

Description

The watchdog timer (WDT) in the μ PD7805x/78005x subseries can be used in watchdog timer mode or interval timer mode.

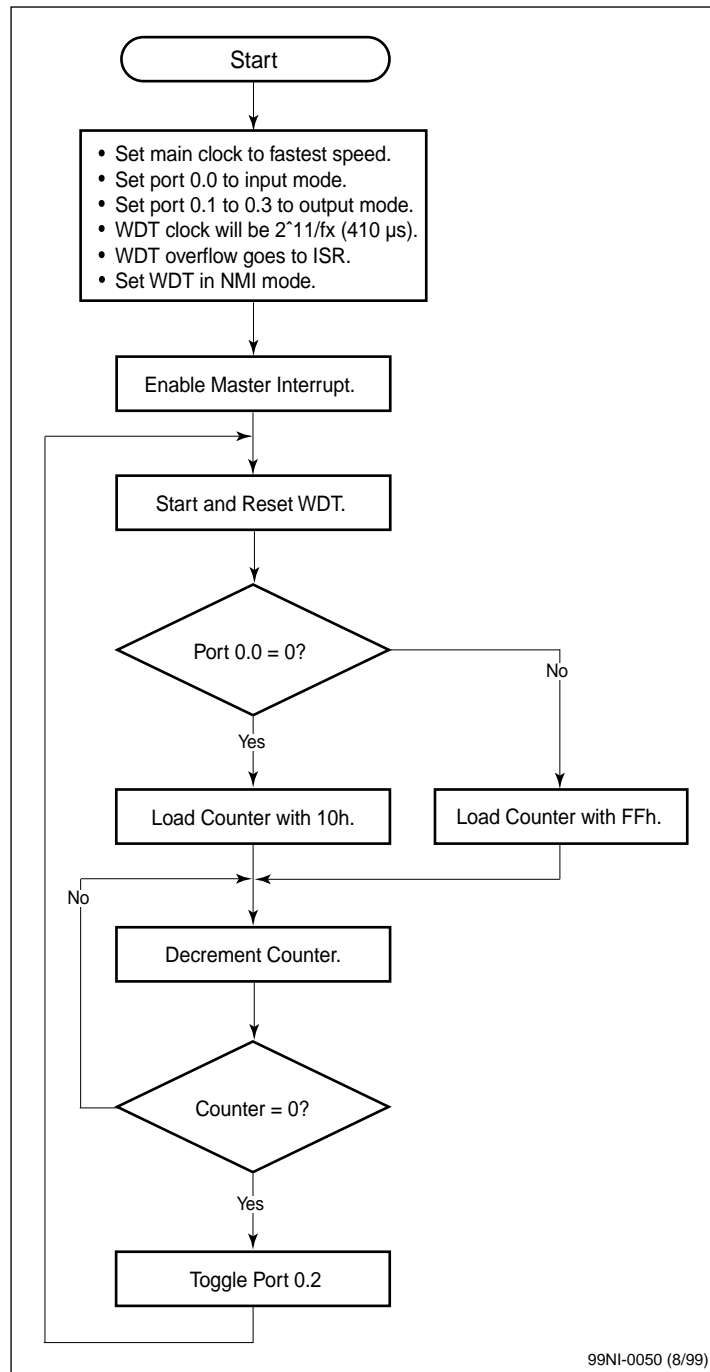
This program demonstrates the WDT in watchdog timer mode 1, where a non-maskable interrupt (NMI) request occurs upon generation of a counter overflow. For demonstration purposes, this program shows two methods of operation. The B register (used as a counter) is set with either 10H or FFH and decrements until it reaches zero. If B is set to 10H, the value of the B register reaches zero before the watchdog timer times out. The program toggles port 0 bit 2, retriggers the watchdog timer, and restarts B counting from 10H. If B is set to FFH, the watchdog timer times out before B reaches zero and generates an interrupt request. In this case, the program continuously toggles port 0 bit 3.

If port 0 bit 0 is set to ground, B is set to 10H and the watchdog timer never times out. If port 0 bit 0 is set to a logical one, B is set to FFH and the watchdog timer always times out.

