```
/*************************************************************
 Target MCU & clock speed: ATtiny13A @ 1.2Mhz internal
 Name    : main.c
 C modules of this project, ISR:
main.c
Custom Headers:
    Nothing
 Author  : Insoo Kim (insoo@hotmail.com)
 Created : Sep 06, 2018 (On Atmel Studio 7)
 Updated : Sep 06, 2018 (On Atmel Studio 7)

 Description:
    A simple tone generator for ATtiny13A.

    The timer and counter control by ATtiny13A is reviewed while reading its datasheet.
    And here i try to test based on my understanding.


 HEX size[Byte]: 864 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₩TMRCNT-Sound-23188757₩Debug"
avrdude -c usbtiny -P usb -p attiny13 -U flash:w:TMRCNT-Sound-23188757.hex:i

 Ref:
 http://blog.podkalicki.com/attiny13-tone-generator/
 Codes:
 https://raw.githubusercontent.com/lpodkalicki/blog/master/avr/attiny13/007_tone_generator/main.c

*************************************************************/

/**
 * Copyright (c) 2016, Łukasz Marcin Podkalicki <lpodkalicki@gmail.com>
 * ATtiny13/007
 * Simple tone generator.
 */

#define F_CPU 1200000

#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include <util/delay.h>

#define BUZZER_PIN  PB0


#define N_1 (_BV(CS00))
```

```
53  #define N_8 (_BV(CS01))
54  #define N_64    (_BV(CS01)|_BV(CS00))
55  #define N_256   (_BV(CS02))
56  #define N_1024  (_BV(CS02)|_BV(CS00))
57
58  typedef struct s_note {
59      uint8_t OCRxn; // 0..255
60      uint8_t N;
61  } note_t;
62
63  typedef struct s_octave {
64      note_t note_C;
65      note_t note_CS;
66      note_t note_D;
67      note_t note_DS;
68      note_t note_E;
69      note_t note_F;
70      note_t note_FS;
71      note_t note_G;
72      note_t note_GS;
73      note_t note_A;
74      note_t note_AS;
75      note_t note_B;
76  } octave_t;
77
78  /*
79   All calculations below are prepared for ATtiny13 default clock source
       (1.2MHz)
80
81   F = F_CPU / (2 * N * (1 + OCRnx)),
82
83   where:
84   - F is a calculated PWM frequency
85   - F_CPU is a clock source (1.2MHz)
86   - the N variable represents the prescale factor (1, 8, 64, 256, or 1024).
87  */
88
89  PROGMEM const octave_t octaves[8] = {
90      { // octave 0
91      .note_C = {142, N_256}, // 16.35 Hz
92      .note_CS = {134, N_256}, // 17.32 Hz
93      .note_D = {127, N_256}, // 18.35 Hz
94      .note_DS = {120, N_256}, // 19.45 Hz
95      .note_E = {113, N_256}, // 20.60 Hz
96      .note_F = {106, N_256}, // 21.83 Hz
97      .note_FS = {100, N_256}, // 23.12 Hz
98      .note_G = {95, N_256}, // 24.50 Hz
99      .note_GS = {89, N_256}, // 25.96 Hz
100     .note_A = {84, N_256}, // 27.50 Hz
101     .note_AS = {79, N_256}, // 29.14 Hz
102     .note_B = {75, N_256} // 30.87 Hz
103     },
104     { // octave 1
105     .note_C = {71, N_256}, // 32.70 Hz
106     .note_CS = {67, N_256}, // 34.65 Hz
107     .note_D = {63, N_256}, // 36.71 Hz
```

```
108       .note_DS = {59, N_256}, // 38.89 Hz
109       .note_E = {56, N_256}, // 41.20 Hz
110       .note_F = {53, N_256}, // 43.65 Hz
111       .note_FS = {50, N_256}, // 46.25 Hz
