

# Operacijski sistemi



Večopravilnost

# Vsebina

- Virtualizacija procesorja
  - večprogramiranje, večprocesiranje
- Večopravilnost
  - sodelovalna in prevzemna
- Neposredno izvajanje
  - neomejeno neposredno izvajanje
  - omejeno neposredno izvajanje
- Preklop procesa

# Virtualizacija procesorja

- Izziv
  - Čeprav je na voljo le en (ali nekaj) procesorjev, se lahko sočasno izvaja poljubno procesov.

Kako ustvariti utvaro  
ogromno procesorjev?





# Virtualizacija procesorja

- Število procesorjev
  - **večprogramiranje / multiprogramiranje**
    - več programov naloženih v pomnilniku, pripravljenih za izvajanje
    - sočasno izvajanje več programov oz. procesov na enem procesorju
    - poudarek na programih oz. procesih in boljši izkoriščenosti virov
  - **večprocesiranje / multiprocesiranje**
    - vzporedno izvajanje več programov oz. procesov na več procesorjih
    - poudarek na procesorjih
      - procesorji si delijo pomnilnik, vodila, naprave itd.
      - procesor: večjedrni (multicore), večnitni (multithreading)

# Virtualizacija procesorja


- Souporaba vira med več entitetami
  - **časovno dodeljevanje** (time sharing)
    - vsaka entiteta uporablja vir nekaj časa
    - **časovna rezina**: čas uporabe vira
    - deljenje procesorja, omrežne povezave
  - **prostorsko dodeljevanje** (space sharing)
    - vsaka entiteta uporablja nek del vira
    - deljenje pomnilnika, diskovnega prostora

# Večopravilnost

- **Večopravilnost** (multitasking)
  - izvajanje več procesov, vrsta večprogramiranja
  - izvedba večprogramiranja preko časovnega dodeljevanja
  - vrsti večopravilnosti
    - **sodelovalna** / brez odvzemanja
      - (cooperative, non-preemptive)
    - **prevzemna** / z odvzemanjem
      - (preemptive)

# Večopravilnost

- **Sodelovalna večopravilnost**
  - temelji na *sodelovanju* procesov
    - Kdaj proces prepusti procesor?
      - znotraj sistemskih klicev
        - običajni klici read, write, mkdir itd.
      - eksplicitno prepuščanje procesorja
        - sistemski klic **yield**
    - predpostavlja se razumno obnašanje procesov
    - računsko intenzivni procesi lahko ugrabijo procesor
      - dolgotrajni tek procesa brez sistema klica
      - potrebno eksplicitno prepuščanje procesorja



**while** (true) {};



# Večopravilnost

- **Prevzemna večopravilnost**

- temelji na prevzemanju procesorja oz. prekinjanju procesa
  - uporaba prekinitve ure (časovnik, timer)
  - znotraj prekinitvenih rokovalnikov ima nadzor OS

- **časovna rezina**

- čas, ki ga ima proces na voljo za izvajanje
- po izteku rezine se procesor dodeli drugemu procesu
- prekinjeni proces pa se postavi v vrsto
- večopravilnost s časovnim dodeljevanjem



# Večopravilnost

Kako procesu prepustiti  
procesor brez, da bi ga  
ugrabil za vedno?



# Neposredno izvajanje

- **Neomejeno** neposredno izvajanje
  - ideja: samo poženi program

OS	Program
ustvari procesni deskriptor alociraj naslovni prostor naloži program v pomnilnik pripravi sklad in registre poženi program, npr. main()  sprosti pomnilnik procesa sprosti ostale vire procesa sprosti procesni deskriptor	izvedi program končaj proces, npr. exit()

# Neposredno izvajanje

- **Neomejeno neposredno izvajanje**
  - hitrost izvajanja
    - program se direktno izvede na procesorju
  - izvajanje v celoti
    - program se v celoti izvede na procesorju
  - neizkoriščenost virov
    - proces čaka na dokončanje V/I operacij
  - težak nadzor
    - program je nemogoče nadzorovati, kaj počne z viri
- uporaba (stari sistemi)
  - paketna obdelava poslov
  - OS le kot knjižnica



# Neposredno izvajanje

- **Težava**

- OS je program, ki ne teče vedno
- pogosto izgubi nadzor na sistemom

- **Rešitev**

- strojna podpora
  - zaščiteni in privilegirani način izvajanja
  - preklon načina izvajanja
  - mehanizem prekinitev
  - mehanizem sistemskih klicev

**anti-primerjava:**  
termostat, ki upravlja ogrevanje



# Neposredno izvajanje

- **Kdaj OS pridobi nadzor?**

- **ob strojni prekinitvi**

- npr. sistemska ura, V/I dogodek, ...

- **ob izjemi pri izvajanju procesa**

- npr. deljenje z 0, zgrešitev strani, ...

- **ob sistemskem klicu**

- npr. ustvarjanje procesa, branje datoteke, ...

