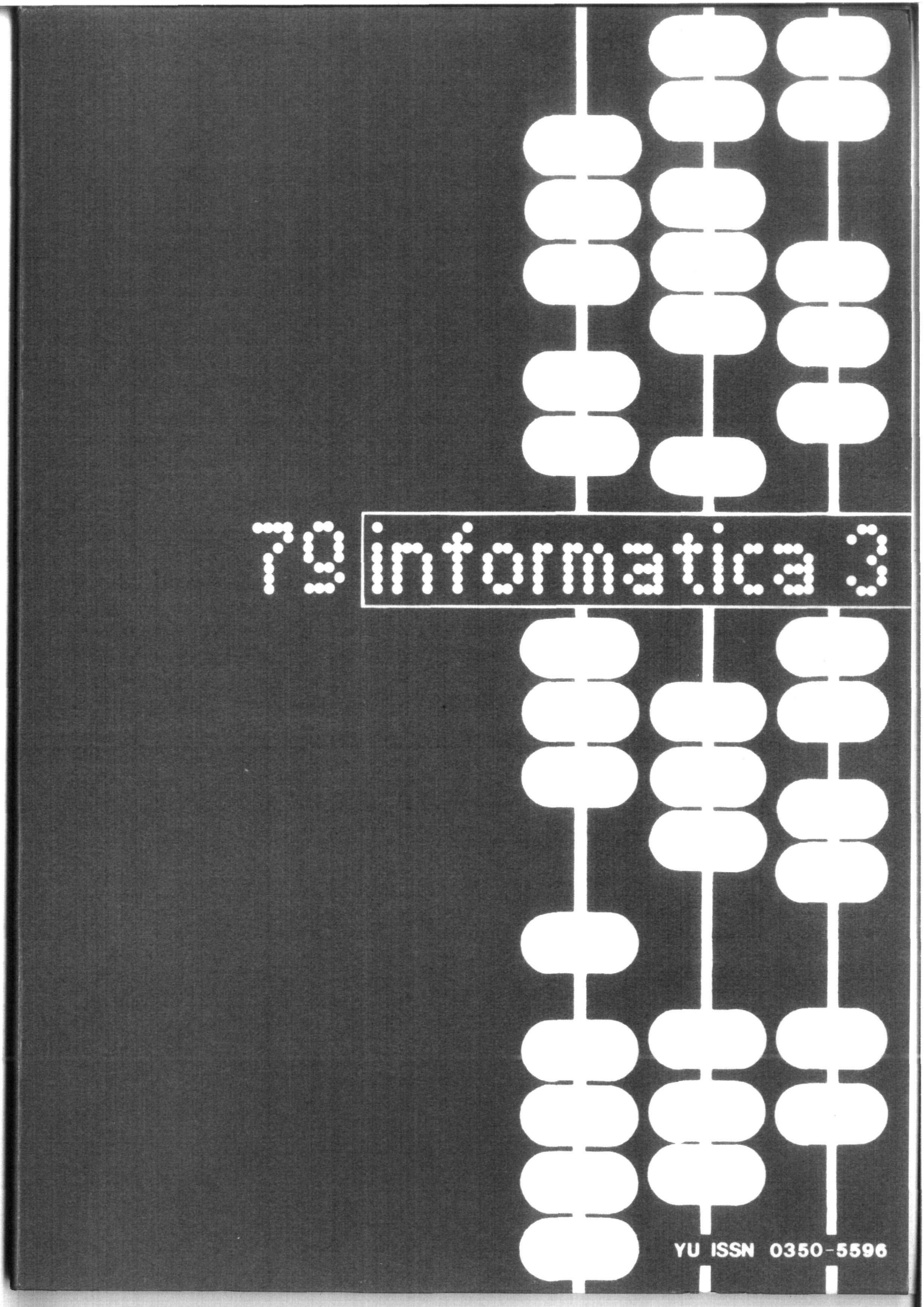
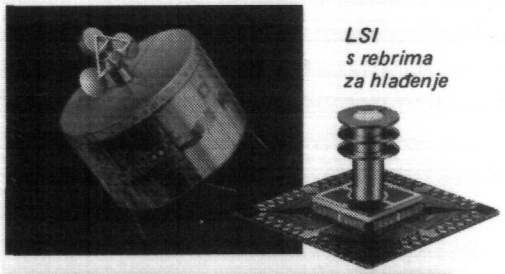
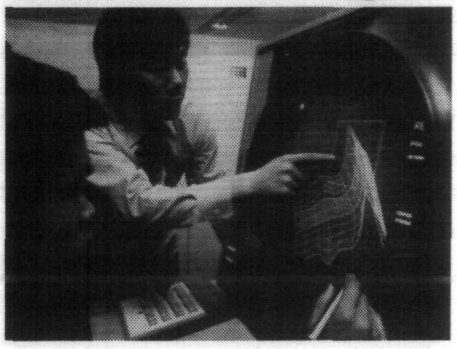


YU ISSN 0350-5596

FACOM kompjutere proizvodi Fujitsu, tvrtka koja najveću pažnju posvećuje sistemima.



LSI s rebrima za hlađenje



Prije svega kompjuter je sistem, tj. sredstvo za obradu podataka koji u sebi sadrži hardware, software i aplikacionu tehnologiju. Naravno razne tvrtke bave se prodajom kompjutera. Ipak, malo je tvrtki koje mogu ponuditi potpuni izbor sredstava za automatsku obradu podataka – konstruirani tako, da osim optimalnih performanci, imaju mogućnost ugradnje u veće sisteme.

FUJITSU je jedna od tvrtki koja to može ponuditi. Kao vodeći proizvođač kompjuterskih sistema u Japanu, FUJITSU proizvodi široki asortiman proizvoda od minikompjutera s jednim LSI čipom do u svijetu najmoćnijih LSI sistema, kao i široki izbor periferne i terminalne opreme.

FACOM kompjuteri obavljaju važne aktivnosti u poslovnim i državno-administrativnim organizacijama u mnogim zemljama širom svijeta. U Japanu, drugom po redu najvećem tržištu kompjutera u svijetu, instalirano je najviše FACOM sistema u usporedbi s drugim modelima ostalih proizvođača. Ovi moćni, pouzdani FACOM kompjuteri sposobni su za obavljanje svih mogućih poslova. Oni upravljaju satelitima u svemiru, daju prikaz atmosferskih prilika real-time grafikonima u boji, obavljaju bankovno poslovanje pomoću on-line sistema za više od 7.000 filijala i ekspozitura i još mnogo, mnogo toga.

FACOM kompjuteri su potpuno integrirani sistemi gdje se kombinacijom visoko-kvalitetne tehnologije, moćnog softwarea i već provjerenih aplikacionih programa postiže efikasnost i pouzdanost kojima nema premca.

Za dalje informacije obratite se na:

zpr **Zavod za primjenu elektroničkih računala i ekonomski inženjering**

41000 ZAGREB Savska c. 56 Telefon: 518-706, 510-760 Telex: 21689 YU ZPR FJ

FUJITSU

Fujitsu Limited·Tokyo, Japan

informatica

Published by INFORMATIKA, Slovene Society for Informatics, 61000 Ljubljana, Jamova 39, Yugoslavia

Headquarters: 61000 Ljubljana, Institut "Jožef Stefan", Jamova 39, Phone: (061)263 261, Cable: JOSTIN Ljubljana, Telex: 31 269 YU JOSTIN.

Annual subscription rate for abroad is US $ 18 for companies, and US $ 6 for individuals.

Printed by: Tiskarna KRESIJA, Ljubljana

DESIGN: Rasto Kirn

JOURNAL OF COMPUTING AND INFORMATICS

YU ISSN 0350 – 5596

VOLUME 3, 1979 – No. 3

CONTENTS

informatica

Časopis izdaja Slovensko društvo INFORMATIKA, 61000 Ljubljana, Jamova 39, Jugoslavija

Uredništvo in uprava: 61000 Ljubljana, Institut "Jožef Stefan", Jamova 39, telef. (061)263-261, telegram JOSTIN, telex: 31 269 YU JOSTIN.

Letna naročnina za delovne organizacije je 300,00 din, za posameznika 100,00 din, prodaja posamezne številke 50,00 din.
Žiro račun št.: 50101-678-51841
Stališče uredništva se lahko razlikuje od mnenja avtorjev.

Pri financiranju revije sodeluje tudi Raziskovalna skupnost Slovenije.

Na podlagi mnenja Republiškega sekretariata za prosveto in kulturo št. 4210-44/79 z dne 1.2.1979, je časopis oproščen temeljnega davka od prometa proizvodov.

Tisk: Tiskarna KRESIJA, Ljubljana

Grafična oprema: Rasto Kirn

ČASOPIS ZA TEHNOLOGIJO RAČUNALNIŠTVA IN PROBLEME INFORMATIKE
ČASOPIS ZA RAČUNARSKU TEHNOLOGIJU I PROBLEME INFORMATIKE
SPISANIE ZA TEHNOLOGIJA NA SMETANJETO I PROBLEMI OD OBLASTA NA INFORMATIKATA

YU ISSN 0350 – 5596

LETNIK 3, 1979 – št. 3

VSEBINA

INFORMATICA 3/1979

SISTEMATSKA PREDSTROŽENOST U IMPLEMENTACIJI FORMALNO IZVEDENIH DETERMINISTIČKIH PROGRAMA

M. ZEČEVIĆ

UDK: 519.688

TEHNIČKA VOJNA AKADEMIJA, KoV JNA

Realizacija determinističkih algoritama formalno opisanih jednostavnim, rigorozno definisanim jezikom pomoću stvarnih industrijskih jezika za programiranje redovno je opterećena greškama. Pod predostrožnošću se, u ovom radu, podrazumeva osiguranje od opasnosti da realizovani program u doba izvršenja proizvodi netačan rezultat bez ikakvog upozorenja. Shvatajući programe pripremljene za kompilaciju kao skupove trasa ostvarivih tokom izvršenja programa, predlaže se metodologija (koja je u suštini dualna "invariant assertions" metoda) za analizu logičkih uslova pod kojima programi mogu da sadrže trase koje okončavaju netačnim rezultatom. Uvodi se naredba "abortne dijagnostike" čija je namena da se trase koje vode netačnom rezultatu prinudi da omanu. Naglašava se problem sinteze abortne dijagnostike i diskutuje se iskustvo u predstrožnim fortran implementacijama.

IMPLEMENTING ABSTRACT DETERMINISTIC PROGRAMS WITH SYSTEMATIC DEFENSIVNESS: Implementation of a deterministic algorithm abstractly described in a simple rigorously defined language, by means of a live industrial programming language is always prone to errors. Defensivness in this paper means safeguarding an implementation against production of incorrect results without any warrning during object time. Viewing compilable programs as sets of traces feasible during program execution a methodology (which is a dual to "invariant assertions"method) is proposed for analysis of logical conditions in which programs may have traces that could lead to incorrect result. A "safeguard"instruction is introduced whose purpose is to force the failure of traces that lead to incorrect result. Safeguard synthesis problem is emphasized and some experience in safeguarding fortran implementations is discussed.

UVOD

Teza ovog rada potekla je iz dvogodišnjih eksperimenata sa mogućnostima formalnog početnog izvođenja softvera. Ovakvo izvođenje treba da omogući, da se u implementaciji poslovi raspodeljuju prema složenosti, umesto po modulima - sa osnovnim ciljem, da se raspodela poslova što bolje prilagodi strukturi kompetentnosti kadra koji radi na proizvodnji određenog softvera. U ovakvom je prilazu jedan od ključnih problema nadzor nad greškama, koje se unose u fazi implementacije formalno kodiranih programa. Tokom eksperimenta, ovaj je problem napadan sa nekoliko heurističkih modela dijagnosticiranja implementacije. Predloženi rad sadrži formalno obrazloženje poslednjeg modela, koji je pokazao više nego zadovoljavajuće rezultate u implementiranju determinističkih programa.

Rad je referiran na XIV.Simp.Informatica,Bled,okt.1979

OSNOVA

Ako pod stanjem postupka računanja (mehanizma mašine) porazumevamo preslikavanje među skupom varijabli postupka i skupom vrednosti, možemo da smatramo da svaki postupak definiše klasu proračuna (procesa), koji se odvijaju u svom prostoru stanja.

Ako je G postupak a r predikat definisan u svim stanjima postupka G i r karakteriše željena konačna stanja postupka, onda sa wp (G, r) označavamo predikat koji karakteriše sva početna stanja postupka iz kojih se izvesno okončava u stanju u kome je r istinit. Ograničimo se na klasu postupaka koji se definišu jezikom u kome su:

(p-1) wp(skip, r) = r ,

(p-2) wp(abort, r) = false ,

(p-3) $wp(x := E, r) = r_E^x$,

elementarne naredbe. Ako P i Q definišu postupke a c i d su predikati definisani u svakom stanju, onda i sledeće kompozicije naredbi definišu postupke:

(p-4) wp(P;Q, r) = wp(P, wp(Q, r))
(komponovanje kalemljenjem - concatenation),

(p-5) wp(if(c→P ▽ d→Q)fi, r) =
(c or d) and
((c⇒wp(P, r))and(d⇒wp(Q, r))),
c and d = false!
(komponovanje selekcijom),

(p-6) wp(do(c→P)od, r) = (∃k)(k⩾0 and H_k(r)), gde je,
za k = 0, H_0 = r and non c, i
za k>0, H_k = wp(if(c→P)fi, H_{k-1}(r)) or H_0
(komponovanje iteriranjem).

Potpuna definicija ovog jezika može se naći u [1].

Sa druge strane, neka je U univerzum računanja i neka je s trasa (trag) jednog proračuna u tom univerzumu. Trasa se sastoji od konačne povorke dodeljivanja vrednosti varijablama postupka sa umetnutim predikatima, čiju istinitost postupak proverava tokom računanja. Dodeljivanja vrednosti i predikati na trasi, jednim se imenom zovu taktovi trase. Ishod proračuna predikata na trasi (u opštem) zavisi od početnog stanja koje se asocira trasi. Trasa, čiji se svi predikati kursom računanja pokažu istinitim, zove se ostvariva; trasa koja u kursu računanja sadrži bar jedan neistinit predikat naziva se neostvariva. Za proračun (u koga se upusti postupak) po neostvarivoj trasi kaže se da je omanuo u taktu u kome je izračunat prvi neistinit predikat. Postupak je deterministički ako se svakom početnom stanju može asocirati najviše jedna ostvariva trasa.

Trasa se može smatrati parcijalnom funkcijom koja preslikava početno stanje postupka u konačno. Trasa može da se posmatra i kao model "egzekucije" programa. Efekat odvijanja determinističkog programa pokrenutog iz početnog stanja x_0 u svemu je isti kao efekat niza dodeljivanja vrednosti na ostvarivoj trasi koja počinje u x_0 . U smislu ovoga modela, jasno je da svaka ostvariva trasa mora da bude konačna.

Neka je $r \subset U$ skup konačnih stanja od interesa a r predikat koji karakteriše ova stanja i neka je sar skup svih početnih stanja iz kojih postoji trasa s koja može da dovede do stanja koje pripada r. Tada je sar inverzija trase (kao funkcije) u odnosu na skup r. Dualno se definiše ser = $\overline{\text{sar}}$, tj. skup početnih stanja iz kojih nema ostvarive trase koja bi mogla okončati u stanju koje pripada komplementu skupa r. U radu će inverzija ser biti od dominantnog interesa.

Trase se mogu kalemiti jedna na drugu, što se označava sa

s;t (trasa t se kalemi na trasu s)

u inverziji u odnosu na skup konačnih stanja r, kalemljenje se odražava kao:

(s;t)er = se(ter) .

Ako su S i T proizvoljni skupovi trasa, kalemljenje T na S definisano je sa:

$$S;T = \bigcup_{s \in S} \bigcup_{t \in T} \{s;t\}$$

a njihova unija sa:

$$S \cup T = \{s \mid s \in S \text{ or } s \in T\}$$

Definicija a inverzije lako se proširuje na skup trasa:

$$Sar = \bigcup_{s \in S} (sar) \text{ , i dualno}$$

$$Ser = \overline{Sar} .$$

Skupovi trasa kao model implementacije detaljno su opisani u [2]. U istom radu izvode se osebine Sar i Ser kao i račun uslovne i totalne korektnosti.

Osobine a i e inverzija koristit će se u dokazu teze rada. Najzad, svakom skupu trasa odgovara jedan postupak računanja i svaki ovakav postupak može se definisati formalnim programom čiji je jezik uveden (p) definicijama. Dokaz ovoga stava nalazi se u [2]. Ako su P i Q naredbe iz jezika (p) a P^+ i Q^+ odgovarajući skupovi trasa, sledećom rekurzijom

definiše se uzajmnost modeliranja postupka programom odnosno skupom trasa:

(m-1) $\underline{skip}^+ = \{true\}$

(m-2) $\underline{abort}^+ = \{false\}$

(m-3) $(\ x := E\ )^+ = \{x = E\}$

(m-4) $(\ P;Q\ )^+ = (\ P^+;Q^+\ )$

(m-5) $(\ \underline{if}(\ c\to P \triangledown d\to Q\ )\underline{fi}\ )^+ = \{c\};P^+ \cup \{d\};Q^+$

(m-6) $(\ \underline{do}(\ c\to P)\underline{od}\ )^+ = \bigcup_{n\geqslant 0} S_n$, gde je

$S_0 = \{\underline{non}\ c\}$, $S_{n+1} = \{c\};P^+;S_n \cup S_0$

Primetiti da je podskup stanja Sar karakterisan predikatom wp(G,r), gde je $G^+=S$, ako je G deterministički program i da je podskup stanja Ser karakterisan predikatom wlp(G,r) bez obzira je li G deterministički ili ne.

TEZA

Trasa s je početni segment trase t ako t počinje s - om, tj.:

$$s \leqslant t = (\exists x)\ (\ s;x = t\ ).$$

Može se primetiti da segmentacija predstavlja parcijalno uređenje uz:

$(\forall t)(\ (\langle\rangle \leqslant t)\ \underline{and}\ t \leqslant t)$, gde je $\langle\rangle$ prazna trasa - što se može uzeti i kao semantička definicija prazne trase. Početni segment S skupa trasa S definisan je sa:

$$S^{\leqslant} = \{x \mid (\exists s)(\ s\in S\ \underline{and}\ x \leqslant s\ )\}\ .$$

Za trasu t se kaže da je nastavak od s u s ako je $s;t \in S$. Skup svih nastavaka trase s u S je

$$S/s = \{t \mid s;t \in S\}\ .$$

Definicija Segment programa G, odnosno njemu asocirani skup trasa S nazivaju se predostrožno zatvoreni ako:

a) ni jedna trasa iz S nema neprazni nastavak u S, tj.

(d-1) $(\forall s)((s\in S)\Rightarrow(S/s = \{\langle\rangle\})$ i

b) postoji jedinstven skup trasa $\{i\}$ nakalemljen na skup S:

(d-2) $S;\{i\} = \bigcup_{s\in S}\{s;\{i\}\}$.

Predikat (non i) naziva se abortna dijagnostika koja predostrožno zatvara segment G. U smislu gornje definicije, za svaki skup trasa koji ima osobinu (d-1), odnosno za pgm koji ga generiše kaže se da se mogu predostrožno zatvoriti. Lako je pokazati da se svaka forma (m-1) do (m-6) može predostrožno zatvoriti. Neka je G segment programa i neka je S njemu asocirani skup trasa koji uživa osobinu (d-1) i neka je segment predostrožno zatvoren abortnom dijagnostikom (non i). Neka je, dalje, G deterministički program, tj. za svako početno stanje postupka definisanog programom G postoji najviše jedna ostvariva trasa $s\in S$ koja vodi u nekakvo konačno stanje. Neka je T nedeterministički skup trasa zatvoren istom abortnom dijagnostikom, tj.:

$$(\forall t)(t\in T)\ \underline{and}\ (\ T/t = \{\langle\rangle\}\ ))\ \text{ i}$$

$$T;\{i\} = \bigcup_{t\in T}\{t;\{i\}\}\ .$$

Ako G generiše S rekurzijom (m) a T se generiše rekurzijom

$$\underline{skip}^+ = \{true\}$$

$$\underline{false}^+ = \{false\}$$

$$(x:=E)^+ = \{x:=E\} \cup \{x:=E_1\}$$

$$(P;Q)^+ = P^+;Q^+$$

$$(\underline{if}\ (c\to P \triangledown d\to Q)fi)^+ = \{c\};P^+ \cup \{d\};Q^+ \cup P_1^+$$

$$(\underline{do}\ c\to P\ \underline{od})^+ = \bigcup_{n\geqslant 0}(S_n \cup P_1^+)$$

gde je E_1 ma kakav izraz a P_1 ma kakva naredba, onda očigledno $S\subseteq T$ i za S i T se kaže da imaju istu strukturu.

Za svako početno stanje postupka može postojati nekoliko ostvarivih trasa koje vode u nekakvo konačno stanje; najviše jedna od njih može biti iz skupa S. Ako je $r\subset U$, skup konačnih stanja interesa, iz osobina inverzija a i e je:

$$(S\subseteq T)\Rightarrow(\ Sar\subseteq Tar\ )\ \text{, ali}$$

$$(S\subseteq T)\ (\ Ter\subseteq Ser\ )\ \text{ jer,}$$

zbog nedeterminizma skupa T, proračun pokrenut iz Tar može okončati stanjem koje pripada r, no može okončati i stanjem koje pripada komplementu od r. Predpostavimo, da je i odabrano tako, da za svako početno stanje i stiče vrednost false na svakoj trasi koja pripada

$$(\ T - S\ );\{i\}\ .$$

U tom slučaju je lako dokazati da je

$$(\ T;\{i\}\ )er = (\ S;\ \{i\}\ )er\ .$$

Šta više, kako je i jedinstveno, to dijagnostika non i jednoznačno određuje segment programa kome pripada trasa računanja koje je omanulo.

Egzistenciju predikata i sa osobinom

$$(\ T-S);\{i\}\ \underline{e}\ \emptyset$$

utvrđuje sledeća

Lema 1: Ako $S \subseteq T$ i $(Te \underline{non}\ r) \subseteq Ser$ $\underline{and}\ (Te \underline{non}\ r) \neq \emptyset$
onda egzistira invarijanta e takva da je

$$\forall x_0 \in (Te \underline{non}\ r)$$

trasa

$$\{e\}\ ;\ t\ ;\ \{e \Rightarrow \underline{non}\ r\}$$

ostvariva uvek kada je trasa $t \in T$ ostvariva
Dokaz leme (u drukčijoj notaciji) naveden je u [3].

Teorema 1: Za skupove trasa S i T i predikate r i e kao u prethodnoj lemi važi

(e-1) $[(T\text{-}S);\{\underline{non}\ e\}] \underline{e} \emptyset$ i

(e-2) $(T;\{\underline{non}\ e\}) \underline{e}\ r = Ser$

Teorema se lako dokazuje neposrednom primenom prethodne leme.
Neformalno rečeno - u problemu implementacije formalnog programa ne insistira se na totalnoj korektnosti implementacije; smatra se dovoljnim ako realizovani program ne bude proizvodio pogrešne rezultate bez ikakvog upozorenja. U predloženoj metodologiji se usvaja da skup trasa T generisan implementacijom sadrži skup S kao podskup a da se greške u implementaciji (ma kakav bio njihov uzrok) manifestuju kao nedeterminizam kojim se program pokrenut iz Ser upušta u proračun trasom koja pripada $(T - S)$.
Predostrožnost u implementaciji realizuje se zatvaranjem kritičnih segmenata abortnim dijagnostikama čime se osigurava da eventualna greška u segmentu neće ostati neotkrivena. Pored toga, svaka omaška ulovljena tokom izvršenja programa neposredno ukazuje na segment koji krije grešku. Uz nešto rutine u komponovanju i , sama abortna dijagnostika može da sadrži dobar deo analize greške.

PRIMER

Posmatrajnmo formalni program G :

```
(a-1)  read (j);
       do  2|j → j:= j div 2 ▽
           non (2 j) and(j≠1) → j:= 3*j + 1
       od ; end ,
```

gde je:
j - celobrojna varijabla,
2|j - znači " j je bez ostatka deljivo sa 2"
div - operator celobrojnog delenja.

S obzirom da je $(2|j)$ and non $(2|j)$ = false , program (a-1) je deterministički.
Ako je x_0 početna vrednost varijable j (koja stiče inicijalizacijom read (j)), onda formalni program (a-1) dodeljuje varijabli j niz vrednosti j_i i, ako okonča, utvrđuje istinitost predikata

$$(a\text{-}2)\quad r:\ (j_0 = x_0)\ \underline{and}\ (\exists i)(j_i = 2^n\ \underline{and}\ h \geq 0)$$

Lako je pokazati da je

$$wlp(G, r) = true,$$

odnosno, da je

$$Ser = U,$$

gde je S skup ostvarivih trasa programa G a U univerzum računanja. Isto tako, lako je uveriti se da

$$(a\text{-}3)\ (j_0 < 1) \Rightarrow \underline{non}\ wp(G, true),$$

odnosno, ni za jedno početno stanje karakterisano sa $j_0 < 1$ ne postoji ni jedna ostvariva trasa u S.
Međutim, wp(G, true) nije poznat, mada do sada eksperimentom nije naden $j_0 > 1$ za koji program G ne bi okončao. Ova analiza ilustruje čestu situaciju u praksi, naime, ne samo, da je nemoguće odrediti wp(G, true), nego nije moguće odrediti ni razumno slab q takav da $q \Rightarrow wp(G, true)$; s druge strane, empirijski je poznato, da je skup početnih stanja za koja program proizvodi tačan rezultat dovoljno velik da opravdava implementaciju. Invarijante koje zadovoljavaju (e-1), odnosno (e-2) određuju se na osnovu niza j_i . Osobina:

$$(a\text{-}4)\quad (j_0 = 2^h) \Leftrightarrow (h = n\ \underline{and}\ j_n = 1)$$

je očigledna. Osobina:

$$(a\text{-}5)\quad (j_0 \neq 2^n\ \underline{and}\ (\forall i) 2 \mid j_i) \rightarrow (\exists h)(3 \mid (j_i * 2^h - 1)\ \underline{and}\ \underline{non}(2 \mid (j_1 * 2^h - 1)/3))$$

je neposredna posledica algoritma formiranja niza. Nažalost invarijanta (a-4) je nepodesna za efikasno proveravanje zbog prisustva induktivno kvantifikovane varijable $\forall i$. U slučaju da je niz konačan ($(\exists n)(j_n = 1)$) invarijanta (a-4) implicira

$$(a\text{-}6)\quad (2|h)\ ,$$

na osnovu tautologije

$$(\forall h)(((h>1)\ \underline{and}\ 3|(2^h-1)) \Rightarrow 2|h)$$

koja se lako dokazuje indukcijom po h.
Slika 1. pokazuje fortran implementaciju formalnog programa G.
Podprogram ABNORMAL je tako kodiran da je naredba CALL ABNORMAL semantički ekvivalentna naredbi abort, dok parametri u pozivu indeksiraju segment izvornog programa koji je

generisao trasu koja je omanula taktom abort $^{+}$. Naredba (f-1) je model zaključka (a-3). Naredba (f-2) je posledica

LIMIT < 0 $\Rightarrow$ r= false,.

odnosno

LIMIT < 0 $\Rightarrow$ Se(non r)= ,

odnosno

LIMIT < 0 $\Rightarrow$ Ser = (J ZERO < 1),

što je već obuhvaćeno naredbom (f-1). Naredba (f-4) prepoznaje vrednost J ZERO koji ispunjava osobinom (a-4) u kom je slučaju trivijalno proveriti rezultat. Uslov u IF naredbi (f-5) je model od (a-6), negacija ovog uslova je abortna dijagnostika. Deo programa [35 CØNTINUE...END] predostrožno zatvara trase čiji bi se poslednji takt nalazio u non r. Na svakoj trasi koja sadrži takt

35 CONTINUE $^{+}$ = {true},

u istom taktu važi 2|J. Uslov u naredbi (f-6) je negacija (veoma) oslabljene osobine (a-5). Izvršenje naredbe CALL ABNORMAL (3HPW2 , 4) otkriva trasu koja ne bi smela da postoji. Naredba (f-7) detektuje početna stanja x_∞

$$(j_0 = x_\infty) \Rightarrow \underline{non}((\exists i)\ (\ j_i = 1)\ \underline{and}\ i < \text{LIMIT} + 2\ )$$

Na kraju treba primetiti da SUBROUTINE PW2, pored sve pažnje i sistematičnosti u odbrani od grešaka, ipak sadrži grešku koja može da dovede do proizvodnje pogrešnog rezultata a da se ni jedna abortna dijagnostika ne aktivira. Naime, program propušta da proveri nastajanje INTEGER OVERFLOW statusa nakon izvršenja naredbe " J = 3*J + 1". Tačan ishod ove naredbe u slučaju nastajanja OVERFLOW statusa zavisi od sistema na kome se program izvršava; no, u opštem, događaji izgledaju tako kao da je program prekinuo sa generisanjem članova niza sa $j_0 = x_0$ i počeo da generiše novi niz čiji je j_0 OVERFLOW vrednost od j. Očigledno je da novi niz može okončati sa $j_n = 1$ bez da aktivira i jedna od abortnih dijagnostika. (Između ostalog, ova analiza pokazuje šta se podrazumeva pod predpostavkom da implementacija uvodi širi skup trasa nego formalni program. Naša nezainteresovanost za tačan ishod naredbe J= 3 * J + 1 u OVERFLOW slučaju nalazi svoj formalni odraz u predpostavci da je skup trasa generisan implementacijom nedeterministički).

