

Tomas Paukštė

KOMPIUTERIŲ ARCHITEKTŪROS UŽDUOTYS

beta 2 versija ☺

**Būčiau labai dėkingas, jei apie rastas klaidas parašytumėte į
tomas.paukste@gmail.com**

Vilnius, 2006

Turinys:

Dešimtainių supakuotų skaičių sudėtis.....	3
Konvertavimas iš vienos skaičiavimo sistemos į kitą	4
Skaičiaus užrašymas baituose.....	5
Mikroprogramavimo kalba (MPL)	11
Požymių registras SF (Status Flag)	15
JMP	17
LOOP	19
Stekas.....	21
AAD, AAM, AAA (AAS), DAA (DAS)	24
AAD	24
AAM.....	25
AAA (AAS).....	26
DAA (DAS).....	26
REP	28

Dešimtainių supakuotų skaičių sudėtis

Sudėti dešimtainius supakuotus skaičius. Pateikti tarpinius rezultatus:

d) $55 + 61 = B6 \rightarrow 16$ Ats.: 16

e) $96 + 67 = FD \rightarrow 103 \rightarrow 63$ Ats.: 63

f) $61 + 45 = A6 \rightarrow 06$ Ats.: 06

Sudėti dešimtainius supakuotus skaičius. Pateikti tarpinius rezultatus.

a. $62 + 63$

b. $68 + 66$

c. $55 + 47$

a) $62 + 63 = C5 \rightarrow 25$ Ats.: 25

b) $68 + 66 = CE \rightarrow D4 \rightarrow 34$ Ats.: 34

c) $55 + 47 = 9C \rightarrow A2 \rightarrow 02$ Ats.: 02

Sprendimas

1) Darome prielaidą, kad šie skaičiai yra supakuoti dešimtainiai.

2) Sudedame skaičius. a. $C5h$ AF=0 b. CEh AF=0 c. $9Ch$ AF=0

3) Tariame, kad skaičiai yra AL ir vykdome DAA algoritmą.

a. $25h$ AF=0 CF=1

b. $34h$ AF=1 CF=1

c. $02h$ AF=1 CF=1

PATAISYMAS: c. iš $12h$ į $02h$

Atsakymas

a. $25h$ AF=0 CF=1 b. $34h$ AF=1 CF=1 c. $02h$ CF=AF=1

Konvertavimas iš vienos skaičiavimo sistemos į kitą

a) Parvesti dešimtaini skaičių 577,03 į aštuntainę pozicinę skaičiavimo sistemą.

Ats.: 1101,0(17270243656050753412).

b) Paversti dešimtainį skaičių -18,017 į šešioliktainę pozicinę skaičiavimo sistemą.

Ats.: -12,0(45A1CAC083126E978D4EDE3B6).

Užduotys:

a) Perversti dešimtainį skaičių 1000.001 į dvejetainę ir šešioliktainę skaičiavimo sistemas.

b) Pervesti dešimtaini skaičių 19.7 į dvejetainę ir šešioliktainę skaičiavimo sistemas.

1. Perversti iš dešimtaines -19,019 į sesioliktaine.

Pirmiausia $19(\text{dec})=13(\text{hex})$

$0,019(\text{dec}) = 0,0(4\text{DD}2\text{F}1\text{A}9\text{FBE}76\text{C}8\text{B}439581062)$

dec = hex

$0,019 \cdot 16 = 0,304$ (imame sveikąją dalį $0 = 0$)

$0,304 \cdot 16 = 4,864$ ($4 = 4$)

$0,864 \cdot 16 = 13,824$ ($13 = \text{D}$)

$0,824 \cdot 16 = 13,184$ ($13 = \text{D}$)

$0,184 \cdot 16 = 2,944$ ($2 = 2$)

$0,944 \cdot 16 = 15,104$ ($15 = \text{F}$)

$0,104 \cdot 16 = 1,664$ ($1 = 1$)

$0,664 \cdot 16 = 10,624$ ($10 = \text{A}$)

$0,624 \cdot 16 = 9,984$ ($9 = 9$)

$0,984 \cdot 16 = 15,744$ ($15 = \text{F}$)

$0,744 \cdot 16 = 11,904$ ($11 = \text{B}$)

$0,904 \cdot 16 = 14,464$ ($14 = \text{E}$)

$0,464 \cdot 16 = 7,424$ ($7 = 7$)

$0,424 \cdot 16 = 6,784$ ($6 = 6$)

$0,784 \cdot 16 = 12,544$ ($12 = \text{C}$)

$0,544 \cdot 16 = 8,704$ ($8 = 8$)

$0,704 \cdot 16 = 11,264$ ($11 = \text{B}$)

$0,264 \cdot 16 = 4,224$ ($4 = 4$)

$0,224 \cdot 16 = 3,584$ ($3 = 3$)

$0,584 \cdot 16 = 9,344$ ($9 = 9$)

$0,344 \cdot 16 = 5,504$ ($5 = 5$)

$0,504 \cdot 16 = 8,064$ ($8 = 8$)

$0,064 \cdot 16 = 1,024$ ($1 = 1$)

$0,024 \cdot 16 = 0,384$ ($0 = 0$)

$0,384 \cdot 16 = 6,144$ ($6 = 6$)

$0,144 \cdot 16 = 2,304$ ($2 = 2$)

$0,304 \cdot 16 = 4,864$ (pradedą periodiskai kartotis)

Taigi $-19,019(\text{dec}) = -13,0(4\text{DD}2\text{F}1\text{A}9\text{FBE}76\text{C}8\text{B}439581062)$ (hex)

Skaičiaus užrašymas baituose

2 baituose: -87

4 baituose: -17.17, -3, -2, 57.17

8 baituose: -43.43

Užrašyti skaičių -43,43 slankaus kablelio formatu 8 baituose.

Sprendimas:

Slankaus skaičiaus 8 baituose formatas toks:

pirmas bitas s saugo skaičiaus ženklą

$((-1)^s$ taigi 1, jei neigiamas)

- 11 bitų skiriama skaičiaus charakteristikai saugoti

(charakteristika = eilė + 3FFh)

- likusiuose 52 bituose saugoma mantisė. Pirmasis skaičius (skaičius prieš kablelį) nesaugomas, jis visada lygus 1).

