应用注释

78K0S/Kx1+

示例程序 (16-位定时器/事件计数器 00)

外部事件计数器

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 16-位定时器/事件计数器 00 的外部事件计数器功能。在示例程序中，利用 16-位定时器/事件计数器 00 的外部事件计数器功能，在估定时间检测到外部脉冲输入的下降边缘时， LED 输出反相。

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： U18888CA1V0AN00（第一版）
出版日期：2008 年 03 月 N

中文版

· 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 16 位计时器/事件计数器 00 的外部事件计数器功能的示例。外部脉冲输入的下降沿每检测 10 次，LED 输出进行反相（仅在第一个反相时为 11 次）。

1.1 初始设置的主要内容

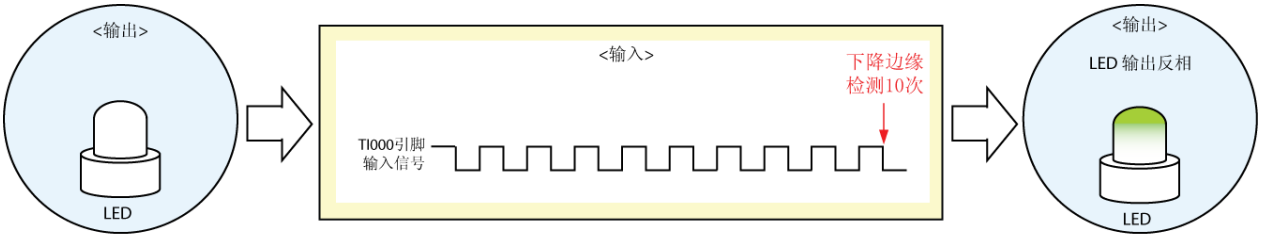
初始设置的主要内容如下。

- 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 V_{LVI}（低压检测电压）设置为 4.3V±0.2V
- 在 V_{DD}（供电电压）大于等于 V_{LVI} 后，当检测到 V_{DD} 小于 V_{LVI} 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 16 位计时器/事件计数器 00
  - 设置 CR000 为比较寄存器
  - 将外部事件计数器的比较值设置为 CR000
  - 将计数时钟设置为 TI000 引脚的有效沿（下降沿）
  - 将工作模式设置为当 TM00 和 CR000 匹配时清零并启动
- 启用 INTTM000 中断

注 通过选项字节进行设置。

1.2 主循环之后的内容

在初始设置完成后，利用 16 位计时器/事件计数器 00 中断（INTTM000）的生成，每 10 次（仅在第一次反相时为 11 次）检测到脉冲输出，LED 输出就会反相。



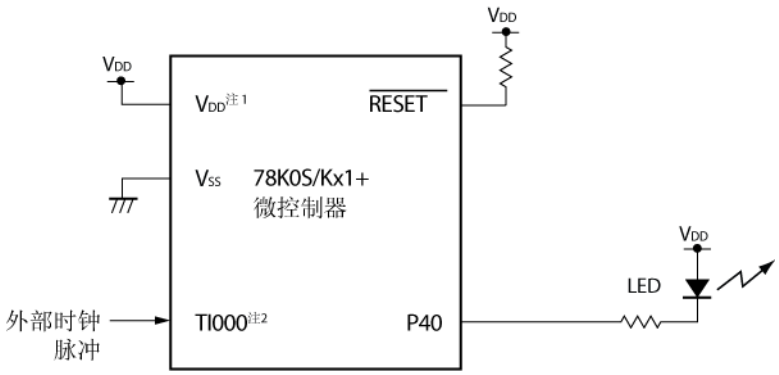
注意事项 关于使用设备的注意事项，请参见各产品（78K0S/KU1+、78K0S/KY1+、78K0S/KA1+、78K0S/KB1+）的用户手册。

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≤VDD≤5.5V。

2. TI000/INTP0/P30： 78K0S/KA1+和 78K0S/KB1+微控制器

TI000/ANI0/TOH1/P20： 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。

●LED

LED 输出对应于 16 位计时器/事件计数器 00 的外部事件计数器功能。

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	●	●	
tm00evc.prw	用于集成开发环境PM+的工作区文件		●	
tm00evc.prj	用于集成开发环境PM+的项目文件		●	
tm00evc.pri tm00evc.prs tm00evc.prm	用于适用78K0S/Kx1+的系统仿真器SM+的项目文件		●注2	
tm00evc0.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	●
tm00evc0.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。

- 外部事件计数器功能： 16 位计时器/事件计数器 00
- VDD<VLVI 检测： 低压检测器（LVI）
- 外部脉冲输入： TI000 注
- LED 输出： P40（输出端口）

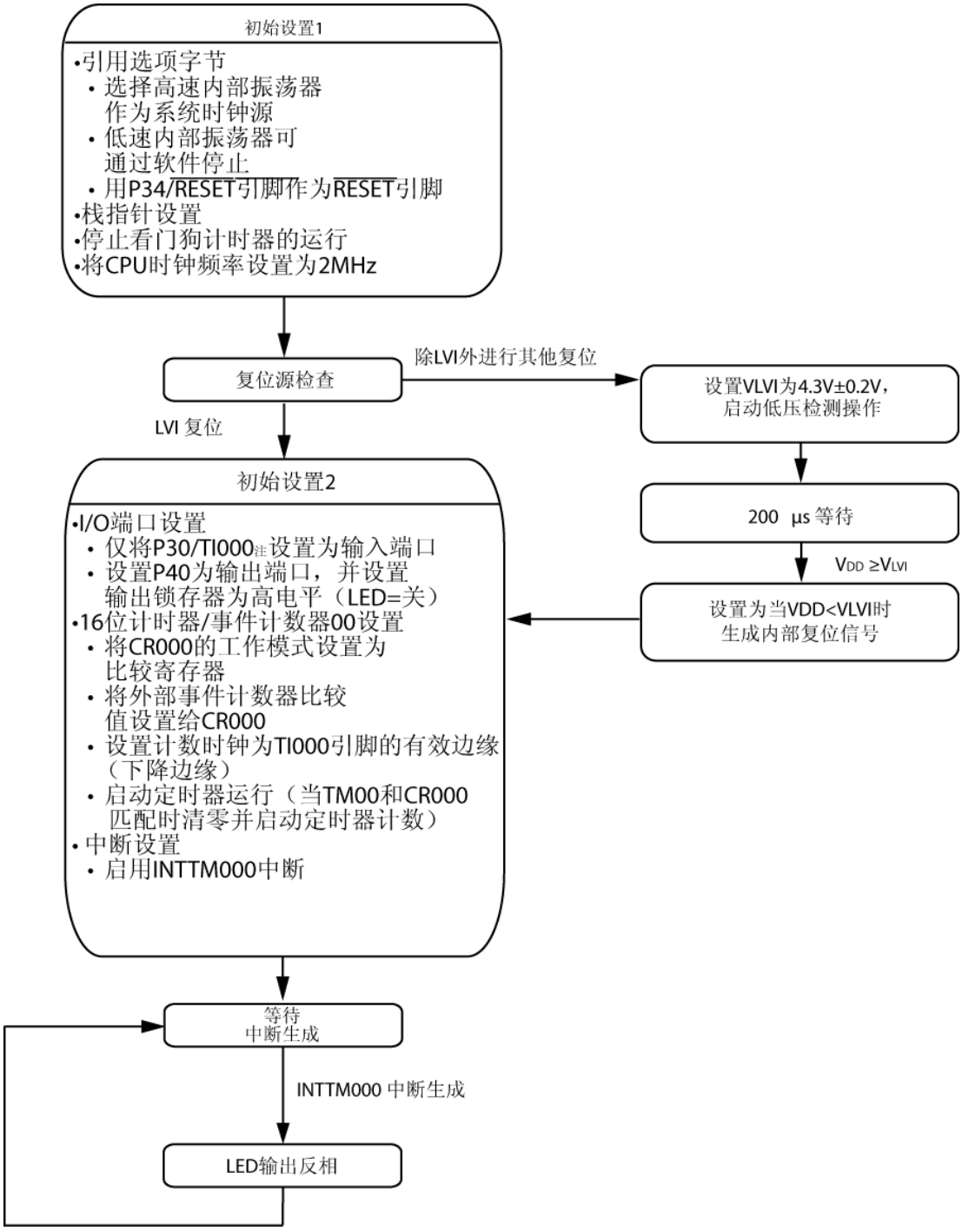
注 TI000/INTP0/P30： 78K0S/KA1+和 78K0S/KB1+微控制器
TI000/ANI0/TOH1/P20： 78K0S/KY1+和 78K0S/KU1+微控制器

