

# Osnovi mikroprocesorskih i mikrokontrolerskih sistema - Tajmeri i brojači

prof. dr Ivan Mezei

13. maj 2022.





## Glava 9

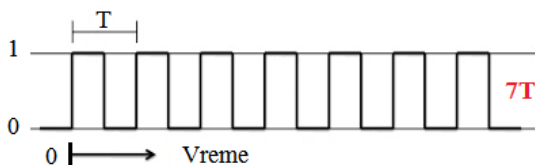
# Tajmeri i brojači

U okviru ovog poglavlja ćemo se upoznati sa tajmerima/brojačima kao jednim od najčešće korišćenih hardverskih resursa mikrokontrolera. Tajmeri i brojači su značajni zbog toga što većina embeded sistema radi u realnom vremenu i reaguje na događaje u svom okruženju te je neophodno imati mogućnost merenja proteklog vremena (tajmer, eng. *timer*), odnosno broja detektovanih događaja (brojač, eng. *counter*). Kada se skraćeno zapisuje tajmer/brojač se obično označava kao T/C (od prvih slova engleskih naziva).

Mikrokontroleri obično imaju jedan ili više tajmerskih modula kao periferijske jedinice koje su integrisane na istom čipu zajedno sa CPU. Tajmeri su obično visoko konfigurabilni, kako bi ih korisnik lako mogao prilagoditi svojoj konkretnoj potrebi. Tajmeri po pravilu mogu da generišu zahteve za prekidom (eng. *interrupt*) ili da se označavanje proteklog vremena vrši postavljanjem određenog indikatora koji se može prozivati radi provere (metoda prozivke, eng. *polling*). Neki od tajmera imaju mogućnost čuvanja podatka o tačnom vremenskom trenutku pojave odgovarajućeg događaja u sistemu i koriste se za rad u sistemima sa realnim vremenom. Koristeći tajmere i pravilno ih konfigurišući, projektant embeded sistema može u značajnoj meri rasteriti CPU dugačke liste zadataka koji uključuju merenje i manipulaciju vremenom.

U najosnovnijoj formi, tajmer je običan brojač, realizovan kao N-bitni brojački registar, koji se okida i uvećava svoju vrednost za jedan periodičnim signalom takta. Osnovni princip rada svakog tajmera je sledeći. Počevši od neke proizvoljne vrednosti brojačkog registra, tipično od nule, brojač se uvećava za jedan (inkrementuje) svaki put kada se na signalu takta pojavi rastuća (ili opadajuća) ivica. Na ovaj način brojač broji broj rastućih ivica signala takta. Po pravilu, tajmeri su

osetljivi ili na rastuću ili na opadajuću ivicu signala takta, nikada na obe. Ukoliko je signal takta periodičan, sa poznatom periodom  $T$ , tada broj odbrojanih ivica u tajmeru  $k$ , zapravo može da se interpretira kao proteklo vreme  $kT$  od trenutka startovanja tajmera (npr. trenutak 0) do pojave poslednjeg događaja (videti sliku 9.1).



Slika 9.1: Primer proteklog vremena u trajanju 7 perioda

## 9.1 Proširenje opsega brojanja

Ukoliko je brojački registar  $N$ -bitni tada je opseg brojanja od  $0-2^N - 1$ . Dakle najveća moguća vrednost izbrojanih perioda je  $2^N * T$ .  $N$  obično ima vrednost 8, 16 ili 32 bita. Ukoliko je  $T$  kratko (visoka učestanost) tada može da se desi situacija da brojač ne može da izbroji potrebno vreme. Tada je neophodno uraditi proširenje opsega brojanja.

Navedeni problem može se rešiti na tri načina:

1. Proširenjem brojačkog opsega tajmera korišćenjem softverske varijable - vrednost varijable se uvećava za jedan u prekidnoj rutini tajmera svaki put kada tajmer dostigne svoj maksimum. Pogledati primer na slici 9.2.
2. Proširenjem brojačkog opsega tajmera kaskadnim vezivanjem više tajmerskih modula (videti sliku 9.3).
3. Korišćenjem preskalera. U slučaju korišćenja preskalera, kada bi on bio konfigurisan da deli ulazni takt sa faktorom  $p$ , tajmer bi efektivno brojao pojavu skupova od  $p$  uzastopih događaja ili merio efektivno  $p$  puta duže vreme.

### Primer

Sledeći primer bliže ilustruje način korišćenja preskalera za povećanje brojačkog opsega tajmera. Preskaler unutar tajmera konfigurisan je da deli ulazni takt sa faktorom 8. Brojač unutar tajmera je 16-bitni.

```

unsigned char slika_led = 0xFF;
unsigned char t0_cnt = 0;

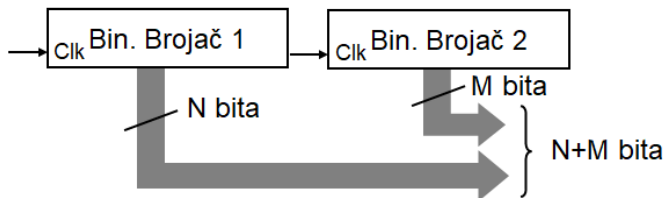
ISR(TIMERO_COMPA_vect)
{
    //prekid tajmera 0 usled dostizanja vrednosti registra OCR0A
    t0_cnt++;
}

int main()
{
    ...
    //inicijalizacija tajmera 0
    ...

    if (t0_cnt == 125){
        t0_cnt = 0;
        slika_led ^= 0x01;
        PORTD = slika_led;
    }
    ...
}

```

Slika 9.2: Primer upotrebe softverske promenljive



Slika 9.3: Kaskadno proširenje opsega brojanja

a) Koji je maksimalni broj događaja koji se može izbrojati sa ovako konfigurisanim tajmerom?