Razlog za ovo je sledeći: iako lema 1 garantuje egzistenciju invarijante sa osobinom (e-1) ona ne pruža nikakav efikasan putokaz za sintezu ove invarijante. Zato smo često prinuđeni, da usvojimo slabiji uslov za invarijantu odn. oštriji uslov za abortnu dijagnostiku. U našem primeru, prisustvo člana $(\forall i)$ u osobini (a-8) onemogućuje efikasno izračunavanje negacije ovog uslova - u stvari, trebalo bi izračunati čitav niz unatrag.

Naravoučenije je: nikakva logička analiza programa - mentalna ili automatska ne isključuje potrebu testiranja programa. Više o ovoj temi može se naći u [4].

ZAKLJUČAK

Rad napada probleme: kako biti sistematski defanzivan u implementaciji formalno izvedenih determinističkih programa. Ovaj problem je posebno delikatan u slučajevima kada je praktično nemoguće odrediti domen početnih uslova za koje je program korektan i/ili domen početnih uslova iz kojih program pouzdano okončava. Ovo je posebno slučaj u razvoju programskih jezika; čak i kod jednostavnog jezika za strogo specijalne nameme praktično je nemoguće odrediti koji sve tekstovi kompiliraju u korektne programe (ili se korektno interpretiraju). S druge strane, formalni programi se redovno izvode posredstvom minimalnog i dobro strukturiranog formalnog jezika, dok se implementacija radi, po pravilu na industrijskom jeziku sa priličnim razlikama u odnosu na ishodni jezik sintaksi i sematici i još gore, sa priličnim nejasnoćama u jednom i u drugom.

Prilaz koji je predložen u ovom radu u suštini je dualan metodologiji koja je posmatra pod nazivom "invariant assertions" koju je uveo Floyd [5] i koja je dosad primenjivana u raznim varijantama (vidi na pr. [4], [6]).

Umesto jezikom za programiranje, implementarni programi predstavljeni su skupovima svojih ostvarivih trasa. Definicije, notacije i osobine trasa i skupova trasa uzeti su onako kako ih je uveo Hoare u [2]. Ova notacija omogućava ne samo nezavisnost od jezika implementacije i mašine na kojoj se programi odvijaju, već omogućava da skup uključuje trase koje su generisane omaškama, nepredviđenim ishodima pojedinih naredbi, neumesnim "defaultima" pa čak i greškama hardvera.

Osim u više manjih projekata, metodologija predostrožnog zatvaranja kritičnih segmenata programa je primenjena u dva srednjo velika projekta - oba kodirana fortranom. Jedan projekat

je predstavljao interaktivni softver za formalni razvoj softverske specifikacije, drugi je bio softver za generisanje i ažuriranje baze podataka. I pored toga što metodologija ne pruža putokaz za sintezu abortne dijagnostike, već se to prepušta inventiji programera, u oba pomenuta projekta pronađena je samo jedna greška koja nije aktivirala abortnu dijagnostiku - greška u implementaciji sinhronizacije u monitoru poruka u interaktivnom softveru. Vreme potrebno za otkrivanje greške koja je pravi uzrok aktiviranja abortne dijagnostike obično je reda 5 min; sem u jednom slučaju, kada je grešku uzrokovao simbol komentara (C) u naredbi GØTØ koja je bila napisana od 25-te kolone - 3 čoveka utrošila su 4 sata na analizu, dok jedan slučajno nije primetio prokleto slovo C.

LITERATURA

1. Dijkstra,E.W. A Discipline of Programming. Prentice-Hall,Englewood Cliffs,. N.J.,1976.

2. Hoare, C.A.R. Some Properties of Predicate Transformers, Journal of the ACM 25(juli 1978), 461-480

3. Katz, S., Manna,Z. Logical Analysis of Programs, Communications of the ACM 19 (april 1976), 188-206

4. Manna, Z. Waldinger,R. The Logic of Computer Programming, IEEE trans. on Soft. Eng. SE-4 (maj 1978), 199-254

5. Floyd,R.W. Assigning Meaning to Programs. Proc. Symp. in Appl. Math., Vol 19, J.T. Schwartz (Ed.), Amer.Math.Soc.,Providence R.J., 1967, 19-32

6. Manna, Z., Waldinger,R. IS "Sometime" Sometimes Better than "Always"? Communications of the ACM 21 (februar 1978), 159-172


```
            SUBROUTINE  PW2 ( J ZERO, N, LIMIT )
            INTEGER  J ZERO, N, LIMIT
            INTEGER  J, HLF
      C
(f-1)       IF( J ZERO .LT. 1 ) CALL  A B N O R M A L ( 3HPW2, 1 )
(f-2)       IF( LIMIT  .LT. 0 ) CALL  A B N O R M A L ( 3HPW2, 2 )
      C
            N = 0
            J = J ZERO
            HLF = 0
      C
(f-3)   10  I F  (  J .EQ. 1  )  G  O  T  O    30
              IF(  N .GT. LIMIT )          GOTO 35
                IF( MOD( 2, J ) .EQ. 0 )   GOTO 20
                  J = 3*J + 1
                  N = N + 1
                  HLF = 0
          20      J = J/2
                  N = N + 1
                  HLF = HLF + 1
            G   O   T   O    10
      C
(f-4)   30  IF( HLF .EQ. N )              R E T U R N
(f-5)       IF( MOD( 2,HLF ) .EQ. 0 )     R E T U R N
              CALL  A B N O R M A L  ( 3HPW2, 3 )
      C
          35  CONTINUE
              DO  40  I = 1, HLF
                J = 2*J
          40  CONTINUE
(f-6)         IF( MOD( 3, (J-1) ) .NE. 0  .OR.  MOD( 2, ((J-1)/3) ) .EQ. 0 )
             $                              CALL  A B N O R M A L ( 3HPW2, 4 )
(f-7)         CALL  A B N O R M A L ( 3HPW2, 5 )
      C
      C     DUMMY
            RETURN
            END
```

Slika 1. Predostrožna realizacija algoritma (a-1)

MALI DISKOVNI OPERACIJSKI SISTEM

B. KASTELIC
M. KOVAČEVIĆ
D. NOVAK

UDK: 681.3.06

INSTITUT JOŽEF STEFAN, LJUBLJANA

Članek obravnava diskovni operacijski sistem za mikroračunalnik z enim gibkim diskom. Opisana je organizacija diskete in spomina, imenik ter delovanje samega diskovnega operacijskega sistema. Na koncu je objavljen program za zapisovanje in čitanje podatkov z diskete s pomočjo vhodno/izhodnih kanalov mikroračunalnika.

TINY DISK ORERATINC SYSTEM. The article presents tiny disk operatinc system for a microcomputer with one floppy disk. Disk organization, memory organization, directory and operation of the DOS is presented. The program for writinc and readinc data from disk by usinc input/output channels of a microcomputer is included.

1.UVOD

V članku je opisan diskovni operacijski sistem za mikroračunalnik s procesorjem M6800 in gibki disk, ki je krmiljen s krmilnikom 1771. Diskovni operacijski sistem omogoča zapisovanje in čitanje podatkov z diskete s pomočjo vhodno/izhodnih kanalov mikroračunalnika oziroma njegovega monitorja ter urejanje podatkov, ki so shranjeni v obliki nizov, na disketi. Vsak niz podatkov ima svoje ime, ki je zapisano v imeniku poleg podatkov o legi niza na disketi. Funkcije diskovnega operacijskega sistema omogočajo brisanje imena niza iz imenika, kompresiranje diskete, izpis imenika ter neposredni dostop do posameznih sektorjev diskete.

Diskovni operacijski sistem je zapisan z izjemo inicialnega nalagalnika, ki je v ROMU, na disketi in se po delih prenaša v mikroračunalniški spomin, kjer zasede le 1,5k zlogov spomina.

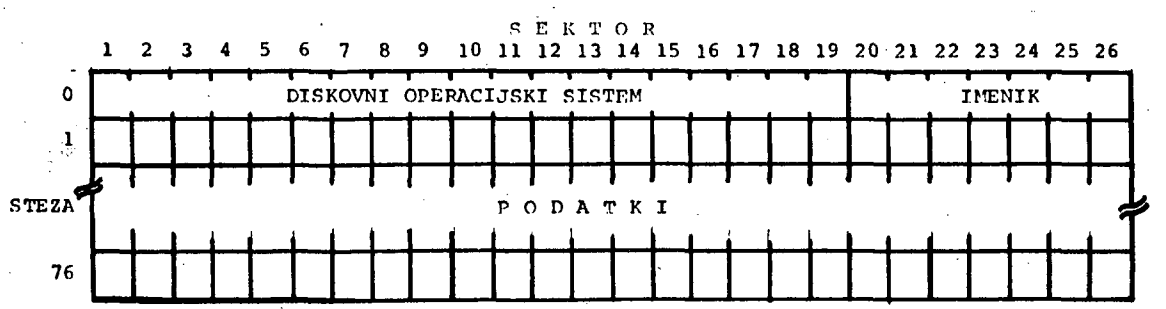
2.ORGANIZACIJA DISKETE

Diskovni operacijski sistem (DOS) uporablja enostranske diskete z normalno (enojno) gostoto zapisa podatkov. Vsaka disketa ima 77 stez, na vsaki stezi pa je 26 sektorjev z 128 zlogi podatkov (format IBM 3740). Ima torej 256k zlogov pomnilnega prostora za zapis podatkov.

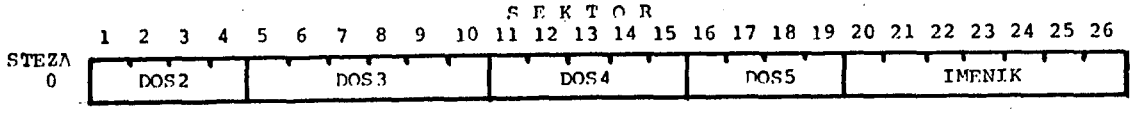
DOS uporablja ničto stezo diskete, za zapis podatkov pa je na razpolago ostalih 76 stez. Diskovni operacijski sistem je shranjen v ROMU (1/4k) in pa na prvih devetnajstih sektorjih ničte steze vsake diskete. Skupaj obsega približno 2,5k zlogov programa. Glede na nalaganje DOSA iz diskete v mikroračunalniški spomin ga lahko razdelimo na pet delov.

DOS1: Inicialni nalagalnik (v ROMU) inicializira krmilno vezje in gibki disk, postavi čitalno/pisalno glavo na stezo nič in prečita prvih deset sektorjev ničte steze v mikroračunalniški spomin.

DOS2: To je program naložen na prvih štirih sektorjih ničte steze. Sestoji se iz podprogramov, ki jih uporabljajo funkcije diskovnega operacijskega sistema in ostane vedno, kadar se uporablja disk v spomiminu. Prvih šest zlogov prvega sektorja je rezerviranih za ime diskete, nato pa si sledijo podprogrami:



SLIKA 1: Ničta steza diskete je uporabljena za diskovni operacijski sistem in za imenik. Vse ostale steze so na razpolago za zapis podatkov.



SLIKA 2: Lega posameznih delov diskovnega operacijskega sistema na ničti stezi diskete.

- čitanje sektorja
- zapisovanje sektorja
- čitanje več sektorjev
- zapisovanje več sektorjev
- izpis imena diskete
- iskanje imena v imeniku

Naslednji trije deli diskovnega operacijskega sistema se izmenično nalagajo v spomin v odvisnosti od zahtevane funkcije.

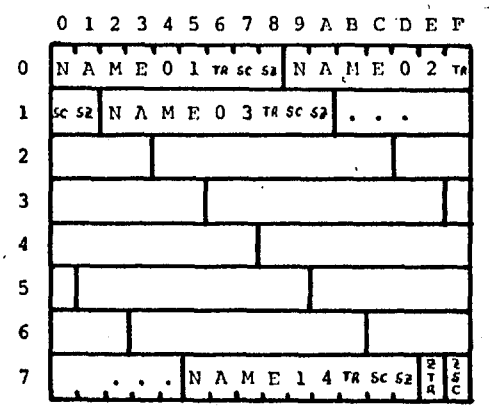
DOS3: Obsega programski vmesnik za navezavo na vhodni in izhodni kanal monitorja ter program za izbiro funkcij diskovnega operacijskega sistema. Ta del DOSA je tudi objavljen na koncu članka.

DOS4: Čitanje poljubnega števila sektorjev iz diskete na poljubno polje spomina, zapisovanje poljubnega polja iz spomina na poljubne sektorje diskete, izpis vsebine imenika, brisanje imena iz imenika.

DOS5: Program za kompresiranje diskete.

3. IMENIK

Imenik (directory) obsega zadnjeh sedem sektorjev ničte steze diskete. Vanj lahko zapišemo maksimalno 98 imen nizov ter osnovnih podatkov o njihovi legi na disketi. Vsi nizi so naloženi na disketi kontinuirano, zato so dovolj le tri informacije o njihovem položaju. Ime niza predstavlja šest alfanumeričnih ASCII znakov, sledijo pa jim trije zlogi: številka steze in številka sektorja začetka niza ter število zaporednih sektorjev.



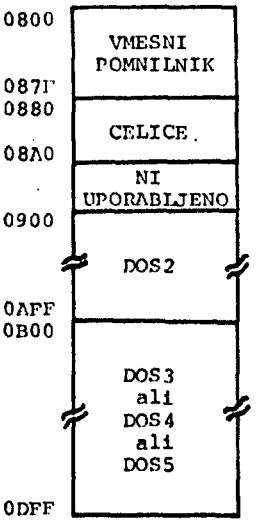
ZLOG	
0 - 5	IME NIZA
6	STEZA
7	SEKTOR
8	DOLŽINA
$7E	ZAČETNA STEZA / 00
$7F	ZAČETNI SEKTOR / 00

SLIKA 3: Sektor imenika

V vsakem sektorju imenika je 14 imen. Na koncu sektorja sta še dva zloga, ki imata vrednost 00, ko se imenik nadaljuje še na naslednjem sektorju. V zadnjem sektorju imenika pa predstavlja predzadnji zlog stezo in zadnji zlog sektor prvega praznega sektorja na disketi. Tukaj se začne zapisovati naslednji niz. Torej, če imamo na disketi zapisanih 22 nizov in je v imeniku prav toliko imen (ni zbrisanih nizov), bo prvih 14 imen v prvem sektorju imenika in ostalih osem v drugem sektorju, ki je tako tudi zadnji sektor imenika V zadnjem sektorju je ostalo prostora še za šest imen. Na tem prostoru so zapisani zlogi ničel in na koncu še začetna steza in sektor, kjer se bo začel nalagati naslednji niz. Dolžina niza je omejena na 255 ($FF) sektorjev, kar je približno 32k zlogov.

4. ORGANIZACIJA SPOMINA

Za delovanje diskovnega operacijskega sistema je potrebno 1,5k zlogov spomina. Na ta del spomina se nalaga diskovni operacijski sistem, poleg tega pa je tukaj še prostor za vmesni pomnilnik in celice za DOS spremenljivke. Področje spomina, kjer se nalaga DOS, je razdeljeno na dva dela. V prvem delu se vedno nahajajo podprogrami, ki jih uporablja večina funkcij diskovnega operacijskega sistema (DOS2), drugi del pa je polje kamor se naloži ustrezni del DOSA v odvisnosti od zahtevane funkcije.



SLIKA 4: Diskovni operacijski sistem uporablja 1,5k zlogov spomina.

5. DELOVANJE DISKOVNEGA OPERACIJSKEGA SISTEMA

Diskovni operacijski sistem lahko pokličemo na dva načina:

- pri prenašanju niza podatkov z diska ali na disk s pomočjo vhodno/izhodnih kanalov monitorja
- pri klicanju funkcij diskovnega operacijskega sistema

5.1.VHODNO/IZHODNI KANALI

Vsi podatki se zapisujejo na disketo in čitajo z nje preko vhodno/izhodnih kanalov. Možnost zapisovanja in čitanja s pomočjo vhodno/izhodnih kanalov ima že monitor, tako da se ob izbiri diskovnega izhodnega ali vhodnega kanala naveže le ustrezni kanal na monitor. Z izbiro kanala se najprej inicializira sam kanal, nato se začno prenašati podatki. Programski vmesnik ("driver") za gibki disk je zapisan na ničti stezi diskete in se pred inicializacijo kanala prenese v spomin.

Pri zapisovanju podatkov na disketo se najprej inicializira krmilno vezje in gibki disk, nato se prenese programski vmesnik v spomin ter se naveže izhodni kanal na monitor. Po vpisu imena, s katerim bo označen zapisovani niz, se začno podatki prenašati v vmesni pomnilnik. Gledano s strani centralne procesne enote se podatek, ki je shranjen v akumulatorju A, ob klicu izhodnega kanala prenese v vmesni pomnilnik. Za vsak podatek posebej, ki bo prenešen preko akumulatorja A, se mora poklicati izhodni kanal. Vmesni pomnilnik ima dolžino enega sektorja. Ko je poln, se zapiše na prvi prazen sektor diskete, nato se začne polniti od začetka. Po končanem zapisovanju se zapiše ime niza in njegova lega na disketi v imenik.

Pri čitanju se prav tako najprej izvede inicializacija, nato vpišemo ime niza, ki ga želimo prečitati in podatki se pričnejo v obratni smeri prenašati v spomin.

5.1.1.NAČINI ZAPISOVANJA

S pomočjo vhodno/izhodnih kanalov monitorja lahko zapisujemo podatke na dva načina. To sta ASCII zapis in binarni pretvorjen v ASCII zapis. Vsi nizi, na glede na način zapisovanja se končujejo s kontrolnim Z.

ASCII zapisi se uporabljajo za zapisovanje izvirnih nizov in vsebujejo le znake, ki jih lahko izpisujemo na konzoli.

Binarni pretvorjen v ASCII zapis predstavlja standardni Motorolin format. Vsak zlog se zapiše kot dva ASCII znaka ($0A → $30,$41)

5.2.FUNKCIJE DISKOVNEGA OPERACIJSKEGA SISTEMA

Za popolno delovanje diskovnega operacijskega sistema je potrbnih še nekaj funkcij, ki urejujejo zapisane nize in omogočajo neposredni dostop do posameznih sektorjev diskete. Kličemo jih tako, da najprej skočimo v diskovni operacijski sistem, nato pa izberemo črko, ki predstavlja ukaz za izvršitev določene funkcije.

D: Izpis imenika ("directory") na konzolo. Izpiše se ime niza, začetna steza, začetni sektor in dolžina vseh nizov, ki so zapisani v imeniku.

K: Brisanje ("kill") imena niza iz imenika. Ime niza in podatki o njegovi legi se izbrišejo iz imenika, oziroma se ne izpisujejo na konzoli pri izpisu imenika, ko je na mestu prvega ASCII znaka imena zapisano heksadecimalno število $80.

S: Kompresiranje ("squeeze") diskete. Vsi nizi so naloženi na disketi kontinuirano drug za drugim. Kadar določeni niz izbrišemo, ostane na njegovem mestu prazen prostor, na katerega ne moremo zapisovati novih nizov. Kompresiranje diskete je potrebno, da se ohranjeni nizi prepišejo drug za drugega in tako povečajo za dolžino zbrisanih nizov uporabni prostor na koncu diskete, kamor se lahko zapisujejo novi nizi. Med kompresiraanjem se tudi ustrezno spremenijo naslovi nizov v imeniku in dokončno izbrišejo imena že prej zbrisanih nizov.

R: Čitanje ("read") poljubnega števila sektorjev s poljubnega dela diskete na poljubno polje spomina.

W: Zapisovanje ("write") poljubnega polja spomina na poljubni del diskete.

H: Izpis vseh ukazov za funkcije diskovnega operacijskega sistema ("help").

6. GENERIRANJE DOSA NA DISKETO

Na vsako novo disketo je potrebno zapisati format IBM 3740 in na ničto stezo diskovni operacijski sistem. Za to sta potrebna še dva manjša programa in sicer eden za inicializacijo diskete (zapis IBM formata) in drugi za zapis diskovnega operacijskega sistema na disketo ter inicializacijo imenika. Ker ta dva programa relativno redko uporabljamo, poleg tega pa tudi nimata prostora na ničti stezi diskete, sta shranjena na posebni disketi, ki jo imenujemo tudi sistemska disketa.

7.SKLEP

Opisani diskovni operacijski sistem je prilagojen za uporabo enostranskih disket, na katerih se zapisujejo podatki z enojno gostoto. Z malo spremembami ga je mogoče prirediti za dvostranske diskete, tako da bi bila diskovni operacijski sistem in imenik še vedno na eni stezi, za podatke pa bi ostalo 154 stez.

Glavna prednost tega diskovnega operacijskega sistema je v tem, da zasede relativno malo spomina in je tako v prvi vrsti namenjn za manjše mikroračunalniške sisteme z majhno spominsko zmogljivostjo.

