# Таймери/лічильники

1 Сторожовий таймер
2 Таймер-лічильник типу A
3 Таймер-лічильник тип  D

## 1 Сторожовий таймер

Мікроконтролер ATtiny2313 має у своєму складі багатофункціональний сторожовий таймер (Watchdog Timer або WDT). Цей таймер має такі основні особливості:

- синхронізація від окремого внутрішнього генератора;
- три режими роботи :
- генерація запиту на переривання;
- системне скидання;
- генерація переривання і скидання;
- час спрацьовування від 16 мс до 8 с;
- можливість апаратного включення охоронного таймера (WDTON) для режиму підвищеної надійності.

Сторожовий таймер (WDT) виконаний у вигляді лічильника імпульсів, які поступають від спеціального внутрішнього генератора з частотою 128 кГц (рисунок 1). Схема WDT формує запит на переривання або системне скидання в той момент, коли вміст лічильника досягає заданого значення.

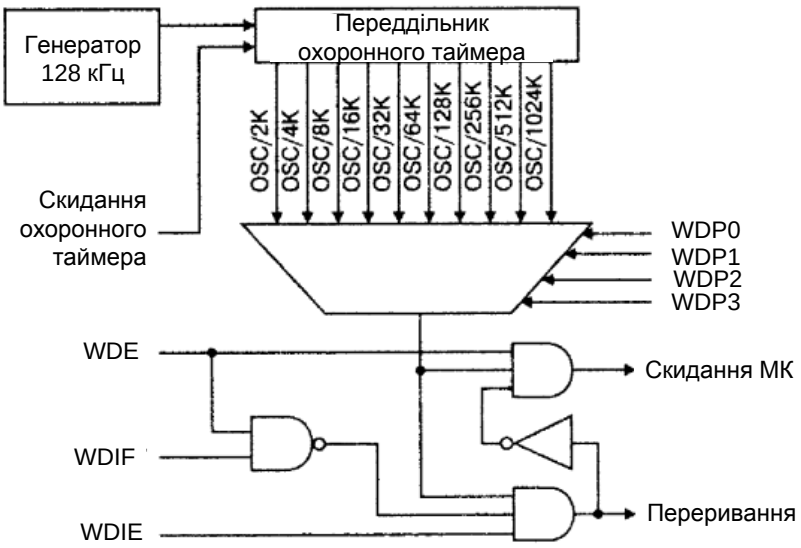


Рисунок 1 –  Блок-схема сторожового таймера

Таблиця 1– Регістр керування сторожовим таймером – WDTCSR

| Біти | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | WDTCSR |
|---|---|---|---|---|---|---|---|---|---|
| | **WDIF** | **WDIE** | **WDP3** | **WDCE** | **WDE** | **WDP2** | **WDP1** | **WDP0** | |
| Читання/Запис | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Біти 5, 2–0 (WDP3–0)**: вибір режиму роботи попереднього дільника охоронного таймера. Вони визначають коефіцієнт ділення попереднього дільника сторожового таймера. Усі можливі коефіцієнти ділення і відповідні їм періоди часу наведені в таблиці 2.

Таблиця 2 – Вибір режиму переддільника сторожового таймера

| **WDP3** | **WDP2** | **WDP1** | **WDP0** | **Кількість циклів генератора до спрацьовування WOT** | **Орієнтовний час при** VCC = 5,0V |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 2 К (2048) циклів | 16 мс |
| 0 | 0 | 0 | 1 | 4 К (4096)циклів | 32 мс |
| 0 | 0 | 1 | 0 | 8 К (8192) циклів | 64 мс |
| 0 | 0 | 1 | 1 | 16 К (16384) циклів | 0,125 с |
| 0 | 1 | 0 | 0 | 32 К (32768) циклів | 0,25 с |
| 0 | 1 | 0 | 1 | 64 К (65536) циклів | 0,5 с |
| 0 | 1 | 1 | 0 | 128 К (131072) циклів | 1,0с |
| 0 | 1 | 1 | 1 | 256 К (262144) циклів | 2,0 с |
| 1 | 0 | 0 | 0 | 512 К (524288) циклів | 4,0 с |
| 1 | 0 | 0 | 1 | 1024 К (1048576) циклів | 8,0 с |
| 1 | 0 | 1 | 0 | Зарезервовано | |
| 1 | 0 | 1 | 1 | | |
| 1 | 1 | 0 | 0 | | |
| 1 | 1 | 0 | 1 | | |
| 1 | 1 | 1 | 0 | | |
| 1 | 1 | 1 | 1 | | |

**Біт 6 (WDIE)**: біт дозволу переривань від сторожового таймера.

**Біт 3 (WDE)**: дозвіл режиму скидання. Прапор WDE дублює прапор WDRF в регістрі MCUSR. Це означає, що WDE завжди встановлений, якщо встановлений WDRF. Для того, щоб очистити WDE, треба спочатку очистити WDRF.

Таблиця 3 – Вибір режиму роботи сторожового таймера

| **WDTON (fuse)** | **WDE** | **WDIE** | **Режим охоронного таймера** | **Дія по завершенню контрольного часу** |
|---|---|---|---|---|
| 0 | 0 | 0 | Таймер зупинений | Нет |
| 0 | 0 | 1 | Режим переривання | Виклик переривання |
| 0 | 1 | 0 | Режим скидання | Системний скид |
| 0 | 1 | 1 | Режим переривання та скидання | Виклик переривання та перехід до режиму системного скидання |
| 1 | x | x | Режим скидання | Системне скидання |

*У нормальному режимі роботи* необхідно, щоб програма періодично скидала охоронний таймер за допомогою команди WDR. Програма має бути написана так, щоб команда скидання завжди приходила раніше, ніж вміст таймера досягне кінця. Якщо система зависне і перестане перезапускати лічильник, то він долічить кінця. Це викличе переривання або системне скидання. В результаті програма почне працювати спочатку.

*У режимі переривань* при витіканні заданого часу система WDT виробляє запит на переривання. Цей запит може використовуватися для пробудження мікроконтролера з будь-якого сплячого режиму.

