

4. Veza prema programskoj podršci

- 4.1. Instrukcije za poziv potprograma
- 4.2. Načini izvedbe stoga
- 4.3. Obrada sklopovskih iznimki
- 4.4. Prevođenje i povezivanje izvornog koda
- 4.5. Pokretanje izvršnih programa
- 4.6. Specifičnosti jezika s dinamičkim prevođenjem

Poziv potprograma:

- mnemonici: CALL, JSR, MOVE
- memorijski operand određuje adresu prve instrukcije potprograma
- ciljna adresa se zadaje ili izravno, ili relativno s obzirom na PC

Povratak iz potprograma:

- mnemonici: RET, RTS, MOVE
- izvođenje se nastavlja nakon odgovarajuće instrukcije CALL

Faza IZVRŠI instrukcije CALL:

1. $PC \rightarrow S$,

2. $X \rightarrow PC$

Lokacija **S** sadrži **povratnu adresu**

Faza IZVRŠI instrukcije RET:

1. $S \rightarrow PC$

Gdje locirati **S**?

Rješenje 1: **S** je poseban registar u upravljačkoj jedinici

problem: gniježđenje potprograma!

Rješenje 2: **S** je prva lokacija potprograma

instrukcija JMS X (Jump to Subroutine, PDP-8):

- PC se pohranjuje na lokaciju X (!) i automatski inkrementira
- Povratak ostvarujemo indirektnim skokom JMP I X

problem: rekurzija!

Rješenje 3: **S** se nalazi u posebnom dijelu radne memorije

taj dio radne memorije nazivamo upravljačkim stogom

Moderna računala kombiniraju rješenja 1. i 3.!

Upravljački stog (call stack):

- dinamička podatkovna struktura, pristup na principu LIFO
- svaki tok izvođenja (dretva) ima dedikirani stog u memoriji
- stogove koristimo za pohranu: i) povratnih adresa, ii) parametara potprograma iii) lokalnih varijabli, ...
- omogućava rekurzivne pozive!
- temeljne instrukcije koje koriste upravljački stog:

CALL SUB:

PUSH PC

PC ← SUB

RET:

POP PC

Primjer rekurzivnog pozivanja:

Begin

.

.

CALL SUB

X: . ; *X* je povratna adresa

.

.

SUB: .

.

CALL SUB

Y: . ; *Y* – povratna adresa

.

.

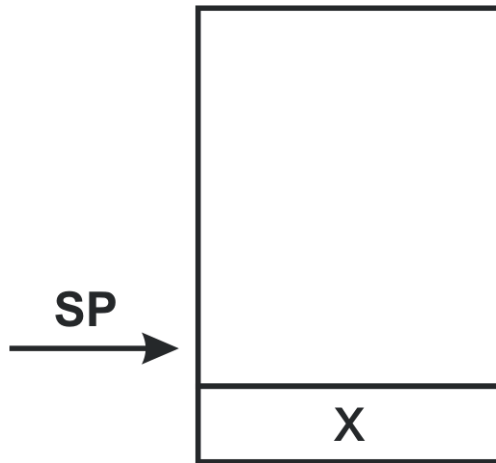
RETURN

.

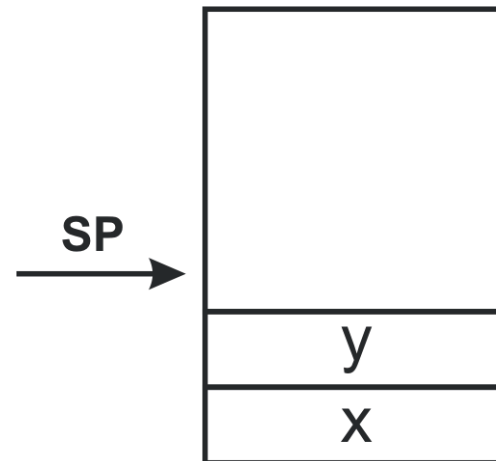
End

Stanje stoga:

Nakon prvog pozivanja



Nakon drugog pozivanja



Treće pozivanje, četvrto pozivanje, ...

Povratak???

Potpuni program:

Begin

.

$N := 2$

CALL SUB

X: .

.

SUB: .

.

$N := N - 1$

IF (N >= 0) THEN CALL SUB

Y: .

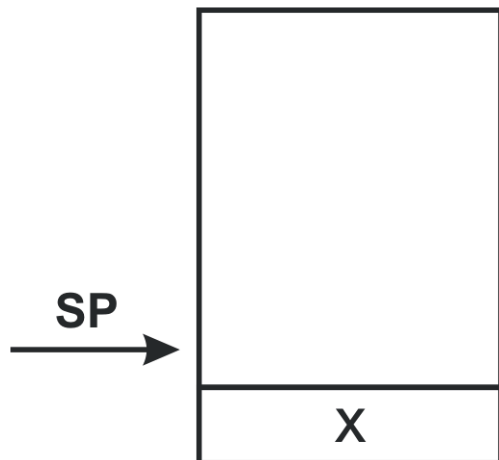
.

RETURN

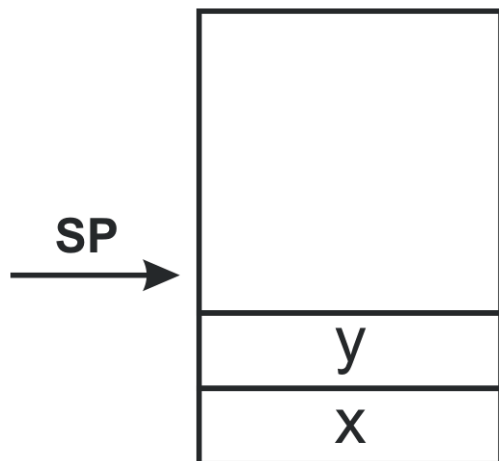
End

Stanje stoga:

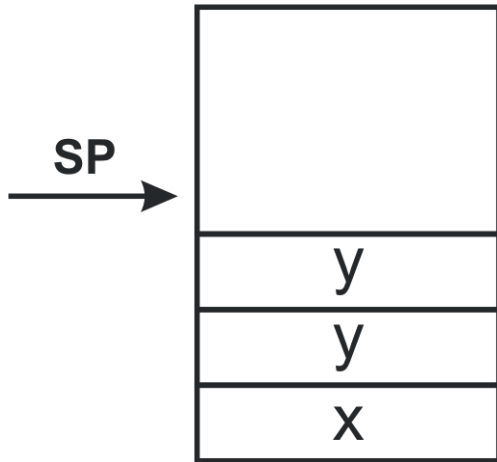
- Prvi poziv ($N = 2$) / poziv iz glavnog programa/:



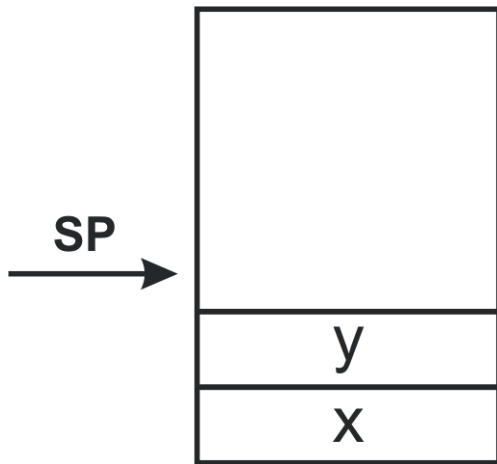
- Drugi poziv ($N = 1$) / poziv iz SUB/:



