```
/*************************************************************
 Target MCU & clock speed: ATtiny13A @ 1.2Mhz internal
 Name: main.c
    Project folder: ISR-Bathroom LED-23189812
 C modules of this project:
    Nothing
 Custom Headers:
    Nothing
 Author  : Insoo Kim (insoo@hotmail.com)
 Created : Sep 26, 2018
 Updated : Sep 29, 2018 (On Atmel Studio 7)

 Description:
    In night, turning on bathroom light is too bright to do one's need.
        So I've used two white LEDs of normal 5mm diameter
        with 510 ohm resistors, and just enough to see toilet crown.
        Here I prepare HW using my first PCB, and code using PCINT & WDT isr.

    ATtiny13A controls LEDs based on the composite input of
        PIR motion sensor and CDS-5 light sensor.

    PIR sensor triggers PCINT3 on PB3, and if sufficiently dark
        by reading CDS-5 thru ADC, then two LEDs will lit up
        for defined period (2 minute as of Sep 29, 2018)

 HEX size[Byte]: 332 out of 1024

How to upload to the target MCU
<For Windows Atmel Studio>
Select Tool - USBtiny
(USBtiny menu should be configured in the external tool menu)

<For CMD window or DOS prompt>
cd " C:₩Users₩Insoo₩Documents₩GitHub₩ATmelStudio₩ATtiny13A₩ClockGen₩ISR-Bathroom LED-23189812₩Debug "
avrdude -c usbtiny -P usb -p attiny13 -U flash:w:ISR-Bathroom LED.hex:i

 Ref: PIR time duration & sensitivity control
 http://qqtrading.com.my/pir-motion-sensor-module-hc-sr501
 Software reset
 https://www.microchip.com/webdoc/AVRLibcReferenceManual/FAQ_1faq_softreset.html
 https://www.avrfreaks.net/forum/software-reset-6
 https://www.avrfreaks.net/forum/software-reset-avr-gcc
 Delay
  ATtiny13 - using delay() function
  https://forum.arduino.cc/index.php?topic=500256.0
  Time limit on delay()?
  http://forum.arduino.cc/index.php?topic=115473.0
  AVR® ADC Noise Reduction Mode
  http://microchipdeveloper.com/8avr:adcnoisereduce
  Wake ADC after interrupting sleep mode
  https://forum.arduino.cc/index.php?topic=197568.0
  Power saving techniques for microprocessors
  https://www.gammon.com.au/power
*************************************************************/
```

```
55  #include <avr/io.h>
56  #include <util/delay.h>
57  #include <avr/interrupt.h>
58  #include <avr/sleep.h>
59  #include <avr/wdt.h>
60
61  // ATtiny13A port usage
62  #define LED0 PB0
63  #define LED1 PB1
64  #define CDS5 PB2
65  #define PIR  PB3
66  #define PIR_INT  PCINT3
67
68  // AVR software reset macro
69  #define soft_reset()        ₩
70  do                          ₩
71  {                           ₩
72      wdt_enable(WDTO_15MS);  ₩
73      for(;;)                 ₩
74      {                       ₩
75      }                       ₩
76  } while(0)
77
78  // Watch Dog Timer period [sec]
79  //#define UNIT_DELAY_WDT 1
80  //#define UNIT_DELAY_WDT 4
81  #define UNIT_DELAY_WDT 8
82
83  // accumulated WDT period [EA] to last LED ON.
84  // total duration of LED ON [sec] = UNIT_DELAY_WDT * SET_DELAY_UNIT
85  //#define SET_DELAY_UNIT 3
86  #define SET_DELAY_UNIT 15
87
88  // turning LED ON threshold of ADC output of CDS-5 voltage dividor
89  #define CDS5_LIGHT_THRESHOLD 50 //0 to 254
90
91  uint8_t i;
92  int adc_result;
93  //uint8_t isrCount=0;
94  uint8_t LEDstatus=0;
95
96  uint8_t WDTtick=0;
97
98  //-------- function prototypes --------
99  void adc_setup (void);
100 void systemInit(void);
101 void WDTsetup(void);
102 void toggleLED0(void);
103 void testADC(void);
104 void blinkLEDcnt(uint8_t );
105 void checkAmbientLight(void);
106
107 //-------- interrupt service routines --------
108 ISR(WDT_vect)
109 {
110     // --------- INCREASE WDT TICK COUNT ----------------------
```

```
111     // increase WDTtick every UNIT_DELAY_WDT sec
112     ++WDTtick;
113     //toggleLED0();
114     //testADC();
115 
116     // On the 1st SET_DELAY_UNIT (2 min), reset WDTtick count
117     //      and stop WDT.
