Ръководство за конфигуриране на таймер в режим на бърза ШИМ в ATMega328P

Програмиране на вградени микроконтролерни системи
Технологично училище "Електронни Системи"към ТУ-София

10 януари 2021 г.

1 Какво представляват таймерите

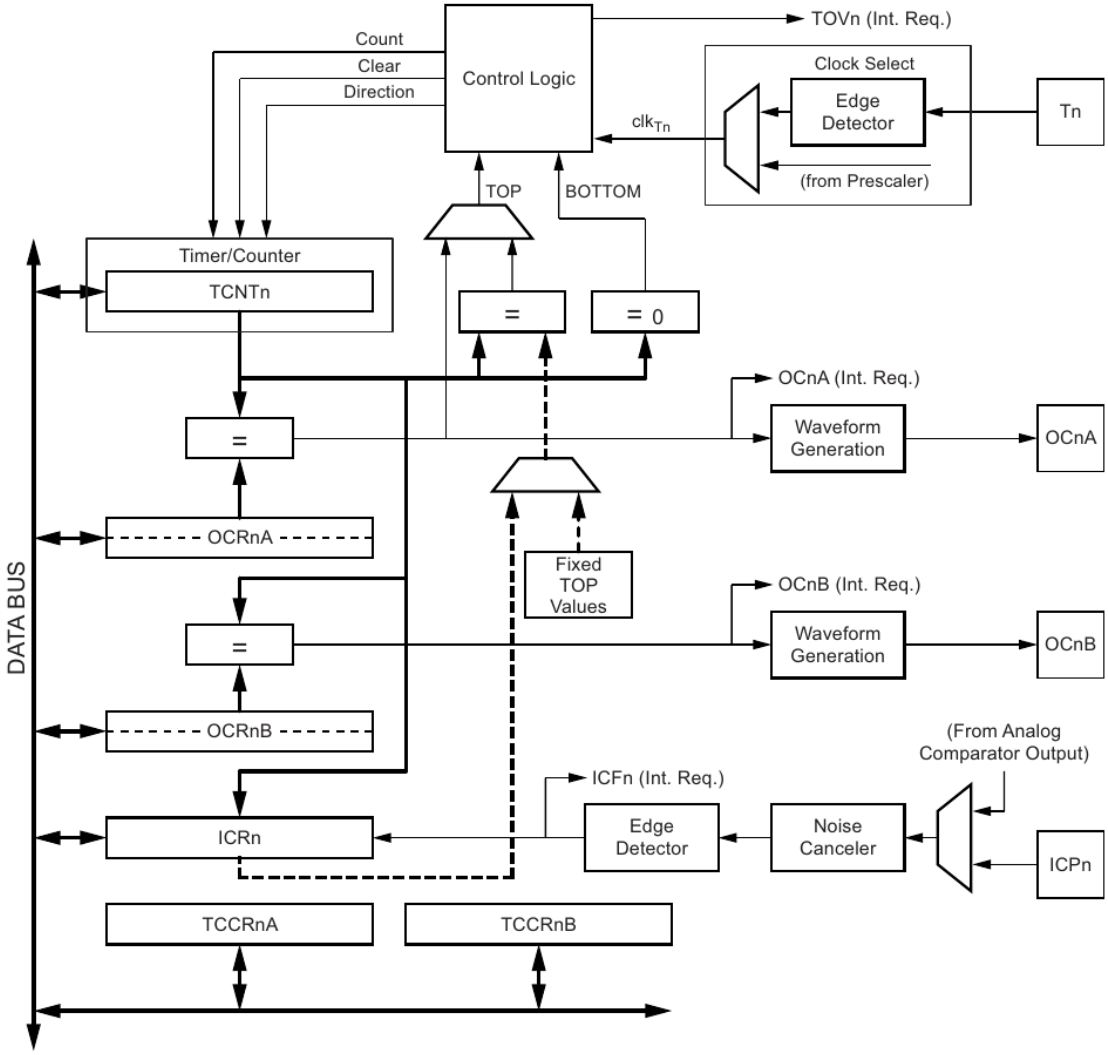
Таймерите са електронни схеми за генериране и измерване на цифрови сигнали с точни времеви параметри. Имат два основни режима: входен (input capture) и изходен (output compare). Намират широко и разнообразно приложение и са задължителна част от всеки модерен микроконтролер и микропроцесор.