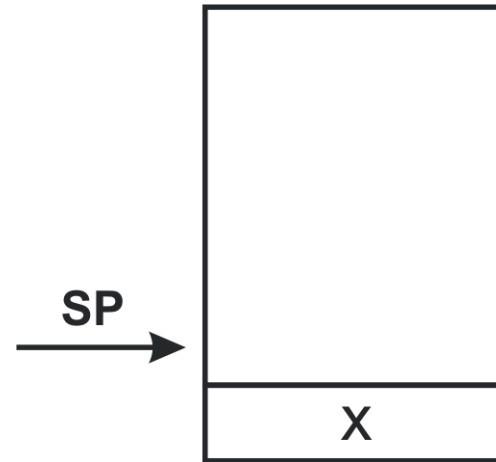
Treći poziv ($N = 0$) /poziv iz SUB/:



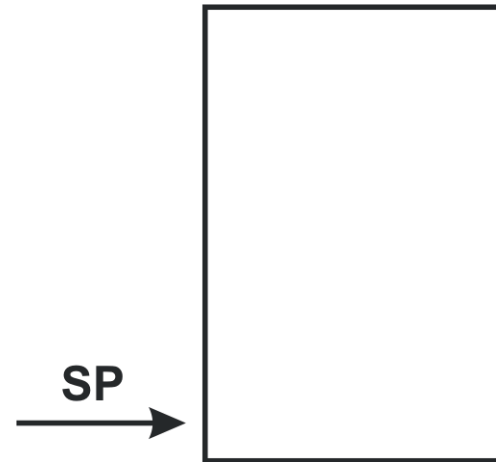
Prvi povratak:



Drugi povratak:



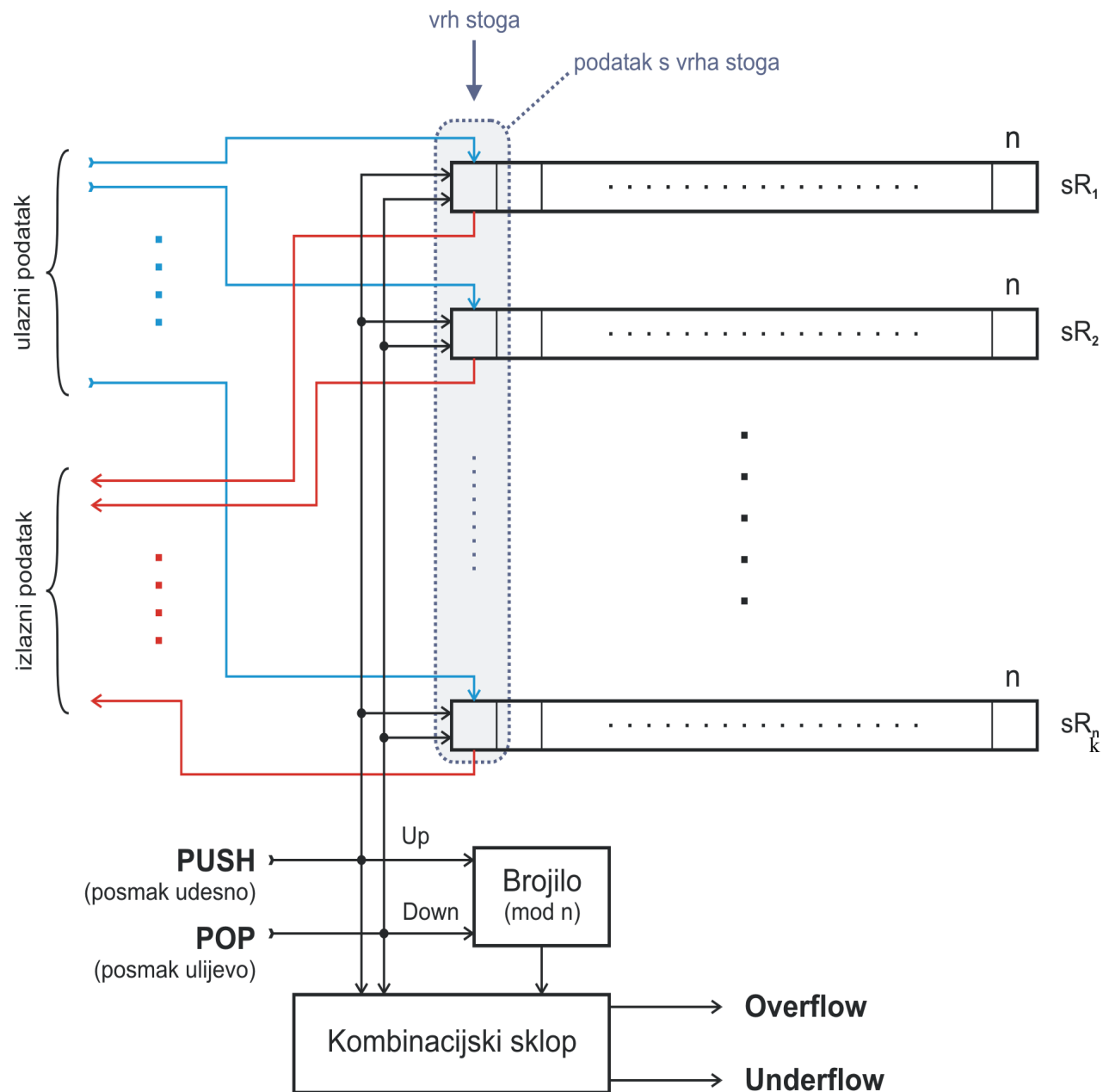
Treći povratak:



Primjeri izvedbe stoga:

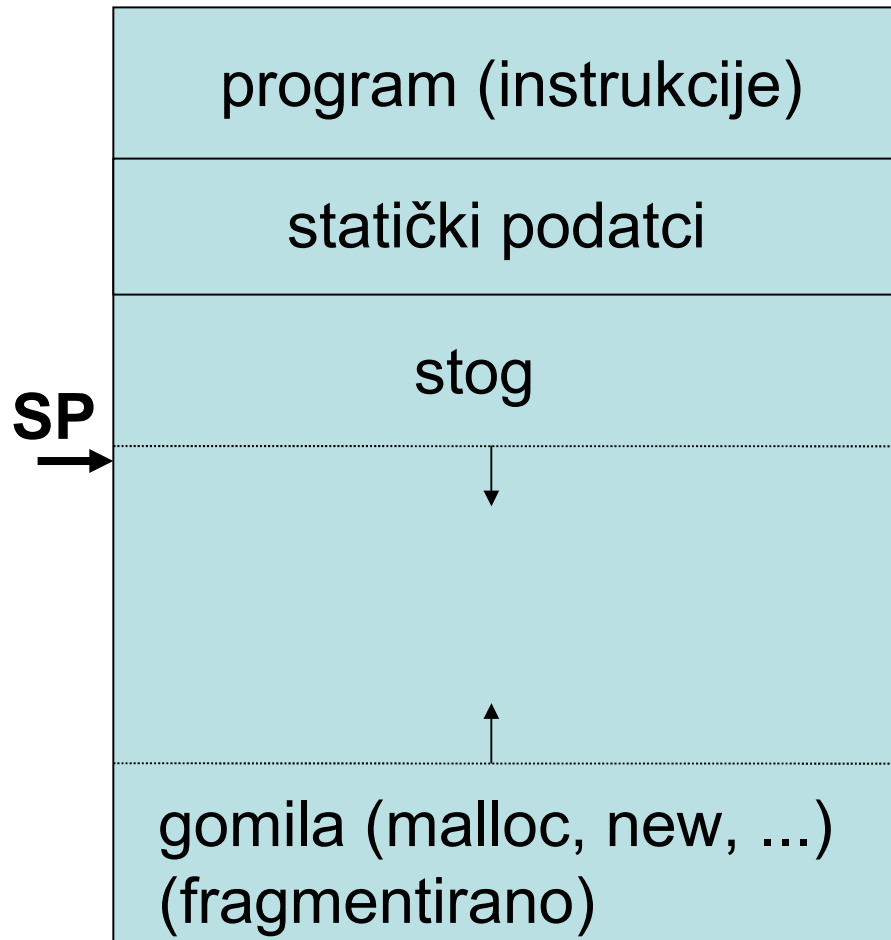
- 1) sklopovska izvedba temeljena na posmačnim registrima
- 2) registarska okna (Berkeley RISC, Sun Sparc)
- 3) rezervirani dio memorije s izravnim pristupom
 - obično se koristi gornji dio podatkovnog segmenta procesa
 - stog obično raste prema dolje
 - dvije mogućnosti:
 - 1) implicitno korištenje namjenskim instrukcijama (call-return, jsr-rts)
 - 2) eksplicitno pohranjivanje link registra (Sparc, MIPS, PowerPC, Arm)

Sklopovska izvedba
stoga dubine n i
duljine riječi k bita:



Izvedba stoga u radnoj memoriji

tipični raspored korištenja
memorije procesa:



izvorni kod (procedure.c):

```
int proc(int i, int j){
    int k, l;
    ...
}
int main(){
    proc(4,5);
}
```

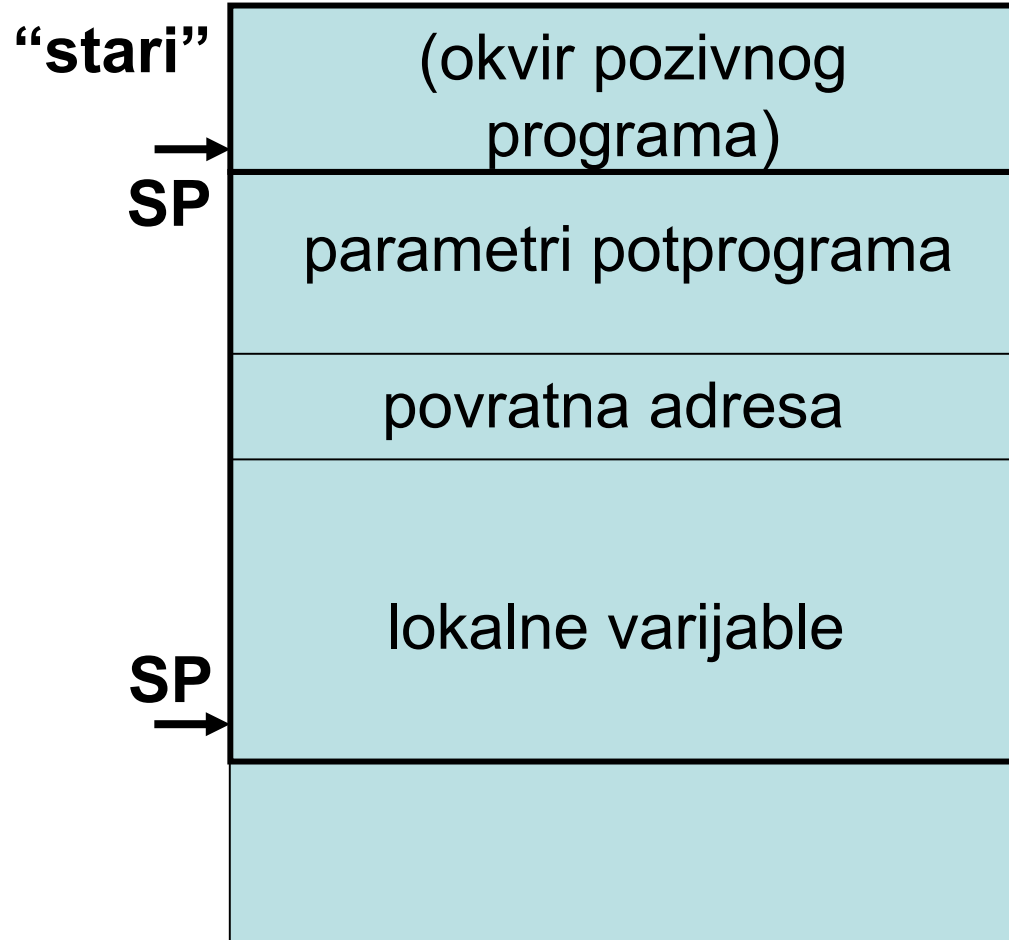
Zbirni kod poziva (x86, procedure.s):

```
...
sub    esp, 8
mov    DWORD PTR [esp+4], 5
mov    DWORD PTR [esp], 4
call   proc
add    esp, 8
....
```

Zbirni kod potprograma:

```
sub    esp, 8
...
add    esp, 8
ret
```

tipični sadržaj stogovnog okvira
(stack frame) potprograma:



gcc -S -fomit-frame-pointer
-mpreferred-stack-boundary=2
-masm=intel procedure.c

Primjena stoga pri obradi iznimki

Što je iznimka (engl. exception)?

- posebne okolnosti kojima se narušava normalno stanje procesora i izvođenje programa
- nasilan prekid normalnog izvođenja programa i prijenos upravljanja bez upotrebe posebnih instrukcija
- pazi: iznimke u programskim jezicima su nešto sasvim drugo!

Iznimke mogu biti dvojake:

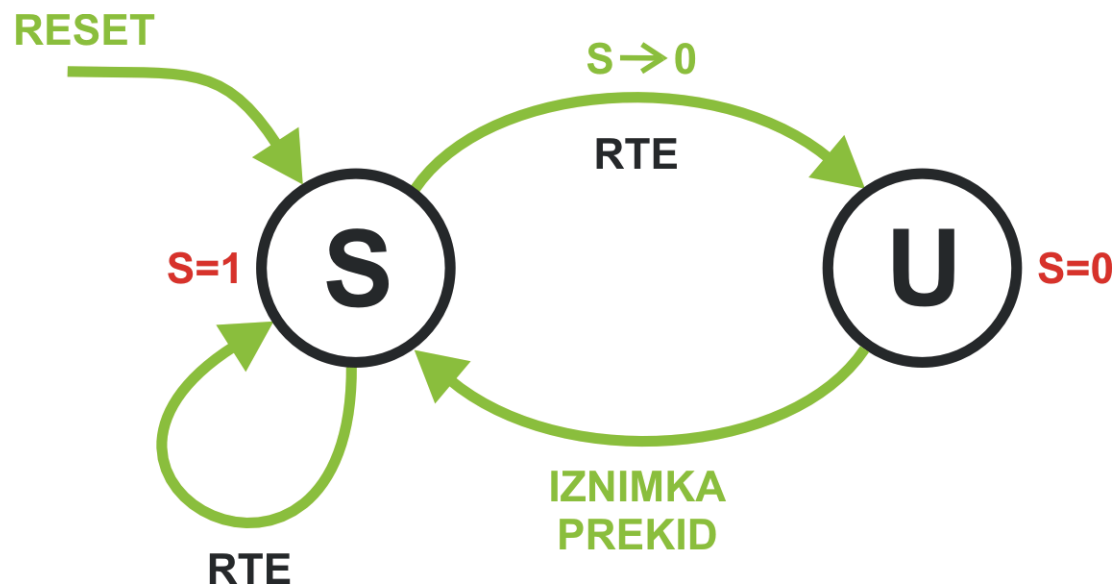
1. vanjske iznimke (prekid, sabirnička pogreška, reset)
2. unutarnje iznimke (dijeljenje nulom, uvjetne i bezuvjetne zamke (engl. trap), povreda privilegiranosti,...)

