Operacijski sustavi

Četvrta laboratorijska vježba

10. ožujka 2023.

Sadržaj

1 Motivacija

Cilj ove laboratorijske vježbe je proučiti rad sustava straničenja i izolacije adresnih prostora procesa u modernim računalima. U sklopu vježbe potrebno je ostvariti simulaciju rada više procesa na jednostavnom računalu koje koristi straničenje.

2 Zadatak

2.1 Simulacija straničenja (4 boda)

Vaš je zadatak ostvariti simulaciju rada više procesa u sustavu koji koristi mehanizam straničenja na zahtjev pomoću arhitekture prikazane na slici 1.

Simulirani sustav se sastoji od N procesa, diska, niza od M okvira i tablice straničenja za svaki simulirani proces. Vaš program kao ulazne parametre prima broj okvira M i broj procesa N te uz ponašanje navedeno u pseudokodu 1 mora sadržavati sljedeće strukture podataka:

- `disk[N]` - Simulirani disk koji služi za pohranu sadržaja stranica,
- `okvir[M]` - Simulirani radni spremnik od M okvira veličine 64 okteta,
- `tablica[N]` - Tablica prevođenja za svaki od N procesa.

Algoritam 1: Pseudokod simulacije.

```
1  za i = 1 do N čini
2  |  stvori proces i;
3  |  inicijaliziraj tablicu straničenja procesa i;
4  kraj
5  t ← 0;
6  ponavljaj
7  |  za svaki proces p čini
8  |  |  x ← nasumična logička adresa;
9  |  |  i ← dohvati_sadržaj(p, x);
10 |  |  i ← i + 1;
11 |  |  zapiši_sadržaj(p, x, i);
12 |  |  t ← t + 1;
13 |  |  spavaj;
14 |  kraj
```

Algoritam 2: Pseudokod pomoćnih funkcija.

```
1  Funkcija dohvati_sadržaj(p, x)
2      y ← dohvati_fizicku_adresu(p, x);
3      i ← vrijednost na adresi y;
5      vrati i;
1  Funkcija zapiši_vrijednost(p, x, i)
2      y ← dohvati_fizicku_adresu(p, x);
3      zapiši vrijednost i na adresu y;
1  Funkcija dohvati_fizicku_adresu(p, x)
2      nađi zapis tablice straničenja procesa p za adresu x;
3      ako adresa x nije prisutna onda
4          ispiši promašaj;
5          pronađi i dodijeli okvir;
6          učitaj sadržaj stranice s diska;
7          ažuriraj tablicu prevođenja procesa p;
8      kraj
9      ispiši adresu x, adresu njenog okvira te sadržaj zapisa u tablici
        prevođenja;
11     vrati fizička adresa;
```

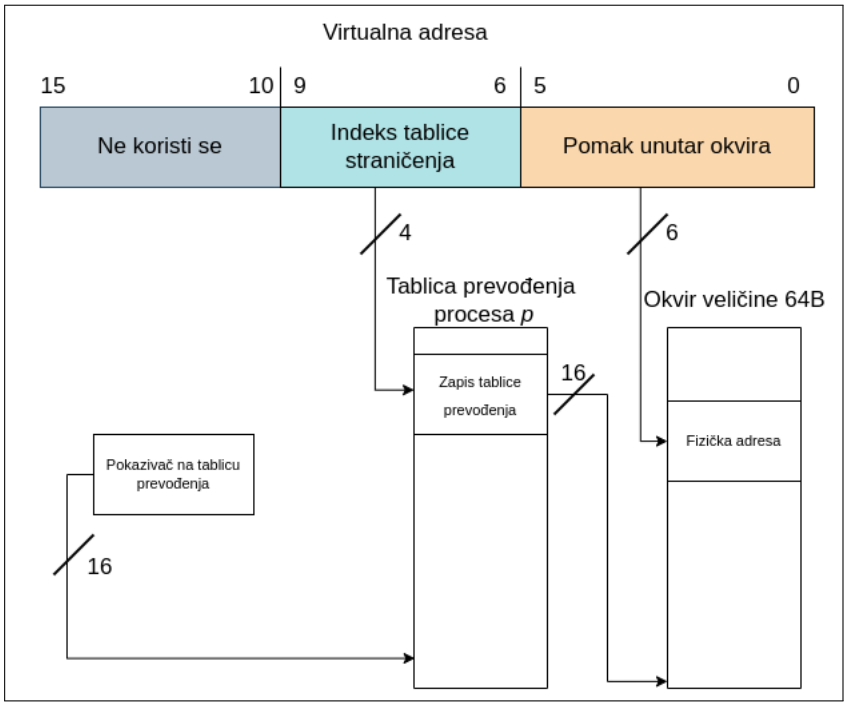
Pomoćne funkcije `dohvati_sadržaj` i `zapiši_vrijednost` služe za pristupanje adresnom prostoru procesa p. Svako rješenje koje simuliranom spremniku pristupa pored ovih funkcija **smatrat će se neispravnim**.