8.LITERATURA

1. B.Kastelic, M.Kovačević, A.Hadži: Krmilno vezje za gibki disk, INFORMATICA, 1/79, str.33-42

2. A.P.Železnikar: Vhodno/izhodni kanali mikroračunalnika, INFORMATICA, 4/78, str.13-25

```
                ********************************************************
                *                                                      *
                *     DISKOVNI OPERACIJSKI SISTEM  -  TRETJI DEL       *
                *                       18.9.79                        *
                *                                                      *
                ********************************************************


0800                    ORG     $0800
0800            BUFFER  RMB     128     VMESNI POMNILNIK

0880            ADDR1   RMB     2       SPOMIN ZA
0882            ADDR2   RMB     2       INDEKSNI REGISTER
0884            ADDR3   RMB     2
0886            ADDR4   RMB     2
0888            STP     RMB     2       SKLAD
088A            FILNAM  RMB     6       IME NIZA
0890            TRACK   RMB     1       NASLOV NIZA
0891            SEKT    RMB     1
0892            SIZE    RMB     1       DOLZINA NIZA V SEKTORJIH
0893            NAP     RMB     1       STEVEC PONOVITEV
0894            BEWR    RMB     2       ZACETNA LOKACIJA V POMNILNIKU
0896            NAMNO   RMB     1       STEVILO IMEN V IMENIKU
0897            WEND    RMB     1       KONEC PISANJA
0898            STBYT   RMB     1       STEVEC ZLOGOV
0899            ZTR     RMB     1       ZACETNA STEZA NASLEDNJEGA NIZA
089A            ZSC     RMB     1       ZACETNI SEKTOR NASLEDNJEGA NIZA
089B            BADR    RMB     2       ZACETNI NASLOV
089D            EADR    RMB     2       KONCNI NASLOV
089F            ZASC    RMB     1       ZASCITA
08A0            NOSEC   RMB     1       STEVILO SEKTORJEV


                *       MONITORSKI PODPROGRAMI

FC00            MON     EQU     $FC00   MONITOR
FE21            CRLF    EQU     $FE21   SKOK V NOVO VRSTICO
FDD4            OUTCH   EQU     $FDD4   IZPIS ASCII ZNAKA
FEE6            OUTST   EQU     $FEE6   IZPIS BESEDILA
FC03            INCH    EQU     $FC03   VPIS ASCII ZNAKA

001D            TOUCH   EQU     $001D   IZHODNI KANAL
001A            TINCH   EQU     $001A   VHODNI KANAL


                *       REGISTRI KRMILNIKA 1771

FBDC                    ORG     $FBDC
FBDC            CMDSTR  RMB     1       UKAZNI/STATUSNI REGISTER
```



```
FBDD            TRACKR  RMB     1       STEZNI REGISTER
FBDE            SECTR   RMB     1       SEKTORSKI REGISTER
FBDF            DATAR   RMB     1       PODATKOVNI REGISTER


                *       NASLOVI DOS PODPROGRAMOV

0907            RSC     EQU     $0907   CITANJE SEKTORJA
0976            WSC     EQU     $0976   ZAPIS SEKTORJA
0A4F            ISK     EQU     $0A4F   ISKANJE IMENA V IMENIKU
0A34            DSKIM   EQU     $0A34   IZPIS IMENA DISKETE


                *       NASLOVI FUNKCIJ DOSA

0AE2            READ    EQU     $0AE2   CITANJE SEKTORJEV
0AE8            WRITE   EQU     $0AE8   ZAPISOVANJE SEKTORJEV
0AEE            DRCT    EQU     $0AEE   IZPIS IMENIKA
0AF4            KILL    EQU     $0AF4   BRISANJE IMENA NIZA
0AFA            SQUE    EQU     $0AFA   KOMPRESIRANJE DISKETE


0B00                    ORG     $0B00
0B00 7E 0C E8           JMP     FUNK    FUNKCIJE DOSA
0B03 7E 0B 09           JMP     OUTINI  IZHODNI KANAL
0B06 7E 0C 11           JMP     ININI   VHODNI KANAL


                *       INICIALIZACIJA IZHODNEGA KANALA
                *               ZA GIBKI DISK

0B09 86 B9      OUTINI  LDAA    #$B9    SPROSTITEV ZASCITE
0B0B B7 08 9F           STAA    ZASC
0B0E BD 0C 78           JSR     NAME    VPIS IMENA
0B11 BD 0A 4F           JSR     ISK     ISKANJE IMENA
0B14 7D 08 92           TST     SIZE    IME OBSTAJA ?
0B17 27 08              BEQ     WRT     NE
0B19 CE 0D 63           LDX     #TEX2   DA
0B1C BD FE E6           JSR     OUTST   IZPIS BESEDILA
0B1F 20 E8              BRA     OUTINI  PONOVITEV
0B21 7F 08 97   WRT     CLR     WEND    ZACETEK ZAPISA
0B24 CE 08 00           LDX     #BUFFER
0B27 FF 08 94           STX     BEWR    ZACETNI NASLOV
0B2A FF 08 84           STX     ADDR3
0B2D 86 80              LDAA    #$80    DOLZINA SEKTORJA IN
0B2F B7 08 98           STAA    STBYT   VMESNEGA POMNILNIKA
0B32 B6 08 90           LDAA    TRACK   ZACETNA STEZA
0B35 B7 FB DF           STAA    DATAR
0B38 86 1E              LDAA    #$1E    ISKANJE STEZE
0B3A B7 FB DC           STAA    CMDSTR
0B3D 3E                 WAI
0B3E B6 08 91           LDAA    SEKT    ZACETNI SEKTOR
0B41 B7 FB DE           STAA    SECTR
```

```
0B44 CE 0B 4A          LDX   #OUTPUT NAVEZAVA IZHODNEGA KANALA
0B47 DF 1E             STX   TOUCH+1 NA MONITOR
0B49 39                RTS

               *   IZHODNI KANAL ZA GIBKI DISK

               *   PRENOS ZLOGA V VMESNI POMNILNIK
0B4A FF 08 86  OUTPUT STX   ADDR4
0B4D FE 08 84          LDX   ADDR3   NASLOV V VMES. POMNILNIKU
0B50 A7 00             STAA  0,X     ZAPIS V VMES. POMNILNIKU
0B52 08                INX
0B53 81 1A             CMPA  #$1A    KONEC (CNTR Z) ?
0B55 27 0C             BEQ   W1      DA
0B57 7A 08 98          DEC   STBYT   SEKTOR POLN ?
0B5A 27 0A             BEQ   W2      DA
0B5C FF 08 84          STX   ADDR3   NE
0B5F FE 08 86          LDX   ADDR4   NASTAVITEV IND.REG.
0B62 39                RTS           NASLEDNJI ZLOG

               *   ZAPIS VMESNEGA POMN. NA DISK
0B63 7C 08 97  W1      INC   WEND    ZADNJI SEKTOR
0B66 37        W2      PSHB
0B67 BD 09 76          JSR   WSC     ZAPIS SEKTORJA
0B6A 33                PULB
0B6B 7C 08 92          INC   SIZE    STEVILO ZAPISANIH SEKTORJEV
0B6E 26 09             BNE   W3      ZAPIS NI PREDOLG
0B70 CE 0D 72          LDX   #TEX4   PREDOLG
0B73 BD FE E6          JSR   OUTST   IZPIS BESEDILA
0B76 7E FC 00          JMP   MON
0B79 BD 0C C4  W3      JSR   INCR    NASLEDNJI SEKTOR
0B7C 7D 08 97          TST   WEND    KONEC ?
0B7F 26 0F             BNE   W5      DA
0B81 86 80             LDAA  #$80    ZACETEK NASLEDNJEGA SEKTORJA
0B83 B7 08 98          STAA  STBYT
0B86 CE 08 00          LDX   #BUFFER ZACETEK BUFFERJA
0B89 FF 08 84          STX   ADDR3
0B8C FE 08 86          LDX   ADDR4
0B8F 39                RTS           NASLEDNJI ZLOG

               *   ZAPIS IMENA TER NASLOVA NIZA V IMENIK
0B90 B6 FB DD  W5      LDAA  TRACKR
0B93 B7 08 99          STAA  ZTR     NOVA ZACETNA STEZA
0B96 B6 FB DE          LDAA  SECTR
0B99 B7 08 9A          STAA  ZSC     NOV ZACETNI SEKTOR
0B9C 86 0E             LDAA  #$0E    PREMIK NA STEZO 00
0B9E B7 FB DC          STAA  CMDSTR
0BA1 3E                WAI
0BA2 F6 08 92          LDAB  SIZE    SHRANITEV ST.SEKTORJEV
0BA5 37                PSHB
0BA6 BD 0A 4F          JSR   ISK     ISKANJE ZADNJEGA SEKTORJA IMENIKA
0BA9 33                PULB
0BAA F7 08 92          STAB  SIZE
0BAD CE 08 00          LDX   #BUFFER ZACETEK VMESNEGA POMNILNIKA
0BB0 5F                CLRB
0BB1 A6 00     V2      LDAA  0,X     ISKANJE PRAZNEGA MESTA
```



```
0BB3 27 1E             BEQ   V1      PRAZNO MESTO
0BB5 08                INX
0BB6 5C                INCB
0BB7 C1 7E             CMPB  #126    SEKTOR PREGLEDAN ?
0BB9 26 F6             BNE   V2      NE
0BBB B6 FB DE          LDAA  SECTR   DA
0BBE 81 1A             CMPA  #26     IMENIK POLN ?
0BC0 26 03             BNE   V3
0BC2 7E 0C D2  F1      JMP   FULL    IMENIK JE POLN
0BC5 6F 00     V3      CLR   0,X     OZNAKA ZA NADALJEVANJE IMENIKA
0BC7 6F 01             CLR   1,X
0BC9 BD 09 76          JSR   WSC     ZAPIS SEKTORJA
0BCC 7C FB DE          INC   SECTR
0BCF CE 08 00          LDX   #BUFFER VPIS IMENA V NASLEDNJI SEKTOR
0BD2 5F                CLRB
0BD3 FF 08 80  V1      STX   ADDR1   VPIS IMENA
0BD6 CE 08 8A          LDX   #FILNAM
0BD9 A6 00     V4      LDAA  0,X     CITANJE ZNAKA IZ SPOMINA
0BDB 8C 08 93          CPX   #NAP    VPISOVANJE KONCANO ?
0BDE 27 13             BEQ   V10     DA
0BE0 08                INX           NE
0BE1 FF 08 82          STX   ADDR2
0BE4 FE 08 80          LDX   ADDR1
0BE7 A7 00             STAA  0,X     VPIS ZNAKA V IMENIK
0BE9 08                INX
0BEA 5C                INCB
0BEB FF 08 80          STX   ADDR1
0BEE FE 08 82          LDX   ADDR2
0BF1 20 E6             BRA   V4      NASLEDNJI SEKTOR
0BF3 FE 08 80  V10     LDX   ADDR1
0BF6 C1 7E             CMPB  #126    SEKTOR POLN ?
0BF8 27 06             BEQ   V5      DA
0BFA 6F 00             CLR   0,X     ZAPIS NICEL DO KONCA SEKTORJA
0BFC 08                INX
0BFD 5C                INCB
0BFE 20 F6             BRA   V6
0C00 B6 08 99  V5      LDAA  ZTR
0C03 A7 00             STAA  0,X     VPIS ZACETNE STEZE NA
0C05 B6 08 9A          LDAA  ZSC     KONCU IMENIKA
0C08 A7 01             STAA  1,X     VPIS ZACETNEGA SEKTORJA
0C0A BD 09 76          JSR   WSC     ZAPIS SEKTORJA NA DISK
0C0D 7F 08 9F          CLR   ZASC    ZASCITA
0C10 39                RTS

               *   INICIALIZACIJA VHODNEGA KANALA
               *       ZA GIBKI DISK

0C11 BD 0C 78  ININI   JSR   NAME    VPIS IMENA
0C14 BD 0A 4F          JSR   ISK     ISKANJE IMENA
0C17 7D 08 92          TST   SIZE    OBSTAJA ?
0C1A 26 08             BNE   R1      DA
0C1C CE 0D 50          LDX   #TEX1   NE
0C1F BD FE E6          JSR   OUTST   IZPIS BESEDILA
0C22 20 ED             BRA   ININI
0C24 CE 08 00  R1      LDX   #BUFFER
0C27 FF 08 94          STX   BEWR    ZACETEK VMESNEGA POMNILNIKA
```

```
0C2A FF 08 84          STX    ADDR3
0C2D B6 08 90          LDAA   TRACK
0C30 B7 FB DF          STAA   DATAR    ZACETNA STEZA
0C33 86 1E             LDAA   #$1E
0C35 B7 FB DC          STAA   CMDSTR   ISKANJE STEZE
0C38 3E                WAI
0C39 B6 08 91          LDAA   SEKT     NASTAVITEV SEKTORJA
0C3C B7 FB DE          STAA   SECTR
0C3F BD 09 07          JSR    RSC      CITANJE SEKTORJA
0C42 CE 0C 48          LDX    #INPUT   NAVEZAVA VHODNEGA KANALA
0C45 DF 1B             STX    TINCH+1  NA MONITOR
0C47 39                RTS

                *      VHODNI KANAL ZA GIBKI DISK

0C48 FF 08 86  INPUT   STX    ADDR4    SHRANITEV IND.REG.
0C4B FE 08 84          LDX    ADDR3
0C4E 8C 08 80          CPX    #BUFFER+128 VMESNI POMNILNIK PRECITAN ?
0C51 26 1B             BNE    INPUT1   NE
0C53 B6 FB DE          LDAA   SECTR    DA
0C56 81 1A             CMPA   #26      CELA STEZA PRECITANA ?
0C58 26 09             BNE    INPUT2   NE
0C5A 7F FB DE          CLR    SECTR    DA
0C5D 86 5E             LDAA   #$5E     PREMIK NA NASLEDNJO STEZO
0C5F B7 FB DC          STAA   CMDSTR
0C62 3E                WAI
0C63 7C FB DE  INPUT2  INC    SECTR
0C66 37                PSHB
0C67 BD 09 07          JSR    RSC      CITANJE SEKTORJA
0C6A 33                PULB
0C6B CE 08 00          LDX    #BUFFER  NASTAVITEV IND.REG.
0C6E A6 00     INPUT1  LDAA   0,X      PODATEK V ACC A
0C70 08                INX
0C71 FF 08 84          STX    ADDR3
0C74 FE 08 86          LDX    ADDR4
0C77 39                RTS

                *      VPIS IMENA

0C78 BD 0A 34  NAME    JSR    DSKIM    IZPIS IMENA DISKA
0C7B CE 0D 8F          LDX    #TEX5    NASLOV BESEDILA
0C7E BD FE E6          JSR    OUTST    IZPIS BESEDILA
0C81 CE 08 8A          LDX    #FILNAM
0C84 BD FC 03          JSR    INCH     VPIS PRVEGA ZNAKA
0C87 8D 2C             BSR    TEST     PIKA ALI ESC ?
0C89 A7 00             STAA   0,X
0C8B 08                INX
0C8C C6 04             LDAB   #4       NASLEDNJE STIRJE ZNAKI
0C8E BD FC 03  NAM2    JSR    INCH
0C91 81 2E             CMPA   #'.      PIKA ?
0C93 27 0D             BEQ    NAM1     DA
0C95 A7 00             STAA   0,X      NE
0C97 08                INX
0C98 5A                DECB
0C99 26 F3             BNE    NAM2
```



```
0C9B 86 2E             LDAA   #'.
0C9D BD FD D4          JSR    OUTCH    IZPIS PIKE
0CA0 20 08             BRA    NAM3
0CA2 86 20     NAM1    LDAA   #$20     ZAPOLNITEV POROSTORA S PRESLEDKI
0CA4 A7 00             STAA   0,X
0CA6 08                INX
0CA7 5A                DECB
0CA8 26 F8             BNE    NAM1
0CAA BD FC 03  NAM3    JSR    INCH     SESTI ZNAK
0CAD 8D 06             BSR    TEST     PIKA ALI ESC ?
0CAF A7 00             STAA   0,X
0CB1 39                RTS
0CB2 7E FC 00  PIKA    JMP    MON

                *      TESTIRANJE PIKE IN ESC

0CB5 81 2E     TEST    CMPA   #'.      PIKA ?
0CB7 26 03             BNE    TEST1    NE
0CB9 7E FC 00          JMP    MON      DA
0CBC 81 1B     TEST1   CMPA   #$1B     ESC ?
0CBE 26 03             BNE    TEST2    NE
0CC0 7E 0C E8          JMP    FUNK     DA
0CC3 39        TEST2   RTS

                *      NASLEDNJI SEKTOR

0CC4 B6 FB DE  INCR    LDAA   SECTR
0CC7 81 1A             CMPA   #26      STEZA POLNA ?
0CC9 26 19             BNE    W4       NE
0CCB B6 FB DD          LDAA   TRACKR    DA
0CCE 81 4C             CMPA   #76      DISK POLN ?
0CD0 26 09             BNE    W6       NE
0CD2 CE 0D 82  FULL    LDX    #TEX3    DA
0CD5 BD FE E6          JSR    OUTST    IZPIS BESEDILA
0CD8 7E FC 00          JMP    MON
0CDB 7F FB DE  W6      CLR    SECTR
0CDE 86 5E             LDAA   #$5E     PREMIK NA NASLEDNJO STEZO
0CE0 B7 FB DC          STAA   CMDSTR
0CE3 3E                WAI
0CE4 7C FB DE  W4      INC    SECTR    NASTAVITEV SEKTORJA
0CE7 39                RTS

                *      FUNKCIJE DISKOVNEGA OPERACIJSKEGA SISTEMA

0CE8 BD 0A 34  FUNK    JSR    DSKIM    IZPIS IMENA DISKETE
0CEB BD FE 21  CMD3    JSR    CRLF
0CEE BD FE 21          JSR    CRLF
0CF1 86 3D     CMD5    LDAA   #'=      IZPIS ENACAJA
0CF3 BD FD D4          JSR    OUTCH
0CF6 BD FC 03          JSR    INCH     VPIS UKAZA
0CF9 81 2E             CMPA   #'.      PIKA ?
0CFB 26 03             BNE    CMD4     NE
```

```
0CFD 7E FC 00          JMP    MON      SKOK V MONITOR
0D00 84 DF     CMD4    ANDA   #$DF     MASKA
0D02 CE 0D 1F          LDX    #TABLE
0D05 A1 00     CMD1    CMPA   0,X      ISKANJE USTREZNEGA UKAZA
0D07 27 12             BEQ    CMD2     UKAZ NAJDEN
0D09 08                INX
0D0A 08                INX
0D0B 08                INX
0D0C 8C 0D 34          CPX    #TBLEND  KONEC TABELE ?
0D0F 26 F4             BNE    CMD1     NE
0D11 86 3F             LDAA   #'?      DA
0D13 BD FD D4          JSR    OUTCH    IZPIS VPRASAJA
0D16 BD FE 21          JSR    CRLF     NOVA VRSTICA
0D19 20 D6             BRA    CMD5     NA ZACETEK
0D1B EE 01     CMD2    LDX    1,X
0D1D 6E 00             JMP    0,X      IZVAJANJE UKAZA

               *       UKAZNA TABELA

0D1F 52        TABLE   FCC    "R"
0D20 0A E2             FDB    READ     CITANJE SEKTORJEV
0D22 57                FCC    "W"
0D23 0A E8             FDB    WRITE    ZAPISOVANJE SEKTORJEV
0D25 44                FCC    "D"
0D26 0A EE             FDB    DRCT     IZPIS IMENIKA
0D28 4B                FCC    "K"
0D29 0A F4             FDB    KILL     BRISANJE IMENA NIZA
0D2B 53                FCC    "S"
0D2C 0A FA             FDB    SQUE     KOMPRESIRANJE DISKETE
0D2E 48                FCC    "H"
0D2F 0D 35             FDB    HELP     IZPIS DOS UKAZOV
0D31 4D                FCC    "M"
0D32 FC 00             FDB    MON      MONITOR
0D34 00        TBLEND  FCB    0

               *       IZPIS DOS UKAZOV

0D35 CE 0D 9D  HELP    LDX    #TEX6
0D38 BD FE E6          JSR    OUTST    IZPIS BESEDILA
0D3B CE 0D 1F          LDX    #TABLE
0D3E A6 00     HELP1   LDAA   0,X      UKAZ
0D40 BD FD D4          JSR    OUTCH    IZPIS UKAZA
0D43 BD FE 21          JSR    CRLF     NOVA VRSTICA
0D46 08                INX
0D47 08                INX
0D48 08                INX
0D49 8C 0D 34          CPX    #TBLEND  KONEC TABELE ?
0D4C 26 F0             BNE    HELP1    NE
0D4E 20 9B             BRA    CMD3     DA


0D50 0D        TEX1    FCB    $D,$A
0D52 4E                FCC    "NAME NOT FOUND"
0D60 0A                FCB    $A,$D,4
0D63 0D        TEX2    FCB    $D,$A
0D65 4E                FCC    "NAME ERROR"
```



```
0D6F 0D                FCB    $D,$A,4
0D72 0A        TEX4    FCB    $A,$D
0D74 54                FCC    "TOO LONG FILE"
0D81 04                FCB    4
0D82 44        TEX3    FCC    "DISK IS FULL"
0D8E 04                FCB    4
0D8F 0D        TEX5    FCB    $D,$A
0D91 20                FCC    "   NAME : "
0D9C 04                FCB    4
0D9D 0D        TEX6    FCB    $D,$A,$A
0DA0 44                FCC    "DOS COMMANDS :"
0DAE 0D                FCB    $D,$A,$A,4
                       END
      NO ERRORS DETECTED


   SYMBOL TABLE:
ADDR1   0880   ADDR2   0882   ADDR3   0884   ADDR4   0886
BADR    089B   BEWR    0894   BUFFER  0800   CMD1    0D05
CMD2    0D1B   CMD3    0CEB   CMD4    0D00   CMD5    0CF1
CMDSTR  FBDC   CRLF    FE21   DATAR   FBDF   DRCT    0AEE
DSKIM   0A34   EADR    089D   F1      0BC2   FILNAM  088A
FULL    0CD2   FUNK    0CE8   HELP    0D35   HELP1   0D3E
INCH    FC03   INCR    0CC4   ININI   0C11   INPUT   0C48
INPUT1  0C6E   INPUT2  0C63   ISK     0A4F   KILL    0AF4
MON     FC00   NAM1    0CA2   NAM2    0C8E   NAM3    0CAA
NAME    0C78   NAMNO   0896   NAP     0893   NOSEC   08A0
OUTCH   FDD4   OUTINI  0B09   OUTPUT  0B4A   OUTST   FEE6
PIKA    0CB2   R1      0C24   READ    0AE2   RSC     0907
SECTR   FBDE   SEKT    0891   SIZE    0892   SQUE    0AFA
STBYT   0898   STP     0888   TABLE   0D1F   TBLEND  0D34
TEST    0CB5   TEST1   0CBC   TEST2   0CC3   TEX1    0D50
TEX2    0D63   TEX3    0D82   TEX4    0D72   TEX5    0D8F
TEX6    0D9D   TINCH   001A   TOUCH   001D   TRACK   0890
TRACKR  FBDD   V1      0BD3   V10     0BF3   V2      0BB1
V3      0BC5   V4      0BD9   V5      0C00   V6      0BF6
W1      0B63   W2      0B66   W3      0B79   W4      0CE4
W5      0B90   W6      0CDB   WEND    0897   WRITE   0AE8
WRT     0B21   WSC     0976   ZASC    089F   ZSC     089A
ZTR     0899

ROSSANA EDITOR
```

INFORMATICA 3/1979

A HYBRID ALGORITHM FOR FINDING ROOTS

D. B. POPOVSKI

UDK: 681.3 : 51

DEPARTMENT OF ENGINEERING,
UNIVERSITY OF BITOLA, BITOLA

In this paper a hybrid algorithm for finding a bracketed root is presented in which Ostrowski´s two-point method and bisection method are used. A FORTRAN IV subroutine and a few numerical examples are given.

JEDAN HIBRIDAN ALGORITAM ZA NALAŽENJE KORENA. U radu je prezentiran jedan hibridan algoritam za nalaženje izolovanog korena u kome se koriste dvotačkasta metoda Ostrovskog i metoda polovljenja. Dat je FORTRAN IV potprogram i nekoliko numeričkih primera.

INTRODUCTION

Ostrowski[1] proposed a method for solving nonlinear equations of the form

$$y(x)=0 \tag{1}$$

in which he uses two-point hyperbolic approximation. In his method, a linear fraction

$$f(x)=\frac{x-a}{bx+c} \tag{2}$$

is fitted to y(x) at three points, two of which are coincident. Thus, a step in the iteration consists of matching y and f at the points x_0 and x_1 , and y´ and f´ at the point x_1 only. The next approximation is given by the zero of (2) i.e. $x_2=a$. Ostrowski showed that the asymptotic convergence rate of the process is 2.414. Since each step requires the evaluation of y and y´ , the efficiency index is $2.414^{1/2}=1.554$. It is better then Newton´s method which has an asymptotic convergence of order 2 and efficiency index $2^{1/2}=1.414$. Ostrowski´s two-point method does not have a guaranteed convergence. In this paper a hybrid algorithm with guaranteed convergence is presented in which the bisection method is used when Ostrowski´s method breaks down.

ALGORITHM

Suppose that a real root of (1) has been bracketed by initial approximations x_0 and x_1 . Thus, $\text{signy}_0 \neq \text{signy}_1$. The bracketed root of (1) may be found by a hybrid algorithm summarized like this:

Calculate y_0 , y_1 and y_1' . Set $x_B=x_0$.

(a) Find x_2 by the Ostrowski´s step

$$x_2=x_1+\frac{(y_1-y_0)y_1(x_1-x_0)}{y_1'y_0(x_1-x_0)-(y_1-y_0)y_1} \tag{3}$$

provided this point is between x_1 and x_B . Otherwise get x_2 by the bisection step

$$x_2=x_1+(x_B-x_1)*0.5 \ . \tag{4}$$

Calculate y_2 and y_2' . If $\text{signy}_2 \neq \text{signy}_1$, set $x_B=x_1$. In any case, replace (x_0,y_0) by (x_1,y_1), (x_1,y_1) by (x_2,y_2) and y_1' by y_2' . Return to (a).

This means that Ostrowski´s step is taken whenever possible, and only the latest two iterates are used even though they may be on the same side of the root. Also one other iterate is retained - the opposite bracketing point x_B (x_B may, in addition, be one of the current iterates). The current point x_1 is used with x_B

in the bisection step whenever the Ostrowski´s step fails to fall within the interval bounded by x_1 and x_B .

SUBROUTINE

The following is a FORTRAN IV realization of the algorithm described in the previous section.

```
      SUBROUTINE O(S,A,B,T,R,M,N)
      N=0
      I=2
      C=A
      D=B
      F=D-C
      CALL S(F,G,C,1)
      CALL S(G,H,D,2)
      IF(F)1,19,2
    1 IF(G)17,18,5
    2 IF(G)5,18,17
    3 C=D
    4 D=P
    5 P=(G-F)*G
      F=H*F*E-P
      IF(F)6,9,6
    6 E=P*E/F
      P=D+E
      IF(P-C)7,9,8
    7 IF(E)9,9,10
    8 IF(E)10,9,9
    9 E=(C-D)*0.5
      P=D+E
   10 IF(ABS(P)*T-ABS(E))11,16,16
   11 IF(I-M)12,15,15
   12 F=G
      CALL S(G,H,P,2)
      I=I+1
      IF(G)13,16,14
   13 IF(F)4,18,3
   14 IF(F)3,18,4
   15 N=1
   16 R=P
      RETURN
   17 N=2
   18 R=D
      RETURN
   19 R=C
      RETURN
      END
```

Description of parameters:

- S - Name of the external subroutine used. Its parameter list must be F,D,X,J. When J=1, it computes, for given argument X, only the function value F. When J=2 it computes F and the derivative D.
- A - Initial approximation x_0 .
- B - Initial approximation x_1 .
- T - Tolerance for the relative error.
- R - Root of the equation y(x)=0.
- M - Maximum number of iterations permitted.
- N - Error parameter coded as follows:
  - N=0 - No error.
  - N=1 - No convergence after M iterations.
  - N=2 - Basic assumption $\text{sign}y_0 \neq \text{sign}y_1$ is not satisfied.

Remarks:

The procedure assumes that function values at initial approximations A and B do not have same sign. If this basic assumption is not satisfied, the procedure is bypassed and gives the error message N=2.

The external subroutine S(F,D,X,J) must be furnished by the user. The following is an example (example 1 from table I).

```
      SUBROUTINE S(F,D,X,J)
      GO TO (2,1),J
    1 D=(X*3.-2.)*X
    2 F=(X-1.)*X*X-1.
      RETURN
      END
```

CERTIFICATION

Subroutine O was tested on an IBM 1130 computer using floating-point arithmetic with 32-bit mantissa (extended precision) for the examples given in table I.

Table I. Numerical examples: A=5, B=0, $T=10^{-4}$, M=20.

	y(x)	R
1	$x^2(x-1)-1$	1.46557123
2	$(x+1)^2(x-2)x-1$	2.05230034
3	$x^3(x^2-1)-10$	1.72099577
4	$x^4(x^2-1)-1$	1.21060779
5	$x^6(x-1)-1$	1.25542287

REFERENCE

1. A.M.Ostrowski, Solution of Equations and Systems of Equations, Academic Press, New York and London, 1960.

INFORMATICA 3/1979

PRIMJENA MIKROPROCESORA MPR-52B NA JEDAN PRIMJER PRAĆENJA CILJEVA U REALNOM VREMENU

MIGDAT HODŽIĆ
SVETO VRHOVAC

UDK: 681.3

SOUR „RUDI ČAJAVEC", RO PE, OOUR RAČUNARSKO-RADARSKA TEHNIKA, BANJA LUKA

U radu je opisana primjena mikroprocesora MPR-52B(Rudi Čajavec,Banja Luka) na problem praćenja ciljeva u vazduhu u realnom vremenu.Model cilja i model mjerenja dati su u diskretnom obliku , sa intervalom odbira T.Poredili smo dva tipa filtara za praćenje: Kalmanov i Wienerov filtar. Performanse filtara testirane su Monte-Karlo simulacijom na nekoliko primjera test-putanja vazdušnih ciljeva.

AN APPLICATION OF THE MICROPROCESSOR MPR-52B TO A REAL-TIME AIR TARGET TRACKING PROBLEM:In this article,we have described an application of the microprocessor MPR-52B (Rudi Čajavec,Banja Luka) to a real-time air target tracking problem.Target and measurement models are given in a discrete form,with sampling interval T.We have compared two tracking filters:Kalman and Wiener-filter . Filters performances have been tested on the several air target test-paths,with a Monte-Carlo simulation.

1.Uvod

Potrebno je pratiti vazdušni cilj u realnom vremenu,na osnovu zašumljenih mjerenja o cilju koja daje radarski senzor,u vremenskim razmacima T.U tom smislu definišu se modeli stanja cilja i mjerenja,kao i jednačine stohastičkog filtra za praćenje.Model stanja cilja se definiše shodno zahtjevanoj tačnosti praćenja i vremenu u kome filtar "obradi" mjerenja.Ova su dva zahtjeva oprečna i kod realizacije sistema za praćenje,potrebno je napraviti odredjeni kompromis izmedju tačnosti i brzine rada filtra za praćenje.Tačnost se definiše u zahtjevima sistema za praćenje a brzina rada prvenstveno zavisi od odabra - nog filtra i brzine rada procesora podataka. U našem slučaju će to biti mikroprocesor MPR-52B. U odjeljku 2. data je postavka problema,model stanja i mjerenja,te jednačine filtara za praćenje. Odjeljak 3. opisuje dijagram toka programa za praćenje napisanog u Asembleru MPR-52B kao i konfiguraciju sistema za testiranje programa.U odjeljku 4. date su karakteristike asembler-programa i jedan konkretan primjer.Zaključak je u odjeljku 5., a literatura u odjeljku 6. U Dodatku su date osnovne karakteristike mikroprocesora MPR-52B.

2.Postavka problema

2.1 Modeli stanja cilja i mjerenja

Model stanja cilja koji se prati opisan je diferencnom jednačinom:

$$x(k+1)=\Phi(k)x(k)+G(k)w(k), \qquad (1)$$

dok su mjerenja o cilju data sa:

$$y(k)=H(k)x(k)+v(k), \qquad (2)$$

gdje su:

- $x(k)$=vektor stanja cilja u momentu kT,
- $w(k)$=vektor šuma na stanje cilja,
- $y(k)$=vektor mjerenja o cilju,
- $v(k)$=vektor šuma na mjerenje,
- $\Phi(k)$=matrica prelaza stanja,
- $G(k)$,$H(k)$=uredjivačke matrice.

Tipičan i najčešće korišten model stanja cilja i mjerenja je opisan sa:

$$x(k)=(D(k),\dot{D}(k),AZ(k),\dot{AZ}(k))^T,$$

$$y(k)=(MD(k),MAZ(k))^T,$$

$$w(k)=(w_D(k),w_{AZ}(k))^T,$$

$$v(k)=(v_{MD}(k),v_{MAZ}(k))^T,$$

$$\Phi(k)=\begin{bmatrix}1 & T & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & T\\ 0 & 0 & 0 & 1\end{bmatrix}=\Phi,$$

$$G(k)=\begin{bmatrix}0 & 0\\ 1 & 0\\ 0 & 0\\ 0 & 1\end{bmatrix}=G,$$

$$H(k)=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}=H,$$

gdje su:

- $D(k)$=daljina do cilja u momentu kT,
- $\dot{D}(k)$=promjena daljine izmedju momenata (k-1)T i kT,
- $AZ(k)$=azimut cilja u momentu kT,
- $\dot{AZ}(k)$=promjena azimuta izmedju momenata (k-1)T i kT,
- $MD(k)$=mjerena daljina u momentu kT,
- $MAZ(k)$=mjereni azimut u momentu kT,
- $w_D(k)$=šum ubrzanja po daljini u mom. kT,
- $w_{AZ}(k)$=šum ubrzanja po azimutu u mom. kT,
- $v_{MD}(k)$=šum na mjerenje daljine u mom. kT,
- $v_{MAZ}(k)$=šum na mjerenje azimuta u mom. kT.

Statistika šumova w(k) i v(k) data je preko očekivanih vrijednosti i kovarijansnih matrica:

$$E(w(k))=E(v(k))=0, \text{ za svako } k$$

$$E(w(k)w^T(n))=E(v(k)v^T(n))=[0], \text{za } k\neq n$$

$$E(w(k)w^T(k))=\mathrm{diag}(\sigma_1^2,\sigma_2^2)=[Q], \text{za svako } k$$

$$E(v(k)v^T(k))=\mathrm{diag}(\sigma_D^2,\sigma_{AZ}^2)=[R], \text{za svako } k.$$

Rad je referiran na XIV.Simp.Informatica,Bled,okt.1979

gdje su:

σ_1^2=varijansa ubrzanja po daljini ,

σ_2^2=varijansa ubrzanja po azimutu ,

σ_D^2=varijansa greške mjerenja daljine,

σ_{AZ}^2=varijansa greške mjerenja azimuta.

Dati model podrazumijeva ubrzanja cilja čije srednje vrijeme trajanja ne prelazi veličinu vremena odbira T.U tom slučaju,šumove w(k) i v(k) možemo aproksimirati bijelim šumom,sa Gausovkom raspodjelom vjerovatnoće. U suprotnom bi ubrzanja izmedju momenata (k-1)T i kT bila u korelaciji,te ih ne bi mogli aproksimirati bijelim šumom.
Veličine varijansi grešaka mjerenja daljine i azimuta,kao i vrijeme T,poznati su za svaki radarski senzor,dok se varijanse ubrzanja po daljini i po azimutu modeliraju jednačinama:

$$\sigma_1^2=A^2T^2(1+4P_1-P_2)/3 \qquad (3)$$

$$\sigma_2^2=\sigma_1^2/D^2(k) \qquad (4)$$

gdje su:

A=maksimalno ubrzanje cilja ,
P_1=vjerovatnoća da će cilj manevrisati sa maksimalnim ubrzanjem A ,
P_2=vjerovatnoća da cilj neće manevrisati.

Napomenimo da se podjednako često koristi i model stanja cilja u pravouglim koordinatama. Prednost modela u polarnim koordinatama je u jednostavnosti,dok model u pravouglim koordinatama daje manje dinamičke greške ocjene stanja,ali je komplikovaniji za implementaciju .

