应用注释

78K0S/Kx1+

示例程序（8-位定时器 H1）

PWM 输出

本文件内容包含示例程序操作概述，使用方法及怎样设置使用 8-位定时器 H1PWM 输出功能。在示例程序中，使用 8-位定时器 H1 的 PWM 输出功能控制脉冲输出负载， LEDs 的亮度每 500ms 改变。

目标设备

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

文件号： U18863CA1V0AN00（第一版）
出版日期： 2008 年 03 月 N

中文版

目录

· 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

本示例提供了一个例子，说明 8-位定时器 H1 的 PWM 输出功能的使用。通过控制脉冲输出负载每 500 ms 改变 LEDs 亮度。

1.1 初始设置的主要内容

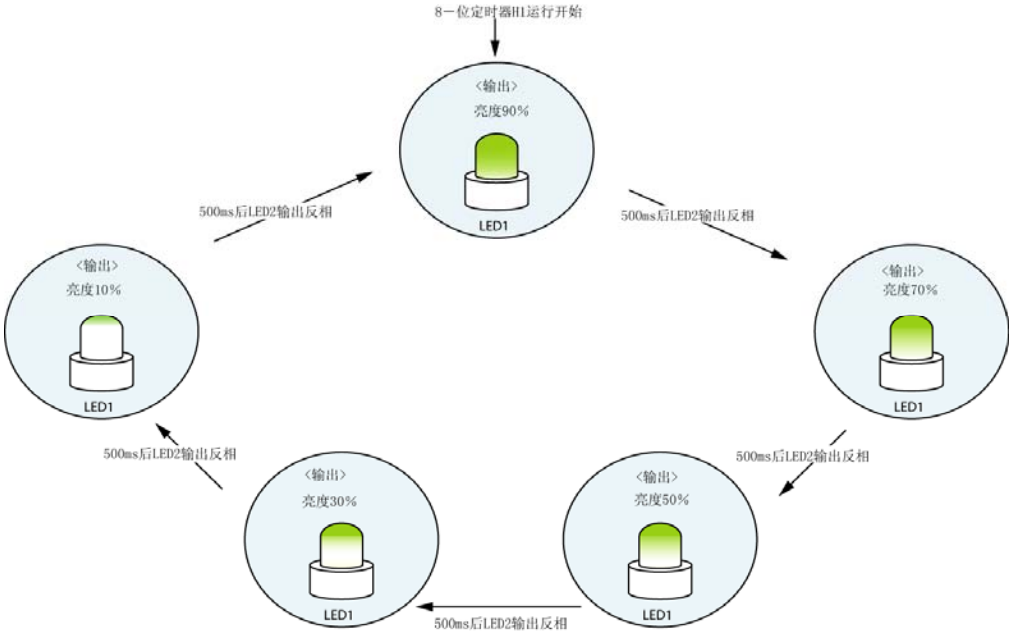
初始设置的主要内容如下。

- 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
- 设置 VLVI (低压检测器电压)为 4.3 V ±0.2 V
- 当 VDD (供电电压)大于或等于 VLVI 后，一旦检测到 VDD 小于 VLVI 就会生成内部复位信号（LVI 复位）
- 设置 CPU 时钟频率为 8 MHz
- 设置 I/O 端口
- 设置 8-为计时器 H1
  - 设置计数时钟为 fXP/2^6 (125 kHz)，设置工作模式为 PWM 输出模式，从 TOH1 启用计时器输出， 并将输出电压（默认）设置为低压
  - 设置 PWM 脉冲输出周期为 2 ms (8 μs × 250) 设置负载为 10%
- 启用 INTTMH1 中断

注 用选项字节进行设置。

1.2 主循环后的内容

在初始设置完成后，通过控制 8-位计时器 H1 的 PWM 输出负载可以改变 LED1 的亮度。利用 8-位计时器 H1 中断 (INTTMH1)，每 500 ms 可以改变负载。当负载改变时，LED2 输出反相。



PWM 输出负载	10%	30%	50%	70%	90%
LED1 亮度	90%	70%	50%	30%	10%

在本示例程序中，“LED1 亮度 = 100 –负载”，因为 PWM 输出活动电平设置为高电平且当其为低电平时 LED1 发光。

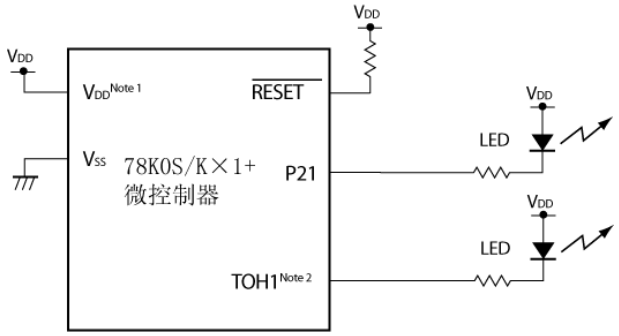
注意事项 关于使用设备的注意事项，请参见各产品(78K0S/KU1+, 78K0S/KY1+, 78K0S/KA1+, 78K0S/KB1+)的用户手册。

第二章 电路图

本章描述在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：



注 **1.** 使用该电路的电压范围为 4.5 V ≤ VDD ≤ 5.5 V。

2. TOH1/P42：78K0S/KA1+ 和 78K0S/KB1+微处理器

TOH1/ANI0/TI000/P20：78K0S/KY1+ 和 78K0S/KU1+ 微处理器

注意事项 1. 将 AVREF 引脚直接连接到 VDD （仅适用于 78K0S/KA1+和 78K0S/KB1+ 微控制器）。

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

3. 所有未使用的引脚保留开路 （不连接），除电路图中的引脚及 AVREF 和 AVSS 引脚外。

2.2 外围硬件

所用外围硬件如下。

- LED1：PWM 输出
- LED2：输出反相与 PWM 输出负载同时改变 （每 500 ms 反相）

第三章 软件

本章描述所下载压缩文件的配置、所用微控制器内部外围功能以及示例程序的初始化设置和操作概览，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	包含的压缩（*.zip）文件		
				
main.asm （汇编语言版） --- main.c （C语言版）	用于硬件初始化处理及微处理器主处理的源文件。	●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	●	●	
tmh1pwm.prw	用于集成开发环境PM+的工作区		●	
tmh1pwm.prj	用于集成开发环境PM+的项目文件		●	
tmh1pwm.pri tmh1pwm.prs tmh1pwm.prm	用于78K0S/Kx1+系统模拟器SM+的项目文件		●注2	
tmh1pwm0.pnl	用于78K0S/Kx1+系统模拟器SM+（用于检查外围硬件运行）的I/O 面板文件		●注2	●
tmh1pwm0.wvo	用于78K0S/Kx1+系统模拟器SM+的计时图文件（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. 78K0S/KU1+ 微处理器中不包含这些文件。

备注  ： 仅包含源文件。

 ： 包含了与集成开发环境 PM+ 和 78K0S/Kx1+ 系统模拟器 SM+ 一起使用的文件。
： 包含了与 78K0S/Kx1+系统模拟器 SM+一起使用的微处理器操作模拟软件。

3.2 所用内部外围功能

示例程序使用微处理器的下列内部外围功能。

- $V_{DD} < V_{LVI}$ 检测： 低压监测器(LVI)
- PWM 输出功能： 8-位计时器 H1
- PWM 输出端口 （LED1）： TOH1[注]
- 输出端口（LED2）： P21