Prilikom pronalaženja slobodnog okvira za dodjeljivanje potrebno je koristiti strategiju zamjene stranica *Least Recently Used (LRU)*. Za implementaciju *LRU* strategije potrebno je koristiti varijablu `t`, čija se vrijednost koristi kao sat. U strukturi zapisa tablice prevođenja (prikazan na slici 2) predviđeno je 5 bitova za pohranu *LRU* metapodataka za stranicu. Ako vrijednost tog polja u bilo kojoj stranici dođe na 31, potrebno je postaviti vrijednost globalne varijable `t` na 0, vrijednosti svih *LRU* metapodataka za sve zapise u svim tablicama straničenja na 0, te *LRU* metapodatak za trenutnu stranicu na 1. Prilikom zamjene odnosno izbacivanja stranica pretpostavite da je njihov sadržaj uvijek mijenjan te ga pohranite na disk. Svaki zapis unutar tablice straničenja mora ostvarivati bit prisutnosti u šestom bitu zapisa.

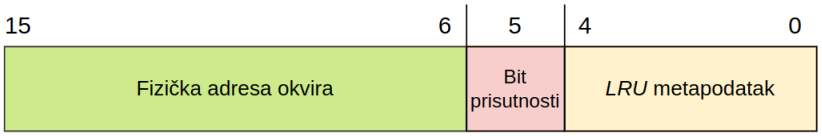
Za potrebe generiranja nasumične logičke adrese preporučamo da generirate samo **parne** adrese kako bi izbjegli problematične rubne slučajeve sa čitanjem na krajevima okvira, što možete postići primjenom logičke operacije **I** sa vrijednosti `0x3FE` na generirani broj.

2.1.1 Kratki opis simulirane arhitekture

Simulirani procesor koristi 16-bitne logičke adrese s veličinom okvira/stranice od 64 okteta. Pretpostavite da se brojevi spremaju prema principu *little-endian*. Struktura virtualne adrese prikazana je na slici 1. Za potrebe prevođenja virtualnih adresa koristi se tablica straničenja s jednom razinom. Radi lakše simulacije prvih 6 bitova logičke adrese se **ne koristi**, čime je logički adresni prostor **ograničen na 1024 okteta**. Iduća 4 bita virtualne adrese koriste kao indeks u tablici straničenja, a zadnjih 6 bitova koriste se kao pomak unutar fizičkog okvira.



