Mikropočítačové Systémy

MIPS

Distribuované vnorené počítačové systémy

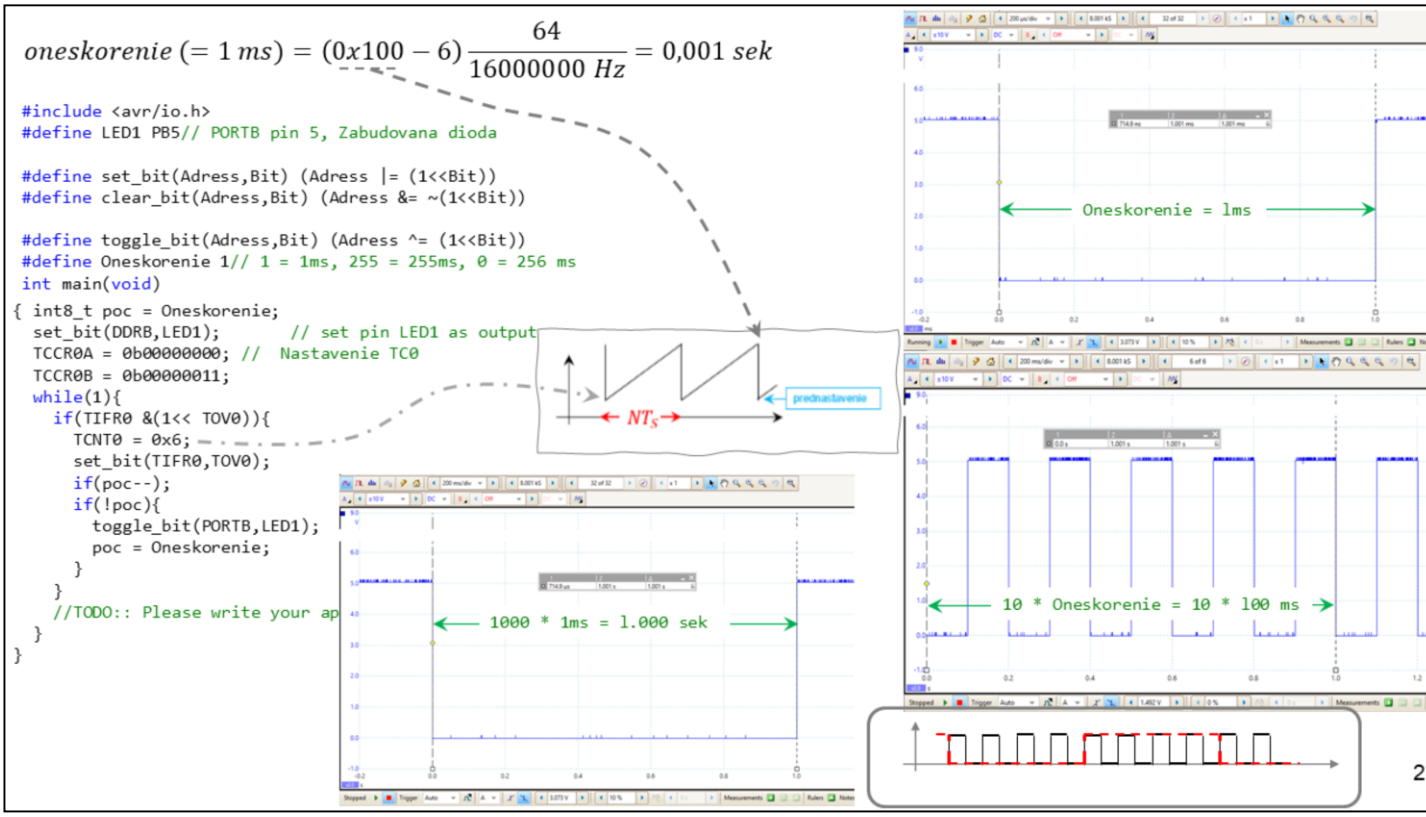
Distributed Embedded Computer System

(Microcontrollers)

Prednáška 4.
Počítadlá / časovače.

Poloha, rýchlosť, čas.

Meranie prietoku.
Hmotnosti.
...



Minulý týždeň na cvičení (viď. návod na cvičenie) ste sa dozvedeli, že niekedy nedokážeme presne generovať väčšie oneskorenie. Tento príklad má dokumentovať, že na základe informácií tejto prednášky dokážeme aj bez prerušenia generovať dlhšie oneskorenia.

Podobne ako v tam spomínanej knižnici použijeme fosc = 16 MHz a 8-bitové počítadlo CT0. Opäť ako na predchádzajúcich prednáškach a cvičeniach sa pokúsime použiť k dosiahnutiu cieľa (implicitné/explicitné) informácie z KL.

V KL sa okrem iného pre CT0 píše, že môžeme použiť preddelič (1/N), kde N =1, 8, 64, 256 a 1024. Ako je zrejmé z priloženého kódu, my sme použili preddelič 1/64. Nie max. číslo a aj napriek tomu sme dosiahli oneskorenie 1sek s presnosťou lepšou ako 1 promile.

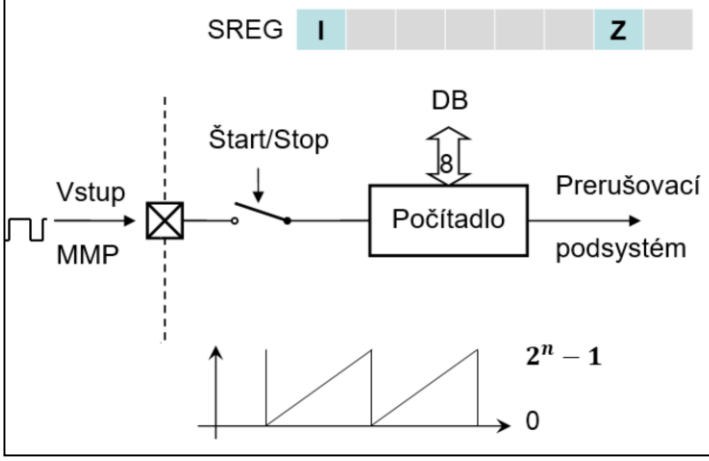
Použili sme informáciu naznačenú vpravo dole. Počas jedného impulzu na vstupe počítadla dokáže procesor vykonať niekoľko inštrukcií. Využijeme to tak, že preddelič môže napočítať až N impulzov fosc. Bez toho aby sme stratili informáciu otom, v ktorom sme impulze počítadla CT1.

Poznámky k obr. Priebehy v pravo sa dajú realizovať pomocou uvedeného programu. Jedno sekundové oneskorenie – na to už potrebujeme premennú int. A spracovanie takejto „informácie“ už je zrejme nad možnosti 64 SC. Ako sme to dosiahli? Cieľ bol jasný presnosť na úrovni promile.

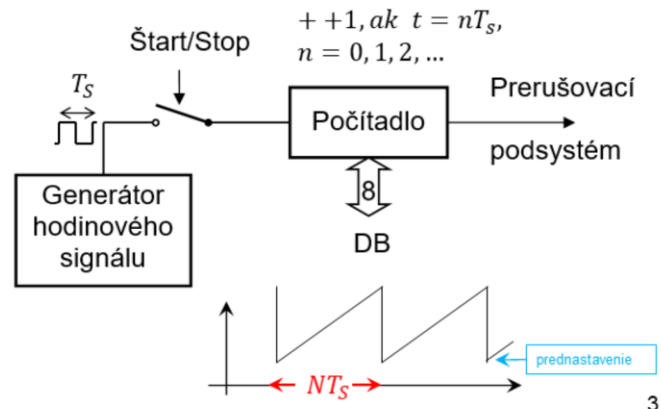
Počítadla / Časovače ⟺ Timer / Counter (T/C)
Všeobecná časť

Počítadlá a časovače sú najčastejšie používanou perifériou MMP. Používajú sa na počítanie impulzov, generovanie časových intervalov, vytvorenie časového oneskorenia. T/C obvody MMP sú realizované čo najuniverzálnejšie. V konkrétnych prevedeniach možno nájsť nasledovné typy, resp. funkcie obvodov.