3.3 初始设置及运行概览

本示例程序进行的操作有：包括低压检测功能的设置在内的初始设置、时钟频率的选择、I/O 端口的设置、16 位计时器/事件计数器 00（外部事件计数器功能）的设置、中断设置。

在初始设置完成后，利用 16 位计时器/事件计数器 00 中断（INTTM000）的生成，每 10 次（仅在第一次反相时为 11 次）检测到脉冲输出，LED 输出就会反相。

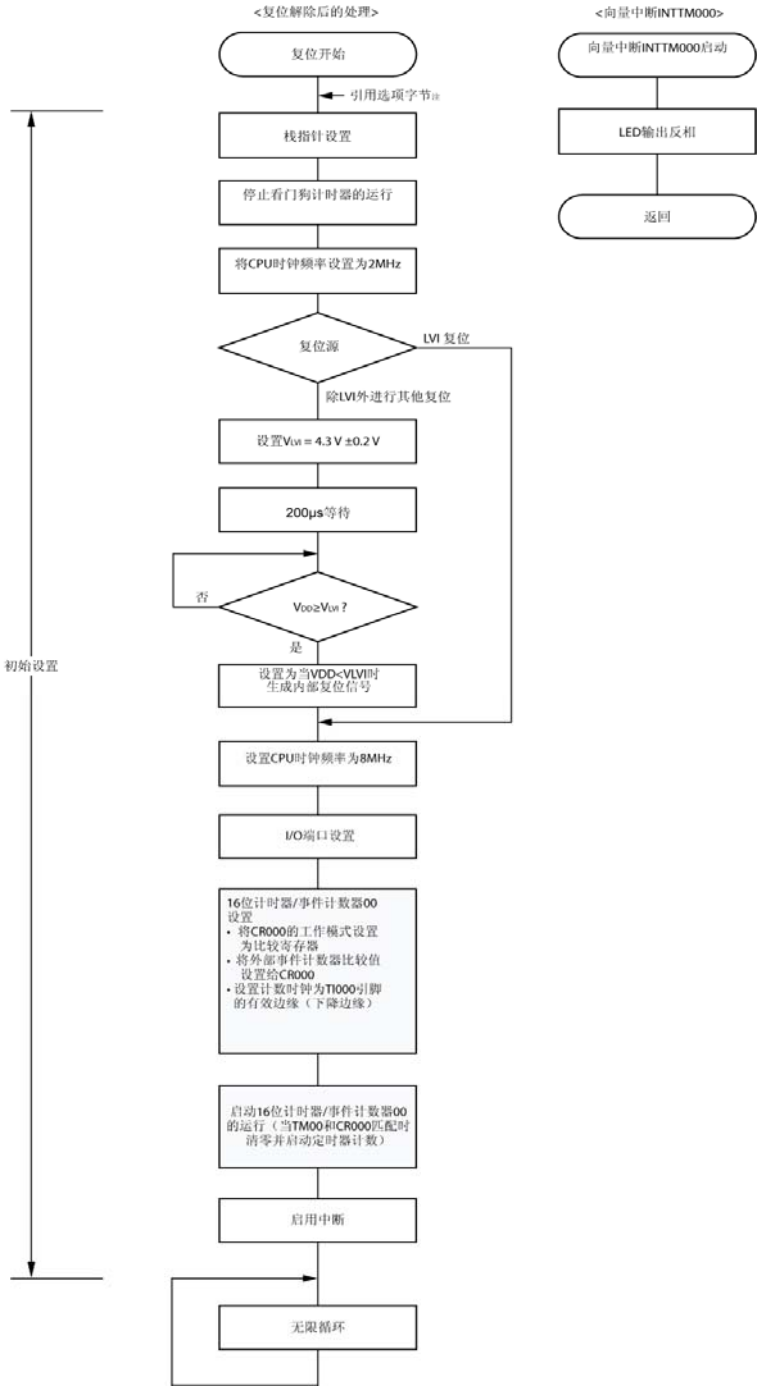
详情在如下所示的状态转换图中描述。



注 TI000/P30：78K0S/KA1+和 78K0S/KB1+微控制器
TI000/P20：78K0S/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 16 位计时器/事件计数器 00 的外部事件计数器功能。

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）检测小于 2.7V过程中生成复位的应用注释。**。

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 16 位计时器/事件计数器 00 的外部事件计数器功能

当使用 16 位计时器/事件计数器 00 作为外部事件计数器时，当 TM00 计数器和 CR000 寄存器匹配时会对外部事件输入（从 TI000 引脚输入）的有效沿进行计数并生成中断信号（INTTM000）。