2.2 Jednačine filtara za praćenje

Za model stanja cilja i mjerenja datih u dijelu 2.1,optimalni estimator stanja je Kalman filter,za koji znamo da minimizira očekivanu dinamičku kvadratnu grešku ocjene. Za inicijaliziranje filtra obično se koriste dva mjerenja,u cilju dobijanja informacije o brzini kretanja objekta koji se prati.Dakle,za iniciranje je potrebno odrediti:

$$\hat{x}(2+) \text{ i } P(2+)$$

gdje je P(2+) matrica kovarijanse greške ocjene,a indekx "2+" označava momenat neposredno poslije prijema drugog mjerenja.Prije dolaska trećeg mjerenja,filtar vrši predikciju stanja cilja i matrice kovarijanse greške:

$$\hat{x}(3-)=\Phi\hat{x}(2+) \qquad (5)$$

$$P(3-)=\Phi P(2+)\Phi^T+GQG^T \; . \qquad (6)$$

Indeks "3-" označava momenat neposredno prije dolaska trećeg mjerenja.Po dolasku trećeg mjerenja odredi se matrica pojačanja filtra:

$$K(3)=P(3-)H^T(HP(3-)H^T+R)^{-1} \qquad (7)$$

te se izvrši korekcija stanja i matrice kovarijanse:

$$\hat{x}(3+)=\hat{x}(3-)+K(3)(y(3)-H\hat{x}(3-)) \qquad (8)$$

$$P(3+)=(I-K(3)H)P(3-) \qquad (9)$$

gdje su:

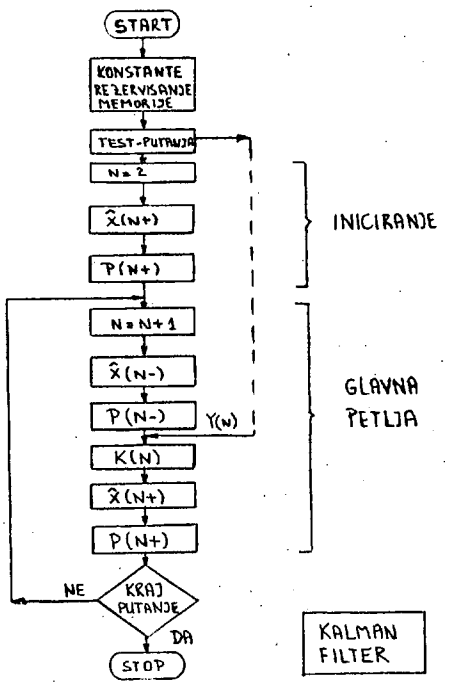
I=jedinična matrica
y(3)=treće mjerenje.

Postupak se zatim ponavlja za nova mjerenja. Kalman-filter je relativno skup i komplikovan za implementaciju,ali daje najtačnije ocjene. Često se pokaže opravdanim korištenje Wienerovog filtra,koji je jednostavniji od Kalmanovog filtra.Dobijemo ga ako u jednačinama zamjenimo K(k) sa K(∞),gdje je K(∞) stacionarno pojačanje Kalmanovog filtra.Filtar se svodi na jednačine za $\hat{x}(k+)$ i $\hat{x}(k-)$,a predhodno moramo "off line" sračunati matricu K(∞) bilo analitički bilo simulacijom na računaru.

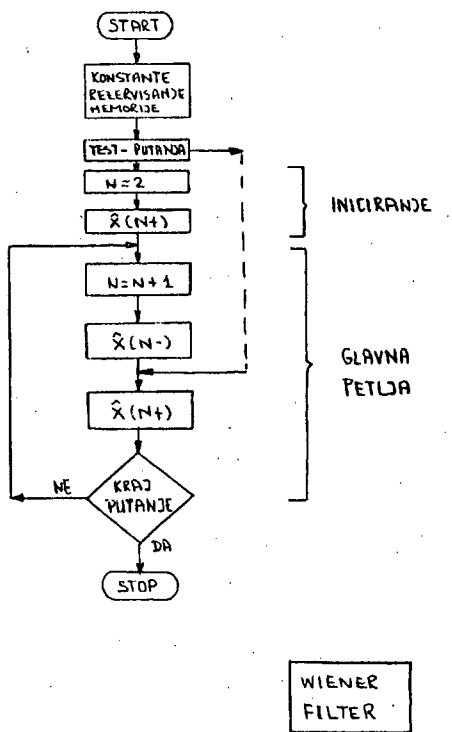
Ušteda u vremenu je značajna,uz odredjeno pogoršanje tačnosti rada filtra koje se najčešće može tolerisati.

3.Dijagram toka programa u Asembleru MPR-52B

Na slikama 1. i 2. dati su dijagrami toka za Kalmanov i Wienerov filtar,odnosno za odgovarajuće asemblerske programe,koje ovde nećemo navoditi zbog njihove veličine.

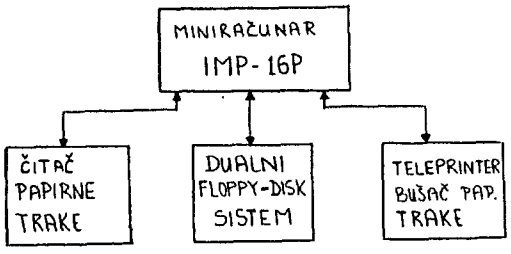


Slika 1.



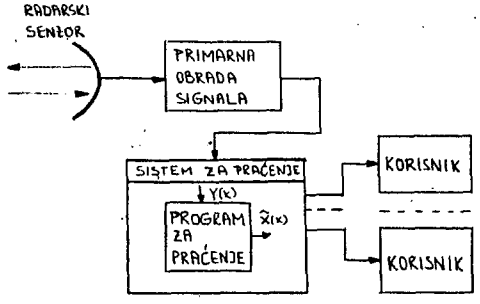
Slika 2.

Prema priloženim dijagramima toka na slikama 1. i 2.,napisani su odgovarajući programi u Asembleru mikroprocesora MPR-52B,a testirani su na miniračunaru IMP-16P. Blok šema kompletnog sistema za testiranje prikazana je na slici 3. Testiranje je izvodjeno za nekoliko tipova putanja ciljeva,i nekoliko tipova radarskih senzora.



Slika 3.

Principijelna blok šema sistema za praćenje u kome će se primjeniti jedan od spomenutih programa za praćenje,data je na slici 4. Sistem za praćenje posjedovaće računar na bazi mikroprocesora MPR-52B Rudi Čajavec.



Slika 4.

4.Karakteristike asemblerskih programa

Poslije testiranja konačnih verzija programa u Asembleru MPR-52B,utvrdjene su slijedeće veličine:

1.Broj instrukcija programa:

Kalman-filter .....~400
Wiener-filter .....~300

2.Broj izvršnih instrukcija:

Kalman-filter ....~200 - iniciranje
~270 - glavna petlja
~470 ukupno

Wiener-filter ....~160 - iniciranje
~210 - glavna petlja
~370 ukupno

Napomena: nisu uzeti u obzir opšti podprogrami(sin,cos,arctg,itd) koji postoje u standardnoj biblioteci opštih programa mikroprocesora MPR-52B. Tačnost programa za sin(x),cos(x) i arctg(x) je reda 10^{-4}.

Stvarna vremena izvršenja programa su:

3.Vrijeme izvršenja:

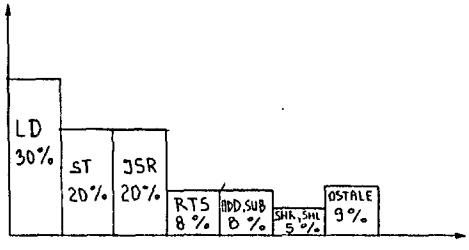
Kalman-filter .... 16.5msec - inic.
12.5msec - gl.pet.
29.0msec ukupno

Wiener-filter ... 14.5msec - inic.
10.0msec - gl.pet.
24.5msec ukupno

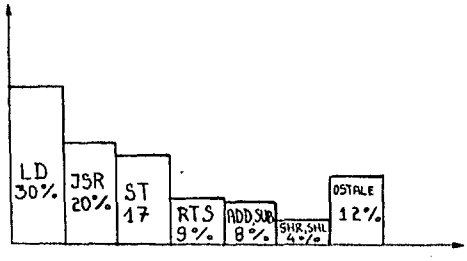
Kad se filtri jednom iniciraju,vrijeme izvršenja u glavnoj petlji koja se ponavlja za svako novo mjerenje je:

Kalman-filter ... 12.5 msec
Wiener-filter ... 10.0 msec.

Ušteda u vremenu dobijena korištenjem Wienerovog filtra je oko 20%,što je u realnim okolnostima veoma značajno.
Učešće pojedinih instrukcija Asemblera MPR-52B u datim programima za praćenje,prikazano je na slikama 5. i 6.,za Kalmanov i Wienerov filter,respektivno.



Slika 5.



Slika 6.

Objašnjenje instrukcija:
LD - punjenje registara
ST - punjenje adrese sadržajem registra
JSR- skok na podprogram
RTS- povratak sa podprograma
ADD,SUB- sabiranje,oduzimanje
SHR,SHL- pomjeranje desno,pomjeranje lijevo

PRIMJER:

1.Karakteristike senzora:

$\sigma_D^2=(900m)^2$, $\sigma_{AZ}^2=(1.8^0)^2$, T=10.0 sec.

2.Zahtjevana tačnost:

po daljini .... greška manja od 200m
po azimutu .... greška manja od 1^0

3.Karakteristike cilja:

brzina .... 600 m/sec
P_1, P_2^A .... 0.1^0 m/sec^2, odnosno 0.3 ,

4.Test-putanja :

pravolinijska na daljini 40 km
kružna poluprečnika 40 km

5.Rezultati:

za oba filtra za praćenje,rezultati su zadovoljavajući(greške po daljini manje od 200m,a po azimutu od 1°).

Performanse obaju filtara bile su predhodno testirane metodom Monte-Karlo,kojom su simulirana realna mjerenja o cilju.Program za Monte-Karlo simulaciju napisan je u Fortranu. Testirani su isti slučajevi koji su kasnije ispitani i na miniračunaru IMP-16P,sa mikroprocesorom MPR-52B. Tačnost rezultata dobijena na miniračunaru IMP-16P je nešto manja od tačnosti dobijene na velikom računaru,zbog kraće dužine riječi (16 bita prema 32,ili 64 bita u duploj preciznosti).Medjutim,rezultati su u granicama zahtjevane tačnosti.

5.Zaključak

Iz priloženih rezultata testiranja asemblerskih programa za Kalmanov i Wienerov filtar za praćenje,jasno je da je za konkretan problem praćenja odabran Wienerov filtar zbog jednostavnije implementacije i uštede u vremenu od približno 20%,uz istovremene zadovoljavajuće rezultate praćenja. Vrijeme izvršenja obaju programa za praćenje na računaru koji će koristiti mikroprocesor MPR-52B je zadovoljavajuće,što indicira upotrebljivost MPR-52B u situacijama praćenja vazdušnih ciljeva u realnom vremenu. Vremena izvršenja programa za praćenje mogu se još smanjiti,ako se neke funkcije programa "hardverizuju" (npr. podprogrami za sin(x),cos(x),itd).

Dodatak

Osnovne karakteristike mikroprocesora MPR-52B su:

Dužina riječi	...	16 bita
Skup instrukcija obuhvata	...	61 instrukciju
Aritmetika	...	paralelna,binarna, sa fiksnom tačkom, sa komplementom do dva
Adresiranje	...	direktno, indirektno, apsolutno, relativno prema programskom brojaču, indeksirano
Memorija	...	do 64K
Broj registara	...	4
Tipične brzine	...	sabiranje registar -registar: 4.9 μsec, sabiranje memorija -registar: 8.4 μsec, punjenje registra sadržajem neke adrese: 10 μsec,

itd.

Za detalje vidjeti reference br.3. i 4.

6.Literatura

1.R.A.Singer,K.W.Behnke,REAL-TIME TRACKING FILTER EVALUATION AND SELECTION FOR TACTICAL APPLICATIONS,IEEE Trans. on Aerospace and Electr.Systems,vol. AES-7, No.1.,January 1971.g.

2.R.A.Sinnger,ESTIMATING OPTIMAL TRACKING FILTER PERFORMANCE FOR MANNED MANEUVERING TARGETS,IEEE Trans. on Aerospace and Elestr. Systems,vol. AES-r,No.4.,July 1970.g.

3.IMP-16P USERS MANNUAL,NSC,Santa Clara,Ca. septembar 1974.g.

4.Priručnik za programiranje i asembler mikroprocesora MPR-52B,Rudi Čajavec, Banja Luka,1977.g.

INFORMATICA 3/1979

MIKRORAČUNALNIŠKA KRIPTOGRAFIJA II

D. NOVAK
A. P. ŽELEZNIKAR

UDK: 681.3.06 : 003.6

INSTITUT JOŽEF STEFAN, LJUBLJANA

Članek opisuje implementacijo standardnega kripcijskega algoritma (DES) na mikroračunalniku (6800).

CRYPTOGRAPHY USING MICROCOMPUTERS II - In the article the implementation of a standard encryption algorithm (DES) on a 6800 based microcomputer is described.

1. UVOD

Že iz naslova je razvidno, da gre v tem članku za nadaljevanje oziroma navezavo na tematiko načeto v prvi letošnji številki Informatice [1]. Opisali bomo implementacijo standardnega podatkovnega kripcijskega algoritma (data encryption standard) na mikroračunalniku s procesorjem 6800. Omenjeni algoritem je izdelal National Bureau of Standards leta 1977. Algoritem je dokaj podrobno opisan v članku Mikroračunalniška kriptografija I [1], zato se bomo osredotočili izključno na implementacijo. Algoritem preslika 64 bitov odkritega teksta v 64 bitov zakritega teksta. Uporablja 64-bitni ključ, iz katerega generira 16 podključev. Ti se lahko izračunavajo sproti ali pred samim kriptiranjem. V našem primeru se podključi izračunajo pred kriptiranjem.

V članku bomo opisali strukturo podatkov oz. tabel in funkcije nad temi podatki. Nakazali bomo nekatere probleme, ki izvirajo iz zmogljivosti oz. nezmogljivosti procesorja in otežujejo implementacijo.

2. SPLOŠNE IMPLEMENTACIJSKE POTEZE

DES algoritem (DES - data encryption standard) je prikazan na sliki 9 (Mikroračunalniška kriptografija I [1]) v obliki diagrama operacij. Zapišimo ta algoritem še v obliki psevdo koda, da nam bodo kasneje dovolj jasne implementacijske podrobnosti (slika 18).

V opisanem algoritmu imamo opravka z naslednjimi funkcijami: permutacijo, seštevanjem po modulu 2, iskanjem po tabeli (pri permutaciji in F-funkciji) in rotiranjem niza bitov (generiranje podključev). Algoritem uporablja naslednje tebele:

- tabela za začetno permutacijo (INIPER)
- tabela za inverzno začetno permutacijo (INVPER)
- tabela za prvo permutacijo znotraj F-funkcije (SELE)
- tabela za drugo permutacijo znotraj F-funkc. (PPER)
- tabele S1-S8 za S-funkcijo
- tabele potrebne pri generiranju podključev

```
MODUL data encryption
  BEGIN
    začetna permutacija;
    razdelitev niza na levo in desno pol.(L(0),D(0));
    FOR i:=0 TO 15 DO
      BEGIN
        L(i+1):=L(i)⊕F(R(i),K(i+1));
        R(i+1):=L(i+1); (*zamen. lev.in des. dela*)
        L(i+1):=R(i);
      END
    FOREND;
    zamenj. levega in desnega dela in stik nizov;
    inverzna začetna permutacija
  END.
```

Opomba: K(i): i-ti podključ
F : funkcija opisana v (1) na sliki 13

Slika 18. DES algoritem

Bitni niz je grupiran v zloge (8 bitov) (slika 19). Posamezne bite znotraj niza naslavljamo z dvema kazalcema, ki sta relativna na začetek niza. Prvi kaže na zlog, v katerem je naslovljeni bit, drugi pa na sam bit oz. mesto znotraj zloga.

ZLOGI	n	...	1	0
BITI ZLOGOV	76543210	...	76543210	76543210
NIZ BITOV	8n-1 8(n-1)	...	11111100 54321098	00000000 76543210

Npr.: 14. bit niza se nahaja v drugi besedi (beseda št. 1) na sedmem mestu (bit št. 6).

Slika 19. Organizacija niza bitov v pomnilniku

3. STRUKTURA KRIPCIJSKEGA PROGRAMA

V tem poglavju bomo opisali klicno strukturo kripcijskega programa (ENCRYP). Za vsako enoto (subrutino) bomo definirali vhodne podatke, izhodne podatke in funkcijo. Tako bomo dobili občutek kako se začetna informacija (čisti tekst) pretvarja (transformira) v rezultat (zakriti tekst). Klicna struktura (slika 20) kaže vgnezdenost posameznih subrutin. Vsak umik pomeni nov nivo vgnezdenja. Subrutina FUNC kliče npr. subrutine PERMUT, KONV in SFUNC. Subrutina SFUNC pa kliče subrutino STRANS.

```
ENCRYP
    PERMUT
    FUNC
        PERMUT
        KONV
        SFUNC
          STRANS
```

Slika 20. Klicna struktura kripcijskega programa ENCRYP

Poglejmo si sedaj po vrsti na sliki 20 predstavljene subrutine:

ENCRYP

VHODI:	PLAINT	osem zlogov čistega teksta
	PODK	kazalec na začetek tabele podključev
IZHODI:	RAZSIR	osem zlogov zakritega teksta

FUNKCIJA: Z uporabo DES algoritma pretvori 64 bitov čistega teksta v 64 bitov zakritega teksta. Uporablja že prej generirano tabelo podključev.

PERMUT

VHODI:	ZACNIZ	kazalec na začetek niza, ki ga želimo permutirati
	KAZALO	kazalec na začetek permutacijske tabele
	STEL	dimenzija permutacijske tabele oz. dolžina izhodnega niza
IZHOD:	REZULT	kazalec na začetek permutiranega niza

FUNKCIJA: Subrutina permutira vhodni niz preko permutacijske tabele v izhodni niz. Element permutacijske tabele pove, kateri bit vhodnega niza naj se naniza k že obstoječemu podnizu.

FUNC

VHODI:	DESNI	kazelec na zač. desnega podniza -R(i)
	LEVI	kazalec na začetek levega podniza -L(i)
	PODKLJ	kazalec na začetek tekočega podključa -K(i+1)
LOK. PODATKI:		
	SELE	permutacijska tabela št. 1
	PPER	permutacijska tabela št. 2
	S12TAB	matrika izbiralnih funkcij S1 in S2
	S34TAB	matrika izbiralnih funkcij S3 in S4
	S56TAB	matrika izbiralnih funkcij S5 in S6
	S78TAB	matrika izbiralnih funkcij S7 in S8
	STABEL	tabela kazalcev na začetke matrik izbiralnih funkcij
IZHODI:	DESNI	kazalec na začetek nespremenjenega desnega podniza
	LEVI	kazalec na začetek levega podniza (L(i+1)=L(i)⊕F(R(i),K(i+1))

FUNKCIJA: Subrutina FUNC izračuna po algoritmu prikazanem na sliki 13 (Mikroračunalniška kriptografija I) vrednost funkcije F(R(i),K(i+1)). K tej vrednosti prišteje po modulu 2 levi podniz L(i).

KONV

VHODI:	X	kazalec na začetek niza dolžine 24 bitov (trije 8-bitni zlogi)
	RZS	kazalec na začetek področja za rezultat

FUNKCIJA: Subrutina KONV razbije 24 bitov spravljenih v treh zaporednih zlogih na štiri šestbitne zloge.

SFUNC

VHOD:	RAZSIR	48-bitni niz v obliki osmih šestbitnih zlogov
LOKALNI PODATKI:		
	STABEL	tabela kazalcev na začetke matrik izbiralnih funkcij
IZHOD:	VMESRE	32-bitni rezultirajoči niz v obliki štirih osembitnih zlogov

FUNKCIJA: Subrutina SFUNC preslika 48-bitni vhodni niz preko matrike izbiralnih funkcij v 32-bitni izhodni niz.

STRANS

VHOD:	STAB	kazalec na začetek matrike izbiralne funkcije
	RZS	kazalec na šestbitni vhodni zlog
IZHOD:	A	štiribitni zlog

FUNKCIJA: Subrutina STRANS preslika en šestbitni zlog preko ustrezne matrike izbiralne funkcije v štiribitni zlog

4. ORGANIZACIJA TABEL

V drugem poglavju smo našteli kup tabel, ki jih uporablja DES algoritem. Poglejmo si organizacijo teh tabel za našo implementacijo. Elementi permutacijskih tabel (začetna permutacija, inverzna začetna permutacija, izbira E in P-permutacija) definirajo premeščanje bitov.

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Slika 21. Tabela za P-permutacijo

Na sliki 21. je prikazana tabela za P-permutacijo. Prvi element (16) pomeni, da prestavimo 16-ti bit na prvo mesto. Vse elemente premutacijskih tabel pretvorimo iz desetiškega v dvojiški sistem. Ker bomo oštevilčili prvi bit z vrednostjo nič, dobimo za gornji primer naslednjo tabelo (slika 22.).

0F	06	13	14
1C	0B	1B	10
00	0E	16	19
04	11	1E	09
01	07	17	0D
1F	1A	02	08
12	0C	1D	05
15	0A	03	18

Slika 22. Tabela za P-permutacijo v heksadecimalni notaciji PPER).

Podobno spremenimo tudi vrednosti v S-tabelah. Pri tem lahko združimo po dve tabeli, ker elementi ne presegajo vrednosti 15 (4 biti). Istoležeča elementa dveh zaporednih tabel združimo v en element (14 iz S_1 in 15 iz S_2 je FE v S12TAB). Tako dobimo namesto osmih tabel le štiri. Pri uporabi teh tabel moramo pač vedeti ali so veljavni gornji štirje biti ali spodnji štirje biti elementa. Iskanje po tabeli pa ostane enako kot je prikazano na sliki 16 (mikroračunalniška kriptografija I).
Vse tabele so prikazane na sliki 23.

5. NEKAJ PODROBNOSTI IZ KRIPCIJSKEGA PROGRAMA

Kompleten listing kripcijskega programa je podan na koncu članka. Za lažje razumevanje si bomo ogledali le dva detajla: subrutino za permutiranje (PERMUT) in pretvorbo 24-bitnega niza v obliko uporabno za izvajanje S-funkcije (KONV).

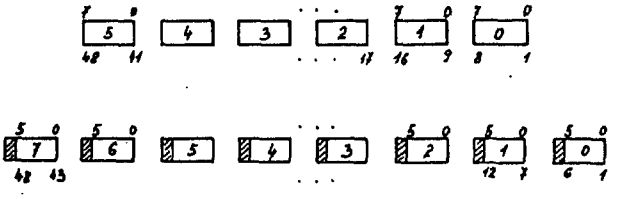
Kot smo že omenili subrutina PERMUT permutira vhodni niz v skladu s permutacijsko tabelo. Pri tem ne pokvari vhodnega niza. Za vsak niz ima dva kazalca: kazalec na zlog v nizu in kazalec na bit v zlogu. Subrutina torej izračuna iz vrednosti elementa v tabeli, vrednosti obeh kazalcev za vhodni niz. Vrednosti kazalcev za izhodni niz sta definirani s pozicijo elementa tabele. Nato se z maskiranjem izloči naslovljeni bit vhodnega niza in se naniza k že obstoječim bitom izhodnega niza (slika 24).

```
SUBRUTINE permut (vhodni niz, permut tabela)
  DO
     prečitaj vrednost naslednjega elem. iz permut.
     tabele;
     izračunaj vrednost kazalcev za izhodni niz;
     maskiraj bit;
     nanizaj bit v izhodni niz;
  UNITL konec tabele.
```

Slika 24. Psevdo kod subrutine PERMUT

Pri maskiranju naslovljenega bita moramo uporabiti ustrezno masko. Zato moramo vrednost kazalca na bit pretvoriti v masko, ki bo imela enico na ustreznem mestu. (Npr.: vrednost 6 je potrebno pretvoriti v B'0100 0000'). Analogno preslikavo moramo izvesti tudi pri dodajanju bita v izhodni niz, le da moramo tokrat postaviti na ustrezno mesto ničlo ali enico glede na vrednost bita v vhodnem nizu.

Pri S-funkciji preslikamo vsakih 6 zaporednih bitov vhodnega niza preko ene od matrik izbiralnih funkcij v 4 zaporedne bite izhodnega niza. Zato je ugodno, če vhodni niz porazdelimo na zaporedne zloge pomnilnika, tako da je v vsakem le šest bitov niza (na mestih 0 do 5). Organizacijo niza pred in po preslikavi kaže slika 25. Za porazdelitev niza uporabimo subrutino KONV.



Slika 25. Organizacija niza pred in po preslikavi s pomočjo subrutine KONV

6. IZRAČUNAVANJE PODKLJUČEV

Algoritem izračunavanja podključev je podrobno opisan v prvem delu članka (slika 10). Na izbiro imamo dve možnosti: da pred začetkom šifriranja izračunamo vseh 16 48-bitnih podključev ali pa, da podključe izračunavamo sproti. Prva varianta je enostavnejša in ima še to prednost, da izračunavanje podključev ne upočasnjuje hitrosti šifriranja. Sam algoritem izračunavanja je preprost. Z ozirom na to, da imamo že napisano subrutino za premeščanje bitov (PERMUT), moramo napisati samo še subrutino za rotiranje 28 zaporednih bitov. V našem programu imamo dejansko dve subrutini. Eno za rotiranje gornjih 28 bitov in eno za rotiranje spodnjih 28 bitov. Vzrok je v neizogibnih nerodnostih, ki izhajajo iz organizacije nizov v pomnilniku.

7. SKLEP

V članku smo imeli namen kolikor mogoče jasno opisati implementacijo DES algoritma, ki je bil podrobno opisan v prvem delu članka (Mikroračunalniška kriptografija I). Opisani program je zapisan v obliki subrutine, ker ga je potrebno vključiti še v komunikacijsko okolje, šifrira se po 64 bitov naenkrat. Isti program je mogoče uporabiti za šifriranje in dešifriranje. Pri dešifriranju mora uporabljati podključe v obratnem vrstnem redu. Program je relativno počasen, saj porabi za en prehod (64 bitov) približno eno sekundo.

