

Programiranje z uporabo prekinitev

- **Napisati prekinitveni servisni program (PSP)**
 - Kaj naj se izvede, ko pride do prekinitve določene vrste?
 - PSP končamo z ukazom RTI
- **Napisati ustrezno inicializacijo (v glavnem programu)**
 - Začetni naslov PSP vpisati v ustrezni prekinitveni vektor
 - Omogočiti želene prekinitev
 - Različni kontrolni biti omogočijo različne tipe prekinitev
 - Z ukazom CLI pobrisati zastavico I v registru CCR
 - Po inicializaciji lahko glavni program normalno teče
 - PSP se bo izvedel avtomatsko, če se pojavi ustrezna prekinitev

Prekinitveni vektorji HC11

Maskirani (se ignorirajo),
če je I-bit v CCR enak 1

16-bitni vektorji

FFD6		SCI serial system
FFD8		SPI serial transfer complete
FFDA		pulse accumulator input edge
FFDC		pulse accumulator overflow
FFDE		timer overflow
FFE0		timer output compare 5
FFE2		timer output compare 4
FFE4		timer output compare 3
FFE6		timer output compare 2
FFE8		timer output compare 1
FFEA		timer input capture 3
FFEC		timer input capture 2
FFEE		timer input capture 1
FFF0		real time interrupt
FFF2		IRQ' pin interrupt
FFF4		XIRQ' pin interrupt
FFF6		SWI
FFF8		illegal opcode trap
FFFA		COP failure (reset)
FFFC		COP clock monitor fail (reset)
FFFE		reset'

najnižja prioriteta



najvišja prioriteta

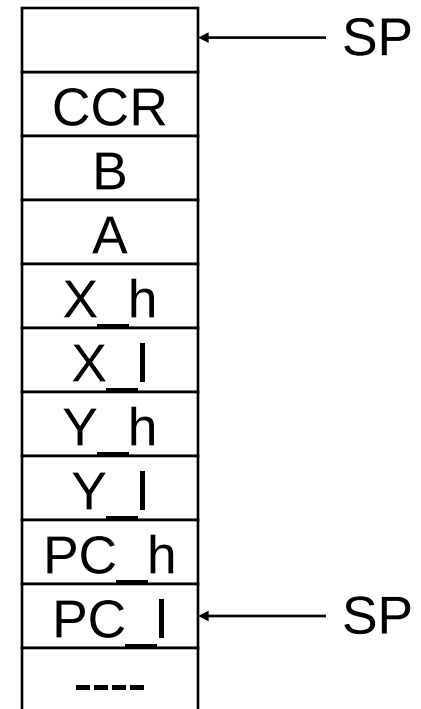
} software interrupts

Večino prekinitev lahko maskiramo z uporabo določenih kontrolnih bitov

Dogajanje ob prekinitvah

Ko pride do prekinitve:

1. HC11 shrani na sklad (PUSH):
PC, registre IY, IX, akumulatorja in CCR
2. HC11 postavi I-bit
Prekinitve se maskirajo – gnezdenja ni
3. Poišče prekinitveni vektor z najvišjo prioriteto
4. Skoči na začetek PSP (kamor kaže vektor)
Izvaja ukaze do **RTI**
5. Restavrira registre (vrednosti so na skladu - PULL)
(pozor: tudi I-bit v CCR)
6. Vrne se na ukaz, pred katerim je prišlo do prekinitve
(prekinitve so spet omogočene)



Nadzor nad prekinitvami

- **I-bit v CCR določa ali so maskirne prekinitve dovoljene**
 - $I = 1$ --> onemogočene, $I = 0$ --> prekinitve dovoljene
- **I bit je vedno postavljen v PSP**
 - Gnezdenje ni dovoljeno
- **Če želimo gnezdenje, ga lahko omogočimo**
 - SEI - set interrupt mask (onemogočimo)
 - Uporabno med inicializacijo, ko prekinitve ne želimo (še niso inicializirane)
 - CLI - clear interrupt mask (omogočimo)
 - Na koncu inicializacije, tudi za gnezdenje prekinitvev

Primer dela z IRQ

Napisati inicializacijo in glavni program

```
count      org    $2000      ;pomnilnik za podatke
           fdb    0          ;števec prekinitev

           org    $E000      ;glavni program
start      lds    #$3FFF      ;inicializacija sklada
           CLI                    ;omogočimo prekinitve

           bra    loop       ;neskončna zanka
```

Glavni program!