Inttm000 信号在下列定时处生成。

- INTTM000 信号生成定时（仅第一次）
  =检测到外部事件输入的有效沿次数×（CR000 设置值+2）

- INTTM000 信号生成定时（第二次及之后各次）
  =检测到外部事件输入的有效沿次数×（CR000 设置值+1）

当 TI000 引脚输入信号以 fXP 时钟周期采样且连续两次检测到有效沿电平时，第一次对有效沿进行检测。因此，可消除短脉冲宽度的噪声。

在使用 16 位计时器/事件计数器 00 作为外部事件计数器时，设置下面的六种寄存器。

- 捕捉/比较控制寄存器 00（CRC00）
- 16 位定时器捕捉/比较寄存器 000（CR000）
- 预换算器模式寄存器 00（PRM00）
- 16 位定时器模式控制寄存器 00（TMC00）
- 端口模式寄存器 x（PMx）[注]
- 端口模式控制寄存器 x（PMCx）[注]

注 因为在使用外部事件计数器功能时用 TI000 引脚作为定时器输入，所以应如下设置。

	PMx 寄存器	PMCx 寄存器
78K0S/KA1+和 78K0S/KB1+微控制器	PM30=1	无需设置
78K0S/KY1+和 78K0S/KU1+微控制器	PM20=1	PMC20=0

<当使用 16 位计时器/事件计数器 00 作为外部事件计数器时的基本操作设置步骤示例>

<1> 设置 CRC00 寄存器

<2> 设置任意值给 CR000 寄存器

<3> 利用 PRM00 寄存器将计数时钟设置为 TI000 引脚的有效沿

<4> 设置 TMC00 寄存器：启动运行

注意事项 <1>至<3>步可任意进行。

（1）设置 CRC00 寄存器

该寄存器控制 CR000 和 CR010 寄存器的运行。

注意事项 当使用 16 位计时器/事件计数器 00 作为外部事件计数器时，不使用 CR010。

图 4-1 捕捉/比较控制寄存器 00（CRC00）的格式

CRC00

0	0	0	0	0	CRC002	CRC001	CRC000

CR000 工作模式选择

0	用作比较寄存器
1	用作捕捉寄存器

CR000 捕捉触发选择

0	TI010引脚有效沿捕捉。
1	捕捉TI000引脚的有效沿沿的相反相位。

CR010 工作模式选择

0	用作比较寄存器
1	用作捕捉寄存器

注意事项 1. 必须先停止定时器运行才能设置 CRC00 寄存器。

2. 当通过 TMC00 寄存器设置为当 TM00 和 CR000 匹配时选择清零并开始模式时，不要指定 CR000 寄存器为捕捉寄存器。

（2）设置 CR000 寄存器

该寄存器具有捕捉寄存器和比较寄存器二者的功能。

图 4-2　16 位定时器捕捉/比较寄存器 000（CR000）的格式

CR000

用 CR000 作为比较寄存器时

设置给 CR000 的值不断与 16 位定时器计数器 00（TM00）的计数值进行比较，如果二者匹配就会生成中断请求（INTTM000）。当 TM00 设置为间隔定时器操作时，它还可以用作放置间隔时间的寄存器。

•间隔时间=（N+1）/fsam

注意事项 1. 通过 TM00 和 CR000 匹配进入清零并开始，在其中将非 0000H 值设置给 CR000 寄存器。

2. 如果 CR000 寄存器的新值小于 TM00 计数器的值，则 TM00 继续计数，溢出，然后从 0 开始重新计数。因此，如果 CR000 寄存器的新值小于原值，在 CR000 寄存器的值改变后定时器必须复位并重新启动。

3. Tm00 计数器停止后 CR000 寄存器的值无法保证。

4. 对于设置为比较模式的 CR000 寄存器，即便输入了捕捉触发也不会进行捕捉操作。

5. 在 TM00 计数器工作过程中改变 CR000 计数器的设置可能会导致故障。

备注　N：　CR000 寄存器设置值（0001H 至 FFFFH）

fsam：TM00 计数器计数时钟频率

（3）设置 PRM00 寄存器

该寄存器用于设置 TM00 计数器的计数时钟及 TI000、TI010 引脚输入的有效沿。

图 4-3　预换算器模式寄存器 00（PRM00）的格式

PRM00

ES110	ES100	ES010	ES000	0	0	PRM001	PRM000

计数时钟 (fsam)选择

0	0	f_{XP}
0	1	$f_{XP}/2^2$
1	0	$f_{XP}/2^8$
1	1	TI000引脚有效沿沿[注]

TI000引脚有效沿沿选择

0	0	下降沿沿
0	1	上升沿沿
1	0	禁止设置
1	1	上升及下降沿沿

TI010引脚有效沿沿选择

0	0	下降沿沿
0	1	上升沿沿
1	0	禁止设置
1	1	上升及下降沿沿

注　外部时钟要求脉冲长度大于内部时钟（f_{XP}）的两个周期。

备注　f_{XP}：供给外围硬件的时钟的振荡频率

注意事项 1. 一定要在停止定时器工作后将数据设置给 **PRM00** 寄存器。

2. 在将 **TI000** 引脚的有效沿设置为计数时钟时，在设置清零并开始模式时不要将 **TI000** 引脚和 **TI000** 引脚的有效沿设置为捕捉触发。

3. 在下面的例子中，需小心注意的是检测到了 **TI0n0** 引脚（**n=0**、**1**）的有效沿。

<1> 高电平输入 **TI0n0** 引脚，在系统复位后 **TM00** 的运行立即启用。

→如果将上升沿或上升及下降沿指定为 **TI0n0** 引脚的有效沿，则在 **TM00** 运行启用后立即检测到上升沿。

<2> 当 **TI0n0** 引脚为高电平时，**TM00** 停止运行，然后当低电平输入 **TI0n0** 引脚时被启用。

→如果将下降沿或上升下降沿指定为 **TI0n0** 引脚的有效沿，则在 **TM00** 运行启用后立即检测到下降沿。

<3> 当 **TI0n0** 引脚为低电平时，**TM00** 停止运行，然后当高电平输入 **TI0n0** 引脚时被启用。

→如果将上升沿或上升及下降沿指定为 **TI0n0** 引脚的有效沿，则在 **TM00** 运行启用后立即检测到上升沿。

注意事项 4. 要与计数时钟一起使用 TI000 的有效沿，要用 fXP 进行采样以消除噪声。在采样有效沿之前不会进行捕捉操作，有效电平会被检测两次，从而消除较短脉冲宽度的噪声。

5. 当 TI010/TO00/Pxx 引脚用作有效沿的输入引脚（TI010）时，无法用作定时器输出引脚（TO00）。当用于定时器输出引脚（TO00）时，无法用作有效沿的输入引脚（TI010）。

