0	停止运行（计数器清为0）。
1	启用运行。

注意事项　当 TMHE1 设置为 1 时，禁止设置 TMHMD1 寄存器的其他各位。

备注　f_{XP}：供给外围硬件的时钟的振荡频率

f_{RL}：内部低速振荡时钟频率

（2）间隔时间设置

利用 8 位定时器 H 比较寄存器 01（CMP01）进行间隔时间的设置。

●间隔时间=（N+1）/f_{CNT}

备注　N：　CMP01 设置值（00H 至 FFH）

f_{CNT}：　8 位定时器 H1 的计数时钟频率

图 4-2　8 位定时器 H 比较寄存器 01（CMP01）的格式

CMP01

注意事项　禁止在定时器计数操作过程中重写 CMP01 寄存器的值。

（3）TOH1 引脚设置

当定时器输出在间隔定时器模式中启用时，一旦 8 位定时器计数器 H1 和 CMP01 寄存器的值相匹配，TOH1 引脚的输出电平就会反相。

要将 TOH1 引脚用于定时器输出，应对端口寄存器 x（Px）、端口模式寄存器 x（PMx）和端口模式控制寄存器 x（PMCx）进行如下设置。

	Px 寄存器	PMx 寄存器	PMCx 寄存器
78K0S/KA1+和 78K0S/KB1+微控制器	P42=0	PM42=0	无需设置
78K0S/KY1+和 78K0S/KU1+微控制器	P20=0	PM20=0	PMC20=0

[例 1] • 将 8 位定时器 H1 的工作模式设置为间隔定时器模式，将计数时钟设置为 $f_{XP}/2^6$（f_{XP}=8MHz），禁用定时器输出（TOH1）

• 设置时间间隔周期为 2ms，启动定时器的运行

（与示例程序中的内容相同）

TMHMD1

TMHE1	0	1	1	0	0	x	0

定时器输出 (TOH1) 控制

0	禁用输出。

定时器输出电平控制 (默认)

x	（因为不使用定时器输出，故设置无效。）

定时器工作模式

0	0	间隔定时器模式

计数时钟的选择

0	1	1	$f_{XP}/2^6$

启用定时器的运行

0	停止运行（计数器清为0）。
1	启用运行。

CMP01 的设置值（N）：249

•计数时钟 f_{CNT} = 8MHz/2^6= 0.125MHz =125kHz

•时间间隔周期 2ms=（N+1）/125kHz

→N= 2ms×125kHz−1=249

先将“001100x0”（x：无关）（在如下所示的例子值“x”设置为 0）设置给 TMHMD1、“249”设置给 CMP01 再将 1 设置给 TMHE1 即可启动定时器的运行。

•汇编语言

```
MOV    TMHMD1, #00110000B
MOV    CMP01,  #249
SET1   TMHE1
```

•C 语言

```
TMHMD1 = 0b00110000;
CMP01 = 249;
TMHE1 = 1;
```

[例 2] • 将 8 位定时器 H1 的工作模式设置为间隔定时器模式，将计数时钟设置为 f_{XP}（f_{XP}=8MHz），将定时器输出电平（默认）设置为低电平

• 设置时间间隔周期为 31.25μs，启动定时器的运行

TMHMD1

TMHE1	0	0	0	0	0	0	1

定时器输出 (TOH1) 控制

1	启用输出。

定时器输出电平控制 (默认)

0	低电平

定时器工作模式

0	0	间隔定时器模式

计数时钟的选择

0	0	0	f_{XP}

启用定时器的运行

0	停止运行（计数器清为0）。
1	启用运行。

CMP01 的设置值（N）：249

•计数时钟 f_{CNT}=8MHz

•时间间隔周期 31.25μs=（N+1）/8MHz

→N=31.25μs×8MHz−1=249

TOH1 引脚设置

•78K0S/KA1+和 78K0S/KB1+微控制器：P42=0、PM42=0

•78K0S/KY1+和 78K0S/KU1+微控制器：P20=0、PM20=0、PMC20=0

对于 78K0S/KA1+和 78K0S/KB1+微控制器，先设置“0”给 P42、设置“0”给 PM42、设置“00000001”给 TMHMD1、设置“249”给 CMP01 再设置 1 给 TMHE1，即可启动定时器的运行。

对于 78K0S/KY1+和 78K0S/KU1+微控制器，先设置“0”给 P20、设置“0”给 PM20、设置“00000001”给 TMHMD1、设置“249”给 CMP01 再设置 1 给 TMHE1，即可启动定时器的运行。

•汇编语言（使用 78K0S/KA1+和 78K0S/KB1+微控制器）

```
CLR1    P4.2
CLR1    PM4.2
MOV     TMHMD1, #00000001B
MOV     CMP01,  #249
SET1    TMHE1
```

•C 语言（使用 78K0S/KA1+和 78K0S/KB1+微控制器）

```
P4.2 = 0;
PM4.2 = 0;
TMHMD1 = 0b00000001;
CMP01 = 249;
TMHE1 = 1;
```

●汇编语言程序示例（与上述[例 1]和示例程序内容相同）

```
XMAIN   CSEG    UNIT
RESET_START:
                •
                •
                •
        MOV     TMHMD1, #00110000B      ; 计数时钟 = fxp/2^6 = 125 kHz,
                                        ; 设置为间隔定时器模式
        MOV     A,      [HL]            ; 从表中读取 LED 闪烁的基准时间初始值
        MOV     CMP01,  A               ; 初始化比较值
        SET1    TMHE1                   ; 启动定时器的运行

        MOV     INTM0,  #00000000B      ; 将 INTP1 的有效沿设置为下降沿
        MOV     IF0,    #00H            ; 事先清除无效中断请求
        CLR1    PMK1                    ; Intp1 中断去屏蔽
        CLR1    TMMKH1                  ; Inttmh1 中断去屏蔽

        EI                              ; 启用向量中断
MAIN_LOOP:
        NOP
        BR      $MAIN_LOOP              ; 转到 MAIN_LOOP
                •
                •
                •
        CLR1    TMHE1                   ; 停止定时器的运行
        MOV     A,      L               ; 读取表格地址的低 8 位
        INC     A                       ; 表格地址增加 1
        AND     A,      #00000011B      ; 屏蔽除 0 位和 1 位以外的各位
        MOV     L,      A               ; 写入表格地址的低 8 位
        MOV     A,      [HL]            ; 读取表格数据
        MOV     CMP01,  A               ; 改变LED闪烁的基准时间
        SET1    TMHE1                   ; 启动定时器的运行
                •
                •
                •
INTERRUPT_TMH1:
        CALL    !SUB_INTERRUPT_TMH1     ; 处理 INTTMH1 中断
        RETI                            ; 从中断服务返回
                •
                •
                •
```

设置间隔时间

设置 8 位定时器 H1 的工作模式和计数时钟

清除 INTTMH1 中断请求标志

启动定时器的运行

启用 INTTMH1 中断服务

停止定时器运行后设置 CMP01 寄存器

启动定时器的运行

通过 INTTMH1 中断生成启动中断服务

•C语言程序示例（与上述[例 1]和示例程序内容相同）

```
void hdwinit(void){
        unsigned char ucCnt200us;          /* 用于 200 us 等待的 8 位变量 */
                    •
                    •
                    •
        TMHMD1 = 0b00110000;               /* 计数时钟 = fxp/2^6 = 125 kHz, */
                                           /* 设置为间隔定时器模式 */
        CMP01  = 250-1;                    /* 初始化LED闪烁基准时间 */
        TMHE1 = 1;                         /* 启动定时器的运行 */

        INTM0 = 0b00000000;                /* 将 INTP1 的有效沿设置为下降沿 */
        IF0   = 0x00;                      /* 事先清除无效中断请求 */
        PMK1  = 0;                         /* INTP1 中断去屏蔽 */
        TMMKH1 = 0;                        /* INTTMH1 中断去屏蔽 */
        return;
}

void main(void){

        EI();                              /* 启用向量中断 */

        while (1){
                NOP();
                NOP();
        }
}
                    •
                    •
                    •
        TMHE1 = 0;                         /* 停止定时器的运行 */
        CMP01 = g_ucCMPdata[g_ucSWcnt];
/* 输入次数改变 LED 闪烁的基准时间 */
        TMHE1 = 1;                         /*  启动定时器的运行 */
                    •
                    •
                    •
interrupt void fn_inttmH1(){

        fn_subinttmH1();                   /* 处理 INTTMH1 中断 */

        return;
                    •
                    •
                    •
```