*У режимі системного скидання* при витіканні заданого часу WDT викликає системне скидання. Це використовується для того, щоб запобігти зависанню системи у разі помилки в програмі.

*Третій режим об'єднує можливості двох перших режимів*. Спочатку викликається переривання, а потім відбувається системне скидання. Цей режим застосовується, наприклад, у тому випадку, коли перед викликом системного скидання необхідно зберегти важливі параметри.

**Біт 7 (WDIF):** прапор переривання від сторожового таймера. Цей біт встановлюється при спрацьовуванні сторожового таймера, якщо вибраний режим переривань.

**Біт 4 (WDCE):** дозвіл зміни режимів сторожового таймера. Цей біт використовується при зміні стану бітів попереднього дільника і біта WDE. Перед тим, як скинути біт WDE і змінити стан біт попереднього дільника, має бути встановлений розряд WDCE. Після запису в нього одиниці біт WDCE буде автоматично скинутий апаратним шляхом після закінчення чотирьох машинних циклів.

*Для зміни режиму роботи сторожового таймера треба виконати певну послідовність дій. Для скидання прапора WDE і зміни коефіцієнта перерахунку таймера необхідно виконати:*

*1) однією командою встановити в одиницю WDCE і біт WDE.*

*2) впродовж наступних чотирьох циклів тактового генератора записати необхідне значення у біт WDE і біти установки коефіцієнта дільника (WDP). Одночасно біт WDCE має бути скинутий. Усі ці три види значень необхідно записувати однією операцією.*

За командою з мнемокодом WDR виконується скидання перерахункової схеми у вихідний (нульовий) стан.

```
; Reset Watchdog
WDR
```

Для запуску сторожового таймера необхідно в програмі виконати команду WDR і потім установити в одиничний стан розряд *WDCE і біт WDE* WDTCSR. Записати необхідне значення у біт WDE, WDIE і біти установки коефіцієнта дільника (WDP). Одночасно біт WDCE має бути скинутий

```
; Turn off global interrupt
cli
; Reset Watchdog Timer
wdr
; Start timed sequence
in rl6, WDTCSR
ori rl6,(1<<WDCE)|(1<<WDE); 00011000b
out WDTCSR, rl6
; -- Got four cycles to set the new values from here -
; Set new prescaler(time-out) value = 64K cycles (-0.5 s)
ldi rl6,(1<<WDE)|(1<<WDP2)|(1<<WDP0); 00001101b
out WDTCSR, rl6
; -- Finished setting new values, used 2 cycles -
; Turn on global interrupt
sei
```

Для зупинки сторожового таймера необхідно очистити біт WDRF в регістрі MCUSR; установити в одиничний стан розряд *WDCE і біт WDE* WDTCSR; записати 0 у біт WDE, WDIE.

```
WDT_off:
; Turn off global interrupt
cli
; Reset Watchdog Timer
wdr
; Clear WDRF in MCUSR
in R16, MCUSR
andi R16,  (0xff & (0<<WDRF))
out  MCUSR,  R16
; Write logical one to WDCE и WDE
; Keep old prescaler setting to prevent unintentional time-
;out
in R16, WDTCSR
ori R16, (1<<WDCE)|(1<<WDE)
out WDTCSR,  r16
; Turn off WDT
ldi R16,  (0<<WDE)|(0<<WDIE)
out WDTCSR,  R16
;  Turn on global interrupt
sei
ret
```

## 2 Таймер-лічильник типу А

Таймер-лічильник типу А є в мікроконтролерів усіх типів. Він має ім'я Т/С0 (Х = 0). Таймер-лічильник типу А формує запит переривання Т/С0 OVF при переповненні восьми розрядного базового лічильника TCNTO. Структурна схема таймера-лічильника типу А зображена на рисунку 2.

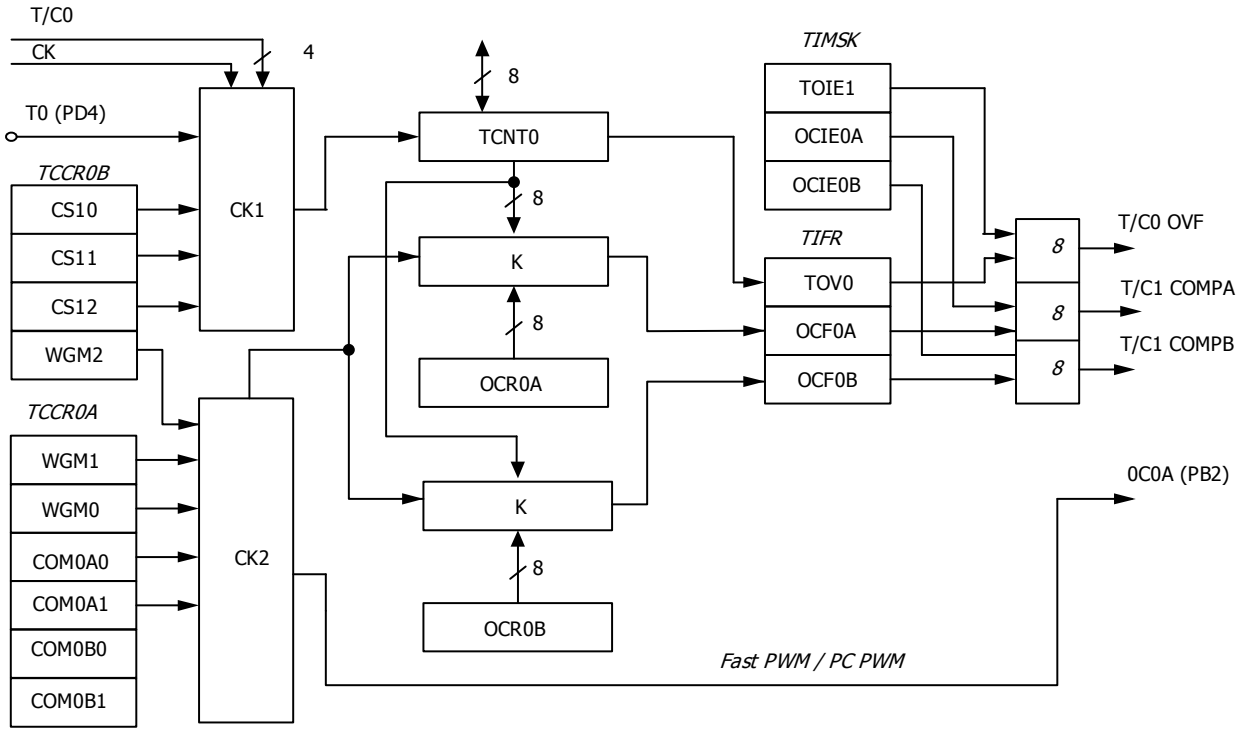


Рисунок 2 – Структурна схема таймера-лічильника типу А.