Počítadlo vonkajších udalostí:



Časovač:



3

Počítadlo vonkajších udalostí

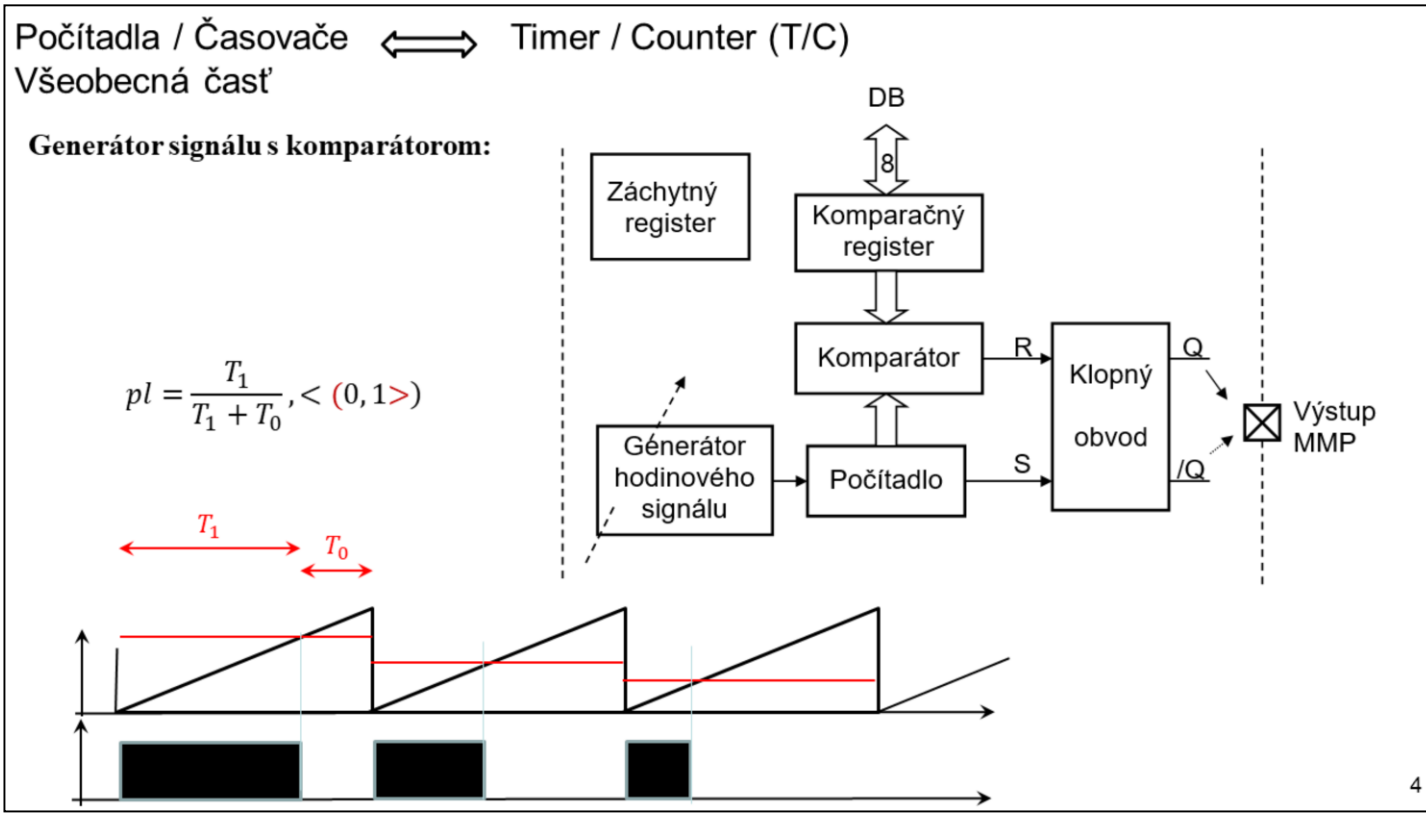
Počítadlo sa používa na počítanie vonkajších udalostí (impulzy s logickou úrovňou „1“, resp. „0“). CPU môže cez vnútornú zbernicu prednastaviť počiatočný stav počítadla a priebežne ho tiež čítať. Pomocou spínača môžeme zastaviť - povoliť načítavanie impulzov počítadlom. Výstup z počítadla, jeho pretečenie, je obvykle použité na generovanie prerušenia. Používa sa na meranie frekvencie signálu privádzaného na pin MMP. Meranie frekvencie je podmienené konštantným časom zopnutia spínača.

Časovač

Obvodová schéma je podobná schéme počítadla udalostí.

Preto sa obidva prípady často kumulujú do jedného pod názvom počítadlo/časovač. Na rozdiel od počítadla nie je na vstup pripojený externý zdroj signálu, ale sa využíva stabilný – presný zdroj frekvencie. Prevažne kryštálom riadený. CPU môže opäť cez vnútornú zbernicu prednastaviť počiatočný stav počítadla a priebežne ho tiež čítať. Pretečenie počítadla signalizuje ukončenie časového intervalu a opäť vyvoláva prerušenie.

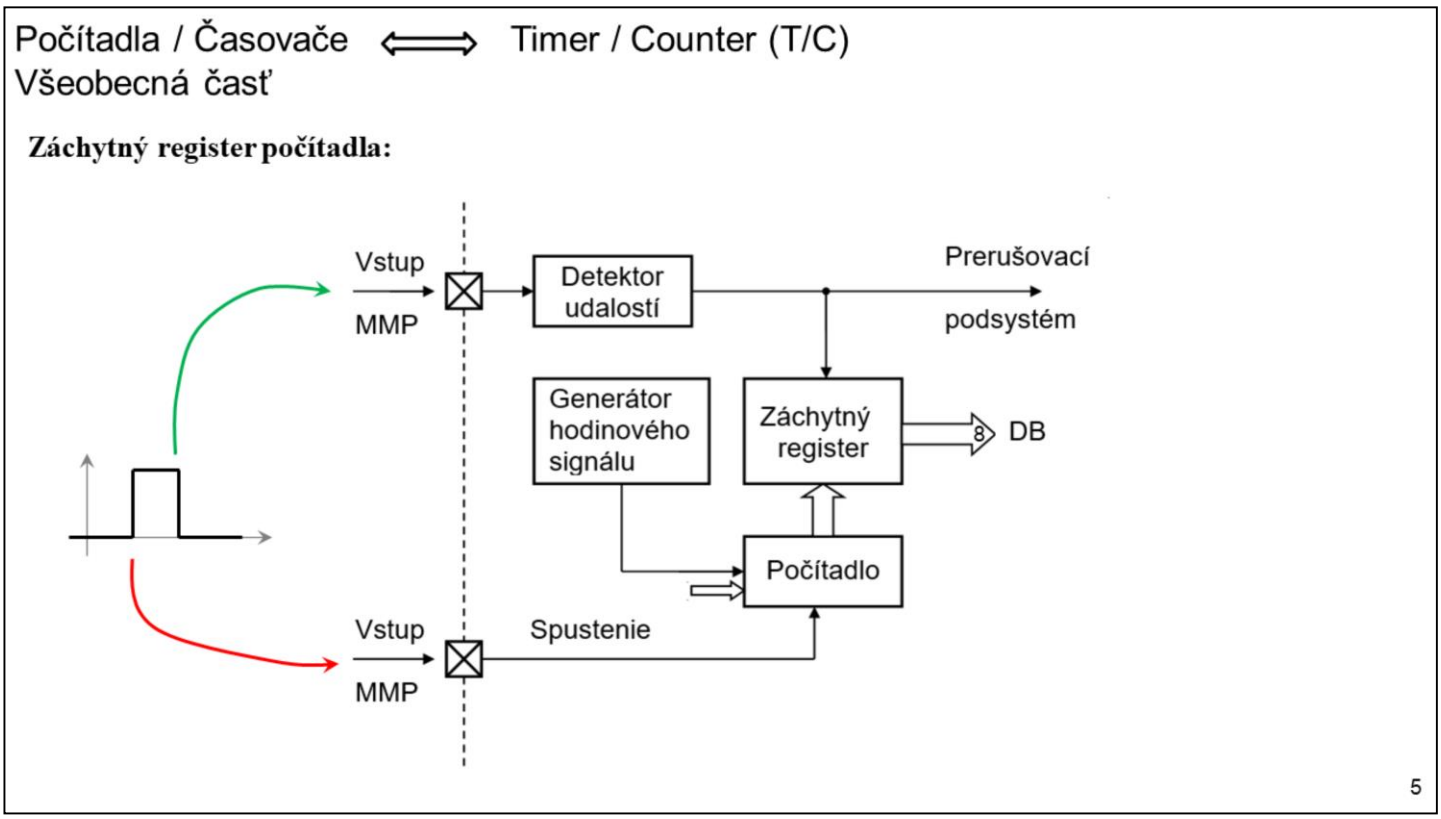
Príkladom použitia je generovanie presného časového úseku. Napr. pre vyššie spomenutý merač frekvencie. Časté použitie je: Generovanie tzv. periódy vzorkovania.



Generátor signálu s komparátorom

Prax často používa zariadenia, kde zdroj signálu má pevnú alebo nastaviteľnú periódu opakovania a konštantné alebo premenlivé plnenie. Aby CPU nebola zaťažovaná programovým generovaním takéhoto signálu, býva MMP vybavený periférnym obvodom generujúcim takýto signál. Uvedený obvod je vlastne svojou funkciou PWM obvod. T.j. obvod šírkovej modulácie impulzov. Základom je voľne bežiace počítadlo na vstupe ktorého je stabilná (preladiteľná) frekvencia. Stav počítadla je porovnávaný v číslicovom komparátore s hodnotou uloženou v komparačnom registre. Pri dosiahnutí zhody vyšle komparátor RESET pre klopný obvod. SET klopného obvodu sa vykoná pri pretečení počítadla. Konkrétna realizácia môže mať vyvedený na pin MMP signál Q, /Q,

resp. oba.



