

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

Ime:	Aleksandar	Broj indeksa:	19095
Prezime:	Gospavić		
LV po redu:	II	Termin:	2
Datum i vreme početka izrade	14.11.2023. 18:15		

Zadatak:

Izračunati, bez transformacije, i uz optimizaciju deljenja stepenom 2, vrednost izraza

$$(B1+1)*([B1*]-1) / (B1-B2^{2/4}),$$

pri čemu su: B1 32b, B2 32b, oba označeni podaci, a operator [X*] znači:

zadržani bitovi u nižim polubajtovima svih bajtova podatka X, a ostali bitovi postavljeni na 1.

a)

Da li ovaj problem može da se reši u zadatim okvirima? Ukoliko ne može, pod kojim uslovima bi mogao da se reši? Rešiti zadati problem u okvirima u kojima može da se reši.

Formirati primere početnih vrednosti koji demonstriraju sve osobine zadatog problema, posebno u pogledu međuprenosa, izlaznog prenosa, ostatka pri deljenju, vrednosti nekog od međurezultata jednakoj nuli, i za eventualne slučajeve kada nije moguće doći do tačnog rezultata.

Za svaki primer početnih vrednosti:

- navesti po čemu je karakterističan - koju osobinu demonstrira
- izračunati rešenje i pokazati kako se dolazi do tog rešenja po koracima nacrtane šeme postupka.

Za svaku operaciju iz izraza datog u problemu, nacrtati šemu izvođenja te operacije kada bi se izvodio instrukcijama x86-32 arhitekture procesora. Na šemi treba da se vide težine pojedinih delova operanada, redosled redosled izračunavanja, operandi i odredišta svakog međurezultata i konačnog rezultata. Šeme prikazati u redosledu u kome bi operacije trebalo izvoditi u asemblerskom programu.

b)

Napisati kod na asemblerskom jeziku za sprovođenje izračunavanja po šemi postupka iz a). Na šemi postupka označiti registre koji su korišćeni u kodu. Uneti napisani kod u emulator na adresi <https://carlosrafaelgn.com.br/Asm86/>; primere početnih vrednosti uneti kao kompletne instrukcije upisivanja vrednosti u promenljive, pri čemu su svi kompleti, osim jednog, podešeni kao komentar.

Izvršiti napisani kod u emulatoru za sve primere početnih vrednosti i ustanoviti da li program radi kako je očekivano.

U izveštaju napisati kratak tekst o tome šta program treba da radi, da li se izvršava ili postoji greška (i gde je greška - priložiti snimak ekrana!) i da li se rezultati poklapaju sa očekivanim rezultatima iz a) za sve primere.

Ukoliko se za neki primer rezultati ne poklapaju sa očekivanim, ustanoviti na kom mestu u kodu dolazi do odstupanja.

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

Rešenje:

a) Primeri vrednosti, šeme operacija

Problem nije moguće rešiti u potpunosti u zadatim okvirima. Podelimo izraz na dva dela. Prvi deo je $(B1+1)*([B1*]-1)$, a drugi $(B1-B2^{2/4})$. Prvi izraz će se smestiti u 64b registru. Međutim, ako rezultat drugog izraza bude isto 64b broj, neće biti moguće podeliti ova dva broja. Zato, se ograničavamo na to da je rezultat izraza $(B1-B2^{2/4})$ 32b broj. Tada je deljenje moguće.

Drugo ograničenje je za vrednost **B1**. Najveća vrednost koju **B1** može da ima je **0x7FFFFFFE**. Zato što za najveći označeni broj u 32b registru – **0x7FFFFFFF**, izraz $(B1+1)$ prelazi opseg označenih brojeva i program se ne ponaša očekivano. $(0x7FFFFFFF + 1) = (0x80000000)$ je u stvari negativan broj.

Još jedno ograničenje koje uvodimo je kod izraza $(B1-B2^{2/4})$. Ako je ovaj izraz jednak 0, deljenje prvog dela izraza drugim neće biti moguće jer će se javiti deljenje nulom. Znači, za **B1** i **B2** ne sme da važi $B2 = 2 * \text{sqrt}(B1)$.

Primeri početnih vrednosti:

- 1) Ulazne vrednosti koje dovode do toga da se broj **B2** nakon kvadriranja nalazi u registru **EDX:EAX**, pri čemu taj broj nije deljiv brojem 4, pa se ostatak pri računanju zanemaruje.