8. LITERATURA

1. A.P.Železnikar: Mikroračunalniška kriptografija I, Informatica 3 (1979), štev.1, stran 24-31.

```
S12TAB 2090  FE 14 8D E1  62 BF 3B 48  93 7A 26 DC  C5 09 50 A7
       20A0  30 DF 47 74  FE 22 8D E1  CA 06 1C AB  69 95 B3 58
       20B0  04 E1 7E B8  AD 46 D2 1B  5F 8C C9 67  93 3A 25 F0
       20C0  FD 8C A8 12  34 F9 41 27  B5 6B 73 CE  0A 50 E6 9D
S34TAB 20D0  7A D0 E9 3E  06 63 9F A5  11 2D 8C 57  BB C4 42 F8
       20E0  DD 87 B0 59  63 F4 06 3A  42 78 25 CE  1C AB EF 91
       20F0  AD 66 94 09  C8 BF 73 D0  FB 11 32 EC  55 2A 8E 47
       2100  31 FA 0D 60  A6 19 D8 87  94 4F 5E B3  CB 75 22 EC
S56TAB 2110  C2 1C A4 F1  97 2A 6B 86  08 D5 33 4F  ED 70 5E B9
       2120  AE FB 42 2C  74 C7 9D 51  65 10 DF EA  03 B9 38 86
       2130  94 E2 F1 5B  2A 8D C7 38  7F 09 4C A5  16 D3 B0 6E
       2140  4B 38 2C C7  91 5E F2 AD  B6 EF 10 79  6A 04 85 D3
S78TAB 2150  D4 2B 82 4E  6F F0 B8 1D  A3 9C 39 E7  55 0A C6 71
       2160  1D F0 DB 87  A4 39 71 4A  CE 53 65 BC  02 EF 98 26
       2170  71 B4 4B 1D  9C C3 E7 2E  0A 6F A6 D8  F0 35 59 82
       2180  26 1B ED 78  41 A4 8A D7  F9 C5 90 0F  3E 52 63 BC
       >C 2198 21C7
       :T
 SELE  2198  1F 00 01 02  03 04 03 04  05 06 07 08  07 08 09 0A
       21A8  0B 0C 0B 0C  0D 0E 0F 10  0F 10 11 12  13 14 13 14
       21B8  15 16 17 18  17 18 19 1A  1B 1C 1B 1C  1D 1E 1F 00
       >C 21C8 21E7
       :T
 PPER  21C8  0F 06 13 14  1C 0B 1B 10  00 0E 16 19  04 11 1E 09
       21D8  01 07 17 0D  1F 1A 02 08  12 0C 1D 05  15 0A 03 18
       >C 2370 23AF
       :T
INIPER 2370  39 31 29 21  19 11 09 01  3B 33 2B 23  1B 13 0B 03
       2380  3D 35 2D 25  1D 15 0D 05  3F 37 2F 27  1F 17 0F 07
       2390  38 30 28 20  18 10 08 00  3A 32 2A 22  1A 12 0A 02
       23A0  3C 34 2C 24  1C 14 0C 04  3E 36 2E 26  1E 16 0E 06
       >C 23B0 23EF
       :T
INVPER 23B0  27 07 2F 0F  37 17 3F 1F  26 06 2E 0E  36 16 3E 1E
       23C0  25 05 2D 0D  35 15 3D 1D  24 04 2C 0C  34 14 3C 1C
       23D0  23 03 2B 0B  33 13 3B 1B  22 02 2A 0A  32 12 3A 1A
       23E0  21 01 29 09  31 11 39 19  20 00 28 08  30 10 38 18
       >
```

SLIKA 23. S - TABELE IN PERMUTACIJSKE TABELE

```
*****************************************************
*                                                   *
*   ENCRYP                                          *
*                                                   *
*  SUBRUTINA PRESLIKA PO DES ALGORITMU 64 BITOV     *
*  IZ PLAINT V 64 BITOV ZAKRITEGA TEKSTA V RAZSIR   *
*                                                   *
*****************************************************
*
PODK   EQU    $24E8
PLAINT RMB    8        CISTI TEXT
TEXT   RMB    8        PERMUTIRANI TEXT
*
*
ENCRYP LDX    #PLAINT  TEXT IZ PLAINT
*                      SE PERMUTIRA IN
*                      SPRAVI V TEXT
       STX    ZACNIZ
       LDX    #TEXT
       STX    REZULT
       LDX    #INIPER
       STX    KAZALO
       LDAA   #$40
       STAA   STEL
       JSR    PERMUT
*
*
       LDX    #TEXT    RAZBIJEMO TEXT NA LEVO
       STX    LEVI     IN DESNO POLOVICO
       LDX    #TEXT+4
       STX    DESNI
       LDX    #PODK
       STX    PODKLJ
       LDAB   #$10
       PSHB
*
ENCRY1 JSR    FUNC     LEVI:=LEVI+F(R,KI)
*
       LDX    LEVI     ZAMENJAVA LEVEGA IN
       STX    TEMP     DESNEGA DELA
       LDX    DESNI
       STX    LEVI
       LDX    TEMP
       STX    DESNI
       PULB
       DECB
       PSHB
       CMPB   #0
       BNE    ENCRY1
*
       LDAB   #4       ZAMENJAVA LEVEGA IN
       LDX    #PLAINT+4 DESNEGA DELA
       STX    DESNI
       LDX    #TEXT+4
       STX    LEVI
EXC    LDX    LEVI
       LDAA   0,X
       INX
       STX    LEVI
       LDX    DESNI
       STAA   0,X
       INX
       STX    DESNI
       DECB
       CMPB   #0
       BNE    EXC
*
*
       LDX    #PLAINT+4 INVERZNA PERMUTACIJA
       STX    ZACNIZ
       LDX    #INVPER
       STX    KAZALO
       LDX    #RAZSIR
       STX    REZULT
       LDAA   #$40
       STAA   STEL
       JSR    PERMUT
       RTS
```

```
*****************************************************
*                                                   *
*          SUBRUTINA PERMUT                         *
*                                                   *
*  SUBRUTINA PERMUT PERMUTIRA IZVIRNI NIZ, KATEREGA *
*  ZACETEK JE DEFINIRAN Z VREDNOSTJO "ZACNIZ" Z UPO-*
*  RABO PERMUTACIJSKE TABELE ("KAZALO MORA BITI INI-*
*  CIALIZIRANO NA ZACETEK TABELE) V PONORNI NIZ "RE-*
*  ZULT", "STEL" MORA VSEBOVATI STEV. EL. PERM. TAB.*
*                                                   *
*****************************************************
*
*
*
       OPT    MEM
       ORG    $2000
KAZALO RMB    2        KAZALEC V PERMUT. TABELI
ZACNIZ RMB    2        KAZALEC NA ZACETEK
*                      IZVIRNEGA NIZA
BESNIZ RMB    2        KAZALEC V IZVIRNEM NIZU
*                      (ZLOG 0:7)
REZULT RMB    2        KAZALEC NA ZACETEK PONORNEGA
*                      NIZA
BESREZ RMB    2        KAZALEC V PONORNEM NIZU
*                      (ZLOG 0:7)
ITER   RMB    1        STEVEC PONOVITEV
STEL   RMB    1        STEVILO ELEMENTOV V PRERMUTACIJI
BIT1   RMB    1        KAZALEC V PONORNEM NIZU
*                      (BIT ZLOGA (BESREZ))
BIT    RMB    1        KAZALEC V IZVIRNEM NIZU
*                      (BIT ZLOGA (BESNIZ))
*
*
PERMUT CLR    ITER
PER6   LDX    KAZALO
       LDAB   0,X      IZ TABELE VZEMI NOVO VREDNOST
       ANDB   #07
       STAB   BIT
       LDAB   0,X      IZRACUNAJ V KATEREM ZLOGU
       LSRB            BESNIZ IN NA KATEREM MESTU BIT
       LSRB            SE NAHAJA BIT IZVIRNEGA NIZA
       LSRB            KI GA MORAMO NANAIZATI
       INX             V PONORNI BIT
       STX    KAZALO
       LDX    ZACNIZ
       STX    BESNIZ
       LDAA   ZACNIZ+1
       ABA
       STAA   BESNIZ+1
*
       LDX    REZULT
       STX    BESREZ   NASTAVI KAZALEC V PONORNEM
       LDAA   ITER     NIZU BESREZ
       LSRA
       LSRA
       LSRA
       LDAB   REZULT+1
       ABA
       STAA   BESREZ+1
*
       LDAA   ITER     NASTAVI PONORNI MASKIRNI BIT
       ANDA   #07      NA USTREZNO MESTO ZNOTRAJ
       LDAB   #01      ZLOGA
PER0   CMPA   #00      (VREDNOST - POZICIJA;
0      BEQ    PER1     PRIMER: 5 00100000)
       ASLB
       DECA
       JMP    PER0
*
PER1   LDX    BESNIZ
       STAB   BIT1     NASTAVI IZVIRNI MASKIRNI
       LDAB   BIT      BIT NA USTREZNO MESTO
       LDAA   #$1      ZNOTRAJ ZLOGA
PER4   CMPB   #00      (VREDNOST - POZICIJA)
       BEQ    PER5
       ASLA
       DECB
*
       JMP    PER4     CE JE IVIRNI BIT=0
PER5   LDAB   BIT1     POJDI NA PER2
       ANDA   0,X      SICER NADALJUJ
       BEQ    PER2
       LDX    BESREZ
       ORAB   0,X
       STAB   0,X      DODAJ 1 V PONORNI NIZ
       JMP    PER3
*PER2  COMB
*      LDX    BESREZ
*      ANDB   0,X
*      STAB   0,X      DODAJ 0 V PONORNI NIZ
*
*PER3  LDAA   ITER     CE JE KONEC TABELE SE VRNI
*      INCA            SICER PONOVI
*      STAA   ITER
*      CMPA   STEL
*      BNE    PER6
*      RTS
*      END
*
*****************************************************
*                                                   *
*       SUBRUTINA FUNC                              *
*                                                   *
*  SUBRUTINA PERMUTIRA NIZ NA KATEREGA KAZE KAZALEC *
*  "DESNI". REZULTATU PRISTEJE PO MODULU DVA PODKLJUC*
*  NA KATEREGA KAZE KAZALEC "DODKLJ". DOBLJENI NIZ  *
*  RAZBIJE NA OSEM SEST BITNIH BESED, KI JIH PRESLIKA*
*  PREKO S-TABEL V NOVI NIZ. TA NIZ PERMUTIRA IN MU *
*  NA KONCU PRISTEJE (MOD2) NIZ NA KATEREGA KAZE    *
*  KAZALEC "LEVI". REZULTAT SE SHRANI V LEVI OPERAND.*
*                                                   *
*****************************************************
*
*
*       UPORABLJA SUBRUTINO: PERMUT, KONV, SFUNC
*
DESNI  RMB    2        ZACETEK DESNEGA OPERANDA
LEVI   RMB    2        ZACETEK LEVEGA OPERANDA
PODKLJ RMB    2        ZACETEK PODKLJUCA
VMESRE RMB    6        VMESNI REZULTAT
VMR    RMB    2        KAZALEC V VMERSRE
RAZSIR RMB    8        VMESNI REZULTAT
RZS    RMB    2        KAZALEC V RAZSIR
TEMP   RMB    2
STAB   RMB    2
STABKA RMB    2        KAZALEC V TABELI ZACETKOV S-TABEL
*
FUNC   LDX    DESNI    ZACETNA PERMUTACIJA
       STX    ZACNIZ   PREKO SELECT E TABELE
       LDX    #SELE
       STX    KAZALO
       LDX    #VMESRE
       STX    REZULT
       STX    VMR
       LDAA   #$30
       STAA   STEL
       JSR    PERMUT
*
       LDAB   #$06
FUNC1  LDX    PODKLJ   VSOTA PO MOD 2 MED
       LDAA   0,X      VMESNIM REZULTATOM IN
       INX             PODKLJUCEM
       STX    PODKLJ
       LDX    VMR
       EORA   0,X
       STAA   0,X
       INX
       STX    VMR
       DECB
       CMPB   #0
       BNE    FUNC1
*
       LDX    #RAZSIR+3 KONVERZIJA 48-BITNEGA
       STX    RZS      VMESNEGA REZULTATA NA OSEM
       LDX    #VMESRE 6-BITNIH BESED
       JSR    KONV
       LDX    #RAZSIR+7
       STX    RZS
       LDX    #VMESRE+3
       JSR    KONV
*
       BSR    SFUNC    S-FUNKCIJA
*
       LDX    PPER     PERMUTACIJA P
       STX    KAZALO
       LDX    #VMESRE
       STX    ZACNIZ
       LDX    #RAZSIR
       STX    REZULT
       LDAA   #$20
```

```
        STAA    STEL
        JSR     PERMUT
*
        LDAB    #4      VSOTA PO MOD 2 MED LEVIM
        LDX     #RAZSIR OPERANDOM IN S-FUNKCIJO
        STX     RZS     (DESNI PODKLJ)
        LDX     LEVI
        STX     TEMP
FUNC2   LDX     RZS
        LDAA    0,X
        INX
        STX     RZS
        LDX     TEMP
        EORA    0,X
        STAA    0,X
        INX
        STX     TEMP
        DECB
        CMPB    #0
        BNE     FUNC2
        RTS
*
*****************************************************
*                                                   *
*         SUBRUTINA  SFUNC                          *
*                                                   *
*  SUBRUTINA PRESLIKA OSEM SEST BITNIH BESED IZ     *
*  PODROCJA "RAZSIR", PREKO S-TABEL V 32 BITNI NIZ, *
*  KI GA SPRAVI V "VMESRE".                         *
*                                                   *
*****************************************************
*
*         UPORABLJA SUBRUTINO STRANS
*
SFUNC   LDX     #VMESRE
        STX     VMR
        LDAB    #7
        PSHB
        LDX     #RAZSIR
        STX     RZS
        LDX     #STABEL
        STX     STABKA
SFUNC4  LDX     STABKA
        LDAA    0,X
        STAA    STAB
        LDAA    1,X
        STAA    STAB+1
        CLRB
        BSR     STRANS
        PULB
        BITB    #1
        BEQ     SFUNC1
        ANDA    #$F
        LDX     VMR
        STAA    0,X
        BRA     SFUNC2
SFUNC1  LDX     VMR
        ANDA    #$F0
        ORAA    0,X
        STAA    0,X
        INX
        STX     VMR
        LDX     STABKA
        INX
        INX
        STX     STABKA
SFUNC2  CMPB    #0
        BEQ     SFUNC3
        DECB
        PSHB
        LDX     RZS
        INX
        STX     RZS
        BRA     SFUNC4
SFUNC3  RTS
*
*****************************************************
*                                                   *
*          SUBRUTINA  STRANS                        *
*                                                   *
*  SUBRUTINA TRANSFORMIRA SESTBITNO BESEDO IZ PODRO-*
*  CJA "RADSIR" NA KATERO KAZE KAZALEC "RZS" PREKO  *
*  TABELE KATERE ZACETEK JE DEFINIRAN Z "STAB" V    *
*  STIRIBITNI ZLOG V AKUMULATORJU A.                *
*                                                   *
*****************************************************
*
*
STRANS  LDX     STAB
        STX     TEMP
        LDX     RZS
        LDAA    #$20
        BITA    0,X
        BEQ     STR1
        LDAB    #$20
STR1    LDAA    #1
        BITA    0,X
        BEQ     STR2
        LDAA    #$10
        BRA     STR3
STR2    CLRA
STR3    ABA
        LSR     0,X
        LDAB    0,X
        ANDB    #$F
        ABA
        CLRB
        ANDA    TEMP+1
        STAA    TEMP+1
        ADCB    TEMP
        STAB    TEMP
        LDX     TEMP
        LDAA    0,X
        RTS
*
*****************************************************
*                                                   *
*          SUBRUTINA  KONV                          *
*                                                   *
*  SUBRUTINA RAZBIJE 24 ZAPOREDNIH BITOV, OZNACENIH *
*  S KAZALCEM V INDEKSNEM REGISTRU, V STIRI SESTBITNE*
*  BESEDE, KI JIH DHRANI V PODROCJE DEFINIRANO Z    *
*  "RZS". (RZS-3, ...., RZS)                        *
*                                                   *
*****************************************************
*
*
KONV    LDAA    0,X
        ANDA    #$3F
        PSHA
        ROL     0,X
        ROL     0,X
        ROL     0,X
        LDAA    0,X
        ANDA    #3
        STAA    0,X
        LDAA    1,X
        ASLA
        ASLA
        ANDA    #$3F
        ORAA    0,X
        PSHA
        LSR     1,X
        LSR     1,X
        LSR     1,X
        LSR     1,X
        LDAA    2,X
        ASLA
        ASLA
        ASLA
        ASLA
        ANDA    #$3F
        ORAA    1,X
        PSHA
        LSR     2,X
        LSR     2,X
        LDAA    2,X
        PSHA
        LDAB    #4
        LDX     RZS
*
KONV1   PULA
        STAA    0,X
        DEX
        DECB
        CMPB    #0
        BNE     KONV1
        RTS
        END
```

INFORMATICA 3/1979

VLOGA RAČUNALNIŠKIH MREŽ SE SPREMINJA

D. L. A. BARBER

UDK: 681.324

DIRECTOR, EUROPEAN INFORMATICS NETWORK
NATIONAL PHYSICAL LABORATORY,
TEDDINGTON, MIDDX. UK

Članek podaja kratek pregled razvoja zasebnih in raziskovalnih računalniških mrež in obravnava njihov vpliv na zasnovo sodobnih informacijskih služb. Nato obravnava vpliv najnovejših dosežkov v mikroelektroniki na načrtovanje mrež. Razpravi o problemih, s katerimi se srečujejo uporabniki, sledijo predlogi za nov pristop k standardizaciji. Nakazane so nekatere možne razvojne usmeritve.

THE CHANGING OF COMPUTER NETWORKS

The paper briefly reviews the development of private and research computer networks, and discusses their influence on the design of present day public data services. It goes on to consider the influence on network design of the recent developments that have taken place in microelectronics. A discussion of the problems facing users is followed by some proposals for a new approach to standardisation and the paper concludes with some forecast of possible future trends.

UVOD

Minilo je približno dvajset let, odkar so ljudje začeli komunicirati z računalniki preko telefonskih linij. V tem času je razvoj skokovito napredoval in dejansko obstajajo cela področja sodobnega življenja, ki jih ne bi bilo mogoče upravljati brez današnjih računalniških mrež. Dramatični tehnološki dosežki, ki smo jim bili priče v zadnjem času, pa obljubljajo v razvoju računalniških mrež še nove globoke spremembe. Zato lahko pričakujemo, da bo vloga računalniških mrež čez nekaj let drugačna kot danes, ko večinoma nudijo potrebno infrastrukturo proizvodnji in prometu. Sprožile bodo socialno revolucijo, v kateri se bo spremenil način dela in preživljanja prostega časa pri običajnih ljudeh.

Članek na kratko podaja razvoj računalniške mreže v zadnjih dveh desetletjih in opisuje današnjo javno podatkovno službo in probleme, s katerimi se srečujejo uporabniki. Nato obravnava, kako bo nova tehnologija spremenila naravo računalniških mrež in nazadnje daje nekaj dolgoročnih napovedi.

Prispevek je bil referiran v Angleščini na XIV. Simpoziju Informatica 79, Bled, 1-6. Oktobra 1979.

KRATEK ZGODOVINSKI PREGLED

Razvoj tehnik multiprogramiranja za prvo velike računalniške sisteme je omogočil, da jih je lahko hkrati uporabljalo več uporabnikov. Zelo ugodno je postalo, da so določeno število uporabnikov priključili na računalnik preko telefonskih linij, tako da so lahko delali v svojih prostorih in jim ni bilo treba potovati v računski center. Te zgodnje računalniške mreže so bile zgrajene v obliki zvezdo s procesorji in spomini v centru in z radialnimi linijami, preko katerih je potekala izmenjava informacij v obliki znakov. Terminali so bili preprosti, vse pa je nadzoroval centralni računalnik.

Z napredkom tehnologije so začeli uporabljati male računalnike za nadzor skupino terminalov

in skupina takih računalnikov je bila priključena na centralni sistem. Podatki so se še vedno obdelovali centralno, toda funkcija rutinskega upravljanja s terminali je bila razporejena po mreži. Informacije so se prenašale med centralnim sistemom in nadzorniki terminalov v obliki blokov znakov. Uvedena je bila kontrola napak s pomočjo preizkusa vsote in s ponovno oddajo nepravilno prenešenih blokov. Iz teh začetkov so zrastle zasebne podatkovne mreže, ki jih uporabljamo že mnogo let in predstavljajo nujno infrastrukturo za letališča, banke in druge narodne in mednarodne organizacije, ki igrajo vedno pomembnejšo vlogo v modernem svetu.(glej 1)

Vzporedno z rastjo zasebnih mrež za velike organizacije so se razvile splošne javne računalniške službe. Njihove usluge lahko koristi vsak z uporabo preprostega terminala in javne telefonske mreže. Tudi te službe uporabljajo male računalnike za nadzor terminalov, kar ima za glavni sistem številne prednosti. Pogosto so terminali enega računskega centra razporejeni v različnih državah.

V poznih šestdesetih letih so številni raziskovalci ugotovili, da delujejo različne zasebne mreže na podobnih principih, vendar niso združljive zaradi drobnih razlik. Postalo je jasno, da bi bilo nadaljnje investiranje v infrastrukturo zasebnih mrež take vrste v temelju zgrešeno. Kot nasprotna rešitev se je okrog leta 1965 sočasno v ZDA in Veliki Britaniji pojavila in razvila ideja, da bi gradili komunikacijske podsisteme, ki bi bili neodvisni od posameznih aplikacij in bi lahko pokrili širok spekter potreb.

Tak podsistem nadzira prenos informacij v standardiziranih blokih ali "paketih" med komunikacijskim podsistemom in računalniškimi podsistemi, ki ga uporabljajo. Posplošena računalniška mreža torej vključuje serijo računalniških sistemov ali "gostiteljev", ki so med seboj povezani s paketi preko podmreže. Specializirani gostitelji upravljajo skupine znakovnih terminalov in zanje organizirajo komuniciranje s paketnimi terminali. Posebni gostitelji imajo pomembno vlogo prilagajati različne vrste terminalov na standardni paketni vmesnik. Tako oblikujejo nekakšna lokalna področja v glavni paketni mreži, ki je ekvivalentna visoko zmogljivi medkrajevni mreži.

V ZDA se je ARPA mreža (glej 2) burno razvijala od zgodnjih sedemdesetih let do danes, ko vključuje okrog 70 paketnih priključkov s povezavami v različnih deželah izven kontinentalnih Združenih držav. V Veliki Britaniji so v National Physical Laboratory -u spoznali, da bo uvajanje priključkov za lokalna področja povezano s specifičnimi problemi, ki so jih začeli proučevati na modelu v laboratoriju. Rezultat poskusa je laboratorijska služba, ki zdaj zajema 20 gostiteljev in okoli 200 terminalov. Povezana je z drugimi mrežami zunaj laboratorija, tako da lahko njegovi sodelavci komunicirajo in delajo eksperimente z ARPA mrežo in obratno. (glej 3)

Do zgodnjih 70' so se za problem paketno priključnih mrež začele zanimati tudi druge raziskovalne skupine in pojavile so se narodne raziskovalne mreže kot na primer francoski Cyclades projekt in GMD mreža v ZRN (glej 4,5). Leta 1973 je več evropskih držav ustanovilo European Informatics Network Project - EIN (glej 6). V njegovem okviru je bila zgrajena mednarodna paketna mreža, ki jo uporabljajo različni narodni raziskovalni laboratoriji.

Pomemben korak v Evropi je bil predlog, da bi razvili mrežo podatkovnih baz za razširjanje znanstvenih in tehničnih informacij - Euronet (glej 7). Commission of the European Communities se je o tem dogovorila z nekaterimi evropskimi PTT organizacijami, tako da bo konec leta 1979 že začela delovati mednarodna javna paketna mreža.

Sredi sedemdesetih let je nekaj evropskih PTT organizacij zgradilo paketne raziskovalne mreže, na primer RCP v Franciji in EPSS v Veliki Britaniji, tako da take mreže delujejo (ali bodo začele v bližnji prihodnosti) v večini evropskih držav. V ZDA deluje od 1975 Telenet, ki bazira na ARPA tehnologiji mreže. V Kanadi od leta 1976 obratuje sistem Datapac družbe Bell Northern, Japonska bo kmalu odprla mrežo DDX. Ker so vse te javne paketne mreže povezane, se bodo kmalu razvile v infrastrukturo svetovnih razsežnosti.

Napredek mikroelektronike pospešuje razvoj paketnih mrež z izdelavo cenenih in zanesljivejših spominskih in logičnih elementov. Viden je tudi vpliv na sisteme, ki so vezani na podmrežo in na izbiro načinov komuniciranja. Novi tehnološki principi so tako pocenili digitalni način prenosa informacij, da ga začenjajo uvajati v telefonijo. Te digitalne mreže so potencialno primerne za prenos podatkov, zato lahko

po letu 1990 pričakujemo enotne oddajne mreže za govor in podatke. Verjetno še ne bodo zajele vsega sveta, zelo pomembne in učinkovite pa bodo na manjših področjih. Lokalne mreže bodo povezane z "redko" mrežo digitalnih kanalov, ki bodo izvedeni kot sinhroni digitalni kanali ali pa kot kanali za prenos paketov preko analognih linij. Zaenkrat ni mogoče napovedati, katera tehnika bo dejansko uporabljena. Odločitev bo odvisna od novih tehnoloških pristopov, na primer od razvoja linij iz optičnih vlaken in multikanalnih satelitov, od političnih tokov, od razvoja uprave, od investicij in obratovalnih stroškov in seveda od tega, kakšne vrste komunikacijskih potreb bo treba zadovoljiti.

Cena fizikalne opreme je dovolj nizka, da je mogoče graditi inteligenčne terminale za domačo uporabo. Lahko predvidevamo, da bodo ljudje za večino rutinskih del uporabljali lokalne sisteme, oddaljene sisteme pa relativno redko. Če se bo ta razvoj nadaljeval, je mogoče, da bo promet bodočnosti povsem drugačen, kot si ga zamišljamo danes, zato bodo potrebne nove zasnove mrež.

JAVNE PAKETNE MREŽE

Ko so PTT organizacije na podlagi raziskovalnih projektov priznale, da je tehnika paketnih priključkov dobra osnova za razvoj javnih podatkovnih mrež, je to sprožilo poplavo praktičnih realizacij (glej 8). Razen tega bodo priporočila CCITT-ja (glej 9) omogočila vključitev znakovnih terminalov v enotno paketno mrežo.

Za PTT pomeni DTE (uporabniški terminal - data terminal equipment) karkoli, kar je povezano z DCE (priključek na mrežo - data circuit terminating equipment), ki predstavlja vmesnik med mrežo in zunanjim svetom. To pomeni, da so terminali lahko računalniki, tiskalniki, tastature in podobno. Prav tako lahko smatramo za terminale računalnike z lastnim sistemom preprostih terminalov ali zasebne lokalne mreže, ki vključujejo več terminalov in računalnikov. Torej lahko govorimo o "navideznih" in "realnih" terminalih (terminalih v običajnem pomenu besede). Dejansko se terminali mrež delijo na CDTE (DTE, ki uporabljajo znake - character handling DTEs) in PDTE (DTE, ki uporabljajo pakete - packet handling DTEs). Komuniciranje med terminali različnih vrst poteka na principu sestavljanja in razstavljanja paketov na znake (PAD - Packet Assembler/Disassembler). Podrobnosti bodo opisane kasneje.

Toda nepopustljivi pohod mikroelektronike se nadaljuje. Prinaša inteligenčne terminale in v zadnjem času male računalnike, ki so s svonimi logičnimi in spominskimi zmogljivostmi že pravi računski centri. Ne potrebujejo PAD vmesnika, ker se promet znakov odvija med njihovimi lastnimi tastaturami, prikazovalniki in lokalnimi spomini. Kadar pa komunicirajo preko mreže, uporabljajo pakete in morajo upoštevati standarde za povezavo s PDTE. Ker te terminale lahko programiramo, obstaja niz načinov za praktično realizacijo vmesnikov. To bo brez dvoma omogočilo sodelovanje zaprtih skupin uporabnikov in povzročilo nadomestitev zasebnih mrež z javnimi. Naš dolgoročni cilj mora biti sprejem univerzalnih standardov, ki bodo omogočili "odprto delo" preko javne mreže vsem vrstam terminalov (glej 10). Temu problemu se v zadnjem času posveča veliko pozornosti in ga članek obravnava v poglavju o protokolih na nivoju uporabnika.

Računalniške mreže danes vključujejo računalnike in klasične vrste terminalov, mikroelektronika pa nam obljublja številne nove možnosti, na primer opremo za sintezo in razpoznavanje govora in uporabo višjih grafičnih tehnik raznih vrst. Industrija komunikacij razvija digitalni faksimile, videotekst in teletekst, v sisteme se vključujejo novi terminali in ni še jasno, kako se bo prilagodila javna mreža.

Povezava novih vrst terminalov v skladno celoto se zdi skrajno težka, toda če je ne bomo znali izvesti, bomo imeli čez nekaj let kup nezdružljivih terminalov in ločenih mrež.

Ideja, da bi uporabniki imeli različne terminale za isti namen, povezane s "svojo" žico na neko "svojo" mrežo, je, milo rečeno, čudna. Prej ali slej bo nekdo moral povezati te funkcije v veliko črno škatlo, ki bo nudila uporabniku kompletne usluge. Povezana bo z enotno javno mrežo preko enega samega fizikalnega vodila. Podroben opis presega okvir članka. Toda večina ljudi ne bo hotela, da bi vsak večer prevzeli monopol nad njihovim televizorjem in telefonsko linijo otroci, ki bi pisali domače naloge s pomočjo videoteksta in oddaljenega računalnika.

PROBLEMI NA NIVOJU UPORABNIKA

Daljinska uporaba računalnikov s terminali je že vsakdanjost, toda javlja se skoraj izključno v zaprtih skupinah uporabnikov, na primer

izpostave bank knjižijo direktno v glavni centrali, letalska služba ima enoten sistem rezervacij itd. Misel, da bi človek lahko opravljal svoje zadeve preko terminala na ulici in javne podatkovne mreže, je razburljiva, toda neizvedljiva brez ustreznih standardov. Ti bi morali urejati vsa vprašanja od priključitve terminala na mrežo do ukazov, s katerimi bi uporabnik dosegel zaželeni objekt. Med tema dvema skrajnostima je še cela lestvica potrebnih dogovorov o podrobnostih upravljanja terminalov, o podatkovnih strukturah, ki jih bomo uporabljali,in tako naprej.

Trenutni pristop k reševanju teh standardizacijskih problemov je uvajanje protokolov ločeno za posamezne nivoje komunikacijskega procesa (glej 11). Protokol je množica pravil, ki urejajo komuniciranje med partnerji. Navadno lahko vnaprej določimo nekaj relativno neodvisnih potrebnih procedur. Vsako opravlja eden izmed protokolov, ki so urejeni po nivojih tako, da je protokol nižjega nivoja orodje protokola višjega nivoja.

Kot primer si oglejmo CCITT priporočilo X.25. To pokriva področje izmenjave paketov med PDCE in PDTE in obravnava tri nivoje. Najprej je tu specifikacija fizikalnih povezav - priključkov in električnih signalov. Sledi procedura za izmenjavo toka bitov, urejenih v nabore, po ISO HDLC standardu. Nato je obdelan paketni nivo, tipi in struktura paketov, ki potujejo med DCE in DTE. Vse javne paketne mreže bodo uporabljale vmesnike X.25 za povezavo s PDTE-ji, zato je ta standard univerzalen. Ker pa ne pove ničesar o strukturi in vsebini podatkovnih polj paketov, pušča široko odprto vprašanje, kako bodo različni terminali z vmesniki X.25 lahko v praksi učinkovito komunicirali.

Stanje nekoliko omilijo tri druge X serije priporočil, X.3, X.28 in X.29. To so protokoli, pomembni pri načrtovanju in delu s PAD programi za javne mreže.

X.3 opisuje lastnosti PAD-a in načine, kako ga uporabimo za nadzor različnih terminalov. X.28 ureja interakcijo med znakovnimi terminali in PAD-om in podaja ukaze in odgovore, ki jih uporablja operater v pogovoru s PAD-om, ko vzpostavlja zvezo z oddaljenim PDTE. X.29 opisuje, kako računalniška služba preko PAD-a dela s posameznimi terminali in kako se izmenjavajo podatki v obliki paketov med PAD-om in PDTE.

PAD prilagaja znakovne terminale paketnim, zato narekuje določena pravila, ki jih morajo upoštevati PDTE, kadar preko javne mreže sodelujejo s PAD-om. Teoretično bi lahko te standarde uporabili tudi za komuniciranje dveh paketnih terminalov, vendar niso bili zasnovani v ta namen in so za to manj primerni. Žal bodo novi, boljši standardi za komuniciranje med PDTE-ji verjetno nezdružljivi s standardi za komuniciranje med PAD-om in PDTE. Ključni vzrok je, da je X.29 nesimetrični protokol, ki predvideva, da je PAD podrejen PDTE, medtem ko v primeru povezave dveh paketnih terminalov ne moremo a priori privzeti, da je en terminal suženj drugega. Posplošeni protokol mora torej vsebovati mehanizem dogovora, ki bo preprečil morebitne spore pri inicializaciji (glej 12). V nekaterih javnih mrežah je preko petdeset PAD parametrov, tako da priključitev na računalniški kompleks ni preprosta. Dejanska cena upravljanja terminalov s PAD-om je verjetno mnogo višja, kot se splošno misli. Sledi opis metode "navideznih terminalov", ki se zdi boljša.