Obzirom da brojač uvećava svoju vrednost nakon što se detektuje 8 uzastopnih događaja (jer je preskaler konfigurisan tako da deli ulazni takt signal sa faktorom 8), maksimalni broj događaja koji se može izbrojati je:  $65536 * 8 = 524288$  događaja

b) Koji broj treba upisati u komparatorski registar da bi tajmer nakon odbrojanja 150000 događaja generisao zahtev za prekidom?

Brojač se mora resetovati kada odbroji ukupno 150000 događaja. Kako se vrednost brojača uvećava za jedan nakon svakih 8 uzastopnih događaja, vrednost koju treba da sadrži komparatorski registar je:  $150000 / 8 = 18750$

c) Da li je sa ovako konfigurisanim sistemom moguće izbrojati tačno 48333 događaja?

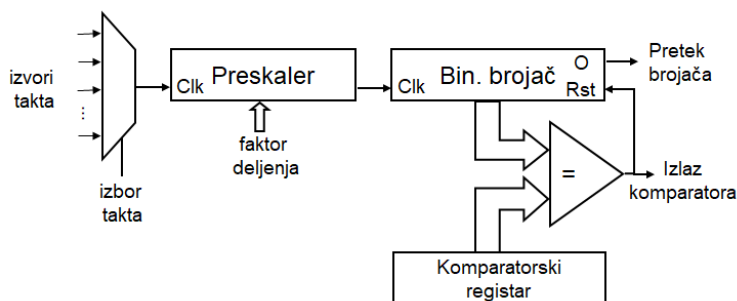
Obzirom da vrednost 48333 nije deljiva sa brojem 8, brojač nije u mogućnosti da izbroji tačno 48333 događaja.

## 9.2 Generalna struktura tajmera/brojača

Da bi se povećale mogućnosti tajmera, pored brojačkog registra on obično sadrži i dodatne module koji čine arhitekturu odgovarajućeg tajmera. Osnovne komponente od kojih se sastoji većina tajmera (videti sliku 9.4) su:

- selektor signala takta (multiplekser) koji dozvoljava izbor jednog od više mogućih izvora signala takta,
- preskaler obezbeđuje mogućnost deljenja učestanosti signala takta (usporevanja signala takta) pre nego što on uđe u brojački blok,
- brojački modul koji sadrži N-bitni brojački registar i obezbeđuje osnovnu brojačku funkciju,
- komparatorski registar omogućava da se definiše maksimalna N-bitna vrednost koju brojač može dostići, i
- komparator jednakosti (N-bitni) omogućava detekciju trenutka kada brojač dostiže vrednost koja je smeštena u komparatorskom registru. Po pravilu, dostizanjem ove vrednosti automatski se vrši resetovanje brojača, aktiviranje prekida ukoliko tajmer to podržava, što je najčešći slučaj, i postavljanje odgovarajućeg indikatora.

U opštem slučaju, takt signal tajmera može da potiče od raznih izvora unutar embeded sistema koji uključuju unutrašnje i spoljašnje izvore takt signala ili čak neke asinhronne izvore događaja koji se koriste kao takt signal. Modul za selekciju signala takta daje mogućnost projektantu embeded sistema da na jedan isti tajmerski modul dovede čitav niz različitih izvora takt signala i da u toku rada menja koji od njih zapravo služi za okidanje tajmera. U većini mikrokontrolera takt signali tajmera potiču od sistemskog takt signala. U mnogim aplikacijama učestanost sistemskog signala takta je previsoka da bi omogućila potrebno merenje proteklog



Slika 9.4: Generalna unutrašnja struktura tajmera

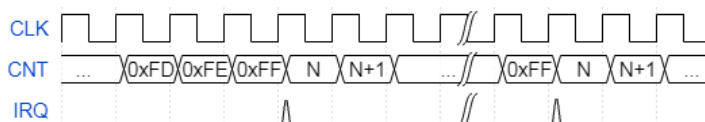
vremena, jer bi zahtevala tajmer sa vrlo velikim brojačkim registrom. Upravo zbog toga javlja se potreba za smanjivanjem učestanosti sistemskog takt signala za potrebe korišćenja unutar tajmera. Funkcija preskalera, odnosno delitelja ulaznog takta, služi da uspori ulazni takt za zadati faktor deljenja. Po pravilu, faktor deljenja se može zadati softverskim upisom u odgovarajući registar koji se nalazi unutar tajmera. Izlaz preskalera, koji predstavlja takt signal, ali sada sa manjom učestanošću od ulaznog takt signala, se zatim vodi na ulaz brojačkog modula.

Brojački registar je najznačajnija komponenta svakog tajmerskog modula. Uobičajene veličine brojačkih registara su 8, 16 ili 32 bita. U većini tajmera, brojački registar je istovremeno i registar sa mogućnošću upisa i čitanja, što omogućava programeru da postavlja vrednost brojačkog registra na proizvoljne početne vrednosti kao i da čita trenutnu vrednost koja je smeštena u brojačkom registru. Brojač menja svoje stanje kada detektuje odgovarajući događaj (obično je to rastuća ivica takta) na ulaznom takt signalu, uvećavajući vrednost brojačkog registra za jedan. Na ovaj način brojač "broji" događaje na ulaznom takt signalu, tj. protekle periode takta.

Obzirom da je brojački registar realizovan kao N-bitni registar, brojač broji od vrednosti 0 do vrednosti  $2^N - 1$ . Kada dostigne vrednost  $2^N - 1$  brojač generiše signal prekoračenja opsega ili preteka (eng. *overflow*) i započinje brojanje ponovo od vrednosti 0. Signal preteka se obično može konfigurisati da izazove pojavu zahteva za prekidom. Na slici 9.5 prikazan je rad jednog 8-bitnog brojača, čiji je opseg brojanja od 0-255 (u heksadecimlanoj notaciji to je od 0x00-0xFF). Ovde je takođe prikazano ponašanje signala zahteva za prekidom (IRQ) koji se aktivira kada se u brojaču desi prekoračenje opsega (prelazak sa vrednosti 255 (0xFF) na vrednost 0). U ovakvim sistemima je neophodno da se prvo upiše odgovarajuća

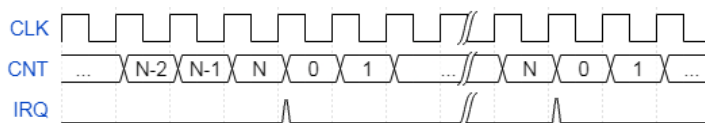


početna vrednost (na slici je to vrednost  $N$ ) u brojački registar tako da se dobije željeni broj perioda koje treba da brojač izbroji pre pojave prekida. Dakle ovde brojimo od neke početne vrednosti do momenta preteka tajmera. Primer ovakvih tajmera su kod mikrokontrolera 8051.