Obrada iznimke za MC 68000

Tipični koraci:

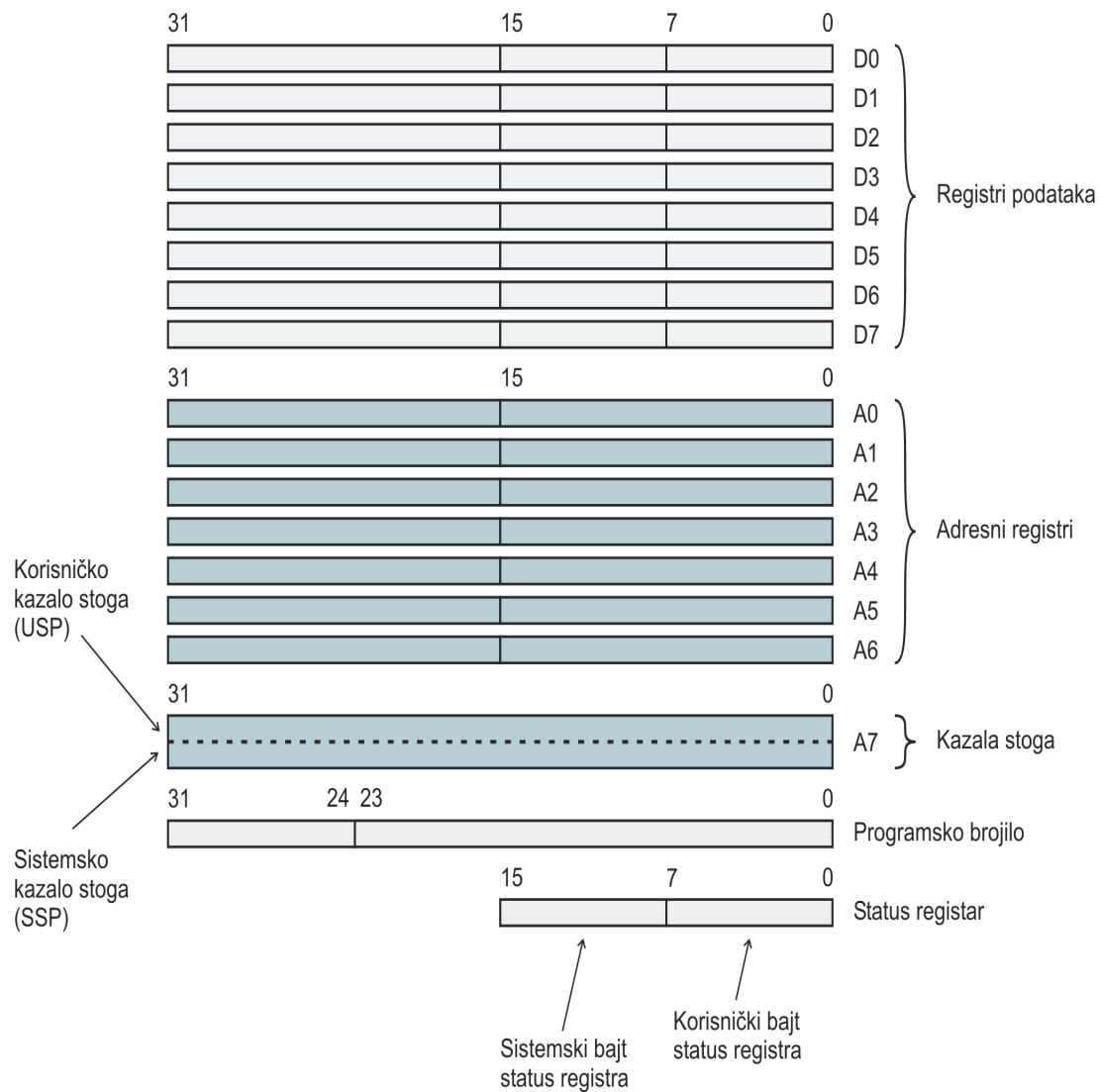
1. korak: 16-bitni statusni registar SR interno se pohranjuje u interni (privremeni) registar. U statusnom registru postavljaju se zastavice u stanje koje odgovara obradi iznimke:
$$S \rightarrow 1 \quad i \quad T \rightarrow 0;$$
2. korak: utvrđuje se vektorski broj iznimke, odnosno vektor iznimke. Vektorski se broj iznimke, prema vrsti iznimke, dobiva od vanjskog uređaja ili ga procesor generira interno;
3. korak: pohranjuje se sadržaj programskog brojila PC i sadržaj interno pohranjenog statusnog registra SR;
4. korak: na temelju vektorskog broja iznimke i tablice vektora iznimaka određuje se novi sadržaj programskog brojila PC i procesor nastavlja rad izvođenjem prve instrukcije iznimke.