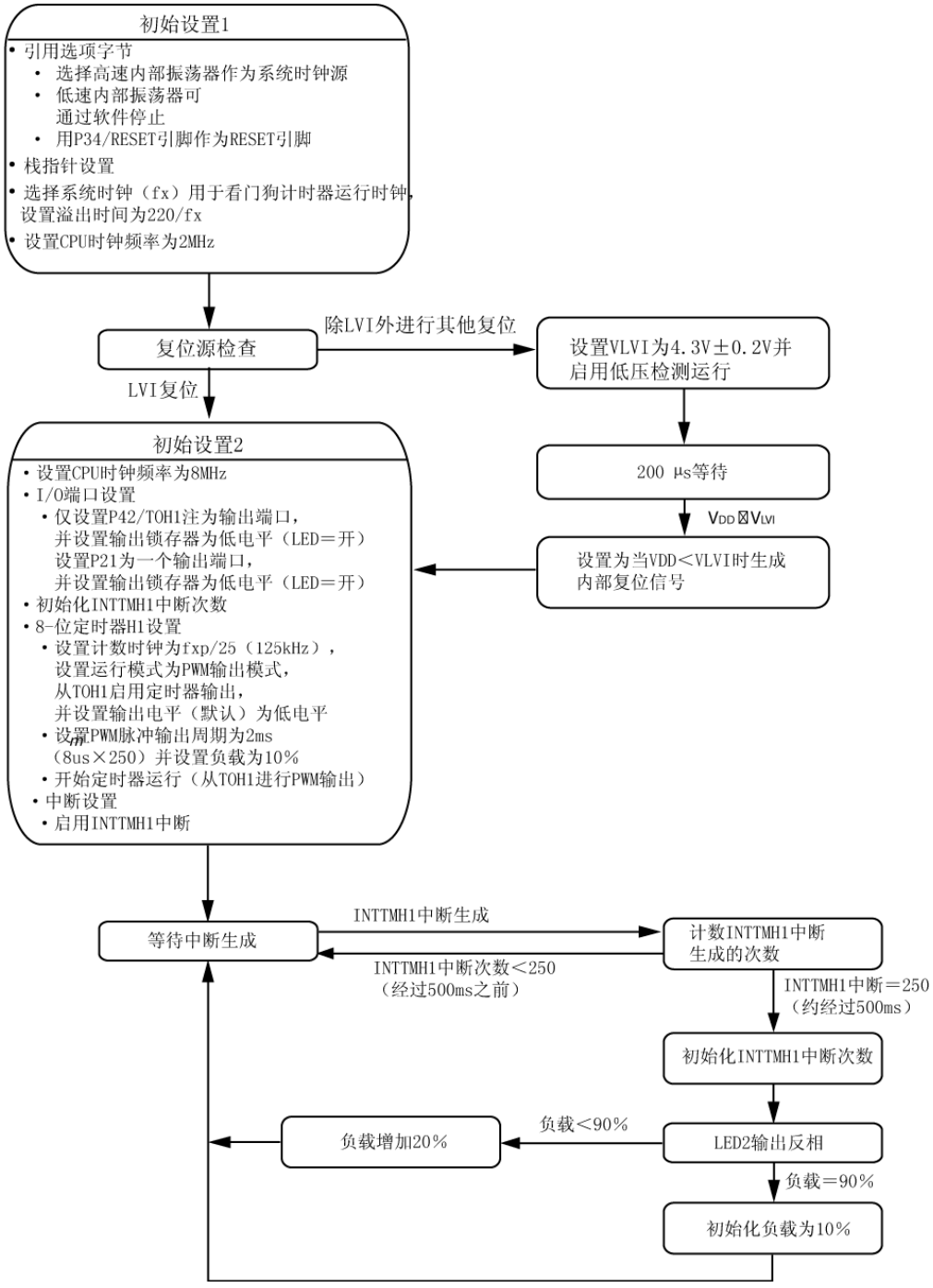
注 TOH1/P42： 78K0S/KA1+和 78K0S/KB1+微处理器
TOH1/ANI0/TI000/P20： 78K0S/KY1+ 和 78K0S/KU1+ 微处理器

3.3 初始设置及操作概览

在本示例程序进行的初始设置中包括了进行低设置检测功能、选择时钟频率、设置 I/O 端口、设置 8-位计时器 H1（PWM 输出）和设置中断。

在初始设置完成后，通过控制 8-位计时器 H1 的 PWM 输出负载改变 LED1 的亮度。利用 8-位计时器 H1 中断（INTTMH1），每 500 ms 改变负载。当负载改变时，LED2 输出反相。