（4）设置 TMC00 寄存器

该寄存器设置 16 位计时器/事件计数器 00 的工作模式、TM00 计数器清零模式及输出定时，并对溢出进行检测。

图4-4 16位定时器模式控制寄存器00（TMC00）的格式

TMC00

0	0	0	0	TMC003	TMC002	TMC001	OVF00

16位定时器计数器00（TM00）溢出检测

0	未检测到溢出。
1	检测到溢出。

			工作模式和清零模式选择	TO00反相定时选择	中断请求生成
0	0	0	停止工作(TM00清为0)	无变化	不产生
0	0	1			
0	1	0	自由运行模式	TM00、CR000匹配或TM00、CR010匹配	<用作比较寄存器> TM00、CR000匹配或TM00、CR010匹配时生成 <用作捕捉寄存器> TI000引脚或TI010引脚的有效沿沿处生成
0	1	1		TM00、CR000匹配，TM00、CR010匹配或TI000引脚的有效沿沿	
1	0	0	TI000引脚的有效沿沿处出现清零并启动	–	
1	0	1			
1	1	0	在TM00、CR000匹配时出现清零并启动	TM00、CR000匹配或TM00、CR010匹配	
1	1	1		TM00、CR000匹配，TM00、CR010匹配或TI000引脚的有效沿沿	

注意事项
1. 当非 0 值和 0（停止运行模式）分别设置给 TMC002 和 TMC003 时，TM00 计数器开始工作。要停止工作，应将 TMC002 和 TMC003 分别设置为 0 和 0。
2. 先停止定时器运行，再写入除 OVF00 标志的其他各位。
3. 在定时器停止时，即便有信号输入 TI000/TI010 引脚，也不会出现定时器计数和定时器中断。
4. 除非 TI000 引脚的有效沿选择为计数时钟，否则应先停止定时器工作再设置为 STOP 模式或系统时钟停止模式；否则，当系统时钟启动时定时器可能会发生故障。
5. 应先停止定时器工作再用 PRM00 寄存器的 4、5 位设置 TI000 引脚的有效沿。
6. 如果设置为在 TM00 和 CR000 匹配时或出现 TI000 引脚的有效沿时进入清零并开始模式
7. 即便在 TM00 计数器溢出后计数到下一个计数时钟之前（TM00 计数器变为 0001H 前）OVF00 标志清零，它也会重新设置且禁止清零。
8. 捕捉操作在计数时钟的下降沿处进行。但是，中断请求（INTTM0n0：n=0、1）出现在下个计数时钟的上升沿处。

[例1] 将计数时钟设置为TI000引脚的有效沿（下降沿），当有效沿检测10次（第一个中断检测11次）时生成中断。
（与示例程序源内容相同）

（1）寄存器设置

<1> CRC00

0	0	0	0	0	x	x	0

CR000 工作模式选择

0	用作比较寄存器

CR000 捕捉触发选择

x	（因为CR000用作比较寄存器，故设置无效。）

CR010 工作模式选择

x	（不使用CR010。）

<2> CR000
设置值（N）：9

<3> PRM00

x	x	0	0	0	0	1	1

计数时钟 (fsam)选择

1	1	TI000引脚有效沿沿

TI000引脚有效沿沿选择

0	0	下降沿沿

TI010引脚有效沿沿选择

x	x	（因为不使用TI010引脚，故设置无效。 但是，禁止设置“1, 0”。）

<4> TMC00

0	0	0	0	1	1	x	0

16位定时器计数器00（TM00）溢出检测

0	未检测到溢出。

			工作模式和清零模式选择	TO00反相定时选择	中断请求生成
1	1	0	在TM00、CR000匹配时出现清零并启动	TM00、CR000匹配或TM00、CR010匹配	<用作比较寄存器> TM00、 CR000匹配或TM00、CR010匹配时生成 <用作捕捉寄存器> TI000引脚或TI010引脚的有效沿沿处生成
1	1	1		TM00、CR000匹配，TM00、CR010匹配或TI000引脚的有效沿沿	

<5> PMx、PMCx

	PMx 寄存器	PMCx 寄存器
78K0S/KA1+和 78K0S/KB1+微控制器	PM30=1	无需设置
78K0S/KY1+和 78K0S/KU1+微控制器	PM20=1	PMC20=0

（2）示例程序

在下面的例子中，“**（1）寄存器设置**”中的“x”设置为“0”。

<1> 汇编语言（使用 78K0S/KA1+和 78K0S/KB1+微控制器时）

```
SET1   PM3.0
MOV    CRC00, #00000000B
MOVW   CR000, #9
MOV    PRM00, #00000011B
MOV    TMC00, #00001100B
```

<2> C 语言（使用 78K0S/KA1+和 78K0S/KB1+微控制器时）

```
PM3.0 = 1;
CRC00 = 0b00000000;
CR000 = 9;
PRM00 = 0b00000011;
TMC00 = 0b00001100;
```

[例2] 将计数时钟设置为TI000引脚的有效沿（上升和下降沿），当有效沿检测100次（第一个中断检测101次）时生成中断。

（1）寄存器设置

<1> CRC00

0	0	0	0	0	x	x	0

CR000 工作模式选择

0	用作比较寄存器

CR000 捕捉触发选择

x	（因为CR000用作比较寄存器，故设置无效。）

CR010 工作模式选择

x	（不使用CR010。）

<2> CR000

设置值（N）：99

<3> PRM00

x	x	1	1	0	0	1	1

计数时钟 (fsam)选择

1	1	TI000引脚有效沿沿

TI000引脚有效沿沿选择

1	1	上升及下降沿沿

TI010引脚有效沿沿选择

x	x	（因为不使用TI010引脚，故设置无效。 但是，禁止设置“1, 0”。）

<4> TMC00

0	0	0	0	1	1	x	0

16位定时器计数器00（TM00）溢出检测

0	未检测到溢出。

			工作模式和清零模式选择	TO00反相定时选择	中断请求生成
1	1	0	在TM00、CR000匹配时出现清零并启动	TM00、CR000匹配或TM00、CR010匹配	<用作比较寄存器> TM00、 CR000匹配或TM00、CR010匹配时生成 <用作捕捉寄存器> TI000引脚或TI010引脚的有效沿处生成
1	1	1		TM00、CR000匹配，TM00、CR010匹配或TI000引脚的有效沿	

<5> PMx、PMCx

	PMx 寄存器	PMCx 寄存器
78K0S/KA1+和 78K0S/KB1+微控制器	PM30=1	无需设置
78K0S/KY1+和 78K0S/KU1+微控制器	PM20=1	PMC20=0

（2）示例程序

在下面的例子中，“**（1）寄存器设置**”中的“x”设置为“0”。

<1> 汇编语言（使用 78K0S/KA1+和 78K0S/KB1+微控制器时）

```
SET1   PM3.0
MOV    CRC00, #00000000B
MOVW   CR000, #99
MOV    PRM00, #00110011B
MOV    TMC00, #00001100B
```