Sveikoji dalis:

$$43 (10) = 32 + 8 + 2 + 1 (10) = 101011 (2)$$

Trupmeninė dalis:

$$0.43 * 2 = 0.86$$

$$0.86 * 2 = 1.72$$

$$0.72 * 2 = 1.44$$

$$0.44 * 2 = 0.88$$

$$0.88 * 2 = 1.76$$

$$0.76 * 2 = 1.52$$

$$0.52 * 2 = 1.04$$

$$0.04 * 2 = 0.08$$

$$0.08 * 2 = 0.16$$

$$0.16 * 2 = 0.32$$

$$0.32 * 2 = 0.64$$

$$0.64 * 2 = 1.28$$

$$0.28 * 2 = 0.56$$

$$0.56 * 2 = 1.12$$

$$0.12 * 2 = 0.24$$

$$0.24 * 2 = 0.48$$

$$0.48 * 2 = 0.96$$

$$0.96 * 2 = 1.92$$

$$0.92 * 2 = 1.84$$

$$0.84 * 2 = 1.68$$

$$0.68 * 2 = 1.36$$

$$0.36 * 2 = 0.72$$

$$0.72 * 2 = 1.44$$

...

(Jei gerai sekasi dauginti, tai verčiau dauginti iš 16!)

Taigi

$$-43.43_{(10)} = -101011.01(10111000010100011110)_{(2)} = (-1)^1 * 2^5 * 1.0101101(10111000010100011110)_{(2)}$$

$$\text{Charakteristika} = 3FFh + 5 = 404h$$

Surašome viską į 8 baitų formatą (mantisę pildome tol, kol užpildom visus 52 bitus)

1 100 0000 0100 0101 1011 0111 0000 1010 0011 1101 0111 0000 1010 0011 1101 0111

(Pirmas bitas parodo ženklą, toliau 11 bitų užrašoma charakteristika 404h, likę 52 bitai - mantisė)

Užrašę tai šešioliktaine sistema gauname C0 45 B7 0A 3D 70 A3 D7

Ats.:

C0 45 B7 0A 3D 70 A3 D7

Užrašyti skaičių -87 skaičiaus su ženklu formatu dviejuose baituose. Atsakymą pateikti šešioliktaine išraiška.

Ats.:

FF A9

Užrašyti skaičių -2 slankaus kablelio formatu keturiuose baituose. Atsakymą pateikti šešioliktaine išraiška.

Ats.:

C0 00 00 00

Užrašyti skaičių -3 slankaus kablelio formatu keturiuose baituose. Atsakymą pateikti šešioliktaine išraiška.

Ats.:

C0 40 00 00

Užrašyti skaičių 57,17 slankaus kablelio formatu keturiuose baituose. Atsakymą pateikti šešioliktaine išraiška.

Ats.:

42 64 AE 14

Užrašyti skaičių -17,17 slankaus kablelio formatu keturiuose baituose. Atsakymą pateikti šešioliktaine išraiška.

Sprendimas:

$$-17 = -10001b$$

$$0,17 = 0,00(10101110000101000111)b$$

$$-10001b = -1,0001 \cdot 2^4$$

$$eilė = 4$$

$$\text{charakteristika} = \text{eilė} + 127 = 131 = 10000011b$$

$$\text{mantisė} = 00010010101110000101000b$$

$$\text{skaičius neigiamas taigi 31 bitas} = 1$$

$$\text{skaičius: } 1 \ 10000011 \ 00010010101110000101001b = C1 \ 89 \ 5C \ 28h$$

Ats.:

C1 89 5C 28

Užduotys:

a) Užrašyti skaičių 51,15 slankaus kablelio formatu keturiuose baituose. Atsakymą pateikti šešiolyktaine išraiška.

b) Užrašyti skaičių 41,37 slankaus kablelio formatu keturiuose baituose. Atsakymą pateikti šešiolyktaine išraiška.

2. Užrašyti desimtaini -78 skaičiu su zenklu formatu dviejuose baituose sesioliktaine sistema.

$$78(\text{dec}) = 1001110(\text{bin})$$

dviejuose baituose tai atrodytu:

$$0000000001001110$$

$$111111110110001 \text{ (invertuojame)}$$

$$111111110110010 \text{ (+1) tai ir yra -78}$$

Sesioliktaineje tai būtų: FFB2

3. Užrašyti desimtain. -3 slankaus kablelio formatu keturiuose baituose sesioliktaine sistema.

$$-3(\text{dec}) = -11 \cdot 2^0 = (-1)^1 \cdot 1,1 \cdot 2^1 \text{ (1 prie 2 yra eilė)}$$

$$\text{charakteristika} = \text{eilė} + 7F = 1 + 7F = 80$$

slankus kablelis 4 baituose atrodo taip: zenklui 1 bitas (jei teigiamas - 0, jei neigiamas 1), charakteristika 8 bitai, o likę 23 mantise.

Tada -3 slankaus kablelio pavidale yra:

$$11000000010000000000000000000000. \text{ Sesioliktainiu pavidalu tai yra C0400000.}$$

Jei yra pvz. 3,1 tai gaunasi periodine ir tada mantise pildai tol kol yra vietos.

2. Paversti skaičiu x (A variantui x = -87) į skaičiaus su zenklu formata dviejuose baituose. Atsakymą pateikti sesioliktainia išraiška.

(ATS: FFA9)

3. Užrašyti desimtaini skaičiu -43,3 slankaus kablelio formatu astuoniuose baituose sesioliktaine sistema.

Uzrasyti -17.17 slankaus kablelio formatu 4 baituose.