设置间隔时间

设置 8 位定时器 H1 的工作模式和计数时钟

清除 INTTMH1 中断请求标志

启动定时器的运行

启用 INTTMH1 中断服务

停止定时器运行后设置 CMP01 寄存器

启动定时器的运行

通过 INTTMH1 中断生成启动中断服务

4.2 设置LED闪烁周期和抖动检测时间

在本示例程序中，LED 闪烁周期和抖动检测时间的设置如下。

（1）设置 LED 闪烁周期

在本示例程序中，8 位定时器 H1 中断（INTTMH1）每生成 250 次，LED 输出就会反相。

- 中断周期（间隔时间）=（N+1）/f_{CNT}
- LED 输出反相周期=中断周期×中断次数
- LED 闪烁周期=LED 输出反相周期×2

备注 N： CMP01 寄存器设置值

f_{CNT}： 8 位定时器 H1 的计数时钟频率

计算示例： 当 CMP01 寄存器的设置值为 249 时会出现下面的结果（在 f_{CNT}=125kHz 的操作过程中）。

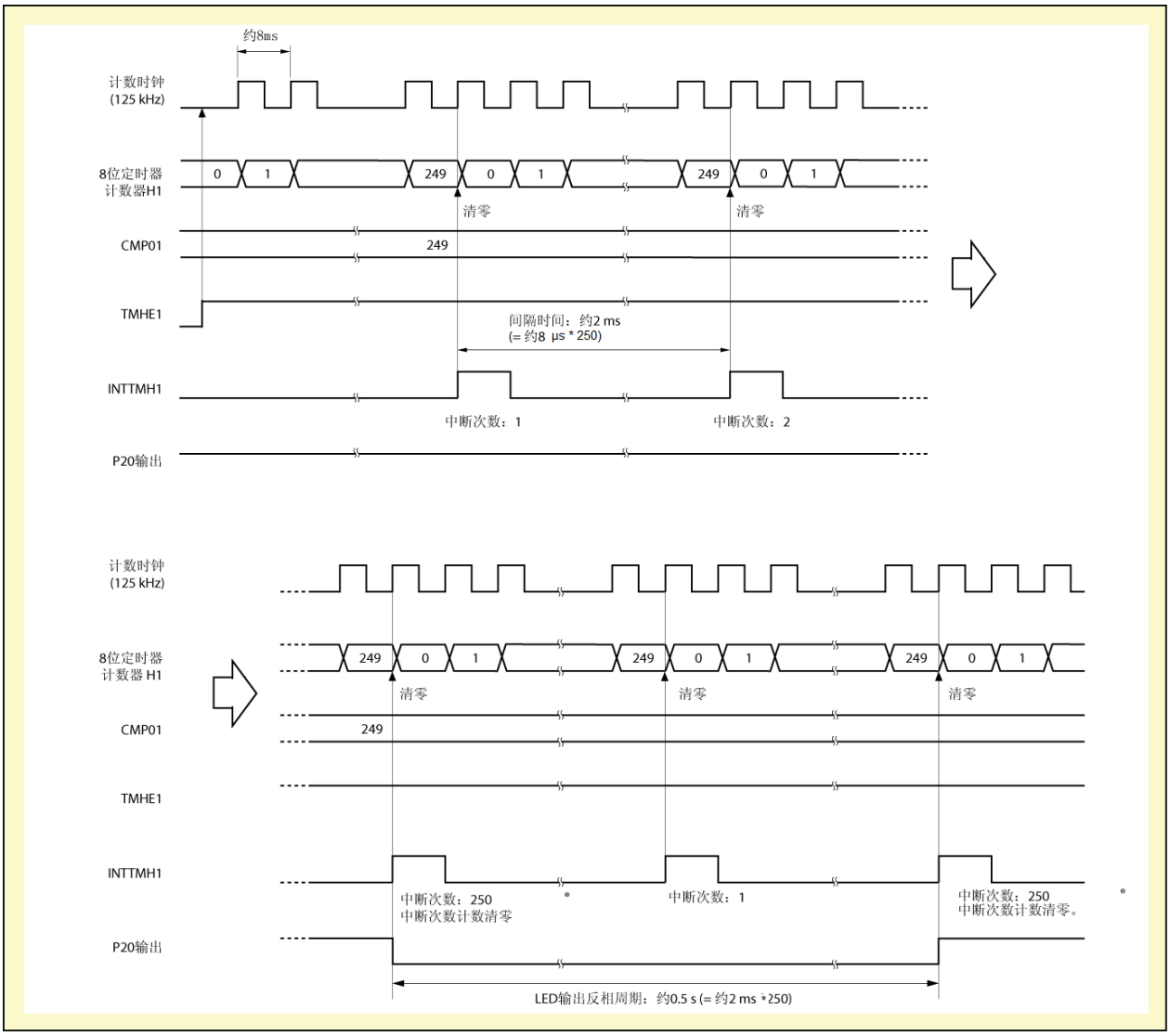
- 中断周期（间隔时间）=（N+1）/f_{CNT}=（249+1）/125kHz=2ms
- LED 输出反相周期=中断周期×中断次数=2ms×250=500ms
- LED 闪烁周期=LED 输出反相周期×2=500ms×2=1s

此外，CMP01 寄存器的设置值随开关输入的次数发生改变，LED 闪烁周期随之改变。

开关输入次数[注]	CMP01 寄存器设置值	中断周期	LED 闪烁周期
0	249	约 2ms （（249+1）/125kHz）	约 1s （约 2ms×250×2）
1	124	约 1ms （（124+1）/125kHz）	约 0.5s （约 1ms×250×2）
2	62	约 0.504ms （（62+1）/125kHz）	约 0.252s （约 504μs×250×2）
3	31	约 0.256ms （（31+1）/125kHz）	约 0.128s （约 256μs×250×2）

注 第四次开关输入后重复第零次开关输入的闪烁周期。

图 4-3 LED 闪烁周期的时间图示例（LED 闪烁周期约为 1s）



备注 当 CMP01 寄存器的设置值为 124、62、31 时，LED 的闪烁周期分别约为 1/2s、1/4s、1/8s。

（2）设置抖动检测时间

在本示例程序中，对 8 位定时器 H1 中断（INTTMH1）的生成进行计数以去除小于等于 10ms 的抖动，从而处理开关输入过程中的抖动（INTP1 中断生成）。