B1 = 0x61CA2456

B2 = 0x12F43

Prvo inkrementiramo broj **B1**, tj registar EAX u koji je inicijalno smešten broj. To daje vrednost **0x0x61CA2457**. Taj rezultat se pomera u registar **ECX**.

Sledeći korak je izračunavanje izraza $([B1*]-1)$. Kako bi se izračunalo neophodno je izvršiti operaciju **OR** između odgovarajućih bitova maskom koja će bitove u nižim polubajtovima svih bajtova zadržati, a ostale postaviti na 1. Na primeru 8 bita $b_7b_6b_5b_4b_3b_2b_1b_0$ primećujemo da kako bi se ovaj efekat postigao potrebno je primeniti masku **11110000**, što odgovara heksadekadnom broju **0xF0**. Za postizanje ovog efekta na 32b broju potrebno je primeniti masku **0xF0F0F0F0**. U trenutnom primeru to izgleda ovako:

```
0110 0001 1100 1010 0010 0100 0101 0110
OR 1111 0000 1111 0000 1111 0000 1111 0000
-----
1111 0001 1111 1010 1111 0100 1111 0110
```

što je jednako **0xF1FAF4F6**, što nakon dekrementiranja postaje **0xF1FAF4F5**. Kako je prva cifra F, broj se sada tretira kao negativni.

Ovaj rezultat se nalazi u EAX i pomeramo ga u EBX. Međurezultat iz prethodnog koraka vraćamo u EAX i množimo registre EAX i EBX. Rezultat množenja

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

0x0x61CA2457 * 0xF1FAF4F5 = 0xFAA504D8A50DB343 se nalazi u 64b registru EDX:EAX. Ovaj međurezultat smeštamo u registar ESI:ECX.

Sledeći korak je računanje **B2^2** i međurezultat se nalazi u registrima **EDX:EAX**

$$B2^2 = 0x12F43 * 0x12F43 = 0x1673FAB89$$

Registar EDX nakon kvadriranja: (0000 0000 0000 0000 0000 0000 0000 0001)₂

Registar EAX nakon kvadriranja: (0110 0111 0011 1111 1010 1011 1000 1001)₂

Deljenje brojem 4 podrazumeva pomeranje registra 2 mesta udesno. Kako se broj B2^2 nalazi u dva registra, neophodno je pamtiti šta se dešava sa bitovima registra koji pamti cifre iz viših bitova. Zbog toga je dvostruko pomeranje udesno neophodno izvršiti jednostrukim pomeranjem dva puta, između kojih se izlazna cifra iz registra EDX umeće na mesto cifre najveće težine registra EAX.

Registar EDX nakon deljenja sa 2, tj nakon pomeranja sadržaja udesno za 1 mesto:

$$EDX = (0000 0000 0000 0000 0000 0000 0000 0000)_2$$

Cifra najmanje težine datog registra (1) sačuvana u carry flagu, iz koga se prebacuje u registar EBX, čije je stanje sada (0000 0000 0000 0000 0000 0000 0000 0001)₂, a nakon pomeranja sadržaja za 31 mesto ulevo:

$$(1000 0000 0000 0000 0000 0000 0000 0000)_2$$

Registar EAX nakon deljenja sa 2, tj. Nakon pomeranja sadržaja udesno za 1 mesto:

$$(0011 0011 1001 1111 1101 0101 1100 0100)_2$$

Da bi se cifra koja je izbačena iz registra EDX, a sada se nalazi na mestu cifre najveće težine u registru ECX, umetnula na mesto cifre najveće težine registra EAX, potrebno je izvršiti OR između odgovarajućih bitova registara EAX i EBX.

Stanje registra EAX je sada:

$$(1011 0011 1001 1111 1101 0101 1100 0100)_2$$

što odgovara heksadekadnom broju 0xB39FD5C4.

Opisani postupak se ponovi još jednom, nakon čega je stanje registra EAX:

$$(0101 1001 1100 1111 1110 1010 1110 0010)_2$$

što odgovara broju 0x59CFEAE2.

Kako je **0x59CFEAE2 * 4 = 0x1673FAB88**, zaključujemo da je ignorisani ostatak 1.

Dalje se od broja B1 oduzima izračunati broj i dobija se **0x7FA3974**, što predstavlja vrednost izraza **(B1 - B2^2/4)**.