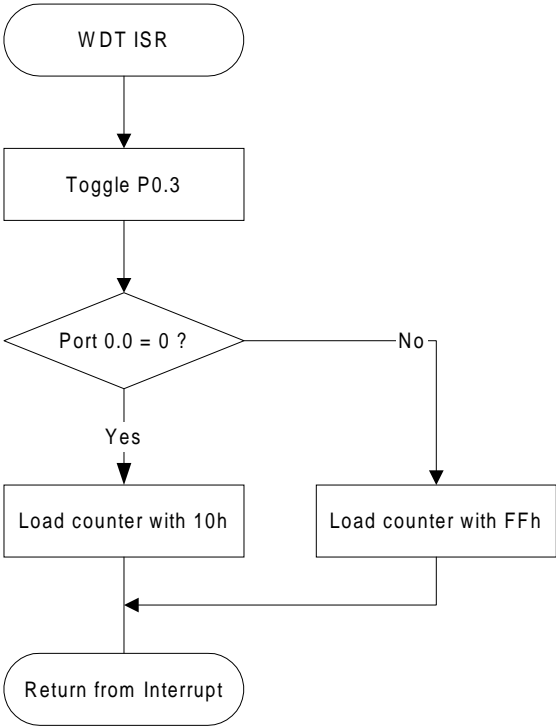
Program Specifications

- ❑ WDT count clock: $f_{\text{fx}}/2^3 = 625 \text{ kHz}$ at 5 MHz main system clock
- ❑ Toggle interval for port 0.2
 - $t = 33.8 \mu\text{s}$ (Assembly language)
 - $t = 85.8 \mu\text{s}$ (C language)
- ❑ Toggle interval for port 0.3
 - $t = 422 \mu\text{s}$ (Assembly language)
 - $t = 434 \mu\text{s}$ (C language)
- ❑ Frequency for port 0.2
 - $f = 14.79 \text{ kHz}$ (Assembly language)
 - $f = 5.83 \text{ kHz}$ (C language)
- ❑ Frequency for port 0.3
 - $f = 1185 \text{ Hz}$ (Assembly language)
 - $f = 1152 \text{ Hz}$ (C language)
- ❑ Pins used in program
 - P00/INTP0/TI00: input selects the counter value to load
 - P02/INTP2: toggles if B is loaded with 10H
 - P03/INTP3: toggles if B is loaded with FFH

Flowchart 1



Flowchart 2



Assembly Language Program

```

*****
; Date:          07/15/1999
;
; Parameters: - fastest CPU clock
;              (fx=5.00 MHz; 1 CPU clock cycle = 200 ns)
;              - WDT interval time is 2^11/fx (410 µs)
;              - use WDT in non-maskable interrupt mode
;              - input port 0.0 = 0: Reset WDT periodically
;              - input port 0.0 = 1: WDT Interrupt
;              - main routine toggles P0.2 (output)
;                  (Port 0.2 toggles every 33.8 µs at fx = 5.00 MHz
;                  when the low value (10h) is loaded.
;              - WDT-ISR toggles P0.3 (output)
;                  (Port 0.3 toggles every period is 422 µs at fx = 5.00 MHz
;                  when the high value (FFh) is loaded.
*****

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

Res_Vec  CSEG AT 0000h                ; Set main program start vector.
        DW  Start
WDT_Vec  ORG 0004h                    ; Set interrupt vector for watchdog timer
        DW  WDT_ISR

;=====
;=      Constants/Variables            =
;=====

LoVal    EQU 10h                      ; Low counter value
HiVal    EQU 0FFh                     ; High counter value

;=====
;=      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
        MOV     P0, #00h              ; Reset port 0 latch
        MOV     PM0, #01h             ; P0.0 is an input; P0.1-P0.3 are outputs
        MOV     TCL2, #00h            ; WDT clock will be 2^11/fx (410 µs)
        MOV     WDTM, #10h            ; Set WDT into overflow and NMI mode
        EI                            ; Enable interrupts
Loop1:   SETI    RUN                   ; Start watchdog timer
        BT      P0.0, $HiLoad         ; Branch if P0.0 = 1
        MOV     B, #LoVal             ; Load low value into B register
        BR     $Loop2                 ; Branch to continue
HiLoad:  MOV     B, #HiVal             ; Load high value into B register
; a loop cycles = 10 clocks
; (2.0 µs at fx=5 MHz)
Loop2:   NOP
        DBNZ    B, $Loop2             ; Delay loop