112       .note_G = {47, N_256}, // 49.00 Hz
113       .note_GS = {44, N_256}, // 51.91 Hz
114       .note_A = {42, N_256}, // 55.00 Hz
115       .note_AS = {39, N_256}, // 58.27 Hz
116       .note_B = {37, N_256} // 61.74 Hz
117       },
118       { // octave 2
119       .note_C = {142, N_64}, // 65.41 Hz
120       .note_CS = {134, N_64}, // 69.30 Hz
121       .note_D = {127, N_64}, // 73.42 Hz
122       .note_DS = {120, N_64}, // 77.78 Hz
123       .note_E = {113, N_64}, // 82.41 Hz
124       .note_F = {106, N_64}, // 87.31 Hz
125       .note_FS = {100, N_64}, // 92.50 Hz
126       .note_G = {95, N_64}, // 98.00 Hz
127       .note_GS = {89, N_64}, // 103.83 Hz
128       .note_A = {84, N_64}, // 110.00 Hz
129       .note_AS = {79, N_64}, // 116.54 Hz
130       .note_B = {75, N_64} // 123.47 Hz
131       },
132       { // octave 3
133       .note_C = {71, N_64}, // 130.81 Hz
134       .note_CS = {67, N_64}, // 138.59 Hz
135       .note_D = {63, N_64}, // 146.83 Hz
136       .note_DS = {59, N_64}, // 155.56 Hz
137       .note_E = {56, N_64}, // 164.81 Hz
138       .note_F = {53, N_64}, // 174.61 Hz
139       .note_FS = {50, N_64}, // 185.00 Hz
140       .note_G = {47, N_64}, // 196.00 Hz
141       .note_GS = {44, N_64}, // 207.65 Hz
142       .note_A = {42, N_64}, // 220.00 Hz
143       .note_AS = {39, N_64}, // 233.08 Hz
144       .note_B = {37, N_64} // 246.94 Hz
145       },
146       { // octave 4
147       .note_C = {35, N_64}, // 261.63 Hz
148       .note_CS = {33, N_64}, // 277.18 Hz
149       .note_D = {31, N_64}, // 293.66 Hz
150       .note_DS = {29, N_64}, // 311.13 Hz
151       .note_E = {27, N_64}, // 329.63 Hz
152       .note_F = {26, N_64}, // 349.23 Hz
153       .note_FS = {24, N_64}, // 369.99 Hz
154       .note_G = {23, N_64}, // 392.00 Hz
155       .note_GS = {22, N_64}, // 415.30 Hz
156       .note_A = {20, N_64}, // 440.00 Hz
157       .note_AS = {19, N_64}, // 466.16 Hz
158       .note_B = {18, N_64} // 493.88 Hz
159       },
160       {  // octave 5
161       .note_C = {142, N_8}, // 523.25 Hz
162       .note_CS = {134, N_8}, // 554.37 Hz
163       .note_D = {127, N_8}, // 587.33 Hz
```

```
164      .note_DS = {120, N_8}, // 622.25 Hz
165      .note_E = {113, N_8}, // 659.25 Hz
166      .note_F = {106, N_8}, // 349.23 Hz
167      .note_FS = {100, N_8}, // 369.99 Hz
168      .note_G = {95, N_8}, // 392.00 Hz
169      .note_GS = {89, N_8}, // 415.30 Hz
170      .note_A = {84, N_8}, // 440.00 Hz
171      .note_AS = {79, N_8}, // 466.16 Hz
172      .note_B = {75, N_8} // 493.88 Hz
173      },
174      {  // octave 6
175      .note_C = {71, N_8}, // 1046.50 Hz
176      .note_CS = {67, N_8}, // 1108.73 Hz
177      .note_D = {63, N_8}, // 1174.66 Hz
178      .note_DS = {59, N_8}, // 1244.51 Hz
179      .note_E = {56, N_8}, // 1318.51 Hz
180      .note_F = {53, N_8}, // 1396.91 Hz
181      .note_FS = {50, N_8}, // 1479.98 Hz
182      .note_G = {47, N_8}, // 1567.98 Hz
183      .note_GS = {44, N_8}, // 1661.22 Hz
184      .note_A = {42, N_8}, // 1760.00 Hz
185      .note_AS = {39, N_8}, // 1864.66 Hz