在用 INTTMH1 中断进行抖动检测时，即便在抖动检测过程中也可对 INTTMH1 中断进行连续计数。这样，可抑制由开关输入引起的 LED 闪烁周期的偏移。

•抖动检测时间（Tc）=T'+T×（M–1）

备注 T：INTTMH1 中断周期

T'： 从 INTP1 边缘检测开始到 INTP1 边缘检测之后生成的第一个 INTTMH1 为止（0<T'≤T）

M： Intp1 边缘检测后 INTTMH1 中断的次数

当设置为 T×（M–1）=10ms 时，

Tc=T'+10ms

0<T'≤T，因此，

10ms<Tc≤T+10ms

↓

抖动检测时间（Tc）>10ms

计算示例： 当中断周期（T）为 2ms（参见**（1）设置LED闪烁周期**中的计算示例）时，INTP1 边缘检测后的INTTMH1 中断次数（M）为 6

Tc =T'+T×（M–1）

=T'+2ms×（6–1）

=T'+10ms

0<T'≤2ms，因此，

10ms<Tc≤12ms

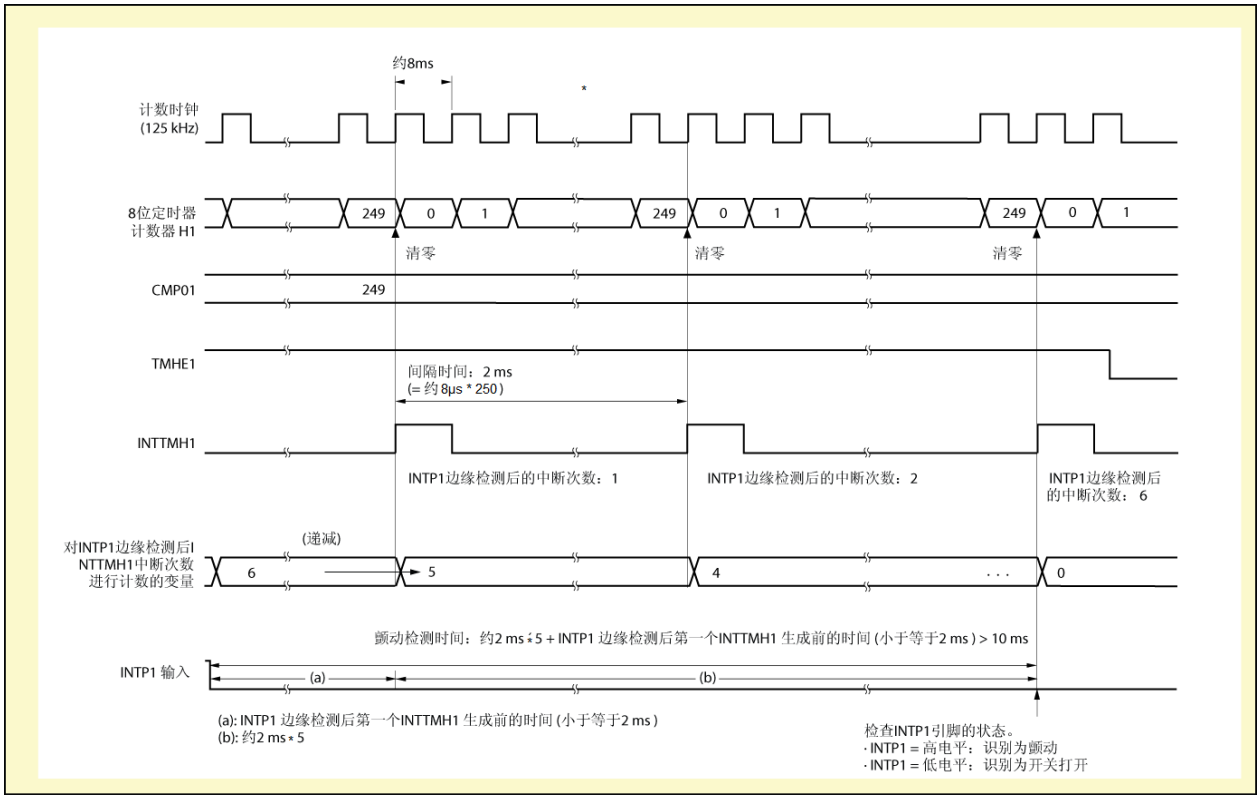
↓

抖动检测时间（Tc）>10ms

下表展示了在本示例程序中，开关输入过程中的中断周期与 INTP1 边缘检测后的 INTTMH1 中断次数之间的对应关系。

LED 闪烁周期	中断周期	Intp1 边缘检测后 INTTMH1 中断的次数	抖动检测时间
约 1s	约 2ms	6	10ms<Tc≤12ms
约 0.5s	约 1ms	11	10ms<Tc≤11ms
约 0.252s	约 0.504ms	21	10.08ms<Tc≤10.584ms
约 0.128s	约 0.256ms	41	10.24ms<Tc≤10.496ms

图 4-4　　抖动检测的时间图示例（开关输入过程中 LED 闪烁周期约为 1s）



备注　　用于对 INTP1 边缘检测之后的 INTTMH1 中断次数进行计数的变量由开关输入过程中的 LED 闪烁周期决定。当 LED 闪烁周期分别约为 1/2s、1/4s、1/8s 时，变量为 11、21、41。

第五章　用系统仿真器SM+进行运行检查

本章描述利用汇编语言文件（源文件+项目文件），示例程序在适用 78K0S/Kx1+的系统仿真器 SM+中如何运行；该汇编语言文件通过选择  进行下载。

注意事项　适用 78K0S/Kx1+的系统仿真器 SM+在 78K0S/KU1+微控制器中不支持（至 2007 年 7 月）。因此，78K0S/KU1+微控制器的运行无法由适用 78K0S/Kx1+的系统仿真器 SM+进行检查。

5.1　构建示例程序

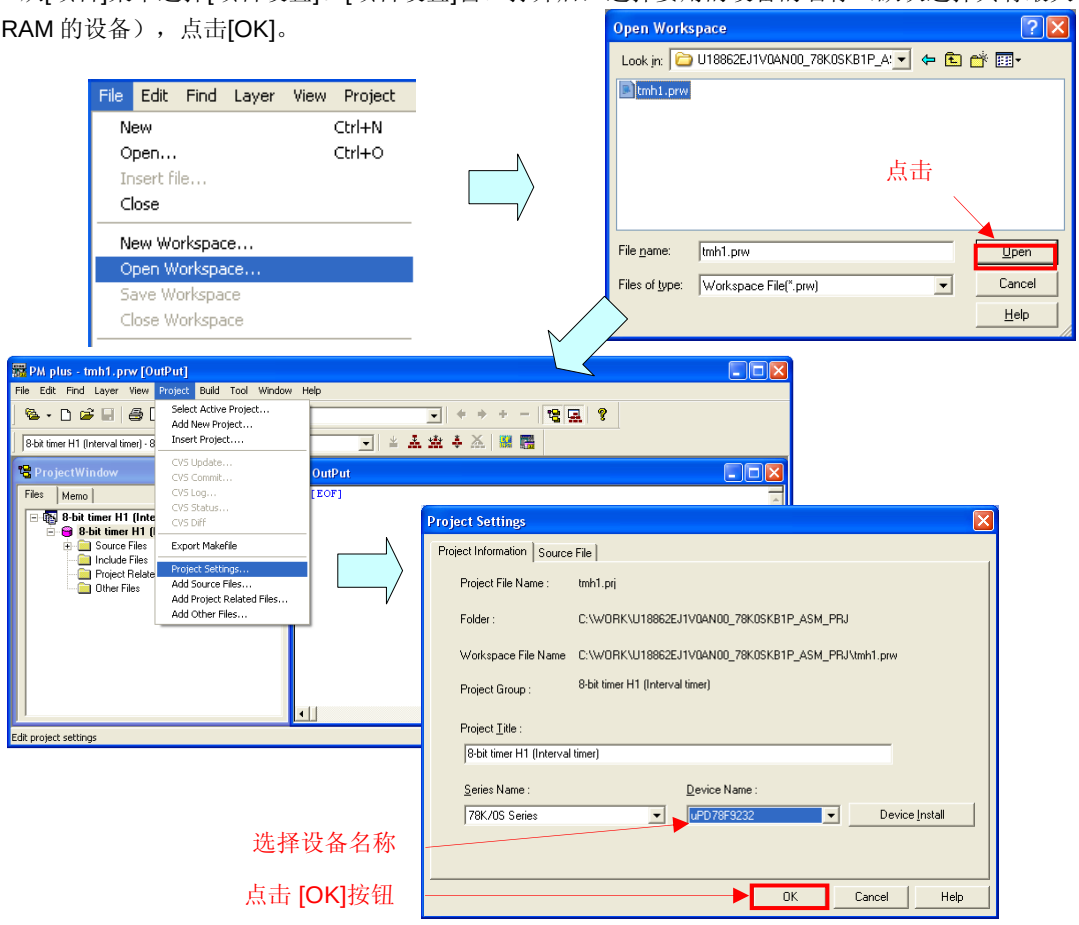
要利用适用 78K0S/Kx1+的系统仿真器SM+（下文称为“SM+”）检查示例程序的运行，在构建示例程序后必须启动SM+。本节描述运行次序的示例，从用集成开发环境PM+构建示例程序到启动SM+；使用  下载的汇编语言文件（源文件+项目文件）。关于如何构建其他下载的程序，请参见**78K0S/Kx1+示例程序启动向导的应用示例**的“**第三章注册集成开发环境PM+项目并执行构建**”。