Napisati PSP:

Vsakič, ko se pojavi prekinitev
IRQ, se števec poveča

```
*** prekinitveno servisni podprogram za IRQ prekinitev
irqhand:   ldx    count      ; IX <-- trenut.vred.števca
           inx                    ; povečaj za 1
           stx    count      ; zapiši števec nazaj
           rti                    ; konec - povratek
```

IRQ vektor: \$FFF2

```
org    $FFF2
fdb    irqhand
```

Reset vektor: \$FFFE

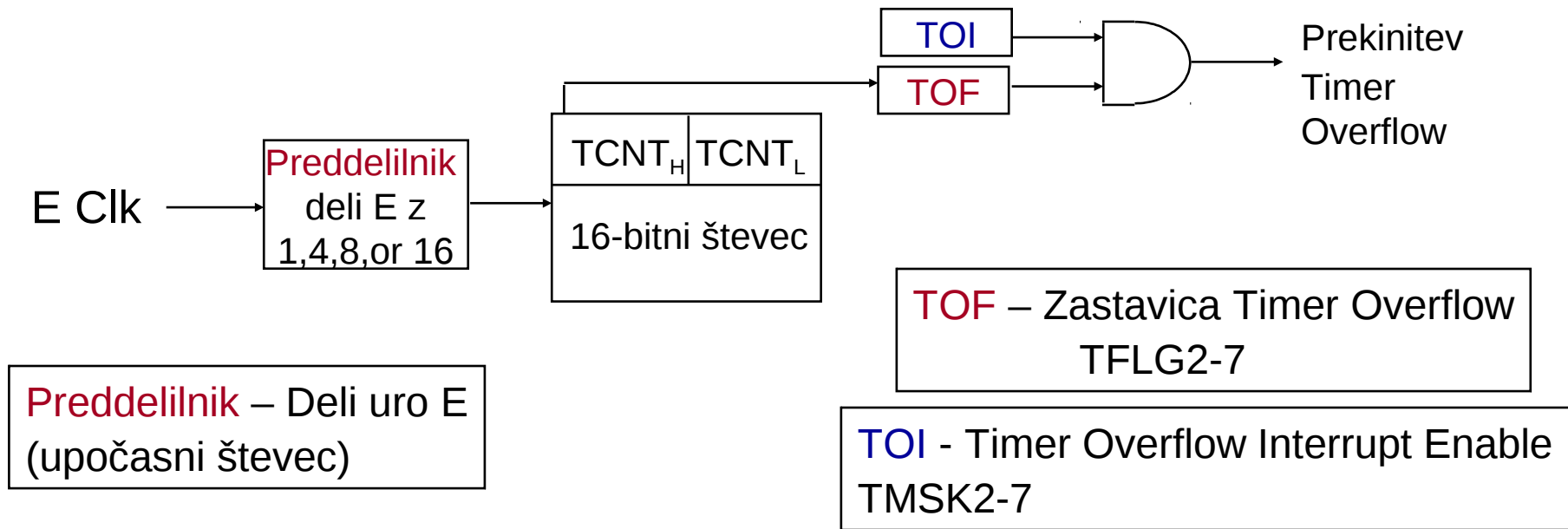
```
org    $FFFE
fdb    start
```

Časovniki (timers)

- **Pogosto moramo delati stvari, povezane s časom**
 - Redne prekinitve
 - Večopravilnost – dodeljevanje časa
 - Osveževanje prikazovalnikov (LCD)
 - Časovne zakasnitve
 - Alarmi
 - Delo z določenimi napravami
 - Merjenje časa
 - Iščemo čas med dvema dogodkoma
 - Določamo frekvenco periodičnih signalov

Čas in HC11

- **16 bitni števec**, ki se neprestano povečuje
 - Dobi vrednost \$0000 ob resetu
 - Lahko ga preberemo na lokacijah \$100E - \$100F (TCNT_H and TCNT_L)
 - Ob prehodu iz 65535 na 0 se postavi bit TOF



Časovnik

- Vrednost časovnika lahko preberemo s 16-bitnim branjem registra TCNT (\$100E)
 - LDX \$100E ; preberi časovnik v X
- Časovnik se običajno poveča ob vsakem ciklu ure, vendar ga lahko upočasnimo, da se povečuje vsakih 4,8, ali 16 ciklov.
 - To lahko storimo le prvih 64 ciklov po resetu
- TOF se postavi, ko pride do prekoračitve
 - Brišemo ga s pisanjem '1' vanj!!!

Ostalih bitov ne smemo spremeniti.

TMSK2 (\$1024)	TOI	RTII	PAOII	PAII	0	0	PR1	PR0
Po resetu:	0	0	0	0	0	0	0	0
TFLG2 (\$1025)	TOF	RTIF	PAOVF	PAIF				
Po resetu:	0	0	0	0	0	0	0	0

PR1,PR0 (preddelilnik):
00: 1 (clock)
01: 4 (clock / 4)
10: 8 (clock / 8)
11: 16 (clock / 16)

Brisanje zastavic

- Zastavice **brišemo** s pisanjem '1' na njihovo mesto
 - Za brisanje zastavice TOF, vpišemo '1' v bit 7 registra TFLG2
 - Ostalih bitov ne smemo spremeniti!

