

Description

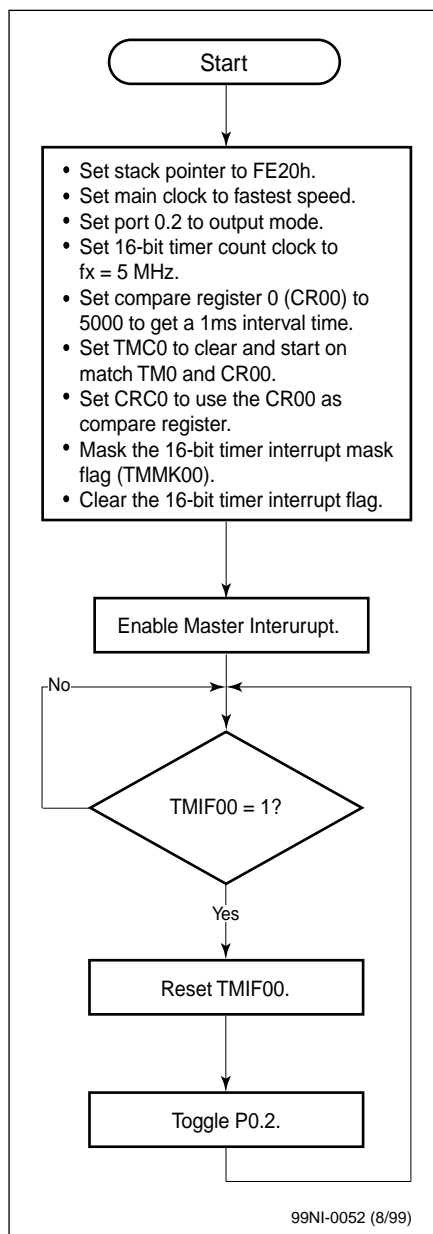
The 16-bit timer/event counter (TM0) in the μ PD7805x/78005x subseries can be used as an interval timer, external event counter, pulse-width modulator output, square-wave output, one-shot pulse output or for pulse-width measurement.

This program demonstrates the 16-bit timer/event counter in interval timer mode. When the count value of the 16-bit timer register (TM0) matches the value set to capture/compare register 0 (CR00), counting continues with the TM0 value cleared to 0 and the interrupt request signal (INTTM00) generated. This sets the interrupt flag TMIF00 in the interrupt request flag register (IF0H). Each time the program detects the flag set to one, it toggles port 0.2 and clears the flag.

Program Specifications

- ☐ Timer count clock: 5 MHz
- ☐ Interval time: 1 ms (2 ms period)
- ☐ Square wave frequency: 500 Hz
- ☐ Interrupt handling: polling the interrupt request flag TMIF00
- ☐ Pins used in program: P02/INTP2 (toggles every 1 ms)

Flowchart



Assembly Language Program

```

;*****
; Date:      07/19/1999
;
; Parameters: - fastest CPU clock
;              (fx=5.00 MHz, 1 CPU clock cycle = 200 ns)
;              - interval time: 1 ms
;              - Count clock:   fx = 5 MHz
;              - Unmask the 16-bit timer interrupt mask bit for
;                  polling the timer interrupt flag
;              - Port 0.2 toggles every 1 ms
;
;*****

;=====
;=      Specify Interrupt Vectors      =
;=====

Res_Vec CSEG AT 0000h          ; Set main program start vector
      DW  Start

;=====
;      Main Program                  =
;=====

MAIN      CSEG
Start:    DI                      ; Disable interrupts
      MOVW      AX, #0FE20h      ; Load SP address
      MOVW      SP, AX          ; Set Stack Pointer
      MOV       OSMS,#01h        ; Don't use scaler
      MOV       PCC, #00h        ; Main system clock at fastest setting
      CLRL      P0.2            ; Latch port 0.2 low
      CLRL      PM0.2           ; Set port 0.2 as output
      MOV       TCL0,#020h       ; Select counter clock to fx = 5 MHz
      MOVW      CR00,#5000       ; Set compare register to 5000 for 1 ms interval
      MOV       TMC0,#0Ch        ; Set to clear and start of match TM0 and CR00
      CLRL      CRC0.0          ; Set CR00 operation to compare register
      SETL      TMMK00           ; Mask the 16-bit timer interrupt mask bit
      CLRL      TMIF00          ; Clear 16-bit timer interrupt flag
      EI                      ; Enable interrupts
Loop:     BF      TMIF00,$$       ; Branch if timer interrupt flag is high
      CLRL      TMIF00          ; Clear timer interrupt flag
      XOR       P0,#04h         ; Toggle port 0.2
      BR        Loop           ; Loop back
      END