# Neposredno izvajanje

- **Kako OS pridobi nadzor?**
  - *proženje* ustreznega rokovalnika
    - PSP – prekinitveni storitveni program
    - **vstop** v jedrni rokovalnik
      - statusni register in PC se porineta na sklad
      - prekinitve se onemogočijo
      - nivoja zaščite procesorja se preklopi na jedrni
      - izvedba rokovalnika
    - **vrnitev** iz rokovalnika v uporabniški proces
      - PC in statusni register se vzameta iz sklada
      - nivo zaščite procesorja se preklopi na uporabniški
  - za ostale registre poskrbi rokovalnik sam



# Neposredno izvajanje

- **Omejeno** neposredno izvajanje

- ideja

- inicializiraj rokovalnike prekinitev in pasti
    - inicializiraj rokovalnik systemskega klica
    - inicializiraj časovnik (prevzemna večopravilnost)
    - poženi program in počakaj, da OS spet dobi nadzor
    - ko dobi nadzor se OS odloči, kako naprej

# Neposredno izvajanje

- **Omejeno** neposredno izvajanje

OS	Strojna oprema	Program
inicializacija	hrani naslove sistemskih rokovalnikov	
stvaritev procesa IRET	izstop iz rokovalnika	izvedi program ... sistemski klic
obdelaj past obdelaj sistemski klic IRET	vstop v rokovalnik  izstop iz rokovalnika	  izvedi program ....

# Preklop procesa

- Večopravilnost

- več procesov je sočasno naloženih v sistemu
- **časovno dodeljevanje** za souporabo procesorja



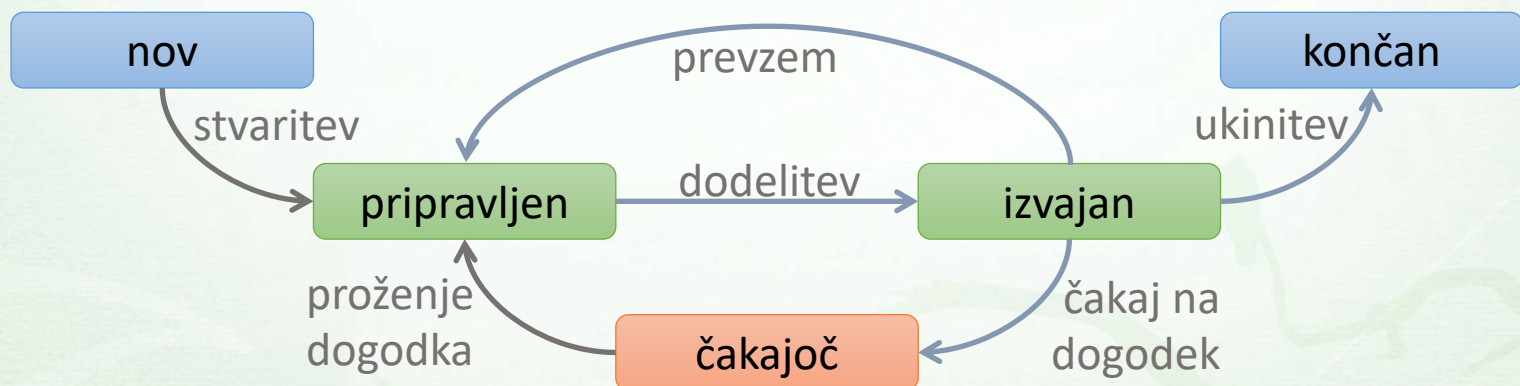
- potrebno je **razvrščanje** procesov na procesorju
- potrebujemo mehanizem zamenjave procesa

Preklop procesa



# Preklop procesa

- **Kdaj** se izvede preklop procesa?
  - na **koncu** rokovalnika, tik preden se vrnemo nazaj v uporabniški način
  - preklop ni vedno potreben
    - lahko se vrnemo v isti uporabniški proces iz katerega smo prišli



# Preklop procesa

- Izvedba preklopa procesa
  - izbira (drugega) pripravljenega procesa (razvrščevalnik)
  - spremembe stanj procesov
    - odhajajoči proces se postavi v ustrezno vrsto in stanje
    - prihajajoči proces postane izvajan
  - preklop konteksta
    - kontekst: PC, sklad, statusni register, ostali registri
    - trenutni kontekst se shrani v deskriptor **odhajajočega** procesa
    - obnovitev konteksta iz deskriptorja **izbranega** procesa
    - menjava naslovnega prostora
  - razno
    - posodobitev računovodskih podatkov
    - praznenje preslikovalnega pomnilnika (TLB)
    - praznenje napovedovalnika skokov