Sprendimas

Slankaus kablelio formatas is esmes apibreziamas taip: $(-1)^s \cdot 2^{\text{eile}} \cdot 1..$

|s|charakteristika|matise|

s - zenklo bitas (uzima viena bita), 0 - teigiamas, 1- neigiamas;
charakteristika = eile + 127 (jei 8 baituose + 3FF) (uzima 8 bitus)
mantise - skaiciau dalis po kablelio (uzima 23 bitus)

1. Pirmiausiai -17.17 neigiamas, tai pirmas bitas 1.

2. Surandame artimiausia MAZESNI uz skaiciu 2 laipsni, (siuo atveju $2^4 = 16$)

2.1. Randame charakteristika: $4 + 127 = 131$;

2.2. Randame mantise, tam sprendzime lygti $16 \cdot x = 17.17$
gauname 1,073125

2.3 Mantise verciame i devjetaini (sesioliktaini ar 8-taini formata, kaip kam patogiau)

Vertimo algoritmas toks:

Mantise (1,073125) dauginam is 2 (16 ar 8), tada rezultata mod 2 (16 ar 8) ir gauta skaiciu uzsirasome. pvz.:

$(1,073125 \cdot 2) \bmod 2 = 0,14625$

$(0,14625 \cdot 2) \bmod 2 = 0,2925$

$(0,2925 \cdot 2) \bmod 2 = 0,585$

$(0,585 \cdot 2) \bmod 2 = 1,17$ ir t.t. 23 tris kartus :)

tada eiles tvarka (is kaires i desine) uzsirasome sveikasiais dalis, gauname:
00010010101110000101000

3. dabar viska sudeliojam i vietas:

zenklo bitas = 1

charakteristika 131 = 10000011

mantise = 00010010101110000101000

11000001100010010101110000101000 = C1895C28

sutinku, tik man atsakymas gavosi c1892b85

Atsakymas

C1895C28

Užrašyt dešimtaini skaičiu 41,37
slankaus kablelio formatu 4 baituose 16 sistema
Sprendimas

Turim skaičiu 41.37

Pirma pasiverciam 41 i dvejetainę (ta turite mokėti) 41->101001
tada 0.37 pasiverciam irgi i dvejetainę:

$$(0,37 * 2) = 0,74 \rightarrow 0$$

$$(0,74 * 2) = 1,48 \rightarrow 1$$

$$(0,48 * 2) = 0,96 \rightarrow 0$$

trumpinu

92 1

84 1

68 1

36 1

72 0

44 1

88 0

76 1

52 1

04 1

08 0

16 0

32 0

64 0

28 1

56 0

12 1

24 0

48 0

96 0

nu ir užteks tarkim:)

ir turime 101001,01011110101110000101000

dabar taip : $1.0100101011110101110000101000 * 2^5$ (bet mantise tik 23 bitu, nuo galo numetu skaičius kad gautusi 23)

tai:

$$1.01001010111101011100001 * 2^5$$

charakteristika tokia $127 + 5$ tai $132 = 10000100$

zenklas -> 0

tai galutinai turime 01000010001001010111101011100001 -> 42257AE1

Atsakymas

42257AE1

3. Uždasyti desimataini skaičiu -37.37 koprosoriaus vidiniu formatu.

Mikroprogramavimo kalba (MPL)

-48, -42, -8, -6, -3, -2, 0, 34, 16382, 16383, besąlyginis valdymas

Nusiųsti -48 į MBR 2 komandom

- 1) $X = 15; D = \text{LEFT_SHIFT}(\text{COM}(1) + \text{COM}(1)); // X = 15, D = -8$
- 2) $\text{MBR} = \text{LEFT_SHIFT}(\text{COM}(X) + D); // \text{COM}(X) = 16, \text{MBR} = (-8 + (-16)) * 2 = -48$

Ats.:

- 1) $X = 15; D = \text{LEFT_SHIFT}(\text{COM}(1) + \text{COM}(1));$
 - 2) $\text{MBR} = \text{LEFT_SHIFT}(\text{COM}(X) + D);$
-

Dviem komandomis įrašyti dešimtainį skaičių -42 į registrą MBR.

- 1) $X = 15; D = \text{LEFT_SHIFT}(1 + 1); // X = 15, D = -4$
- 2) $\text{MBR} = \text{LEFT_SHIFT}(\text{COM}(X) + \text{COM}(D)); // \text{COM}(15) = -16, \text{COM}(4) = -5$

Ats.:

- 1) $X = 15; D = \text{LEFT_SHIFT}(1 + 1);$
 - 2) $\text{MBR} = \text{LEFT_SHIFT}(\text{COM}(X) + \text{COM}(D));$
-

Įrašyti -8 į MBR per vieną komandą

- 1) $\text{MBR} = \text{LEFT_SHIFT}(\text{COM}(1) + \text{COM}(1)); // \text{COM}(1) = -2, -2 + (-2) = -4, -4 * 2 = -8$

Ats.:

$\text{MBR} = \text{LEFT_SHIFT}(\text{COM}(1) + \text{COM}(1));$

Įrašyti -6 į MBR per vieną komandą

$-6 = \dots 111010$
 $\text{COM}(1)$
 $\dots 11110$
 $+\dots 11111$
 $=\dots 11101$
 $\text{LEFT_SHIFT}(\dots 11101) \rightarrow \dots 111010$

Ats.:

$\text{MBR} = \text{LEFT_SHIFT}(\text{COM}(1) + (-1));$

Parasyti mikrokomanda, kuri registre MBR suformuoja reikšmę -3, nenaudojant skaitymo is atminties

-3(dec) = 11111101(bin)
COM(1): 11111110
LEFT_SHIFT: 11111100
+1: 11111101

Ats.:

MBR = LEFT_SHIFT (COM (1)) + 1;

Parašyti mikrokomandą MPL kalba be konstantų ir registų, kuri į MBR įrašo -2.

COM(REG) invertuoja registro REG bitus (0 pakeičia 1, ir atvirkščiai). Taigi $MBR + COM(MBR)$, kad ir kokia pradinė MBR reikšmė būtų, grąžins 11111111111111 (registras MBR yra šešiolikos bitų). Paslinkę šią reikšmę į kairę per vieną bitą gausime 11111111111110. Tai ir yra skaičius -2 (papildomu kodu), nes $11111111111110 + 0000000000000010 = 0$ (16-oje bitų). Prašom išžiūrėti į "mūsų" Mikroprograminio lygio architektūrą (doc. A. Mitašiūno "Kompiuterių architektūra", 16 psl.), užrašas $MBR = LEFT_SHIFT(MBR + COM(MBR))$; MPL kalba reikš 6-os, 15-os, 29-os, 30-os ir 36-os magistralių atidarymą atitinkamais pocikliais. COM - complement (pakeisti, invertuoti).

Ats.:

MBR = LEFT_SHIFT (COM (MBR) + MBR);

Nesinaudojant konstantiniais registrais irasyti į MBR 0

Sprendimas

$MBR = COM (MBR) + MBR;$
 $MBR = COM (MBR) + COM (MBR);$

Cia naudojame dvi gudrybes: $COM (MBR) + MBR = -1$ ir $COM (MBR) = 0$. Kadangi nėra antro tokio registro susieto su dviejais sumatoriaus įėjimais, rezultata išsaugome MBR.

