

Integrirani računarski sistemi

Tajmeri

Odsek za elektroniku

Univerzitet u Beogradu - Elektrotehnički fakultet

poslednja izmena 3. mart 2022



1 Tajmer A

2 Tajmer B

3 Primeri

- Debaunsiranje
- Multipleksiranje LED displeja
- PWM
- Merenje učestanosti



1 Tajmer A

2 Tajmer B

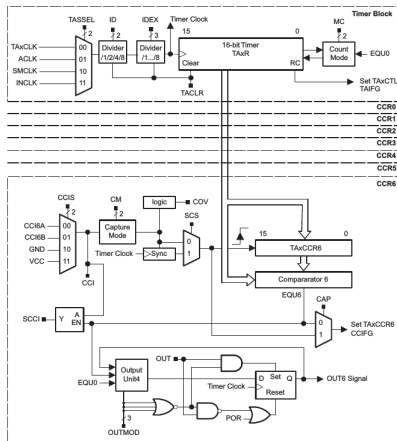
3 Primeri

- Debaunsiranje
- Multipleksiranje LED displeja
- PWM
- Merenje učestanosti



Tajmer A

Tajmer A je 16-bitni tajmer sa nekoliko *capture/compare* blokova, čiji broj varira kod različitih predstavnika familije od 3 do 7



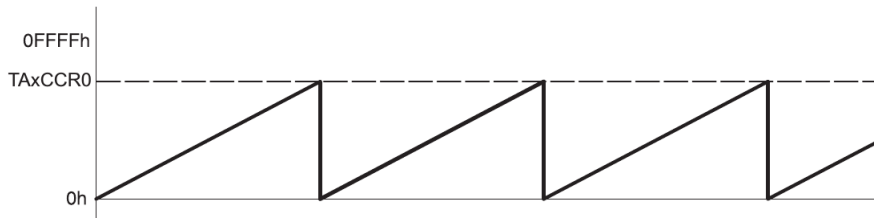
Modovi rada brojača TAxR

MC	Mode	Description
00	Stop	The timer is halted.
01	Up	The timer repeatedly counts from zero to the value of TAxCCR0
10	Continuous	The timer repeatedly counts from zero to 0FFFFh.
11	Up/down	The timer repeatedly counts from zero up to the value of TAxCCR0 and back down to zero.

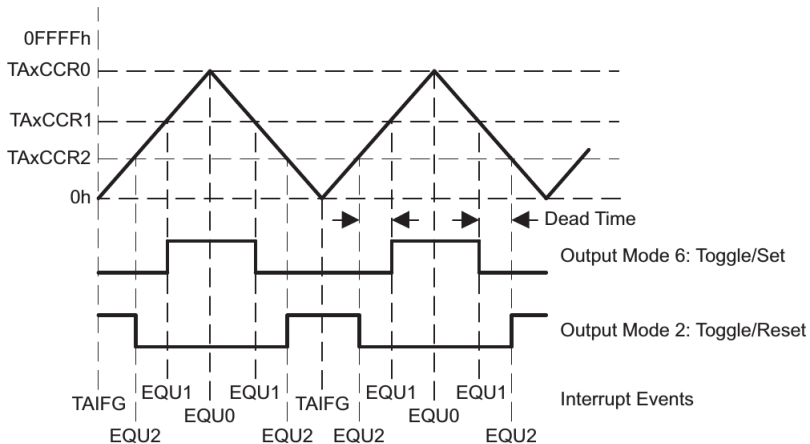


Brojač na gore (UP mod)

Po dostizanju vrednosti `TAxCCR0` setuje se `CCIFG` fleg a pri resetovanju brojača na `0x0000` setuje se `TAxIFG` fleg



UP/DOWN brojač



CAPTURE/COMPARE blokovi

Svaki CC blok može da radi u CAPTURE ili COMPARE modu. To je određeno bitom CAP u kontrolnom registru TAxCTLn

CAPTURE mod se koristi za merenje učestanosti, tj. intervala između uzastopnih događaja na nekom pinu. Svaka promena na određenom pinu kopira trenutnu vrednost brojača TAxR u odgovarajući TAxCCRn registar is setuje odgovarajući CCIFG fleg.