Slika 9.5: Generisanje prekida kod tajmera bez komparatora

Jednostavniji način je da brojački registar kreće da broji od nule do neke vrednosti. Tada je potrebno postojanje komparatorskog registra i komparatora jednako kao sastavnih delova tajmera. Korišćenjem ova dva modula može se generisati signal koji će biti aktivan kada brojač dostigne vrednost koja je specificirana u komparatorskom registru. Ovaj signal može zatim generisati odgovarajući zahtev za prekidom (IRQ) i resetovati brojač, kao što je prikazano na slici 9.6. Kada 8-bitni brojač (CNT), koji ovog puta kreće da broji od 0, dostigne vrednost upisanu u komparatorski registar (npr.  $N$ ) tada se javlja zahtev za prekidom (IRQ).



Slika 9.6: Generisanje prekida kod tajmera sa komparatorom

## 9.3 Primene brojača

U ovoj sekciji će biti prikazane sledeće primene brojača: brojač događaja, merač proteklog vremena, generator periodičnog signala, *watchdog* tajmer, merač realnog vremena, *baud rate* generator, generator impulsno-širinske modulacije.

### 9.3.1 Brojač događaja

U ovom režimu tajmer radi u brojačkoj funkciji i broji detektovane događaje na svom ulaznom takt signalu. Kod ove primene, ulazni takt signal na neki način

mora da bude u vezi sa događajem čiji broj realizacija se broji. Zbog ove specifične veze sa izvorom događaja, u opštem slučaju, takt signal neće i ne mora da ima osobinu periodičnosti, koja je inače karakteristična za takt signale. Dakle pojava impulsa može biti veoma asinhrona i neperiodična. Način rada brojača događaja biće ilustrovan na sledećem primeru. Razmotrimo aplikaciju gde imamo potrebu za brojanjem kutija koje se kreću na pokretnoj traci u nekoj fabrici. U ovom slučaju, pored pokretne trake bi bio postavljen odgovarajući senzor koji bi generisao impuls svaki put kada jedna kutija prođe na pokretnoj traci pored senzora. Ovaj impulсни signal (uz eventualnu dodatnu obradu zavisno od senzora) bi se zatim vodio na takt ulaz tajmera. U ovoj konfiguraciji, prolazak svake kutije bi izazvao uvećavanje vrednosti brojačkog registra unutar tajmera za jedan.

### 9.3.2 Merač proteklog vremena

Kada je ulazni takt tajmera periodičan signal učestanosti  $f$ , tajmer se može iskoristiti za merenje proteklog vremena između dva događaja. Ukoliko je učestanost ulaznog takt signala  $f$  Hz, njegova perioda će biti jednaka  $T = 1/f$  sekundi. U ovakvom sistemu, kada brojač izbroji  $k$  događaja na ulaznom takt signalu, on je istovremeno izmerio vreme od  $kT = k/f$  sekundi (podsetimo se slike 9.1).

#### Primer

Neka se oscilator učestanosti 32,768 kHz koristi se za generisanje takt signala, koji je povezan na ulaz preskalera sa faktorom deljenja 16.

a. Ako se tajmer resetuje nakon što brojač izbroji vrednost 0xA9B1 koliko će vremena proteći između dva reseta?

Učestanost i perioda takta koji zapravo okida brojač unutar tajmera u ovako konfigurisanom sistemu je:

$$f_{brojača} = 32,768 \text{ kHz} / 16 = 2,048 \text{ kHz}$$

,

$$T_{brojača} = 1/2,048 \text{ ms} = 488 \text{ } \mu\text{s}$$

Obzirom da je broj perioda ulaznog takt signala koje su bile odbrojane kada je brojač dostigao vrednost 0xA9B1 jednak  $0xA9B1 = 43441$ , a trajanje svake periode iznosi 488  $\mu\text{s}$ , proteklo vreme između dva reseta tajmera iznosi:

$$43441 * 488 \text{ } \mu\text{s} = 21,2 \text{ sekundi}$$

**b.** Da li se u datoj konfiguraciji može dobiti informacija da je proteklo 150 ms? Analizirati greške prilikom merenja vremenskog intervala od 150 ms?

Obzirom da je potrebno signalizirati protok vremena od 150 ms, brojač mora da odbroji ukupno:

$$(150 \text{ ms}) * (2,048 \text{ kHz}) = 307,2$$

perioda takt signala.

Obzirom da se u komparatorski registar može smestiti samo celobrojna vrednost, u ovom slučaju u njega je potrebno upisati vrednost 307.

Stvarni vremenski interval koji će biti izmeren u ovom slučaju iznosi:

$$307 * 1/2,048 \text{ ms} = 149,902 \text{ ms}$$

Apsolutna i relativna greška merenja 150 ms vremenskog intervala sa ovako konfigurisanim sistemom iznose  $98 \mu\text{s}$ , i 0.06 %, respektivno. U zavisnosti od krajnje aplikacije, ova greška može biti prihvatljiva ili ne.

### Primer

Podsetimo se primera brojanja kutija na pokretnoj traci u fabrici i pretpostavimo da bismo želeli da merimo i prosečno vreme koje protekne između prolaska dve kutije.