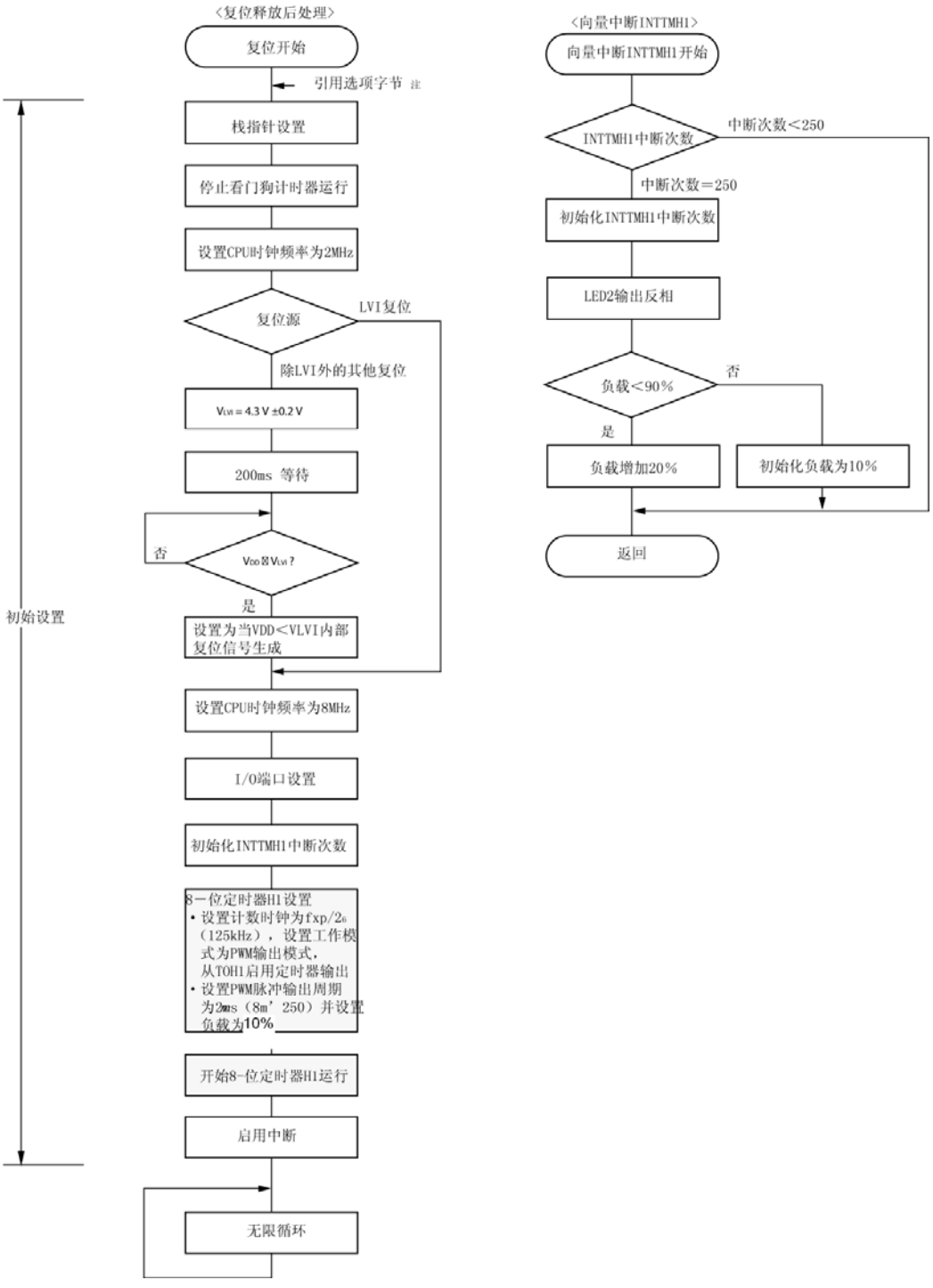
详情在如下所示的状态转变图中描述。



注 TOH1/P42： 78K0S/KA1+ 和 78K0S/KB1+ 微处理器
TOH1/ANI0/TI000/P20: 78K0S/KY1+ 和 78K0S/KU1+ 微处理器

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处理器在复位解除后自动进行。在本示例程序中，通过引用选项字节对下面的内容进行设置。

- 用高速内部振荡器时钟（8MHz（典型））做系统时钟源
- 利用软件可停止低速内部振荡器
- 用 P34/$\overline{\text{RESET}}$ 引脚做 $\overline{\text{RESET}}$ 引脚

第四章 设置方法

本章描述 8-位定时器 H1 的 PWM 输出功能。

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 8-位定时器H1 的PWM输出功能设置

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 使用 PWM 输出模式的 8-位定时器 H1，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 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) 设置 PWM 脉冲输出周期和负载

8-位定时器 H 比较寄存器 01 （CMP01）用于设置 PWM 脉冲输出周期，8-位定时器 H 比较寄存器 11（CMP11）用于设置负载。

- PWM 脉冲输出周期= (N + 1)/f_{CNT}
- 负载 = (M + 1)/(N + 1)

备注 N: CMP01 设置值
M: CMP11 设置值
f_{CNT}: 8-位定时器 H1 计时始终频率

注意事项 CMP01 寄存器设置值（N）和 CMP11 寄存器设置值（M）必须在下列范围内。
00H ≤ CMP11 (M) < CMP01 (N) ≤ FFH

图 4-2 8 位定时器 H 比较寄存器 01（CMP01）的格式

CMP01

注意事项 禁止在计时器计数操作过程中重写 CMP01 寄存器的值。

图 4-3. 8-位定时器 H 比较寄存器 11（CMP11）的格式

CMP11

注意事项 1. CMP11 寄存器的值可在计时器计数操作过程中重写。这用 3 个或更多运行时钟（TMHMD1 寄存器的 CKS12 到 CKS10 位所选择的信号），直至更改的 CMP11 寄存器的值转存到寄存器中。
2. 在计时器计数运行设置为停止时（TMHE1 = 0），CMP11 寄存器必须设置为启动计时器计数操作（TMHE1 = 1）。（即使将值设置为与 CMP11 寄存器相同，该设置必须重新设置。）
3. CMP11 值在计时器操作时重写，重写后的比较值在重写前计数值与比较值匹配定时时有效。如计数值和比较值匹配时定时和从 CPU 写入 CMP11 冲突，写入后的比较值在下一次计数值与写入前的比较值匹配时有效。

(3) TOH1 引脚设置

使用 PWM 输出模式 8-位计时器，按如下方式设置端口寄存器 x (Px)，端口模式寄存器 x (PMx)和端口模式控制寄存器 x (PMCx)。

	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

[示例 1]

- 设置 8-位定时器 H1 的工作模式为 PWM 输出模式，设置计数始终为 $f_{XP}/2^6$ （f_{XP} = 8 MHz），启用计时器输出（TOH1），设置输出电压（默认）为低压。
- 设置 PWM 脉冲输出周期为 2 ms，设置负载为 10%，启用计时器运行（与示例程序内容相同）

TMHMD1

TMHE1	0	1	1	1	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	低压

定时器工作模式

1	0	PWM 输出模式

计数始终选择

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} = 8 MHz/2^6 = 0.125 MHz = 125 kHz
- PWM 脉冲输出周期 2 ms = (N + 1)/125 kHz

→ N = 2 ms × 125 kHz – 1 = 249

CMP11 设置值（M）：24

- 0.1 (= 负载 10%) = (M + 1)/(249 + 1)

→ M = 0.1 × 250 – 1 = 24

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“00111001”给 TMHMD1、设置“249”给CMP01、设置“24”给CMP1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，设置“0”给 P20、设置“0”给 PM20、设置“0”给 PMC20、设置“00111001”给TMHMD1、设置“249”给CMP01、设置“24”给CMP11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,  #00111001B
MOV    CMP01,   #249
MOV    CMP11,   #24
SET1   TMHE1
```

• C 语言（使用 78K0S/KA1+ 和 78K0S/KB1+ 微控制器）

```
P4.2 = 0;
PM4.2 = 0;
TMHMD1 = 0b00111001;
CMP01 = 249;
CMP11 = 24;
TMHE1 = 1;
```

[示例 2]

- 设置 8-位定时器 H1 工作模式为 PWM 输出模式，设置计数始终为 f_{XP}（f_{XP} = 8 MHz），启用定时器输出（TOH1），设置定时器输出电压（默认）为高电平。
- 设置 PWM 脉冲输出周期为 31.25 μs，设置负载为 50%，启用定时器运行。

TMHMD1

TMHE1	0	0	0	1	0	1	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)

1	高电平

定时器工作模式

1	0	PWM 输出模式

计数始终选择

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} = 8 MHz
- PWM 脉冲输出周期 31.25 μs = (N + 1)/8 MHz

→ N = 31.25 μs × 8 MHz – 1 = 249

CMP11 设置值（M）：124

- 0.5 (= 负载 50%) = (M + 1)/(249 + 1)

→ M = 0.5 × 250 – 1 = 124

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”为 P42、设置“0”为 PM42、设置“00001011”为 TMHMD1、设置“249”给 CMP01、设置“124”给 CMP11 再设置 1 给 TMHE1，即可启动定时器的运行。

用于 78K0S/KY1+和 78K0S/KU1+微控制，先设置“0”给 P20、设置“0”给 PM20、设置“0”给 PMC20、设置“00001011”给 TMHMD1、设置“249”给 CMP01、设置“124”给 CMP11 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, #00001011B
MOV    CMP01,  #249
MOV    CMP11,  #124
SET1   TMHE1
```