Kako je rezultat levog dela izraza u ESI:ECX premeštamo ga u EDX:EAX kako bi

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

izvršili deljenje. Rezultat deljenja se nalazi u EAX:

0xFAA504D8A50DB343 / 0x7FA3974 = 0x542488B3,

a ostatak u EBX: **0xFE41E727**

- 2) Ulazne vrednosti kod kojih je jedno međurešenje neispravno, pri čemu je i konačno rešenje netačno.

B1 = 0x7FFFFFFF

B2 = 0x2

Izraz $(B1+1)$ daje rezultat 0x80000000 što prelazi opseg pozitivnih označenih brojeva. Emulator tretira ovaj broj kao negativan i za rezultat izraza $(B1+1)*([B1*]-1)$ daje 0x0000000100000000 dok je pravi rezultat: 0x(7FFF FFFF 0000 0000). Aktivira se Carry Flag i Overflow Flag. Ovaj netačan rezultat povlači sve ostalo pa ni krajnji rezultat nije dobar.

- 3) Ulazne vrednosti kada se javlja deljenje 0.

B1 = 0x1900

B2 = 0xA0

$B1+1 = 0x1901$

$[B1*]-1 = 0xF0F0F9EF$

$(B1+1)*([B1*]-1) = 0xFFFFFE87795950EF$

$B2^{2/4} = 0x1900$

$B1 - (B2^{2/4}) = 0x0$

Konačni izraz je 0xFFFFFE87795950EF / 0 što nije moguće.

- 4) Ulazna vrednosti kada je rezultat izraza $(B1 - (B2^{2/4}))$ 64b broj i nije moguće podeliti dva 64b broja.

B1 = 0xA12F1

B2 = 0x7FFF12

Postupak prvog dela zadatka je isti kao u prethodnim primerima.

$(B1+1)*([B1*]-1) = 0xFFFF68B0EFAE86E0$

$B1 - (B2^{2/4}) = 0xFFFF0003B89DBA0$

Ovaj drugi izraz je 64b i ne može se podeliti prvim izrazom. Emulator prikazuje rešenje: 0xFFFD756A dok je pravo rešenje 9.

- 5) Primer prosečnih vrednosti.

B1 = 0xE723C3

B2 = 0x3C4

$(B1+1)*([B1*]-1) = 0xFFFF26DA3838EDB48$

$(B2^{2/4}) = 0x38B84$

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

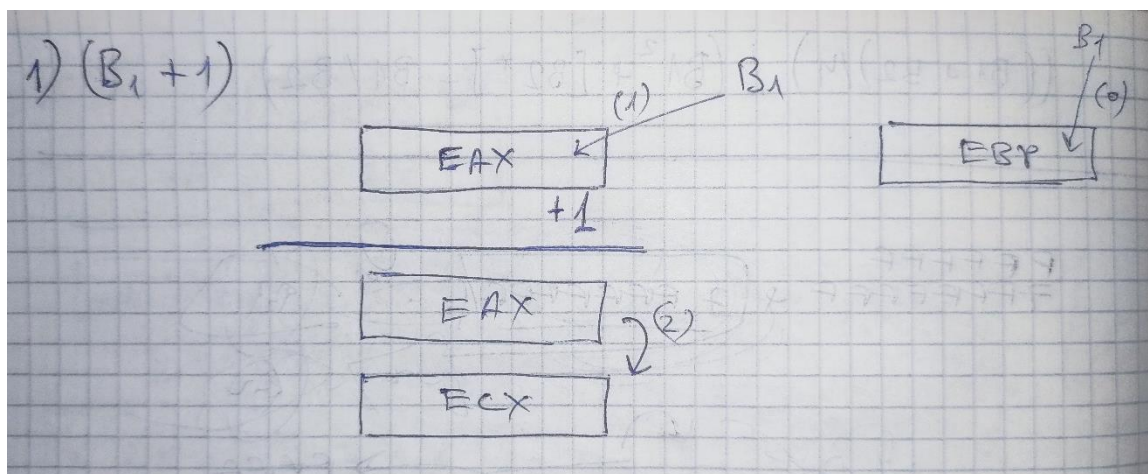
$$B1 - (B2^{2/4}) = 0xE3983F$$

$$(B1+1)*([B1*]-1) / (B1 - (B2^{2/4})) = 0xF0BC041A \text{ i ostatak } 0xFFCC68E2$$