Najjednostavnije rešenje bi bilo da se računa prosečno vreme između prolaska dve kutije pomoću odgovarajućeg bafera koji se pomera u vremenu. Na primer, prosečno vreme između prolaska dve kutije izračunato usrednjavanjem vremena poslednjih 8 prolazaka. U ovom slučaju mogli bismo koristiti dodatni tajmer (jedan se već koristi za brojanje kutija) koji bi bio taktovan sa taktom poznate učestanosti. Koristeći izlaz senzora za detekciju prolaska kutije softver bi na svaki zahtev za prekidom pročitao tekuće stanje dodatnog tajmera.

Oduzimajući trenutnu vrednost dodatnog tajmera od vrednosti koja je bila očitana prilikom prethodnog generisanja zahteva za prekidom, moguće je izračunati vreme koje je proteklo između prolaska poslednje dve kutije. Ova vremena između prolazaka dve sukcesivne kutije bi zatim bila smeštana u jedan cirkularni bafer dubine 8. Računajući prosečnu vrednost vremena koja se trenutno nalaze unutar cirkularnog bafera mogli bismo izračunati prosečno vreme između prolaska dve kutije.

### Primer

Podesiti 8-bitni tajmer koji ima preskaler (sa faktorima 1, 8, 64, 256 i 1024) i komparatorski registar, tako da vreme između dva prekida bude  $T = 1 \text{ ms}$ . Učestanost takta je 16 MHz.

Rešenje:

$$f_{osc} = 16 \text{ MHz}, T = 1 \text{ ms}$$

Tajmer treba da odbroji  $n = T/T_{osc} = 16000$ . Ova vrednost je veća od opsega brojanja tajmera (256) pa uzimamo faktor deljenja  $p = 64$ . Tada,  $n = T/(pT_{osc}) = 250$  pa u komparatorski registar treba upisati ovu vrednost umanjenu za 1, tj. 249 (0xF9). Umanjenje za 1 se vrši jer je opseg brojanja od 0 do 249 što daje 250 koliko treba da se odbroji.

### 9.3.3 Generator periodičnog signala

Tajmerski modul je moguće koristiti i za generisanje odgovarajućih signala na nekom izlaznom pinu mikrokontrolera i na taj način realizovati jednostavan funkcijski generator. Izlazni pin može biti opšteg tipa ili namenski kao deo strukture tajmerskog modula.

#### Primer

Pomoću 8-bitnog tajmera koji ima komparatorski registar, realizovati na izlaznom pinu signal frekvencije 40 kHz, ako je frekvencija takta 16 MHz. Koja vrednost je potrebno da se upiše u komparatorski registar?

Svaka periodična povorka pravougaonih signala sa faktorom ispune 50 % (kao na slici 9.1) se sastoji od signala koji ima visok nivo polovinu periode i potom nizak nivo drugu polovinu periode. Da bismo generisali takav signal na nekom pinu potrebno je izmeriti poluperiodu i nakon toga invertovati vrednost signala na izlazu i to ponavljati periodično. Ovde obratiti pažnju da u ovakvim zadacima u jednačinama figuriše **poluperioda** izlaznog signala, a ne perioda!

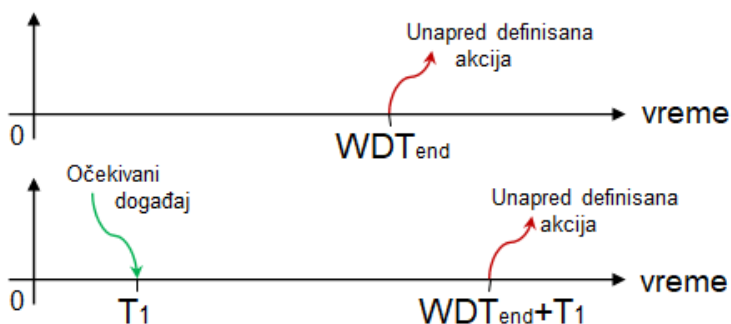
$$f_{osc} = 16 \text{ MHz}, f_{out} = 40 \text{ kHz}$$

Tajmer treba da odbroji  $n = T_{out}/2 * 1/T_{osc} = 200$ . Ova vrednost je manja od opsega brojanja tajmera (256) pa se može direktno upisati  $n - 1 = 199$ , tj. 0xC7. U slučaju da je  $n > 256$  potrebno je ili korišćenje preskalera, ili korišćenje softverske varijable, ili oba, za proširenje opsega.

### 9.3.4 Watchdog tajmer

*Watchdog* tajmer (skr. WDT) je posebna vrsta tajmera koji meri proteklo vreme između dva događaja. WDT se koristi da generiše odgovarajuću unapred definisanu akciju unutar sistema u slučaju da istekne unapred definisani period vremena (WDTend), a ne pojavi se događaj koji treba da reinicijalizuje WDT. Unapred definisana akcija može biti, na primer, da generiše zahtev za prekidom, da resetuje

čitav sistem ili neki njegov deo, da pokrene prelazak iz režima štednje energije u redovan režim rada mikrokontrolera itd. Ukoliko se unutar zadatog vremena  $WDT_{end}$  u nekom trenutku  $T_1$  pojavi očekivani događaj, tada je moguće uraditi reinicijalizaciju tajmera. Sada očekivani interval vremena za pojavu događaja postaje  $WDT_{end} + T_1$ . Na gornjem delu slike 9.7 je prikazan slučaj kada se očekivani događaj nije desio u toku unapred zadatog intervala trajanja  $WDT_{end}$ . Tada je aktivirana unapred definisana akcija u trenutku  $WDT_{end}$ . U donjem delu slike se vidi pojava očekivanog događaja u trenutku  $T_1$  i potom reinicijalizacija tajmera koja sada produžava vremenski interval očekivanja na  $WDT_{end} + T_1$ .



Slika 9.7: Vremenska ilustracija rada WDT

Maksimalno dozvoljeno vreme u okviru kojeg se očekuje pojava događaja ( $WDT_{end}$ ) koji će reinicijalizovati watchdog tajmer se podešava softverski. Unapred definisana akcija će se izvršiti u slučaju da istekne unapred definisani period vremena ( $WDT_{end}$ ), a ne dođe do pojave događaja koji reinicijalizuje WDT. Da bi smo sprečili WDT da aktivira unapred definisanu akciju, nakon detekcije pojave odgovarajućeg događaja, potrebno je reinicijalizovati WDT. Ovo se najčešće radi unutar prekidnog podprograma koji je asociran događaju koji treba da aktivira reinicijalizaciju WDT-a.

