```c
/*
 * 32bit_timer_int.c
 *
 * Created: 03/06/2013 15:18:19
* Author: SpinMos
* www.spinmos.com
*
===============================================================================
* Copyright (c) 2013, SpinMos company
===============================================================================
*
* Description
* ---------------------------------------------------
 * Use of the event system cascading two timers
 *
 * Timers C, E, type 0
 *    Timer C: When overflows gives an event
      which is used to increment timer E.
      When timer E reaches the TOP value,
      interrupts the program toggling PORT C bit 1

      It toggles every 1s
 */


#define F_CPU 32000000UL
#define __AVR_ATxmega128A3U
#include "avr/io.h"
#include "avr/interrupt.h"
#include "util/delay.h"

// Global functions
void Set32Mhz();

// ------------------------------------ Functions
// Sets 32Mhz internal oscillator
void Set32Mhz()
{
      OSC.CTRL|=OSC_RC32MEN_bm;
      while (!(OSC.STATUS & OSC_RC32MRDY_bm));
      CCP=CCP_IOREG_gc;
      CLK.CTRL=CLK_SCLKSEL_RC32M_gc;
      // If you want to disable RC2M
      OSC.CTRL&=(~OSC_RC2MEN_bm);
}

// Overflow timer E overflow interrupt
// toggles pin C1
ISR(TCE0_OVF_vect)
{
      PORTC.OUTTGL = 0x02;
}

// ----------------------------------------------------- MAIN
int main(void)
{
      // Defines 32Mhz internal clock
      Set32Mhz();

      PORTC_DIR = 0x02;                           // C1 is output

      // Timer C0 configuration
      TCC0_CTRLA = TC_CLKSEL_DIV8_gc;      // Div8, increments at 0.25 us

      // Event channel 0, multiplex configuration
      // When Timer C0 overflows produces an event
      EVSYS_CH0MUX = EVSYS_CHMUX_TCC0_OVF_gc;
```

```
    // Timer E multiplex configuration
    // Increments timer E0 at every channel 0 event
    // which is every 16.384 ms (65536 * 0.25us)
    TCE0_CTRLA = TC_CLKSEL_EVCH0_gc;      // clock timer E = event channel 0

    // low level interrupt
    TCE0_INTCTRLA = TC0_OVFINTLVL0_bm;
    TCE0.CNT = 0x00;
    // 61 times 16.384ms = 0.9994 seconds = 1 second
    TCE0_PER = 61;

    // Enable global interrupts
    SREG = CPU_I_bm;
    // Enable low level interrupts
    PMIC.CTRL = PMIC_LOLVLEN_bm;

    while(1)
    {
    }
}
```