<2> C 语言（使用 78K0S/KA1+和 78K0S/KB1+微控制器时）

```
PM3.0 = 1;
CRC00 = 0b00000000;
CR000 = 99;
PRM00 = 0b00110011;
TMC00 = 0b00001100;
```

[该示例程序源文件节选]

如下所示为附件A程序列表，与 16 位计时器/事件计数器 00 功能相关（与上面提到的[例 1]内容相同）。

（1）汇编语言

```
EVENTCNT      EQU    10              ; 定时器 00 外部事件计数器比较值
                     •
                     •
                     •
XMAIN  CSEG   UNIT
RESET_START:
                     •
                     •
                     •
      MOV     CRC00, #00000000B     ; 用 CR000 作为比较寄存器
      MOVW    AX,    #EVENTCNT-1    ; 设置比较值
      MOVW    CR000, AX             ; 设置比较值
      MOV     PRM00, #00000011B     ; 将计数时钟设置为 TI000 引脚的下降沿
      MOV     TOC00, #00000000B     ; 不进行定时器输出
      MOV     TMC00, #00001100B     ; 启动定时器的运行（当 TM00 和 CR000 匹配时清零并启动）

      MOV     IF0,   #00H           ; 事先清除无效中断请求
      CLR1    TMMK000               ; 不屏蔽 INTTM000 中断

      EI                            ; 启用向量中断

_LOOP:
      NOP
      BR      $MAIN_LOOP            ; 转到 MAIN_LOOP

INTERRUPT_TM000:
      XOR     P4,    #00000001B     ; LED 输出反相
      RETI                          ; 从中断服务返回

end
```

设置外部事件计数器的比较值

将 CR000 的工作模式设置为比较寄存器

设置计数时钟

启动定时器的运行

清除 INTTM000 中断请求标志

启用 INTTM000 中断服务

通过 INTTM000 中断生成启动中断服务

（2）C 语言

```
#define eventCnt 10                 /*  定时器 00 外部事件计数器比较值 */
                    •
                    •
                    •
void hdwinit(void){
      unsigned char ucCnt200us;     /*  用于 200us 等待的 8 位变量 */
                    •
                    •
                    •
      CRC00 = 0b00000000;           /*  用 CR000 作为比较寄存器 */
      CR000 = eventCnt-1;           /*  初始化比较值 */
      PRM00 = 0b00000011;           /*  将计数时钟设置为 TI000 引脚的下降沿 */
      TOC00 = 0b00000000;           /*  不进行定时器输出 */
      TMC00 = 0b00001100;           /*  启动定时器的运行（当 TM00 和 CR000 匹配时清零并启动） */

      IF0   = 0x00;                 /*  事先清除无效中断请求 */
      TMMK000 = 0;                  /*  不屏蔽 INTTM000 中断 */
      return;
}

void main(void){

      EI();                         /*  允许中断 */

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

__interrupt void fn_inttm000(){

      P4 ^= 0b00000001;             /*  LED 输出反相 */

      return;
}
```

设置外部事件计数器的比较值

将 CR000 的工作模式设置为比较寄存器

设置计数时钟

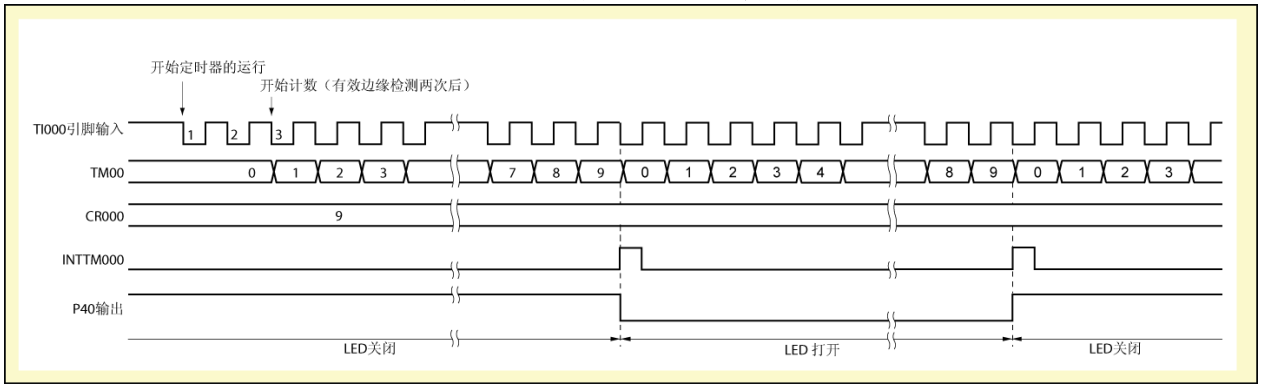
启动定时器的运行

清除 INTTM000 中断请求标志

启用 INTTM000 中断服务

通过 INTTM000 中断生成启动中断服务

图 4-5　　外部事件计数器操作时间图示例（使用本示例源程序时）



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 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

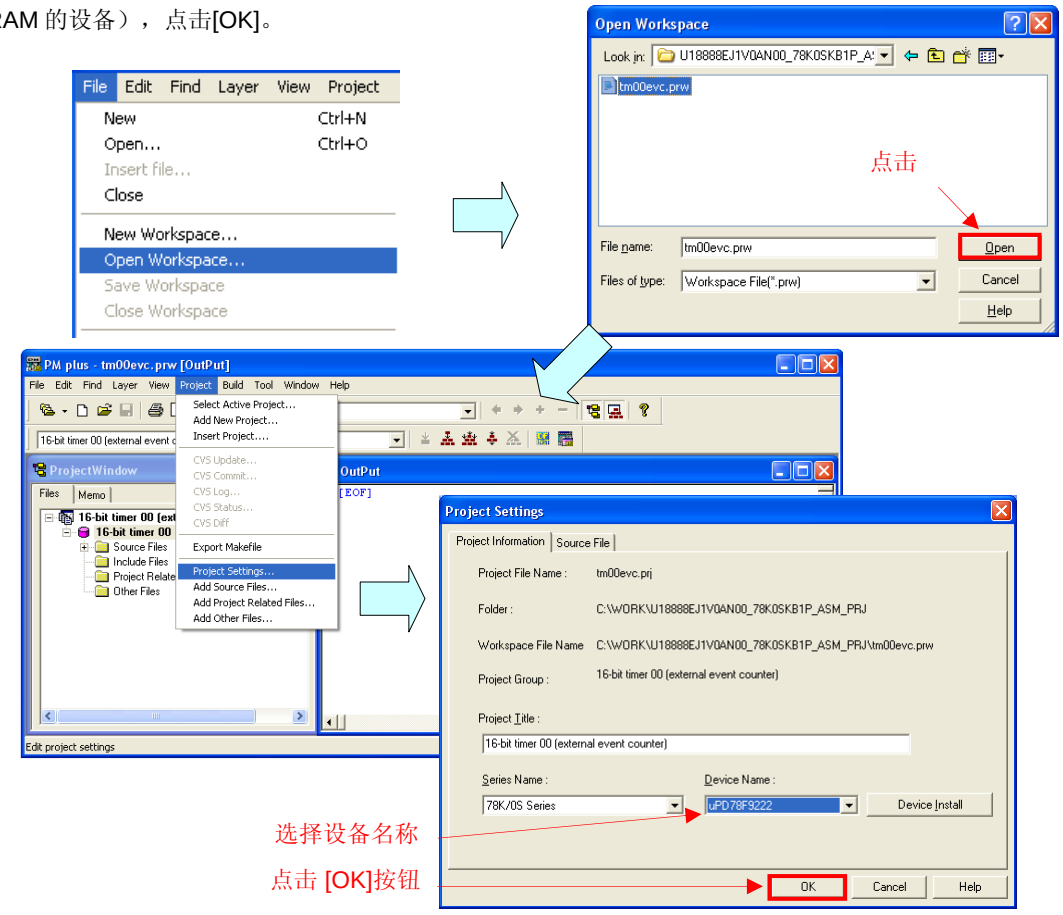
要利用适用 78K0S/Kx1+的系统仿真器SM+（下文称为“SM+”）检查示例程序的运行，在构建示例程序后必须启动SM+。本节描述运行次序的示例，从用集成开发环境PM+构建示例程序到启动SM+；使用选择 下载的汇编语言文件（源文件+项目文件）。关于如何构建其他下载的程序，请参见**78K0S/Kx1+示例程序启动向导的应用注释**的“**第三章注册集成开发环境PM+项目并执行构建**”。