Тактовий сигнал мікроконтролера СК надходить у перерахункову схему, що представляє собою десяти розрядний лічильник, де виконується розподіл частоти тактового сигналу на 8, 64, 256 і 1024. Сигнали з чотирьох виходів перерахункової схеми надходять у схему керування СК1 (мультиплексор). При наявності в мікроконтролері таймера-лічильника Т/С1 ці ж сигнали надходять у Т/С1.

У схему керування надходять також тактовий сигнал СК1 і сигнал із зовнішнього джерела, прийнятий на вхід Т0.

Схема керування в залежності від комбінації станів розрядів CS00, CS01 і CS02 регістра керування TCCR0B передає один із сигналів, що надходять, на рахунковий вхід базового лічильника TCNTO, що веде рахунок на додавання. При переповненні лічильника TCNTO встановлюється в одиничний стан розряд TOV0 регістра TIFR і при одиничному стані розряду TOIE0 регістра TIMSK у блок переривань надходить запит переривання Т/С0 OVF.

Попередній дільник таймерів/лічильників 0 і 1 містить чотири ступені розподілу: СК/8, СК/64, СК/256 і СК/1024, де СК - вхідний тактовий сигнал. Крім того, як джерела тактових сигналів можуть бути використані сигнали від зовнішніх джерел, тактовий сигнал СК і нульовий тактовий сигнал (STOP).

Таблиця 4 – Регістр керування таймером/лічильником 0 - TCCR0B

| Біти | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | **FOC0A** | **FOC0B** | | | **WGM02** | **CS02** | **CS01** | **CS00** | **TCCR0B** |
| Читання/Запис | W | W | R | R | R | R/W | R/W | R/W | |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Біт 7, 6 — FOCO0A, FOCO0B** – примусова зміна сигналу на виході спів падання (канал А, В). Біт FOC0A, FOC0B активний тільки для режимів Normal, CTC. Не викликає переривання, таймер у режимі CTC не скидається. При читанні регістра значення біт FOCO0A, FOCO0B завжди дорівнює нулю.

Таблиця 5 – Керування таймером/лічильником 0 - TCCR0B

| CS02 | CS01 | CS00 | Опис |
|---|---|---|---|
| 0 | 0 | 0 | Таймер лічильник 0 зупинений |
| 0 | 0 | 1 | СК |
| 0 | 1 | 0 | СК/8 |
| 0 | 1 | 1 | СК/64 |
| 1 | 0 | 0 | СК/256 |
| 1 | 0 | 1 | СК/1024 |
| 1 | 1 | 0 | Зовнішній вхід Т0, спадаючий |
| 1 | 1 | 1 | Зовнішній вхід Т0, наростаючий |

Настройка ТС0 на режим лічби (Normal)

```
LDI   R16,$0F
OUT TCNT0, R16      Load the TCNTO register with $0F (00001111)
LDI   R16, 1
OUT TCCR0 , R16     Load the TCCRO register with $01 (00000001)
```

У режимі порівняння (CTC) значення регістра TCNT0 порівнюється зі значенням регістра OCR0x (OCR0A, OCR0B) та встановлюється відповідна ознака переривання OCF0x (OCF0A або OCF0B) та, якщо біт OCIE0x (OCIE0A, OCIE0B) регістра маски переривання TIFSK встановлений у 1, генерується запит переривання. Також може змінюватись стан виходу OC0A мікроконтролера. Це визначається станом біт COM0x1:COM0x0 регістра керування TCCR0A (табл. 6) у відповідності з табл. 7.

Таблиця 6 – Регістр керування таймером/лічильником 0 - TCCR0A

| Біти | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | **COM0A1** | **COM0A0** | **COM0B1** | **COM0B0** | | | **WGM01** | **WGM00** |
| Читання/Запис | R/W | R/W | R/W | R/W | R | R | R/W | R/W |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Таблиця 7 – Керування виводом OC0A в режимах Normal и CTC

| COM0x1 | COM0x0 | Опис |
|---|---|---|
| 0 | 0 | Таймер/лічильник 0 відключений від виводу OC0A (PB2) |
| 0 | 1 | Стан виводу змінюється на протилежний |
| 1 | 0 | Вивід скидається в 0 |
| 1 | 1 | Вивід встановлюється в 1 |

Режим Fast PWM («Швидкий ШІМ») дозволяє генерувати високочастотний сигнал з широтно -імпульсною модуляцією. Особливістю роботи схеми порівняння в даному режимі є подвійна буферізація запису в регістр OCR0x, яка полягає у тому, що число, що записується, зберігається в спеціальному буферному регістрі, а зміна змісту регістру порівняння відбувається тільки у момент коли лічильник досягне максимальне значення (рис.4).