Kao primer upotrebe WDT-a razmotrimo sledeću situaciju. Posmatrajmo savremeni automobil čija vrata se otvaraju daljinskim upravljačem nakon što korisnik pritisne odgovarajući taster. Ova akcija otključava vrata da bi korisnik mogao da ih otvori i uđe u automobil. Međutim, ukoliko se nakon nekog unapred definisanog perioda vremena, na primer 30 sekundi, vrata ne otvore brava koja ih zaključava će se opet aktivirati. U ovom primeru, WDT se startuje u trenutku pritiska tastera na daljinskom upravljaču. Podrazumevana akcija koje se izvršava ukoliko

WDT dostigne svoju predefinisanu vrednost (30 sekundi) je zaključavanje vrata. Očekivani događaj koji u ovom primeru isključuje WDT je otvaranje vrata.

Sličan primer imamo i kada posmatramo sistem pristupa nekom objektu kojem se pristupa upotrebom RFID kartice. Prisanjanje kartice u zonu čitača otključava vrata za pristup. Ako se vrata ne otvore u zadatom vremenu, aktivira se podrazumevana akcija - zaključavanje vrata. Ako se vrata otvore pre isteka podrazumevanog maksimalnog vremena onda se reinicijalizuje WDT. Pri tome se WDT i zaustavlja dok su vrata otvorena. Tek ponovnim zatvaranjem vrata se WDT postavlja u stanje čekanja na događaj prislanjanja RFID kartice u zonu čitača.

Za WDT se može reći i da je vrsta sigurnosnog uređaja. U normalnim okolnostima, podrazumeva se da će WDT biti servisiran (reinicijalizovan, zaustavljen ili resetovan) pre isteka definisanog perioda. U protivnom, nešto nije u redu sa sistemom i potrebno je izvršiti odgovarajući oporavak sistema izvršavanjem podrazumevane akcije. U embedded sistemima, watchdog tajmeri koriste se u različite svrhe: za zaustavljanje programa koji odstupaju od normalnog režima rada, za izlazak iz mrtvih petlji unutar programa, za detekciju nepravilnih transakcija na sistemskim magistralama itd.

### 9.3.5 Merač realnog vremena

Jedna od čestih primena tajmera unutar embedded sistema jeste merenje protoka realnog vremena. Merač protoka realnog vremena (eng. *Real-Time Clock* - RTC) je tajmer koji je konfigurisan da meri prolazak sekundi, minuta, sati, itd. Vrlo često, potrebna rezolucija RTC modula mora se spustiti na delove sekunde, i takvi RTC moduli se nazivaju hronometrima (eng. *chronometer*). Sa druge strane, nekada je potrebno meriti vremena koja se mere danima, nedeljama, mesecima, pa čak i godinama, kada se RTC moduli nazivaju RTC kalendarima (eng. *Real-Time Clock Calendar* - RTCC). Iako se funkcionalnost RTCC u potpunosti može realizovati softverski, postojanje posebnog tajmera namenjenog za ove svrhe može uštedeti veliki broj CPU ciklusa.

### 9.3.6 Baud rate generator

Tajmeri se takođe mogu iskoristiti za generisanje periodičnih signala koji su neophodni u serijskim komunikacionim protokolima da bi se obezbedila željena brzina razmene podataka. U ovim aplikacijama, učestanost ulaznog takt signala se deli sa željenom brzinom slanja podataka (eng. *Baud Rate*) da bi se odredio potreban broj brojanja između slanja sukcesivnih bita, odnosno da bi se odredilo vreme trajanja jednog bita (eng. *Bit Time*). Izračunata vrednost se zatim smešta

u komparatorski registar tajmera i na taj način se generiše zahtev za prekidom u tačno potrebnim trenucima.

### Učestanost takta

Pored uobičajene učestanosti takta od 12 ili 16 MHz koriste se i druge učestanosti. Na primer, 11,0592 MHz se često koristi jer predstavlja celobrojni multipl brzina serijske komunikacije (npr.  $110592/2400 = 46$ ). Slično se koristi i kristal učestanosti 18,432 MHz. Pored toga važi i  $11059200 = 214 * 33 * 52$  Tako da se korišćenjem ovih faktora može dobiti 86400 (broj sekundi u danu) što je pogodno za RTCC primene. Dalje, 15-bitni brojač koji radi na 32768 Hz će imati pretek svake sekunde što je pogodno za sat!

### 9.3.7 Generator impulsno-širinske modulacije

Impulsno širinska modulacija (eng. *Pulse Width Modulation* - PWM) se često koristi unutar embeded sistema. Ova modulacija može se realizovati korišćenjem tajmera. PWM modul generiše periodičan signal čiji se faktor ispune (eng. *Duty Cycle*) može kontrolisati od strane mikrokontrolera, odnosno softvera. Obzirom da se i učestanost ulaznog takta tajmera može kontrolisati softverski (izborom odgovarajućeg izvora takt signala pomoću multipleksa), kod ove primene tajmera moguće je softverski kontrolisati i učestanost i faktor ispune generisanog periodičnog signala.

PWM signali imaju veliku primenu u sistemima koji treba da kontrolišu količinu energije koja se prenosi ka nekom uređaju. Energija PWM signala je funkcija njegovog faktora ispune. Primeri primena u kojima se koriste PWM signali su: kontrola brzine jednosmernog motora, kontrola temperature grejača, kontrola inteziteta svetlosti LED dioda itd.