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+。

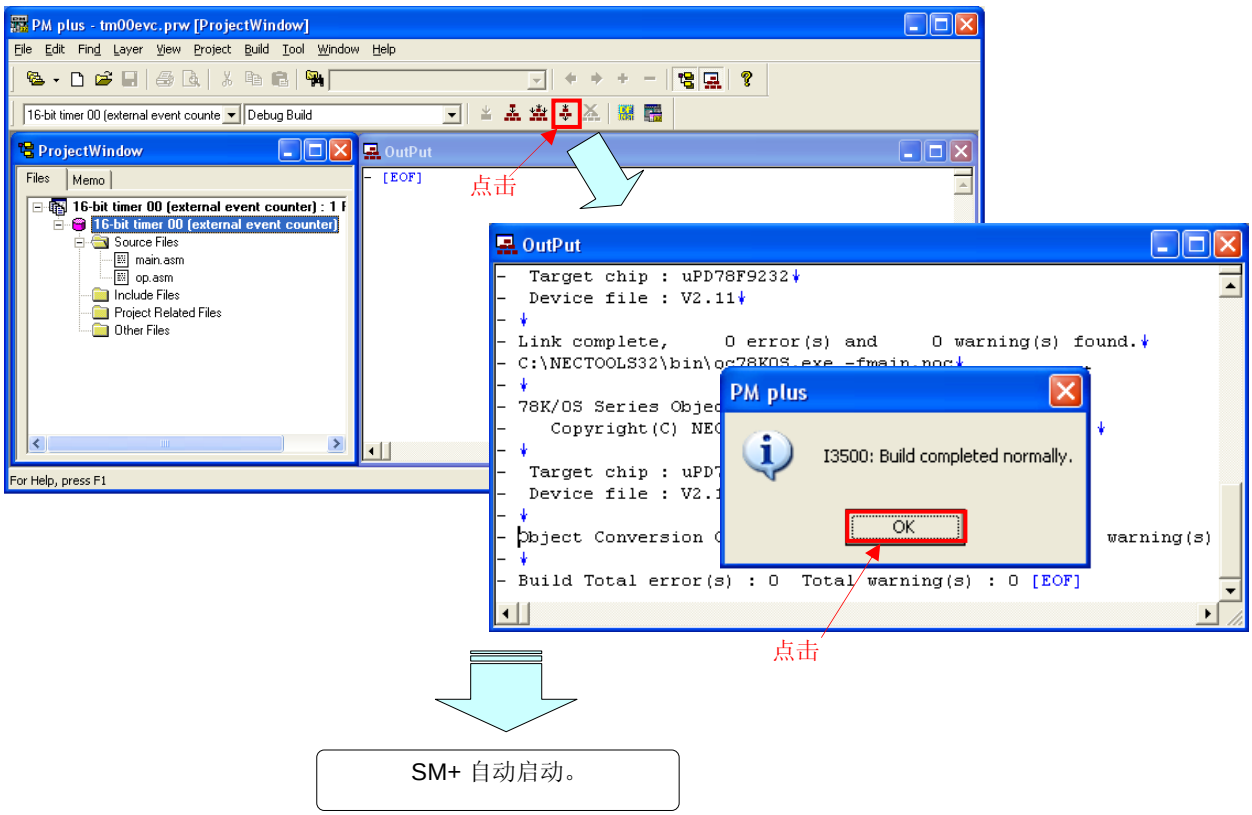
（2）在[文件]菜单点击[打开]再点击[打开工作区]，选择“tm00evc.prw”。工作区生成，源文件将自动放入该工作区中。

（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+。



[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。

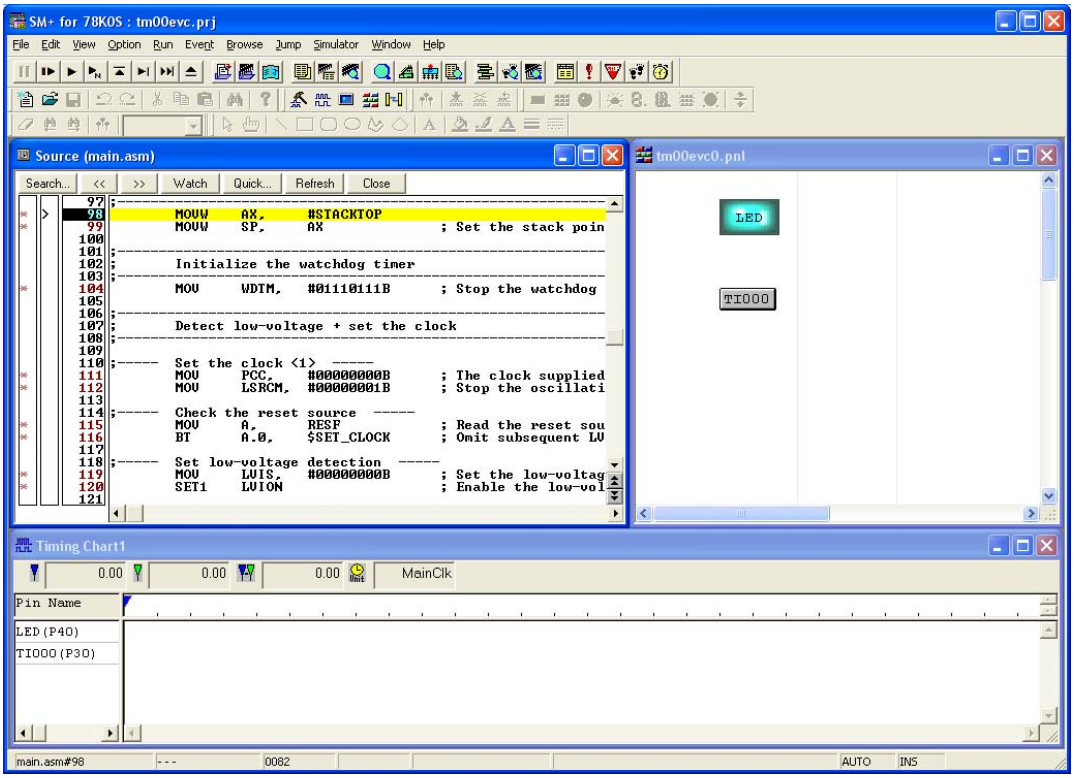
<1> 从[工具]菜单选择[编译器选项]。
<2> 显示[编译器选项]对话框。 选择[启动例程]标签。
<3> 不选[使用标准库的固定区域]复选框。 （其他复选框不动。）