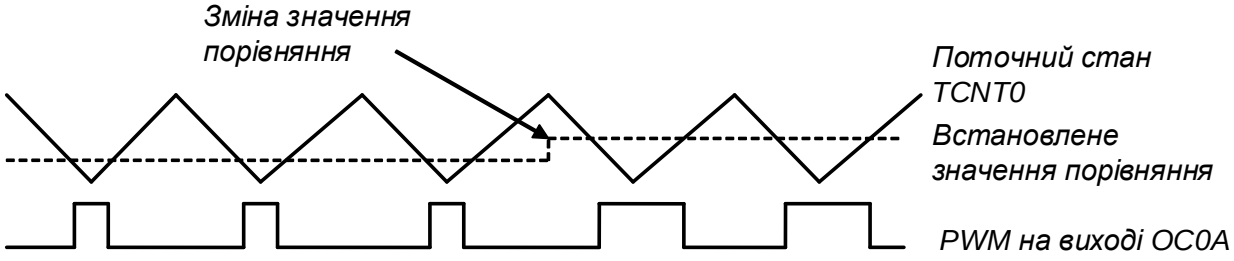


Рисунок 4 – Встановлення нового значення в регістр порівняння OCR0x

Стан виходу OC0A мікроконтролера в даному режимі також визначається станом біт COM0x1:COM0x0 регістра TCCR0A (табл. 8).

Таблиця 8 – Керування виводом OCA в режимах PWM

| COM0x1 | COM0x0 | Опис |
|---|---|---|
| 0 | 0 | Таймер/лічильник 0 відключений від виводу OC0A (PB2) |
| 0 | 1 | Таймер/лічильник 0 відключений від виводу OC0A (PB2) |
| 1 | 0 | 0 коли TCNT0 = OCR0, 1 при скиданні лічильника – в режимі Fast PWM або по скиданню при зворотній лічбі – в режимі Phase Correct PWM (неінвертований ШІМ) |
| 1 | 1 | 1 коли TCNT0 = OCR0, 0 при скиданні лічильника – в режимі Fast PWM або по скиданню при зворотній лічбі – в режимі Phase Correct PWM (інвертований ШІМ) |

У режимі Phase Correct PWM («ШІМ з точною фазою») регістр TCNT0 функціонує як реверсивний лічильник, зміна стану якого відбувається з кожним імпульсом тактового сигналу. Стан лічильника спочатку змінюється від $00 до максимального значення $FF, а далі зворотно до $00. Відповідно максимальна частота сигналу в даному режимі у 2 рази менша максимальної частоти сигналу в режимі Fast PWM. Такий режим рекомендується використовувати для задач керування двигунами. Коли лічильник TCNT0 досяг мінімального значення ($00) також відбувається зміна напрямку лічби й одночасно встановлюється флаг переривання TOV0. Коли значення TCNT0 дорівнює значенню регістра порівняння OCR0x встановлюється флаг OCF0x та змінюється стан виводу OC0A. Характер зміни стану OC0A визначається, станом біт COM0x1:COM0x0 регістра керування TCCR0A (табл. 8).

Біти WGM01:0 регістра TCCR0A разом з бітом WGM02 регістра TCCR0B керують режимом роботи TC0, напрямком лічби, вибором максимального значення для TCNT0 (TOP)(табл. 9).

Таблиця 9 – Режими роботи таймера/лічильника TC0

| № | WGM2 (TCCR0B) | WGM1 (TCCR0A) | WGM0 (TCCR0A) | Назва режиму | Верхня межа (TOP) | OCRx змінюється | Флаг TOV встановлюється від |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Normal | 0xFF | Безпосередньо | 0xFF |
| 1 | 0 | 0 | 1 | PWM, Phase Correct | 0xFF | При досягненні 0xFF | 0x00 |
| 2 | 0 | 1 | 0 | CTC | OCRA | Безпосередньо | 0xFF |
| 3 | 0 | 1 | 1 | Fast PWM | 0xFF | При досягненні 0xFF | 0xFF |
| 4 | 1 | 0 | 0 | Резерв | - | - | - |
| 5 | 1 | 0 | 1 | PWM, Phase Correct | OCRA | При досягненні 0xFF | 0x00 |
| 6 | 1 | 1 | 0 | Резерв | - | - | - |
| 7 | 1 | 1 | 1 | Fast PWM | OCRA | При досягненні 0xFF | 0xFF |

Таблиця 9 – Регістр масок переривань TIMSK

| Біти | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | **TOIE1** | **OCIE1A** | **OCIE1B** | | **ICIE1** | **OCIE0B** | **TOIE0** | **OCIE0A** |
| Читання/Запис | R/W | R/W | R/W | R/W | R | R | R/W | R/W |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Біт 7, 1 – TOIE1, TOIE0: дозвіл переривання у режимі Normal таймера/лічильника 1 та таймера/ лічильника 0.

Біт 6,5 – OCIE1A, OCIE1B: дозвіл переривання таймера/лічильника 1 у режимі порівняння (CTC) в каналі A, B.

Біт 2,0 – OCIE0B, OCIE0A: дозвіл переривання таймера/лічильника 0 у режимі порівняння (CTC) в каналі A, B.

Біт 3 – ICIE1: дозвіл переривання таймера/лічильника 1 у режимі захоплення.

Таблиця 10 – Регістр ознак переривань TIFR

| Біти | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | **TOV1** | **OCF1A** | **OCF1B** | | **ICF1** | **OCF0B** | **TOV0** | **OCF0A** |
| Читання/Запис | R/W | R/W | R/W | R/W | R | R | R/W | R/W |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Біт 7,1– TOV1, TOV0: ознака переповнення таймера/лічильника 1, таймера/лічильника 0.

Біт 6,5, 2, 0 OCF1A,OCF1B,OCF0B, OCF0A: ознака співпадання в каналі A, B таймера/лічильника 1 та таймера/лічильника 0.

2 — OCF0B: Флаг совпадения в канале B.

Біт 3 — ICF1: ознака захоплення змісту таймера/лічильника 1

## 3 Таймер-лічильник тип D

Таймер-лічильник типу D має ім'я T/C1. Він містить шістнадцатирозрядний базовий лічильник і виконує функції захоплення, порівняння та широтно імпульсної модуляції (PWM). Структурна схема таймера-лічильника зображена на рисунку 5.