Dijagram stanja za MC 68000



RTE -Return from Exception /povlaštena instrukcija/

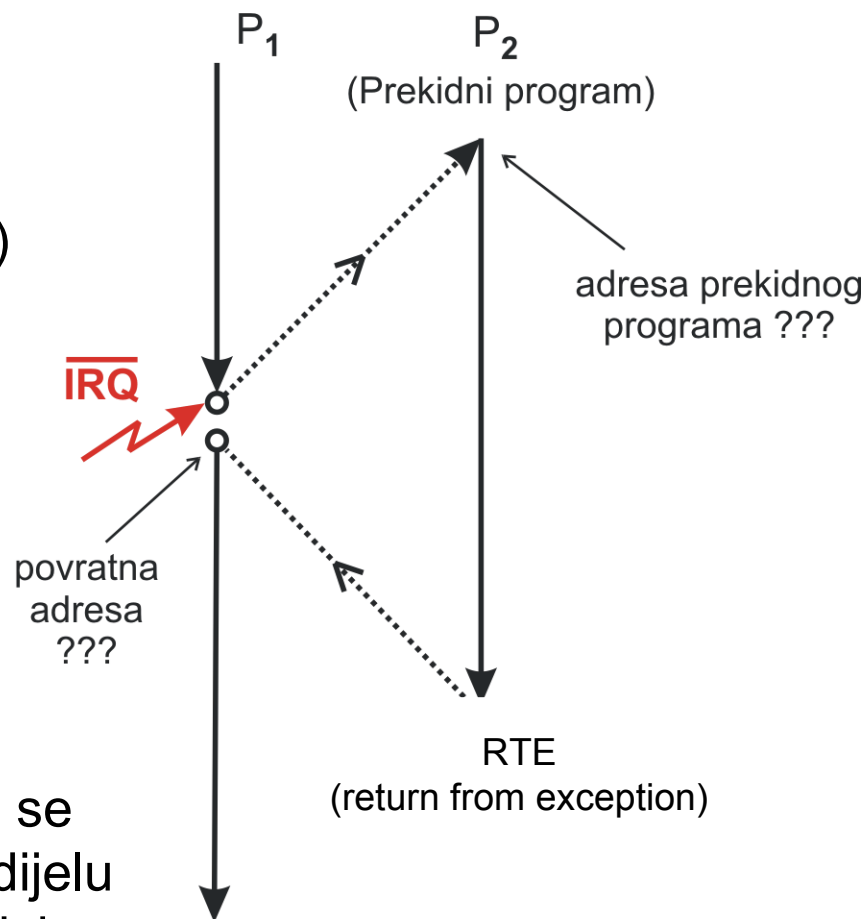
Programski model MC 68000



Na stog se tijekom prijenosa upravljanja s prekinutog na prekidni program ($P_1 \rightarrow P_2$) **pohranjuje**

MINIMALNI KONTEKST:

- Sadržaj programskog brojila (4 bajta)
- Sadržaj statusnog registra (2 bajta)




Adresa prekidnog programa dobiva se prozivanjem tablice u dedikiranom dijelu radne memorije pri čemu indeks ovisi o vrsti iznimke!

Primjer uporabe stoga

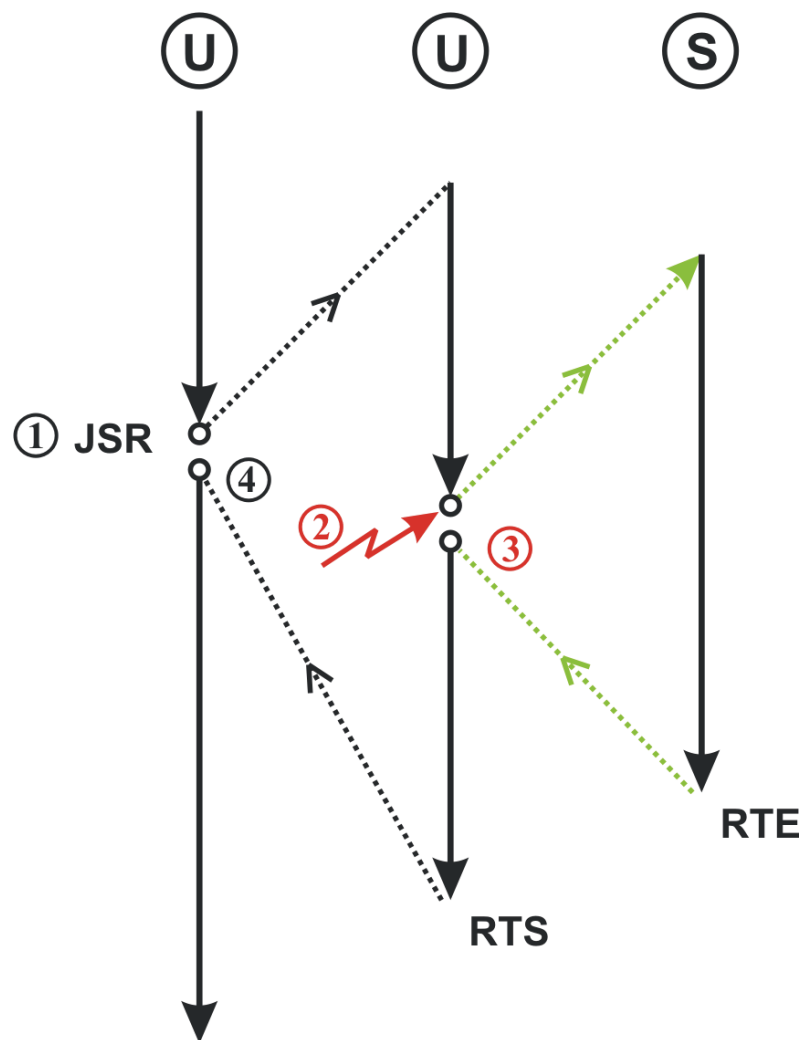
Analiza slučaja: MC 68000

Scenarij:

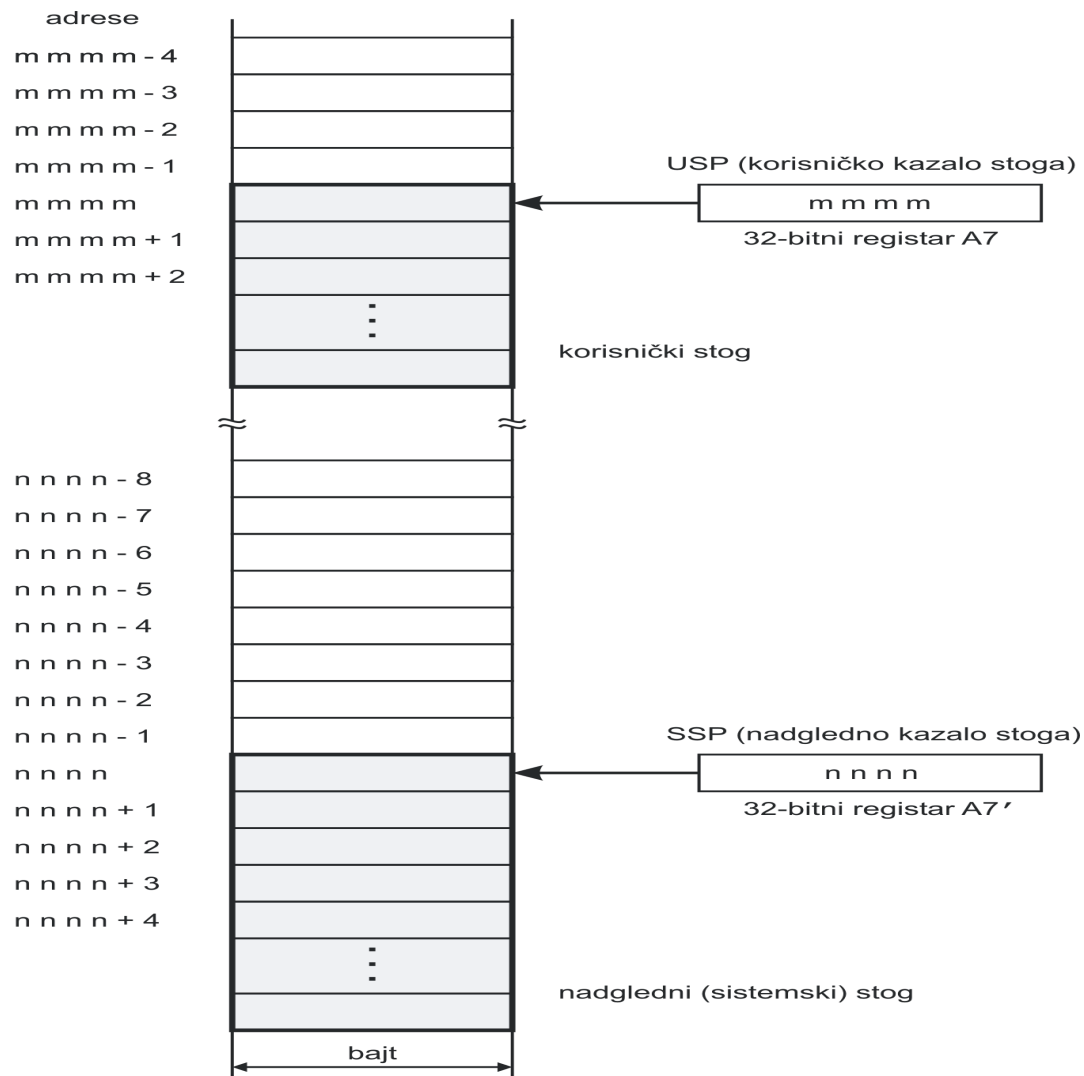
- ① Procesor je u korisničkom načinu rada (User Mode)
 - Poziva se potprogram
 - Nastavlja se izvođenje potprograma
- ②  Dogodila se iznimka (PREKID)
 - Obrada prekida
- ③ Vraćanje u potprogram
- ④ Vraćanje iz potprograma