关于如何操作PM+的详情，请参见**PM+项目管理器用户手册**。

（1）启动 PM+。

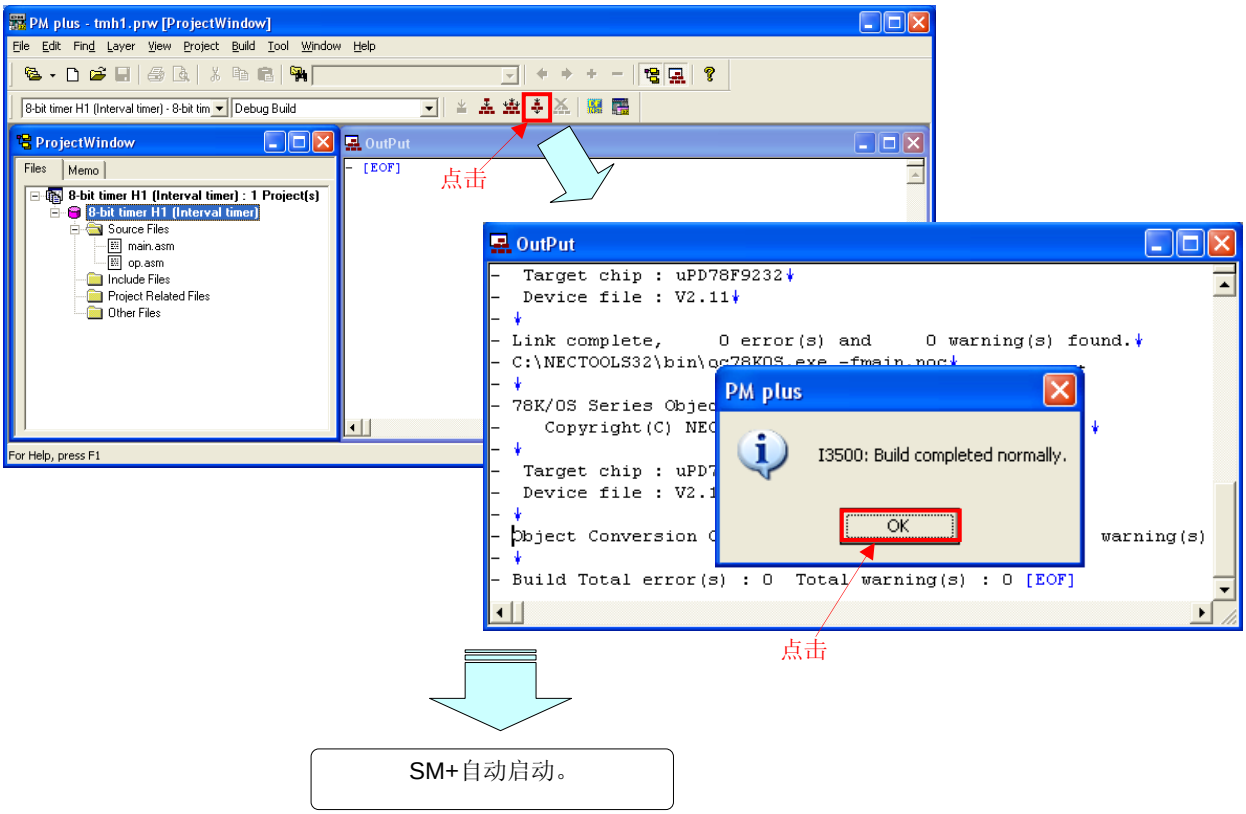
（2）在[文件]菜单点击[打开]再点击[打开工作区]，选择“tmh1.prw”。工作区生成，源文件将自动放入该工作区中。

（3）从[项目]菜单选择[项目设置]。[项目设置]窗口打开后，选择要用的设备的名称（默认选择具有最大 ROM 或 RAM 的设备），点击[OK]。



（4）点击  构建→调试]按钮）。当“main.asm”和“op.asm”源文件正常构建时，将显示信息“I3500：Build completed normally.（构建正常完成）”。

（5）点击消息窗口中的[OK]按钮，自动启动 SM+。



5.2 随SM+运行

本节描述在 SM+的 I/O 面板窗口或时间图窗口中对运行进行检查的示例。

关于如何操作SM+的详情，请参见SM+系统仿真器操作用户手册。

[列]构建错误

在用 PM+进行构建时，如果显示错误消息“A006 File not found ‘C:\NECTOOLS32\LIB78K0S\s0sl.rel’（文件未找到）”或“*** ERROR F206 Segment ‘@@DATA’ can’t allocate to memory - ignored.（片段‘@@DATA’无法放入存储器中——忽略）”，应按照下面的步骤改变编译器的选项设置。