Kljub svojim pomanjkljivostim tri X priporočila CCITT omogočajo računalniku, da lahko v PAD-u poda parametre za delo z različnimi znakovnimi terminali. To je pomemben korak k odprti mreži, kjer lahko vsak terminal zahteva poljubno uslugo. Toda samo dejstvo, da vse, kar tipkamo na terminalu, potuje v oddaljeni računalnik in da se vse, kar računalnik generira, pokaže na pravem terminalu, še ne jamči, da so terminali univerzalno uporabni za interaktivno delo z računalniško službo, čeprav bodo res pokrili nekaj precej primitivnih potreb.

Tačas,ko so PTT organizacije razvijale javne mreže predvsem za delo s terminali preko opisanih linij, se je vrsta raziskovalnih projektov začela ukvarjati s problemom interakcije dveh kompleksnih sistemov, na primer dveh lokalnih privatnih mrež, od katerih vsaka opravlja vrsto služb, ima svoj lasten šop terminalov in je včasih tudi priključena na nekaj lokalnih računalnikov.

Načrtovali so, da bi razvili nivoje protokolov, ki bi bili drugačni, toda analogni protokolom CCITT. Morda so bila principielni razlog za razhajanje različna izhodišča. Prve raziskovalne mreže so univerzalno uporabljale koncept datagramov. To so paketi, ki vsebujejo popolne instrukcije, ki omogočajo prenos paketa kot enovite celote. Današnje javne mreže pa so načrtovane za prenos več paketov skupaj v navidezno enotnem tokokrogu. Raziskovalni projekti so se

prvotno odločili za datagramske podmreže, ker so po naravi preprostejše v tem, da obravnavajo vsak paket ločeno od drugih. Toda pri večini klicev med člani mreže morajo biti paketi združeni, zato so raziskovalci razvili koncept transportne službe. To je nivo protokola, ki vključuje software v vseh računalniških sistemih in omogoča sočasni pretok več skupin paketov, jih sortira, razvršča v zaporedja in ukrepa ob izgubi paketa. Transportna služba realizira zanesljive kanale vglavnem s pomočjo uporabniških računalnikov in ne v DCE kot pri X.25. Vključuje tudi direktno kontrolo pretoka med dvema DTE, ki v priporočilu X.25 ni predvidena.

Opustitev te za uporabnike izredno koristne možnosti je bila žal potrebna, da so se lahko uskladili pogledi raznih PTT organizacij na snovanje javnih paketnih mrež. Uporabniki raziskovalnih mrež so se znašli v težkem položaju, kajti transportne postaje, ki so jih razvili za mreže na principu datagramov, imajo vprogramirane funkcije, za katere javna mreža ni prilagojena. Še več, naraščajo napačna mnenja, da ob X.25 sploh ni potrebna transportna služba. V teh nesrečnih okoliščinah je malo verjetno, da bo sprejet dogovor o splošni transportni službi, ki bo omogočila sočasen prenos serij paketov med poljubnima računalniškima sistemoma v javni mreži. Kaže sicer, da bodo nekatere raziskovalne skupine uspele združiti svoje rešitve v enoten standard, ki pa bo koristil predvsem bodočim uporabnikom javnih mrež(glej 13).

Obstajata še dva nivoja protokola, za katera menijo raziskovalci, da sta pomembna. To sta protokol za navidezne terminale in protokol za prenos nizov. Protokol za navidezne terminale je razširitev koncepta protokola za upravljanje skupin terminalov in ima v tem pogledu enak pomen kot tri znana X priporočila. Toda cilji vodilnih navideznih terminalov so precej bolj ambiciozni, kajti namen je omogočiti večini terminalov interakcijo s poljubno vrsto računalniškega sistema. Osvojen je princip prilagajanja računalniškega sistema za upravljanje hipotetičnega terminala z obsežno serijo značilnih lastnosti (glej 14). Vsak računalniški sistem naj bi urejal pretok informacij med lastnimi terminali in tem hipotetičnim terminalom. To je možno spričo razpoložljive računalniške zmogljivosti.

Razvoj protokola za navidezne terminale v veliki meri izpodriva napredek v mikroelektroniki, ki napoveduje pohod cenenih inteligenčnih terminalov. Poglavitni problem pri navideznih terminalih je prilagoditev različnih vrst realnih terminalov na hipotetične terminalske specifikacije. Večina problemov izgine, če je terminal inteligenčni, ker so specifikacije lahko poljubne in je to popolnoma interen komunikacijski problem. Dejanska izmenjava informacij med dvema inteligentnima terminaloma potem poteka z izmenjavo nizov. S tem postane aktualen protokol za prenos nizov, o katerem raziskovalci precej razpravljajo (glej 15).

PROTOKOLI ZA MANIPULACIJO Z NIZI

V splošnem vključujejo protokoli za prenos nizov osnovni prenosni mehanizem in kanonično strukturo niza, v katero se pred oddajo uredijo biti, ki so bili pred tem urejeni po formatu oddajnega računalnika. Pri sprejemu se niz iz kanonične oblike prevede v format sprejemnega računalnika. Na ta način se nizi prenašajo med sistemi v kanonični obliki in posamezen sistem mora poznati le algoritme za prevod v svoj format in iz njega. Pomembna prednost tega pristopa je sposobnost zanesljivega prenosa z možnostjo povratka na dogovorjene stadije komunikacijskega procesa, kajti oba računalnika lahko prilagodita svoje akcije na kanonično strukturo niza. Toda v praksi je težko izbrati univerzalno kanonično obliko.

Če različni računalniški sistemi izmenjavajo nize na enak način ali so člani iste organizacijske enote, lahko prikrojimo njihov software tako, da bodo prenašani nizi razumljivi vsakemu od njih. Vsaj v principu in verjetno tudi v praksi pri nekaterih računalniških sistemih in nekaterih organizacijah. Toda pri odprtem delu preko javnih podatkovnih mrež bodo naraščale zahteve po izmenjavi nizov med različnimi organizacijami, ki imajo različne sisteme in načine poslovanja. Predlagani protokoli bodo brez dvoma omogočili prenos nizov preko javne mreže, dosti teže pa bo razbrati pomen nizov na naslovni strani. Za to bo potreben nekakšen protokol za interpretacijo niza ali bolje družina protokolov, ki bodo bazirali na notranji strukturi originalnih nizov. Večina nizov ima tako strukturo, da jo konvencionalni protokol ignorira. Lahko pa jo s pridom uporabimo, če ugotovimo skupne elemente vseh predstavitev, ki se pojavljajo v normalnih sistemih.

Kot primer vzemimo tekst v naravnem jeziku, ki ga en uporabnik sporoča drugemu. Pomen teksta se

ne spremeni, če spremenimo vrstni red im poskrbimo, da se fraze, stavki in odstavki ohranijo. Resnično, to velja tudi pri prevodu v drug naraven jezik. Tako si lahko izmislimo protokol za prenos čistega teksta, zasnovan na točkah za popravljanje na primer na mejah odstavkov. Tak protokol bi lahko pred oddajo dodajal odstavkom (uporabniku neznana) kontrolna števila in jih po uspešnem prenosu odstranjeval.

Prepričajmo se še na primeru prenosa nizov v jeziku BASIC. Pri organizaciji kontrolnih mehanizmov bi se lahko oprli na definirano strukturo BASIC-a, pri popravljanju bi se na primer vračali na določeno vrstico.

Zdi se, da so raziskovalci zelo malo delali na standardih za interpretacijo nizov, nekaj napredka pa je bilo v komercialnem svetu, kjer zdaj razpravljajo o standardih za predstavitev informacij (glej 16).

PROTOKOLI ZA KOMPRESIJO NIZOV

V komercialnih sistemih bodočnosti bo verjetno potreben še en dogovor glede prenosa nizov. Treba bo izločiti le čiste koristne podatke, da se bo zmanjšala količina informacij, prenašanih od enega sistema k drugemu. Zahteva bo še pomembnejša pri uporabi javnih mrež, kjer se zaračunava količina prenašanih informacij. Na primer kadar stranka v eni organizaciji naroči članek pri službi druge organizacije.

En način bi bil, da naročnik oblikuje niz po pravilih svoje organizacije, ga napolni s potrebnimi podatki in ga končno odpošlje dobavitelju. Varianta bi bila, da bi zahteval cenik uslug, toda to naj bi urejal poseben mehanizem. Operater (ali njegov lokalni računalniški sistem) bi po obliki niza ugotovil, da predstavlja naročilo, ga primerno obdelal in odgovoril s fakturo. Ta bi bila oblikovana po pravilih dobaviteljeve organizacije. Naročniku bi jo vrnili po enakih metodah, kot smo jih spoznali pri oddaji naročila, to se pravi po konvencionalnem protokolu za prenos nizov, o katerem smo že razpravljali. Cela procedura bi bila le avtomatizem ročnih procesov, ki ga razvijajo že mnogo let.

Alternativni pristop bi bil, da bi idetificirali bistvene informacije v tipičnem postopku naročanja in fakturiranja in izmenjavali med sistemoma dveh organizacij le te skrčene podatke. Naročnik bi še vedno splošno uporabljal naročilnice v svojem lokalnem formatu (kot fizične papirnate ali kot prikaz na vhodnem terminalu), toda prenašali bi le ekstrakt bistvenih informacij, ki bi jih sprejemni sistem rekonstruiral. Dobavitelju bi bile lahko predstavljene v formatu njegove organizacije in sploh ni nujno, da bi bile podobne originalni naročilnici. Ta druga vrsta procedure je precej tuja današnjim metodam in zahteva dosti višjo stopnjo soglasja med sodelujočima stranema. Toda končno bo potreba po cenejšem komuniciranju močno vzpodbudila sklepanje takih dogovorov.

Zgornji premislek nam kaže, da bodo dolgoročno gledano potrebni novi protokoli za prenos specifičnih informacij pri vsaki posebni interakciji. Jasno, da morajo biti ti protokoli odvisni od namena uporabe, a v nekaterih pogledih jih bo morda mogoče posplošiti. Najprej je potrebna s[illegible]ndardna abeceda za začetek dialoga med sistemi. Tukaj je bil verjetno dovolj dobro sprejet ISO sedembitni kod. Druga potreba je metoda poimenovanja razredov protokolov, da bo sprejemnik lahko izbral pravilnega, tretja pa splošno sprejeta sekvenca za razveljavljanje, tako da se postopek lahko vrne na izhodiščni nivo izbire protokola, če se zdi, da nekaj ni prav.

IZDAJANJE STANDARDOV

Ob vedno večji kompleksnosti standardov naraščajo težave pri doseganju soglasja, ker nesporazumi povzročajo odlaganje sklenitve dogovora o preciznem izražanju. Res je preciznost pri tekstih v naravnem jeziku nemogoča in treba se bo izražati bolj formalno, da se bomo izognili dvoumnostim.

Uporaba formalnega opisovanja bo olajšala avtomatično določanje vrste protokola in morda celo omogočila prenos standardov v programe, ki bodo implementirani v določeno področje hardwarea. Žal ni verjetno, da bodo tvorci standardov sprejeli metode formalnega opisovanja v doglednem času.

Kadar postane interakcija zelo kompleksna, je vprašljivo, če je sprejemanje takih standardov sploh smotrno. Alternativni pristop je, da bi uporabili vse razpoložljive standarde za vzpostavitev zvez, prenos nizov itd. in s pomočjo računalnika sestavili protokol za vsako novo situacijo. Za to mora biti v "glavi" ali "telesu" niza dovolj informacij, da se sprejemnik lahko odloči, kako bo niz interpretiral. Potem se na-

piše ustrezen program in se naslednji podobni nizi sprejemajo avtomatično.

Pri tem mnogo obeta nadomestitev kodiranih formatnih znakov, ki se zdaj splošno uporabljajo za kontrolo lege podatkov, s frazami v naravnem jeziku. To daje snovalcu niza svobodo, da lahko svoje namere opiše na poljuben razumljiv način. Za lažjo identifikacijo bi bile fraze lahko omejene z dvema univerzalno predpisanima znakoma, na primer z () in (). Namesto znaka za preskok na novo stran bi pisalo kar ()'SKOČI NA NOVO STRAN'() ali karkoli, če bi le imelo pravilen pomen za osebo, ki gleda niz kot zaporedje znakov.

Na osnovi tega obrazca lahko sistemski programer na sprejemni strani analizira niz tujega oddajnika in določi, kaj naj se pokaže na prikazovalniku, ali niz pravilno procesira. Lahko bi sestavil lokalno proceduro formatiranja za procesiranje poljubnega niza iz tega oddajnika, če bi privzel, da se pravila oddajanja ohranjajo. V bistvu to kaže nezmožnost sedanjih tehnik, na primer umetne inteligence, da bi dale avtomatizirano metodo interpretacije nizov, in daje priložnost človeški inteligenci, da se izkaže, ko je najbolj potrebno. Če bodo razvite standardne oblike nizov za posebne namene, bo to zasluga začetnih izkušenj s takimi protokoli proste oblike(glej 17).

Če bodo standardni nizi definirani kar " iz zraka", toliko bolje! Toda ta pristop odpira možnosti za odprto delo med sistemi, ki bi radi kontaktirali, pa se njihovi projektanti niso dogovorili o podrobnostih. To je posebno pomembno v zvezi z vsesplošnim uvajanjem domačih računalnikov, ker na tem področju nihče ni odgovoren za izdelavo standardov za komuniciranje preko javne mreže, pa se zdi, da je še najbolj modro prepustiti njihov razvoj evoluciji.

BODOČE USMERITVE

Članek je prikazal, kako je napredek tehnologije pospešil razvoj računalniških mrež z decentraliziranimi sistemi. V teh mrežah se bo vedno več uporabnikovih zahtev procesiralo lokalno ob avtonomni izmenjavi nizov med lokalnimi sistemi. Ta usmeritev se bo nadaljevala, ker prihajajoče izboljšave v mikro elektroniki kažejo, da je kompleksnost sistemov mogoče še povečati. Računalniške enote bodo dobile več procesorske in spominske moči ob enaki ceni. Zato lahko predvidevamo, da bodo nekateri problemi pri usklajevanju različnih vrst terminalov rešeni z zelo obširnimi lokalnimi procesnimi operacijami.

Govorna enota na primer bi izvajala lokalno analizo zvoka in omogočila govorno/slušno interakcijo uporabnika in terminala pri kreiranju lokalne datoteke, ne da bi bilo treba originalni govor prenašati v centralni sistem za razpoznavanje. In kopirni terminal bi lahko lokalno prepoznaval tekst in tvoril datoteko teksta namesto moduliranega vala. Takrat bodo velike črne škatle v naših domovih postale resničnost. Morda bodo to majhne črne škatle, ki jih bo mogoče spraviti v žep in bodo vsebovale tudi radio. S slikovnim vhodom in izhodom preko solidne kamere in prikazovalnika, z govornim vhodom, da se bomo lahko pogovarjali z drugimi ljudmi in njihovimi računalniki. Morda bo leta 1984 za to še prezgodaj, toda žepni kalkulator se je že razvil v priročnega učitelja pravopisa in jezikovnega prevajalca z govornim izhodom, tako da ni mogoče napovedati, kaj bo sledilo.

SKLEP

Ta članek je podal pregled razvoja javnih podatkovnih mrež in obravnaval, kako se današnja stopnja standardizacije kosa z zahtevami uporabnikov mrež. Verjetno je jasno, da se bodo relativno hitro razširile preprostejše aplikacije, ki so rezultat prenosa principov današnjih privatnih zvezdastih mrež v javno uporabo. Bolj kompleksno uporabo mrež pa bo oviralo pomanjkanje ustreznih standardov za prenos nizov med sistemi.

Problem ni primarno tehnične narave, kajti zahteve in rešitve za omogočitev učinkovitega prenosa nizov so večinoma znane. Težava je pri sklepanju dogovorov med zagovorniki različnih protokolov. Vsi predlogi imajo dobre in slabe strani, zato ni jasnih argumentov za izbiro enega ali drugega protokola. Izkušnje kažejo, da je v taki situaciji rešitev spora stvar sreče in je odvisna od tega, kateri protokoli imajo večjo politično in komercialno podporo. Zanimivo je, da tako pri vsem izbiranju sploh ne gre več za to, kakšna rešitev bo prevladala. Protokoli bodo realizirani z mikro kodom in nekoč morda celo s LSI čipi. Pomembno vprašanje torej ni, kateri protokol naj izberemo, ker to lahko najbolje naredimo z metanjem kovanca. Važno je, da se hitro zedinimo pri uvajanju vseh novosti, ki omogočajo učinkovito komuniciranje.

REFERENCES

1. Davies, D W and Barber, D L A
Communication Networks for Computers
John Wiley, 1973.

2. Roberts, L
Computer Network Development to achieve Resource Sharing
AFIPS proceedings SJCC, Vol. 36, 1970

3. Scantlebury, R L and Wilkinson, P T
The NPL Data Communications Network
ICCC proceedings, Stockholm, 1974

4. Pouzin, L
The CYCLADES network - present state and Development Trends
Sym. Com. Net. Trends and Applications
IEEE (1975) 8-12

5. Sarbinowski, H
Network Activities in the Federal Republic of Germany; Pilots to a Public Packet Switching Service.
Telecommunications, Oct 1977.

6. Barber, D L A
EIN - An Example of Co-operative Research in Europe.
Altafrequenza, July 1979.

7. Euronet News
Directorate General Scientific and Technical Information,
Commission of the European Communities,
Luxembourg.

8. Public Data Networks
CEPT/Eurodata Foundation 1978

9. Provisional Recommendations X.3, X.25, X.28 and X.29 on Packet Switched Data Transmission Services
CCITT, Geneva, 1978.

10. ISD/TC97/SC16 N117
Reference Model of Open Systems Architecture (Version 3 November 1978)

11. Davies, D W et al
Computer Networks and their Protocols
John Wiley, 1979.

12. Hertweck, F R, Raubold, E and Vogt, F
X.25 Based Process-Process Communications
Proc. Symposium on Computer Network Protocols, Liege, 1978

13. A Transport Service
Post Office Study Group 3
DCPU, NPL, 1979.

14. Barber, D L A
The Role and Nature of a Virtual Terminal
Computer Communications Review,
Vol.7, No.3.

Proposal for a Standard Virtual Terminal Protocol
IFIP WG6.1, INWG Protocol Note 91, 1978

15. Neigus, N
File Transfer Protocol
ARPANET Protocol Handbook, 1973

A Basic File Transfer Protocol
Post Office Study Group 2
DCPU, NPL, 1979,

16. ISO/TC97/SC14/WG1 N25
The Development of Data Standards for use in Information Processing and Interchange

17. Barber, D L A
"#<"You Don't Know Me, But....">#"
Computer Weekly, March 1979

Barber, D L A
Protocols for Intelligent Terminals
Computer Communications Review
Vol. 9 No. 3 July, 1979.

INFORMATICA 3/1979

KODIRANJE IN DEKODIRANJE KOREKTURNEGA KODA Z MIKRO RAČUNALNIKOM

M. ŠUBELJ
J. KORENINI
F. NOVAK
R. TROBEC

UDK: 681.327.63 : 621.3

INSTITUT JOŽEF STEFAN, LJUBLJANA

V članku je obravnavano kodiranje in dekodiranje korekturnega koda realizirano z mikro računalnikom. Najprej je definirana Hammingova razdalja in princip dekodiranja "idealni opazovalec". Tipi korekturnih kodov so razdeljeni na kode, ki e napak popravijo in kode, ki (e-1) napak popravijo in e-to napako detektirajo.
Primer prikazuje princip oddaje kodiranih znakov in sprejem in dekodiranje znakov. V zbirnem jeziku za procesor F-8 sta dodana tudi programska modula za kodiranje in dekodiranje.

CODING AND DECODING OF A CORRECTION CODE BY A MICROCOMPUTER - Coding and decoding of a correction code implemented in microcomputer is discussed in the paper. The definition of "Hamming distance" is given and the "ideal observer" decoding technique is described.
Different correction codes are classified as e-error-correcting codes and (e-1)-error-correcting, e-error-detecting codes.
An example of transmission and reception of correction codes in practice is shown. Program modules for coding and decoding written in F8 assembly language are also given.

1. UVOD

Pri prenosu informacije preko telefonskega kabla se pojavi problem zanesljivega prenosa. Motnje, ki jih lahko povzročijo preklopi na liniji, distorzija signala, atmosferske motnje, presluh med linijama, lahko povzročijo, da bit informacije spremeni svojo električno vrednost.

Za zanesljiv prenos poznamo več metod kontrole napake pri prenosu. Mi se bomo omejili na kontrolo napake, ki omogoča popravljanje napake, v nekaterih primerih pa samo odkrivanje. Izvedba je programska in nam omogoča veliko prilagodljivost in relativno enostavno modifikacijo na drugačen kod. Znaki se lahko prenašajo paralelno (vsi biti znaka istočasno) ali serijsko. Sinhronizacija in paralelno serijska in serijsko paralelna pretvorba je lahko izvedena aparaturno (ACIA) ali programsko.

2. KOREKTURNI KOD IN RAZDALJE

Predpostavlja se, da se informacija prenaša preko simetričnega binarnega kanala. Vhod v kanal tvori zaporedje binarnih kodnih besed $W = \{ w_1, w_2, \ldots, w_S \}$ dolžine n. Vse besede imajo isto verjetnost, da pridejo pri oddaji na vrsto. Na izhodu iz kanala imamo sprejemnik, ki se odloči po pravilu "idealnega opazovalca".

Dekoder "idealni opazovalec" se odloča za tisto varianto, kjer je verjetnost napake najmanjša ali drugače povedano, za tisto besedo koda $W = \{ w_1, \ldots, w_s \}$, do katere je Hammingova razdalja najmanjša. Hammingova razdalja $d(w_1, w_2)$ med dvema binarnima besedama w_1 in w_2 dolžine n je število binarnih znakov, v katerih se razlikujeta besedi w_1 in w_2.

Predpostavimo, da je zaradi motenj možna napaka na vsakem binarnem znaku. Ker je vhodna beseda dolžine n, lahko na izhodu sprejmemo 2^n različnih besed dolžine n.

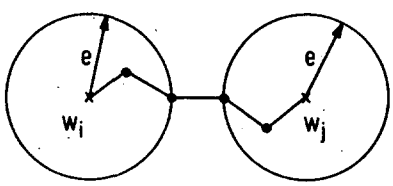
Predpostavimo, naj vir pošilja zaporedje besed

$W = \{w_1, w_2, \ldots, w_s\}$ in sprejemnik sprejema zaporedje besed $V = \{v_1, v_2, \ldots, v_{2^n}\}$.

Zanima nas, kakšna je povezava med odposlanimi in sprejetimi besedami. Oglejmo si Hammingovo razdaljo med besedami koda $W = \{w_1, w_2, \ldots, w_s\}$:

a) $d(wi, wj) = 2e + 1 \quad i = 1,2,\ldots,s \quad i \neq j \quad j = 1,2,\ldots,s$

Predočimo si še grafično razdaljo med besedama wi in wj.

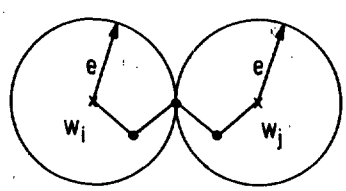


$d(wi, wj) = 5 \Rightarrow e = 2$

Vidimo,da vsaka sprejeta beseda $vj \in V$ $j=1,2,\ldots, 2^n$ pade v okolje samo ene izmed besed $W = \{w_1, w_2, \ldots, w_s\}$ s Hammingovo razdaljo $d(wi, vj) \leq e \quad i = 1,2,\ldots s$ in $j = 1,2,\ldots,2^n$. Naloga dekoderja "idealni opazovalec" je, da najde to besedo $wi \in W$, $i = 1,2,\ldots,s$. Beseda dekodirana po tem principu je bila pravilno dekodirana v odposlano besedo, če je bila napaka e ali manj kratna.

b) $d(wi, wj) = 2e \quad i = 1,2,\ldots,s \quad i \neq j \quad j = 1,2,\ldots,s$

Predočimo si grafično razdaljo med besedama.



$d(wi, wj) = 4 \Rightarrow e = 2$

Oglejmo si Hammingovo razdaljo med sprejeto besedo $vj \in V \quad j = 1,2,\ldots,2^n$ in besedami koda $W = \{w_1, w_2, \ldots, w_s\}$.

Razdalja je:

- $d(wi,vj) = e$ ali $\quad i = 1,2,\ldots,s \quad j = 1,2,\ldots,2^n$
- $d(wi,vj) = d(wk,vj) = e \quad i = 1,2,\ldots,s \quad j = 1,2,\ldots,2^n \quad k = 1,2,\ldots,s$

V prvem primeru se lahko odločimo. Sprejeto besedo vj, dekodiramo kot wi. Beseda je pravilno dekodirana, če je bila napaka e-l ali manj kratna.

V drugem primeru besede ni možno dekodirati, saj je razdalja besede vj enaka do dveh besed wi in wk. Napako se lahko samo detektira.

3. PRINCIP ODDAJE IN SPREJEMA KOREKTURNEGA KODA

Zgled obravnava primer prenosa informacije s kodom 4 korekturnih in 4 informacijskih bitov. Hammingova razdalja med besedami koda je 4, zato tak kod popravlja enojne in detektira dvojne napake na 8 bitih. Za 8 bitno besedo smo se odločili zaradi najpogostejše aparaturne organizacije mikroračunalniških komponent (CPU, ACIA). Seveda je možno izbrati tudi kod, ki bo imel drugačno razmerje med informacijskimi in korekturnimi biti pri enaki dolžini besede.

V primeru koda 4 informacijskih in 4 korekturnih bitov oddajnik odda zlog (8 bitov) informacije kot dve 8 bitni kodi. Najprej kodira zgornje 4 bite informacije s Hammingovim kodom (8 bitov) in ga odda. Za tem kodira še spodnje 4 bite informacije in kod odda.

Sprejemnik dela v obratnem vrstnem redu. Sprejme kod (8 bitov),ga dekodira (4 biti) in shrani na zgornja 4 mesta vmesnika informacije. Za tem sprejme še drugi kod, ga dekodira in prida na spodnja štiri mesta vmesnika informacije. Modula za kodiranje in dekodiranje se sklicujeta na isto tabelo Hammingovih kodov. Osnovni moduli za sprejem in oddajo so napisani kot subrutine. Ogljemo si jih podrobneje opisane v psevdo kodu:

a) ODDAJA:

TRBY: subrutina odda zlog (8 bitov) informacije kodirane s kodom Hammingove razdalje 4.
Vhodni parameter : adresa lokacije, iz katere bo zlog oddan.

```
SUBROUTINE TRBY
  VLOŽITEV ZLOGA;
  POMIK ZA 4 MESTA V DESNO;
  CALL KOHE
  CALL TRC
  VLOŽITEV ZLOGA;
  OHRANITEV SP. 4 BITOV;
  CALL KOHE
  CALL TRC
ENDSUBROUTINE
```

KOHE : subrutina 4 bitom priredi kod Hammingove razdalje 4 (8 bitov).
Vhodni parameter: 4 informacijski biti v reg. DIO.
Izhodni parameter: kod Hammingove razdalje (8 bitov) v reg. DIO.

```
SUBROUTINE KOHE
  ADRESIRANJE ZAČETNE ADRESE TABELE
    HAMMINGOVIH KODOV;
  RELATIVNI PREMIK ADRESE ZA 4 INF. BITE
    (IZRAČUN ADRESE KODA);
  VLOŽITEV IN SHRANITEV KODA;
ENDSUBROUTINE
```

TRC : subrutina odda znak na izhod za oddajo.
Vhodni register: znak (Hemmingov kod) v reg. DIO.

```
SUBROUTINE TRC
  DOUNTIL (MOŽNOST ODDAJE ZNAKA)
  ENDDO
  ODDAJA ZNAKA NA IZHOD;
ENDSUBROUTINE
```

b) SPREJEM :

REBY : sprejme zlog (8 bitov) informacije kodirane s kodom Hammingove razdalje 4.
Vhodni parameter je adresa lokacije, kamor bo sprejeti zlog shranjen.

```
SUBROUTINE REBY
  CALL REC
  CALL DEHE
  SHRANITEV ZG. 4 BITOV ZLOGA;
  CALL REC
  CALL DEHE
  SHRANITEV SP. 4 BITOV ZLOGA;
ENDSUBROUTINE
```

DEHE : subrutina kod Hammingove razdalje 4 (8 bitov) dekodira v 4 bite. V primeru, ko se ne more odločiti (d)wi,vj) = d(wk,vj)) postavi zastavico "programska detekcija napake".
Vhodni parameter: kod Hammingove razdalje v reg. DIO.
Izhodni parameter: kod dekodiran v 4 informacijske bite v reg. DIO ali DIO nespremenjen in zastavica "programska detekcija napake" v reg. DRS.

```
SUBROUTINE DEHE
  ŠTEVEC KODOV=0;
  ADRESIRANJE ZAČETKA TABELE HAMMINGOVIH
    KODOV;
  DOUNTIL (TEST VSEH KODOV ALI DEKODIRANJE)
    ŠTEVEC BITOV=8
    EXCLUSIVNI OR MED SPREJETIM ZNAKOM IN
      ADRESNIM ZNAKOM TABELE KODOV
      (ADRESIRANJE LOKACIJE NASLEDNJEGA KODA)
    IF (STA ENAKA, REZULTAT=0)
      THEN
        DEKODIRANI SPREJETI ZNAK=ŠTEVEC KODOV,
          IZTOP;
      ELSE
        DOUNTIL (POJAVITEV 1. ENICE V REZULTATU)
          PRIPRAVA NASLEDNJEGA MESTA;
          ŠTEVEC BITOV=ŠTEVEC BITOV-1;
        ENDDO
        DOUNTIL (TEST OSTALIH MEST REZULTATA)
        ENDDO
          IF (NI VEČ ENIC)
            THEN
              SPREJETI ZNAK=ŠTEVEC KODOV,
                IZTOP;
            ELSE
              ŠTEVEC KODOV=ŠTEVEC KODOV+1;
          ENDIF
    ENDIF
  ENDDO
    POSTAVITEV ZASTAVICE PROG.DET.NAPAKE;
ENDSUBROUTINE
```

REC : subrutina sprejme znak preko vhoda za sprejem
Izhodni parameter: sprejeti znak (Hammingov kod) v reg. DIO.

```
SUBROUTINE REC
  DOUNTIL (MOŽNOST SPREJEMA ZNAKA)
  ENDDO
  SPREJEM ZNAKA PREKO VHODA;
ENDSUBROUTINE
```

4. SKLEP

Cilj članka je predstaviti možnost prenosa podatkov kodiranih s konkretnim kodom pri komunikaciji med mikro računalniki. Ta dograditev, sicer zmanjša množino koristne informacije prenešene v časovnem intervalu, vendar poveča zanesljivost prenosa in izboljša kvaliteto kanala.