На рахунковий вхід шістнадцатирозрядного базового лічильника TCNT1H, L з виходу схеми керування СК1 може надходити тактовий сигнал мікроконтролера СК, чи один з чотирьох сигналів з перерахункової схеми T/C1, чи сигнал із зовнішнього джерела, що надходить на вхід T1. В якості входу T1 використовується вивід порту PD5. Вибір сигналу визначається комбінацією станів розрядів CS10, CS11 і CS12 регістра керування TCCR1B. При переповненні базового лічильника встановлюється в одиничний стан розряд TOV1 регістра TIFR і при одиничному стані розряд TOV1 регістра TIMSK у блок перериванні надходить запит переривання T/C1 OVF.

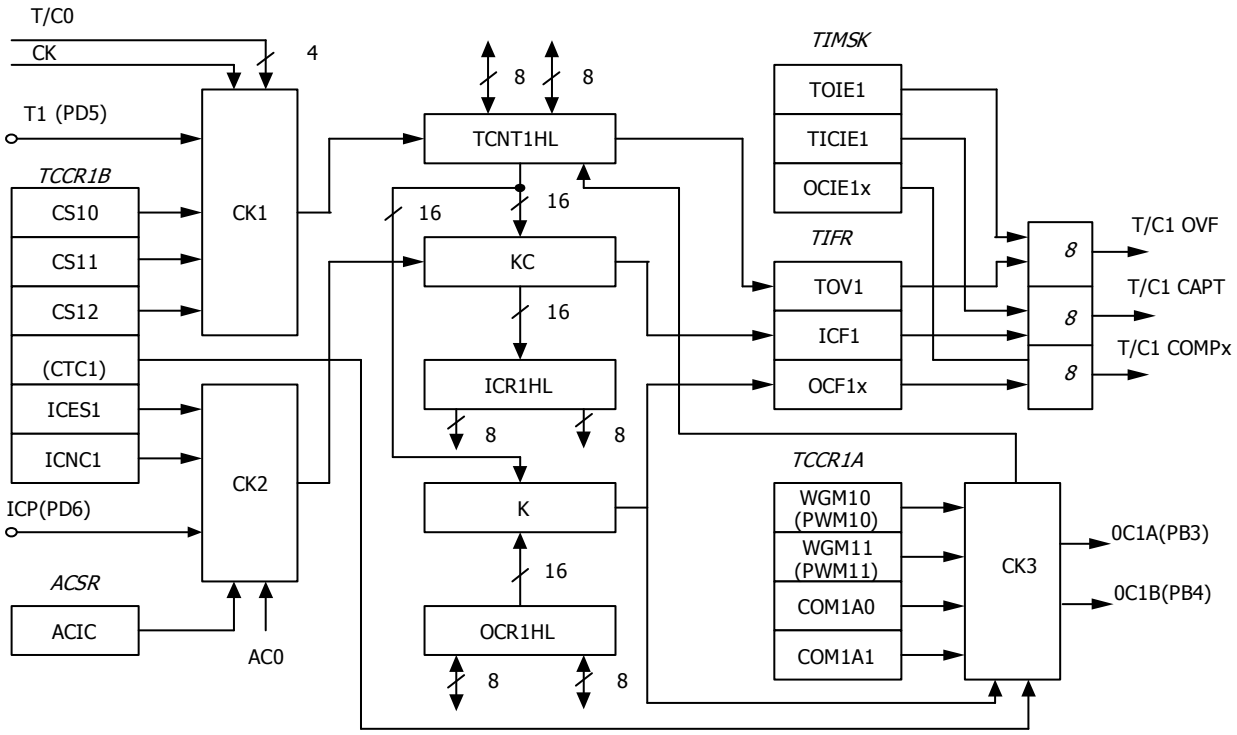


Рисунок 4 – Структурна схема таймера-лічильника типу D

Настройка T/C1 на режим лічби

```
TIM_OVF1:
        LDI     R16, $FF
        OUT     TCNT1H, R16
        LDI     R16, $E5
        OUT     TCNT1L ,R16
        LDI     R16,0
        OUT     TCCR1A, R16
        LDI     R16,1
        OUT     TCCR1B, R16
        LDI     R16, $80
        OUT     TIMSK, R16
        LDI     R16, $DF
        OUT     SPL, R16
        SEI
```

Схема керування СК2 керує виконанням функції захоплення, що полягає в передачі коду, сформованого в базовому лічильнику, через ключову схему КС у шістнадцятирозрядний регістр захоплення ICR1H, L. При цьому встановлюється в одиничний стан розряд ICF1 регістра TIFR і при одиничному стані розряду ICIE1 регістра TIMSK у блок переривань надходить запит переривання T/C1 CAPT.

Захоплення виконується при зміні значення зовнішнього сигналу, що поступає на вхід ICP, чи внутрішнього сигналу ACO, що надходить з

аналогового компаратора. Вибір сигналу визначається станом розряду ACI регістра ACSR, що входить до складу аналогового компаратора. При ACI = 0 використовується зовнішній сигнал, при ACI = 1 – внутрішній. Вид зміни сигналу, при якому виконується захоплення, визначається станом розряду ICES1 регістра TCCR1B. При ICES1 =0 захоплення виконується з появою негативного фронту сигналу, а при ICES1 = 1 - позитивного фронту.

Розряд ICNC1 регістра TCCR1B керує роботою схеми подавлення завад. При ICNC1 = 0 захоплення виконується з кожною появою фронту заданої полярності.

При ICNC1 = 1 захоплення відбувається, якщо перед появою фронту на протязі чотирьох тактів сигнал зберігає постійне значення.

Таблиця 11 – Регістр керування B таймера/лічильника 1 – TCCR1B

| Біт | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| $2E($4E) | ICNC1 | ICES1 | — | WGM13 | WGM12 | CS12 | CS11 | CS10 |
| Читання/Запис | R/W | R/W | R | R/W | R/W | R/W | R/W | R/W |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Настройка T/C1 на режим захоплення

```
TIM_CAPT1:
        LDI R16, 0
        OUT TCNT1H, R16
        OUT TCNT1L, R16
        OUT ICR1H, R16
        OUT ICR1L, R16
        OUT ACSR, R16
        OUT TIMSK, R16
        OUT TCCR1A, R16
        LDI R16,$01
        OUT TCCR1B, R16
        SEI
```

Схема керування СКЗ керує виконанням функції порівняння/PWM. Функція порівняння полягає у видачі визначеного значення сигналу на виході OC1x при збігу кодів у базовому лічильнику і шістнадцятирозрядному регістрі порівняння OCR1H, L, яке виявляється за допомогою компаратора К. При цьому також встановлюється в одиничний стан розряд OCF1Ax регістра TIFR і при одиничному стані розряду OCI1x регістра TIMSK у блок переривань надходить запит переривання T/C1 COMP.