<1> 从[工具]菜单选择[编译器选项]。

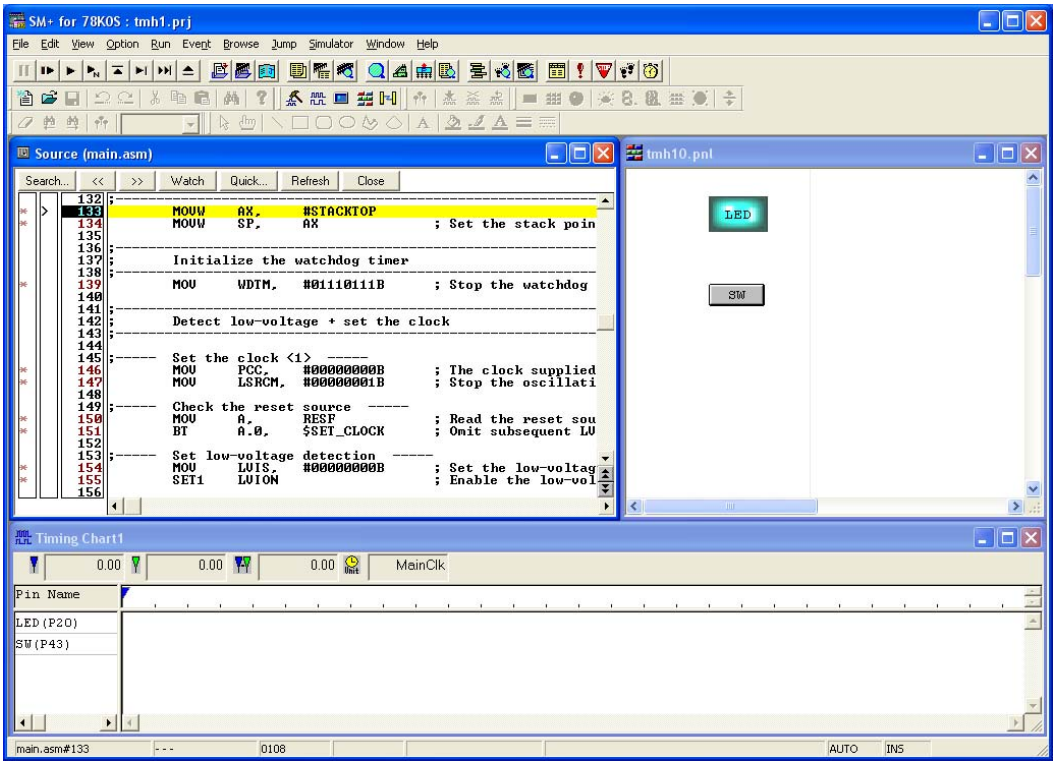
<2> 显示[编译器选项]对话框。 选择[启动例程]标签。

<3> 不选[使用标准库的固定区域]复选框。 （其他复选框不动。）

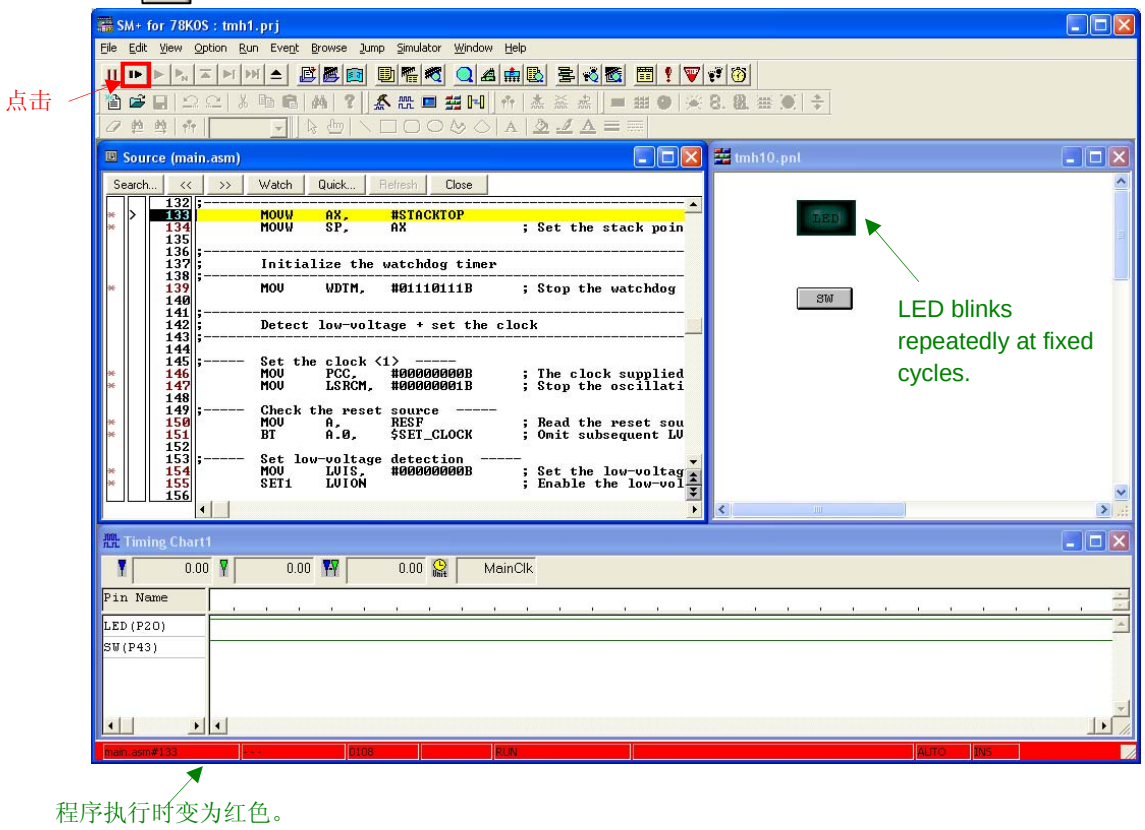
当[使用标准库的固定区域]复选框不选时，将保留 118 字节的 RAM 区作为固定标准库区供使用；但是，标准库（如 getchar 函数和 malloc 函数）会被禁用。

点击  图标下载文件并在本示例程序中进行使用时，默认不选中[使用标准库的固定区域]复选框。

（1）点击PM+中的[构建→调试]启动SM+（参见5.1），屏幕显示如下。（这是使用汇编语言源文件时的屏幕示例。）



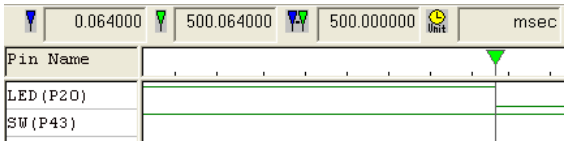
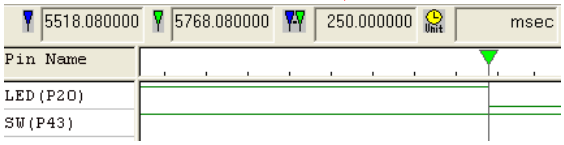
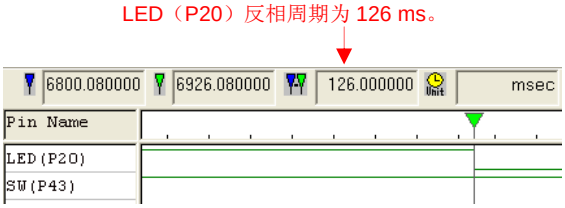
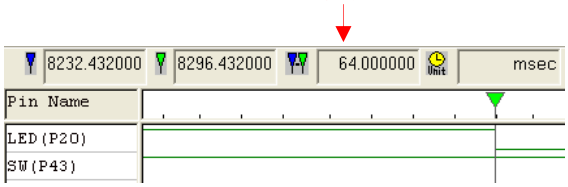
（2）点击  重新启动]按钮）。在 CPU 复位后，程序开始执行，屏幕显示如下。



（3）在程序执行过程中，在 I/O 面板窗口中点击[SW]按钮。

在 I/O 面板窗口中检查[LED]的闪烁周期，在时间图窗口中检查波形，确认二者随[SW]按钮的输入次数而改变。

注 1

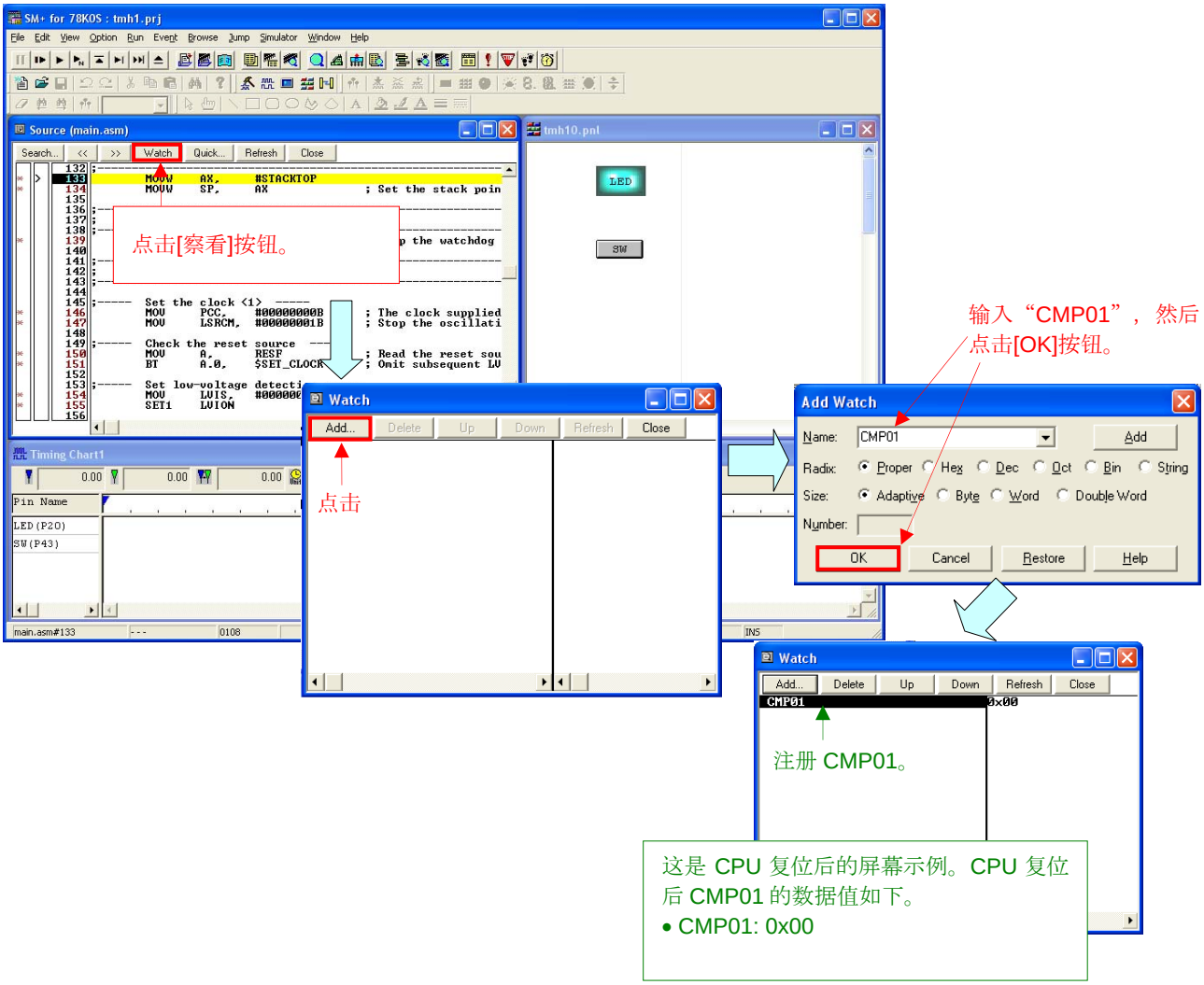
I/O面板窗口	时间图窗口
闪烁周期约为 1s 注 2。 LED 不点击。 SW	LED（P20）反相周期为 500ms。 
闪烁周期约为 1/2s 注 2。 LED 点击一次。 SW	LED（P20）反相周期为 250 ms。 
闪烁周期约为 1/4s 注 2。 LED 点击两次。 SW	LED（P20）反相周期为 126 ms。 
闪烁周期约为 1/8s 注 2。 LED 点击三次。 SW	LED（P20）反相周期为 64 ms。 