Záchytný register počítadla

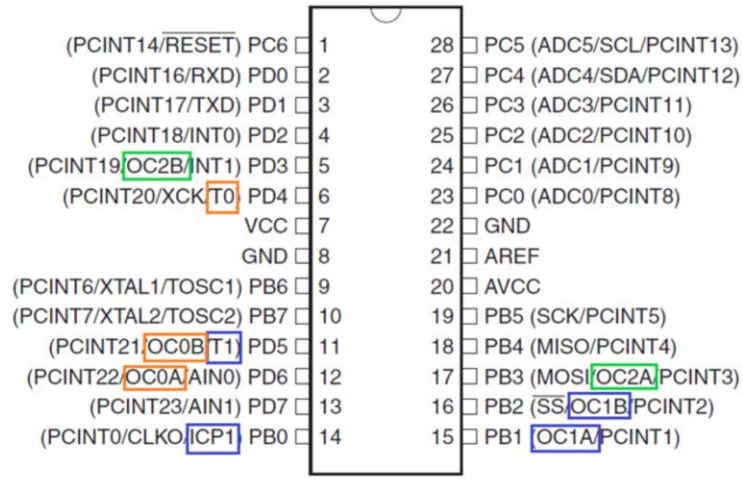
Mnohé technické aplikácie predpokladajú meranie času trvania nejakej udalosti. Meranie kratších časových úsekov by sme programovo nemuseli s dostatočnou presnosťou zvládnuť. Z tohto dôvodu býva počítadlo doplnené o tzv. záchytný register, v ktorom sa zaznamená stav počítadla pri výskyte externej udalosti na sledovanom vstupe MMP. Spustenie počítadla môže byť spojené aj s jeho prednastavením, a môže byť nezávislé na behu programu. Ak detektor udalostí zaznamená zmenu na odpovedajúcom vstupe, odpamätá sa obsah počítadla a vygeneruje sa požiadavka o prerušenie.

ATMEGA 328P T/C:

Má implementované:

- Dva 8-bitové T/C's so samostatným preddeličom (Prescaler)
- Jeden 16-bit T/C so samostatným preddeličom (Prescaler)
- Real Time Counter (RTC) so samostatným oscillátorom
- Šesť PWM kanálov

T/C0 8 – bitov < 0, 255 >
T/C2 8 – bitov < 0, 255 > (32kHz - RTC)
T/C1 16 – bitov < 0, 65535 >

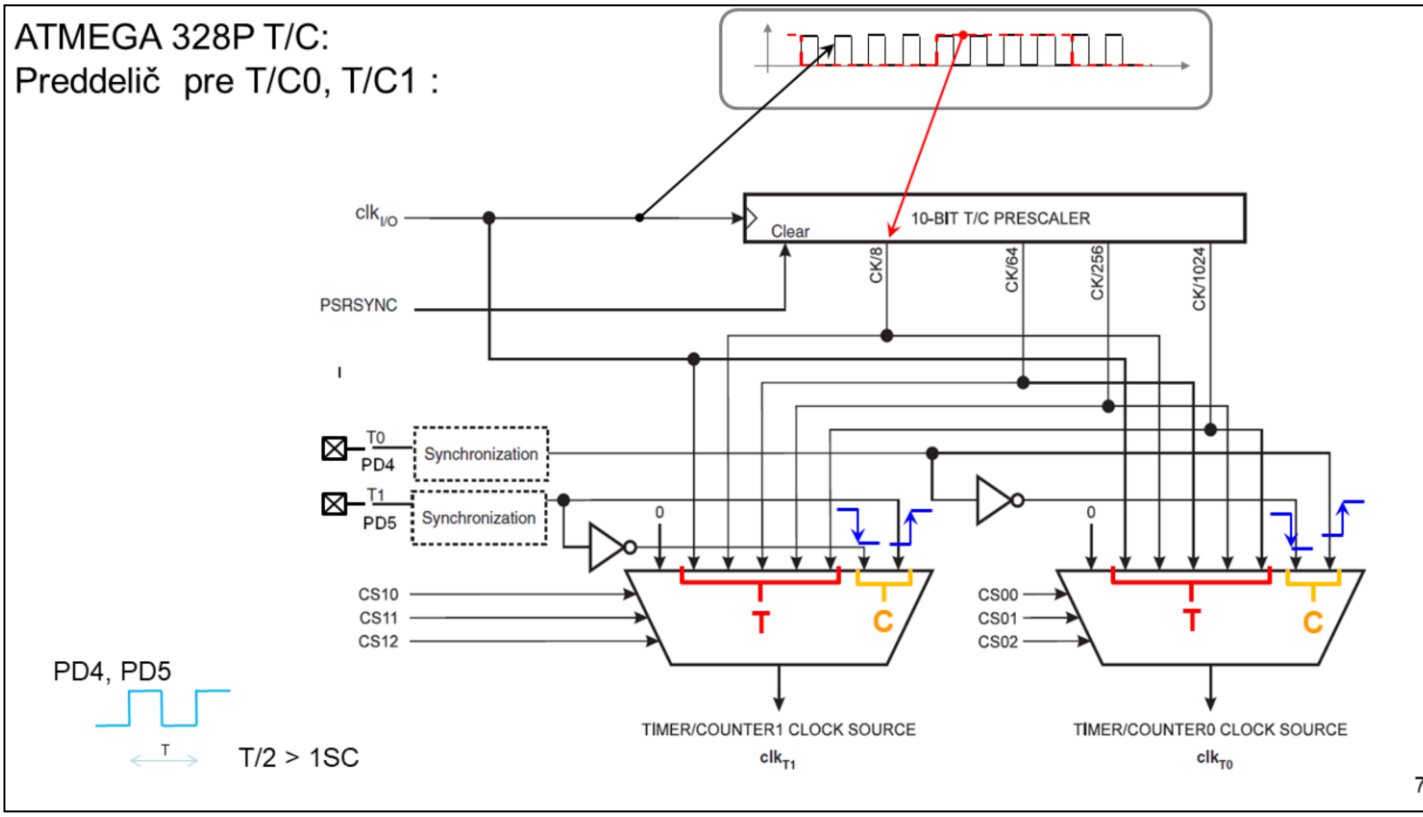


Procesory AVR majú niekoľko **T/C**. Niektoré sú 8-bitové a niektoré sú 16-bitové. **T/C0** a **T/C2** sú 8 bitové a **T/C1** je 16 bitové.

T/C2 môže byť taktovaný z externého kryštálu s frekvenciou 32kHz, čo mu umožňuje pracovať nezávisle na frekvencii oscilátora obvodu. V spojení s útlmovými režimami dokáže tento obvod pracovať vo funkcii RTC. Napr.: každú sekundu vykoná „inkrement“ cez všetky rády sek. min. hod. ... a „uspí“ sa.

Na tejto prednáške sa budeme prevážne venovať TC1. A to z toho dôvodu, že je to 16b-vý T/C. TC0 a TC2 sú trocha jednoduchšie.

K ostatným C/T sa vyjadríme len sporadicky.



Preddelič generuje signály, ktoré načítavajú **T/C0** a **T/C1**. Každé počítadlo ma 8 rôznych vstupov.

1.) Vstupný signál je vypnutý.

2.) Na vstup počítadla je privedený signál odvodený od pracovnej frekvencie procesora $f_{CLK\_I/O}$ alebo jeden z vývodov preddeliča $f_{CLK\_I/O}/8$, $f_{CLK\_I/O}/64$, $f_{CLK\_I/O}/256$, $f_{CLK\_I/O}/1024$.

3.) Do **T/C0** je privedený signál z pinu **PD4** a do **T/C1** je privedený signál z pinu **PD5**.