```

```
        XOR        P0,#04h          ; Toggle port 0.2
        BR         $Loop1          ; Branch back to Loop1

;=====
;=      Interrupt Routine          =
;=====
ISR      CSEG
WDT_ISR: XOR        P0,#08h          ; Toggle Port 0.3
        BT         P0.0, $High10    ; Branch if P0.0 = 1
        MOV        B, #LoVal        ; Load low value into B register
        RETI       ; Return from interrupt
High10:  MOV        B, #HiVal        ; Load high value into B register
        RETI       ; Return from interrupt

        END
```

C Language Program

```

/*****
; Date:          07/15/1999
;
; Parameters:-  fastest CPU clock
;               (fx=5.00 MHz; 1 CPU clock cycle = 200 ns)
;               - WDT interval time is 2^11/fx (410 µs)
;               - use WDT in non-maskable interrupt mode
;               - input port 0.0 = 0 : Reset WDT periodically
;               - input port 0.0 = 1 : WDT Interrupt
;               - main routine toggles P0.2 (output)
;                 (Port 0.2 toggle period is 85.8 µs at fx = 5.00 MHz
;                 when the low value (10h) is loaded.
;               - WDT-ISR toggles P0.3 (output)
;                 (Port 0.3 toggle period is 434 µs at fx = 5.00 MHz
;                 when the high value (FFh) is loaded.
;*****/
/* extension functions in K0/K0S compiler */
#pragma sfr          /* key word to allow SFR names in C code */
#pragma HALT        /* key word for HALT instruction in C code */
#pragma STOP        /* key word for STOP instruction in C code */
#pragma NOP         /* key word for NOP instruction in C code */
#pragma DI          /* key word for DI instruction in C code */
#pragma EI          /* key word for EI instruction in C code */

/*=====
;      Specify Interrupt Vectors      =
;=====*/

/* Set interrupt vector for the Watchdog timer */
#pragma interrupt INTWDT WDT_ISR

/*;=====
;      Constants/Variables      =
;=====*/

#define TRUE        1
#define FALSE       0
#define LoVal       0x10          /* low counter value to reset WDT */
#define HiVal       0xff          /* High counter value to time out WDT */
unsigned char value;             /* counter value in high speed RAM*/

/*=====
;      Main Program      =
;=====*/

void main(void)
{
    DI();                        /* Disable interrupts */
    OSMS = 0x01;                 /* Don't use scaler */
    PCC = 0x00;                 /* Main system clock at fastest setting */
    P0 = 0x00;                 /* Latch port 0 low */
    PM0 = 0x01;                 /* Set P0.0 an input, P0.1-P0.3 outputs */
    TCL2 = 0x00;                 /* interval timer = 2^11/fx(410 µs) */
    WDTM = 0x10;                 /* Set WDT into overflow and NMI mode */
    EI();                        /* Enable interrupts */
    while(TRUE)

```

```

    {
        RUN = 1;                /* Start watchdog timer */
        if(P0.0 == 1)           /* Test P0.0 state */
            value = HiVal;     /* load high value if P0.0 = HIGH */
        else
            value = LoVal;     /* load low value if P0.0 = LOW */
        while( TRUE )         /* loop until value reaches 0 */
        {
            NOP();            /* Execute NOP instruction */
            value--;          /* Decrement the value by 1 */
            if(value == 0)    /* if the value reaches 0 then exit */
                break;       /* exit from while loop */
        }                    /* end of while loop */
        P0 ^= 0x04;          /* toggle P0.2 */
    }                          /* end of while loop */
}                              /* end of function main() */

/*=====
;      Interrupt Routine      =
;=====*/

void WDT_ISR(void)
{
    P0 ^= 0x08;              /* toggle P0.3 */
    if(P0.0 == 1)           /* Test P0.0 state */
        value = HiVal;     /* Load high value if P0.0 = HIGH */
    else
        value = LoVal;     /* load low value if P0.0 = LOW */
}
/* end of WDT_ISR */
-

```



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.