注 **1.** 第四次输入[SW]按钮后，重复第零次[SW]按钮输入时的闪烁周期。

2. 根据所用 PC 的操作环境，结果与实际闪烁周期会有差异。

[附 1]　　利用 SM+察看功能，可检查 CMP01 寄存器数据值的改变。

<1> 在“源文件”窗口中点击[察看]按钮可打开[察看]窗口。
<2> 点击[添加]打开[添加察看]窗口。（此时，[察看]窗口保持打开。）
<3> 在[察看]窗口中，在[名称]区输入“CMP01”并点击[OK]按钮对“CMP01”进行注册，再关闭[添加察看]窗口。



<4> 执行程序并在 I/O 面板窗口中点击[SW]按钮。检查[察看]窗口中的 CMP01 数据值随[SW]按钮的输入次数而改变。

[SW]按钮输入次数[注]	[察看]窗口的数据值
0	CMP01：0xF9（249）
1	CMP01：0x7C（124）
2	CMP01：0x3D（61）
3	CMP01：0x1E（30）

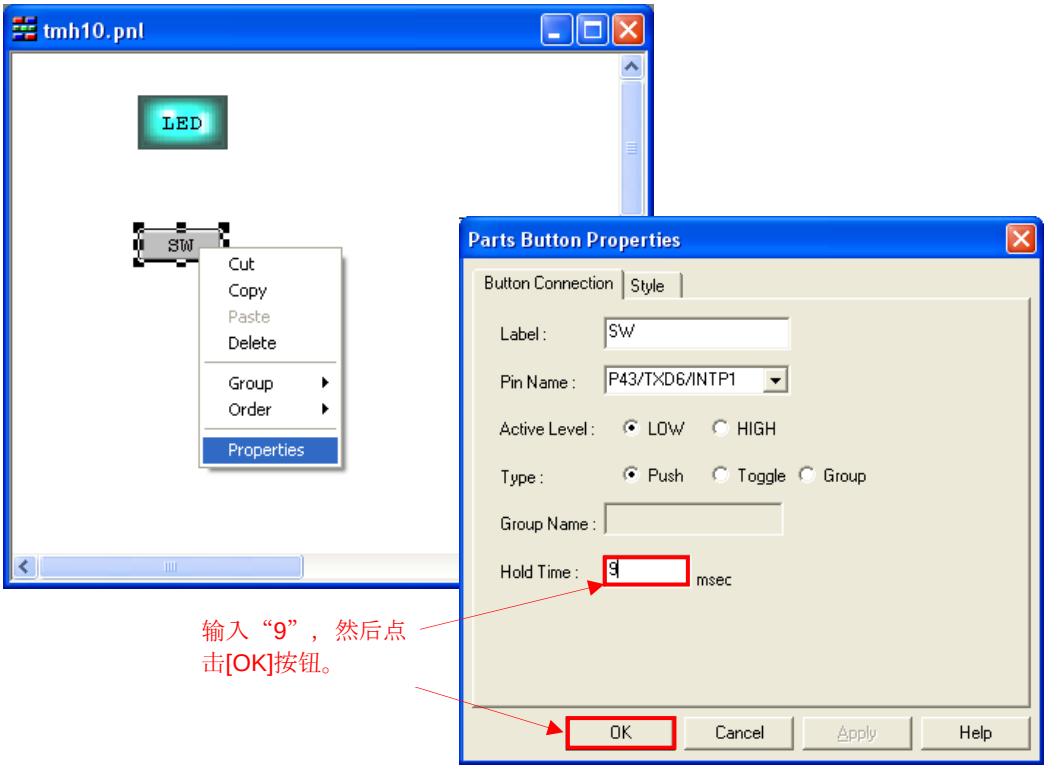
注　　第四次开关输入后重复第零次开关输入时的发光模式。

[附 2] [SW]按钮保持时间可设置为小于 10ms，从而可检查是否检测了抖动。

<1>。在工具栏上选择 

<2> 在 I/O 面板窗口中右键点击[SW]按钮，选择[属性]。

<3> 对于保持时间（Hold Time）输入“9”，点击[OK]按钮。



<4> 在工具栏上选择. 

<5> 执行程序，点击[SW]按钮。因为按钮保持时间为 9ms，所以即便点击了[SW]按钮，也会识别出抖动，故 LED 闪烁周期不会改变。

第六章 相关文件

文件名称		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+系统模拟器操作用户手册		PDF
78K0S/KA1+ 简化闪存写入手册 MINICUBE2 信息		PDF
78K0S/Kx1+ 应用注释	示例程序启动向导	PDF
	示例程序（初始设置）LED 照明开关控制	PDF
	示例程序（中断）开关输入引起的外部中断	PDF
	示例程序（低压检测）电压低于2.7V检测时复位发生。	PDF
	示例程序（8-位 定时器H1）PWM 输出	PDF