COMPARE mod se koristi za generisanje PWM-a i signala različitih učestanosti. Svako izjednačavanje vrednosti brojača TAxR sa vrednošću u TAxCCRn registru setuje odgovarajući TAxCCRn CCIFG fleg



Izlazna jedinica

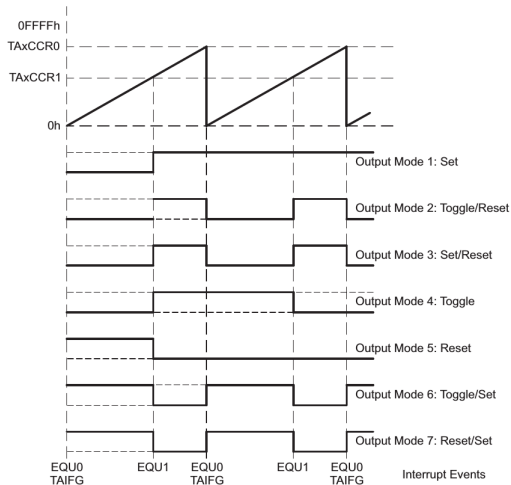
Svaki CC blok sadrži izlaznu jedinicu koja se koristi za generisanje signala kao što je PWM

Izlazna jedinica može da radi u osam različitih modova

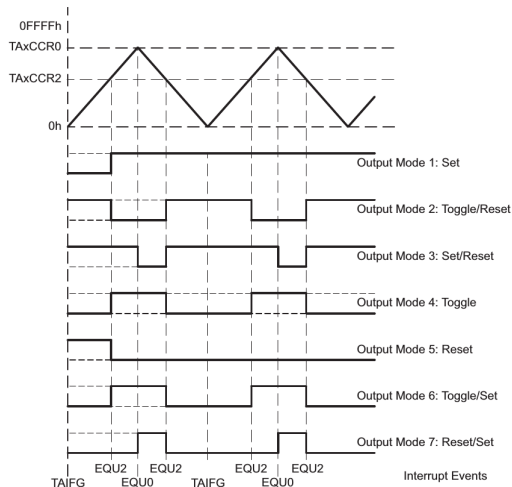
OUTMODx	Mode	Description
000	Output	The output signal OUTn is defined by the OUT bit. The OUTn signal updates immediately when OUT is updated.
001	Set	The output is set when the timer <i>counts</i> to the TAxCCRn value. It remains set until a reset of the timer, or until another output mode is selected and affects the output.
010	Toggle/Reset	The output is toggled when the timer <i>counts</i> to the TAxCCRn value. It is reset when the timer <i>counts</i> to the TAxCCR0 value.
011	Set/Reset	The output is set when the timer <i>counts</i> to the TAxCCRn value. It is reset when the timer <i>counts</i> to the TAxCCR0 value.
100	Toggle	The output is toggled when the timer <i>counts</i> to the TAxCCRn value. The output period is double the timer period.
101	Reset	The output is reset when the timer <i>counts</i> to the TAxCCRn value. It remains reset until another output mode is selected and affects the output.
110	Toggle/Set	The output is toggled when the timer <i>counts</i> to the TAxCCRn value. It is set when the timer <i>counts</i> to the TAxCCR0 value.
111	Reset/Set	The output is reset when the timer <i>counts</i> to the TAxCCRn value. It is set when the timer <i>counts</i> to the TAxCCR0 value.



Rad izlaznog bloka za brojač u modu brojanja na gore



Rad izlaznog bloka za brojač u UP/DOWN modu brojanja



Prekidi Tajmera A

Dva prekidna vektora su povezana sa tajmerom A:

- **TAxCCR0** vektor za **TAxCCR0 CCIFG** fleg
- **TAxIV** vektor za ostale **CCIFG** flegove i **TAxIFG**

U **CAPTURE** modu **CCIFG** fleg se setuje kada se na spoljni događaj registar **TAxCCRn** napuni vrednošću brojača **TAxR**

U **COMPARE** modu **CCIFG** fleg se setuje kada vrednost brojača **TAxR** dostigne vrednost u **TAxCCRn** registru

TAIFG fleg se setuje kada vrednost brojača **TAxR** dostigne 0x0000

Unutar **TAxIV** prekida se poliranjem određuje koji je događaj izazvao prekid

Registar prekidnih vektora

15	14	13	12	11	10	9	8
TAIV							
r0	r0	r0	r0	r0	r0	r0	r0
7	6	5	4	3	2	1	0
TAIV							
r0	r0	r0	r0	r-(0)	r-(0)	r-(0)	r0