literatura: S. Ribarić, Naprednije arhitekture mikroprocesora

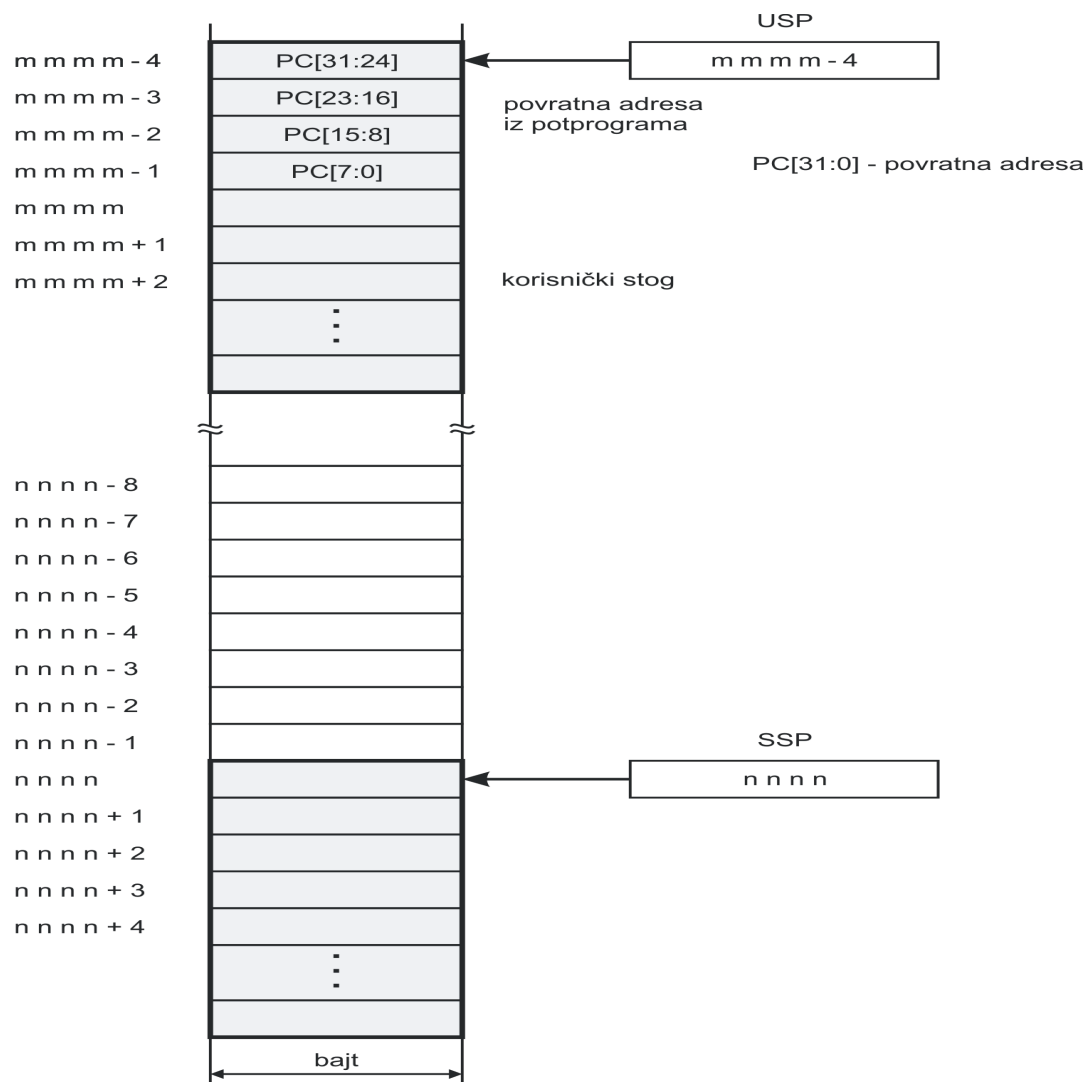
Grafički prikaz scenarija:



Stanja kazala stoga
i stogova **prije**
pozivanja potprograma

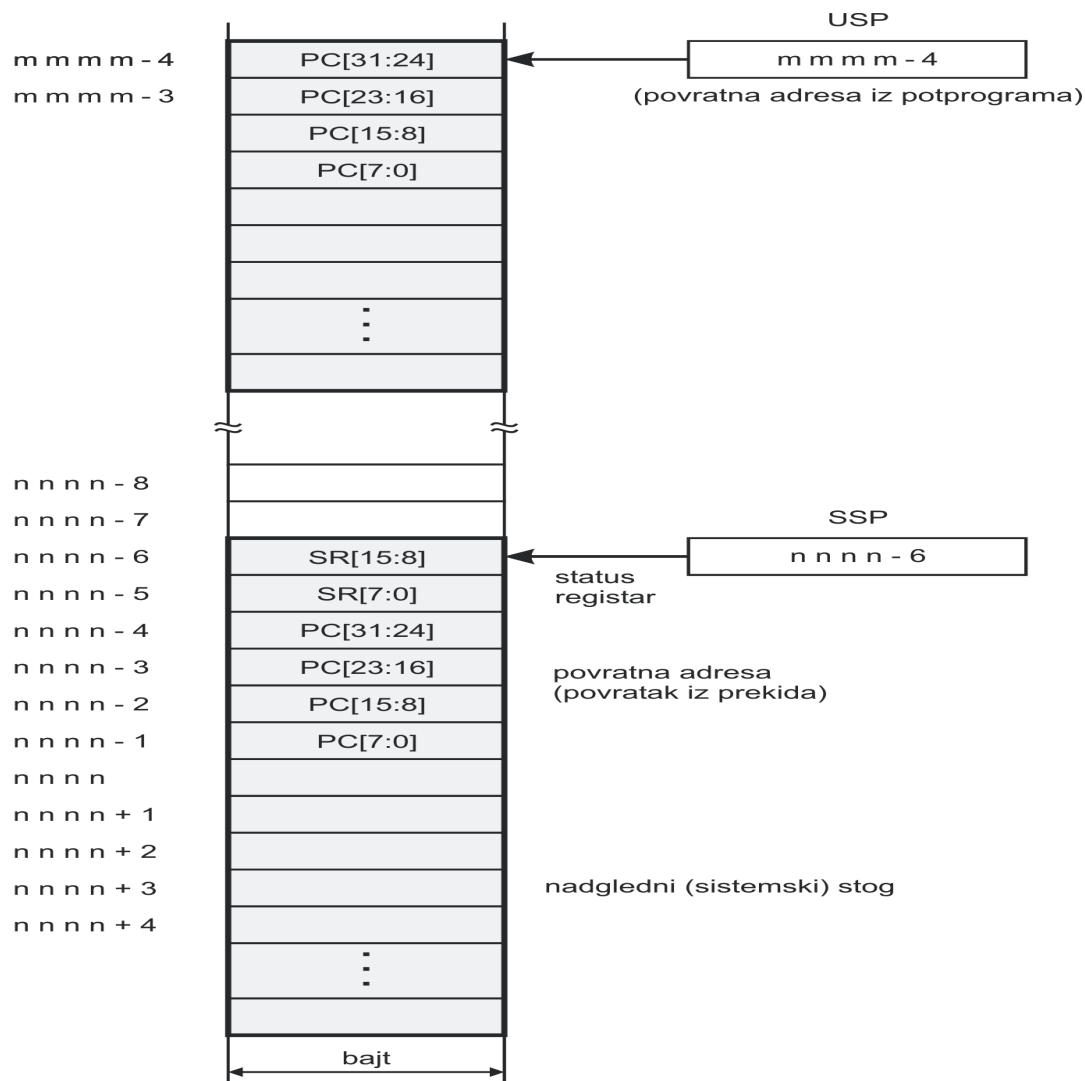


Stanje neposredno
nakon
grananja u potprogram

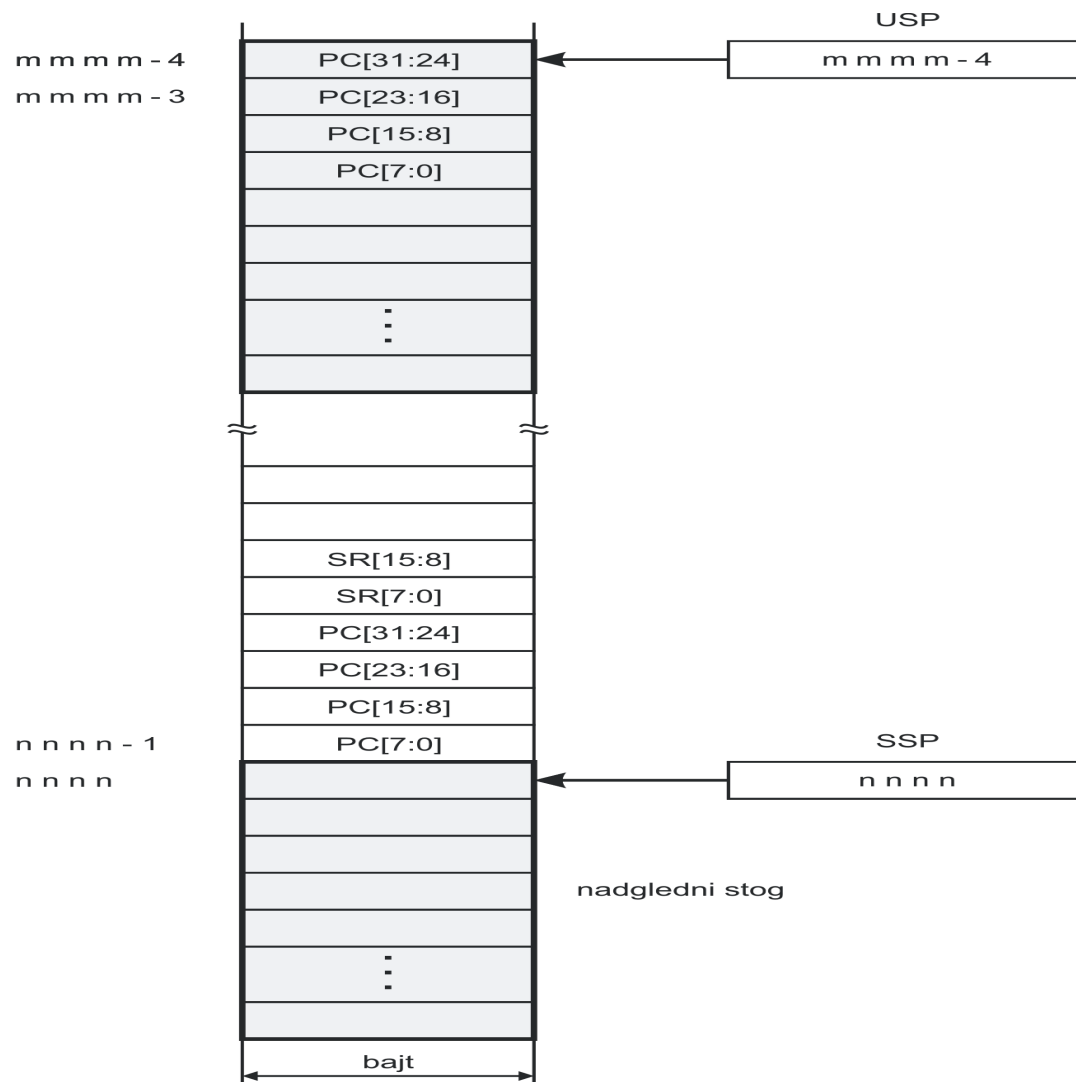


Dogodio se **prekid!**

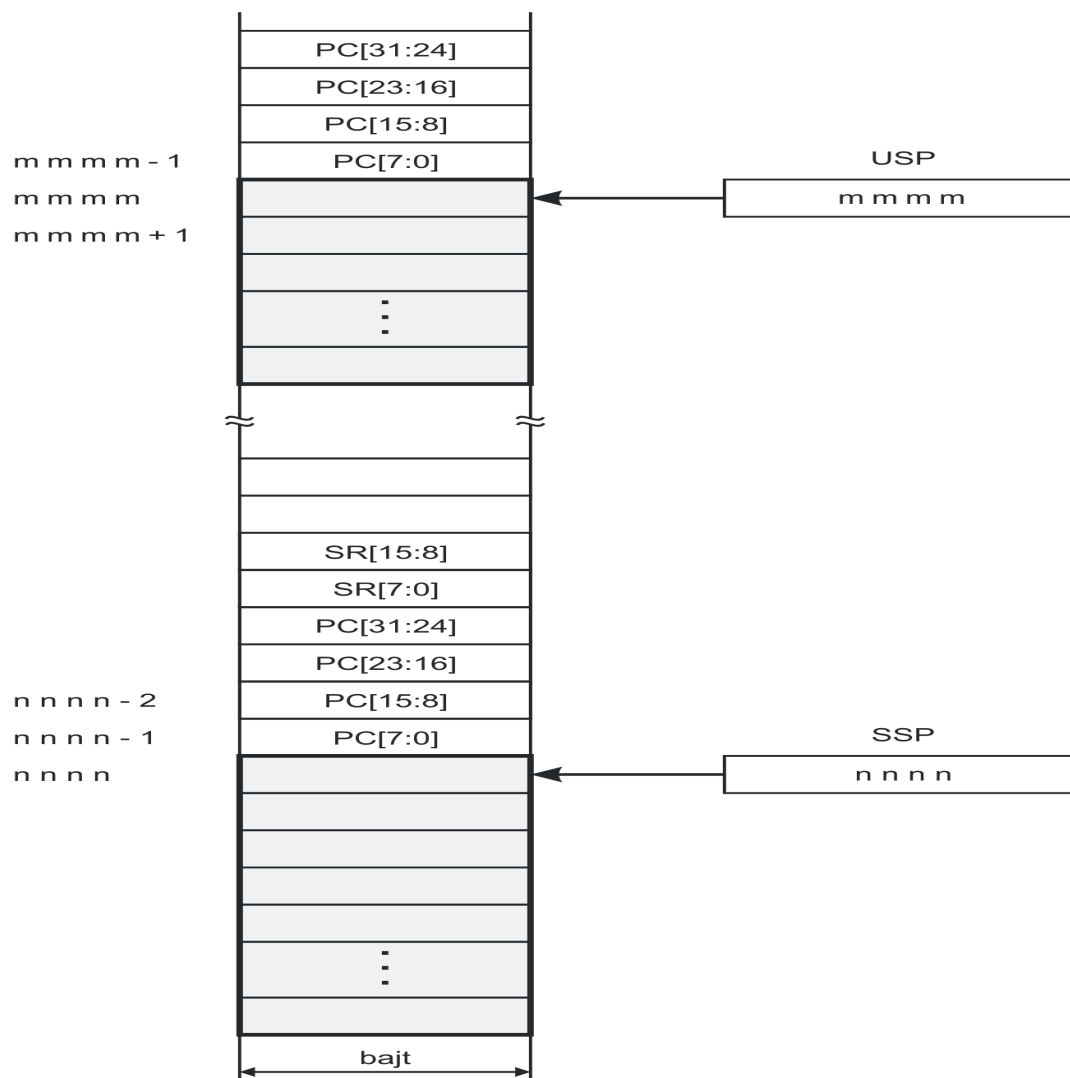
(Iznimka se
obrađuje u