附件A 程序列表

作为程序列表示例，78K0S/KB1+微控制器的源程序如下所示。

● main.asm (汇编语言版)

```
;*******************************************************************************
;
;           日电电子     78K0S/KB1+
;
;*******************************************************************************
;           78K0S/KB1+       示例程序
;*******************************************************************************
;           8 位定时器 H1
;*******************************************************************************
;<<历史>>
;           2007.7.--            发布
;*******************************************************************************
;
;<<概述>>
;
; 本示例程序展示了使用 8 位定时器 H1 的间隔定时器功能的示例。通过使用 8 位定时器 H1 中断，P20
; 引脚输出反相，从而使得 LED 灯闪烁。改变 LED 闪烁周期的方法是，在生成开关输入中断时重写定时器
; 的比较寄存器。
;
;
; <主要设置内容>
;
; - 停止看门狗计时器的运行
; - 将低压检测电压（VLVI）设置为 4.3 V +-0.2 V
; - 在 VDD >= VLVI 后当 VDD < VLVI 时生成内部复位信号（低压检测器）
; - 设置 CPU 时钟为 8MHz
; - 设置供给外围硬件的时钟为 8MHz
; - 将外部中断 INTP1 的有效沿设置为下降沿
; - 将开关输入过程中的抖动检测时间设置为 10ms
;
;
; <8 位定时器 H1 的设置>
; - 设置为间隔定时器模式
; - 禁用 TOH1 引脚的定时器输出
; - 计数时钟 = fxp/2^6 (125 kHz)
; - 定时器周期的初始值 = 2 ms (8[us/clk] x 250[计数] = 2[ms])
;
;
; <开关输入次数及 LED 闪烁周期>
;
;   +---------------------------------+
;   |  SW 输入  | LED 闪烁     |
;   |   (P43)     |周期(P20)    |
;   |-------------|----------------|
;   |   0 次       |  1 秒          |
;   |   1 次       |  1/2 秒        |
;   |   2 次       | 1/4 秒         |
;   | 3 次         | 1/8 秒         |
;   +---------------------------------+
;   # 在第四次开关输入后，重复第零次开关输入时的闪烁周期。
;
```

```
;
;<<I/O 端口设置>>
;
; 输入：P43
; 输出：P00-P03, P20-P23, P30-P33, P40-P42, P44-P47, P120-P123, P130
; # 所有未使用的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      INTERRUPT_TMH1          ;(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
;
;========================================================================
XROM    CSEG    AT      0100H
;-----  设置定时器 H1 的周期  -----
        DB      250-1           ; 2ms 时间间隔比较值
        DB      125-1           ; 1ms 时间间隔比较值
        DB      63-1            ; 0.5ms 时间间隔比较值
        DB      32-1            ; 0.25ms 时间间隔比较值
;-----  处理抖动  -----
        DB      5+1             ; 处理抖动的计数值（2ms 时间间隔）
        DB      10+1            ; 处理抖动的计数值（1ms 时间间隔）
        DB      20+1            ; 处理抖动的计数值（0.5ms 时间间隔）
        DB      40+1            ; 处理抖动的计数值（0.25ms 时间间隔）

;========================================================================
;
;       定义 RAM
;
;========================================================================
XRAM    DSEG    SADDR
CNT_TMH1:       DS      1               ; 对 INTTMH1 中断进行计数

;========================================================================
;
;       定义存储器栈区
;
```

```
;===============================================================================
XSTK    DSEG    AT      0FEE0H
STACKEND:
        DS      20H                     ; 存储器栈区 = 32 字节
STACKTOP:                               ; 存储器栈区起始地址 = FF00H

;*******************************************************************************
;
;       Reset 后的初始化
;
;*******************************************************************************
XMAIN   CSEG    UNIT
RESET_START:
;-------------------------------------------------------------------------------
;       初始化栈指针
;-------------------------------------------------------------------------------
        MOVW    AX,     #STACKTOP
        MOVW    SP,     AX              ; 设置栈指针

;-------------------------------------------------------------------------------
;       初始化看门狗计时器
;-------------------------------------------------------------------------------
        MOV     WDTM,  #01110111B       ; 停止看门狗计时器的运行

;-------------------------------------------------------------------------------
;       检测低压+设置时钟
;-------------------------------------------------------------------------------

;-----  设置时钟<1>  -----
        MOV     PCC,    #00000000B      ; 供给 CPU 的时钟(fcpu)= fxp（= fx/4 = 2 MHz）
        MOV     LSRCM, #00000001B       ; 停止低速内部振荡器的振荡

;-----  检查复位源 -----
        MOV     A,      RESF            ; 读取复位源
        BT      A.0,    $SET_CLOCK      ; 在 LVI 复位时，省略后续 LVI 相关处理，转到 SET_CLOCK

;-----  设置低压检测  -----
        MOV     LVIS,   #00000000B      ; 将低压检查电压（VLVI）设置为 4.3 V +-0.2 V
        SET1    LVION                   ; 启用低压检测器的运行

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

;-----  VDD >= VLVI 等待处理  -----
WAIT_LVI:
        NOP
        BT      LVIF,   $WAIT_LVI       ; 如果 VDD < VLVI，则分支执行

        SET1    LVIMD                   ; 设置为当 VDD < VLVI 时生成内部复位信号

;-----  设置时钟<2>  -----
SET_CLOCK:
        MOV     PPCC,   #00000000B      ; 设置供给外围硬件的时钟(fxp) = fx (= 8 MHz)
                                        ;  -> 供给 CPU 的时钟(fcpu)= fxp = 8 MHz

;-------------------------------------------------------------------------------
;       初始化端口 0
;-------------------------------------------------------------------------------
```

```
        MOV     P0,     #00000000B      ; 将 P00-P03 的输出锁存器设置为低
        MOV     PM0,    #11110000B      ; 设置 P00-P03 为输出模式

;---------------------------------------------------------------------
;       初始化端口 2
;---------------------------------------------------------------------
        MOV     P2,     #00000001B      ; 设置 P21-P23 的输出锁存器为低，设置 P20 的输出锁存器为高（关闭
LED）
        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 的输出锁存器为低
        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 的输出锁存器为低
        MOV     PU4,    #00001000B      ; 将片上的上拉电阻连接到 P43
        MOV     PM4,    #00001000B      ; 设置 P40-P42、P44-P47 为输出模式，设置 P43 为输入模式

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

;---------------------------------------------------------------------
;       初始化端口 13
;---------------------------------------------------------------------
        MOV     P13,    #00000001B      ; 将 P130 的输出锁存器设置为高

;---------------------------------------------------------------------
;       初始化通用寄存器和 RAM
;---------------------------------------------------------------------
        MOV     CNT_TMH1, #250          ; 初始化 INTTMH1 中断次数
        MOVW    HL,     #0100H ；将表格地址指定给 HL（用于 INTP1 中断）

;---------------------------------------------------------------------
;       设置 8 位定时器 H1
;---------------------------------------------------------------------
        MOV     TMHMD1,         #00110000B      ; 计数时钟 = fxp/2^6 = 125 kHz, 设置为间隔定时器模式
        MOV     A,      [HL]            ; 从表中读取 LED 闪烁的基准时间初始值
        MOV     CMP01, A                ; 初始化比较值
        SET1    TMHE1                   ; 启动定时器的运行

;---------------------------------------------------------------------
;       设置中断
;---------------------------------------------------------------------
        MOV     INTM0,  #00000000B      ; 将 INTP1 的有效沿设置为下降沿
        MOV     IF0,    #00H            ; 事先清除无效中断请求
        CLR1    PMK1                    ; Intp1 中断去屏蔽
        CLR1    TMMKH1                  ; Inttmh1 中断去屏蔽

        EI                              ; 启用向量中断

;*************************************************************************
;
;       主循环
```

```
;
;*******************************************************************************
MAIN_LOOP:
        NOP
        BR      $MAIN_LOOP              ; 转到 MAIN_LOOP

;*******************************************************************************
;
;       外部中断 INTP1
;
;*******************************************************************************
INTERRUPT_P1:
        PUSH    AX                      ; 将 AX 寄存器数据保存到栈中

;----- 处理抖动的 10ms 等待 -----
        MOV     A,      [HL+4]  ; 读取对应于定时器 H1 周期的计数值
WAIT_CHAT:
        NOP
        BF      TMIFH1,$WAIT_CHAT       ; 等待 INTTMH1 中断
        CLR1    TMIFH1                  ; 清除 INTTMH1 中断请求标志
        CALL    !SUB_INTERRUPT_TMH1; 处理 INTTMH1 中断
        DEC     A                       ; A 寄存器减少 1
        BNZ     $WAIT_CHAT              ; 如果 A = 0 为非，就分支执行

        CLR1    PIF1                    ; 清除 INTP1 中断请求

;----- 寻找抖动检测 -----
        BT      P4.3,   $END_INTP1      ; 如果无开关输入，分支执行

;----- 改变 TMH1 时间间隔周期 -----
        CLR1    TMHE1                   ; 停止定时器的运行

        MOV     A,      L               ; 读取表格地址的低 8 位
        INC     A                       ; 表格地址增加 1
        AND     A,      #00000011B      ; 屏蔽除 0 位和 1 位以外的各位
        MOV     L,      A               ; 写入表格地址的低 8 位
        MOV     A,      [HL]            ; 读取表格数据
        MOV     CMP01, A                ; 改变 LED 闪烁的基准时间

        SET1    TMHE1                   ; 启动定时器的运行

        MOV     CNT_TMH1, #250          ; 初始化 INTTMH1 中断次数

END_INTP1:
        POP     AX                      ; 还原 AX 寄存器数据
        RETI                            ; 从中断服务返回

;*******************************************************************************
;
;       中断 INTTMH1
;
;*******************************************************************************
INTERRUPT_TMH1:
        CALL    !SUB_INTERRUPT_TMH1; 处理 INTTMH1 中断
        RETI                            ; 从中断服务返回

;-------------------------------------------------------------------------
;       用于测量 INTTMH1 中断次数的子例程
;-------------------------------------------------------------------------
SUB_INTERRUPT_TMH1:
        DBNZ    CNT_TMH1, $END_INTTMH1      ; 如果 INTTMH1 中断次数< 250，则分支执行
```

```
        MOV     CNT_TMH1, #250                  ; 初始化 INTTMH1 中断次数

        XOR     P2,       #00000001B    ; LED 输出反相
END_INTTMH1:
        RET                             ; 从子例程返回

end
```