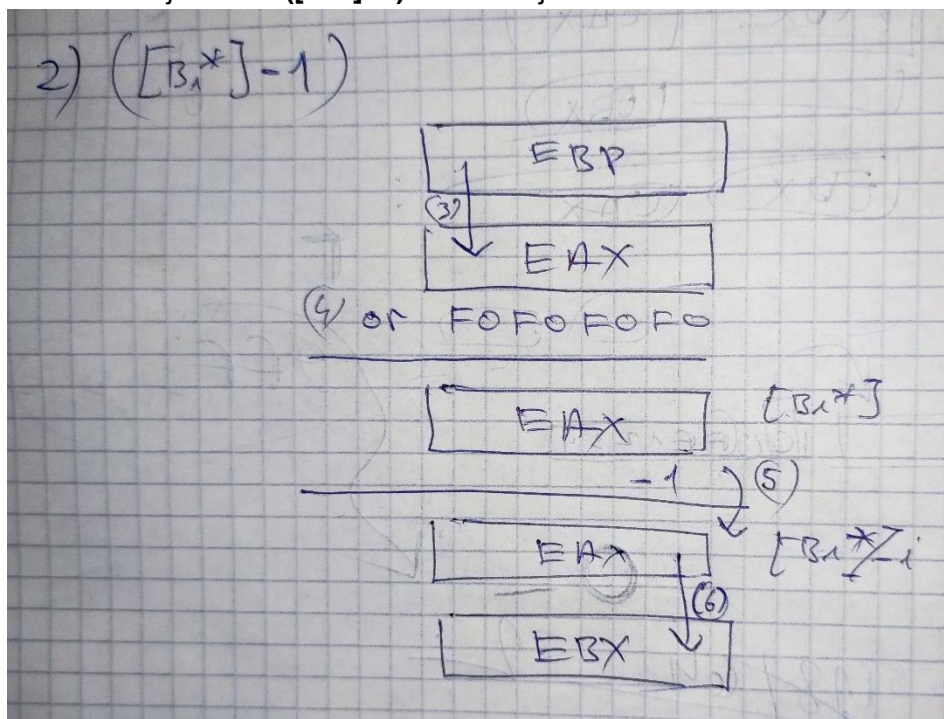
Dati problem rastavlja se na 6 potproblema:

- Izračunavanje izraza **(B1+1)**
- Izračunavanje izraza **([B1*]+1)**
- Izračunavanje izraza **(B1+1)*([B1*]+1)**
- Izračunavanje izraza **(B2^2/4)**
- Izračunavanje izraza **(B1-B2^2/4)**
- Izračunavanje izraza oblika **A / C**