Ats.:

1) $MBR = COM (MBR) + MBR;$
2) $MBR = COM (MBR) + COM (MBR);$

Įrašyti 34 į MBR dviem komandomis

Visų pirma, reikia suprasti, ką reiškia pasakymas "dviem komandomis". MPL kalba užrašyta viena eilutė reiškia vieną komandą. Ta viena komanda turi būti įvykdoma per vieną ciklą. Pvz. negalima komanda $MBR = A + D; MBR = MBR + C;$ nes tam reikia dviejų ciklų (reiktų rašyti dviejose eilutėse - dviem komandomis).

O įrašyti 34 į MBR galima taip:

$X = 15; D = 1 + 1;$
 $MBR = LEFT_SHIFT(X + D);$

Pastaba: vietoje registro D negalėjome naudoti jokio kito registro, nes:

1) registras X yra sujungtas tik su kairiuoju sumatoriaus įėjimu, taigi kitas dėmuo turi būti sujungtas su dešiniuoju (žiūrėti schemą 16 psl.), o tokie registrai yra tik C, D, MBR ir konstantiniai.

2) tačiau nėra magistralės jungiančios registrą C su sumatoriaus IŠĖJIMU, taigi jam nieko negalima tiesiogiai priskirti (negalėtume užrašyti $C = 1 + 1$)

Tiems, kas dar nesuprato: perstumti bitus į kairę - tai tas pats, kas reikšmę dauginti iš 2, nes visi bitai perstumiami "per vieną dvejetainį laipsnį". Perstumti bitus į dešinę - tas pats, kas reikšmę dalinti iš dviejų be liekanos.

Ats.:

1) $X = 15$; $D = 1 + 1$;

2) $MBR = \text{LEFT_SHIFT}(X + D)$;

Užrašyti MPL komandą, įrašančią dešimtainį skaičių 16382 į registrą MBR.

Visų pirma, kadangi reikia įrašyti didelį skaičių, tai jau aišku, kad kažkas bus daroma su konstantiniu registru SIGN :)

$\text{SIGN} = 1000000000000000 = -32768$

$\text{COM}(\text{SIGN}) = 0111111111111111 = 32767$

$\text{COM}(1) = -2$

$\text{COM}(\text{SIGN}) + \text{COM}(1) = 32765$

Visa tai padalinus iš dviejų (be liekanos), gausime tai, ko reikia. RIGHT_SHIFT būtent ir veikia kaip dalyba iš dviejų be liekanos.

Taigi pilna komanda: $\text{MBR} = \text{RIGHT_SHIFT}(\text{COM}(\text{SIGN}) + \text{COM}(1))$

Ats.:

$\text{MBR} = \text{RIGHT_SHIFT}(\text{COM}(\text{SIGN}) + \text{COM}(1))$;

Parašyti mikrokomandą, kuri skaičių 16383 nusiunčia į registrą MBR.

Ats.: $\text{MBR} = \text{RIGHT_SHIFT}(\text{SIGN} + (-1))$ arba $\text{MBR} = \text{RIGHT_SHIFT}(\text{COM}(\text{SIGN}) + 0)$

7. Parašyti mikrokomandą, kuri skaičiu 16383 nusiuncia į registrą MBR.

Ideja tokia siunčiame į kairįjį sumatorių $\text{SIGN}(1000000000000000)$ ir jį invertuojame ($0111111111111111(\text{bin}) = 32767(\text{dec})$) ir gautą rezultatą pastumiame į dešinę, t.y. $\text{div } 2$ ($0011111111111111(\text{bin}) = 16383$). Tai būtų $\text{MBR} = \text{RIGHT_SHIFT}(\text{COM}(\text{SIGN}) + 0)$.

7. Parašyti mikrokomandą, kuri skaičiu 16383 nusiuncia į registrą MBR.

(abiem variantams tas pats?)

Ats:

$\text{MBR} = \text{RIGHT_SHIFT}(\text{SIGN} + (-1))$;

arba

$\text{MBR} = \text{RIGHT_SHIFT}(\text{COM}(\text{SIGN}) + 0)$;

Parašyti besąlyginio valdymo perdavimo mikrokomandą.

Ats.:

GOTO label;

7. Parašyti mikrokomandą, kuri į registrus A ir D nusiuncia sesioliktaine reikšme 8001.

Mano versija:

$D = (\text{SIGN} + 1); A = D;$

Poŝymų registras SF (Status Flag)

Registras SF = 0000. Sudėties komanda prie dešimtainės reikšmės 222 pridedama dešimtainė reikšmė 98. Užrašyti naują SF reikšmę.

Ats.:

SF = 0011h.

Registras SF = 0000. Atimties komanda iš dešimtainės reikšmės 99 atimama dešimtainė reikšmė -33. Užrašyti naują SF reikšmę.

Ats.:

SF = 0895h.

Registras SF = 0000. Duotos dvi dešimtainės reikšmės: 99 ir -33. Atliekama operacija CMP. Užrašyti naują SF reikšmę.

Ats.:

SF = 0895h.

// Nustatant SF, atimties komanda ekvivalenti CMP.

Registras SF=0000.

Baitu sudėties komanda prie 10-tinės reikšmės 247 yra prideda 10-ainė reikšmę 137. Pagal rezultatą suformuoti SF

Sprendimas

Sk.bezenklo - sk.suzenklu - binary

247 = -9 - 11110111

137 = -119 - 10001001

- 1) Ziurime skaičiu be zenklo suma: $247+137=384$ - iseina už ribų (ribos sk.be zenklo yra 0..256), reikšmės CF=1;
- 2) Ziurime sk. su zenklu suma: $-9 + -119=-128$ - neiseina už ribų (ribos sk. su zenklu yra -128..127), reikšmės OF=0;
- 3) Sudedam binary mode kad nustatyti likusius flagus:

1111 0111

+ 1000 1001

=11000 0000 -> SF =1(zenklo bitas =1), ZF=0(atsakymas 0) ; AF=1 (pernešimas iš jaunesnio pusbaičio buvo), PF=0 (nes vienetu skaičius atsakyme nelyginis).