При работа в изходен режим таймерът генерира поредица от правоъгълни импулси с програмируеми честота, коефициент на запълване и поляритет. Този режим се използва за хардуерна широчинно-импулсна модулация, управление на сервомотори, регулиране на яркостта на светодиоди, издаване на прости звуци и др. Таймерът може да изпраща сигнал за прекъсване към ядрото на микроконтролера/микропроцесора на всеки период. Това позволява изпълняването на подпрограма през точно определен интервал от време, което се използва от почти всички модерни операционни системи за имплементирането на некооперативна многозадачност (preemptive multitasking).

Входният режим на таймера се използва за засичане и измерване на продължителността и периода на цифров сигнал. Таймерът може да изпраща сигнал за прекъсване към ядрото на микроконтролера/микропроцесора при фронт на входния сигнал.

На фигура 1 е показана блоковата схема на 16-битовия таймер, наличен в микроконтролера ATMega328P. Този таймер разполага с два изходни канала и един входен канал. TCNT1[1] е броячният регистър. Неговата стойност се инкрементира при всеки импулс на тактовия сигнал. Стойността на този регистър постоянно се сравнява със стойностите на двата изходни регистъра: OCR1A и OCR1B. При изравняване с някоя от тях, в зависимост от конфигурацията на таймера, се променя нивото на съответния изходен пин или се изпълнява прекъсване. При засичане на преден или заден (в зависимост от конфигурацията) фронт на входен сигнал текущата стойност на броячния регистър се копира във входния регистър ICR1.

[1]На схемата n е номерът на таймера. ATMega328P има 3 таймера и n може да бъде 0, 1 или 2. Разглежданият тук 16-битов таймер е с номер 1.



Фигура 1: Блокова схема на 16-битов таймер в ATMega328P

Когато броячният регистър достигне максималната си стойност се нулира. Максималната стойност може да бъде $2^{16} - 1$, една от предвидените за тази цел хардуерни константи или съдържанието на един от регистрите OCR1A и ICR1. Поради това един от трите канала не може да се използва, когато е зададена максимална стойност, различна от предвидените константи. Таймерите в повечето модерни микроконтролери разполагат с отделен регистър, предназначен да съхранява максималната стойност на броене.

В зависимост от конфигурацията, честотата на броене (инкрементиране на TCNT1) може да бъде честотата на ядрото (обичайно 16 MHz), разделена на 1, 8, 64, 256 или 1024, или да се подава външно през извод на микроконтролера. Освен това, тактовият сигнал за броене може да изключен. Това на практика спира таймера на пауза.

Таймерът се конфигурира чрез трите конфигурационни регистъра TCCR1A, TCCR1B и TCCR1C (не е показан на схемата и много рядко се използва). Чрез задаване на подходящи стойност на отделните битове в тези регистри могат да се конфигурират режима на таймера, максималната стойност на броене, тактовата честота, поляритета на изходните пинове и някои други функционалности. Значението на всеки бит в тези регистри е описано подробно в документацията на микроконтролера.[1]

Таймерът може да изпраща към ядрото на микроконтролера/микропроцесора сигнал за прекъсване при различни събития. Чрез регистъра TIMSK1 се конфигурира кои събития да водят до изпълняването на прекъсване.

Някои функционалности, с които разглежданият таймер не разполага, но се срещат често в други микроконтролери са възможност да брои в обратна посока, отделен регистър за максимална стойност на броене, възможност даден канал да се използва както като входен, така и като изходен, комплементарни изходи, възможност за свързване на таймери в master-slave конфигурация, DMA и разнообразни режими със специфични предназначения, например режим за управление на BLDC мотор или на POV екран.