• C 语言（使用 78K0S/KA1+和 78K0S/KB1+微控制器）

```
P4.2 = 0;
PM4.2 = 0;
TMHMD1 = 0b00001011;
CMP01 = 249;
CMP11 = 124;
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    P4,    #00000000B          ; 将P40-P47 的输出锁存器设置为低（P42:打开LED1）
       MOV    PM4,   #00000000B          ; 设置P40-P47 为输出模式
                    •
                    •
                    •
       MOV    TMHMD1,       #00111001B   ; 计数时钟 = fxp/2^6 = 1
                                         ; PWM 模式 启用 TOH1 输出
       MOV    CMP01, #250-1              ; 初始化CMP01（周期：2 ms）
       MOV    CMP11, #25-1               ; 初始化CMP11（负载：10%）
       SET1   TMHE1                      ; 启动定时器运行

       MOV    IF0,   #0                  ; 事先清除无效中断请求
       CLR1   TMMKH1                     ; 将INTTMH1 中断去屏蔽
       EI                                ; 启用向量中断

_LOOP:
       NOP
       BR     $MAIN_                     ; 转到 MAIN_LOOP
                    •
                    •
                    •
INTERRUPT_TMH1:
       DBNZ   CNT_TMH1, $END_IN          ; INTTMH1 中断次数< 250，则分支执行

       MOV    CNT_TMH1, #250             ; 初始化中断次数
       XOR    P2,    #00000010B          ; LED2 反相

       CMP    CMP11, #225-1              ; 与负载为 90%相比
       BC     $INC_CMP11                 ; 如负载< 90%，则分支执行

       MOV    CMP11, #25-1               ; 初始化负载为 10%
       BR     $END_INTTMH1               ; 分支执行为 END_INTTMH1

INC_CMP11:
       ADD    CMP11, #50                 ;负载增加 20%
                    •
                    •
                    •
```

设置 P42 和 PM42 输出锁存器为 0

设置 PWM 脉冲输出周期

设置 8-位定时器 H1 运行模式，计数时钟，TOH1 输出，输出电平（默认）

启用定时器运行

设置负载

将 INTTMH1 中断请求标清零

启用 INTTMH1 中断服务

利用 INTTMH1 中断生成启用中断服务

在 8-位定时器 H1 运行时负载可改变

- 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 200us 等待的 8-位变量*/
                     •
                     •
                     •
       P4     = 0b00000000;       /* 将P40-P47 输出锁存器设置为低（P42：打开LED1）*/
       PM4    = 0b00000000;       /* 设置P40-P47 为输出模式*/
                     •
                     •
                     •
       MHMD1 = 0b00111001;        /* 计数时钟 = fxp/2^6 = 125 kHz, PWM 模式，  */
                                  /*启用 TOH1 输出*/
       CMP01 = 250-1;             /* 初始化CMP01 （周期：2 ms）*/
       CMP11 = 25-1;              /* 初始化CMP11（负载：10%）*/
       TMHE1 = 1;                 /* 启用计时器运行*/

       IF0    = 0x0;              /* 事先清除无效中断请求*/
       TMMKH1 = 0;                /* 将INTTMH1 中断去屏蔽*/