Da bi se tajmer koristio kao PWM generator, potrebno je da se u komparatorski registar upiše tzv. vršna vrednost. Dok god je stanje brojača manje od ove vršne vrednosti dotle je izlaz visok i obrnuto, ukoliko je stanje brojača veće od vršne vrednosti izlaz je nizak. Jasno je da se promenom vršne vrednosti menja faktor ispune. Rezolucija PWM signala zavisi od broja bita brojačkog registra. Učestanost PWM signala može se kontrolisati pomoću preskalera i selektora takt signala.

### Primer

Kao primer korišćenja PWM signala, razmotrimo kontrolu inteziteta svetla svetleće diode (LED), tzv. dimer. Sjajnost LED direktno je srazmerna jačini struje koja protiče kroz nju. Obične LED traže nekoliko mA struje za osvetljaj i njihov često možemo direktno povezati na port mikrokontrolera. Ukoliko su struje ipak veće, reda stotinak mA, tada se obično pribegava upotrebi nekog drajvera (npr. ULN2003) pogotovo ako se koristi više dioda u nizu. Ovaj 7-kanalni drajver koristi Darlington spregu tranzistora koja ima veliko strujno pojačanje na svakom kanalu. Međutim postoje i tzv. visoko sjajne LED. Pretpostavimo da u ovom primeru koristimo LED koja može da generiše svetlost maksimalne sjajnosti od 300 lumena (lm) kada kroz nju protiče struja jačine oko 1 A. Pretpostavimo da želimo da kontrolišemo sjajnost diode pomoću mikrokontrolera u 6 mogućih nivoa od isključene do maksimalne sjajnosti: 0 lm, 50 lm, 100 lm, 150 lm, 200 lm, 250 lm i 300 lm. Pretpostavimo da koristimo neki standardni mikrokontroler sa 3.3 V izlazima koji mogu da generišu maksimalno 20 mA, a da je napajanje LED diode 5 V.

Obzirom da mikrokontroler radi na 3.3 V i može da generiše samo 20 mA po jednom izlazu, a LED dioda se napaja sa 5 V napajanjem i zahteva struju jačine do 1 A moramo isprojektovati odgovarajući sprežni interfejs. Jedno moguće rešenje je na bazi snažnog mosfet tranzistora čijom strujom drejna, u čijem kolu je LED, ćemo upravljati na bazi PWM signala. Primetimo da svaki sledeći nivo sjajnosti zahteva povećanje od 1/6 faktora ispunje. Pod pretpostavkom da koristimo 8-bitni tajmer, vršne vrednosti koje bi bilo potrebno upisati u registar PWM modula da bi se postigle željene vrednosti sjajnosti su: 0, 42, 85, 127, 170 i 212. Periodu takta tajmera treba odabrati dovoljno velikom tako da ne bude vidljivog treptanja LED.

Danas na raspolaganju postoje i različiti gotovi LED drajveri za dimer aplikacije kao moduli ili kao integrisana kola (npr. MAXIM 16834).

## 9.4 Tajmeri/brojači mikrokontrolera ATmega328P

Unutar mikrokontrolera ATmega328P imamo tri tajmera/brojača (T/C):

1. 8-bitni T/C 0
2. 16-bitni T/C 1
3. 8-bitni T/C 2

Svaki tajmer ima odgovarajući ulaz sa nekog pina koji se najčešće koristi kao ulaz za takt ili kao brojački ulaz. Pored toga ima i odgovarajuće izlaze na pinovima na kojima je moguće generisati neki signal upotrebom tajmera. Na ovaj način je moguća realizacija svih primena opisanih u prethodnim sekcijama.



U tabeli vektora prekida je rezervisano 11 T/C prekida (1 za WDT, po tri za T/C 0 i 2 i 4 za T/C 1). Ovo je detaljnije obrađeno u prethodnom poglavlju 8 o prekidima. U nastavku će biti dato objašnjenje rada na primeru T/C 0 ali slično je i sa ostalim T/C.

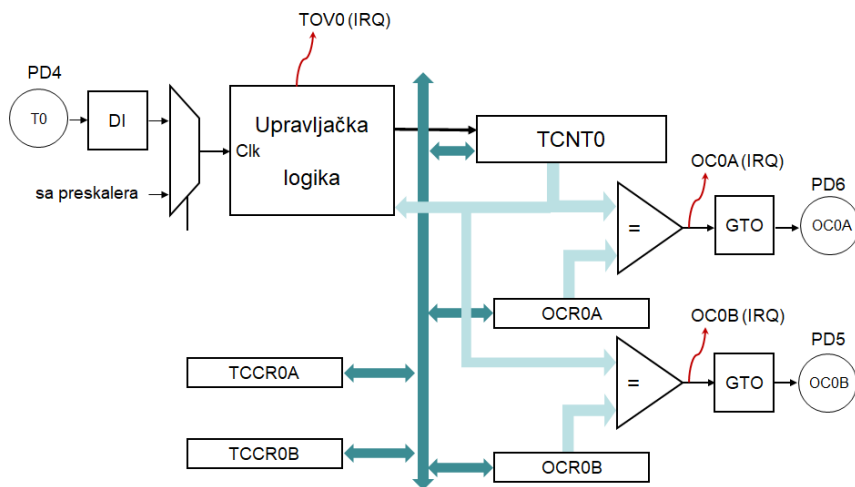
Blok šema strukture T/C 0 data je na slici 9.8. Osnovne karakteristike T/C 0 su:

- Višestruke mogućnosti izbora izvora takta sa preskalera uključujući i detekciju ivica (DI) na spoljašnjem pinu PD4 (T0 ulaz)
- Dve nezavisne izlazne komparatorske (eng. *output compare*) jedinice koje sadrže i dva komparatorska registra (OCR0A i OCR0B)
- Mogućnost resetovanja tajmera i upisa početne vrednosti prilikom dostizanja vrednosti u komparatorskom registru (eng. *auto-reload* opcija)
- Posедуje modul za impulsno - širinsku modulaciju (PWM) i ima mogućnost promenljive PWM periode
- Mogućnost realizacije frekvencijskog generatora upotrebom blokova za generisanje talasnih oblika (GTO) na pinovima PD5 (OC0B izlaz) i PD6 (OC0A izlaz).
- Tri nezavisna izvora prekida (TOV0 - usled preteka tajmera bez komparatora; OC0A, OC0B - dva usled preteka tajmera sa komparatorom)
- podešavanje faktora deljenja, modova rad i dr. pomoću registara TCCR0A i TCCR0B
- upravljačka logika prati stanje 8-bitnog brojačkog registra TCNT0 i generiše TOV0 prekid (ukoliko je dozvoljen) kada se desi pretek brojača
- dozvole prekida se podešavaju u registru TIMSK0 (eng. *Timer/Counter Interrupt Mask Register 0*), a u registru TIFR0 (eng. *Timer/Counter Interrupt Flag Register 0*) je moguće očitati indikatore koji pokazuju da se desio neki od tri moguća prekida T/C 0.