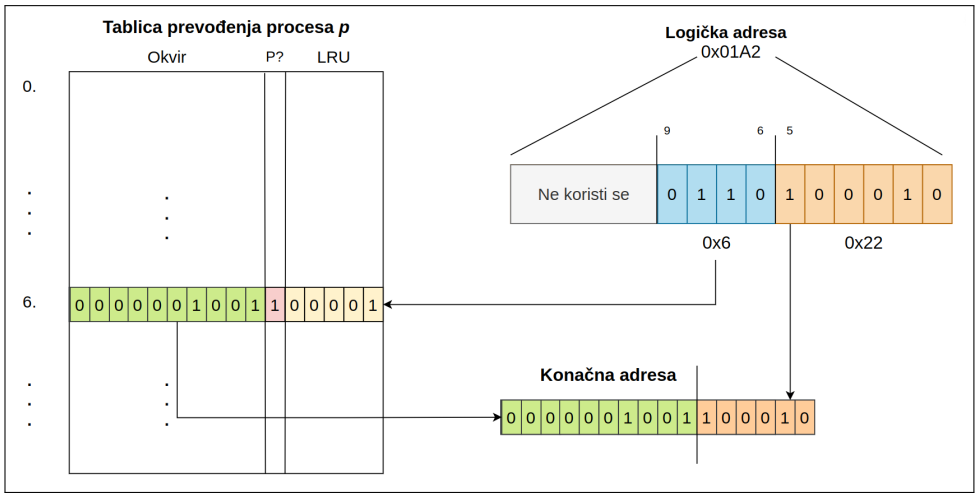
Slika 1: Dijagram sustava straničenja simuliranog računala.



Slika 2: Struktura zapisa u tablici prevođenja.

2.1.2 Primjer jednog koraka simulacije

Pretpostavimo da smo nasumično generirali adresu `0x1A2`, da je ta stranica prisutna te da je vrijednost varijable $\mathbf{t} = 3$. Kako bi pronašli odgovarajući zapis u tablici prevođenja, potrebno je razložiti adresu na polja specificirana na slici 1. Postupak je prikazan na slici 3. Vrijednost dobivenog indeksa je 6 te provjeravamo sedmi zapis u tablici (brojimo od nule). Razlaganjem vrijednosti sedmog zapisa uočavamo da je adresa okvira 0x9, da je bit prisutnosti postavljen na 1 te da je prethodna LRU vrijednost 1. Adresa podatka se potom formira iz adrese okvira i pomaka unutar okvira, a LRU podatak u zapisu tablice postavlja se 3 (trenutna vrijednost varijable $\mathbf{t}$ odnosno sata).



Slika 3: Primjer prevođenja adrese.

2.1.3 Primjer ispisa

Pretpostavimo da simuliramo dva procesa sa jednim dostupnim okvirom te da su u četiri iteracije simulacije oba procesa pristupala adresi `0x1FE`.

Primjer 1: Ispis simulacije.

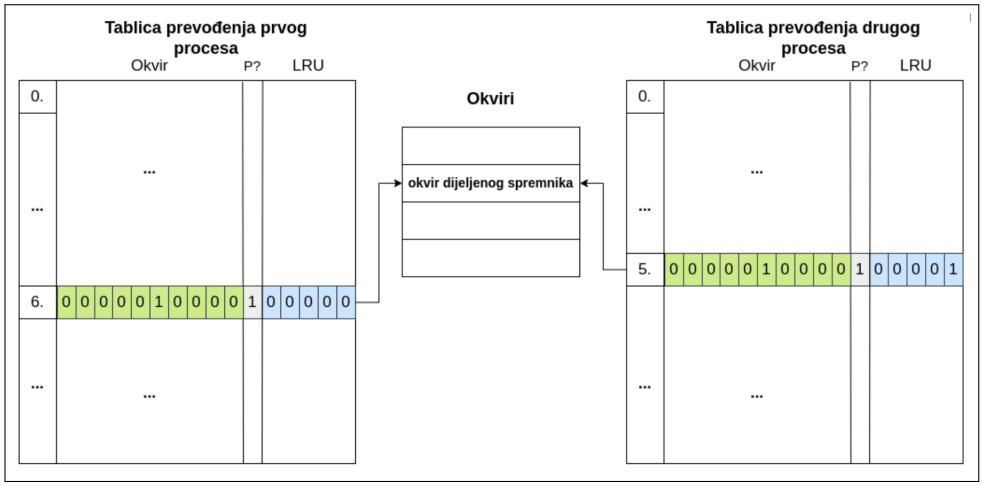
```
$ ./lab4 2 1
---------------------------
proces: 0
        t: 0
        log. adresa: 0x01fe
        Promasaj!
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0020
        sadrzaj adrese: 0

---------------------------
proces: 1
        t: 1
        log. adresa: 0x01fe
        Promasaj!
                Izbacujem stranicu 0x01c0 iz procesa 0
                lru izbacene stranice: 0x0000
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0021
        sadrzaj adrese: 0

---------------------------
proces: 0
        t: 2
        log. adresa: 0x01fe
        Promasaj!
                Izbacujem stranicu 0x01c0 iz procesa 1
                lru izbacene stranice: 0x0001
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0022
        sadrzaj adrese: 1
---------------------------
proces: 1
        t: 3
        log. adresa: 0x01fe
        Promasaj!
                Izbacujem stranicu 0x01c0 iz procesa 0
                lru izbacene stranice: 0x0002
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0023
        sadrzaj adrese: 1
^C
```