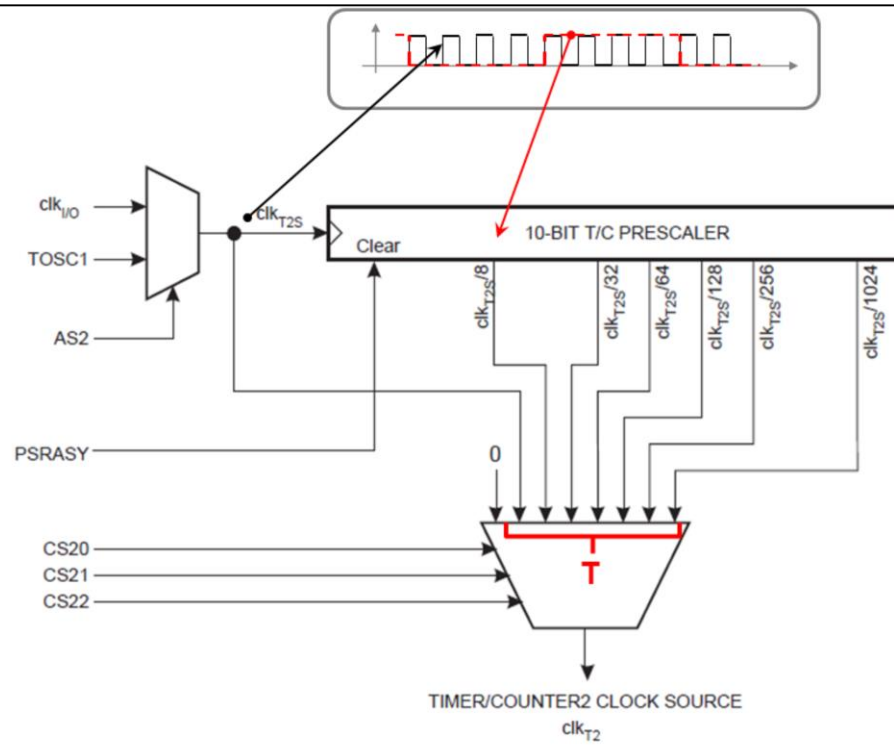
Počítadlo externých udalostí inkrementuje buď pri nábežnej, resp. dobežnej hrane na vstupoch **T0**, **T1**. Vstupy sú synchronizované pomocou $f_{CLK\_I/O}$. Takýto signál prechádza cez detektor hrán. Detektor hrán generuje pulzy buď pri kladnej, resp. zápornej hrane pre jednotlivé počítadlá. Jedna polperióda externého signálu musí byť

dlhšia ako jeden **SC** procesora.

Preddelič je voľne bežiaci. Jeho prednastavenie je ***nezávislé*** na pripojenom počítadle. Ak počítadlo pripojíme k preddeliču, môže načítať 1 až N+1 systémových hodinových signálov, kde N je deliaci pomer preddeliča (8, 64, 256, or 1024), než počítadlo prvý krát inkrementuje svoj obsah.

Pozor na programové zosynchronizovanie preddeliča s počítadlom – vynulovanie. Vynulovaný preddelič zapôsobí na obe počítadlá.

ATMEGA 328P T/C
Preddelič pre T/C2 :



8

Preddelič generuje signály, ktoré načítava **T/C2**. Počítadlo ma 8 rôznych vstupov.

1.) Vstupný signál je vypnutý.

2.) Na vstup počítadla je privedený signál odvodený od pracovnej frekvencie procesora $f_{CLK\_I/O}$ alebo jeden z vývodov preddeliča $f_{CLK\_I/O}$ /8, $f_{CLK\_I/O}$ /32, $f_{CLK\_I/O}$ /64, $f_{CLK\_I/O}$/128, $f_{CLK\_I/O}$/256, $f_{CLK\_I/O}$/1024.

Preddelič je voľne bežiaci. Jeho prednastavenie je ***nezávislé*** na pripojenom počítadle. Ak počítadlo pripojíme k preddeliču, môže načítať 1 až N+1 systémových hodinových signálov, kde N je deliaci pomer preddeliča (8, 32, 64, 128, 256, or 1024), než počítadlo prvý krát inkrementuje svoj obsah.

Pozor na programové zosynchronizovanie preddeliča

s počítadlom – vynulovanie.

ATMEGA 328P T/C.Preddelič pre T/C0, T/C1.
Registre:

General Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
0x23 (0x43)	TSM	-	-	-	-	-	PSRASY	PSRSYNC	GTCCR
Read/Write	R/W	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

V KL nájdeme, napr.: CS02:0
$2:0 \equiv 2,1,0$

Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
0x81	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Timer/Counter Control Register

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0B	FOC0B	-	-	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

x = 0 pre **T/C0** a x = 1 pre **T/C1**.

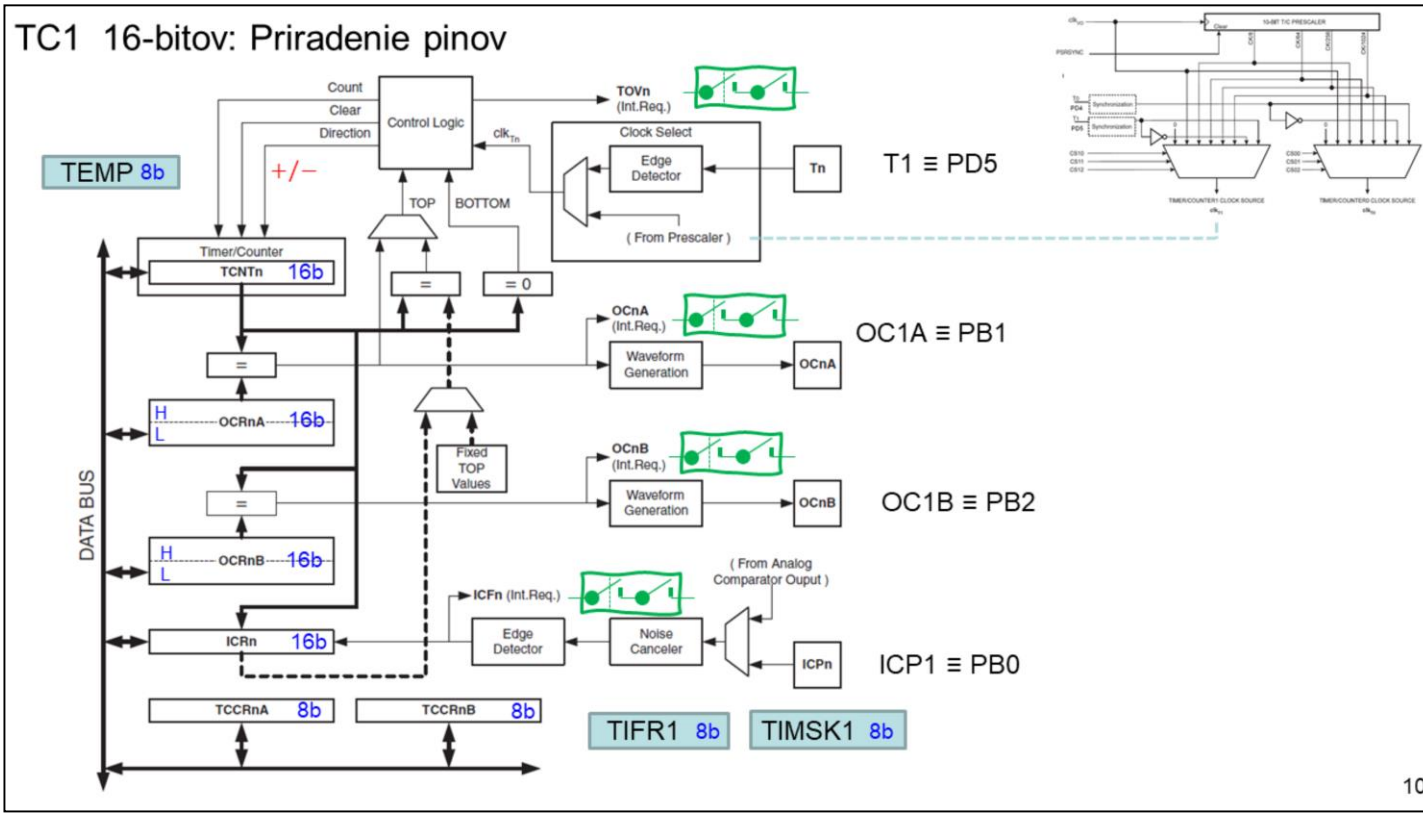
CSx2	CSx1	CSx0	
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$ (No prescaling)
0	1	0	$clk_{I/O}/8$ (From prescaler)
0	1	1	$Clk_{I/O}/64$ (From prescaler)
1	0	0	$Clk_{I/O}/256$ (From prescaler)
1	0	1	$clk_{I/O}/1024$ (From prescaler)
1	1	0	External clock source on Tx pin. Clock on falling edge.
1	1	1	External clock source on Tx pin. Clock on rising edge.

!!! Nesmieme zabudnúť, že toto celé je spriahnuté s f_{OSC} !!!

9

Zápis 1 do **PSRSYNC** vynuluje preddelič pre **T/C1** aj **T/C0**. Tento byt sa vynuluje po vykonaní operácie.

Prednastavenie preddeliča pre **T/C0** sa vykoná nastavením bitov **CS02:0** v registri **TCCR0B** a pre **T/C1** sa vykoná nastavením bitov **CS12:0** v registri **TCCR1B**. Okrem **TCCR1B** patria k **TC1** aj ďalšie riadiace registre: **TCCR1A** a **TCCR1C.**



Z blokového zapojenia je zrejmé, že sa jedná o komplikovane fungujúce zariadenie. Mnohé registre súvisiace s TC1 nie uvedené. Dokonca ani to čo sme dokreslili nie je konečný počet registrov, ktoré súvisia s činnosťou TC1.