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。

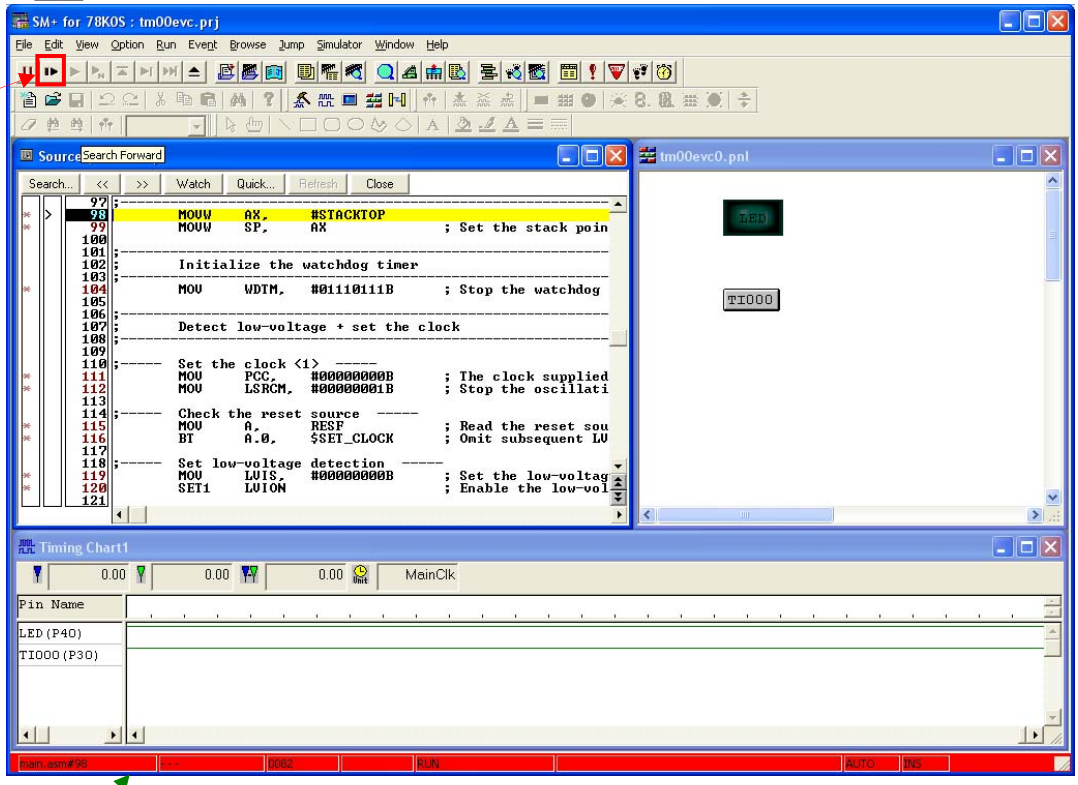
5.2　随SM+运行

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。

（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。

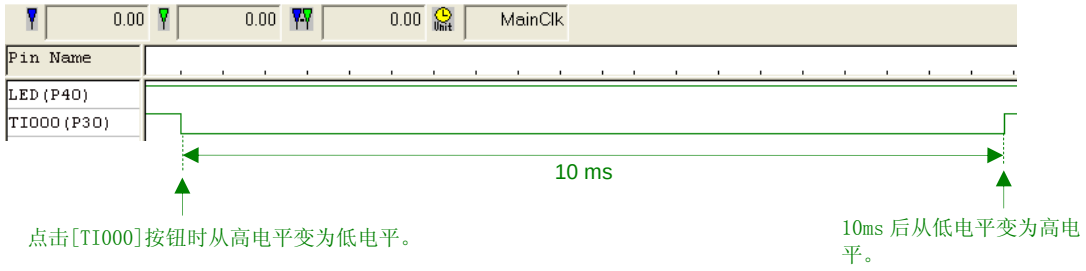


程序执行时变为红色。

（3）在程序执行过程中，在 I/O 面板窗口中点击[TI000]按钮。

当点击[TI000]按钮时，TI000 的波形如下。

时间图窗口



点击[TI000]按钮 10 次，检查有效沿（下降沿）检查 10 次[注]，且 I/O 面板窗口中的[LED]输出和时间图窗口中显示的 LED 波形均反相。

I/O面板窗口	时间图窗口
LED TI000 [TI000]按钮每点击 10 次，LED 输出就会反相。 LED TI000	LED 输出反相 0.00 0.00 0.00 MainClk Pin Name LED(P40) TI000(P30) [TI000]按钮点击次数：10 LED 输出反相 0.00 0.00 0.00 MainClk Pin Name LED(P40) TI000(P30) [TI000]按钮点击次数：10

注　对于实际设备，只有第一次 LED 输出反相出现在 TI000 引脚的有效沿（下降沿）检测到 11 次时。（第二次及之后的 LED 输出反相出现在有效沿检测到 10 次时。）