```

C Language Program

```

/*****
; Date:          07/19/1999
;
; Parameters: - fastest CPU clock
;              (fx = 5.00 MHz, 1 CPU clock cycle = 200 ns)
;              - interval time: 1 ms
;              - Count clock:   fx = 5 MHz
;              - Unmask the 16-bit timer interrupt mask bit for
;                  polling the timer interrupt flag
;              - Port 0.2 toggles every 1 ms
;
; *****/
/* extension functions in K0/K0S compiler */
#pragma sfr /* key word to allow SFR names in C code */
#pragma asm /* key word to allow ASM statements in C code */
#pragma DI /* key word for DI instruction in C code */
#pragma EI /* key word for EI instruction in C code */
/*=====
; Constants/Variables =
;=====*/

#define TRUE      1
#define FALSE     0

/*=====
; Main Program =
;=====*/
void main(void)
{
    OSMS = 0x01; /* Don't use scaler */
    PCC = 0x00; /* Main system clock at fastest setting */
    P0.2 = 0; /* Latch port 0.2 low */
    PM0.2 = 0; /* Set port 0.2 as output */
    TCL0 = 0x20; /* Select counter clock to fx = 5 MHz */
    CR00 = 5000; /* Set compare register to 5000 for a 1 ms interval */
    TMC0 = 0x0c; /* Set to clear and start on match of TM0 and CR00 */
    CRC0.0 = 0; /* Set CR00 operation to compare register */
    TMMK00 = 1; /* Mask the 16-bit timer interrupt mask bit */
    TMIF00 = 0; /* Clear timer interrupt flag */
    EI(); /* Enable interrupts */
    while(TRUE) /* while loop*/
    {
        while( !TMIF00 ); /* Wait until timer interrupt flag is high */
        TMIF00 = 0; /* Clear timer interrupt flag */
        P0 ^= 0x04; /* toggle port 0.2 */
    } /* end of while loop */
} /* end of function main() */

```



For literature, call **1-800-366-9782** 7 a.m. to 6 p.m. Pacific time
or FAX your request to **1-800-729-9288**
or visit our web site at **www.necel.com**

In North America: No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Electronics Inc. (NECEL). The information in this document is subject to change without notice. All devices sold by NECEL are covered by the provisions appearing in NECEL Terms and Conditions of Sales only. Including the limitation of liability, warranty, and patent provisions. NECEL makes no warranty, express, statutory, implied or by description, regarding information set forth herein or regarding the freedom of the described devices from patent infringement. NECEL assumes no responsibility for any errors that may appear in this document. NECEL makes no commitments to update or to keep current information contained in this document. The devices listed in this document are not suitable for use in applications such as, but not limited to, aircraft control systems, aerospace equipment, submarine cables, nuclear reactor control systems, and life support systems. "Standard" quality grade devices are recommended for computers, office equipment, communication equipment, test and measurement equipment, machine tools, industrial robots, audio and visual equipment, and other consumer products. For automotive and transportation equipment, traffic control systems, anti-disaster and anti-crime systems, it is recommended that the customer contact the responsible NECEL salesperson to determine the reliability requirements for any such application and any cost adder. NECEL does not recommend or approve use of any of its products in life support devices or systems or in any application where failure could result in injury or death. If customers wish to use NECEL devices in applications not intended by NECEL, customer must contact the responsible NECEL salespeople to determine NECEL's willingness to support a given application.