5. LITERATURA

/1/ L. Gyergyek: Statistične metode v teoriji sistemov, teorija o informacijah, Fakulteta za elektrotehniko 1971.

/2/ Electronics Book Series: Basics of data communications, Mc Graw-Hill 1976.

/3/ Mostek: Programming Guide


```
                *       1.2.1979 M.SUBELJ
                *
                        TITLE  SUBRUTINA  KODIRANJE HAMMINGA
                *
                *
                *
                *SUB.KOHE DOLOCI 4 INFORMACIJSKIM BITOM KOD
                *HAMMINGOVE RAZDALJE 4.KOD SESTAVLJAJO 4
                *INFORMACIJSKI BITI + 4 KOREKTURNI BITI.
                *
                *
                *
                *VHOD: DIO = 4 INFORMACIJSKI  BITI  (SP.4 MESTA)
                *
                *IZHOD: DIO = KOD HAMMINGOVE RAZDALJE
                *
                *UPOR.REG.:ACC,W,K,DIO
                *
                *
                DIO     EQU     H'07'           PODATKI V I/O SUB .
                *
                *
                *
0000 08         KOHE    LR      K,P
                *
0001 2A 00 34           DCI     TAHEM           VLOZITEV ADRESE TABELE KODOV
0004 47                 LR      A,DIO
0005 8E                 ADC                     IN ADRESIRANJE KODA
                *
0006 16                 LM                      VLOZITEV IN
0007 57                 LR      DIO,A           SHRANITEV KODA
                *
0008 0C                 PK
                *
                        TITLE  SUBRUTINA DEKODIRANJE HAMMINGA
                *
                *
                *
                *SUB. DEHE DEKODIRA KOD HAMMINGOVE RAZDALJE 3
                *( 7 BITNA BESEDA) ALI RAZDALJE 4 (8 BITNA BESEDA).
                *PRI KODU RAZDALJE 3 POPRAVI ENOJNO NAPAKO.
                *PRI KODU RAZDALJE 4 POPRAVI ENOJNO NAPAKO IN
                *DETEKTIRA DVOJNO NAPAKO.
                *
                *
                *
                *VHOD: DIO= KOD HAMMINGOVE RAZDALJE
                *
                *
                *NAPAKA  0 REDA ALI 1. REDA
                *
                *IZHOD: DIO= DEKODIRANA BESEDA (SPODNJI 4.BITI)
                *
                *NAPAKA VEC KOT 1.REDA
                *
                *IZHOD: DIO= NESPREMENJEN
                *       ERFL= B'XXX1XXXX' FLAG NAPAK
                *
                *UPOR.REG.:ACC,W,K,XU,XL,DRS,DIO
```

```
                        *
                        *
                        *
                        XU      EQU     H'00'           DELOVNI REGISTER
                        XL      EQU     H'01'           DELOVNI REGISTER
                        DRS     EQU     H'05'           PODATKI V SUB. LRS  IN  SRS
                        DIO     EQU     H'07'           PODATKI V I/O SUB .
                        *
                        *
                        *
0009 08                 DEHE    LR      K,P
                        *
000A 70                         LIS     H'00'
000B 50                         LR      XU,A            VLOZITEV STEVCA KODOV XU=0
                        *
000C 2A 00 34                   DCI     TAHEH           ADRESIRANJE ZACETKA TABELA KODOV
                        *
000F 78                 DEHE0   LIS     H'08'           DOUNTIL (TESTIRANJE VSEH KODOV)
0010 51                         LR      XL,A            VLOZITEV STEVCA BITOV XL=8
                        *
0011 47                         LR      A,DIO           VLOZITEV SPREJETEGA ZNAKA IN
0012 8C                         XM                      PRIMERJANJE Z ZNAKOM TABELA
0013 84 11                      BZ      DEHE3           IF ( STA ENAKA) THEN3 DEHE3,ELSE...
                        *
0015 21 FF              DEHE1   NI      H'FF'           DOUNTIL (POJAVITEV 1.ENICE)
0017 91 09                      BM      DEHE6           IF (ENICA) THEN DEHE6,ELSE...
                        *
0019 13                         SL      1               PRIPRAVA NASLEDNJEGA BITA
001A 31                         DS      XL              STEVEC BITOV XL=XL-1
001B 90 F9                      BR      DEHE1           ENDDO
                        *
001D 21 FF              DEHE2   NI      H'FF'           TESTIRANJE ALI JE 2.ENICA
001F 91 09                      BM      DEHE4           IF (ENICA) THEN DEHE4,ELSE...
                        *
0021 13                 DEHE6   SL      1               (DEHE6 PRIPRAVA NOVEGA MESTA)
0022 31                         DS      XL
0023 94 F9                      BNZ     DEHE2           IF (TEST VSI BITI) THEN...,ELSE DEHE2
                        *
0025 40                 DEHE3   LR      A,XU            VLOZITEV DEKODIRANE BESEDE
0026 57                         LR      DIO,A
0027 90 0B                      BR      DEHE5           IZHOD IZ SUB.
                        *
0029 40                 DEHE4   LR      A,XU
002A 1F                         INC
002B 50                         LR      XU,A            POVECANJE STEVCA KODOV IN TEST
002C 23 10                      XI      H'10'           ALI SO TESTIRANI VSI  KODI
002E 94 E0                      BNZ     DEHE0           ENDDO
                        *
0030 20 10                      LI      H'10'           POSTAVITEV FLAG-A PROGRAMSKE
0032 55                         LR      DRS,A           DETEKCIJE NAPAKE PRI SPREJEMU
                        *
0033 0C                 DEHE5   PK
                        *
                        *
                        *                               TABELA KODOV (8 BITNA KODA)
                        *
0034 00                 TAHEH   DC      H'00'           W0
0035 D2                         DC      H'D2'           W1
0036 55                         DC      H'55'           W2
0037 87                         DC      H'87'           W3
0038 99                         DC      H'99'           W4
0039 4B                         DC      H'4B'           W5
003A CC                         DC      H'CC'           W6
003B 1E                         DC      H'1E'           W7
003C E1                         DC      H'E1'           W8
003D 33                         DC      H'33'           W9
003E B4                         DC      H'B4'           W10
003F 66                         DC      H'66'           W11
0040 78                         DC      H'78'           W12
0041 AA                         DC      H'AA'           W13
0042 2D                         DC      H'2D'           W14
0043 FF                         DC      H'FF'           W15
                        *
                        *
                        *
                      END
//NUMBER OF ERRORS=  0
```

INFORMATICA 3/1979

ISKUSTVA U OBRADI PODATAKA NA MALIM POSLOVNIM SISTEMIMA

W. JURIŠIĆ-KETTE

UDK: 681.3.06

RIZ – OD ZAGREG, FILOZOFSKI FAKULTET, ZAGREB

Definiraju se zadaci i dijelovi poslovne obrade, te podobnost malih poslovnih sistema za tu svrhu. Ističu se prednosti malih sistema zbog sjedinjenja svojstava strojne i ručne obrade. Dana je kratka analiza utroška vremena za primjer fakturiranja. Prikazana je ideja organizacije programskih paketa iz ostvarene proizvodnje.

SOME EXPERIENCE WITH DATA PROCESSING ON THE SMALL BUSINESS ORIENTED SYSTEMS: The area and elements at the business data processing as well as the fitness at the small business system for the said use are overviawed. The advantage of the small systems because of merging the properties of machine and manual processing is pointed out. In the example of the invoice generation the short analysis of the required processing time is given. The basic idea of the organisation of the realised software packages is presented.

ZADACI POSLOVNIH OBRADA

Osnovni zadaci poslovne obrade su generiranje i bilježenje poslovnih dogadjaja, te izvodjenje zaključaka na bazi promjena stanja uvjetovanih tim dogadjajem. Poslovni dogadjaj se odnosi na odredjeni subjekt poslovanja i mjerljiv je, te se izražava numeričkim pokazateljima. Jedinice mjere mogu biti financijske (din.,DM.,..) ili količinske (kg., kom., bod. i sl.). Na temelju postojećeg stanja poslovanja moguće je izvesti različite zaključke koji rezultiraju u izlaznim dokumentima. Algoritam za izvodjenje zaključaka je specifičan za svaki tip obrade i djelatnost. Principi obrade poslovanja bilo koje jedinke udruženog rada su neovisni o djelatnosti ili vrsti poslovnih subjekata. Općeniti pristup problematici poslovne obrade ukazuje na tri dijela obrade:

1. Generiranje poslovnih dogadjaja
2. Formiranje i nadopuna evidencije stanja
3. Gradnja i upotreba informacijskog sistema.

Realizacija navedenih dijelova zahtijeva brzu i jednostavnu metodu pohranjivanja podataka, te maksimalnu iskoristivost pohranjenih podataka. Nameće se pitanje:"Kako to postići?". Potrebno je odrediti metodu, profil kadrova, opremu za obradu poslovanja svake jedinične organizacije udruženog rada, kao i vertikalno povezivanje obrade u složenoj organizaciji udruženog rada.Rješenje je u izboru jednog od tri postojeća tipa organizacije obrade poslovanja:

1. Ručna obrada
2. Obrada uz primjenu malog poslovnog sistema
3. Obrada uz primjenu velikog računskog sistem

Unutar navedenih tipova postoji razradjena ljestvica tehnika obrade koje ne unose značajne promjene u tekuća razmatranja.

Rad je referiran na XIV.Simpoziju Informatica 79,Bled, 1-6.oktobra 1979.

MOGUĆNOST UPOTREBE I PODOBNOST MALIH SISTEMA ZA POSLOVNU OBRADU

Medju prvim masovnim primjenama računskih sistema istakle su se poslovne obrade. Obrada poslovanja na računskim sistemima donijela je bitne prednosti u rukovanju poslovnim informacijama. Posebno je ubrzala postupak bilježenja i grupiranja podataka. Osnovni parametri u odabiranju stroja za obradu bili su: kapacitet memorije, mogućnost brzog unosa velike količine podataka, te brzog ispisa izlaznih dokumenata i informacija. Organizacija i metodologija obrade mjenjala se s tehnološkim stupnjem razvoja računskih sistema. Od malih sporih sistema nedovoljnog kapaciteta, koji su zahtijevali skupo posluživanje, preko srednjih s poboljšanom uslugom i jednako skupim pogonom dolazimo do velikih sistema. Zbog povećanog kapaciteta i brzine propusna moć sistema raste, a jedinična cijena obrade pada. U radnim organizacijama s izrazito velikim količinama poslovnih dogadjaja u jedinici vremena, te s kompleksnim i glomaznim bazama podataka (banke, gradske skupštine i sl.) veliki sistem donosi poboljšanje u poslovanju. Manje radne organizacije, koje predstavljaju glavninu potrošača, ne mogu ekonomično iskoristiti kapacitete velikog stroja. Slijedi distribucija "kompjuterske energije" na više potrošača. Dobro organizirani pogon smanjuje jediničnu cijenu usluge. Ovakva organizacija davanja kompjuterskih usluga na velikim sistemima omogućava nastajanje računskih centara, kao radnih organizacija uslužnih djelatnosti. Oni opslužuju cijeli niz korisnika. Računski centar odredjuje način izvodjenja obrade podataka, koja pored niza pozitivnih rezultata donosi i odredjene poteškoće organizacione i kadrovske prirode.

Mali sistemi, rezultat tehnološkog napretka, opremljeni su kasetnim i disketnim vanjskim memorijama, pisačima brzine od 10 do 150 znakova u sekundi, zaslonima različitih kapaciteta i tastaturom. Kapaciteti vanjskih memorija uglavnom zadovoljavaju potrebe baze podataka OOUR-a i OUR-a, a često se mogu proširiti diskom kapaciteta od 2,5 Mbajta na više. Brzina pisača zadovoljava potrebe formiranja izlaznih dokumenata. Pisač je često opremljen dodatnim uredjajima za rukovanje sa specifičnim izlaznim dokumentima. Zaslon je posebno pogodan za kontrolu nadopune baze podataka, kao i za izlaz informacija ako se ne zahtijeva pisani dokument. Tastatura sistema je oblikovana tako, da pogoduje kontinuiranoj komunikaciji sa strojem. Osim toga tastatura ima funkciju upravljačke ulazne jedinice.

Ovakav sistem je prihvaćen od velikog broja korisnika zbog slijedećih razloga:

1. Sistem je smješten na izvoru podataka, te predstavlja integralnu jedinku u organizaciji poslovanja.
2. Sistem je jednostavan za rukovanje i nema potrebe za specijaliziranim kadrovima.
3. Logička kontrola ulaznih podataka odvija se istovremeno s unosom, te u slučaju dijagnosticiranja greške izvorni dokument je prisutan u obradi.
4. Vremenski tok obrade ovisi o potrebama i mogućnostima radne organizacije korisnika.
5. Sistem omogućava nadopunu podataka na već postojećem izlaznom dokumentu (knjigovodstvene kartice i sl.).
6. Ne zahtjeva posebno klimatiziran prostor.

Stoga obrada poslovnih podataka na malim poslovnim računalima donosi novu kvalitetu u vodjenju poslovne politike, jer pruža brzo i točno sve relevantne podatke vezane na poslovanje.

Radne organizacije se često nalaze pred dilemom, koji tip obrade primjeniti: ručnu obradu, obradu na malom sistemu, ili obradu u zajedničkom računskom centru t.j. rad na velikom sistemu. Uputno je razmotriti prednosti i nedostatke svakog od tri navedena tipa obrade, i to na primjeru fakturiranja u poslovnoj jedinici. Ostale obrade se u ovom trenutku ne razmatraju. Predpostavlja se veličina računa od 10 do 15 stavaka, a ostali elementi su uobičajeni.

Iskustvo je pokazalo, da je za ručnu obradu računa, kojeg smo uzeli kao primjer, potrebno organizirati tri faze rada. To su: priprema sadržaja, prijepis i kontrola. Obzirom da se u pripremi sadržaja obavlja najveći dio poslova (sastavljanje specifikacije, obračun cijena, poreza i rabata, navodjenje posebnih uvjeta prodaje) to je predvidivo vrijeme za ove radove oko 20 min.. Nakon prijepisa slijedi kontrola, a ona zahtijeva dosta vremena, jer se ponavlja načinjeni obračun.
Izrada tog računa na malom poslovnom sistemu bitno je kraća, i izvodi ga isključivo fakturista. Zaglavlje računa se ispisuje automatski na osnovu postojeće baze podataka koja uključuje i datoteku kupaca. Polje specifikacija se formira iz datoteke robe, a poziva se šifrom robe preko tastature. Završni dio s pripadnim obračunom ispisuje se automatski. Prosječno vrijeme izrade računa je oko 5 min.. Kontrola

je vrlo jednostavna i brza, a izvodi se u momentu kada je izvorni dokument u ruci fakturiste i ne zahtjeva dodatno traženje.
Veliki sistem na ovom zadatku traži organizaciju rada u vremenski i kadrovski odvojenim fazama. Fakturista priprema sadržaj, operator unosnog medija upisuje sadržaj na prenosni medij, dok operater stroja vodi proces obrade na sistemu. Mogućnost nastanka greške je višestruka i to prilikom formiranja sadržaja i upisa na prenosni medij (papirna kartica, traka,disketa ili sl.). Posebnu poteškoću predstavlja razbijenost obrade u odnosu na vrijeme i mjesto rada. Vremenski ritam izrade računa u centru odredjuje centar i odvija se u skupinama.Deklarirano vrijeme izrade računa od nekoliko sekundi samo je prividno. Za fakturistu stvarno vrijeme počinje odlaskom podataka iz odjela u centar i svršava primitkom gotovog računa. Ovo vrijeme mjerljivo je u satima ili danima. Za razliku od navedenog, mali sistem zahtjeva 5 min., ali to je i ukupno vrijeme izrade.

Navedena razmatranja su iznesena u tablici 1. Podaci su iskustveni, te ih treba promatrati kao procjenu veličina, a ne u točnim iznosima.

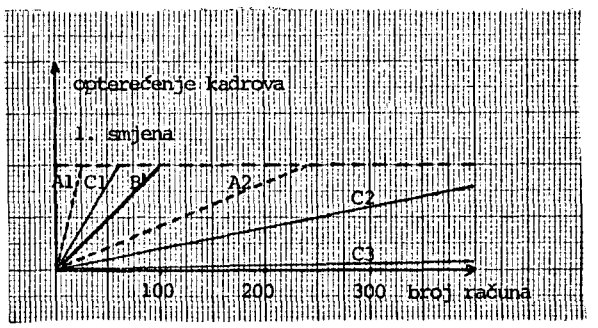
	faze rada	profil kadrova	broj radnika	vrijeme obrade po fazama	ukupno vrijeme (min)	direktna iskoristivost u info.sis.	propusna moć za jednu smj.	približno vrijeme za izradu 100 računa
RUČNA OBRADA	1.priprema sadržaja	fakturista	2	20min.	22	ne	24	4 fak/dan
	2.izrada 3.kontrola	daktilograf		2min.			240	0,4 dak/dan
MALI POSL. SISTEM		fakturista	1	5min.	5	da	96	1 fak/dan
VELIKI SISTEM	1.priprema sadržaja	fakturista		8min.			60	1,66fak/dan
	2.upis na nosilac podataka	operater ulaznog uredjaja	3	1min.	9,1	da	480	0,2 buš/dan
	3.obrada	operater stroja		0,1min.			4800	0,02 op/dan

Tablica 1.

Analiza tablice nameće niz zaključaka:

1. Za svaki tip obrade postoji ograničenje propusne moći.
2. Obrade primjenom stroja nemaju prednost samo u brzini rada, već i u činjenici da one formiraju podatke direktno iskoristive za ostale poslovne zadatke (robno i saldakonti knjigovodstvo, statističku distribuciju robe na tržištu i sl.). Naglašava se da je upravo postojanje baze podataka prvenstveni razlog uvodjenja EOP-a.
3. Svaki od navedenih tipova obrade odredjuje različitu kadrovsku strukturu i opterećenje. U posljednjim kolonama tablice 1. prikazano je opterećenje ljudi i strojeva za izradu 100 računa prosječne veličine od 10 do 15 stavaka.

Na sl.1. pokazano je opterećenje kadrova ovisno o broju računa i tipu obrade s naznakom maksimalnog opterećenja u jednoj smjeni.



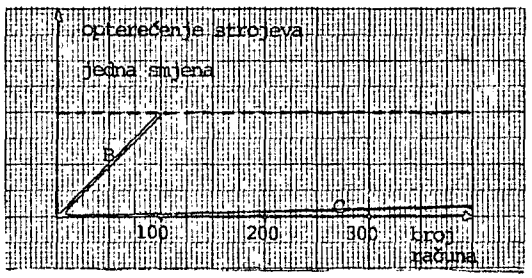
Legenda:

A 1 - Fakturista u ručnoj obradi
A 2 - Daktilograf u ručnoj obradi
B 1 - Fakturista na malom sistemu
C 1 - Fakturista uz rad na velikom sistemu
C 2 - Operater na unosnom mediju
C 3 - Operater velikog sistema.

Sl. 1. Opterećenje kadrova u smjeni

Iz prikaza na sl.1. može se zaključiti da je fakturista na malom poslovnom sistemu najbolje iskorišten. Posebno je značajno da samo jedna osoba izradjuje cijeli račun.

Na slici 2. je prikazan odnos opterećenja strojeva u smjeni ovisno o broju računa.



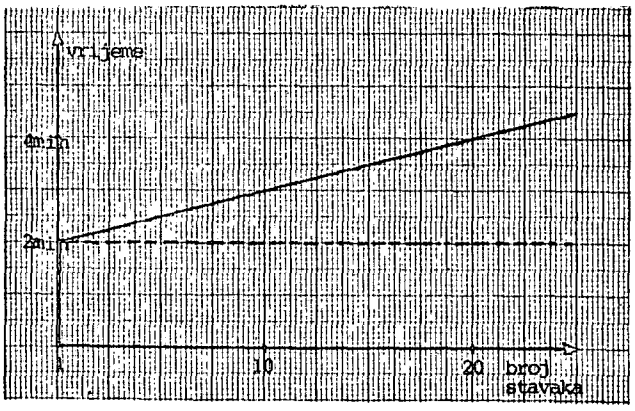
Legenda:

B - mali poslovni sistem
C- veliki sistem

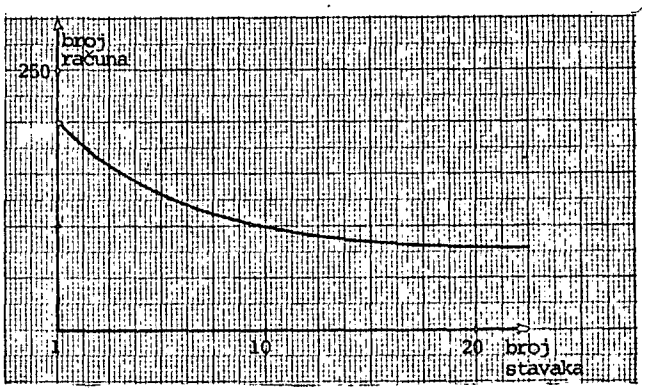
Sl.2. Opterećenje strojeva u smjeni

U slici 2. se vidi da je mali sistem zasićen kod približno 100 računa na dan. Pri izboru tipa obrade treba uzeti u obzir obe komponente kadrovsku i strojnu. S aspekta minimalnog opterećenja kadrova, odnosno najveće propusne moći po čovjeku evidentno je da mali sistem ima prednost. Problem se javlja u slučajevima kada je opterećenje stroja ispod 50%, odnosno kada prelazi maksimum. S obzirom da se isti sistem koristi za više poslovnih obrada, to problem neiskorištenih kapaciteta ne dolazi u obzir.U slučaju bitno većeg opterećenja od maksimalnog uvodi se druga smjena odnosno dodatni sistemi.

Opterećenje stroja ne ovisi samo o broju računa, već i o broju stavaka po računu kako je to pokazano na sl.3. i sl.4.



Sl.3. Vrijeme izrade računa ovisno o broju stavaka



Sl.4. Broj izradjenih računa u ovisnosti o broju stavaka

Prikazani rezultati upućuju na prednost korištenja malih poslovnih sistema u obradi poslovanja.

STRUKTURA PROGRAMSKE PODRŠKE

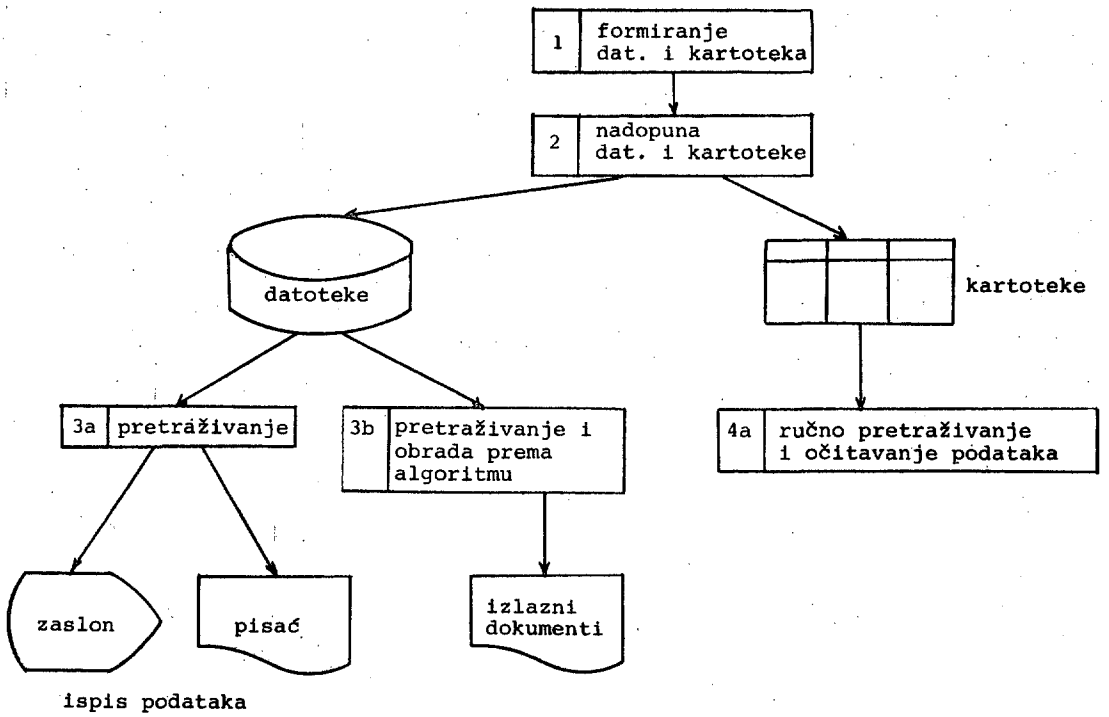
Obrada poslovanja OUR-a je zahtjev najveće grupe potrošača, te je na tom području najviše učinjeno. Strojna obrada integralnog poslovanja OUR-a uvodi se postupno i ostvaruje u zaokruženim cjelinama. Poslovne cjeline odredjuju se dvojako: prema vrsti aktivnosti, i prema organizacionoj shemi radne organizacije. Podjela prema vrsti aktivnosti je horizontalna podjela na knjigovodstvene obrade, fakturiranje, praćenje prodaje i obrade osobnih dohodaka. Knjigovodstvene obrade obuhvaćaju materjalno, troškovno, robno, pogonsko, financijsko, i saldakonti knjigovodstvo. Dodatno se mogu pratiti osnovna sredstva te posebne aktivnosti.

Vertikalna podjela je uvjetovana organizacionom strukturom radne organizacije. Za svaku organizacionu jedinku izvode se sve horizontalne obrade, da bi se pojedine istovrsne obrade vezale i vertikalno. Vertikalno povezivanje je bitno kod knjigovodstvenih obrada s financijskim pokazateljima, dok se kod drugih obrada ne koriste.

Horizontalnom podjelom odredjene su poslovne cjeline čija je strojna obrada uvjetovana izgradnjom programskih paketa. Obzirom na organizaciju, izradjeni programski paketi za male poslovne sisteme (Audit 5, Audit 6 i drugi) grupiraju se u našem slučaju u:

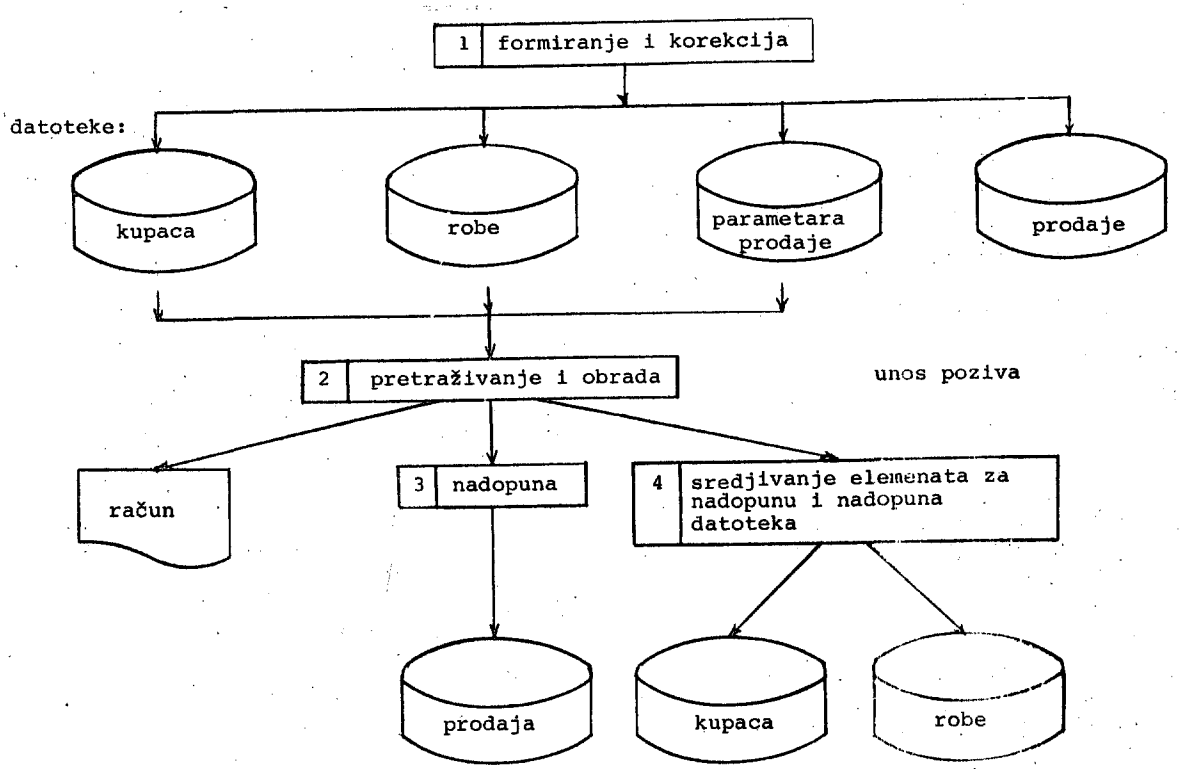
- knjigovodstvene pakete
- fakturne pakete
- pakete praćenja prodaje i
- pakete za obradu osobnih dohodaka.