Table 17-8. TAxIV Register Description

Bit	Field	Type	Reset	Description
15-0	TAIV	R	0h	<p>Timer_A interrupt vector value</p> <p>00h = No interrupt pending</p> <p>02h = Interrupt Source: Capture/compare 1; Interrupt Flag: TAxCCR1 CCIFG; Interrupt Priority: Highest</p> <p>04h = Interrupt Source: Capture/compare 2; Interrupt Flag: TAxCCR2 CCIFG</p> <p>06h = Interrupt Source: Capture/compare 3; Interrupt Flag: TAxCCR3 CCIFG</p> <p>08h = Interrupt Source: Capture/compare 4; Interrupt Flag: TAxCCR4 CCIFG</p> <p>0Ah = Interrupt Source: Capture/compare 5; Interrupt Flag: TAxCCR5 CCIFG</p> <p>0Ch = Interrupt Source: Capture/compare 6; Interrupt Flag: TAxCCR6 CCIFG</p> <p>0Eh = Interrupt Source: Timer overflow; Interrupt Flag: TAxCTL TAIFG; Interrupt Priority: Lowest</p>



Primer prekidnih rutina tajmera A

```
; Interrupt handler for TA0CCR0 CCIFG.                                Cycles
CCIFG_0_HND
;      ...      ; Start of handler Interrupt latency      6
;      RETI      5

; Interrupt handler for TA0IFG, TA0CCR1 through TA0CCR6 CCIFG.

TA0_HND      ...      ; Interrupt latency      6
      ADD      &TA0IV,PC      ; Add offset to Jump table      3
      RETI      ; Vector 0: No interrupt      5
      JMP      CCIFG_1_HND      ; Vector 2: TA0CCR1      2
      JMP      CCIFG_2_HND      ; Vector 4: TA0CCR2      2
      JMP      CCIFG_3_HND      ; Vector 6: TA0CCR3      2
      JMP      CCIFG_4_HND      ; Vector 8: TA0CCR4      2
      JMP      CCIFG_5_HND      ; Vector 10: TA0CCR5      2
      JMP      CCIFG_6_HND      ; Vector 12: TA0CCR6      2

TA0IFG_HND      ; Vector 14: TA0IFG Flag
...      ; Task starts here
      RETI      5

CCIFG_6_HND      ; Vector 12: TA0CCR6
...      ; Task starts here
      RETI      ; Back to main program      5

CCIFG_5_HND      ; Vector 10: TA0CCR5
...      ; Task starts here
      RETI      ; Back to main program      5

CCIFG_4_HND      ; Vector 8: TA0CCR4
...      ; Task starts here
      RETI      ; Back to main program      5

CCIFG_3_HND      ; Vector 6: TA0CCR3
...      ; Task starts here
      RETI      ; Back to main program      5

CCIFG_2_HND      ; Vector 4: TA0CCR2
...      ; Task starts here
      RETI      ; Back to main program      5

CCIFG_1_HND      ; Vector 2: TA0CCR1
...      ; Task starts here
      RETI      ; Back to main program      5
```



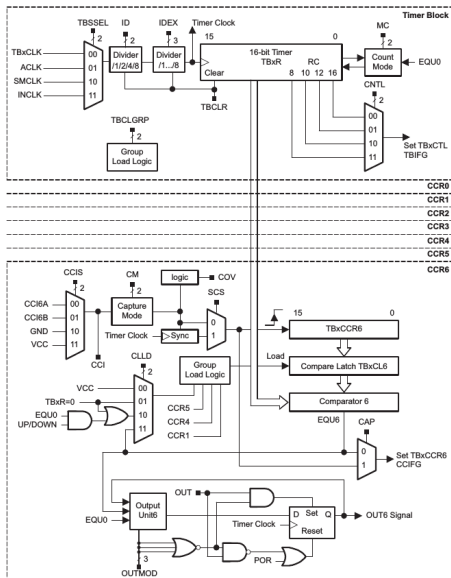
1 Tajmer A

2 Tajmer B

3 Primeri

- Debaunsiranje
- Multipleksiranje LED displeja
- PWM
- Merenje učestanosti





Vrlo sličnih karakteristika kao i tajmer A

Razlike

- izbor širine brojačkog registra na 8,10,12 ili 16 bita
- mogućnost grupisanja više CC jedinica
- svi izlazi mogu da se stave u stanje visoke impedanse
- TBxCCRn registri su duplo baferisani