Таблиця 12 – Регістр керування А таймера/лічилника 1 – TCCR1A

| Біт | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| $2F($4F) | COM1A1 | COM1A0 | COM1B1 | COM1B0 | | | WGM11 | WGM10 |
| Читання/Запис | R/W | R/W | R | R/W | R/W | R/W | R/W | R/W |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Настройка Т/С1 на режим порівняння

```
TIM_C0MP1:
        LDI R16,0
        OUT TCNT1H,  R16
        LDI R16, $30
        OUT TCNT1L, R16
        LDI R16, $40
        OUT TCCR1A, R16
        LDI R16,1
        OUT TCCR1B, R16
        LDI R16, 0
        OUT OCR1AH, R16
        LDI R16, $47
        OUT OCR1AL, R16
        LDI  R16 ,$40
        OUT TIMSK,  R16
        SEI
```

Функція PWM полягає у видачі на вихід OC1 імпульсного сигналу з заданим періодом повторення і заданою тривалістю імпульсу. При цьому також періодично формується запит переривання Т/С1 COMP.

Робота схеми СК3 визначається комбінацією станів розрядів WGM11, WGM10 (PWM10, PWM11), COM1x0 і COM1x1 регістру керування TCCR1A. При нульовому стані всіх чотирьох розрядів функція порівняння/PWM не виконується і вихід OC1 відключений від виводу порту

При WGM 10=0, WGM 11=0 і інші комбінації станів розрядів COM1x0 і COM1x1 виконується функція порівняння (табл.13). При виконанні функції порівняння режим роботи базового лічильника залежить від стану розряду WGM12 (CTC1) регістра керування TCCR1B. При WGM12=1 базовий лічильник при збігу кодів скидається в нульовий стан і продовжує рахунок, починаючи з 0.

При WGM12 = 0 він продовжує рахунок до переповнення і далі веде рахунок, починаючи з 0. При одиничному стані хоча б одного з розрядів

WGM10 і WGM11 і одиничному стані розряду COM1x1 виконується функція PWM. У цьому випадку базовий лічильник веде лічбу на додавання до одержання числа 255 або 511, або 1023, переходить у режим рахунку на вирахування, веде лічбу на вирахування до одержання числа 0 і знову повертається в режим рахунку на додавання. Вибір максимального числа (Nmax), до якого ведеться рахунок на додавання, визначається комбінацією станів розрядів WGM11 і WGM10 регістра керування TCCR1A .

Сигнал PWM формується шляхом зміни значення сигналу на виході OC1x при збігу кодів у базовому лічильнику і регістрі OCR1x у процесі рахунка на додавання і на вирахування. Вид зміни сигналу залежить від стану розряду COM1x0. На рисунку 5 зображені графіки зміни числа в базовому лічильнику (TCNT1) і часові діаграми сигналу PWM при різних станах розряду COM1x0.

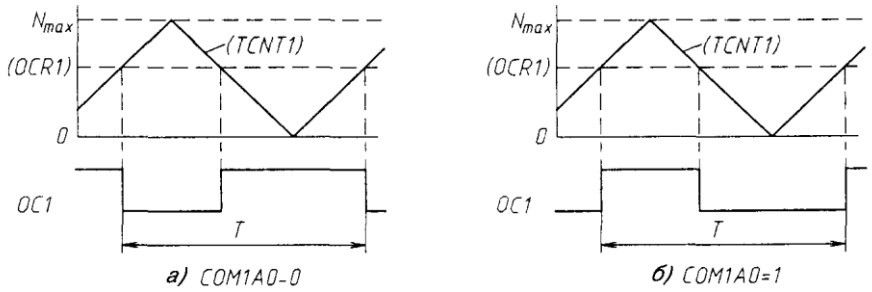


Рисунок 5 – Графіки зміни числа в базовому лічильнику (TCNT1) і тимчасові діаграми сигналу PWM при різних станах розряду COM1A0.

Період сигналу PWM (T) залежить від максимального числа, до якого виконується рахунок на додавання. При Nmax = 255 період у 510 разів більше періоду проходження імпульсів на рахунковому вході базового лічильника. При Nmax=511 і 1023 це відношення дорівнює 1022 і 20461 відповідно.

Розряд WGM12 регістра TCCR1B у режимі PWM не використовується. Запит переривання T/C1 OVF формується при переході базового лічильника від числа 0 до числа 1. При записі коду в регістр OCR1x код запам'ятовується в регістрі тимчасового збереження. Перезапис коду в регістр OCR1x виконується з появою в базовому лічильнику максимального числа, що запобігає появі в сигналі PWM імпульсу з випадковою тривалістю.

Вивід порту PB3 використовується як вивід OC1A, вивід порту PB4 використовується як вивід OC1B у мікроконтролера типу 2313

Настройка T/C1 на режим PWM

```
PWM:
        LDI   R16,$C1
```

```
        OUT TCCR1A, R16
        LDI   R16,1
        OUT TCCR1B, R16
        LDI   R16, 0
        OUT OCR1AH, R16
        LDI   R16, $47
        OUT OCR1AL, R16

LOOP: RJMP LOOP
```

