Građa računala

Vježbe 11
Iznimke i prekidi 2

Interrupt

- Kada neka komponenta računala želi komunicirati s procesorom, šalje prekid (interrupt)
- Prekidi su organizirani hijerarhijski
  - 7 razina
  - *Interrupt bit mask* – u statusnom registru
  - Ukoliko je razina prekida manja ili jednaka interrupt bit mask, prekid se ignorira

Prekidi – tablica vektora

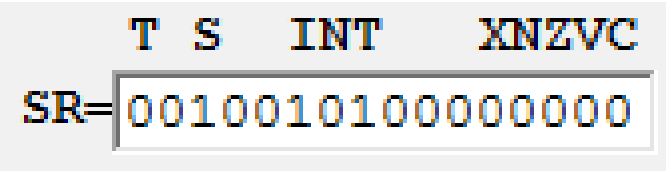
Vector Number	Offset	Assignment
0	000	Reset: Initial interrupt stack pointer
1	004	Reset: Initial program counter
2	008	Bus error
3	00C	Address error
4	010	Illegal instruction
5	014	Divide by zero
6	018	CHK, CHK2 instruction
7	01C	cpTRAPcc, TRAPcc, TRAPV instruction
8	020	Privilege violation
9	024	Trace
10	028	A-line emulator
11	02C	F-line emulator
12	030	Reserved
13	034	Coprocessor protocol violation
14	038	Format error
15	03C	Unitialized interrupt
16-23	040-05C	Reserved
24	060	Spurious interrupt
25	064	Autovector (level 1)
26	068	Autovector (level 2)
27	06C	Autovector (level 3)
28	070	Autovector (level 4)
29	074	Autovector (level 5)
30	078	Autovector (level 6)
31	07C	Autovector (level 7)
32-47	080-0BC	TRAP #0-15
48	0C0	FPCP Branch or set on unordered condition
49	0C4	FPCP Inexact result
50	0C8	FPCP Divide by zero
51	0CC	FPCP Underflow
52	0D0	FPCP Operand error
53	0D4	FPCP Overflow
54	0D8	FPCP Signaling NAN
55	0DC	Reserved
56	0E0	PMMU configuration
57	0E4	PMMU illegal operation
58	0E8	PMMU access level
59-63	0EC-0FC	Reserved
64-255	100-3FC	User defined vectors

FPCP=floating point coprocessor
PMMU=paged memory management unit

- 2 načina obrade:
  - Autovektori
  - Vektori koje primamo od komponente
- Mi ćemo se baviti autovektorima
  - Indeksi 25-31