2.2 Simulacija dijeljenog spremnika (2 boda)

U prethodnim laboratorijskim vježbama susreli ste se sa mehanizmom dijeljenog spremnika (funkcije `shmget` i `shmat`), a u ovom dijelu vježbe Vaš je zadatak proučiti kako se taj mehanizam može ostvariti pomoću straničenja te proširiti simulaciju tako da simuliranim procesima omogućite međusobnu komunikaciju.

2.2.1 Opis mehanizma dijeljenog spremnika

Sustav straničenja smo dosad koristili tako da svaki proces ima svoj adresni prostor. Međutim, takav pristup ne dozvoljava da procesi međusobno dijele djelove svojeg adresnog prostora, što je potrebno za ostvarenje nekih mehanizama komunikaciju među procesima. Rješenje za taj problem jest mehanizam dijeljenog spremnika. Ostvarenje dijeljenog spremnika pomoću straničenja prikazano je na Slici 4. Dijeljenom spremniku dodijeljen je jedan okvir koji mora biti adekvatno povezan sa svakim proces koji želi koristiti dijeljeni spremnika. Unutar svakog procesa odabire se slobodna logička adresa za adresu spremnika te se u tablici straničenja tog procesa u odgovarajuću vrijednost upiše adresa okvira dijeljenog spremnika. Ovdje je bitno napomenuti da logička adresa spremnika unutar pojedinog procesa ne mora biti ista za svaki proces, može se razlikovati od procesa do procesa.



Slika 4: Sadržaj tablica prevođenja prilikom korištenja dijeljenog spremnika

2.2.2 Zadatak

Prilikom pokretanja svaki proces treba zauzeti dijeljeni spremnik određene veličine, a Vaš simulirani sustav treba ispravno popuniti tablice straničenja kako bi dijeljeni spremnik ispravno radio.

Radi jednostavnosti pretpostavite slijedeće:

- U cijelom sustavu uvijek postoji samo jedan dijeljeni spremnik,
- Dijeljeni spremnik uvijek je veličine jedne stranice, neovisno o tome hoće li cijeli taj prostor biti iskorišten,
- Svaki proces ima istu logičku adresu za dijeljeni spremnik,
- U sustavu postoji odvojeno mjesto na disku za spremanje sadržaja stranice dijeljenog spremnika,
- Prilikom odabira okvira za zamjenu tretirajte okvir zajedničkog spremnika kao najmanje prioritetan, tj. njega uvijek odaberite za zamjenu.