Prisimename SF formatą

F E D C B A 9 8 7 6 5 4 3 2 1 0

x x x x OF DF IF TF SF ZF x AF x PF x CF

iš salygos ir sprendimo
SF= 0000 0000 1001 0001 ~ 0091h

Ats.:
0091h

Užduotys:

- a) Registras SF=0000. Sudėties komanda prie dešimtainės reikšmės 69 pridedama dešimtainė reikšmė 99. Užrašyti naują SF reikšmę.
 - b) Registras SF=0000. Sudėties komanda prie dešimtainės reikšmės 103 pridedama dešimtainė reikšmė 111. Užrašyti naują SF reikšmę.
 - c) Registras SF=0000. Sudėties komanda prie dešimtainės reikšmės 236 pridedama dešimtainė reikšmė 138. Užrašyti naują SF reikšmę.
 - d) Registras SF=0000. Sudėties komanda prie dešimtainės reikšmės 249 pridedama dešimtainė reikšmė 138. Užrašyti naują SF reikšmę.
 - e) Registras SF=0000. Sudėties komanda prie dešimtainės reikšmės 88 pridedama dešimtainė reikšmė -44. Užrašyti naują SF reikšmę.
 - f) Registras SF=FFFF. Sudėties komanda prie dešimtainės reikšmės -138 pridedama dešimtainė reikšmė -145. Užrašyti naują SF reikšmę.
 - g) Registras SF=0000. Sudėties komanda prie dešimtainės reikšmės -119 pridedama dešimtainė reikšmė 122. Užrašyti naują SF reikšmę.
 - h) Registras SF=0000. Atimties komanda iš dešimtainės reikšmės 99 atimama dešimtainė reikšmė -33. Užrašyti naują SF reikšmę.
-

JMP

AX = 0003, BX = 0000, CX = 0001, DX = 0000.

FFFA EB A1 JMP number

valdymo perdavimo adresas?

Sprendimas

FFFCh - IP reikšmė (nes kadangi komanda 2 baitų ilgio, o IP rodo į sekancia komanda), EBA1h - opkodas su operandais (2 baitai).

EBh - JMP NEAR (vidinis artimas) su 1 baito poslinkiu.

IP lygus IP + poslinkis išplėstas iki 2 baitų. A1h 1 baite (-95) -> FFA1h 2 baituose.

FFFCh
+FFA1h

1FF9Dh

Sumažiname iki IP registro dydžio (2 baitai)

Ats.:

FF9Dh

AX = 0003, BX = 0002, CX = 0001, DX = 0000.

B901 EB80 jmp number.

koks sekancios komandos adresas?

Sprendimas

B901 - IP registro reikšmė, EB80 - opkodas ir operandai.

nžn ar "jmp number" yra nurodyta sąlygoje, todėl nagrinėju opkodą. EB tai JMP NEAR (vidinis artimas) su vieno baito poslinkiu. Todėl 80 yra poslinkis.

Dėmesio!!! 80h užrašius viename baite, gauname skaičių su ženklu papildomu kodu. IP yra 2 baitų dydžio. Išplėčiame 80h (-128) iki 2 baitų, gauname FF80h (-128).

Valdymo perdavimo adresas bus IP +FF80h.

Kita nuomone/papildymas: IP +FF80h yra teisinga, bet reikia tureti omeny, kad vykdant komandą IP rodo į sekancia komanda (musu komanda 2B ilgio), todėl IP + FF80h = B901h + 2h + FF80h = B883

Ats.:

B883h

8EDC E9 12 34 jmp SKIP

Ats.: $8EDC + 1234 + 3 = (ATS: A113)$

9854 EB EC90 jmp poslinkis

EB yra jmp vidinis artimas, t.y poslinkis 1 baitas.

Tada adresas bus perduodamas: 9856(prie esamo adreso pridedame du, nes IP yra ne einamoji komanda, o kitos skaitomos komandos adresas t.y. 9856)+FFEC(priekyje prirasome FF, nes EC yra neigiamas, ne vyriausiam bite yra vienetas, o poslinkis traktuojamas kaip $-128 - 127 = -9842$)

**Ats.:
9842h**

b.8EDC E9 12 34 jmp SKIP (ATS: A113)

LOOP

Įvykdžius nurodytą komandą apskaičiuoti sekančios komandos efektyvų adresą, kai AX=0003, BX=0002, CX=0001, DX=0000.

FFFE E2 F2 90 loop number (FFFE yra poslinkis kodo segmente)

Sprendimas

FFFEh yra poslinkis kodo segmente - IP reikšmė prieš komandos įvykdymą. IP po loop number yra IP=0000 (1 baido opkodas, 1 baido poslinkis).

loop sumažina cx vienetu ir tada tikrina ar jis nelygus 0. jei nelygus tai vykdomas valdymo perdavimas. cx=1 todėl sekančios komandos EA yra IP reikšmė

**Ats.:
0000h**

Nesutvarkyta informacija:

Duotos registrų reikšmės:

AX = 0002, BX = 0001, CX = 0000, DX = 0003 ir tokia komanda

FFFD E2 FD loop cikl1 (pastaba: FFFD yra ip dabartine reiksme).

Klausimas: rasti registru ax, bx, cx, dx, ip suma ivykdzius nurodyta komanda.

3. Įvykdžius nurodyta komanda apskaičiuot sekančios komandos efektyvu adresą, kai AX=0003, BX=0002, CX=0001, DX=0000:

FFFE E2 F2 90 loop number (FFFE yra poslinkis kodo segmente)

3. Pateikti valdymo perdavimo adreso apskaiciavima:

a. E71F E2 FA loop INIT (ATS: E71D)

4. Duotos registru reiksmes ax=0002, bx=0001, cx=0000, dx=0003 ir tokia komanda FFFD E2 FD loop cikl1 (pastaba: FFFD yra ip dabartine reiksme). Klausimas: rasti registru ax, bx, cx, dx, ip suma ivykdzius nurodyta komanda.

CALL

4. Skaicia netikslus bet minti galima pagauti:
duota:

71C0 87 78 91 call number

(ATS: 033B)

pasakyti kokių adresų bus perduotas valdymas. Vėl atsakymas sesioliktainis.

Autoriaus pastaba: Čia pirmas skaičius sreiškia dabartinį IP. Dėl to paskutiniai tyris baitai - komandos kodas, na tu paskutiniai baitai poslinkis kuriuo reikia perduoti valdymą. Tai su dedam $71C3 + 9178$ ir gauname naują reikšmę. Jei tai 5 baitai, pirma bita pamirštame.

