

### *On-Chip Peripheral Program Example*

August 1999

---

#### **Description**

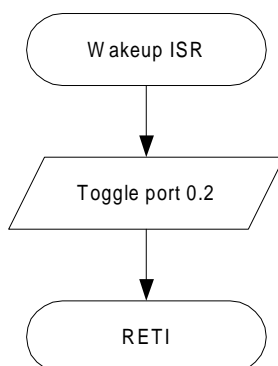
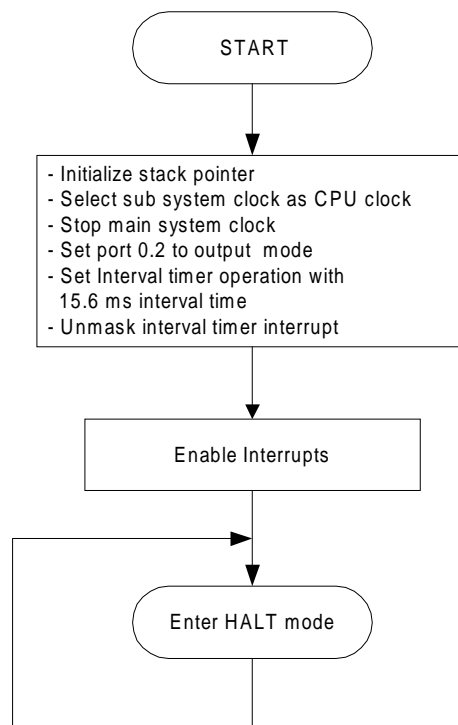
Halt mode is one of the standby functions on the 7805x/78005x subseries and is used to reduce CPU power consumption. The execution of the HALT instruction sets the HALT mode. The HALT mode stops the CPU operation clock, however, system clock oscillator continues. The HALT mode is exited upon any interrupt request. If interrupt handling is enabled (EI), the wake up starts executing the code in the ISR. In case interrupt handling is disabled (DI), the next instruction after the HALT instruction is executed.

In this program, the interval timer (part of the watch timer) generates an interrupt every 15.6 ms. This interrupt wakes up the microcontroller and toggles a port in the interrupt service routine (ISR). After returning from the ISR, the micro enters the HALT mode again.

#### **Program Specifications**

- ❑ CPU runs from sub system clock ( $f_x = 32.768 \text{ kHz}$ )
- ❑ CPU wakes-up every 15.6 ms for toggling a port pin
- ❑ Pins used in program: P02/INTP2 (toggles every 15.6 ms)

## Flowchart



### Assembly Language Program

```

;*****
; Date:      08/24/1999
;
; Parameters: - CPU clock is subsystem clock,
;              (fx = 32.768 kHz, 1 CPU clock cycle = 61.04  $\mu$ s)
;              - Interval timer clock source:  subsystem clock (32.768 kHz)
;              - Interval time:                15.6 ms
;              - Port 0.2 toggles each time after wakeup (15.6 ms)
;*****

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

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

        ORG      001Eh
        DW      INTER_ISR            ; Interval timer interrupt vector

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

MAIN      CSEG
Start:    DI                          ; Disable interrupts
        MOVW     AX, #0FE20h          ; Load SP address
        MOVW     SP, AX              ; Set Stack Pointer
        MOV      PCC, #30h           ; Use sub system clock as CPU clock
        SET1     MCC                 ; Stop main system clock
        CLR1     P0.2                ; Latch port 0.2 low
        CLR1     PM0.2               ; Set port 0.2 to output mode
        MOV      TCL2, #10h          ; Select counter clock to fxt = 32.768 kHz
        MOV      TMC2, #56h          ; Set TMC2 register to:
        ; - Interval timer operation enable
        ; - 15.6 ms interval time
        CLR1     TMMK3               ; Unmask the interval timer interrupt
        EI                          ; Enable interrupts

Loop:     NOP
        HALT                          ; Enter sub-Halt mode (standby mode)
        NOP
        BR       Loop                ; Branch back

;=====
;      Watch timer ISR                  =
;=====

        INTER_ISR:
        XOR             P0, #04h      ; Toggle port 0.2 to indicate system wakeup
        RETI

        END

```

## C Language Program

```

/*****
; Date:          08/24/1999
;
; Parameters: - CPU clock is subsystem clock,
;              (fx = 32.768 kHz, 1 CPU clock cycle = 61.04  $\mu$ s)
;              - Interval timer clock source:  subsystem clock (32.768 kHz)
;              - Interval time:                15.6 ms
;              - Port 0.2 toggles each time after wakeup (15.6 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 HALT          /* key word for HALT 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 interval timer */

#pragma interrupt INTTM3 INTER_ISR

/*=====
;      Main Program                  =
;=====*/
void main(void)
{
    DI();                /* Disable interrupts */
    PCC = 0x30;           /* Use sub system clock as CPU clock */
    MCC = 1;             /* Stop main system clock */
    P0.2 = 0;            /* Latch port 0.2 to low */
    PM0.2 = 0;           /* Set port 0.2 to output mode */
    TCL2 = 0x10;         /* Select counter clock to 32.768 kHz */
    TMC2 = 0x56;         /* Set TMC2 register to:
                        - Interval timer operation enable
                        - 15.6 ms interval time */

    TMMK3= 0;            /* Unmask the interval timer interrupt */
    EI();                /* Enable interrupts */
    while(1)
    {
        NOP();
        HALT();          /* Enter sub HALT mode (standby mode) */
        NOP();
    }
    /* end of while loop */
}
/* end of function main() */

/*=====
;=      Interval timer ISR          =
;=====*/
void INTER_ISR(void)
{
    P0 ^= 0x04;          /* Toggle port 0.2 to indicate system wakeup */
}

```



---

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](http://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.