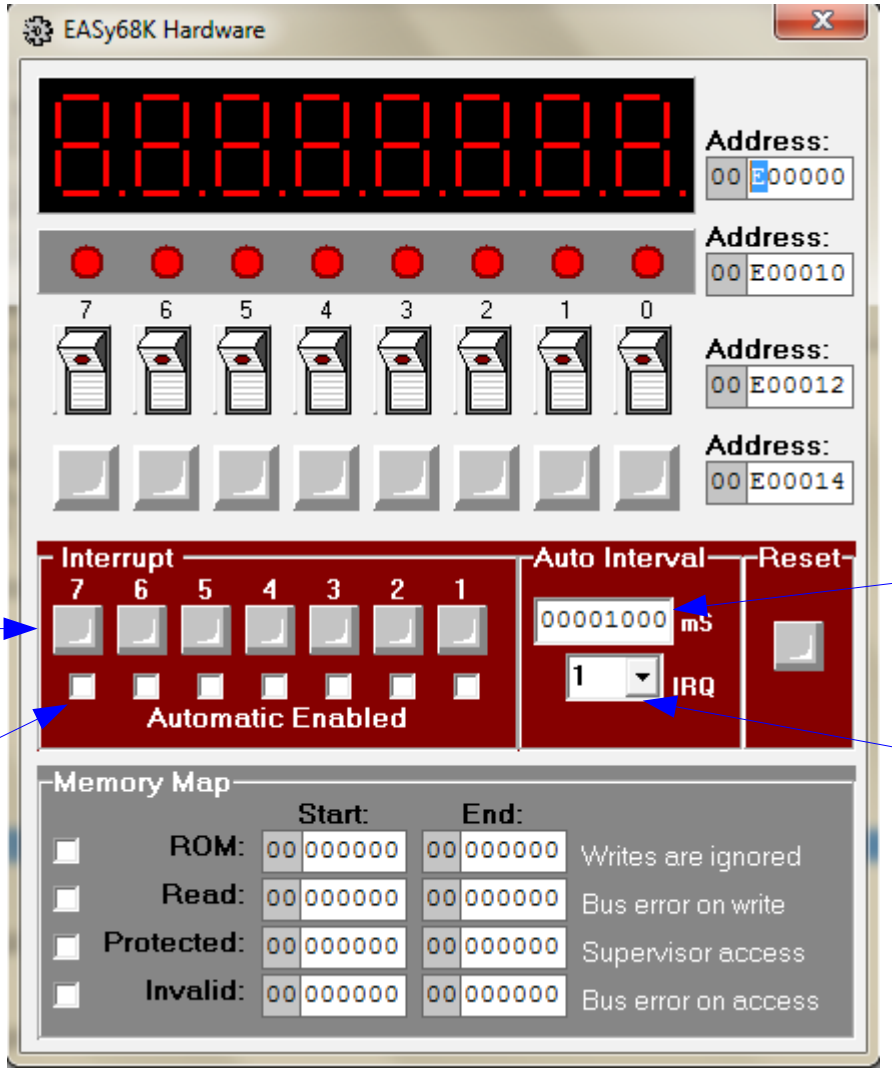
Autovektori

- *Interrupt bit mask*
  - 3 bita statusnog registra



- Bitovi određuju vektor za obradu
  - Jedan vektor po razini!
  - Općenito, računalo treba "ispitati" komponente da vidi koja je tražila prekid (nije moguće u simulatoru)
- U Easy68k ćemo
  - Ručno uzrokovati prekid proizvoljne razine ili
  - Postaviti periodično uzrokovanje prekida u "Hardware" prozoru
  - Uzrokovati prekid tipkovnicom

Prekidi - hardver



Gumbi za uzrokovanje prekida

Označiti kućicu da bi se prekid izvodio periodično

Vrijeme između dvaju prekida razine odabrane u izborniku

Prekidi

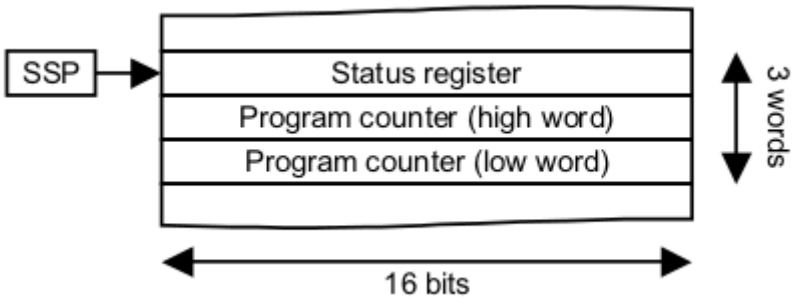
- Definiranje je ekvivalentno definiranju ostalih iznimaka

```
START:
    MOVE.L   #IRQ1, $64
    MOVE.L   #IRQ2, $68
    MOVE.L   #IRQ3, $6C
    MOVE.L   #IRQ4, $70
    MOVE.L   #IRQ5, $74
    MOVE.L   #IRQ6, $78
    MOVE.L   #IRQ7, $7C

    ... kod ...
```

```
IRQ1:  obrada IRQ1
       RTE

...

IRQ6:  obrada IRQ6
       RTE

IRQ7:  obrada IRQ7
       RTE
```

Prekidi - stack

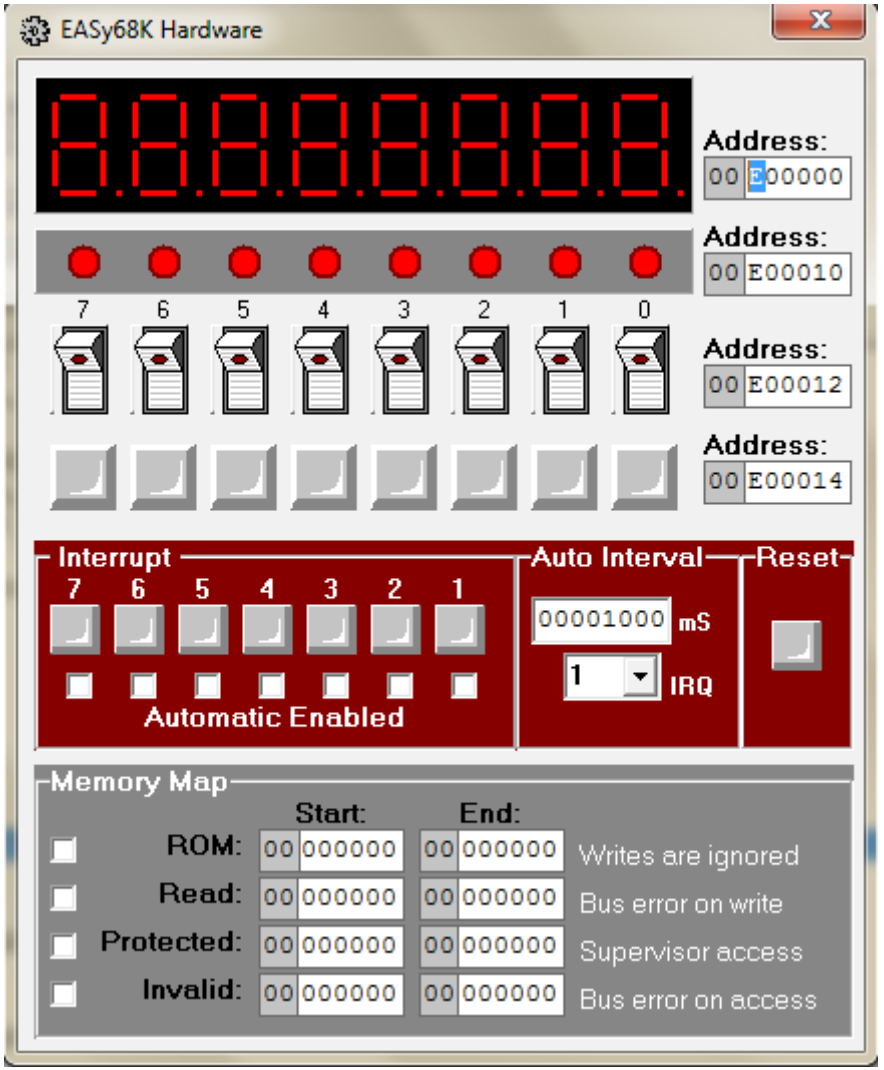
- Redoslijed izvršavanja
  1) Trenutna instrukcija se dovrši
  2) Na stack se stavljaju SR i PC
  3) Postavlja se razina prekida
  4) Izvršava se kod za obradu
  5) Povratak – pomoću RTE instrukcije