1) Izračunavanje izraza **(B1+1)** i smeštanje međurezultata u registar ECX

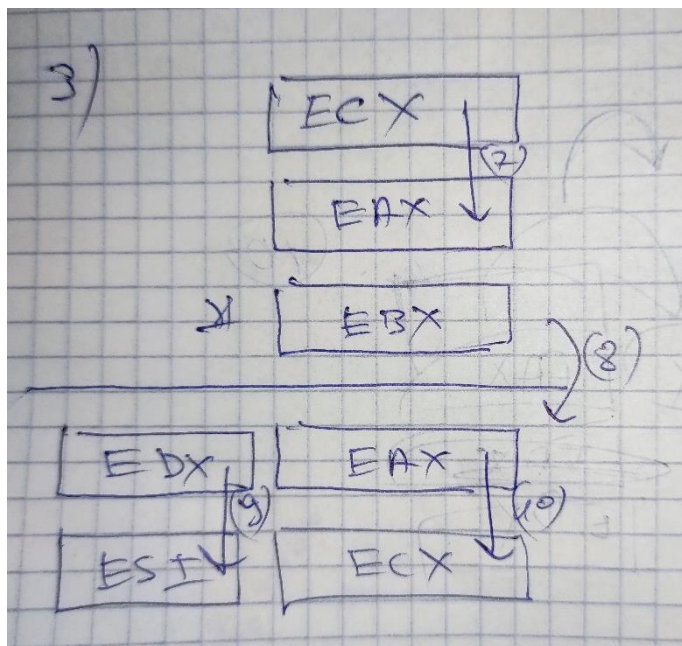


2) Izračunavanje izraza **([B1*]+1)** i smeštanje međurezultata u EBX

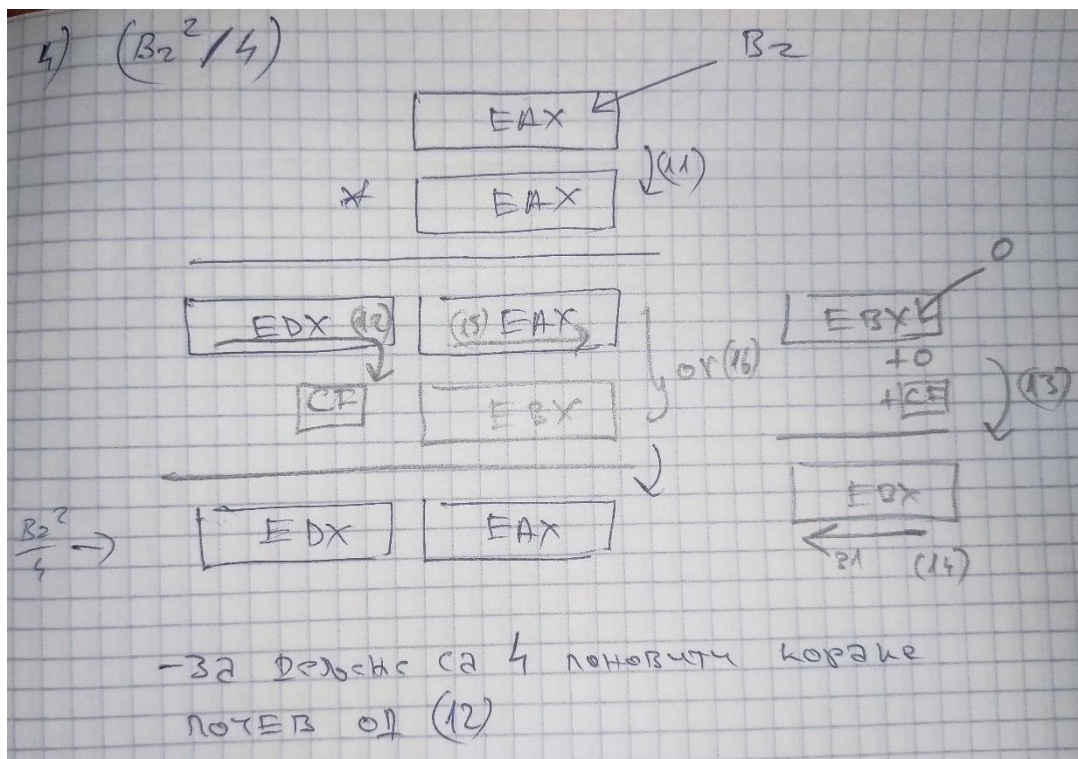


Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

3) Izračunavanje izraza $(B1+1)*([B1*]+1)$ i smeštanje međurezultata u ESI:ECX

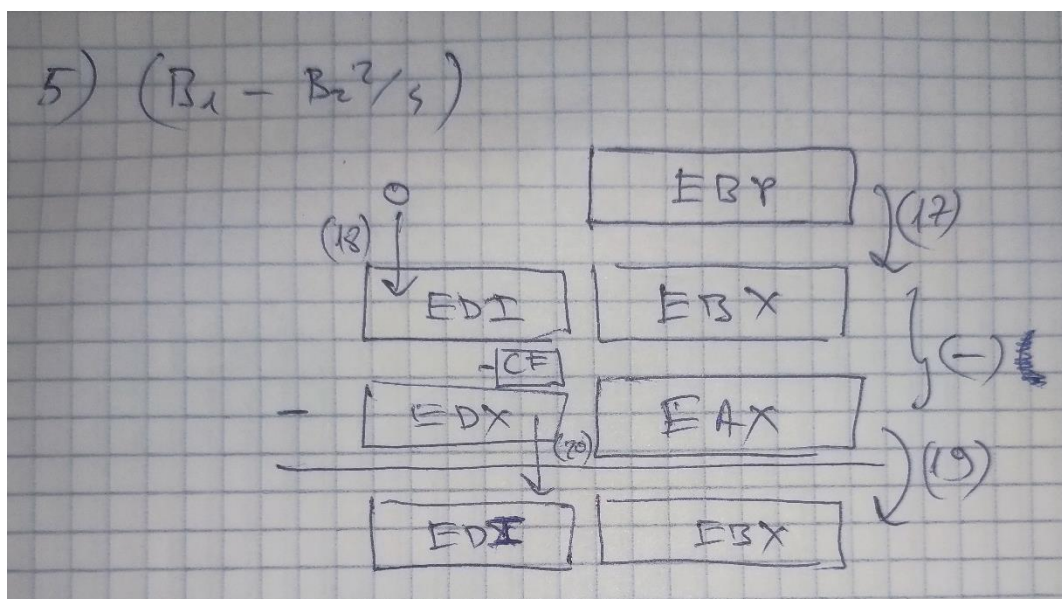


4) Izračunavanje izraza $(B2^2/4)$

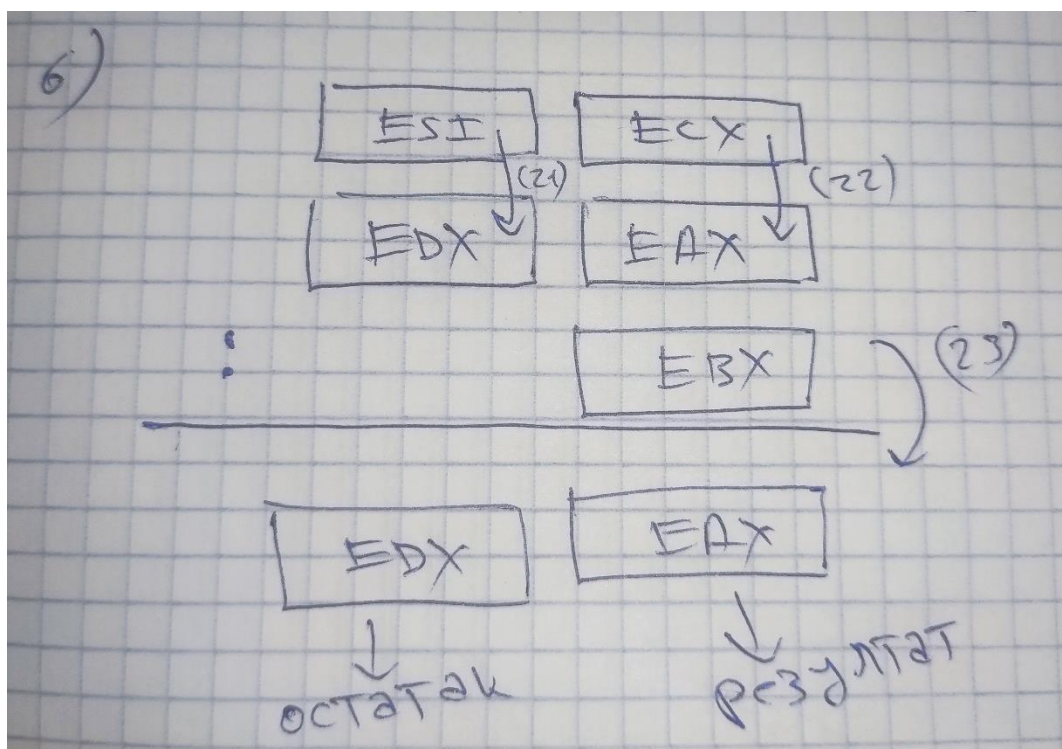


Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

- 5) Izračunavanje izraza $(B_1 - B_2^{2/4})$ i smeštanje međurezultata u EDI:EBX



- 6) Izračunavanje izraza oblika A / C



Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

b) Kod rešenja, izveštaj o testiranju

; (B1+1)*([B1*]-1) / (B1-B2^2/4)

; B1 - 32b oznacen

; B2 - 32b oznacen

; (B1+1)

;mov eax, 61CA2456h

;mov eax, 07ffffffh

;mov eax, 1900h

;mov eax, 0a12f1h

mov eax, 0e723c3h

mov ebp, eax

inc eax

mov ecx, eax

; ([B1*]-1)