2 Конфигуриране на таймер в режим на бърза ШИМ

Ще разгледаме конфигурирането на канал B на таймер 1 в ATMega328P в режим на бърза широчинно-импулсна модулация. Режимът на изходния пин ще бъде неинвертиращ и ще се изпълнява прекъсване при изравняване на брояча с изходния регистър. Честотата ще бъде 10 kHz и коефициентът на запълване ще бъде 25%. Конфигурирането в други режими се извършва по аналогичен начин. При конфигурирането на таймер се използват побитови оператори за манипулиране на отделните битове.

Първата стъпка при конфигурирането на таймер в ATMega328P е да се нулират контролните регистри. Това изключва таймера, ако преди това е бил включен. Също така се нулира регистъра, разрешаващ прекъсвания от таймера. Това забранява всички прекъсвания от таймера, ако такива са били разрешени.

```
1 TCCR1A = 0;
2 TCCR1B = 0;
3 TIMSK1 = 0;
```

Също така се нулира броячния регистър. Това се извършва при спрян таймер. В противен случай стойността му ще се променя докато се конфигурират останалите регистри, а това не е желателно.

```
4 TCNT1  = 0;
```

Задава се режим на таймера бърза ШИМ с максимална стойност на броене в регистър ICR1. Според таблица 15-5 от документацията на микроконтролера[1] това е режим номер 14 и за да бъде избран трябва битовете WGM11, WGM12 и WGM13 да се зададат единици. Номерата на битовете са дефинирани със съответните имена. Образуването на маска за задаване на конкретен бит става посредством преместването наляво на единица толкова пъти, колкото е номера на бита, който искаме да зададем като единица. Обърнете внимание, че трите бита, с които работим, не са в един и същи регистър.

```
5 TCCR1A |= (1 << WGM11);
6 TCCR1B |= (1 << WGM12) | (1 << WGM13);
```

Избира се входна честота за брояча. Възможностите са 16 MHz, 2 MHz, 250 kHz, 62,5 kHz и 15,625 kHz (честотата на ядрото, разделена на 1, 8, 64, 256 или 1024). Тази честота не може бъде по-голяма от $2^{16} \cdot f_{OCPWM}$, където f_{OCPWM} е желаната изходна честота, защото таймерът е 16-битов. Желателно е да бъде максимално висока за да позволи максимално голяма прецизност при задаването на коефициента на запълване. Избира се входна честота 16 MHz, тоест честотата на ядрото, разделена на 1. Според таблица 15-6 от документацията на микроконтролера[1] тази честота се задава с бита CS10. Важно е да се отбележи, че след задаването на този бит таймерът се включва и стойността на TCNT1 започва да се променя. Повечето модерни микроконтролери разполагат с отделен бит, който стартира таймера, но поради тази особеност на ATMega328P в някои случаи е желателно CSx битовете да се конфигурират последни. Обърнете внимание, че се използва операторът |= (присвояване с побитово ИЛИ). Ако се използва = (обикновено присвояване на стойност) ще се нулират битовете, които вече са конфигурирани.

```
7 TCCR1B |= (1 << CS10);
```

Формулата за изчисляване на максималната стойност на броене е следната:

$$TOP = \frac{f_{clkIO}}{f_{OCPWM} \cdot N} - 1 \tag{1}$$

където N е делителят на честотата на ядрото (f_{clkIO}). Тази формула се извежда чрез решаване на уравнението на стр. 102 от документацията на микроконтролера[1] по отношение на TOP. За честота на ШИМ 10 kHz получаваме:

$$TOP = \frac{16 \cdot 10^6}{10 \cdot 10^3 \cdot 1} - 1 = 1599 \tag{2}$$

Резултатът се записва в регистър ICR1.

```
8  ICR1 = 1599;
```

Стойността на изходния регистър се определя от максималната стойност на броене и коефициента на запълване.