Таблиця 13 – Режими роботи таймера/лічильника TC1

| Номер режиму | WGM13 | WGM12 | WGM11 | WGM10 | Назварежиму | Модуль лічби (TOP) | Оновлення регістрів OCR1A та OCR1B | Момент установки флагу TOV1 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Normal | $FFFF | Безпосередньо | $FFFF |
| 1 | 0 | 0 | 0 | 1 | Phase correct PWM, 8-бітний | $00FF | При TOP | $0000 |
| 2 | 0 | 0 | 1 | 0 | Phase correct PWM, 9-бітний | $01FF | При TOP | $0000 |
| 3 | 0 | 0 | 1 | 1 | Phase correct PWM, 10-бітний | $03FF | При TOP | $0000 |
| 4 | 0 | 1 | 0 | 0 | CTC (скидання при співпадінні) | OCR1A | Безпосередньо | $FFFF |
| 5 | 0 | 1 | 0 | 1 | Fast PWM, 8-бітний | $00FF | При TOP | При TOP |
| 6 | 0 | 1 | 1 | 0 | Fast PWM, 9-бітний | $01FF | При TOP | При TOP |
| 7 | 0 | 1 | 1 | 1 | Fast PWM, 10-бітний | $03FF | При TOP | При TOP |
| 8 | 1 | 0 | 0 | 0 | Phase and Frequency Correct PWM | ICR1 | $0000 | $0000 |
| 9 | 1 | 0 | 0 | 1 | Phase and Frequency Correct PWM | OCR1A | $0000 | $0000 |
| 10 | 1 | 0 | 1 | 0 | Phase correct PWM | ICR1 | При TOP | $0000 |
| 11 | 1 | 0 | 1 | 1 | Phase correct PWM | OCR1A | При TOP | $0000 |
| 12 | 1 | 1 | 0 | 0 | CTC (сброс при совпадении) | ICR1 | Безпосередньо | $FFFF |
| 13 | 1 | 1 | 0 | 1 | Зарезервировано | - | - | - |
| 14 | 1 | 1 | 1 | 0 | Fast PWM | ICR1 | При TOP | При TOP |
| 15 | 1 | 1 | 1 | 1 | Fast PWM | OCR1A | При TOP | При TOP |

Розглянемо приклад створення ефекту «бігуча одиниця» з використанням таймера лічильника.

Основні вузли таймера/лічильника TC1 (рисунок 4):

– TIMSK - Timer Interrupt MaSK register - регістр маски переривання;

- TIFR - Timer Interrupt Flag Register - регістр ознак переривань;
- TCCR1A - Timer/Counter1 Control Register A – регістр керування 1 таймера A;
- TCCR1B - Timer/Counter1 Control Register B – регістр керування 1 таймера B;
- ICR1 - timer/counter1 Input Capture Register1 - вхідний регістр 1-го таймера
- TCNT1 - Timer/CouNTer1 - регістр стану таймера
- K - компаратор (Comparator) 16-bit
- OCR1 - timer/counter Output1 Compare Register - вихідний регістр компаратора

Для формування часових інтервалів потрібно задіяти:
- TIMSK – він визначає, які переривання таймера будемо використовувати;
- TCCR1B – регістр керування таймером TC1;
- TCNT0 – регістр стану таймера. Його потрібно обнуляти за перериванням компаратора;
- OCR1A – до нього завантажується число з яким порівнює компаратор.

Таблиця 6 – Регістр масок переривання таймера/лічильника – TIMSK

| Біти | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| $37 ($57) | **TOIE 1** | **OCIE 1** | | | **TICIE1** | | **TOIE0** | |
| Читання/Запис | R/W | R/W | R | R | R/W | R | R/W | R |
| Початковий стан | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit7 - TOIE1 - Timer/Counter1 Overflow Interrupt Enable - дозвіл переривання по переповненню 1-го таймера
- Bit 6 - OCIE1A - Timer/Counter1 Output Compare Match Interrupt Enable – дозвіл переривання компаратора 1-го таймера
- Bit 3 - TICIE1 - Timer/Counter1 Input Capture Interrupt Enable - дозвіл переривання 1-го таймера
- Bit 1 - TOIE0 - Timer/Counter0 Overflow Interrupt Enable - дозвіл переривання по переповненню 0-го таймера

З регістра масок переривання таймера/лічильника – TIMSK потрібне тільки переривання компаратора, тому потрібно встановити у TIMSK 6-й біт, а решта залишається 0.

TIMSK = 0b01000000

Визначимось з регістром TCCR1B– цікавить тільки 3 молодших біта (CS10…CS12)

Для створення ефекту «бігуча одиниця» світлодіоди повинні перемикатись орієнтовно один раз за секунду. При наявності 8 світлодіодів час перемикання кожного з світлодіодів повинен бути 1/8с - 0,125с (125 мс). При тактовій частоті мікроконтролера 10 МГц його період 100 нс. Максимальне значення таймера: $2^{16} = 65535$.

Наша задача – підібрати тактову частоту таймера так, щоб він рахував до 65535 за 125мс.

При тактовій частоті 10 МГц таймер дорахує до кінця за 100 нс * 65536= =6,6 мс – мало!

Якщо тактову частоту поділити на 8:

6,6 * 8 = 52 мс - також мало.

А якщо на 64?

6,6 * 64 = 419 мс, що більше ніж 125мс.

Потрібно встановити Bit1, Bit0 TCCR1B для коефіцієнту поділу частоти на 64. Так

TCCR1B = 0b00000011

Визначимо число, що буде завантажено до OCR1A з яким буде порівнювати компаратор поточний стан таймера. Тактова частота таймера в 64 рази менше частоти кварцу. Значить її період - у 64 рази більше:

100 нс * 64 = 6,4 мкс.

Підрахуємо кількість тактових імпульсів за час 125 мс:

125мс / 6,4 мкс = 19531

Затримка у 125мс дорівнює 19531 імпульсів. Саме це число завантажимо у OCR1A. Цей регістр складається з двох 8 бітних регістрів OCR1AH та OCR1AL. Перетворимо 19531 у шістнадцяти розрядне число й завантажимо його до OCR1AH (старша частина) та OCR1AL (молодша частина).

19531(10) = 4C4B(16).