       return;
}

void main(void){

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

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

__interrupt void fn_inttmH1(){
                     •
                     •
                     •
              if (CMP11 >= 225-1){/* 当负载最小为 90%时处理*/
                     CMP11 = 25-1;/* 初始化负载为 10% */
              }
              else {
                     CMP11 += 50; /* 将负载增加 20% */
                     •
                     •
                     •
```

设置 P42 和 PM42 输出锁存器为 0

设置 8-位定时器 H1 工作模式、计数时钟、TOH1 输出和输出电平（默认）

设置 PWM 脉冲输出周期

启用计时器运行

清除 INTTMH1 中断请求标志

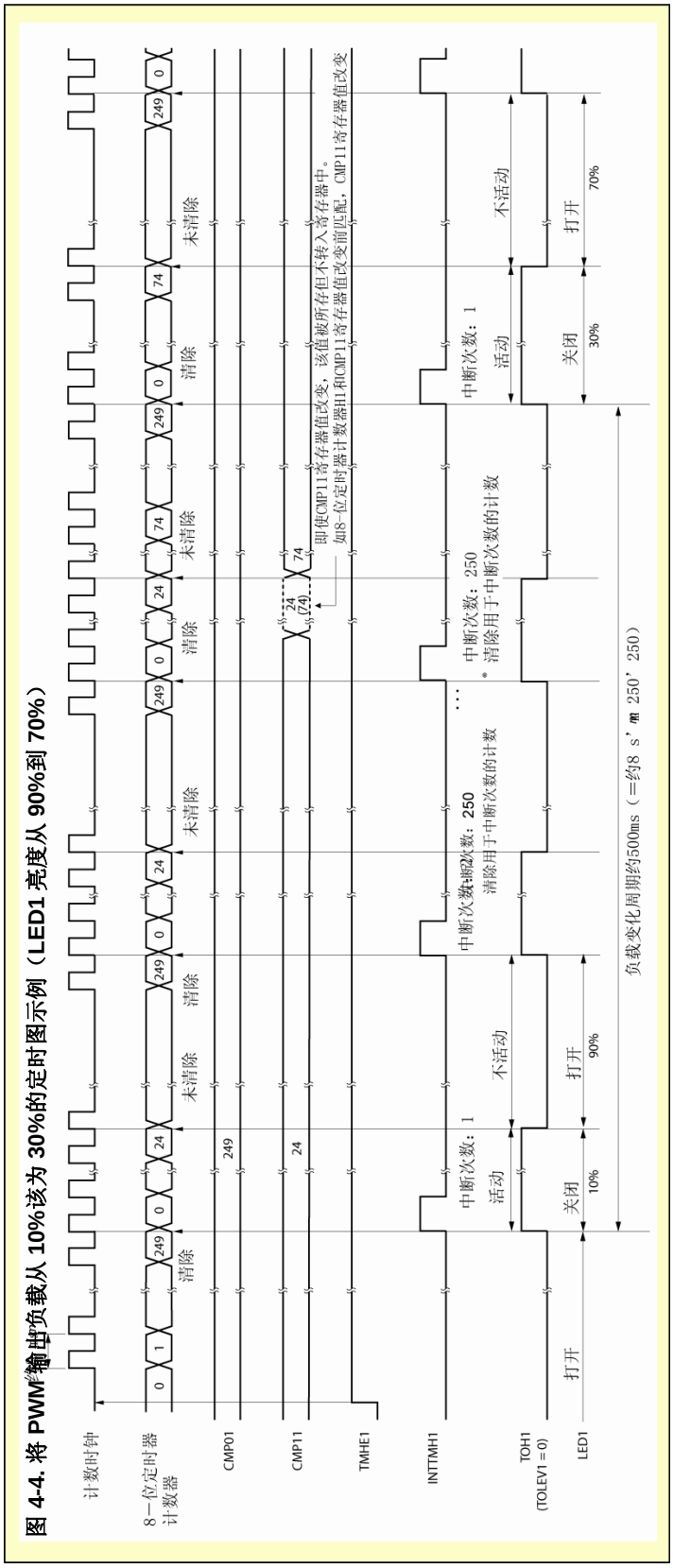
设置负载

启用 INTTMH1 中断服务

利用 INTTMH1 中断生成启用中断服务

在 8-位定时器 H1 运行可改变负载

图 4-4. 将 PWM 输出负载从 10%该为 30%的定时图示例（LED1 亮度从 90%到 70%）



第五章 用系统模拟器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 图标进行下载。

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

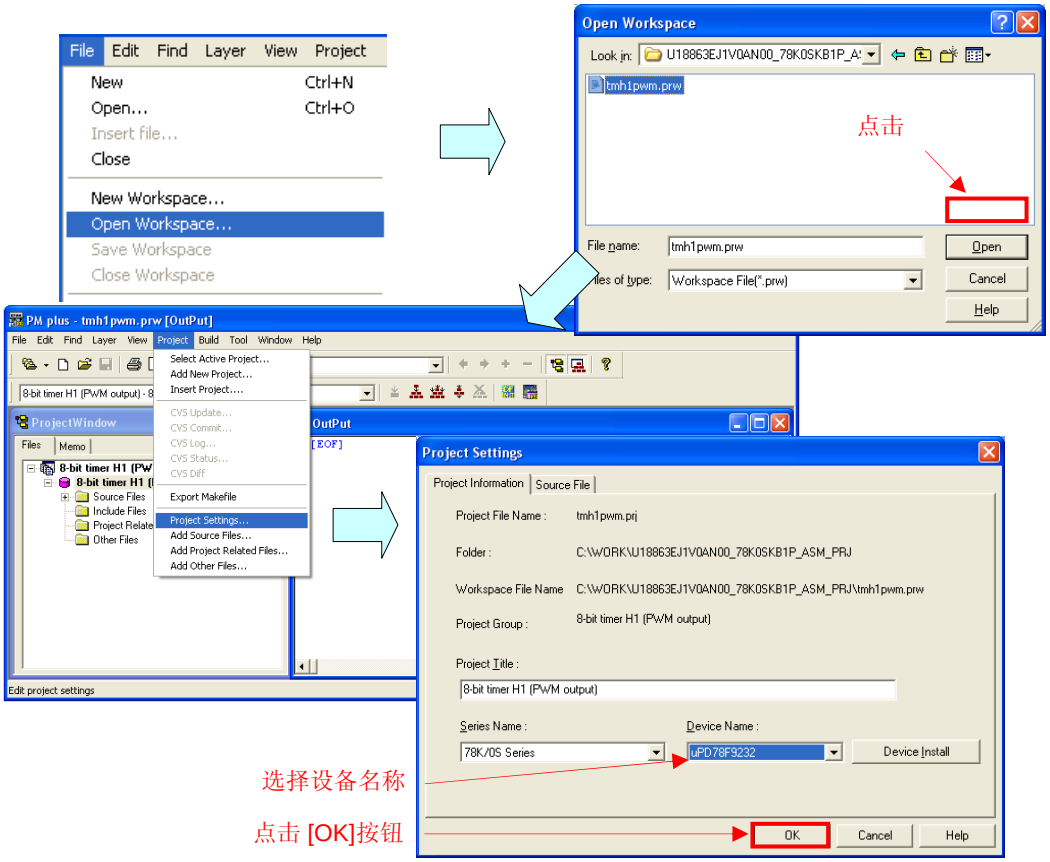
要利用适用 78K0S/Kx1+的系统仿真器SM+（下文称为“SM+”）检查示例程序的运行，在构建示例程序后必须启动SM+。本节描述运行次序的示例，从用集成开发环境PM+构建示例程序到启动SM+；使用 选择下载的汇编语言文件（源文件+项目文件）。关于如何构建其他下载的程序，请参见**78K0S/Kx1+示例程序启动向导的应用示例**的“**第三章注册集成开发环境PM+项目并执行构建**”。

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+.

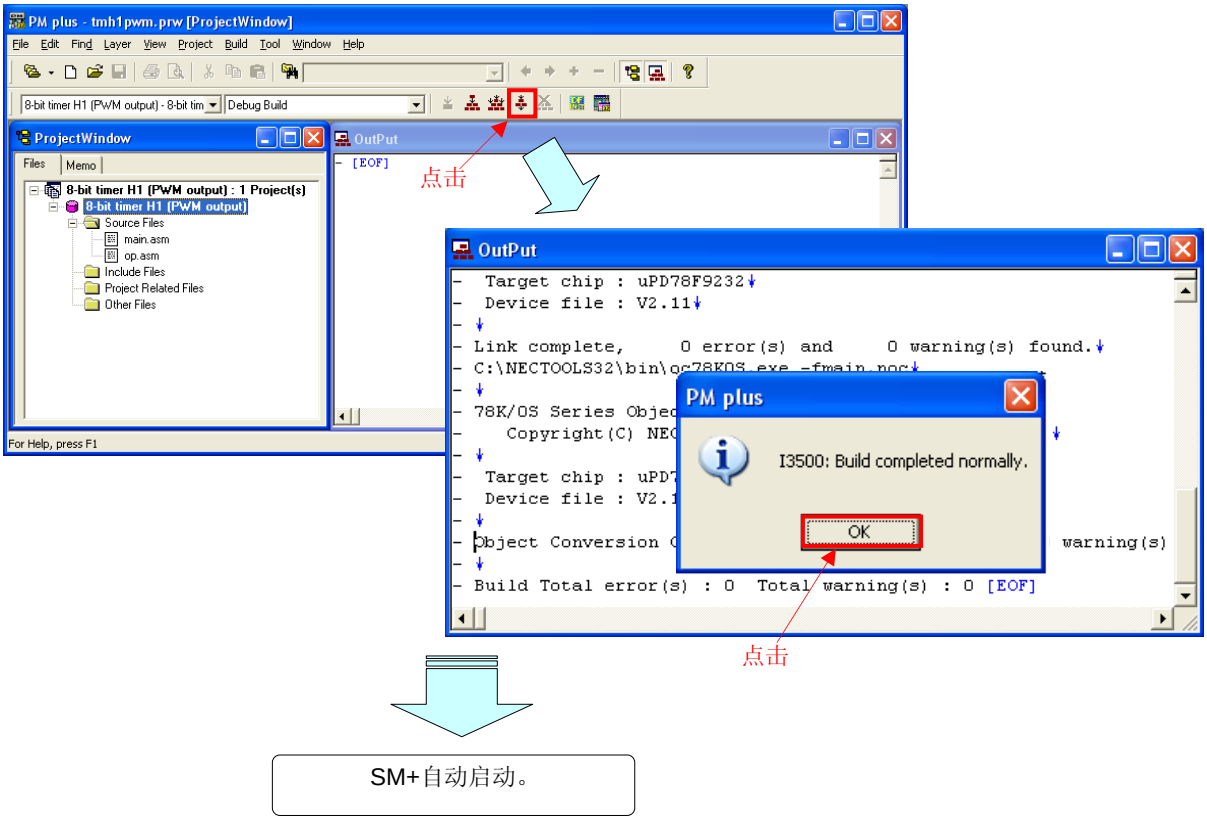
(2) 在[文件]菜单点击[打开]再点击[打开工作区]，选择“tmh1pwm.prw”。工作区生成，源文件将自动放入该工作区中。

(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) 点击信息窗口的 [OK]按钮自动启动 SM+。



5.2 带SM+运行

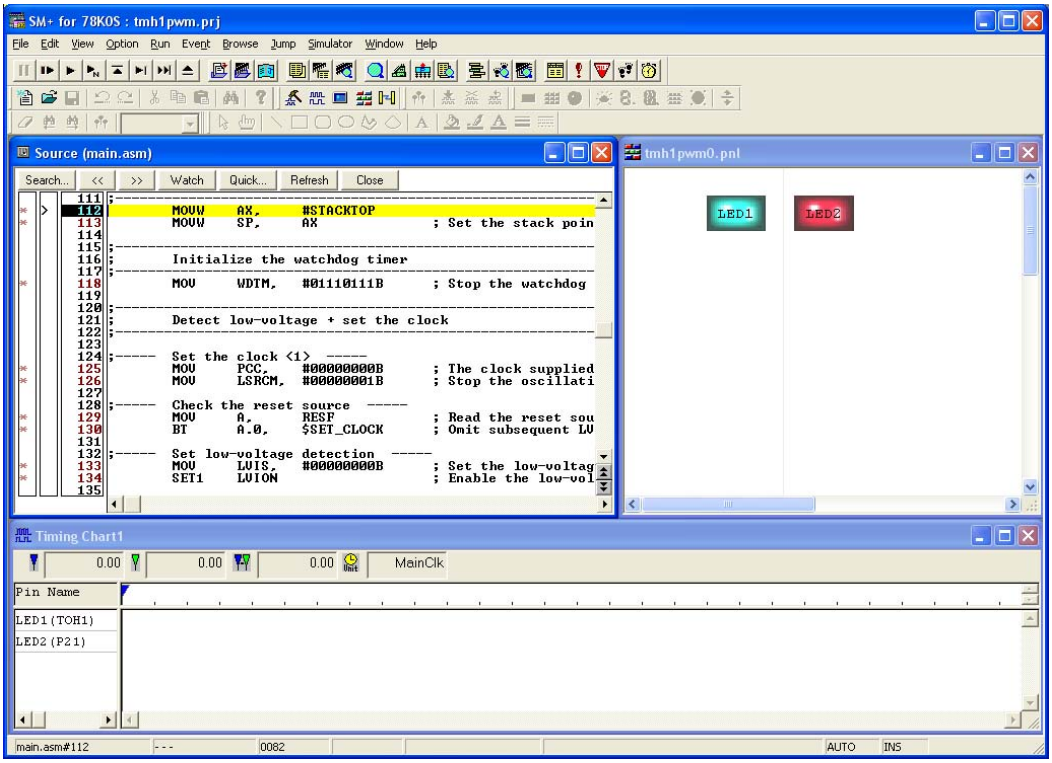
本节描述在 SM+的 I/O 面板窗口或时间图窗口中对运行进行检查的示例。
关于如何操作SM+的详情，请参见**SM+系统仿真器操作用户手册**。

[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。

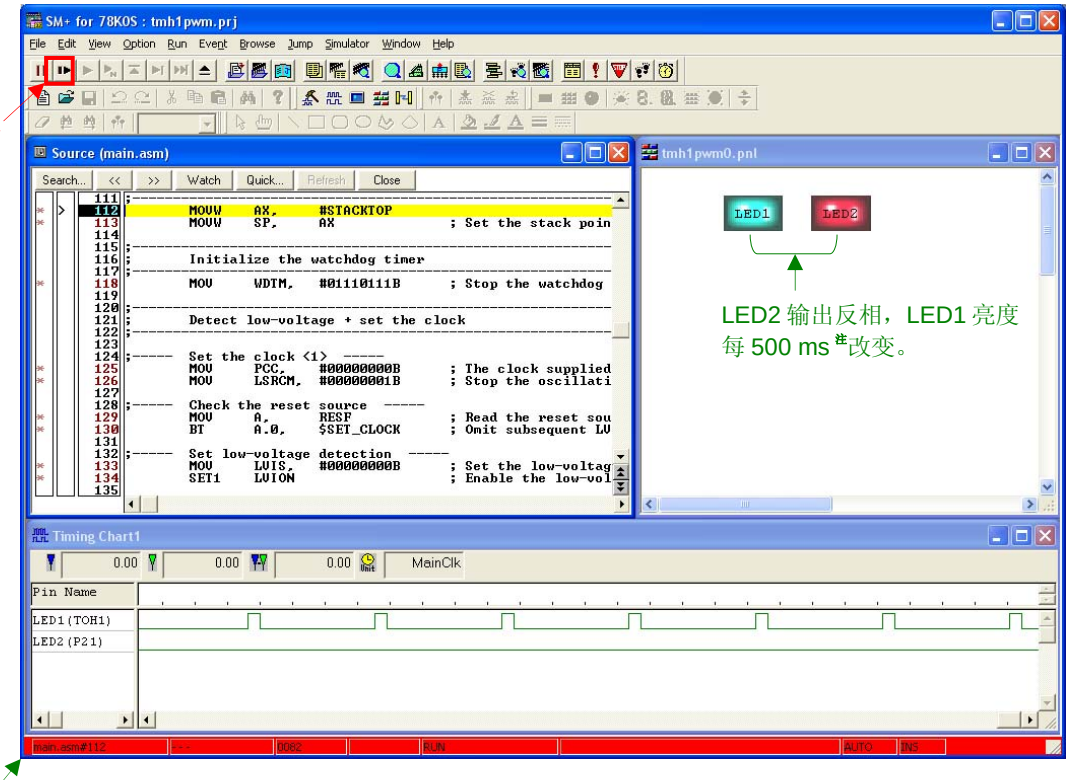
<1>从[工具]菜单选择[编译器选项]。
<2>显示[编译器选项]对话框。 选择[启动例程]标签。
<3>不选[使用标准库的固定区域]复选框。 （其他复选框不动。）

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。

(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。）



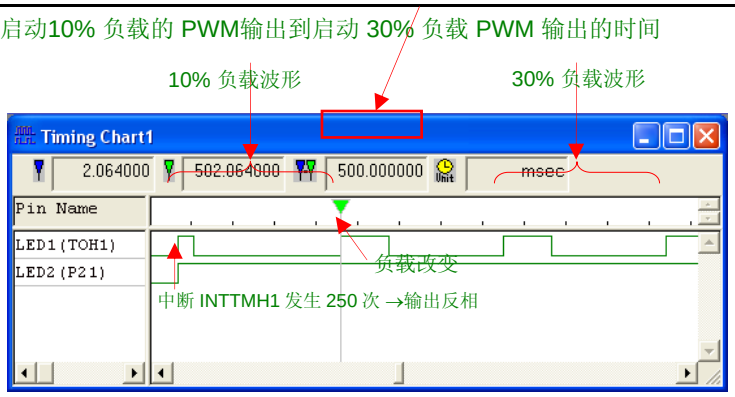
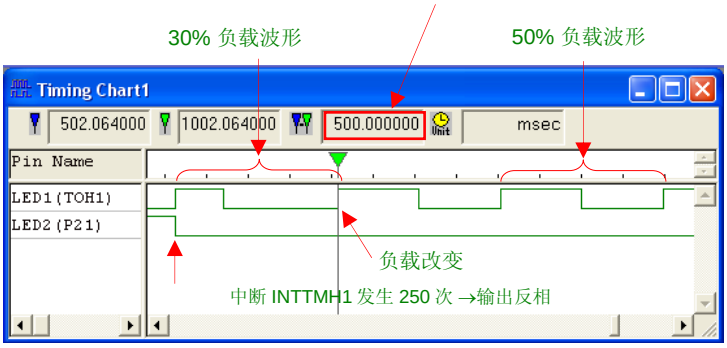
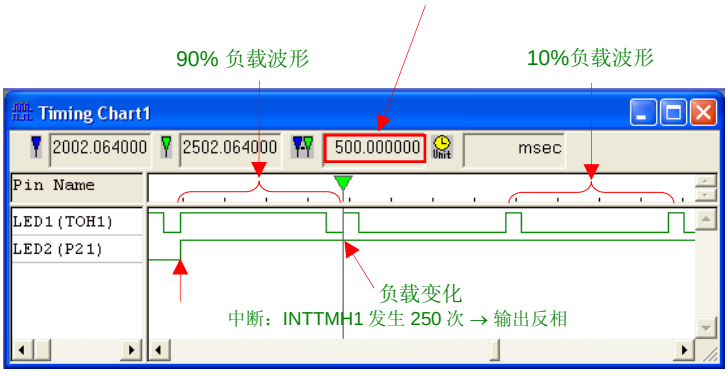
