16bit resolution dimmer

Major drawback from using 16bit timer is huge PWM frequency drop. In 8bit mode it would be 62.5 kHz.

Timer settings:

- timer TOP value equal ICR (input capture register). This gives variable dimmer resolution, if needed.
- OCR (output compare register) value changes pulse width.
- non-inverting mode.
- fast PWM mode. Provides a highest frequency for PWM waveform generation.
- No prescaler, PWM frequency is 244 Hz.

Code snippet:

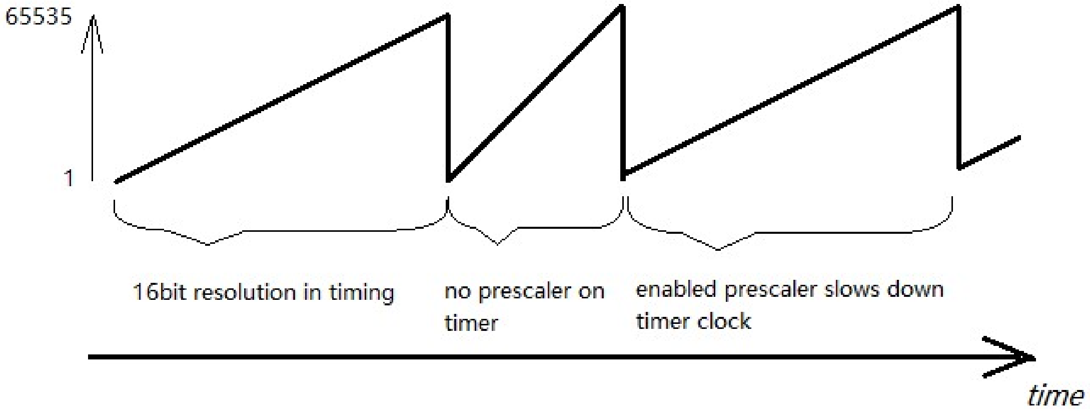
```
#define DIM_16bit   0xFFFF

void setupPWM16(uint16_t resolution){

    /* set D9 as output */
    DDRB = DDRB | 0b00000010;

    /* Timer/Counter Control Register setup/ initialisation */
    TCCR1A = (1 << COM1A1) | (1 << WGM11);
    TCCR1B = (1 << WGM13) | (1 << WGM12) | (1 << CS10);

    ICR1 = resolution;

}

/* during start-up */
setupPWM16(DIM_16bit);

/* on DMX512 signal arrival or during operation */
uint16_t MSB = DMXslot[0];
uint8_t LSB = DMXslot[1];
MSB = MSB << 8;
OCR1A = MSB + LSB;         // also can use OCR1AH/L for direct DMXslot assignment
```

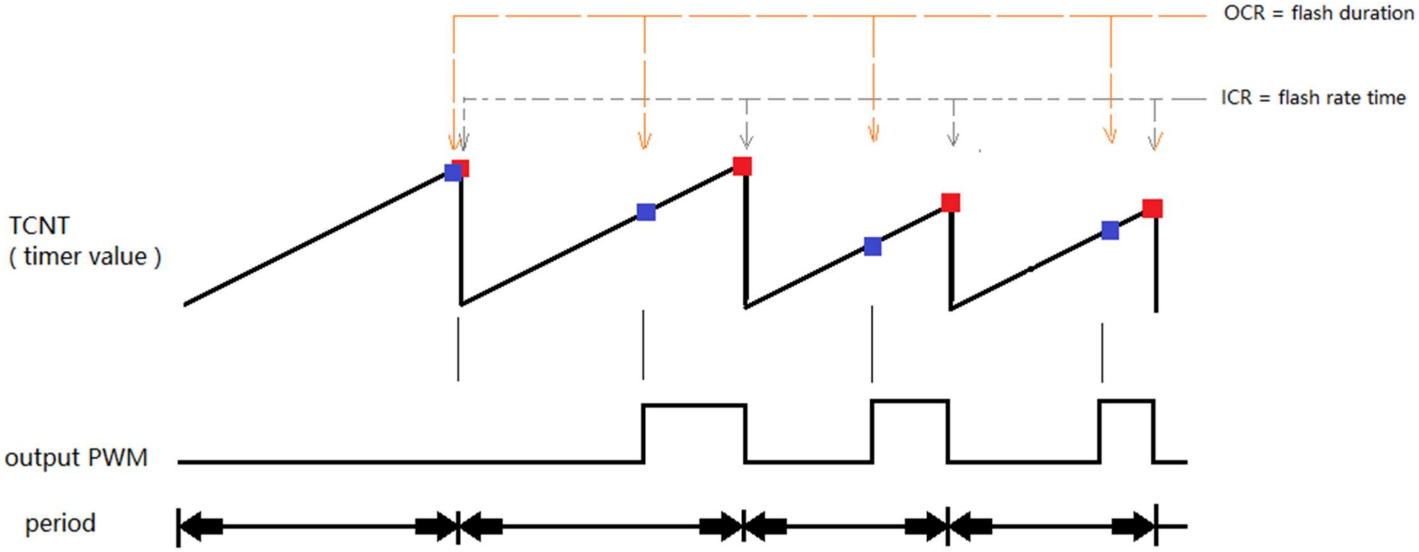
Strobe effect without fixed carrier frequency

During this experiment I noticed that – generally strobes (for example Martin Atomic3000) for timing uses AC frequency. Probably some zero crossing detector circuit. But, using MCU timer it is possible to create dynamic PWM signal period.

Timer settings:

- for lowest possible frequency, prescaler set to maximal value – 1024. In such configuration 16bit timer can generate frequency from 0.2 – 60 Hz (approximately). Usual flash rate frequency is 0.5 – 25 Hz.
- inverted output (needed for my LED driver).
- fast PWM mode.
- timer TOP value equal ICR (input capture register). This gives variable frequency (flash rate).
- OCR (output compare register) value changes pulse width (flash duration).


65535
1
16bit resolution in timing
no prescaler on timer
enabled prescaler slows down timer clock
time



Code snippet:

```
/* set D9 as output */
DDRB = DDRB | 0b00000010;

/* Timer/Counter Control Register setup/ initialisation */
TCCR1A = 1 << COM1A1 | 1 << COM1A0 | 1 << WGM11;
TCCR1B = 1 << WGM13 | 1 << WGM12 | 1 << CS12 | 1 << CS10;

/* Input Capture and Output Compare Register setup during operation */
OCR1A = DMXslot[0] << 8;      // flash duration (also can use OCR1AH)
ICR1 = DMXslot[1] << 8;       // flash rate time (interested in bit8-bit15, or can use
                              // ICR1H = DMXslot
```