● main.c (C 语言版)

```
*********************************************************************************

        日电电子 78K0S/KB1+

*********************************************************************************
        78K0S/KB1+      示例程序
*********************************************************************************
        8 位定时器 H1
*********************************************************************************
<<历史>>
        2007.7.--           发布
*********************************************************************************

<<概述>>

本示例程序展示了使用 8 位定时器 H1 的间隔定时器功能的示例。通过使用 8 位定时器 H1 中断，P20 引脚输出反相，
从而使得 LED 灯闪烁。 改变 LED 闪烁周期的方法是，在生成开关输入中断时重写定时器的比较寄存器。


 <主要设置内容>

 - 声明由中断运行的函数：INTP1 -> fn_intp1()
 - 声明由中断运行的函数：INTTMH1 -> fn_inttmH1()
 - 停止看门狗计时器的运行
 - 将低压检查电压（VLVI）设置为 4.3 V +-0.2 V
 - 在 VDD >= VLVI 后当 VDD < VLVI 时生成内部复位信号（低压检测器）
 - 设置 CPU 时钟为 8MHz
 - 设置供给外围硬件的时钟为 8MHz
 - 将外部中断 INTP1 的有效沿设置为下降沿
 - 将开关输入过程中的抖动检测时间设置为 10ms


 <8 位定时器 H1 的设置>
 - 设置为间隔定时器模式
 - 禁用 TOH1 引脚的定时器输出
 - 计数时钟 = fxp/2^6 (125 kHz)
 - 定时器周期的初始值 = 2 ms (8[us/clk] x 250[计数] = 2[ms])


 <开关输入次数及 LED 闪烁周期>

   +--------------------------------------+
 |      SW 输入    |  LED 闪烁   |
 |       (P43)       |周期（P20）|
 |---------------------|-----------------|
 |       0 次          |  1 秒           |
```

```
|    1 次 |  1/2 秒  |
|    2 次 |   1/4 秒 |
|    3 次 |   1/8 秒 |
+-------------------------+
# 在第四次开关输入后，重复第零次开关输入时的闪烁周期。


<<I/O 端口设置>>

 输入：P43
 输出：P00-P03, P20-P23, P30-P33, P40-P42, P44-P47, P120-P123, P130
 # 所有未使用的端口设置为输出模式。

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


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

        预处理指令 (#pragma)

============================================================================*/
#pragmaSFR                              /* 可在 C 源程序级描述 SFR 名称 */
#pragmaEI                               /* 可在 C 源程序级描述 EI 指令 */
#pragmaNOP                              /* 可在 C 源程序级描述 NOP 指令 */
#pragma interrupt INTP1 fn_intp1        /* 中断函数声明：INTP1 */
#pragma interrupt INTTMH1 fn_inttmH1    /* 中断函数声明:INTTMH1 */

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

        声明函数原型

============================================================================*/
void fn_subinttmH1();                   /* INTTMH1 中断子例程 */

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

        定义全局变量

============================================================================*/
sreg unsigned char g_ucSWcnt = 0;        /* 用于对开关输入次数进行计数的 8 位变量 */
sreg unsigned char g_ucTMH1cnt = 0;      /* 用于对 INTTMH1 中断次数进行计数的 8 位变量 */
const unsigned char g_ucChat[4] = {5+1,10+1,20+1,40+1};              /* 用于去除抖动的 8 位常量表 */
const unsigned char g_ucCMPdata[4] = {250-1,125-1,63-1,32-1}; /* 用于 LED 闪烁基准时间的 8 位常量表 */

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

        Reset 后的初始化
```

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

/*-------------------------------------------------------------------------
        初始化看门狗计时器 + 检测低压 + 设置时钟
-------------------------------------------------------------------------*/
        /* 初始化看门狗计时器 */
        WDTM  = 0b01110111;                 /* 停止看门狗计时器的运行 */

        /* 设置时钟<1> */
        PCC   = 0b00000000;                 /* 供给 CPU 的时钟(fcpu)= fxp（= fx/4 = 2 MHz） */
        LSRCM = 0b00000001;                 /* 停止低速内部振荡器的振荡 */

        /* 检查复位源 */
        if (!(RESF & 0b00000001)){          /* 在 LVI 复位时，省略后续 LVI 相关处理 */

                /* 设置低压检测 */
                LVIS  = 0b00000000;         /* 将低压检测电压（VLVI）设置为 4.3 V +-0.2 V */
                LVION = 1;                  /* 启用低压检测器的运行 */

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

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

        LVIMD = 1;                          /* 设置为当 VDD < VLVI 时生成内部复位信号 */
        }

        /* 设置时钟<2> */
        PPCC  = 0b00000000;                 /* 设置供给外围硬件的时钟(fxp) = fx (= 8 MHz)
                                               -> 供给 CPU 的时钟(fcpu)= fxp = 8 MHz */

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

/*-------------------------------------------------------------------------
        初始化端口 2
-------------------------------------------------------------------------*/
        P2    = 0b00000001;                 /* 设置 P21-P23 的输出锁存器为低，设置 P20 的输出锁存器为高（关闭
LED） */
        PM2   = 0b11110000;                 /* 设置 P20-P23 为输出模式 */
```

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

/*---------------------------------------------------------------------------
        初始化端口 4
---------------------------------------------------------------------------*/
        P4    = 0b00000000;             /* 设置 P40-P47 的输出锁存器为低 */
        PU4   = 0b00001000;             /* 将片上的上拉电阻连接到 P43 */
        PM4   = 0b00001000;             /* 设置 P40-P42、P44-P47 为输出模式，设置 P43 为输入模式 */

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

/*---------------------------------------------------------------------------
        初始化端口 13
---------------------------------------------------------------------------*/
        P13   = 0b00000001;             /* 将 P130 的输出锁存器设置为高 */

/*---------------------------------------------------------------------------
        设置 8 位定时器 H1
---------------------------------------------------------------------------*/
        TMHMD1 = 0b00110000;            /* 计数时钟 = fxp/2^6 = 125 kHz, 设置为间隔定时器模式 */
        CMP01 = 250-1;                  /* 初始化 LED 闪烁的基准时间 */
        TMHE1 = 1;                      /* 启动定时器的运行 */

/*---------------------------------------------------------------------------
        设置中断
---------------------------------------------------------------------------*/
        INTM0 = 0b00000000;             /* 将 INTP1 的有效沿设置为下降沿 */
        IF0   = 0x00;                   /* 事先清除无效中断请求 */
        PMK1  = 0;                      /* Intp1 中断去屏蔽 */
        TMMKH1 = 0;                     /* Inttmh1 中断去屏蔽 */

        return;
}

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

        主循环

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

```
void main(void){

        EI();                                   /* 启用向量中断 */

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

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

        外部中断 INTP1

************************************************************************/
__interrupt void fn_intp1(){
        unsigned char ucChat;           /* 用于去除抖动的 8 位变量 */

        for (ucChat = g_ucChat[g_ucSWcnt] ; ucChat > 0 ; ucChat--){   /* 等待大约 10ms（用于去除抖动） */
                while (!TMIFH1){  /* 等待 INTTMH1 中断请求 */
                        NOP();
                }

                TMIFH1 = 0;             /* 清除 INTTMH1 中断请求标志 */
                fn_subinttmH1();  /* 处理 INTTMH1 中断 */
        }

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

        if (!P4.3){                     /* 如果 SW 打开 10ms 或以上则进行处理 */
                g_ucSWcnt = (g_ucSWcnt + 1) & 0b00000011;       /* 开关输入次数增加 1 */

                TMHE1 = 0;                      /* 停止定时器的运行 */
                CMP01 = g_ucCMPdata[g_ucSWcnt];         /* 根据开关输入次数改变 LED 闪烁的基准时间 */
                TMHE1 = 1;                      /* 启动定时器的运行 */

                g_ucTMH1cnt = 0;                /* 清除 INTTMH1 中断次数 */
        }

        return;
}

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

        中断 INTTMH1

************************************************************************/
__interrupt void fn_inttmH1(){
```

```
        fn_subinttmH1();            /* 处理 INTTMH1 中断 */

        return;
}

/*---------------------------------------------------------------------------
        用于测量 INTTMH1 中断次数的子例程
---------------------------------------------------------------------------*/
void fn_subinttmH1(){

        if (++g_ucTMH1cnt == 250){      /* 当 INTTMH1 中断次数为 250 进行处理 */
                g_ucTMH1cnt = 0;        /* 清除 INTTMH1 中断次数 */
                P2 ^= 0b00000001;       /* LED 输出反相 */
        }

        return;
}
```

● op.asm（汇编语言和 C 语言版共用）

```
;=========================================================================
;
;          选项字节
;
;=========================================================================
OPBT    CSEG    AT          0080H
        DB      10011100B       ; 选项字节区
;                ||||
;                |||+----------- 低速内部振荡器可用软件停止
;                |++------------ 高速内部时钟(8 MHz)选择为系统时钟源
;                +-------------- P34/RESET 用作 RESET 引脚

        DB      11111111B       ; 保护字节区（用于自编程模式）
;               ||||||||
;               ++++++++----------- 所有模块均可写入或删除

end
```

附件B 修订记录

版本	出版日期	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