1 Tajmer A

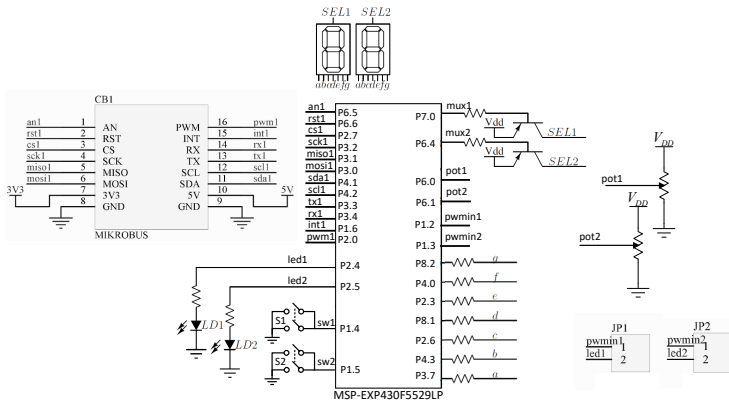
2 Tajmer B

3 Primeri

- Debaunsiranje
- Multipleksiranje LED displeja
- PWM
- Merenje učestanosti



Opis hardvera



1 Tajmer A

2 Tajmer B

3 Primeri

- Debaunsiranje
- Multipleksiranje LED displeja
- PWM
- Merenje učestanosti



Postoji više načina na koje tajmer može da se iskoristi za debaunsiranje tastera

- periodično generisanje prekida, gde se u okviru prekidne rutine očitava stanje ulaznih pinova
 - može samo da se poredi trenutno stanje sa prethodnim
 - može i malo naprednije (Ganssle debaunsiranje)
- po prijemu prekida sa ulaznog porta pokreće se rad tajmera, a po isteku perioda brojanja očitaju se stanja tastera



Zadatak – Detekcija pritiska tastera (samo tajmer)

Zadatak

Napisati program za razvojni sistem koji na svaki pritisak jednog tastera menja stanje uključenosti LED diode

Napomene

Ispitivanje da li je taster pritisnut vršiti periodičnim poliranjem pomoću tajmera

- Potrebno je inicijalizovati tajmer da generiše prekid sa periodom od 32 ms
- U okviru prekidne rutine tajmera treba očitati stanje tastera i uporediti sa prethodnim stanjem

Rešenje

`button-toggle-timer`

Zadatak – Detekcija pritiska tastera (prekid porta i tajmer)

Zadatak

Napisati program za razvojni sistem koji na svaki pritisak jednog tastera menja stanje uključenosti LED diode

Napomene

Ispitivanje da li je taster pritisnut vršiti pomoću prekidnih rutina porta i tajmera

- U prekidnoj rutini porta
 - pokreće se tajmer sa trajanjem 32 ms
 - zabranjuje dalje generisanje prekida porta
- U prekidnoj rutini tajmera
 - očitava se stanje tastera
 - isključuje se tajmer
 - dozvoljava se prekid porta

Zadatak – Detekcija pritiska tastera (prekid porta i tajmer)

Zadatak

Napisati program za razvojni sistem koji na svaki pritisak jednog tastera menja stanje uključenosti LED diode

Napomene

Ispitivanje da li je taster pritisnut vršiti pomoću prekidnih rutina porta i tajmera

Rešenje

`button-toggle-isr-timer`



1 Tajmer A

2 Tajmer B

3 Primeri

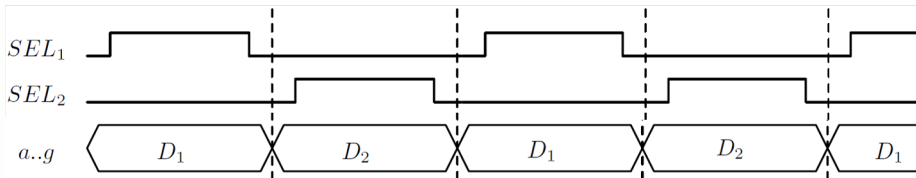
- Debaunsiranje
- Multipleksiranje LED displeja
- PWM
- Merenje učestanosti



