Precautions in Dynamic Updating Timer Data & PWM Duty for AT8 Series

Description: The guidelines for dynamic updating Timer Data and PWM Duty of AT8 series.

Reason: To ensure the PWM output as expected when dynamic updating PWM frequency, Timer Data and PWM Duty should be updated only when Timer overflow occurs.

Solution: The procedure to dynamic update Timer Data and PWM Duty of AT8 series are described below.

1. Set the initial value of Timer Data and PWM Duty. The value of PWM Duty must be smaller than Timer Data.
2. Enable the Timer interrupt.
3. Update Timer Data and PWM Duty in the interrupt service routine. Please handle the overflow flag of Timer first when using multiple interrupts.

Example 1: ASM example code

```
V_MAIN:
        movia   0x00
        iost    IOSTB                           ; Set PortB as output
        disi                                    ; Disable all interrupts
        movia   0x00
        movar   TMRH
        movia   0xFF
        sfun    TMR1                            ; Load 0xFF to TMR1 (Timer1[9:0]=0x0FF)
        movia   0x80
        sfun    PWM1DUTY                        ; Load 0x80 to PWM1DUTY LB register ( PWM1DUTY[9:0]=0x080 )

        movia   C_PWM1_En | C_TMR1_Reload | C_TMR1_En
        sfun    T1CR1                           ; Enable Timer1, Initial value reloaded from TMR1, Non-stop mode
        movia   C_TMR1_ClkSrc_Inst
        sfun    T1CR2                           ; Timer1 clock source = instruction clock

        movia   C_INT_TMR1
        movar   INTE                            ; Enable Timer1 overflow interrupt
        eni                                     ; Enable all unmasked interrupts
;-------------------------------------------------------------------------
L_MAIN_LOOP:
        clrwdt
        fgoto   L_MAIN_LOOP
;-------------------------------------------------------------------------
V_INT:
        movar   R_AccBuf                        ; Store ACC value
        swapr   R_AccBuf,C_SaveToReg
        movr    STATUS,C_SaveToAcc
        movar   R_StatusBuf                     ; Store STATUS value
;-------------------------------------------------------------
L_TIME1_INT:
        btrss   INTF,C_INT_TMR1_Bit             ; Skip next instruction, if T1IF=1
        lgoto   L_RET2Main

        movia   0x01
        xorar   PORTB,1                         ; PB0 Toggle
        movia   ~C_INT_TMR1
        movar   INTF                            ; Clear T1IF (Timer1 overflow interrupt flag bit)

        movia   0x00
        movar   TMRH
        movia   0x80
        sfun    TMR1                            ; Load 0x80 to TMR1 (Timer1[9:0]=0x0FF)
        movia   0x40
        sfun    PWM1DUTY                        ; Load 0x40 to PWM1DUTY LB register ( PWM1DUTY[9:0]=0x080 )
L_RET2Main:
        movr    R_StatusBuf,C_SaveToAcc
        movar   STATUS                          ; Restore STATUS value
        swapr   R_AccBuf,C_SaveToAcc            ; Restore ACC value
        retie                                   ; Return from interrupt and enable interrupt globally
```

Example 2: C example code

```
void main(void)
{
    IOSTB = 0;                                          // Set PortB as output

    DISI();                                             // Disable all interrupts

    TMRH = 0;
    TMR1 = 0xFF;                                        // Load 0xFF to TMR1 (Timer1[9:0]=0x0FF)
    PWM1DUTY = 0x80;                                    // Load 0x80 to PWM1DUTY LB register ( PWM1DUTY[9:0]=0x080 )

    T1CR1 = C_PWM1_En | C_TMR1_Reload | C_TMR1_En;      // Enable Timer1, initial value reloaded from TMR1, Non-stop mode
    T1CR2 = C_TMR1_ClkSrc_Inst;                         // Timer1 clock source = instruction clock

    INTE = C_INT_TMR1;                                  // Enable Timer1 overflow interrupt
    ENI();                                              // Enable all unmasked interrupts

    while(1)
    {
        CLRWDT();
    }
}

//interrupt service routine
void isr(void) __interrupt(0)
{
    if(INTFbits.T1IF)
    {
        PORTB ^= 1;                                     // PB0 Toggle
        INTF= (unsigned char)~(C_INT_TMR1);             // Clear T1IF flag bit

        TMRH = 0;
        TMR1 = 0x80;                                    // Update 0x80 to TMR1 (Timer1[9:0]=0x080)
        PWM1DUTY = 0x40;                                // Update 0x40 to PWM1DUTY LB register ( PWM1DUTY[9:0]=0x040 )
    }
}
```

This applied to below listed IC Body :

1. **AT8A series: AT8A513F / AT8A513G / AT8A513H / AT8A52E / AT8A53E / AT8A54A / AT8A54E / AT8AE513F**
2. **AT8B series: AT8B60D / AT8B62A / AT8B62F / AT8BM62D / AT8BE62D / AT8BE64A .**
3. **AT8T series: AT8TM52D / AT8TE64A.**