(2) 点击  重新启动]按钮）。在 CPU 复位后，程序开始执行，屏幕显示如下。



注 这将与实际周期不同，取决于所使用的个人电脑的运行环境

(3) 在程序执行过程中，在时间图窗口中检查波形，每次 PWM 输出负载 （[LED1] 亮度）改变时， [LED2]输出反相。

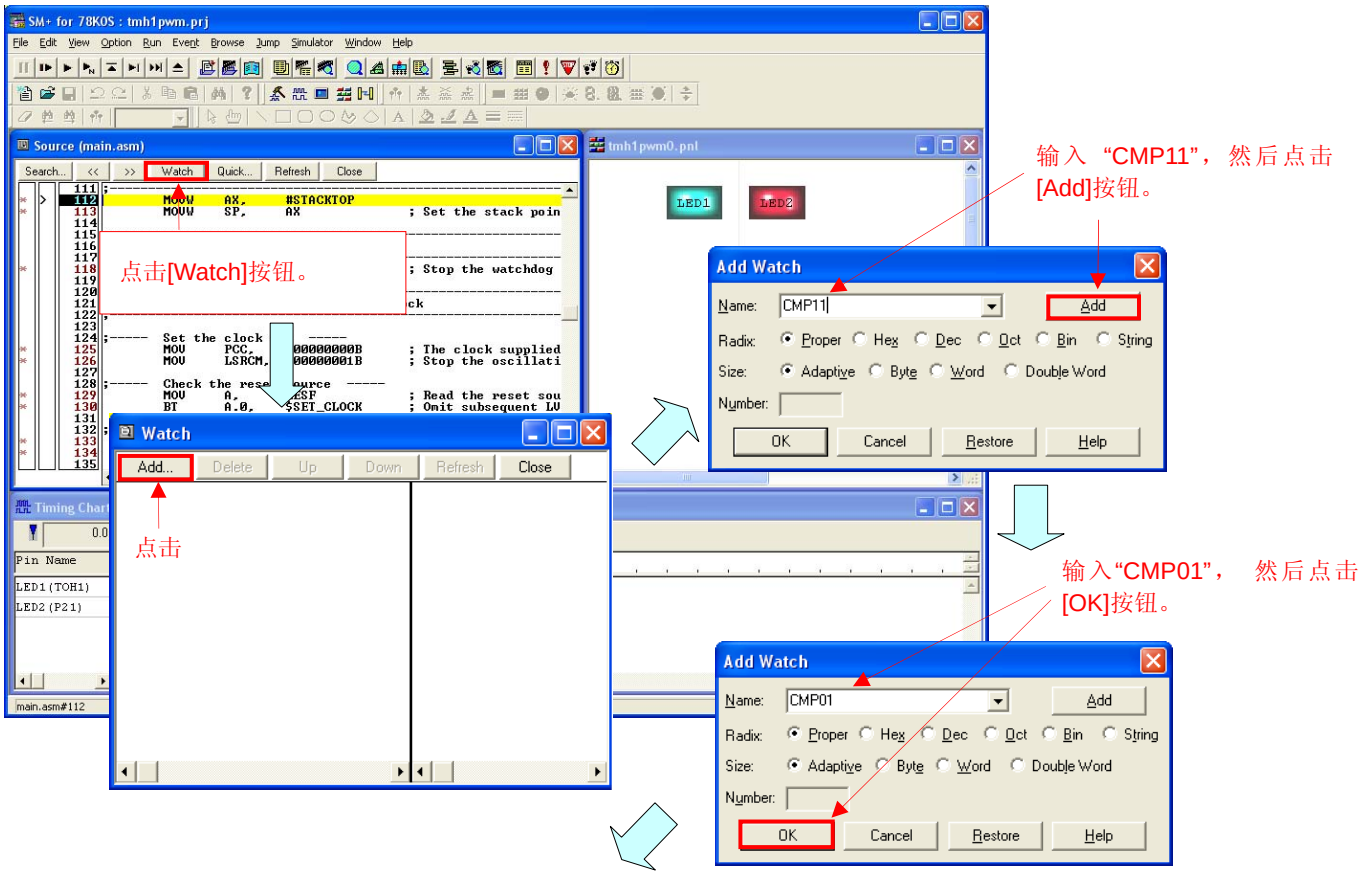
Note

PWM输出负载 ([LED1] 亮度)	时间图窗口
负载: 10% （亮度： 90%） ↓ 负载: 30% （亮度： 70%）	启动10% 负载的 PWM输出到启动 30% 负载 PWM 输出的时间 
负载: 30% （亮度： 70%） ↓ 负载: 50%（亮度： 50%）	从启动 30% 负载 PWM 输出到启动 50% 负载 PWM 输出的时间 
• • •	• • •
负载: 90%（亮度： 10%） ↓ 负载: 10% （亮度： 90%）	从启动 90% 负载 PWM 输出到启动 10% 负载 PWM 输出的时间 

注 PWM 输出在 90%负载后开始重复 10%负载变化。

[附件]　　利用 SM+察看功能，可检查 CMP01 寄存器和 CMP11 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” 和 “CMP11” 并点击[OK]按钮对 “CMP01” 和 “CMP11” 进行注册，再关闭[添加察看]窗口。



<4> 执行程序并检查 [Watch] 窗口中 CMP01 和 CMP11 数据值变化。

PWM 输出负载 ([LED1] 亮度[注]	[察看]窗口的数据值
负载: 10% （亮度：90%）	CMP01: 0xF9 (249), CMP11: 0x18 (24)
负载: 30% （亮度： 70%）	CMP01: 0xF9 (249), CMP11: 0x4A (74)
负载: 50% （亮度： 50%）	CMP01: 0xF9 (249), CMP11: 0x7C (124)
负载: 70% （亮度： 30%）	CMP01: 0xF9 (249), CMP11: 0xAE (174)
负载: 90% （亮度：10%）	CMP01: 0xF9 (249), CMP11: 0xE0 (224)

注 PWM 输出在 90%负载后开始重复10%负载变化。

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）内部定时器	PDF

附件A 程序列表

作为程序列表示例，78K0S/KB1+微处理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-位计时器 H1 (PWM 输出)
;*******************************************************************************
;<<历史>>
;           2007.7.--             发布
;*******************************************************************************
;
;<<概述>>
;
;本示例程序提供使用 8-位计时器 H1 的 PWM 输出功能的例子。
;利用连接 8-位定时器定时器输出（TOH1 引脚）H1 到 LED1 通过 PWM 输出负载控制 LED1 亮度。
;利用定时器 H1 中断周期性改变负载，周期为 500 ms，同时，LED2 输出反相。
;
;
;  <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 8 MHz
;  - 设置供给外围硬件的时钟为 8 MHz
;
;
;  <8-位计时器 H1 设置>
;  - 设置为 PWM 模式
;  - 启动 TOH1 引脚计时器输出
;  - 计数时钟 = fxp/2^6 (125 kHz)
;  - 计时器周期 = 2 ms (8[us/clk] x 250[计数] = 2[ms])
;
;
;  <PWM 输出负载和 LEDs>
;
;  - LED2 输出反相，同时，负载按下列顺序每 500 ms 变化。
;    +-----------------+-----------------------------+
;          | PWM 输出负载 | 10% | 30% | 50% | 70% | 90% | (因此，从 10%开始重复变化。)