Napomena: TRAP #15 pozivi se ne mogu prekinuti!

Još malo o hardveru

- Hardver prozor daje i 4 "uređaja" za manipulaciju memorije
- Zadaje se adresa na kojoj ti uređaji "žive"
- Hardveru možemo pristupati i programski!
  - Help -> TRAP -> Simulator Environment



Vanjski uređaji

- LED
  - Možemo vidjeti stanje jednog bajta memorije
- Digitalni ekran
  - Znamenke zauzimaju svaka po bajt
- Prekidači i gumbi
  - Pišu 1 na odgovarajući bit kada su uključeni, 0 kada su isključeni
  - Prekidač je uključen sve dok ga ne isključimo
  - Gumb je uključen samo dok ga držimo pritisnutim

```
    ORG    $1000
START:
    ; dohvaćamo ledice
    MOVE.L  #32, D0
    KOMOVE  #2, D1
    TRAP #15

    ; gasimo sve osim
    ; druge zdesna
    MOVE.L  D1, A1
    MOVE.B  #%00000010, (A1)
    END    START
```

Vanjski uređaji

Digitalni ekran:

- Jedna znamenka = jedan bajt
- Jedan bajt je prazan između svake znamenke
- Kodovi
  - Kao u zadatku v5-06.x68

```
    ORG    $1000
START:
    ; dohvaćamo ekran
    MOVE.L  #32, D0
    MOVE.B  #1, D1
    TRAP #15

    ; napišemo 1234
    MOVE.L  D1, A1
    MOVE.B  #$06, (A1)
    MOVE.B  #$5B, 2(A1)
    MOVE.B  #$4F, 4(A1)
    MOVE.B  #$66, 6(A1)
    END    START
```

Ponavljanje - TRAP

- Instrukcija za uzrokovanje iznimki
  - Omogućuje pozivanje sistemskih metoda bez da "kršimo pravila"
  - "neiznimne" iznimke jer spadaju u standardni rad procesora
- Vektori 32 – 47 (dekadski)
  - 16 vektora ali uz mogućnost prenošenja dodatnih parametara u registrima
  - TRAP #0 – TRAP #15
- TRAP #15 – preddefinirane iznimke za lakši rad s emulatorom (specifično za Easy68k emulator)
  - Text I/O
  - File I/O
  - Network I/O itd.

Neke TRAP mogućnosti

- TRAP #15 - zadatak stavljamo u D0
  - #8 – ispis vremena od ponoći (u stotinkama)
  - Crtanje (Help -> Graphics)
    - #80 – definira boju, BGR kod stavljamo u donja 24 bita D1
    - #82 – crta piksel, koordinate su u D1 i D2
    - #95 – ispis teksta na proizvoljne koordinate, koordinate u D1 i D2
    - #84 – crtanje linije, koordinate od D1,D2 do D3,D4

Crtanje

- Moramo prvo postaviti boju "olovke"

```
    ORG    $1000
START:
    ; postavljamo boju za crtanje
    MOVE.L  #80, D0
    MOVE.L  #$0000FF00, D1
    TRAP    #15

    ; crtamo liniju
    MOVE.L  #50, D1
    MOVE.L  #50, D2
    MOVE.L  #125, D3
    MOVE.L  #160, D4
    MOVE.L  #84, D0
    TRAP    #15

     END     START
```

Input s tipkovnice

- Zadatak
  - #62 – uključi/isključi IRQ tipkovnice
  - #19 – provjeri koji je gumb pritisnut
- Napomena – pogledati
  - Help -> keycodes
  - Help -> peripheral I/O

```
    ORG    $1000
START:
    ; postavljamo IRQ1 vektor
    MOVE.L  #OBRADA, $64

    ; postavljamo tipkovnicu na IRQ1
    MOVE.L  #62, D0
    MOVE.L  #$0103, D1
    TRAP #15

LOOP:
    NOP
    BRA LOOP

OBRADA:
    MOVE.L  #19, D0
    MOVE.L  #'A'<<24+'S'<<16+'D'<<8+'F', D1
    TRAP #15
    RTE

    END     START
```