nadglednom načinu)

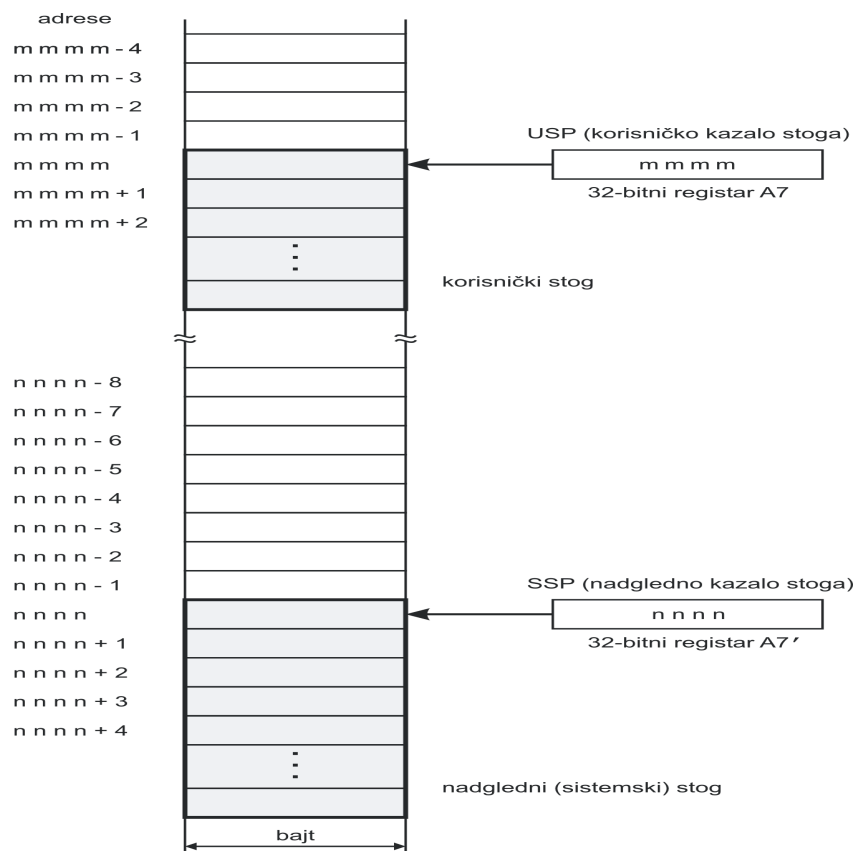


Stanje stogova **nakon**
vraćanja u
potprogram

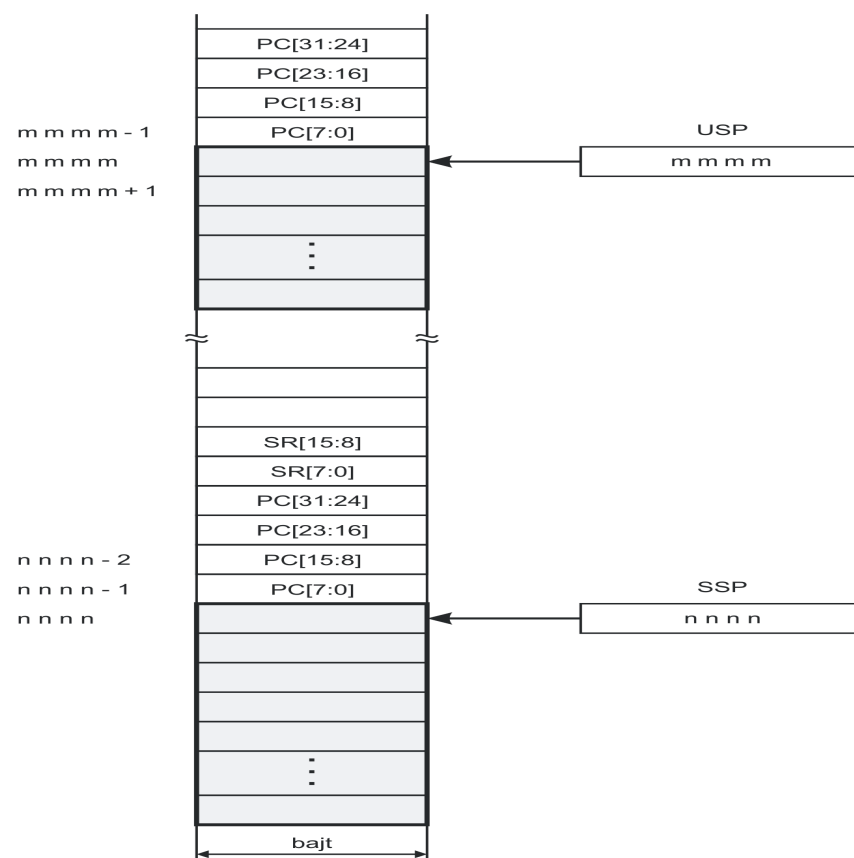


Stanje stoga **nakon**
vraćanja iz
potprograma





Stanje **prije** izvođenja programa



Stanje **nakon** izvođenja programa