```

```
;   +-----------------+-----------------------------+
;   | LED1 亮度       | 90%                         | 70%     | 50%     | 30%      | 10%     |
;   +-----------------+-----------------------------+
;   # PWM 输出为高活动，LED1 为低活动；因此，LED
;    亮度= 100 –负载因数。
;
;
;<<I/O 端口设置>>
;
; 输入：-
; 输出：P00-P03, P20-P23, P30-P33, P40-P47, P120-P123, P130
; #所有未使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      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 RAM
;
;========================================================================
XRAM    DSEG    SADDR
CNT_TMH1:       DS      1               ; 用于计算 INTTMH1 中断
```

```
;===============================================================================
;
;           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      ; 设置低压检测电压(VLVI)为 4.3 V +-0.2 V
        SET1    LVION                   ; 启用低压监测器的运行

        MOV     A,      #40             ; 分配 200 us 等待计数值
;-----  200 us 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，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      ; 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,     #00000000B      ; 将 P20-P23 的输出锁存器设置为低(P21: 开 LED2)
        MOV     PM2,    #11110000B      ; 设置 P20-P23 为输出模式

;---------------------------------------------------------------------------
;       初始化端口 3
;---------------------------------------------------------------------------
        MOV     P3,     #00000000B      ; 将 P30-P33 的输出锁存器设置为低
        MOV     PM3,    #11110000B      ; 设置 P30-P33 为输出模式

;---------------------------------------------------------------------------
;       初始化端口 4
;---------------------------------------------------------------------------
        MOV     P4,     #00000000B      ; 将 P40-P47 的输出锁存器设置为低(P42: 开 LED1)
        MOV     PM4,    #00000000B      ; 设置 P40-P47 为输出模式

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

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