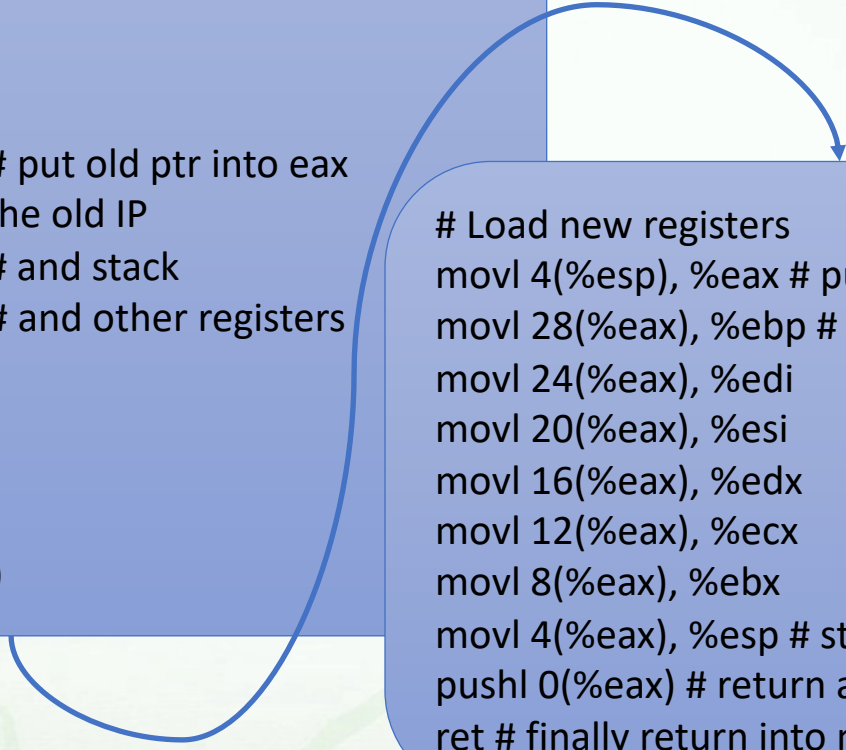
# Preklop procesa

- Preklop konteksta

```
# void swtch(struct context*old, struct context*new);  
# Save current register context in old  
# and then load register context from new  
.globl swtch  
swtch:
```

```
    # Save old registers  
    movl 4(%esp), %eax # put old ptr into eax  
    popl 0(%eax) # save the old IP  
    movl %esp, 4(%eax) # and stack  
    movl %ebx, 8(%eax) # and other registers  
    movl %ecx, 12(%eax)  
    movl %edx, 16(%eax)  
    movl %esi, 20(%eax)  
    movl %edi, 24(%eax)  
    movl %ebp, 28(%eax)
```

```
    # Load new registers  
    movl 4(%esp), %eax # put new ptr into eax  
    movl 28(%eax), %ebp # restore other registers  
    movl 24(%eax), %edi  
    movl 20(%eax), %esi  
    movl 16(%eax), %edx  
    movl 12(%eax), %ecx  
    movl 8(%eax), %ebx  
    movl 4(%eax), %esp # stack is switched here  
    pushl 0(%eax) # return addr put in place  
    ret # finally return into new ctxt
```





# Preklop procesa

- Učinkovitost

- strojna oprema

- hitrost glavnega pomnilnika (hranjenje deskriptorja)
    - arhitektura procesorja
      - registri, posebni ukazi za obdelavo vseh registrov
    - stanje procesorja
      - flush predpomnilnik, flush TLBs, flush branch predictor

- operacijski sistem

- zapletenost komponent
  - razvrščevalni algoritem za izbiro procesa
  - vzdrževalna opravila
    - čiščenje, staranje procesov, enakomerna obremenitev virov, zaznavanje in razreševanje smrtnih objemov itd.

# Preklop procesa

- Učinkovitost

- Koliko časa traja preklop procesa?

- Koliko časa traja sistemski klic?

- 1996, Linux 1.3.37, 200 MHz P6 CPU

- syscall: 4  $\mu$ s, context switch: 6  $\mu$ s

- modern systems 3 GHz

- pod 1  $\mu$ s

- Ukaz `lmbench`

Processor, Processes - times in microseconds - smaller is better

Host	OS	Mhz	null call	null I/O	stat	open clos	slct TCP	sig inst	sig hdl	fork proc	exec proc	sh proc	
BigMac.lo	Darwin	18.7.0	2800	<b>0.44</b>	<b>0.94</b>	<b>4.87</b>	<b>13.6</b>	<b>19.7</b>	<b>0.71</b>	<b>4.13</b>	<b>541.</b>	<b>2681</b>	<b>4834</b>

# Preklop procesa

- Izzivi

- Kaj se zgodi, če se sredi izvajanja systemskega klica proži še časovnik?
- Kaj se zgodi, če se sredi rokovanja prekinitve proži še ena prekinitve?

- Ideje

- onemogočanje prekinitvev
- tehnike zaklepanja internih podatkovnih struktur
- ... več v nadaljevanju