Zeleno nakreslené vypínače predstavujú do série zapojené **povolenia** globálneho a lokálneho prerušenia. Register TEMP nie je programátorovi prístupný. Je to pomocný register, ktorý použijeme na zápis / čítanie 16b-ových registrov pomocou 8-bitovej DB.

TC1 16-bitov: Vlastnosti

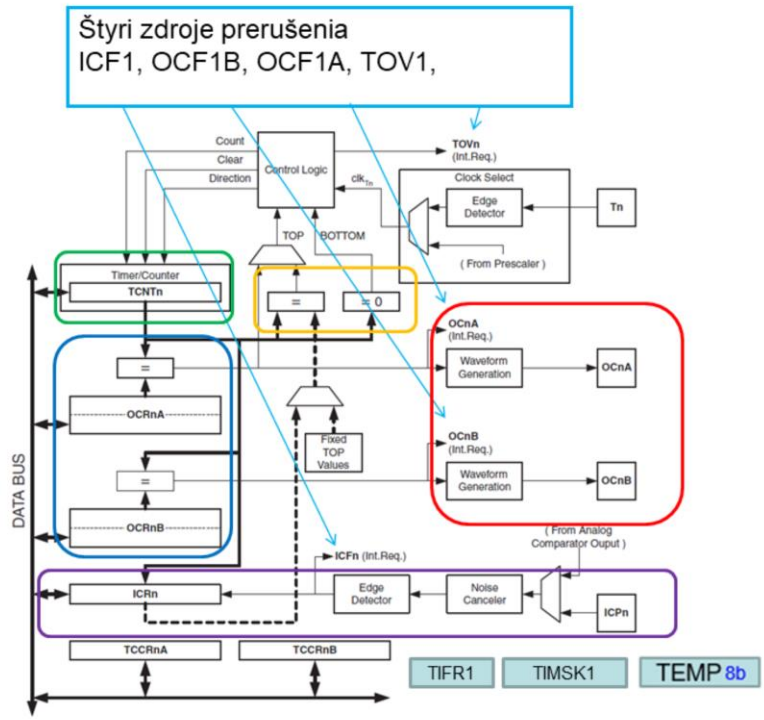
16-b-ové počítadlo: 0 až 65535

2 komparačné bloky

Nulovanie počítadla pri zhode, automatické znovunastavenie

PWM s nastaviteľnou periódou

Jeden záchytný vstupný blok s jednoduchým filtrom



11

TC1 16-bitov: Vlastnosti

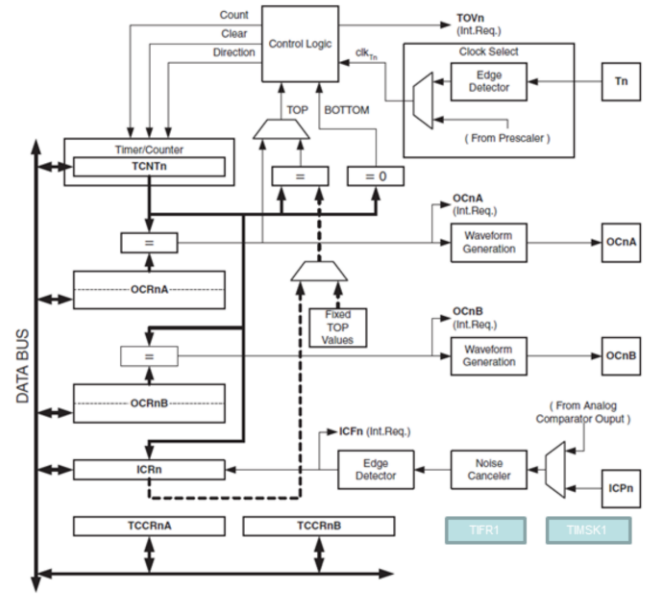
16 – bitové registre:
– **TCNT1** – počítadlo/časovač,
– **ICR1** – **input capture register**.
– **OCR1A, OCR1B – output compare register.**
  – generovanie PWM, resp.
  – tvarovanie signálu na pine **OC1A**, **OC1B.**

Pri prístupe k týmto registrom treba pristupovať špeciálnym spôsobom.
„8-bitový stroj spracováva 16bitové operandy.“

AVR CPU pristupuje k 16-bitovým registrom cez 8-bitovú *data bus*. ⇒ musia byť čítané, resp. zapisované na dva krát.

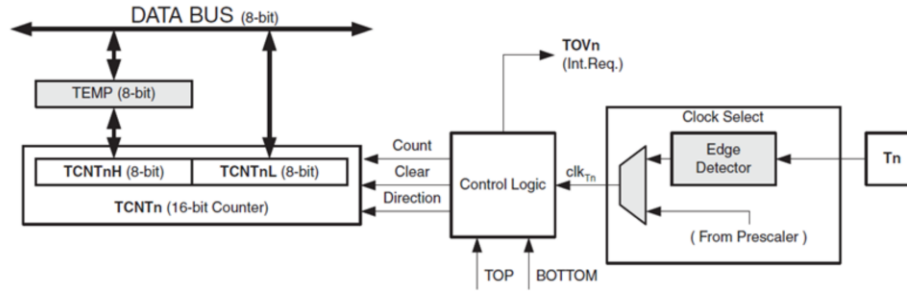
8 – bitové registre:
– **TCCR1A** – počítadlo/časovač kontrol register,
– **TCCR1B** – počítadlo/časovač kontrol register,
– **TIFR1** – timer interrupt flag register
– **TIMSK1** – timer interrupt mask register
(maska pre každé prerušenie zvlášť)

Read/Write operácie s 16 bitovými registrami **OCR1A** a **OCR1B** nevyžaduje použitie *temporary registra.*



TC1 16-bitov: Vlastnosti

TCNT1 = 256 * **TCNT1H** + **TCNT1L**



Popis signálov: (n = 1)

Count	**TCNT1** += 1, resp.
	TCNT1 –= 1 („modulo aritmetika“)
Direction	Prepínanie medzi inkrement a dekrement.
Clear	Vynulovanie **TCNT1** (**TCNT1** = 0x0000).
clk$_{T1}$	hodinový signál pre **T/C1**
TOP	Signalizácia, že **TCNT1** dosiahlo maximum.
BOTTOM	Signalizácia, že **TCNT1** dosiahlo minimum (0x0000) .

Definície:

BOTTOM - počítadlo dosiahne *BOTTOM* keď nadobudne hodnotu 0x0000.
MAX - počítadlo dosiahne *MAX*imum keď nadobudne hodnotu 0xFFFF (dekadicky 65535).
TOP - počítadlo dosiahne *TOP* – obsah rovný najväčšej možnej hodnote.
TOP môže byť nastavená na jednu z nasledovných hodnôt:
0x00FF, 0x01FF, 0x03FF, alebo na hodnotu uloženú v registri **OCR1A**, **ICR1**.
Priradenie (TOP = 0x...) je dané nastaveným – zvoleným módom.