```
;-------------------------------------------------------------------------------
;       初始化 RAM
;-------------------------------------------------------------------------------
        MOV     CNT_TMH1, #250          ; 初始化 INTTMH1 中断数目

;-------------------------------------------------------------------------------
;       设置 8-位计时器 H1
;-------------------------------------------------------------------------------
        MOV     TMHMD1,         #00111001B      ; 计数时钟= fxp/2^6 = 125 kHz, PWM 模式, 启动 TOH1 输出
        MOV     CMP01, #250-1; 初始化 CMP01 (周期：2 ms)
        MOV     CMP11, #25-1    ; 初始化 CMP11 (负载：10%)
        SET1    TMHE1                   ; 开始计时器运行

;-------------------------------------------------------------------------------
;       设置中断
;-------------------------------------------------------------------------------
        MOV     IF0,    #00H            ; 事先清除无效中断请求
        CLR1    TMMKH1                  ; 将 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

;*******************************************************************************
;
;       中断 INTTMH1
;
;*******************************************************************************
INTERRUPT_TMH1:
        DBNZ    CNT_TMH1, $END_INTTMH1      ; 如果 INTTMH1 中断数目< 250，则分支执行

        MOV     CNT_TMH1, #250              ; 初始化中断数目
        XOR     P2,     #00000010B          ; LED2 反相

        CMP     CMP11, #225-1   ; 将负载与 90%时比较
        BC      $INC_CMP11                  ; 如< 90%，则分支执行

        MOV     CMP11, #25-1            ; 初始化负载为 10%
        BR      $END_INTTMH1            ; 分支执行为 END_INTTMH1
```

```
INC_CMP11:
        ADD     CMP11, #50              ;负载增加 20%

END_INTTMH1:
        RETI                            ; 从中断服务返回
```

结束

● main.c (C 语言版)

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

        日电电子    78K0S/KB1+

*********************************************************************************
        78K0S/KB1+      示例程序
*********************************************************************************
        8-位计时器 H1 (PWM 输出)
*********************************************************************************
<<历史>>
        2007.7.--           发布
*********************************************************************************

<<概述>>

本示例程序提供使用 8－位定时器 H1 的 PWM 输出功能的例子。LED1 亮度是由连接 8－位定时器 H1 的定时器输出
  (TOH1 引脚)和 LED1 通过 PWM 输出负载控制的。利用定时器 H1 中断周期性改变负载，周期为 500 ms，同时，
  LED2 输出反相。


 <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 8 MHz
 - 设置供给外围硬件的时钟为 8 MHz


 <8-位定时器 H1 设置>
 - 设置为 PWM 模式
 - 启用 TOH1 引脚的计时器输出
 - 计数时钟= fxp/2^6 (125 kHz)
 - 计时器周期= 2 ms (8[us/clk] x 250[计数] = 2[ms])


 <PWM 输出负载和 LEDs>

 - LED2 输出反相，同时，负载按下列顺序每 500 ms 改变。
   +-----------------+----------------------------+
   | PWM 输出负载| 10% | 30% | 50% | 70% | 90% |                （此后，从 10%开始重复变化。）
   +-----------------+----------------------------+
   | LED1 亮度 | 90% | 70% | 50% | 30% | 10% |
   +-----------------+----------------------------+
   # PWM 输出为高活动，LED1 为低活动；因此，LED
     亮度 = 100 –负载因数。
```

```
<<I/O 端口设置>>

 输入：-
 输出：P00-P03, P20-P23, P30-P33, P40-P47, P120-P123, P130
 # 所有未使用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 INTTMH1 fn_inttmH1    /* 中断函数声明：INTTMH1 */

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

        定义全局变量

============================================================================*/
sreg unsigned char ucTMH1cnt = 250;     /*用于计数 INTTMH1 中断数目的 8-位变量*/

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

        RESET 后的初始化

*****************************************************************************/
void hdwinit(void){
        unsigned char ucCnt200us;       /* 用于 200 us 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;                          /* 启用低压检测器运行*/

            for (ucCnt200us = 0; ucCnt200us < 9; ucCnt200us++){          /* 等待 200 us 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;               /* 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    = 0b00000000;               /* 将 P20-P23 的输出锁存器设置为低(P21:打开 LED2) */
      PM2   = 0b11110000;               /* 设置 P20-P23 为输出模式*/

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

/*---------------------------------------------------------------------------
      初始化端口 4
---------------------------------------------------------------------------*/
      P4    = 0b00000000;               /* 将 P40-P47 的输出锁存器设置为低(P42: 打开 LED1) */
      PM4   = 0b00000000;               /* 设置 P40-P47 为输出模式*/

/*---------------------------------------------------------------------------
      初始化端口 12
---------------------------------------------------------------------------*/
      P12   = 0b00000000;               /* 将 P120-P123 的输出锁存器设置为低*/
```

```
        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 = 0b00111001;            /* 计数时钟= fxp/2^6 = 125 kHz, PWM 模式, 启用 TOH1 输出*/
        CMP01 = 250-1;                  /* 初始化 CMP01 (周期：2 ms) */
        CMP11 = 25-1;                   /* 初始化 CMP11 (周期：10%) */
        TMHE1 = 1;                              /* 开始计时器运行*/

/*---------------------------------------------------------------------------
        设置中断
---------------------------------------------------------------------------*/
        IF0   = 0x00;                   /* 事先将无效中断请求清零*/
        TMMKH1 = 0;                             /* INTTMH1 中断去屏蔽*/

        返回;
}

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

        主循环

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

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

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

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

        中断 INTTMH1

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

        if (--ucTMH1cnt == 0){          /* 当 INTTMH1 中断数目为 250 时处理*/
```

```
            ucTMH1cnt = 250;                    /* 初始化中断数目*/
            P2 ^= 0b00000010;                   /* Reverse LED2 */

            if (CMP11 >= 225-1){        /* 当负载至少为 90%时处理*/
                  CMP11 = 25-1;     /* 初始化负载为 10% */
            }
            else {
                  CMP11 += 50;      /*负载增加 20% */
            }
      }

      返回;
}
```

● op.asm (汇编语言和 C 语言共用)

```
;=========================================================================
;
;         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
;                               |++-----------     高速内部振荡器时钟(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

结束
```

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