mov eax, ebp ;B1

or eax, 0F0F0F0F0H

dec eax

; (B1+1) * ([B1*]-1) u ESI:ECX

mov ebx, eax

mov eax, ecx

imul ebx

mov ecx, eax

mov esi, edx

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

; $B2^2$ u EDX:EAX

;mov eax, 12F43h

;mov eax, 2

;mov eax, 0A0h

;mov eax, 7fff12h

mov eax, 3c4h

imul eax

; $(B2^2 / 4)$ u EDX:EAX

sar edx, 1

mov ebx, 0

adc ebx, 0

sal ebx, 31

shr eax, 1

or eax, ebx

sar edx, 1

mov ebx, 0

adc ebx, 0

sal ebx, 31

shr eax, 1

or eax, ebx

; $(B1 - (B2^2 / 4))$ u EDI:EBX

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

`mov ebx, ebp ;B1`

`sub ebx, eax`

`mov edi, 0`

`sbb edi, edx`

`; deljenje (levi/desni)`

`mov edx, esi`

`mov eax, ecx`

`idiv ebx`

`; u EAX (ostatak u EDX)`

Prvi deo koda smešta B1 u registar EAX i inkrementira ga. Rezultat se prebacuje u ECX. Drugi deo koda ponovo smešta B1 u registar EAX i radi maskiranje i dekrementiranje. Razlog za maskiranje je objašnjen u prvom primeru iz zadatka pod a). Dobijeni rezultati iz prvo i drugog dela koda se množe i smeštaju u registar ESI:ECX. Pošto su oba broja 32b, rezultat staje u 64b registar.

Sada se B2 smešta u EAX, i $B2^2$ se dobija jednostavim množenjem registra EAX sa samim sobom. Rezultat se nalazi u EDX:EAX. Deljenje sa 4 je ekvivalentno pomeranju registra za 2 mesta udesno. Međutim, kako se rezultat kvadriranja nalazi u 64b registru nije moguće odjednom pomeriti za dva mesta udesno jer će doći do gubljenja nižih bitova registra EDX. Zato se deljenje sa 4 vrši deljenjem sa 2 dva puta. Prvo se EDX pomeri za 1 mesto udesno, a bit koji je ispao iz registra EDX se sada nalazi u CF. Prebacimo ga u EBX i pomerimo za 31 mesto ulevo kako bi ga operacijom OR setovali na mesto gde je trebao da dođe. Pre toga se i registar EAX pomeri za jedno mesto udesno. Ovaj postupak se ponovi dva puta. Detaljnije u 1. primeru u zadatku pod a).

Za izraz $B1 - B2^2/4$ je problem što oduzimamo 32b registar od 64b registra. Međutim, kako radimo sa označenima ovo je moguće, proširimo još jedan registar znakom iz B1 i dobijeni rezultat smestimo u 64b.

Ali, ako je ovaj rezultat 64b nemamo mogućnost deljenja dva 64b broja. Zato, ograničenje programa je da rezultat $B1 - B2^2/4$ mora biti 32b broj inače emulator neće izbaciti tačno rešenje.

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

2) B1 = 0x7FFFFFFF, B2 = 0x2

Prilikom inkrementiranja B1 dolazi do prekoračenja opsega. Aktivira se Overflow Flag.
Dobijeni rezultat: 0x0

Očekivani rezultat: 0x80038B9C

```
8 ;mov eax, 61CA2456h
9 mov eax, 07ffffffh
10 ;mov eax, 1900h
11 ;mov eax, 0a12f1h
12 ;mov eax, 0e723c3h
13 mov ebp, eax
14 inc eax
15 → mov ecx, eax
16
17 ; ([B1*]-1)
18 mov eax, ebp ;B1
19 or eax, 0F0F0F0F0H
20 dec eax
21
22 ; (B1+1) * ([B1*]-1)      u ESI:ECX
23 mov ebx, eax
24 mov ecx, ecx
25 imul ebx
26 mov ecx, eax
27 mov esi, edx
```

Registers				✕
EAX	0x80000000	EBX	0x00000000	
ECX	0x00000000	EDX	0x00000000	
ESI	0x00000000	EDI	0x00000000	
EBP	0x7FFFFFFF	ESP	0x00020400	
EIP	0x0002040C	Ln 15, Col 1		
Flags				
Carry	0	Dir	0	
Int	0	Overflow	1	
Sign	1	Zero	0	

3) B1 = 0x1900, B2 = 0xA0

Dolazi do deljenja nulom. Emulator izbacuje grešku Division by 0

```
34 ; (B2^2 / 4)
35 sar edx, 1
36 mov ebx, 0
37 adc ebx, 0
38 sal ebx, 31
39 shr eax, 1
40 or eax, ebx
41
42 sar edx, 1
43 mov ebx, 0
44 adc ebx, 0
45 sal ebx, 31
46 shr eax, 1
47 or eax, ebx
48
49 ; (B1-(B2^2 / 4))      u EDI:
50 mov ebx, ebp ;B1
51 sub ebx, eax
52 mov edi, 0
53 sbb edi, edx
54
55 ; deljenje (levi/desni)
56 mov edx, esi
57 mov eax, ecx
58
59 → idiv ebx ; u EAX (ostatak u EDX)
60
```

