Modificarea codului de test pentru lucrul cu simulatorul CVAVR si Astudio

1. Se vor comenta liniile de cod in init.c (se inhiba watchdog-ul)

```
/*
#asm("wdr")
// Write 2 consecutive values to enable watchdog
// this is NOT a mistake !
WDTCSR=0x18;
WDTCSR=0x08;
*/
```

2. Se modifica frecventa intreruperilor pentru Timer1 (in init.c)

```
//OCR1AH=0x4C;
//OCR1AL=0x40;

OCR1AH=0x00;
OCR1AL=0x40;
```

3. se comenteaza bucla infinita (in main.c)

```
void main (void)
{
unsigned char temp,i;

        Init_initController();  // this must be the first "init" action/call!
        #asm("sei")          // enable interrupts
        LED1 = 1;            // initial state, will be changed by timer 1

        while(TRUE)
        {
         }

        /*
                wdogtrig();                 // call often else processor will reset
                if(rx_counter0)     // if a character is available on serial port USART0
                {
                        temp = getchar();
                        if(temp == '?')
                        printf("\r\nSwVersion:%d.%d\r\n", SW_VERSION/10, SW_VERSION%10);
                        else
                        putchar(temp+1);              // echo back the character + 1 ("a" becomes "b", etc)
                }

                 if(SW1 == 0)       // pressed
                  {
                 delay_ms(30);   // debounce switch
                         if(SW1 == 0)
                        {                    // LED will blink slow or fast
                        while(SW1==0)
```

```
                                wdogtrig();         // wait for release
            // alternate between values and values/4 for OCR1A register
            // 4C40H / 4 = 1310H
            // new frequency = old frequency * 4
            if(OCR1AH == 0x4C)
               {TCNT1H=0; TCNT1L=0; OCR1AH = 0x13; OCR1AL = 0x10;}
            else
               {TCNT1H=0; TCNT1L=0; OCR1AH = 0x4C; OCR1AL = 0x40;}
          }
       }

       // measure time intervals on oscilloscope connected to pin TESTP
       for(i=0; i<3; i++) {
          TESTP = 1;
          delay_us(1);
          TESTP = 0;   // may check accuracy of 1us interval on oscilloscope
       }
    }
    */

}// end main loop
```

4. Se modifica rutina de intreruperi Timer1, adaugindu-se functia aplicatiei definite de tema – **MyApplication()** in main.c

```
/*
 * Timer 1 Output Compare A interrupt is used to blink LED
 */
interrupt [TIM1_COMPA] void timer1_compa_isr(void)
{
LED1 = ~LED1; // invert LED
// se adauga functia proprie

MyApplication();
}
```

Functia **MyApplication()** se va scrie in fisierul main.c.

Se compileaza cu CVAVR si se simuleaza cu Astudio – se va alege procesorul **ATMega 164P** (pentru a se putea simula intreruperile).

Se pune un breakpoint la linia **MyApplication()** si se simuleaza (depaneaza) aplicatia.

Pentru testarea reala se decomenteaza toate liniile comentate anterior.