13

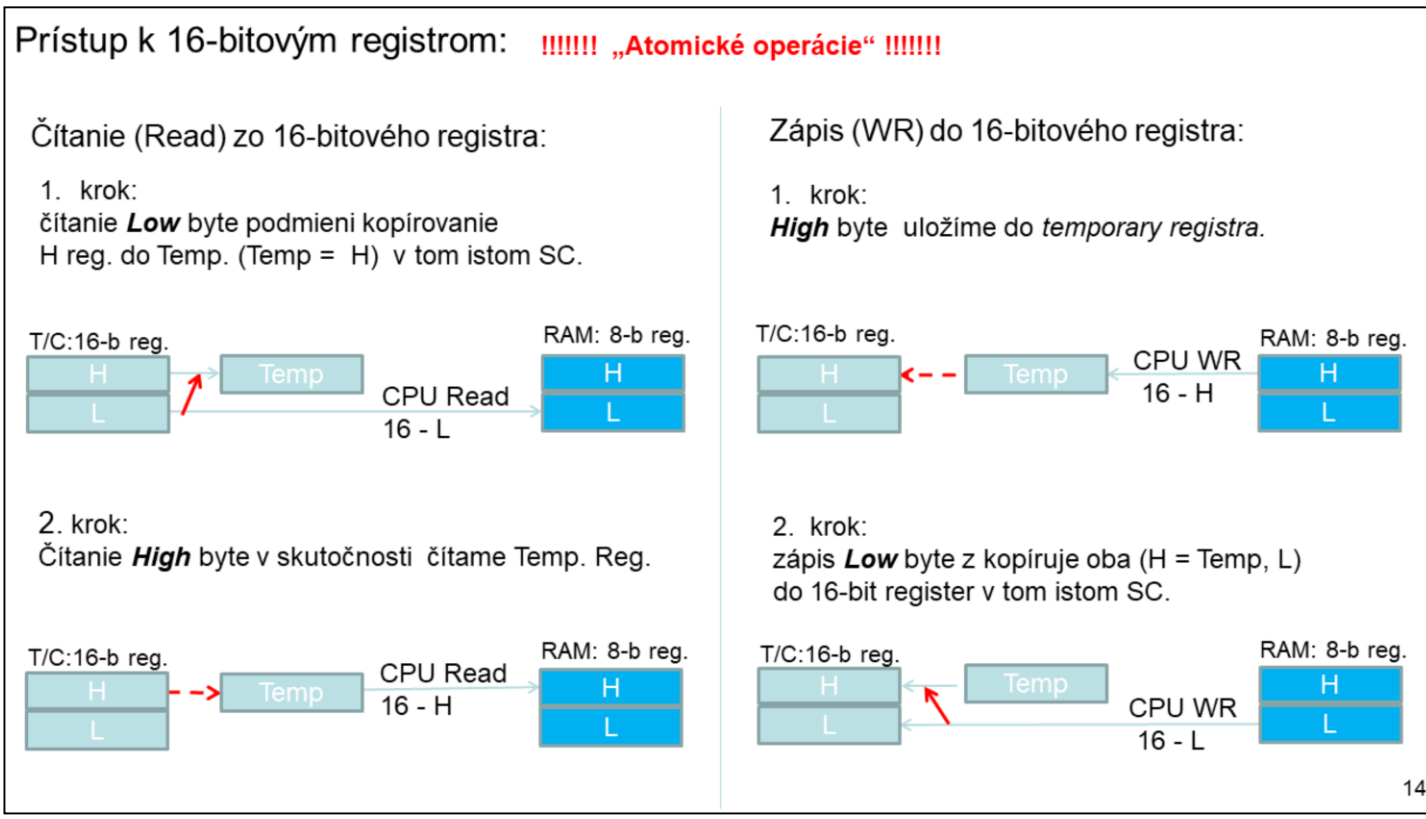
Treba si uvedomiť, že 16-bitový register je sprístupnený na dva kroky, a že je mapovaný ako dva 8-bitové registre. Napr.: takto:

TCNT1 = 256 * **TCNT1H** + **TCNT1L**

T.j. ak čítam **TCNT1H** čítam vlastne **TEMP** register. A ak zapisujem do **TCNT1H** zapisujem do **TEMP** registra. Obsah **TCNT1H** sa prekopíruje do **TEMP** pri čítaní **TCNT1L**. Obsah **TEMP** sa prekopíruje do **TCNT1H** po zápise do **TCNT1L**.

Ak sa objaví interrupt medzi dvomi inštrukciami pri prístupe k 16-bitovému registru, a obsluha prerušenia

pristupuje k tomu istému temporary registru, výsledok mimo prerušenia bude zlý. Z tohto dôvodu treba počas prístupu k 16-bitovému registru zakázať prerušenie.



Prístup k 16-bitovým registrom

Registre **TCNT1**, **OCR1A**, **OCR1B**, a **ICR1** sú 16-bitové. AVR CPU k nim pristupuje cez 8-bitovú *data bus*. TJ. musia byť čítané, resp. zapisované na dva krát. Každé 16-bitové počítadlo má jeden 8-bitový pomocný register – ***temporary***, do ktorého sa ukladá horný byte (*High*) 16-bitového registra.

Tento pomocný register je zdieľaný pre všetky 16-bitové registre počítadla. Prístup k spodnému bytu „zapne“ 16-bitové čítanie, resp. zápis.

Keď CPU zapíše ***Low*** byte 16-bitového registra, ***High*** byte je uložený v *temporary registry*, a zápis ***Low*** byte skopíruje oba do 16-bit register v tom istom strojovom cykle.

Keď CPU číta ***Low*** byte 16-bitového registra ***High*** byte sa skopíruje do *temporary registra* v tom istom *SC*. Ak

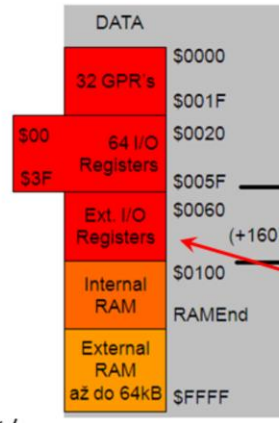
chceme zapísať do 16-bitového registra ***High*** byte musí byť zapísaný pred ***Low*** byte. Pri 16-bitovom čítaní ***Low*** byte treba čítať skorej.

Read/Write operácie s 16 bitovými registrami **OCR1A** a **OCR1B** nevyžaduje použitie *temporary registra*.

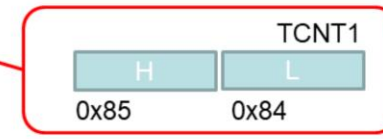
.

Príklad: WRITE/READ TCNT1 – neuvažujeme prerušenie

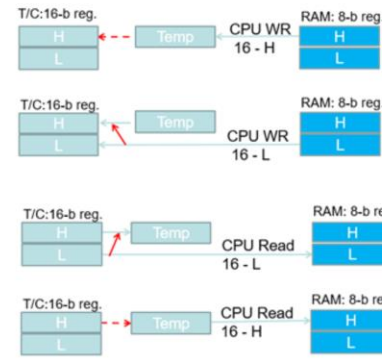
```
...
/* Do TCNT1 vložíme 0x01FE */
TCNT1 = 0x1FE;

/* Prečítanie obsahu TCNT1 do pom_int */
I = TCNT1;
```



ATMEGA 328P: IRAM = 2kB
RAMEnd = $0100+$07FF =$08FF




```
unsigned int pom_int;
.............................
TCNT1 = 0x01FE;/* Prednastavenie TCNT1 */
LDI R30,0x84   ;ZL = adr.TCNT1L
LDI R31,0x00   ;ZH = 0
LDI R24,0xFE   ;pom_int (R25,R24) = 0x01FE
LDI R25,0x01
STD Z+1,R25    ;zapis do Temp = 0x01
STD Z+0,R24    ;(zapis do TCNT1L) & (zapis Temp do TCNTH1)

pom_int = TCNT1;/* Vycitanie obsahu TCNT1 */
LDD R24,Z+0    ; pom_int (R25,R24) = <TCNT1>
LDD R25,Z+1
```



15

Tento príklad ukazuje ako treba pristupovať 16-bitovému registru. Pr. predpokladá, že *temporary register* sa v prerušení nemení. C-ko pristupuje k registru ako 16-bitovému.

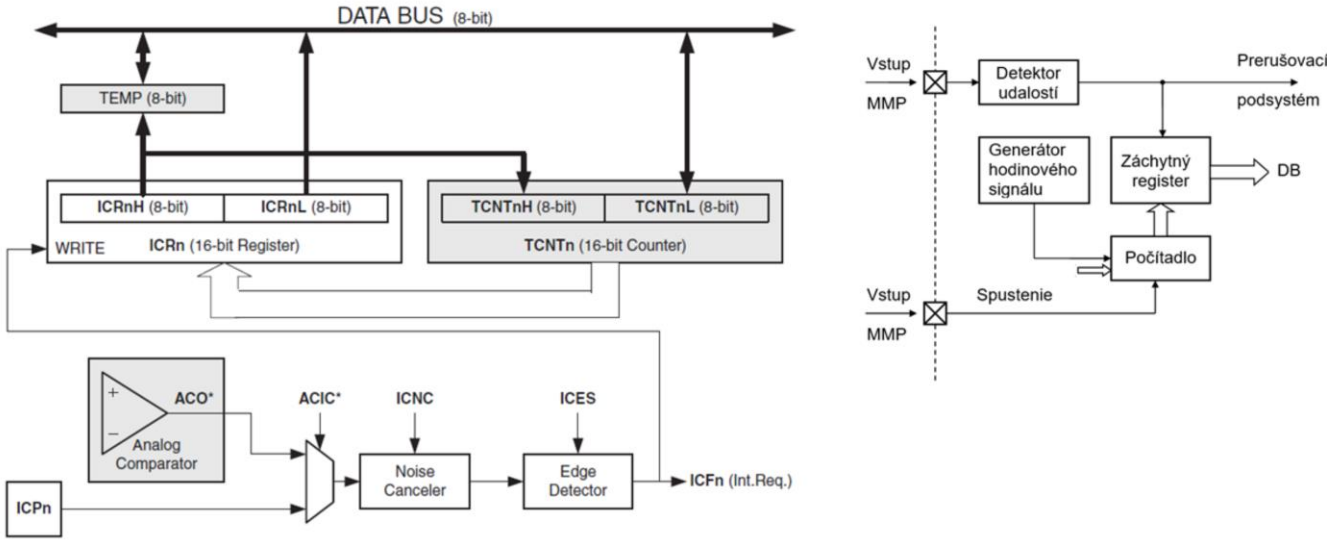
Príklad: WRITE/READ TCNT1
– uvažujeme prerušenie

```
void  Timer1_16bWrite_TCNT1(unsigned int
pom_int){
uint8_t st_reg;
/* Odlozenie global Interrupt Flag */
st_reg = SREG;

/* Vypnutie interrupts */
cli();

/* Prednastavenie pocitadla */
TCNT1 = pom_int;
STS 0x0085,R25
STS 0x0084,R24

/* Obnovenie global Interrupt Flag */
SREG = st_reg;
}
```

```
unsigned int Timer1_16bRead_TCNT1(void){
uint8_t st_reg;
unsigned int pom_int;
/* Odlozenie global Interrupt Flag */
st_reg = SREG;

/* Vypnutie interrupts */
cli();

/* Precitanie TCNT1 do pom_int */
pom_int = TCNT1;
LDS R18,0x0084
LDS R19,0x0085

/* Obnovenie global Interrupt Flag */
SREG = st_reg;
return pom_int;
}
```

16

TC1 Záchytná jednotka – Input Capture Unit



T/C má zabudovaný vstupnú záchytnú jednotku externých udalostí.