118     if ( (WDTtick >= SET_DELAY_UNIT) && (LEDstatus == 1) )
119     {
120 
121         WDTtick = 0; // Reset WDT counter value
122         WDTCR &= ~(1<<WDTIE); // Disable watchdog timer interrupts
123         PORTB &= ~(1<<LED0) & ~(1<<LED1); // off LEDs
124         LEDstatus = 0; // make LEDstatus updated as 0
125         //soft_reset();
126     }//if (WDTtick >= SET_DELAY_UNIT)
127 
128 }//ISR(WDT_vect)
129 
130 ISR(PCINT0_vect)
131 {
132     //GIMSK &= ~(1<<PCIE); // disable PCINT interrupt
133 
134     //isrCount++;
135     //if ((isrCount % 2) == 1)
136     //if ( ((isrCount % 2) == 1) || ((isrCount % 2) == 0) )
137     {
138         //toggleLED0();
139         adc_setup();
140         checkAmbientLight();
141     //GIMSK |= (1<<PCIE); // enable PCINT interrupt
142     }
143 
144 }//ISR(PCINT0_vect)
145 
146 int main(void)
147 {
148     systemInit();
149     WDTsetup();
150     adc_setup();
151 
152     // Use the Power Down sleep mode
153     set_sleep_mode(SLEEP_MODE_PWR_DOWN);
154 
155     // ADC noise reduction sleep mode
156     //set_sleep_mode(SLEEP_MODE_ADC);
157 
158 
159     while (1)
160     {
161         // go to sleep and wait for interrupt...
162         // 33 uA as of Sep 27, 2018 when sleep
163         sleep_mode();
164 
165     }//while (1)
166 }//main
```

```
167
168 void checkAmbientLight(void)
169 {
170     adc_result = ADCH;
171
172     // Start the next conversion
173     ADCSRA |= (1 << ADSC);
174
175     //if LED is off
176     if (LEDstatus == 0)
177     {
178         //and if ambient light is dark enough
179         if (adc_result < CDS5_LIGHT_THRESHOLD)
180         {
181             WDTtick = 0; // Reset WDT counter value
182             WDTCR |= (1<<WDTIE); // Enable watchdog timer interrupts
183             PORTB |= (1<<LED0) | (1<<LED1); // turn on LEDs
184             LEDstatus = 1; // make LEDstatus updated as 1
185         }//if (adc_result < CDS5_LIGHT_THRESHOLD)
186     }//if (LEDstatus == 0)
187 }//checkAmbientLight
188
189 void testADC(void)
190 {
191     adc_result = ADCH;
192
193     // Start the next conversion
194     ADCSRA |= (1 << ADSC);
195
196     if (adc_result < 20)
197         blinkLEDcnt(1);
198     else if (adc_result < 50)
199         blinkLEDcnt(2);
200     else if (adc_result < 100)
201         blinkLEDcnt(3);
202     else
203         blinkLEDcnt(4);
204
205 }//testADC
206
207 //Not so perfect if using WDT isr with _delay_ms
208 //  for maybe there's only one timer circuit in ATtiny13a
209 void blinkLEDcnt(uint8_t num)
210 {
211     uint8_t i;
212
213     for(i=0; i<num; i++)
214     {
215         PORTB ^= (1<<LED0); // toggle on/off LED
216         _delay_ms(150);
217     }
218     PORTB &= ~(1<<LED0); //off LED
219     _delay_ms(500);
220 }//blinkLEDcnt
221
222 void toggleLED0(void)
```

```
223 {
224     PORTB ^= (1<<LED0);
225 }//toggleLED0
226 
227 void systemInit(void)
228 {
229     DDRB |= (1<<LED0) | (1<<LED1); //output for LEDs
230     PORTB &= ~(1<<LED0) & ~(1<<LED1);
231     DDRB &= ~(1<<PIR); //input for PIR motion sensor
232 
233     MCUCR &= ~(1<<ISC01) | ~(1<<ISC00); // Trigger INT0 on rising edge
234     PCMSK |= (1<<PIR_INT);   // pin change mask: listen to portb, pin PB3
235     GIMSK |= (1<<PCIE); // enable PCINT interrupt
236     sei();          // enable all interrupts
237 }//systemInit
238 
239 void WDTsetup(void)
240 {
241     // set timer prescaler to UNIT_DELAY_WDT seconds
242     // datasheet p43.
243     switch (UNIT_DELAY_WDT)
244     {
245         case 50:
246         WDTCR |= (1<<WDP2) | (1<<WDP0); // 0.5s
247         break;
248         case 1:
249         WDTCR |= (1<<WDP2) | (1<<WDP1); // 1s
250         break;
251         case 4:
252         WDTCR |= (1<<WDP3); // 4s
253         break;
254         case 8:
255         WDTCR |= (1<<WDP3) | (1<<WDP0); // 8s
256         break;
257         default:
258         WDTCR |= (1<<WDP3) | (1<<WDP0); // 8s
259     }
260 
261     // Enable watchdog timer interrupts
262     //WDTCR |= (1<<WDTIE);
263 }//WDTsetup
264 
265 void adc_setup (void)
266 {
267     // Set the ADC input to PB2/ADC1, left adjust result
268     ADMUX |= (1 << MUX0) | (1 << ADLAR);
269 
270     // Set the prescaler to clock/128 & enable ADC
271     // At 9.6 MHz this is 75 kHz.
272     // See ATtiny13 datasheet, Table 14.4.
273     // Also works fine for 1.2Mhz clock of ATtiny13a (2018.9.27)
274     ADCSRA |= (1 << ADPS1) | (1 << ADPS0) | (1 << ADEN);
275 
276     // Start the first conversion
277     ADCSRA |= (1 << ADSC);
278 }//adc_setup
```