Pomoću simuliranog mehanizma dijeljenog spremnika potrebno je ostvariti komunikaciju proizvođač-potrošač korištenjem dijeljenog reda poruka, pri čemu se prvi pokrenuti proces smatra proizvođačem, a ostali potrošačima. Prošireni pseudokod simulacije prikazan je u Algoritmu 3. Proces proizvođač broj poruka jednak broju procesa te ih stavlja u zajednički red, dok proces potrošač uzima poruke iz zajedničkog reda. Ovaj dio zadatka **treba se odvijati zajedno sa simulacijom opisanom u prvom zadatku**, te je posebnu pozornost potrebno obratiti da procesi nasumičnim pisanjem ne ometaju rad dijeljenog spremnika. Kao i u prvom dijelu zadatka, adresnom prostoru procesa trebate pristupati pomoću funkcija `dohvati_sadržaj` i `zapiši_vrijednost`, što znači da ih trebate koristiti i u dijelu koda koji pristupa zajedničkom spremniku. Red poruka ne treba biti kompliciran te se može sastojati od polja poruka veličine 2 okteta, pri čemu svaki proces potrošač koristi svoj identifikator za pristup redu poruka.

Posebnu pažnju obratite na slijedeće slučajeve:

- Prilikom izbacivanja stranice iz određenog procesa potrebno je poništiti samo jednu vrijednost unutar jedne tablice straničenja. Razmislite koje sve vrijednosti tablica straničenja morate poništiti kada izbacujete okvir dijeljenog spremnika.

Algoritam 3: Prošireni pseudokod simulacije.

```
1  za i = 1 do N čini
2  |  stvori proces i;
3  |  inicijaliziraj tablicu straničenja procesa i;
4  |  zauzmi zajednički spremnik;
5  kraj
6  t ← 0;
7  ponavljaj
8  |  za svaki proces p čini
9  |  |  ako p je proizvođač onda
10 |  |  |  stavi N − 1 poruka u zajednički spremnik;
11 |  |  inače
12 |  |  |  uzmi poruku iz zajedničkog spremnika;
13 |  |  kraj
14 |  |  x ← nasumična logička adresa;
15 |  |  i ← dohvati_sadržaj(p, x);
16 |  |  i ← i + 1;
17 |  |  zapiši_sadržaj(p, x, i);
18 |  |  t ← t + 1;
19 |  |  spavaj;
20 |  kraj
```

2.2.3 Primjer ispisa

Pretpostavimo da simuliramo dva procesa sa jednim dostupnim okvirom te da su u četiri iteracije simulacije oba procesa pristupala adresi `0x1FE`.

Primjer 2: Ispis proširene simulacije.

```
--------------------------
proces: 0
        t: 0
        Poslao poruku: 4567
        Promasaj!
                Izbacujem okvir dijeljenog spremnika
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0020
        sadrzaj adrese: 0
--------------------------
proces: 1
        t: 1
        Promasaj!
                Izbacujem stranicu 0x01c0 iz procesa 0
                lru izbacene stranice: 0x0000
                dodijeljen okvir 0x0000
        Primio poruku: 4567
        Promasaj!
                Izbacujem okvir dijeljenog spremnika
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0021
        sadrzaj adrese: 0
--------------------------
proces: 0
        t: 2
        Promasaj!
                Izbacujem stranicu 0x01c0 iz procesa 1
                lru izbacene stranice: 0x0001
                dodijeljen okvir 0x0000
        Poslao poruku: 23c6
        Promasaj!
                Izbacujem okvir dijeljenog spremnika
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0022
        sadrzaj adrese: 1
--------------------------
proces: 1
        t: 3
        Promasaj!
                Izbacujem stranicu 0x01c0 iz procesa 0
                lru izbacene stranice: 0x0002
                dodijeljen okvir 0x0000
        Primio poruku: 23c6
        Promasaj!
                Izbacujem okvir dijeljenog spremnika
                dodijeljen okvir 0x0000
        fiz. adresa: 0x003e
        zapis tablice: 0x0023
        sadrzaj adrese: 1
^C
```