OCR1AH = 0x4C

OCR1AL = 0x4B

Програма на мові Assembler для створення ефекту «бігуча одиниця» з використанням таймера лічильника TC1.

```
.include "2313def.inc"
.cseg
.org 0
.def temp =R16
.def temp1 =R20

rjmp Reset            ;вектора переривань
rjmp INT_0
```

```
rjmp INT_1
rjmp Timer1_capt1
rjmp Timer1_comp1
rjmp Timer1_OVF1
rjmp Timer0_OVF0
rjmp UART_RX
rjmp UART_UDRE
rjmp UART_TX
rjmp ANA_COMP

;Reset:
INT_0:
INT_1:
Timer1_capt1:
;Timer1_comp1:
Timer1_OVF1:
Timer0_OVF0:
UART_RX:
UART_UDRE:
UART_TX:
ANA_COMP:
          reti
;****************************************************
; Ініціалізація
;****************************************************
Reset:    ldi Temp,0b11111111   ;настройка портів
          out DDRB,Temp

          ldi Temp,0b01000000   ;дозвіл переривання компаратора
          out TIMSK,Temp

          ldi Temp,0b00000011   ;тактовий сигнал = CK/64
          out TCCR1B,Temp

          ldi Temp,0x4C         ;ініціалізація компаратора
          out OCR1AH,Temp
          ldi Temp,0x4B
          out OCR1AL,Temp

          ldi Temp,RamEnd       ;установка вказівника стека
          out SPL,Temp

          ldi Temp1,0b00000001  ;ініціалізація індикатора

          ldi Temp,0            ;скид таймера
          out TCNT1H,Temp
          out TCNT1L,Temp

          sei                   ;дозвіл преривання
;****************************************************
; ОСНОВНИЙ ЦИКЛ
;****************************************************
Inf:      rjmp Inf              ;безкінцевий цикл


;****************************************************
; ОБРОБЛЮВАЧ ПЕРЕРИВАННЯ КОМПАРАТОРА
;****************************************************

Timer1_comp1:
         ldi Temp,0            ;скид таймера
         out TCNT1H,Temp
         out TCNT1L,Temp
```

```
Shift:   cpi Temp1,0b10000000  ;порівняти з кінцевим знач.
         breq Init             ;якщо дорівнює - завантаження поч. знач.

         lsl Temp1             ;інакше - зсув ліворуч
         rjmp Output           ;перейти на вивід в порт

Init:    ldi Temp1,0b00000001  ;завантажити поч. значення
Output:  out PortB,Temp1       ;вивід в порт

         reti                  ;вихід з оброблювача
```

Програма формування ШІМ для регулювання яскравості світлодіода (PB2) за натисненням кнопок «-» (PD0) та «+» (PD1)

```
; ********************************************************************
; Для AVR:  ATtiny2313
; Тактова частота:  8МГц
; ********************************************************************
; Функції:  Ініціалізація ШІМ, керування двома кнопками
; ********************************************************************

      .include "tn2313def.inc"
      .def temp = r16
init:
      ldi temp,RAMEND
      out SPL,temp

; ========== Ініціалізація преддільника ======================

      ldi temp,(1<<CLKPCE)    ; Активуємо дозвіл запису в
      out CLKPR, temp         ; регістр CLKPR
      clr temp                ; Обнуляємо регістр temp
      out CLKPR, temp         ; Поділ тактової частоти в регистрі CLKPR =1

; ================== Ініціалізація портів ================

      ldi temp,0b00000011
      out PortD,temp        ; відключаємо резистори підтяжки портів D
      com temp
      out DDRD,temp         ; переведення всіх біт порта D на вивід
      ser temp
      out DDRB,temp
      clr temp
      out PortB, temp

; ==================== Ініціалізація таймера T0 ==========================

      ldi temp,$C3;           (1<<COM0A1)|(1<<COM0A0)|(1<<WGM01)|(1<<WGM00)
      out TCCR0A, temp
      ldi temp,1;             (1<<CS00)
      out TCCR0B, temp
; ====================Визначення шпаруватості ШІМ ======================
      ldi r18, 9
Correct:
      out OCR0A, r18
; ======================Main prgram=====================================
Sensor:
      SBIS PinD,0           ; читаємо 0 біт регістра введення
      ldi r20,0b00000001    ; та записуємо в r20 значення 1, якщо PD0=0 біт
```

```
      SBIS PinD,1             ; читаємо 1 біт регістра введення
      ldi r20,0b00000010      ; та записуємо в r20 значення 2, якщо PD1=0 біт
      SBRC r20,0              ; якщо 0 біт регістра r20 дорівнює 1,
      rjmp plus               ; то переходимо на мітку plus
      SBRC r20,1              ; якщо 1 біт регістра r20 дорівнює 1,
      rjmp minus              ; то переходимо на мітку minus
      rjmp Sensor
; ============================== calculation step PWM ====================
plus:
      rcall delay                    ; Перехід до підпрограми затримки
      ldi temp, 121                  ; Перевіряємо, чи нема переповнення  при
      add r18, temp                  ; наступному кроці
      brcs plus_1                    ; якщо переповнення є, то переходим на
                                    ; мітку plus_1
      ldi temp, 112                  ; Якщо переповнення не має, зсуваємось на
      sub r18, temp                  ; один крок вперед, який дорівнює заданому
                                    ; кроку шпаруватості
      rjmp    Correct                ; Повернення на опитування кнопок
plus_1:
      ldi r18,135                    ; Записуємо кінцеве положення кроку
      rjmp    Correct                ; Повернення на опитування кнопок

minus:
      rcall delay                    ; Перехід до підпрограми затримки
      ldi temp, 10                   ; Перевіряємо, чи немає переповнення
                                    ; (результат менше нуля)  при
      sub r18, temp                  ; наступному кроці
      brcs minus_1                   ; Якщо переповнення є, то переходимо на
                                    ; мітку minus_1
      inc r18                        ; Якщо переповнення немає, зсуваємось на
      rjmp    Correct                ; один крок назад, який рівний заданому
                                    ; кроку шпаруватості
minus_1:
      ldi r18, 9                     ; Записуємо початкове положення кроку
      rjmp    Correct                ; Повернення на опитування кнопок

; ============================== п/п затримки ============
delay:
  clr r20
  clr r21
  ldi r22,1

delay1:
  dec r20
  brne delay1
  dec r21
  brne delay1
  dec r22
  brne delay1

  ret
```