Pateikti valdymo perdavimo adresų apskaičiavimą:

3. Pateikti valdymo perdavimo adresų apskaičiavimą;

07FD 75 F1 90 jne ABC

004C 75 14 90 90 90 jne ABC

0F01 E8 FF00 call DEF

005D E8 FFBC call DEF

4. Apskaičiuoti valdymo perdavimo adresą:

71E0 E8 D1A2 call number

E8 yra call vidinis tiesioginis, t.y. poslinkis 2 baitai. tada adresas bus perduodamas

$71E0 + A2D1$ (sukeista vietomis, nes pirmiau masininiame kode eina jaunesnysis po to vyresnysis) = 14B1

Stekas

Duota AX=0100, BX=0001, DX=0002, CX=0015, SP=0000, SS=0522.

Vykdomas toks programos fragmentas:

PUSH SP

Kokia reikšmė bus įdėta į steką?

Sprendimas

Procesorius visų pirma sumažins 2 SP registro reikšmę (nes steke operuojame žodžiais), o tik po to įdės į steką.

Tai yra "ypatinga" 8086 (8088) procesorių savybė, tarkim Pentium i vėlesni dėš SP nesumažinta...

Ats.:

FFFE

Raskite steko viršūnės reikšmę įvykdžius komandą 3410 9A EBF E 1234 (call text)

SS = ABCD

SP = 0002

BP = AF00

CX = 0010

Ats.:

3415

9. Registru AX ir BX reikšmes yra CCCC, registru IP ir SP reikšmes yra desimtainis skaičius 256. Steko viršūnės reikšmė yra 1234. Vykdam fragmentą:

a. pop ss
mov sp, bx

b. pop ss
mov sp, bx

pirmąją komandą kyla pertraukimo signalas. Kokia sekanti reikšmė bus įrašyta į steką.

1. Registru AX ir BX reikšmės yra DC5C, registru IP ir SP reikšmės yra dešimtainis skaičius 1024. steko viršūnės reikšmė yra 128. Vykdam fragmentą:

a. mov ss, ax
mov sp, bx

b. mov si, ax
mov sp, bx

pirmąją komandą kyla pertraukimo signalas. Kokia sekanti reikšmė bus įrašyta į steką?

9. Duotos registru SS ir SP reikšmės. Klausimas: kokia bus registro SP reikšmė išvykdžius komandą `int 3`

9. Registru AX ir BX reikšmės yra CCCC, registru IP ir SP reikšmės yra dešimtainis skaičius 256. Steko viršūnės reikšmė yra 1234. Vykdamas fragmentą:

a. `pop ss`
`mov sp, bx`

b. `pop ss`
`mov sp, bx`

pirmąją komandą kyla pertraukimo signalas. Kokia sekanti reikšmė bus įrašyta į steką.

Registru AX ir BX reikšmės yra DC5C, registru IP ir SP reikšmės yra dešimtainis skaičius 1024. Steko viršūnės reikšmė yra 128. Vykdamas fragmentą:

`mov ss, ax`
`mov sp, bx`

`mov si, ax`
`mov sp, bx`

pirmąją komandą kyla pertraukimo signalas. Kokia sekanti reikšmė bus įrašyta į steką?

9. Duotos registru SS ir SP reikšmės. Klausimas: kokia bus registro SP reikšmė išvykdžius komandą `int 3`

9. Registras SS=ABCD, SP=00F2, BP=AF00, CX=0010. Kokia registro SP reikšmė 16-ainėje sistemoje išvykdžius grįžimo iš tolimos procedūros komandą: CACBCC.

SS=ABCD SP=00F2 BP=AF00 CX=0010

SP po sugrįžimo iš procedūros su komanda CACBCC?

Dieve gelbėk!

Dieve gelbėk!

Sprendimas

Pagal idėją žodis "sugrįžimo" duoda mums sužinoti, kad komanda RET buvo panaudota -> ir tiksliai pirmosios dvi raidės "CA" = 1100 1010 tai komandos RET binary code

Taigi toliau komanda RET daro sekancius veiksmus ->

IP talpinamas į steką

SP didinamas 2

CS talpinamas į steką

SP didinamas 2

SP = SP + betarpishkas operandas

o operandas mes turime - tai tie "CBCC"

taigi SP = SP + 2 + 2 + CCB (nes CC vyresnysis baitas operando)

SP = CDC1

Sprendė Verri bet nzn ar tai teisinga

Atsakymas

CDC1

AAD, AAM, AAA (AAS), DAA (DAS)

AAD

Algoritmas:

$AL = AH * 10 + AL$

$AH = 0$

Registras $AL = 07$, $BX = AF00$, $CX = 0002$. Kokia bus registro AX reikšmė, įvykdžius komandą AAD?

Ats.:

$AX = 0007h$

$AX = 0C05$. Kokia bus registro AX reikšmė, įvykdžius komandą AAD?

Ats.:

$AX = 007Dh$

Registras $AL = 07$, $AH = 05$, $BX = AF00$, $CX = 0001$. Kokia bus registro AX reikšmė šešioliktainėje sistemoje įvykdžius komandą AAD?

Komanda AAD atlieka tokius pertvarkimus:

$AL := AH * 10 + AL$;

$AH := 0$;

AL reikšmė bus $57 = 39h$

$AH = 0$

Tai AX reikšmė bus $0039h$

Ats.:

$AX = 0039h$

1. $AL=FC$ $AH=07$ $CX=0000$ vykdoma komanda AAD. $AX=?$

$AL = FC$, $AH = 07$, $CX = 0$. Parašykite AX reikšmę po AAD.

AAD (ASCII Adjust AX before Division) operacija:

$AL = AH * 10 + AL$

$AH = 0$,

taigi:

$AL = 07h * 0Ah + FCh = (1)42h$

$AH = 0$

$AX = 0042h$

Ats.:

$AX = 0042h$

$AL = FF$, $AH = 06$, $CX = 0000$. Koks AX po AAD ?

Visų pirma pervedame į nesupakuotą dešimtainių formatą. Tai darome pagal AAS algoritimą, bet neatsižvelgdami į AF (taip reikia:). Gauname 0509h (2 baituose). Tai dešimtainis skaičius 59.