### 9.4.1 Modovi rada

Tajmer/brojač 0 ima 4 karakteristična načina (moda) rada:

- Normalni,



Slika 9.8: Blok šema T/C 0

- CTC (eng. *Clear Timer on Compare Match*),
- Brzi PWM, i
- Fazno korektni PWM.

### Normalni mod

U ovom modu rada smer brojanja je uvek na gore i stanje brojača se nikada ne gubi, već dolazi do preteka brojača kada dođe do maksimalne vrednosti (0xFF), nakon čega nastavlja da broji od 0x00. U ovom modu rada, TOV0 indikator (eng. *Timer/Counter Overflow Flag 0*) će biti setovan u trenutku kada vrednost brojača prelazi sa 0xFF na 0x00. Ovaj indikator predstavlja jedan od 3 moguća izvora prekida koji potiču od tajmera 0 i nalazi se u TIFR0 registru.

### CTC mod

Kod ovog načina rada registar OCR0A se koristi kako bi se odredio rezoluciju tajmera. Brojač se automatski postavlja na 0 kada vrednost brojačkog registra TCNT0 dostigne vrednost komparatorskog registra OCR0A. Tada se automatski setuje indikator OCF0A, a može biti i aktiviran prekid. Ukoliko se koristi gene-

risanje talasnih oblika na izlaznom pinu, može da se precizno podesi učestanost signala na izlaznom pinu OC0A tako što se dozvoli automatska promena stanja pina svaki put kada dode do poklapanja vrednosti. Učestanost izlaznog signala računa se prema jednačini:

$$f_{OC0A} = \frac{f_{clk}}{2N(1 + OCR0A)}$$

gde je N faktor deljenja učestanosti (1, 8, 64, 256 ili 1024). Ovaj režim rada ćemo koristiti u okviru zadataka na vežbama.

### Brzi PWM mod

U ovom režimu rada generator signala generiše PWM signal. Početna vrednost brojača je 0x00, a krajnja vrednost je 0xFF. Vrednost izlaznog pina OC0A, odnosno OC0B se postavlja na 1 pri preteku tajmera (prelasku sa 0xFF na 0x00), a vraća se na 0 pri dostizanju vrednosti u OCR0A, odnosno OCR0B registru. Moguća je i opcija pri kojoj je logika na izlaznim pinovima invertovana.

### Fazno korektni PWM mod

U ovom modu tokom jedne periode PWM signala tajmer broji prvo unapred od 0x00 do 0xFF, a zatim unazad od 0xFF do 0x00. Promene stanja na izlazu se dešavaju u trenucima dostizanja vrednosti u OCR0A, odnosno OCR0B registru.

Detaljnija objašnjenja modova rada kao drugih aspekata rada T/C 0 i ostalih tajmera je moguće pogledati u dokumentaciji proizvođača [38].

## 9.5 Pitanja

1. Nabrojati neke od najčešće korišćenih aplikacija tajmera i brojača.
2. Nacrtati generalnu strukturu tajmera.
3. Objasniti osnovne komponente tajmera.
4. Nacrtati i objasniti vremenske dijagrame aktivacije prekida tajmera bez komparatora.
5. Nacrtati i objasniti vremenske dijagrame aktivacije prekida tajmera sa komparatorom.

6. Kako se može proširiti opseg brojanja tajmera? Objasniti svaki od načina.
7. Dat je tajmer sa preskalerom koji je konfigurisan da deli ulazni takt sa faktorom \_\_\_\_\_. Brojač unutar tajmera je \_\_\_-bitni. Koji je maksimalni broj događaja koji se može izbrojati sa ovako konfigurisanim tajmerom?
8. Dat je tajmer sa preskalerom koji je konfigurisan da deli ulazni takt sa faktorom \_\_\_\_\_. Koji broj treba upisati u komparatorski registar da bi tajmer nakon odbrojanja 75000 događaja generisao zahtev za prekidom?
9. Za mikrokontroler važi  $f_{osc}=X$  kHz i preskaler deli sa faktorom  $p$ . Ako se tajmer resetuje nakon odbrojanih 10000 impulsa, koliko je vreme između 2 resetovanja tajmera?
10. Objasniti svrhu, principe rada i oblasti primene watchdog tajmera.
11. Kakvi mogu biti merači realnog vremena?
12. Čemu služe baud rate generatori?
13. Objasniti princip rada impulsno-širinskog generatora i navesti neke oblasti primene.
14. Navesti tajmere/brojače ATmega328P i dati osnovne karakteristike za T/C 0.
15. Objasniti koja tri prekida može T/C 0 da generiše.
16. Objasniti blok šemu i način rada T/C 0.
17. Koje sve režime rada podržava T/C 0? Objasniti ih.
18. Neka je  $f_{osc}=16$  MHz, koliko je potrebno impulsa da odbroji T/C 0 u CTC modu da bi između dva prekida proteklo 1 ms? Kako je to moguće realizovati?