$$OCB = (TOP + 1) \cdot D \tag{3}$$

За коефициент на запълване 25% получаваме:

$$OCB = (1599 + 1) \cdot 0,25 = 400 \tag{4}$$

Резултатът се записва в регистър OCR1B.

```
10  OCR1B = 400;
```

Според таблица 15-3 от документацията на микроконтролера[1] в режим на бърза ШИМ режимът на изходните пинове се задава с битове COM1A0, COM1A1, COM1B0 и COM1B1. В неинвертиращ режим при изравняване на изходния регистър и броячния регистър пинът за съответния канал се установява в логическа нула, а при нулиране на броячния регистър пинът се установява в логическа единица. В инвертиращ режим е обратното.

Задава се неинвертиращ режим за канал B:

```
11  TCCR1A |= (1 << COM1B1);
```

Чрез регистър TIMSK1 се определя при кои събития се изпраща сигнал за прекъсване към ядрото. Както става ясно от точка 15.11.8 от документацията на микроконтролера[1], за да се разреши прекъсване при изравняване на брояча с изходния регистър (Compare match) на канал B трябва да се зададе бит OCIE1B в единица.

```
12  TIMSK1 |= (1 << OCIE1B);
```

Функцията, която се изпълнява при прекъсването (Interrupt Service Routine) има някои особености. Указател към нея трябва да се запише на точно определен адрес, за да може ядрото да я намери и изпълни при получаване на сигнал за съответното прекъсване. Този указател е част от векторната таблица, която е масив от указатели към функциите за всички различни прекъсвания и се намира на адрес 0x0000 в програмната памет. Поради тази и други особености за дефиниране на функция, изпълнявана при прекъсване, се използва макроса ISR(име_на_прекъсването). Имената на прекъсванията се образуват от имената в таблица 11-1 от документацията на микроконтролера[1] като интервалите се заместят с долни черти и се добави "_vect"в края. Можете да намерите имената на прекъсванията в [2] и [3].

Функцията, изпълнявана при изравняване на брояча с изходния регистър на канал B се дефинира по следния начин:

```
13  ISR(TIMER1_COMPB_vect)
14  {
15    // code
16  }
```

Добра практика е докато се конфигурират прекъсвания да се забранят глобално прекъсванията, за да не се изпълни неочаквано някое прекъсване преди всички свързани с него регистри да са конфигурирани правилно. Това може да не е нужно, ако програмистът е прочел подробно нужната документация и е убеден, че конфигурира всички регистри в правилния ред. Прекъсванията в ATMega микроконтролерите се разрешават и забраняват глобално съответно с функциите sei() и cli(). В Arduino IDE са налични и функциите interrupts() и noInterrupts(), които правят същото.

Готовата програма за конфигуриране на таймера изглежда по следния начин:

```
void setup()
{
    noInterrupts();

    TCCR1A = 0;
    TCCR1B = 0;
    TIMSK1 = 0;
    TCNT1  = 0;

    TCCR1A |= (1 << WGM11);
    TCCR1B |= (1 << WGM12) | (1 << WGM13);

    ICR1 = 1599;
    OCR1B = 400;

    TCCR1A |= (1 << COM1B1);
    TIMSK1 |= (1 << OCIE1B);
    TCCR1B |= (1 << CS10);

    interrupts();
}

void loop()
{
    // code
}

ISR(TIMER1_COMPB_vect)
{
    // more code
}
```

Литература

[1] Atmel, *ATMega328P - 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash*, 2015. url: `https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf` (цитирано на стр. 3—5).

[2] *avr-libc User manual*, 2016. url: `https://www.nongnu.org/avr-libc/user-manual/group__avr__interrupts.html` (цитирано на стр. 5).

[3] В. Гаристов и И. Стефанов, *Материали по Програмиране на ВМКС*, 2020. url: `https://gitlab.com/tues-embedded/vmks/-/tree/master` (цитирано на стр. 5).