Pervedame 59 į skaičiaus be ženklo formatą 2 baituose (AX registre) ir gauname $AX=3Bh$

karoče žinau paprastesnį būdą (pagal AAD vykdymo algoritma):

$AL = AH \times 10 + AL$;

$AH = 0$;

taigi pirmiausia AH padauginam iš 10 (dešimtainio):

$06h \times 10 = 3Ch$

pridedam AL :

$FF + 3C = 13B$

Taigi: $AL = 3B$, $AH = 00$, bendrai $AX = 003B$

Ats.:

$AX = 003Bh$

AAM

Algoritmas:

$AH = AL \div 10$

$AL = AL \bmod 10$

Duotos registrų reikšmės: $AH = 09$, $AL = 89$, $CX = 0002$. Kokia bus registro AX reikšmė įvykdžius komandą AAM ?

Ats.:

AX = 0D07h

AX = 0713. Vykdoma komanda AAM. AX = ?

Ats.:

AX = 0109h

AAA (AAS)

Algoritmas:

if ((AL and 0Fh) > 9 or (AF = 1)) then

 AL = AL + (-) 6

 AH = AH + (-) 1

 AF = 1

 CF = 1

else

 AF = 0

 CF = 0

endif

AL = AL and 0Fh

2. Duota registro AX reiksme ax=000B ir AF = 1.Klausimas: Kokia bus registro AX reiksme ivykdzius komanda AAS.

DAA (DAS)

Algoritmas:

if ((AL and 0Fh) > 9 or (AF = 1)) then

 AL := AL + (-) 6

 AF := 1

endif

if ((AL > 9Fh) or (CF = 1)) then

 AL := AL + (-) 60h

 CF := 1

Endif

kitam variantui lygtais buvo: rasti SF reiksme

po komandos DAA.

2. AL=FB AF=1 vykdoma komanda DAA. AL=?

3. AL=FF AF=0;
AL po DAA

Sprendimas

Vykdoma DAA konspekte aprašyta algoritmą.

AL and 0Fh lygu F. F yra daugiau už 9, todėl pridedame (operacija Decimal Adjust for Addition) prie AL 6 ir nustatome AF, gauname 5 (iš tiesų 105h bet AL tai tik 1 baitas).

AL (05) nėra daugiau už 9Fh, bet kai prie al pridejome 6, užsidejo cf=1, todėl dar pridedame 60h.

AL =65, AF =1, CF=1

IMHO, ats. turi but 05h. I CF neatsizvelgiame, nes jis užsidejo veiksmo metu.@

Atsakymas

65

REP

Registru SI ir DI reikšmės yra 0004, registras CX=0003, registras SF=0600. kokios bus registru SI ir DI reikšmės įvykdžius komandą: **rep scasw**?

Ats.:

DI = FFFE, SI = 0004

Duota: SI = 000E, DI = 000E, Cx = 0002 SF = 0C00 (cia a varianto reiksmes). Kokia bus SI ir DI suma atlikus: **rep stosw**.

Atsakymas 000C, DF = 1 (atrodo).

Registru SI ir DI reikšmės yra 000A; CX=0010(CX=0011), SF=FFFF. Kokia bus registru SI ir DI reikšmių suma, įvykdžius komanda: **rep lodsw**.

10. Registru SI ir DI reikšmės yra 001A, regist. CX=0002, reg. SF=0C00.
Kokio bus registru SI ir DI reikšmių suma, įvykdžius komanda: **rep stosw**?

Cia reikia žinoti eilutinių komandų vykdymo schema ir SF registro sudetį.
Is SF nustatome, kad DF = 1. O stosw (raide w rodo, kad bus operuojama su žodžiais).

Tai vadinasi SI ir DI reikšmės bus mazinamos 2. (is DF=1 ir tai kad operuojama su žodžiais).

Komanda STOS modifikuoja tik DI registro reikšmę.

Pirma karta vykdoma komanda

rep stosw (atliekami tokie veiksmai tikrina ar CX = 0(jei lygu baigia pakartojimo komanda), jei CX <> 0, tai CX <- CX - 1 (musu atveju CX = CX - 1 = 0001), tada DI = DI - 2 + 0018)

Antra karta vykdoma komanda

rep stosw (CX <> 0, tai CX <- CX - 1 (musu atveju CX = CX - 1 = 0000), tada DI = DI - 2 + 0016)

Trecia karta vykdoma komanda

rep stosw (CX = 0, baigiama vykdyti pakartojimo komanda)

SI reikšmė nepakito. Išliko 001A. O DI tapo 0016.

Tada DI+SI=0016+001A=0030.

(Reikia atkreipti dėmesį, kad pakartojimo komandos nutraukiamos ir dar kai Z=ZF, t.y. specifiniai atvejai).

10. Duota: SI = 000E, DI = 000E, Cx = 0002 SF = 0C00 (cia a varianto reiksmes). Kokia bus SI ir DI suma atlikus: **rep stosw**.

Atsakymas 000C, DF = 1 (atrodo).

9. Registų SI ir DI reikšmės yra 0004, registras CX=0003, registras SF=0600. kokios bus registų SI ir DI reikšmės įvykdžius komandą: **rep scasw?** (ATS: DI=FFFE, SI=0004)

10. SI=E587 DI=fff8 CX=0011 SF=0000 Kokia SI ir DI suma įvykdžius komanda **rep stosb?**

Duotos registų reikšmės:

SI=EE8B

DI=12CD

CX=0029

SF=0000

Vykdoma REP **lodsw**.

Reikia rasti SI + DI po komandos įvykdymo.

Sprendimas

Visų pirmą patikriname kokia yra DF požymio reikšmė, ji yra 0 (naudojamės duota SF registro reikšme).

Dėl prefikso bus didinama SI ir DI reikšmė. LODSW (man atrodo, kad ši komanda didina tik SI registro reikšmę, nes ši komanda į AX pakrauna iš DS:SI) komanda bus kartojama 29h kartų (CX registras nurodo kartojimų kiekį).

Kadangi pakraunamas žodis, o ne baitas, todėl bus pridedama po 2, t.y. SI=EE8Bh +2*29h =EE8Bh +52h =EEDDh.