# Bibliografija

- [1] Arthur W. Burks, Herman H. Goldstine, and John von Neumann, "Preliminary discussion of the logical design of an electronic computing instrument", The Institute of Advanced Study, Princeton, USA, 1946-47.
- [2] J. Biggs, J. Myers, J. Kufel et al. A natively flexible 32-bit Arm microprocessor, *Nature* 595, pp. 532–536, 2021.
- [3] F. Arute, K. Arya, R. Babbush et al. Quantum supremacy using a programmable superconducting processor. *Nature* 574, pp. 505–510, 2019.
- [4] J. Hochstetter, R. Zhu, A. Loeffler et al. Avalanches and edge-of-chaos learning in neuromorphic nanowire networks. *Nat Commun* 12, 4008, 2021.
- [5] B. B. Brey, *Intel Microprocessors*, 8th Edition, Pearson, 2009.
- [6] S. Harris, D. Hariss, *Digital Design and Computer Architecture*, Arm edition, Morgan Kaufmann, 2015.
- [7] D. Patterson, J.L. Hennessy, *Computer Organization and Design RISC-V Edition: The Hardware Software Interface*, The Morgan Kaufmann Series in Computer Architecture and Design, 2nd Edition, 2021.
- [8] M. M. Mano, C. R. Kime, T. Martin, *Logic and Computer Design Fundamentals*, Fifth edition, Pearson, 2015.
- [9] R. S. Sandige, M. L. Sandige, *Fundamentals of Digital and Computer Design with VHDL*, McGraw Hill, 2012.
- [10] L. Null, J. Lobur *Essentials of Computer Organization and Architecture*, 5th Edition, Jones and Barlett Publishers, 2018.

- [11] E.O. Hwang, Digital Logic and Microprocessor Design With VHDL with Interfacing, Cengage Learning, Second Edition, 2017.
- [12] I. Mezei "Evolution of an educational microprocessor", Computer Applications in Engineering Education, 28(5), Wiley, pp. 1265-1277, 2020.
- [13] W. Stallings, Computer organization and architecture, 9th Edition, Pearson, 2013.
- [14] M. Abd-El-Barr, H. El-Rewini, Fundamentals Of Computer Organization And Architecture, John Wiley and Sons, Inc., 2005.
- [15] C. Hamacher, Z. Vranesic, S. Zaky, N. Manjikian, Computer Organization and Embedded Systems, 6th Edition, McGraw-Hill, 2012.
- [16] Manuel Jiménez, Rogelio Palomera, Isidoro Couvertier, Introduction to Embedded Systems, Springer, 2014.
- [17] D. Patterson, "Reduced Instruction Set Computers Then and Now" in Computer, vol. 50, no. 12, pp. 10-12, 2017. <https://doi.ieeecomputersociety.org/10.1109/MC.2017.4451206>
- [18] D. A. Patterson, J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, Fourth Edition, Morgan Kaufmann, 2011.
- [19] J.L. Gustafson, Moore's Law. In: D. Padua D. (eds) Encyclopedia of Parallel Computing. Springer, 2011.
- [20] S.A. McKee, R.W. Wisniewski, Memory Wall. In: D. Padua (eds) Encyclopedia of Parallel Computing. Springer, 2011.
- [21] P. Bose, Power Wall. In: D. Padua (eds) Encyclopedia of Parallel Computing. Springer, 2011.
- [22] A. Tsakyridis, T. Alexoudi, A. Miliou, N. Pleros, and C. Vagionas, "10 Gb/s optical random access memory (RAM) cell," Opt. Lett. 44, pp. 1821-1824, 2019.
- [23] W.W. Royce, "Managing the Development of Large Software Systems: Concepts and Techniques", IEEE Wescon Proc., Aug. 1970.
- [24] C.G. Bell, A. Newell, Computer structures: Readings and Examples, McGraw-Hill, 1971.

- [25] S. Brown, Z. Vranesic, Fundamentals of Digital Logic with VHDL Design, McGraw Hill, 2009.
- [26] D. L. Perry, VHDL: Programming by Example, Fourth edition, McGraw Hill, 2002.
- [27] L. Null, J. Lobur, Essentials of Computer Organization and Architecture, Jones and Bartlet Learning, 3rd Ed., 2010.
- [28] D. Patterson, J.L. Hennessy, Computer Organization and Design RISC-V Edition: The Hardware Software Interface, The Morgan Kaufmann Series in Computer Architecture and Design, 1st Edition, 2017.
- [29] S. L. Harris, D. Harris, Digital Design and Computer Architecture, RISC-V Edition, Morgan Kaufman, 1st edition, 2021.
- [30] D. A. Patterson, J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, ARM edition, The Morgan Kaufmann Series in Computer Architecture and Design, 2016.
- [31] D. Kushner, "The making of arduino", IEEE Spectrum 26, 2011.
- [32] M. Banzi, "How Arduino is open-sourcing imagination", TEDtalk, Scotland, 2012. <https://www.youtube.com/watch?v=UoBUXOOdLXY> (datum pristupa: 31.01.2022.)
- [33] Sparkfun, "Arduino shields v2", <https://learn.sparkfun.com/tutorials/arduino-shields-v2>, (datum pristupa: 01.02.2022.)
- [34] Arduino Uno Rev3 schematic, [https://www.arduino.cc/en/uploads/Main/Arduino\\_Uno\\_Rev3-schematic.pdf](https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf) (datum pristupa: 02.02.2022.)
- [35] Arduino software, <https://www.arduino.cc/en/Main/Software> (datum pristupa: 02.02.2022.)
- [36] 74HC595 datasheet, Philips, 1998, <https://www.arduino.cc/en/uploads/Tutorial/595datasheet.pdf> (datum pristupa: 03.03.2022.)
- [37] 74HC165 datasheet, Texas Instruments, 2015 <https://www.ti.com/lit/ds/symlink/sn74hc165.pdf> (datum pristupa: 03.03.2022.)
- [38] ATmega328 datasheet, Atmel, 2015, [https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P\\_Datasheet.pdf](https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf) (datum pristupa: 15.03.2022.)



- [39] Tutorials 83, Fundamentals of RS-232 serial communications, Analog Devices, 2001. <https://www.maximintegrated.com/en/design/technical-documents/tutorials/8/83.html> (datum pristupa: 31.03.2022.)