- **Rešitev:**

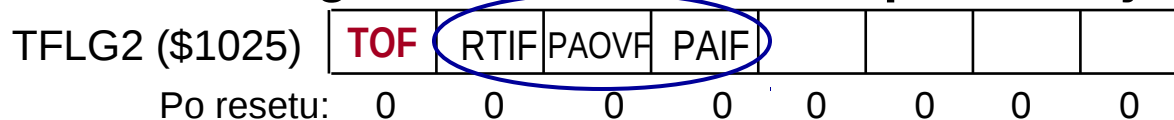
- LDAA #%10000000
 - STAA TFLG2 ;clear TOF

10010000 - vrednost reg.
10000000 - maska
10010000 - rezultat 'or'
Pobriše TOF in PAIF

- **Napaka:**

- LDX #TFLG2
 - BSET 0,X #%10000000

- Na ta način se pobrišejo vsi biti, ki so bili postavljeni na 1. Prebere register, OR z masko, zapiše nazaj.



Delo s časovnikom

1 E perioda = 1 / 1.2288MHz

```
TCNT    EQU    $100E
```

```
* Generate a 10ms delay by waiting 12288 cycles
```

```
        LDD    TCNT    ; D <-- vrednost časovnika
        ADDD   #12288  ; D <-- vrednost časovnika po 10ms
LOOP    CMPD   TCNT    ; je časovnik že enak ali večji D
        BHI    LOOP    ; če ne, 10ms še ni poteklo
        ...
```

- **Težave:**

- Kaj, če pri prištevanju 12288 k D povzroči prenos?
 - To se zgodi, če je števec > 53247
- Zakasnitve, večje od 65536 ciklov (53.3 ms), niso možne
- Slabša natančnost

Zaradi navedenih težav in slabosti se ta način ne uporablja!

Prekinitve 'Realnega Časa' (RTI)

- **Prekinitve 'Realnega Časa' so izvor periodičnih prekinitev**
 - Če so omogočene, povzročijo prekinitev v enakomernih časovnih intervalih
 - Dolžine časovnih intervalov so lahko različne:

		<u>Ura E</u>	
<u>RTR1</u>	<u>RTR0</u>	<u>Deljena z:</u>	<u>Perioda</u>
0	0	8K	6.2/3ms
0	1	16K	13.1/3ms
1	0	32K	26.2/3ms
1	1	64K	53.1/3ms

PACTL (\$1026)	DDRA7	PAEN	PAMOD	PEDGE	0	0	RTR1	RTR0
Po resetu:	0	0	0	0	0	0	0	0

Prekinitve 'Realnega Časa' (RTI)

Ko določena perioda poteče, se postavi zastavica RTIF in (glede na RTII) generira prekinitvev

TMSK2 (\$1024)	TOI	RTII	PAOII	PAII	0	0	PR1	PR0
Reset to:	0	0	0	0	0	0	0	0
TFLG2 (\$1025)	TOF	RTIF	PAOVF	PAIF				
Reset to:	0	0	0	0	0	0	0	0

Omogočimo prekinitve RTI:

```
LDX    #TMSK2
BSET   0,x  #%01000000
```

Zakaj v TMSK 2 ne vpišemo #\$40?
--> S tem bi 'povozili' vse ostale bite!

Brisanje zastavice RTI (npr. v PSP):

```
LDAA  #%01000000
STAA  TFLG2
```

RTI - prekinitveni vektor: \$FFF0-\$FFF1

Daljše periode z RTI

Najdaljša perioda za RTI je 53.33 ms. Če želimo daljše periode, moramo šteti prekinitve => dobimo večkratnike periode.

```
TMASK2    EQU    $1024
TFLG2     EQU    $1025
PACTL     EQU    $1026
```

```
ONESEC    EQU    75      ;število RTI-jev na 13.33ms za skupno 1s
```

```
RTICOUNT  org    $2000
           RMB    1      ; števec koliko RTI-jev se je zgodilo
```

**** nastavitve prekinitvenega-vektorja za RTI

```
           org    $FFF0
           fdb    RTIHAND
```

```
           org    $E000
```

lds	...		; inicializacija sklada (kazalca)
ldaa	#ONESEC		; inicializacija števca
staa	RTICOUNT		; "
ldx	#PACTL		; nastavi PACTL za RTI na 13.33ms
bclr	0,x	\$02	; tako da daš RTR1,0 na '01'
bset	0,x	\$01	; "
ldaa	##%01000000		; pobriši RTI zastavico
staa	TFLG2		; tako da vpišeš '1' v RTIF v TFLG2
ldx	#TMASK2		; omogoči RTI-je
bset	0,x	##%01000000	; tako da postaviš RTII v TMASK2 na 1
cli			; globalno omogoči prekinitve
loop	bra	loop	; neskončna zanka