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	样本编程启动向导	PDF
	样本编程（初始设置）LED 照明开关控制	PDF
	样本编程（中断）开关输入引起的外部中断	PDF
	样本编程（低压检测）电压低于2.7V检测时复位发生。	PDF
	样本编程（16-位 定时器/事件计数器00）内部定时器	PDF
	样本编程（16-位定时器/事件计数器00）脉冲宽度测量	PDF
	样本编程（16-位定时器/事件计数器00）PPG输出	PDF
	样本编程（16-位定时器/事件计数器00）一步脉冲输出	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
;*******************************************************************************
;16 位定时器 00（外部事件计数器）
;*******************************************************************************
;<<历史>>
;       2007.7.--            发布
;*******************************************************************************
;
;<<概述>>
;
; 本示例程序展示使用 16 位定时器 00 的外部事件计数器功能的示例。输入 TI000 引脚的外部时钟
; 脉冲的下降沿的每 10 次计数都会生成中断，导致 LED 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 8MHz
; - 设置供给外围硬件的时钟为 8MHz
;
;
; <16 位定时器 00 设置>
; - 工作模式：当 TM00 和 CR000 匹配时清零并启动定时器计数
; - 不进行定时器输出
; - 将计数时钟设置为 TI000 引脚的下降沿
; - 将计数器比较值设置为 10
;
;
;<<I/O 端口设置>>
;
; 输入：P30
; 输出：P00-P03, P20-P23, P31-P33, P40-P47, P120-P123, P130
; # 所有未使用的端口设置为输出模式。
;
```

```
;************************************************************************

;========================================================================
;
;       定义符号
;
;========================================================================
EVENTCNT    EQU     10              ; 定时器 00 外部事件计数器比较值

;========================================================================
;
;       向量表
;
;========================================================================
XVCTCSEG    AT      0000H
        DW      RESET_START         ;(00)   RESET
        DW      RESET_START         ;(02)   --
        DW      RESET_START         ;(04)   --
        DW      RESET_START         ;(06)   INTLVI
        DW      RESET_START         ;(08)   INTP0
        DW      RESET_START         ;(0A)   INTP1
        DW      RESET_START         ;(0C)   INTTMH1
        DW      INTERRUPT_TM000     ;(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

;========================================================================
;
;       定义存储器栈区
;
;========================================================================
XSTKDSEG    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          ; 将低压检测电压（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
;-----  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，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,     #00000000B      ; 设置 P20-P23 的输出锁存器为低
        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,    #11110001B      ; 设置 P31-P33 为输出模式、P30/TI000 为输入模式

;---------------------------------------------------------------------
;       初始化端口 4
;---------------------------------------------------------------------
        MOV     P4,     #00000001B      ; 设置 P41-P47 的输出锁存器为低，设置 P40 的输出锁存器为高（关闭
LED）
        MOV     PM4,    #00000000B      ; 设置 P40-P47 为输出模式

;---------------------------------------------------------------------
;       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

;---------------------------------------------------------------------
;       设置 16 位定时器 00
;---------------------------------------------------------------------
        MOV     CRC00, #00000000B       ; 用 CR000 作为比较寄存器
        MOVW    AX,                 #EVENTCNT-1    ; 设置比较值
        MOVW    CR000,  AX              ; 设置比较值
        MOV     PRM00, #00000011B       ; 将计数时钟设置为 TI000 引脚的下降沿
        MOV     TOC00, #00000000B       ; 不进行定时器输出
        MOV     TMC00, #00001100B       ; 启动定时器的运行（当 TM00 和 CR000 匹配时清零并启动）

;---------------------------------------------------------------------
;       设置中断
```

```
;-------------------------------------------------------------------------
        MOV     IF0,        #00H                    ; 事先清除无效中断请求
        CLR1    TMMK000                             ; 不屏蔽 INTTM000 中断

        EI                                          ; 启用向量中断

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

;*************************************************************************
;
;       中断 INTTM000
;
;*************************************************************************
INTERRUPT_TM000:
        XOR     P4,         #00000001B              ; LED 输出反相
        RETI                                        ; 从中断服务返回

end
```

● main.c (C 语言版)

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

        日电电子 78K0S/KB1+

*****************************************************************************
        78K0S/KB1+        示例程序
*****************************************************************************
16 位定时器 00（外部事件计数器）
*****************************************************************************
<<历史>>
        2007.7.--            发布
*****************************************************************************

<<概述>>

本示例程序展示使用 16 位定时器 00 的外部事件计数器功能的示例。输入 TI000 引脚的外部时钟脉冲的下降沿的每
10 次计数都会生成中断，导致 LED 输出反相。


 <主要设置内容>

 - 声明由中断运行的函数：INTTM000 -> fn_inttm000()
 - 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


 <16 位定时器 00 设置>
 - 工作模式：当 TM00 和 CR000 匹配时清零并启动定时器计数
 - 不进行定时器输出
 - 将计数时钟设置为 TI000 引脚的下降沿
 - 将计数器比较值设置为 10


<<I/O 端口设置>>

 输入：P30
 输出：P00-P03, P20-P23, P31-P33, P40-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)

==========================================================================*/
#pragma     SFR                                 /* 可在 C 源程序级描述 SFR 名称 */
#pragma     EI                                  /* 可在 C 源程序级描述 EI 指令 */
#pragma     NOP                                 /* 可在 C 源程序级描述 NOP 指令 */
#pragma interrupt INTTM000 fn_inttm000 /* 中断函数声明：INTTM000 */

#define eventCnt 10                     /* 定时器 00 外部事件计数器比较值 */

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

        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;     /* 将低压检查电压（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    = 0b00000000;             /* 设置 P20-P23 的输出锁存器为低 */
    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   = 0b11110001;             /* 设置 P31-P33 为输出模式、P30/TI000 为输入模式 */

  /*---------------------------------------------------------------------
    初始化端口 4
  ---------------------------------------------------------------------*/
    P4    = 0b00000001;             /* 设置 P41-P47 的输出锁存器为低，设置 P40 的输出锁存器为高（关闭
LED） */
    PM4   = 0b00000000;             /* 设置 P40-P47 为输出模式 */

  /*---------------------------------------------------------------------
    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 16 位定时器 00
  ---------------------------------------------------------------------*/
    CRC00 = 0b00000000;             /* 用 CR000 作为比较寄存器 */
    CR000 = eventCnt-1;             /* 初始化比较值 */
    PRM00 = 0b00000011;             /* 将计数时钟设置为 TI000 引脚的下降沿 */
    TOC00 = 0b00000000;             /* 不进行定时器输出 */
    TMC00 = 0b00001100;             /* 启动定时器的运行（当 TM00 和 CR000 匹配时清零并启动） */
```

```
/*---------------------------------------------------------------------------
        设置中断
---------------------------------------------------------------------------*/
        IF0   = 0x00;                       /* 事先清除无效中断请求 */
        TMMK000 = 0;                        /* INTTM000 中断去屏蔽 */

        return;
}

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

        主循环

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

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

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

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

        中断 INTTM000

*****************************************************************************/
__interrupt void fn_inttm000(){

        P4 ^= 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 引脚用作 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
```

附录B 修订记录

版本	出版日期	页码	修订
第一版	2008年2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