186      .note_B = {37, N_8} // 1975.53 Hz
187      },
188      {  // octave 7
189      .note_C = {35, N_8}, // 2093.00 Hz
190      .note_CS = {33, N_8}, // 2217.46 Hz
191      .note_D = {31, N_8}, // 2349.32 Hz
192      .note_DS = {29, N_8}, // 2489.02 Hz
193      .note_E = {27, N_8}, // 2637.02 Hz
194      .note_F = {26, N_8}, // 2793.83 Hz
195      .note_FS = {24, N_8}, // 2959.96 Hz
196      .note_G = {23, N_8}, // 3135.96 Hz
197      .note_GS = {22, N_8}, // 3322.44 Hz
198      .note_A = {20, N_8}, // 3520.00 Hz
199      .note_AS = {19, N_8}, // 3729.31 Hz
200      .note_B = {18, N_8} // 3951.07 Hz
201      }
202  };
203
204  static void
205  tone(uint8_t octave, uint8_t note)
206  {
207      uint32_t ret;
208      note_t *val;
209      ret = pgm_read_word_near((uint8_t *)&octaves + sizeof(octave_t) * octave +
           sizeof(note_t) * note);
210      val = (note_t *)&ret;
211      TCCR0B = (TCCR0B & ~((1<<CS02)|(1<<CS01)|(1<<CS00))) | val->N; // set
           prescaler
212      OCR0A = val->OCRxn - 1; // set the OCRnx
213  }
214
215  static void
216  stop(void)
217  {
```

```
218
219     TCCR0B &= ~((1<<CS02)|(1<<CS01)|(1<<CS00)); // stop the timer
220 }
221
222 int
223 main(void)
224 {
225     uint8_t i, j;
226
227     /* setup */
228     DDRB |= _BV(BUZZER_PIN); // set BUZZER pin as OUTPUT
229     TCCR0A |= _BV(WGM01); // set timer mode to CTC
230     TCCR0A |= _BV(COM0A0); // connect PWM pin to Channel A of Timer0
231
232     /* Walk throwgh all octaves */
233     for (i = 0; i < 8; ++i) {
234         for (j = 0; j < 12; ++j) {
235             tone(i, j);
236             _delay_ms(80);
237         }
238     }
239
240     stop();
241     _delay_ms(1500);
242
243
244     /* loop */
245     while (1) {
246         /* Polish song "Wlaz ł  kotek na p ł otek" in loop */
247         tone(4, 7); // G
248         _delay_ms(500);
249         tone(4, 4); // E
250         _delay_ms(500);
251         tone(4, 4); // E
252         _delay_ms(500);
253         tone(4, 5); // F
254         _delay_ms(500);
255         tone(4, 2); // D
256         _delay_ms(500);
257         tone(4, 2); // D
258         _delay_ms(500);
259         tone(4, 0); // C
260         _delay_ms(200);
261         tone(4, 4); // E
262         _delay_ms(300);
263         tone(4, 7); // G
264         _delay_ms(1000);
265
266         stop();
267         _delay_ms(2000);
268
269         tone(4, 7); // G
270         _delay_ms(500);
271         tone(4, 4); // E
272         _delay_ms(500);
273         tone(4, 4); // E
```

```
274             _delay_ms(500);
275             tone(4, 5); // F
276             _delay_ms(500);
277             tone(4, 2); // D
278             _delay_ms(500);
279             tone(4, 2); // D
280             _delay_ms(500);
281             tone(4, 0); // C
282             _delay_ms(200);
283             tone(4, 4); // E
284             _delay_ms(300);
285             tone(4, 0); // C
286             _delay_ms(1000);
287
288             stop();
289             _delay_ms(5000);
290         }
291
292     }
293
```