Suma:

SI+DI

EEDDh

+12CDh

101AAh

Pagal sąlyga, neduota ar mes kur nors įrašome tą reikšmę, todėl neaišku ar rašyti 101AAh, ar 01AAh. T.y. nėra pasakyta ar tą sumą turi suskaičiuoti procesorius ar studentas, studentas "nepersipildo" (jo skaičių diapazonas nėra ribotas kaip procesoriaus), tad reikia rašyti 101AAh, bet jeigu tarkim bus parašyta add si, di, tai čia pernešimas bus.

Atsakymas

SI + DI = 101AAh

LODSB	No operands	<p>Load byte at DS:[SI] into AL. Update SI.</p> <p>Algorithm:</p> <ul style="list-style-type: none">• AL = DS:[SI]• if DF = 0 then<ul style="list-style-type: none">◦ SI = SI + 1else<ul style="list-style-type: none">◦ SI = SI - 1 <p>Example:</p> <pre>ORG 100h LEA SI, a1 MOV CX, 5 MOV AH, 0Eh m: LODSB INT 10h LOOP m RET a1 DB 'H', 'e', 'l', 'l', 'o'</pre> <table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td colspan="6">unchanged</td></tr></table>	C	Z	S	O	P	A	unchanged					
C	Z	S	O	P	A									
unchanged														

LODSW	No operands	<p>Load word at DS:[SI] into AX. Update SI.</p> <p>Algorithm:</p> <ul style="list-style-type: none"> • $AX = DS:[SI]$ • if $DF = 0$ then <ul style="list-style-type: none"> ◦ $SI = SI + 2$ else <ul style="list-style-type: none"> ◦ $SI = SI - 2$ <p>Example:</p> <pre>ORG 100h LEA SI, a1 MOV CX, 5 REP LODSW ; finally there will be 555h in AX. RET</pre>
-------	-------------	--

		<div>a1 dw 111h, 222h, 333h, 444h, 555h</div> <table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td colspan="6">unchanged</td></tr></table>	C	Z	S	O	P	A	unchanged					
C	Z	S	O	P	A									
unchanged														

MOVSB	No operands	<p>Copy byte at DS:[SI] to ES:[DI]. Update SI and DI.</p> <p>Algorithm:</p> <ul style="list-style-type: none">• ES:[DI] = DS:[SI]• if DF = 0 then<ul style="list-style-type: none">◦ SI = SI + 1◦ DI = DI + 1else<ul style="list-style-type: none">◦ SI = SI - 1◦ DI = DI - 1 <p>Example:</p> <p>ORG 100h</p> <p>CLD LEA SI, a1 LEA DI, a2 MOV CX, 5 REP MOVSB</p> <p>RET</p> <p>a1 DB 1,2,3,4,5 a2 DB 5 DUP(0)</p> <table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td colspan="6">unchanged</td></tr></table>	C	Z	S	O	P	A	unchanged					
C	Z	S	O	P	A									
unchanged														

MOVSW	No operands	<p>Copy word at DS:[SI] to ES:[DI]. Update SI and DI.</p> <p>Algorithm:</p> <ul style="list-style-type: none"> • ES:[DI] = DS:[SI] • if DF = 0 then <ul style="list-style-type: none"> ◦ SI = SI + 2 ◦ DI = DI + 2 else <ul style="list-style-type: none"> ◦ SI = SI - 2
-------	-------------	---

		<div>○ DI = DI - 2</div> <div>Example:</div> <div>ORG 100h</div> <div>CLD</div> <div>LEA SI, a1</div> <div>LEA DI, a2</div> <div>MOV CX, 5</div> <div>REP MOVSW</div> <div>RET</div> <div>a1 DW 1,2,3,4,5</div> <div>a2 DW 5 DUP(0)</div> <div><table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td colspan="6">unchanged</td></tr></table></div>	C	Z	S	O	P	A	unchanged					
C	Z	S	O	P	A									
unchanged														

SCASB	No operands	<p>Compare bytes: AL from ES:[DI].</p> <p>Algorithm:</p> <ul style="list-style-type: none">• ES:[DI] - AL• set flags according to result: OF, SF, ZF, AF, PF, CF• if DF = 0 then<ul style="list-style-type: none">◦ DI = DI + 1 <p>else</p> <ul style="list-style-type: none">◦ DI = DI - 1 <table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td>r</td><td>r</td><td>r</td><td>r</td><td>r</td><td>r</td></tr></table>	C	Z	S	O	P	A	r	r	r	r	r	r
C	Z	S	O	P	A									
r	r	r	r	r	r									

SCASW	No operands	<p>Compare words: AX from ES:[DI].</p> <p>Algorithm:</p> <ul style="list-style-type: none"> • $ES:[DI] - AX$ • set flags according to result: OF, SF, ZF, AF, PF, CF • if $DF = 0$ then <ul style="list-style-type: none"> ○ $DI = DI + 2$ <p>else</p>
-------	-------------	--

		<ul style="list-style-type: none"> DI = DI - 2 <div> <div>CZSOPA</div> <div>rrrrrr</div> </div>
--	--	--

STOSB	No operands	<p>Store byte in AL into ES:[DI]. Update DI.</p> <p>Algorithm:</p> <ul style="list-style-type: none"> ES:[DI] = AL if DF = 0 then <ul style="list-style-type: none"> DI = DI + 1 else <ul style="list-style-type: none"> DI = DI - 1 <p>Example:</p> <pre>ORG 100h LEA DI, a1 MOV AL, 12h MOV CX, 5 REP STOSB RET a1 DB 5 dup(0)</pre> <div> <div>CZSOPA</div> <div>unchanged</div> </div>
-------	-------------	---

STOSW	No operands	<p>Store word in AX into ES:[DI]. Update DI.</p> <p>Algorithm:</p> <ul style="list-style-type: none"> ES:[DI] = AX if DF = 0 then <ul style="list-style-type: none"> DI = DI + 2 else <ul style="list-style-type: none"> DI = DI - 2 <p>Example:</p> <pre>ORG 100h</pre>
-------	-------------	---

		<div>LEA DI, a1 MOV AX, 1234h MOV CX, 5</div> <div>REP STOSW</div> <div>RET</div> <div>a1 DW 5 dup(0)</div> <table><tr><td>C</td><td>Z</td><td>S</td><td>O</td><td>P</td><td>A</td></tr><tr><td colspan="6">unchanged</td></tr></table>	C	Z	S	O	P	A	unchanged					
C	Z	S	O	P	A									
unchanged														