Nadaljevanje



Daljše periode z RTI

```
RTIHAND:  ldaa  #%01000000    ; pobriši RTI zastavico  
          staa  TFLG2          ;
```

```
          dec   RTICOUNT      ; zmanjšaj števec
```

```
          bne   RTIDONE       ; če ni nič, nadaljuj
```

***** Naslednji del kode se izvrši enkrat na sekundo *****

```
          bsr   UPDATECLOCK    ; kliči uporabniški podprogram  
  
          ldaa  #ONESEC        ; ponastavi (resetiraj) števec  
  
          staa  RTICOUNT       ; shrani števec  
RTIDONE  rti
```

Alarmi

- **Alarme realiziramo s funkcijo 'Output compare'**
 - Čas, ob katerem naj se sproži alarm, zapišemo v register
 - Ko je vrednost števca enaka vrednosti v registru, se postavi zastavica
 - Če je omogočena, se sproži prekinitev. (možno je tudi vpisati vrednost na izhodni priključek)
- **Osnovna ideja**
 - Preberemo trenutno vrednost števca
 - K vrednosti prištejemo število ciklov, po katerem naj se alarm sproži
 - Rezultat vpišemo v register 'Output compare'
 - Pobrišemo ustrezno zastavico in čakamo...
- **Daljše periode realiziramo s štetjem alarmov (podobno kot v primeru RTI)**

Nadzor nad 'Output Compare'

<u>OC Register</u>	<u>Naslov</u>
TOC1	\$1016-\$1017
TOC2	\$1018-\$1019
TOC3	\$101A-\$101B
TOC4	\$101C-\$101D
TOC5	\$101E-\$101F

Registri 'Output Compare':
5 registrov == 5 alarmov

TFLG1 (\$1023)	OC1F	OC2F	OC3F	OC4F	OC5F	IC1F	IC2F	IC3F
Po resetu:	0	0	0	0	0	0	0	0

TMSK1 (\$1022)	OC1I	OC2I	OC3I	OC4I	OC5I	IC1I	IC2I	IC3I
Po resetu:	0	0	0	0	0	0	0	0

TCTL1 (\$1020)	OM2	OL2	OM3	OL3	OM4	OL4	OM5	OL5
Po resetu:	0	0	0	0	0	0	0	0

Vsak OC ima tudi priključek, povezan z njim:
Lahko ga postavimo na 0, 1 ali ga invertiramo,
ko se alarm sproži...

Zastavice OC Flags se postavijo, ko se alarm sproži.
Brišemo s pisanjem '1'.

OC Interrupt enable: '1'
pomeni, da OC generira prekinitev.

<u>OMx</u>	<u>OLx</u>	<u>Vpliv na priključek</u>
0	0	Ni vpliva
0	1	Invertira
1	0	Postavi na 0
1	1	Postavi na 1

Funkcija 'Input Capture'

- Funkcija 'Input Capture' deluje glede na stanje treh vhodnih priključkov (to so priključki 68hc11, ne vrata A čipa PIA)
 - PA_2 , PA_1 , PA_0
- Če na enem od teh priključkov pride do spremembe stanja (fronte), se zgodi naslednje:
 - Vrednost števca se vpiše v ustrezen register (TIC1, TIC2, or TIC3)
 - Postavi se ustrezna zastavica (IC1F, IC2F, ali IC3F v registru TFLG1)
 - Generira se prekinitev (če želimo)

Uporaba 'Input Capture'

- **Določanje periode signala**
 - Izmerimo čas med dvema pozitivnima (ali negativnima) frontama
- **Iskanje dolžine impulza**
 - Izmerimo čas med pozitivno in negativno fronto signala
- **Zelo natančno določimo čas zunanjega dogodka**
 - Mnogo natančneje, kot z uporabo prekinitev – do cikla natančno
- **Kot tri dodatne priključke za zunanje prekinitve**

Prekinitveni vektorji in pomembni registri

Prekinitveni vektorji:

FFDE			timer overflow
FFE0			timer output compare 5
FFE2			timer output compare 4
FFE4			timer output compare 3
FFE6			timer output compare 2
FFE8			timer output compare 1
FFEA			timer input capture 3
FFEC			timer input capture 2
FFEE			timer input capture 1
FFF0			real time interrupt
FFF2			IRQ' pin interrupt
FFFE			reset

Registri za delo s časovniki:

TCTL1 \$1020

TFLG2 \$1025

TMSK1 \$1022

PACTL \$1026

TFLG1 \$1023

TCNT \$100E

TMSK2 \$1024