Registers				✕
EAX	0x795950EF	EBX	0x00000000	
ECX	0x795950EF	EDX	0xFFFFFE87	
ESI	0xFFFFFE87	EDI	0x00000000	
EBP	0x00001900	ESP	0x00020400	
EIP	0x00020480	Ln 59, Col 1		
Flags				
Carry	0	Dir	0	
Int	0	Overflow	0	
Sign	0	Zero	1	

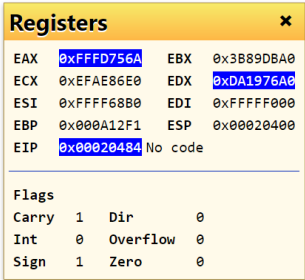
Division by 0

Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

4) B1 = 0xA12F1, B2 = 0x7FFF12

Primer kada nije moguće podeliti dva 64b broja. Registar EDI nije 0 => nije moguće podeliti ova dva broja, a kako emulator radi deljenje 32b izbacuje netačan rezultat: 0xFFFFD756A
Očekivani rezultat: 0x9

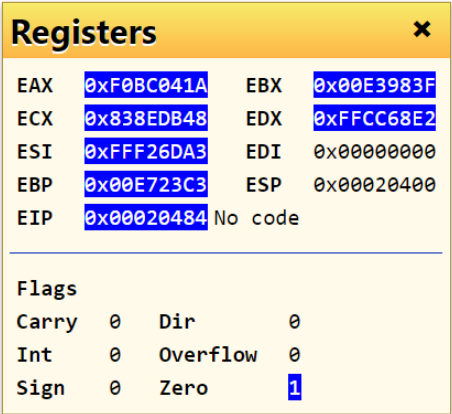
```
34 ; (B2^2 / 4)          u EDX:EAX
35 sar edx, 1
36 mov ebx, 0
37 adc ebx, 0
38 sal ebx, 31
39 shr eax, 1
40 or eax, ebx
41
42 sar edx, 1
43 mov ebx, 0
44 adc ebx, 0
45 sal ebx, 31
46 shr eax, 1
47 or eax, ebx
48
49 ; (B1-(B2^2 / 4))      u EDI:EBX
50 mov ebx, ebp ;B1
51 sub ebx, eax
52 mov edi, 0
53 sbb edi, edx
54
55 ; deljenje (levi/desni)
56 mov edx, esi
57 mov eax, ecx
58
59 idiv ebx                ; u EAX (ostatak u EDX)
60
```



5) B1 = 0xE723C3, B2 = 0x3C4

Primer prosečnih vrednosti kada se dobija tačan očekivani rezultat: 0xF0BC041A

```
39 snr eax, 1
40 or eax, ebx
41
42 sar edx, 1
43 mov ebx, 0
44 adc ebx, 0
45 sal ebx, 31
46 shr eax, 1
47 or eax, ebx
48
49 ; (B1-(B2^2 / 4))      u EDI:EBX
50 mov ebx, ebp ;B1
51 sub ebx, eax
52 mov edi, 0
53 sbb edi, edx
54
55 ; deljenje (levi/desni)
56 mov edx, esi
57 mov eax, ecx
58
59 idiv ebx                ; u EAX (ostatak u EDX)
60
```



Arhitektura i organizacija računara 1 – laboratorijska vežba - Izveštaj

Samoevaluacija

Na skali 0-5 (0 - „nikako“, „nimalo“; 5 - „potpuno“), u kom stepenu smatrate da ste:

1) <i>bili savladali gradivo PRE početka rada na vežbi</i>	4
2) <i>razumeli zadatak</i>	5
3) <i>ispunili zahteve zadatka a)</i>	4
4) <i>ispunili zahteve zadatka b)</i>	4
5) <i>istestirali i opisali funkcionisanje svog rešenja</i>	4
6) <i>razumeli ponašanje svog rešenja i pojedinih instrukcija i mehanizama</i>	4
7) <i>imali dovoljno vremena za vežbu</i>	3
8) <i>unapredili svoje znanje u toku vežbe</i>	5

Aleksandar Gospavić, 17.11.2023., 23:06