Pomocou časových vzoriek možno počítať frekvenciu, plnenie a ďalšie parametre impulzu.

17

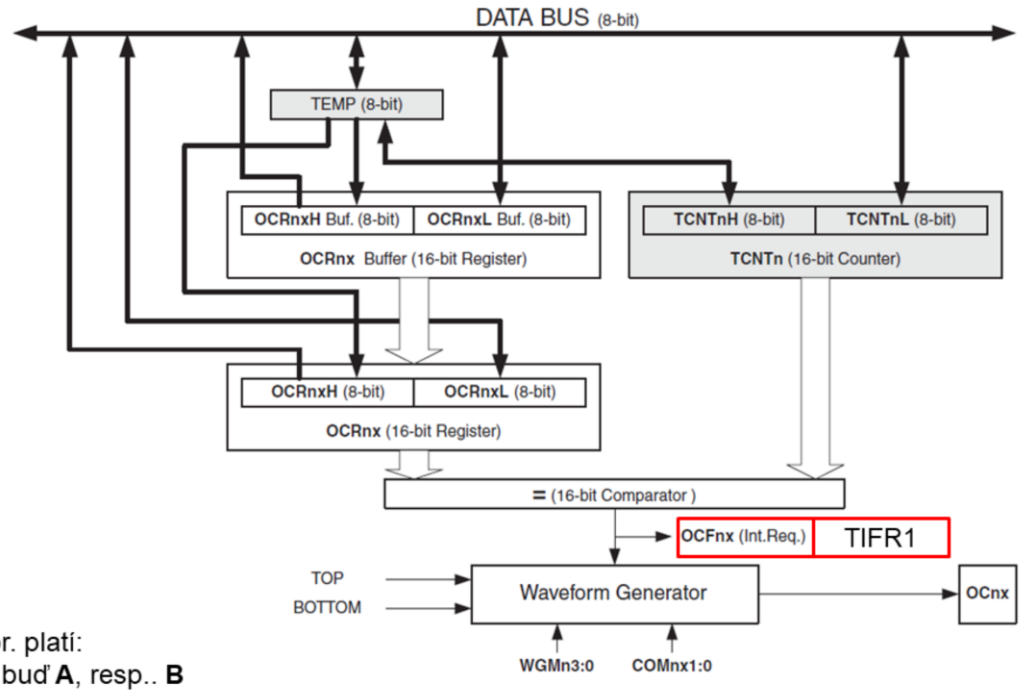
Ak sa na pine *Input Capture pin* (**ICP1**) objaví „udalosť“ zhodná s nastavenou v detektore hrán, nastaví – zapne sa odchytenie. V tomto okamžiku sa odchytí 16-bitová hodnota počítadla **TCNT1** je zapísaná do *Input Capture Register* (**ICR1**). *Input Capture Flag* (**ICF1**) sa nastaví v tom istom **SC**. Ak je povolené prerušenie (**TICIE1** = 1) nastavený bit (**ICF1**) generuje prerušenie *Input Capture Interrupt*. Pri vstupe do prerušenia sa bit (**ICF1**) automaticky vynuluje. Opäť poznamenajme, že bit možno vynulovať aj softwerovo. „16-bitové čítanie“.

Do registra **ICR1** možno zapisovať len vtedy keď je použitý na tvarovanie výstupného priebehu. **ICR1** register definuje v tomto režime hodnotu **TOP** počítadla. *Waveform Generation mode* (WGM13:0) treba nastaviť skôr ako budeme nastavovať hodnotu **TOP** do registra **ICR1**. Obsah **ICR1** treba zapisovať ako 16-bitové číslo. Dôvod

prečo konštruktéry zabudovali tento blok do počítadla je ten, že chceli odbremeniť procesor v kritickom čase. V prerušení treba rýchle prečítať obsah **ICR1** registra. Toto prerušenie *Input Capture Interrupt* má relatívne vysokú prioritu.

Meranie plnenia externého signálu vyžaduje prepnutie záchytnej hrany po každom odchytení. Zmenu treba vykonať prakticky okamžite po prečítaní **ICR1** registra. Po zmene záchytnej hrany treba vynulovať **ICF1** bit – softwarovo.

TC1: Výstupný komparačný blok – Output Compare Units



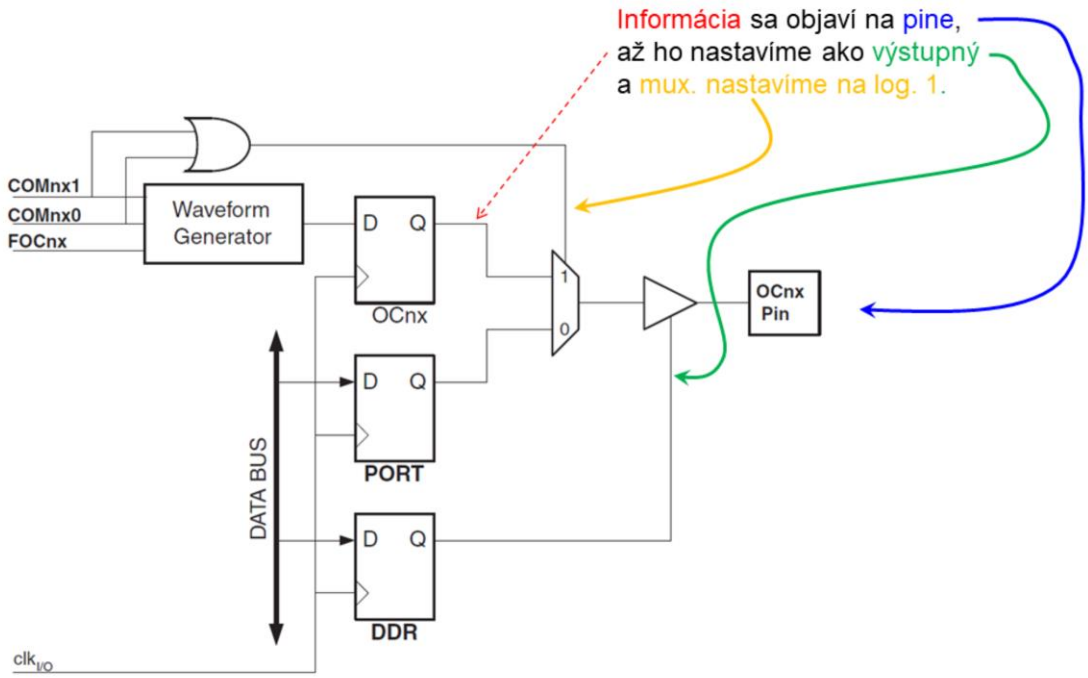
Pre tento obr. platí:
n = **1**, a x je buď **A**, resp.. **B**

18

16-bitový komparátor priebežne porovnáva obsah **TCNT1** s obsahom *Output Compare Register* (**OCR1x**). Ak sa obsah **TCNT1** = **OCR1x** komparátor ohlási zhodu. Pri zhode sa v ďalšom **SC** nastaví *Output Compare Flag* (**OCF1x**). Ak je prerušenie povolené (**OCIE1x =1**), **Output Compare Flag** generuje prerušenie. Flag **OCF1x** je nulovaný po vyvolaní prerušenia. *Waveform Generator* využíva signál zhody pri generovaní výstupu v zhode s nastaveným módom: *Waveform Generation mode* (WGM13:0) bity and *Compare Output mode* (COM1x1:0) bity.

Prístup k registrom **OCR1x** je relatívne zložitý. Niektoré PWM módy umožňujú použitie vyrovnávacej pamäte. CPU pristupuje k **OCR1x** cez buffer register. Ak je vypnuté, CPU pristupuje k registru **OCR1x** priamo.