Multipleksiranje LED displeja

Sedmosegmentni LED displeji koji su postavljeni na ploči koriste zajedničke linije $a..g$

- za istovremeni prikaz različitih cifara potrebno je vršiti multipleksiranje displeja
- treperenje se ne vidi ukoliko se prikaz na displejima menja sa učestanošću većom od ~ 100 Hz



Zadatak – Prikaz na multipleksiranom LED displeju

Zadatak

Napisati program za razvojni sistem koji na multipleksiranom LED displeju prikazuje broj 12

Napomene

Multipleksiranje LED displeja vršiti pomoću tajmera

- pošto su nam potrebne cifre za prikaz, potrebno je iz broja izdvojiti pojedinačne cifre
 - `div i mod`
 - `shift and add 3` (*double-dabble*)
 - MSP430 DADD



Shift and add 3 (*double-dabble*)

- iterativan algoritam, ne zavisi od dostupnog hardvera
- sastoji se iz serije pomeranja bita i sabiranja

Ako je binarni zapis $BIN = bin_7..bin_0$ a BCD zapis $BCD = bcd_{11}..bcd_0 = D_2D_1D_0$ (posmatramo samo trocifrene brojeve manje od 256), algoritam je

- $bcd_{11}..bcd_0 = 000..00$
- onoliko puta koliko BIN ima bita
 - ako je $D_i \geq 5$ dodaje se 3 na D_i
 - šiftuje se BCD za jedno mesto ulevo
 - $bcd_0 = bin_7$
 - šiftuje se BIN za jedno mesto ulevo



MSP430 ima DADD – Decimal add instrukciju

- sabiranje brojeva koji su dati u BCD kodu
- primenom Hornerovog algoritma možemo iterativno od binarnog broja pomoću DADD instrukcije doći do BCD zapisa

Ako je $BIN = bin_7..bin_0$ i $BCD = bcd_{11}..bcd_0$, algoritam je

- 1 $BCD = 0$
- 2 onoliko puta koliko BIN ima bita, počev od MSB indeksa
 - 1 $BCD = (BCD \ll 1) | bin_i$, odnosno

```
RLA  BIN          ; BIN MSB to carry
DADD BCD,BCD      ; BCD = BCD<<1 + C
```



Zadatak – Prikaz na multipleksiranom LED displeju

Zadatak

Napisati program za razvojni sistem koji na multipleksiranom LED displeju prikazuje broj 1234

Napomene

Multipleksiranje LED displeja vršiti pomoću tajmera

Rešenje

`timer-multiplex`



1 Tajmer A

2 Tajmer B

3 **Primeri**

- Debaunsiranje
- Multipleksiranje LED displeja
- **PWM**
- Merenje učestanosti



Zadatak – PWM

Napisati program za razvojni sistem kojim se podešava intenzitet osvetljaja LED diode. LED dioda se pobuđuje PWM signalom koji se generiše pomoću tajmera A0, a intenzitet osvetljaja se podešava jednim tasterom u 16 koraka.

Napomene

Intenzitet osvetljaja diode je proporcionalan faktoru ispunjenosti impulsa PWM signala

Rešenje

timer-pwm-led



1 Tajmer A

2 Tajmer B

3 Primeri

- Debaunsiranje
- Multipleksiranje LED displeja
- PWM
- Merenje učestanosti



Zadatak – Merenje učestanosti

Napisati program za razvojni sistem kojim se pomoću CAPTURE funkcionalnosti tajmera A0 meri učestanost spoljašnjeg signala. Učestanost prikazati u hercima na multipleksiranom LED displeju.

Napomene

- signal se generiše pomoću tajmera B, CC izlaz 2 (pin P7.4)
- signal se očitava na CC ulazu 2 (pin P1.3) tajmera A0
 - potrebno je meriti broj tick-ova tajmera između uzastopnih uzlaznih ivica
 - na osnovu učestanosti tajmera moguće je izračunati učestanost signala



Zadatak – Merenje učestanosti

Napisati program za razvojni sistem kojim se pomoću CAPTURE funkcionalnosti tajmera A meri učestanost spoljašnjeg signala. Učestanost prikazati u hercima na multipleksiranom LED displeju.

Napomene

- signal se generiše pomoću tajmera B, CC izlaz 2 (pin P7.4)
- signal se očitava na CC ulazu 2 (pin P1.3) tajmera A0

Rešenje

`timer-freq-meas`