TC1: Využitie výstupnej komparačnej jednotky



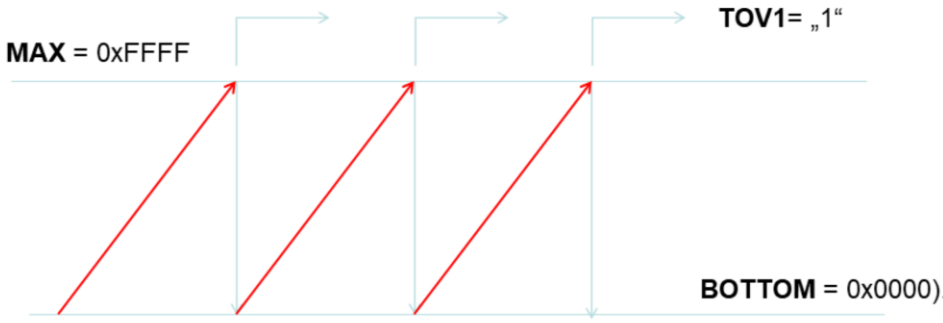
19

Opäť niekoľko zložitejších vecí vynecháme.
Poznamenajme, že bity **COM1x1:0** nie sú dvojnásobne ukladané do vyrovnávacej pamäte zmena na týchto bitoch pôsobí okamžite.
Compare Output mode (**COM1x1:0**) bity majú dve funkcie. *Waveform Generator* používa bity **COM1x1:0** na definovanie stavu výstupu *Output Compare* (**OC1x**) pri nasledovnej zhode. Po druhé bity **COM1x1:0** riadia zdroj pre pin **OC1x**.

Ak nastavíme režim práce (**COM1x1:0** = 3) tak, že výstup komparačného registra generátora signálov (*Waveform Generator*) je „smerovaný“ na pin *Output Compare* (**OC1x**) procesora, potom treba nastaviť pin v **DDR** registri ako výstupný.

TC1: Normal Mode (WGM13:0 = 0)

V tomto móde sa vždy počíta smerom hore.



TIFR1 – Timer/Counter1 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x16 (0x36)	–	–	ICF1	–	–	OCF1B	OCF1A	TOV1	TIFR1
Read/Write	R	R	R/W	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Info z KL:
TOV1 is automatically cleared when the Timer/Counter1 Overflow Interrupt Vector is executed. Alternatively, TOV1 can be cleared by writing a logic one to its bit location.

20

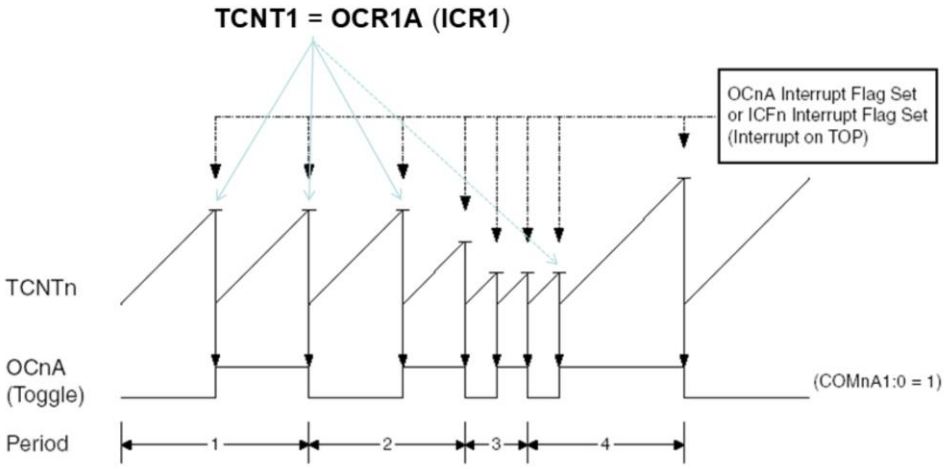
Normal Mode (= 0)

Normal mode (**WGM13:0** = 0). V tomto móde sa vždy počíta smerom hore. Nenuluje sa. Jednoducho sa pretočí z (**MAX** = 0xFFFF) na **BOTTOM** (0x0000). V tomto móde sa bit *Timer/Counter Overflow Flag* (**TOV1**) nastaví v tom istom **SC** ako sa **TCNT1** vynuluje.

TC1: Clear Timer on Compare Match (CTC) Mode (WGM13:0 = 4, 12)

Mode	WGM13	WGM12	WGM11	WGM10	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
12	1	1	0	0	CTC	ICR1	Immediate	MAX

$$f_{OCnA} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRnA)}$$



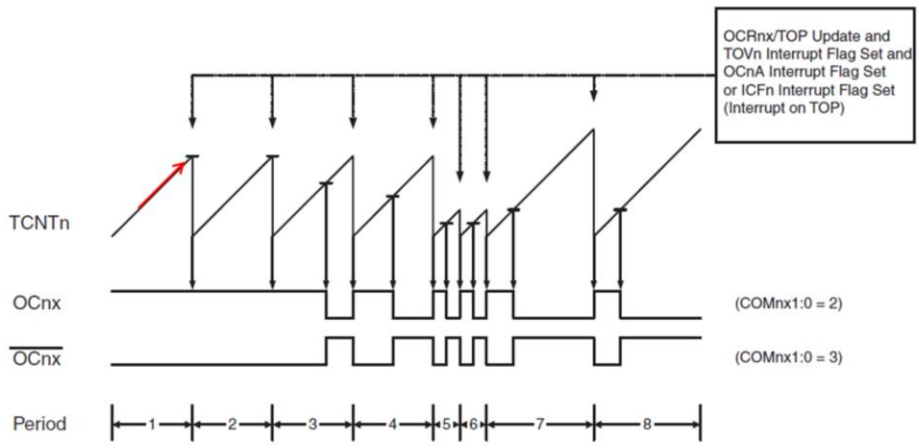
21

Možno si spomeniete na prednášky Zo ZP. Tam sa spomínali chyby programátorov. Jedna veľká vojenská loď sa len tak „knísala“ na vode len preto, že programátor sa rozhodol deliť NULOU. Vzorec vpravo hore, je blbuvzdorný. Ak Je preddelič nastavený na N=0, tak sa nič nedeje. A ak by aj niekto zapísal OCRnA = 0, nič sa nestane, pretože je tam poistka v podobe JEDNOTKY.

TC1: Fast PWM Mode (WGM13:0= 5,6,7,14,15)

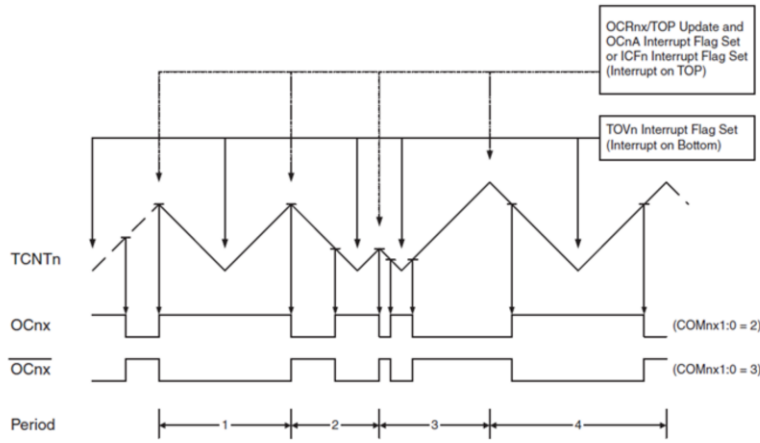
Mode	WGM13	WGM12	WGM11	WGM10	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

$$R_{FPWM} = \frac{\log(TOP+1)}{\log(2)}$$



TC1: Phase Correct PWM Mode (WGM13:0= 1,2,3,10,11)

Mode	WGM13	WGM12	WGM11	WGM10	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM



TC1: Možné módy generovania signálu na výstupe

Mode	WGM13	WGM12	WGM11	WGM10	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

24

TC1: **Pr. 1.**

Nastavme **T/C1** tak, aby generoval
- *časové značky 50 ms bez prerušenia*
- *časové značky 50 ms s prerušením*

Frekvencia oscilátora: f_{OSC} = 8 000 000 Hz.
Jeden SC = 0,125 us, počítadlo počíta smerom nahor.

0,05 s / 0,000 000 125 s = 400 000 SC. (16b T/C = ?)

Preddelič:
Ak budeme načítavať $clk_{I/O}/8$ treba načítať 50 000 impulzov.
Prednastavenie počítadla:
$0x10000 - 50\,000_{10} = 0x3CB0$.
Ak budeme načítavať $clk_{I/O}/64$ treba načítať 6250 impulzov.
Prečo sme nenavrhli deleno 256?

25

Tento príklad je tak zvolený, aby sme poukázali na nevýhody celočíselného delenia.

Ak ho chcete použiť pre ATMEGA 328 musíte niektoré informácie prispôsobiť realite.