Osnovna koncepcija ostvarene organizacije programske podrške zajednička je za sve knjigovodstvene pakete, i pokazana je na sl.5..



Sl.5. Organizacija knjigovodstvenih obrada

Iako se u osnovi radi o istom postupku, realizacija programske podrške se bitno razlikuje za svako knjigovodstvo. Razlike su izražene u formatu i sadržaju baze podataka te algoritmi-

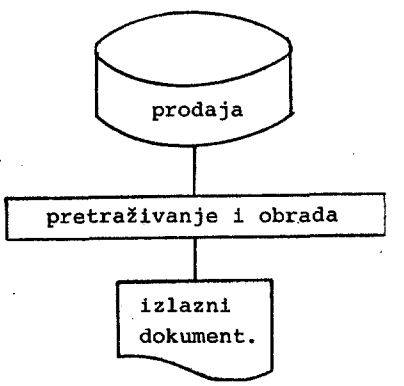


Sl.6. Organizacija obrade fakturnog poslovanja

ma za obrade i izrade izlaznih dokumenata.Paralelno s formiranjem i nadopunom datoteka formira se i nadopunjuje kartoteka, standardni element ručne obrade. Time je dobiveno istovremeno dvostruko vodjenje knjigovodstva: ručno i strojno. Ovo je još jedna od bitnih prednosti koje pruža obrada na malom poslovnom sistemu, a različito je od upotrebe velikog sistema kod kojega ovaj oblik nadopune knjigovodstvene kartice nije moguć.

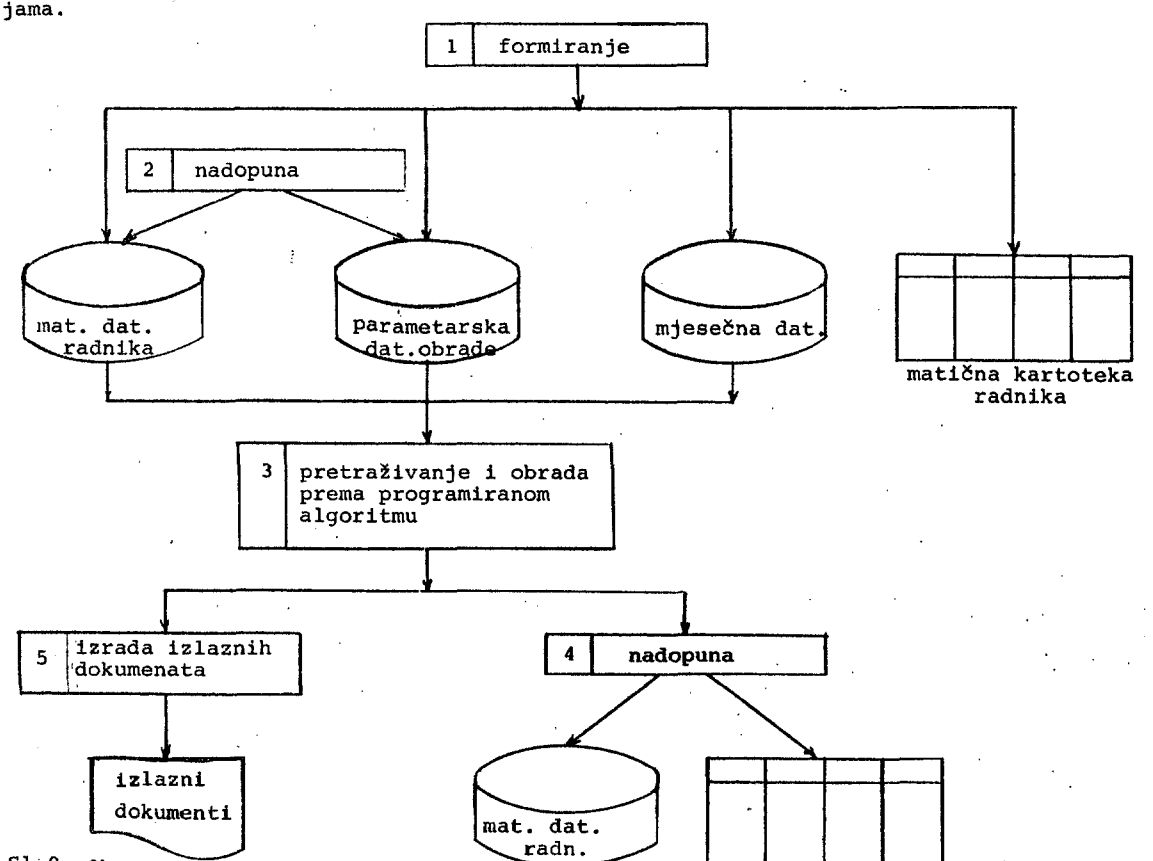
Obrada fakturnog poslovanja na malom poslovnom sistemu takodjer donosi niz prednosti u usporedbi s ručnom obradom, odnosno obradom na velikom sistemu. Osnovne prednosti su brzina,točnost, i neposredno dobijanje elemenata za knjiženje u robnom i saldakonti knjigovodstvu, te za praćenje prodaje. Ostvarena organizacija obrade prikazana je na slici 6..
Programski paket za fakturiranje realiziran je u dvije varijante. Fakturiranje na bazi otpremnice predstavlja jednostavniju varijantu i ne uvjetuje vremensku organizaciju tokova dokumenata. Fakturiranje na bazi zaključnice uključuje i izradu otpremnice.Ova varijanta zahtjeva kontinuiranu nadopunu datoteke robe i to istovremeno s procesom izrade računa. Ova varijanta zahtjeva strogu vremensku distribuciju obrade i često je neprihvatljiva u radnim organizacijama.

Paket praćenja prodaje uvjetovan je instalacijom paketa za fakturiranje. Osnovni elementi organizacije prikazani su na sl.7.. Izvor podataka je datoteka prodaje. Na temelju izlaznih dokumenata,može se zaključiti o distribuciji robe po kupcima, obradjenosti tržišta, te eventualnom smanjenju potrošnje odredjene robe na specificiranoj lokaciji, i slično. Obrada se odvija u zadanim vremenskim intervalima, obično mjesečno, i ne predstavlja posebno opterećenje za stroj.



Sl.7. Praćenje prodaje

Obrada osobnih dohodaka je vrlo složena i opsežna po konstrukciji algoritama, sadržaju baze podataka i količini izlaznih dokumenata. Osnovna shema ostvarene organizacije programske podrške prikazana je na slici 8.



Sl.8. Obrada osobnih dohodaka

Prednost korištenja malog poslovnog sistema u obradi osobnih dohodaka očituje se u mogućnosti nadopune matičnik podataka, kontroli ulaznih podataka, te mogućnosti višestruke probne obrade bez faza 4. i 5. Nedostatak primjene ovih računala je u velikoj količini ispisa koji jako opterećuje ugradjeni spori pisač. Stoga je preporučljivo da se izlazni padaci spreme na disketu, vrpcu ili kasetu te ispišu na posebnom brzom pisaču.

SUMARNI PREGLED OSTVARENIH REZULTATA

Uvodjene obrade poslovanja OUR-a primjenom malih poslovnih sistema bila je osnovna djelatnost odjela informatike radne organizacije Jugoturbina-EAB. Sada je ovu djelatnost nastavila ekonomska jedinica informatika u sastavu radne organizacije RIZ-OD u Zagrebu. U radnoj organizaciji Jugoturbina-EAB aktivnost je bila usmerjena na sisteme Olivetti Audit 5 i Audit 6.

Sistem Audit 5 je konfiguriran s procesorom Mostek 5065, glavnom memorijom 2 K okteta, dvije jedinice magnetskih kaseta kapaciteta 2 x 256 okteta, tastaturom i pisačem koji je opremljen uredjajem za automatsko uvodjenje obrazaca. Programski jezik je BAL, a operacioni sistem ne postoji. Za ovaj sistem izradjeno je deset programskih paketa. Paketi su primjenjeni u vodjenju poslovanja 49 OUR-a iz 22 radne organizacije. S obzirom na vrlo malu propusnu moć sistema prosječno se instaliraju tri do četiri paketa po sistemu, ovisno o veličini obrade. Glavnu poteškoću u radu s Auditom 5 predstavlja sekvencijalna vanjska memorija koja ograničava brzinu rada, te je zahtijevala specifična riješenja u radu s datotekama.

Sistem Audit 6 razlikuje se od predhodnog u kapacitetu glavne memorije od 4KB, tipu vanjske memorijje gdje su uvedene dvije jedinice magnetskih disketa i postojanju operacionog sistema. Za ovaj sistem izradjeno je sedam programskih paketa. Četiri su knjigovodstvena, a ostali su fakturiranje, praćenje prodaje i obrada osobnih dohodaka. S aspekta korisnika paketi imaju istu funkciju kao i na sistemu Audit 5. S obzirom na promjene u opremi i sistemskoj programskoj podršci organizacija i izvedba paketa je bitno različita. Dobiveno je poboljšanje u brzini obrade kao i neke dodatne mogučnosti rada. Programska podrška je instalirana u 16 OUR-a koji su u sastavu 5 radnih organizacija. S obzirom na povećanu propusnu moć sistema ovdje se maksimum obrade penje na više od 6 paketa po sistemu.

Daljnji rad uglavnom usmijeren na sisteme RIZ--NIXDORF prihvatio je RIZ-OD. Sistem 8820 konfiguriran je s procesorom četvrte generacije (INTEL 8080), glavnom memorijom 32K okteta, dvije jedinice magnetskih disketa, matričnim pisačem brzine 100 znakova u sekundi, alfanumeričkom, numeričkom i funkcijskom tastaturom, video terminalom kapaciteta 1920 znakova (24x80) i uredjaje za automatsko uvodjenje papirnih obrazaca. Sistem se može proširiti nizom dodatnih uredjaja. Sistemska programska podrška uključuje operacioni sistem, programske jezike PL/1 i BASIC kao i biblioteku pomoćnih programa. Za ovaj sistem u toku je izrada kompletne programske podrške. Prva iskustva kod korisnika zkazuju na veliku prednost primjene video terminala kojim je omogućen neposredan dijalog izmedju sistema i korisnika. Glavnina uputa za korištenje kao i programskih upita daju se na ekranu. Posebna prednost ekrana je u aplikacijama koje uključuju informacijski sistem. Iskustva stečena u dosadašnje radu prenjeti će se i na druge sisteme kao na primjer Olivetti BCS.

INFORMATICA 3/1979

PROCESIVANJE TEKSTOV Z MIKRORAČUNALNIKI I.

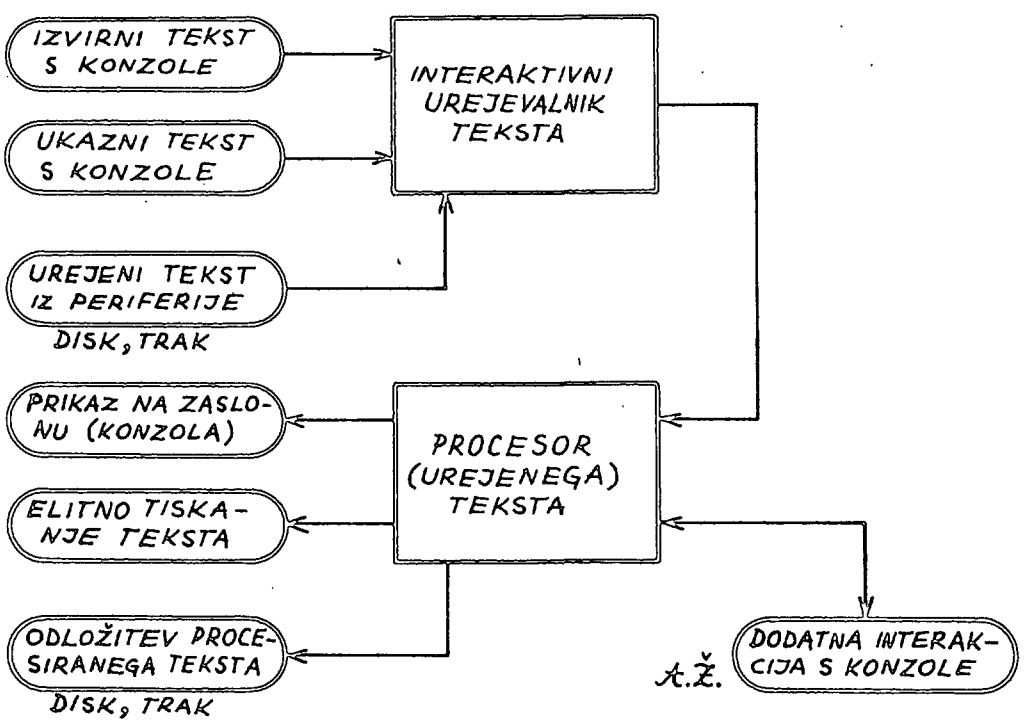
A. P. ŽELEZNIKAR

UDK: 681.3 – 181.4

INSTITUT JOŽEF STEFAN, LJUBLJANA

TA ČLANEK OPISUJE NEKATERE KONCEPTE PROCESORJEV TEKSTA IN SE UKVARJA S KONKRETNIM UKAZNIM JEZIKOM. NA PRIMERIH POKAŽE, KAKO DELUJEJO POSAMEZNE SKUPINE UKAZOV, KI SO POMEŠANE Z IZVIRNIM TEKSTOM. V ČLANKU SO NAJPREJ OPISANI REGISTRI PROCESORJA TEKSTA TER SKUPINE UKAZOV ZA OBLIKOVANJE STRANI, POLNENJE, PORAVNAVO IN CENTRIRANJE TEKSTA, ZA VERTIKALNO IZDAJANJE PROSTORA, DOLŽINO VRSTICE IN NJENO UMIKANJE, ZA MAKROJE, ODVRNITVE IN PASTI, ZA TABULIRANJE, PISANJE NASLOVOV IN ZUNANJO KOMUNIKACIJO. OPISANA JE OSNOVNA KONFIGURACIJA SISTEMA ZA PROCESIRANJE TEKSTA IN ZGRADBA PROCESORSKEGA PROGRAMA.

TEXT PROCESSING USING MICROCOMPUTERS. THIS ARTICLE DESCRIBES CONCEPTS OF TEXT PROCESSORS AND DEALS WITH A CONCRETE COMMAND LANGUAGE. BY EXAMPLES IT IS SHOWN IN WHICH WAY PARTICULAR COMMANDS AND COMMAND SEQUENCES CAN BE USED BEING MIXED WITH THE SOURCE TEXT. FIRST, THE ARTICLE DESCRIBES PROCESSOR REGISTERS AND THEIR USE, PAGE CONTROL, TEXT FILLING, ADJUSTING, AND CENTERING, VERTICAL SPACING, LINE LENGTH, AND INDENTING, MACROS, DIVERSIONS, AND LINE TRAPS, TABULATION, TITLE FORMING, AND EXTERNAL COMMUNICATION. FURTHER, A SYSTEM CONFIGURATION FOR TEXT PROCESSING IS DESCRIBED AND THE STRUCTURE OF PROCESSOR PROGRAM IS DISCUSSED.



SLIKA 1. OSNOVNA SHEMA UREJANJA IN PROCESIRANJA TEKSTA Z MIKRORAČUNALNIKOM

1. UVOD

PROCESIRANJE TEKSTOV (ANGLEŠKI SINONIMI: WORD PROCESSING, TEXT PROCESSING) SI USPEŠNO UTIRA POT V VSAKODNEVNO ADMINISTRATIVNO DEJAVNOST IN PROCESORJI TEKSTA POSTAJAJO VSEBOLJ TUDI TRŽNO ZANIMIVI IZDELKI. PROCESIRANJE TEKSTOV SE UVELJAVLJA TUDI NA MALIH RAČUNALNIŠKIH SISTEMIH, SE ZLASTI NA OSEBNIH RAČUNALNIKIH, KJER JE MOGOČE TUDI S CENENO RAČUNALNIŠKO OPREMO DOBITI REZULTATE (DOKUMENTE IN NJIHOVO OBLIKO), KI NE ZAOSTAJAJO ZA NAJDRAŽJIMI SISTEMI ZA PROCESIRANJE TEKSTOV. V TEM SESTAVKU SE BOMO OMEJILI NA KONKRETEN SISTEM, KI GA JE MOGOČE REALIZIRATI NA MIKRORAČUNALNIKIH S STANDARDNO PERIFERNO OPREMO.

DVA SISTEMA PROCESORJEV TEKSTA STA ZNAČILNA:

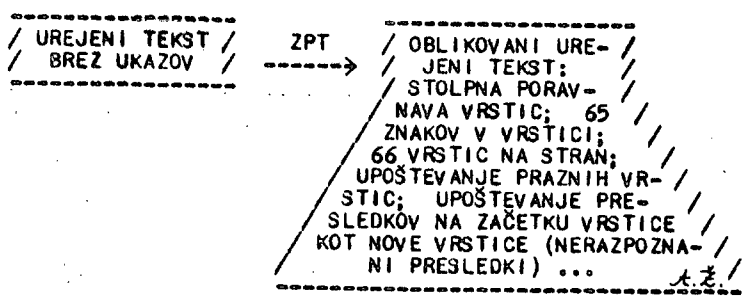
VRSTIČNI IN
ZASLONSKI.

TA ZNAČILNOST JE PRAVZAPRAV VEZANA BOLJ NA UREJEVALNIK TEKSTA, KI JE PROCESORJU TEKSTA PRIDRUŽEN. PRED IZPISOM TEKSTA V ELITNI (PROCESIRANI) OBLIKI MORAMO TEKST UREDITI, OBLIKOVATI ZA IZPIS IN TO NALOGO OPRAVIMO Z UREJEVALNIKOM TEKSTA. SLIKA 1 PRIKAZUJE GROBO SHEMO STANDARDNEGA SISTEMA ZA PROCESIRANJE TEKSTOV, KOT GA IMAMO NA MIKRORAČUNALNIKIH.

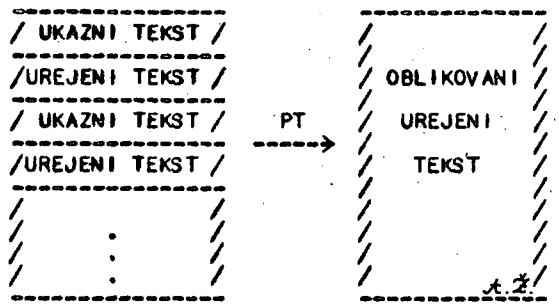
VRSTIČNI UREJEVALNIK TEKSTA VZAME ZA IZHODIŠČE UREJANJA, TO JE PISANJA, POPRAVLJANJA, DODAJANJA, SPREMINJANJA IN ODVZEMANJA TEKSTA, VRSTICO, KI JO OŠTEVILČI, TAKO DA LAHKO VRSTICO NAVAJA. VRSTIČNI UREJEVALNIK OMOGOČA, DA VODIMO EVIDENCO, V KATERIH VRSTICAH SE NAHAJAJO DOLOČENI TEKSTOVNI SEGMENTI.

ZASLONSKI UREJEVALNIK MORA NATANKO POSNEMATI FUNKCIJE ZASLONSKEGA TERMINALA; TO KAR SE VIDI NA ZASLONU, SE TUDI DEJANSKO OPRAVLJA V PODATKOVNI ZBIRKI TEKSTA, KI GA UREJAMO. TA METODA IMA DOLOČENE PREDNOSTI ZA UPORABNIKA, KI MU JE VIZUALNA SLIKA NA ZASLONU DOKAZ, DA JE OPRAVIL TO, KAR JE ŽELEL.

UREJEVALNIKI TEKSTA SO DANES PRAKTIČNO PRISOTNI V VSAKI MINI- IN MIKRORAČUNALNIŠKI KONFIGURACIJI IN JIH V TEM ČLANKU NE BOMO OBRAVNAVALI. UKVARJALI PA SE BOMO Z UKAZNIM JEZIKOM IN PRIMERI OBDELAVE BESEDIL TER Z METODAMI ZA RAZLIČNE (ELITNE) IZPISE, KI JIH TAKŠEN PROCESOR TEKSTA OMOGOČA.



SLIKA 3. ZAČETNI PROCESOR TEKSTA (ZPT) JE REALIZIRAN TAKRAT, KO MIMAMO V TEKSTU ZA IZPIS NOBENEGA UKAZA



SLIKA 2. TRANSFORMACIJA MEŠANEGA UKAZNEGA IN UREJENEGA TEKSTA V OBLIKOVANI UREJENI TEKST S POMOČJO PROCESORJA TEKSTA 'PT'

2. KAJ JE PROCESIRANJE TEKSTA

PROCESIRANJE TEKSTA ALI OBDELAVA BESEDILA Z MIKRORAČUNALNIŠKIM SISTEMOM SE OPRAVLJA S POSEBNIM, DOVOLJ ZAPLETENIM PROGRAMOM. V MALIH SISTEMIH JE TA PROGRAM REZIDENTEN IN SE NAHAJA V POMNILNIKU TIPA ROM; V NORMALNIH SISTEMIH JE TA PROGRAM SHRANJEN NA DISKU, OD KODER GA LAHKO POKLIČEMO V IZVAJANJE, KO GA VSTAVIMO V HITRI (GLAVNI) POMNILNIK.

PROCESIRANJE TEKSTOV OMOGOČA ELITNO OBLIKOVANJE ČASOPISOV, KNJIG, KATALOGOV, PROSPEKTOV, PRIROČNIKOV, SEZNAMOV, PISEM, POKLICNIH DOKUMENTOV ITD. TAKO OBSEŽNIH NALOG NE BI BILO MOGOČE OPRAVITI ROČNO ZARADI PREVELIKIH ČASOVNIH IN DELOVNIH ZAHTEV.

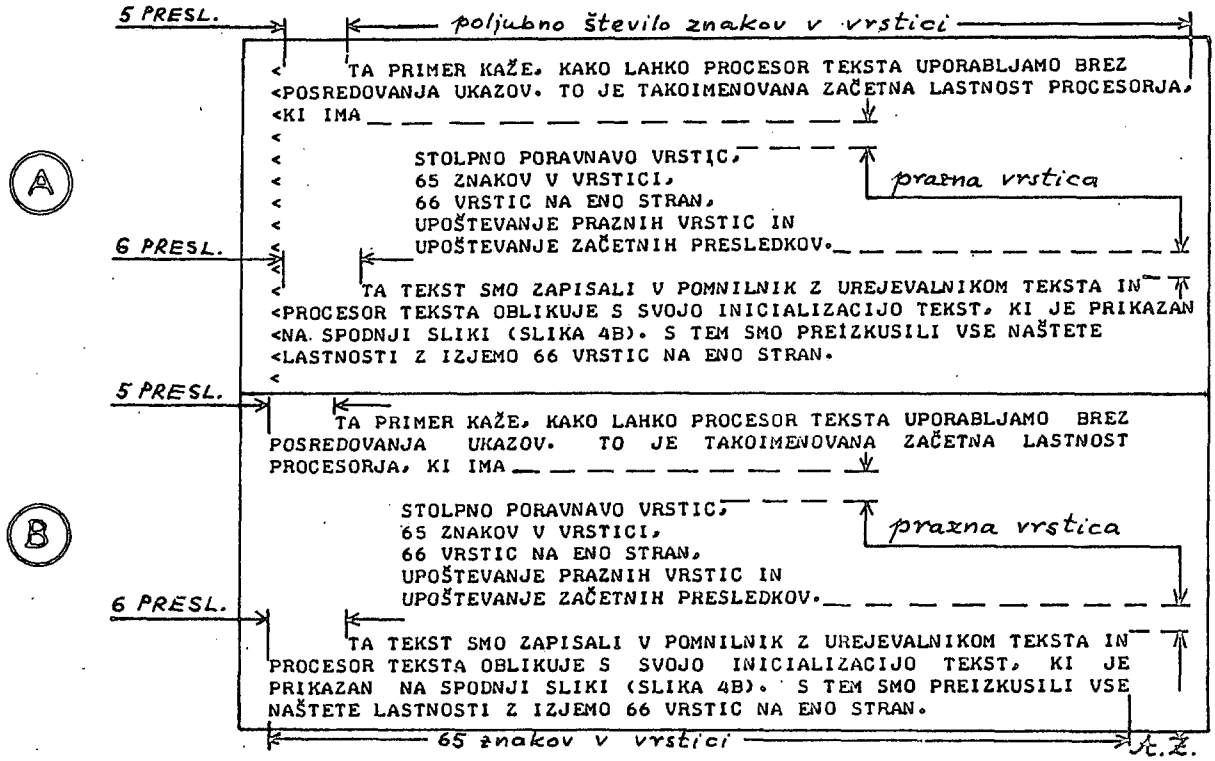
OSNOVNA ZAMISEL PROCESORJA TEKSTA JE OBLIKOVANJE STRANI. TEKSTOVNA STRAN NAJ IMA NPR.

NASTAVLJIVI LEVI ROB,
NASTAVLJIVO DOLŽINO VRSTICE,
ŠTEVILKO STRANI V SREDINI ZGORAJ,
POSEBEN NASLOV NA VSAKI STRANI,
OPOMBE NA SPODNJEM ROBU STRANI,
ENOTNO OBLIKO ODSTAVKOV ITN.

VSE TE FUNKCIJE NAJ BODO ZAGOTOVLJENE AVTOMATIČNO IN SEVEDA SPREMENLJIVO S POMOČJO USTREZNIH UKAZOV IN NJIHOVIH ZAPOREDIJ. TUDI PORAVNAVA VRSTIC NAJ BO PROŽNA, KO IMAMO NPR.

LEVO PORAVNAVO VRSTIC,
DESNO PORAVNAVO VRSTIC,
CENTRIRANO PORAVNAVO VRSTIC IN
STOLPNO PORAVNAVO VRSTIC.

PROCESOR TEKSTA IMA SVOJO, DOLOČENO ZALOGO UKAZOV. TA ZALOGA SE SPREMINJA OD PROCESORJA DO PROCESORJA, VENDAR JE UČINEK DOLOČENIH



SLIKA 4. PREIZKUS PROCESORJA TEKSTA PRI AVTOMATIČNI INICIALIZACIJI BREZ POSREDOVANJA UKAZOV: (A) UREJENI TEKST IN (B) IZPISANI TEKST (OBDELANO BESEDILO S PROCESORJEM)

UKAZOV PRI PROCESORJIH TEKSTA ENAK. PROCESOR TEKSTA, KI GA BOMO OPISALI, UPORABLJA

MEŠANO TEKSTNO IN UKAZNO OBLIKO. KAJ TO POMENI? TEKST, KI GA ŽELIMO OBLIKOVATI IN UKAZI, S KATERIMI GA OBLIKUJEMO, SO POMEŠANI. TO POMENI, DA IMAMO RAZMERE, KOT JIH KAŽE SLIKA 2. OBLIKOVANI (IZPISANI) TEKST SEVEDA NE VSEBUJE UKAZOV: TO JE ČISTI IN USTREZNO OBLIKOVANI TEKST. V TEM PRIMERU MORA PROCESOR ENOLIČNO RAZPOZNATI UKAZNI TEKST OD OSTALEGA TEKSTA.

VSAKA UKAZNA VRSTICA ZAČENJA V PRVEM STOLPCU S POSEBNIM ZNAKOM, KI JE

'.' ALI ':'

VENDAR IMA PROCESOR TEKSTA SVOJE ZAČETNE LASTNOSTI TUDI, ČE MU V POSTOPKU OBDELAVE KAKŠNEGA BESEDILA NISMO POSREDOVALI NOBENEGA UKAZA. V TEM PRIMERU LAHKO PIŠEMO TEKSTE V OBLIKI, KOT KAŽE SLIKA 3, OZIROMA KOT JE S PRIMEROM POKAZANO NA SLIKI 4.

SLIKA 4 KAŽE, DA JE MOGOČE PROCESOR TEKSTA TAKOJ UPORABITI, ČE SE PISEC STRINJA Z NASLEDNJIMI PRAVILI:

(1) DA BO IMEL PRI IZPISU STOLPEC S 65 TISKANIMI ZNAKI V VRSTICI IN 66 VRSTIC NA ENI STRANI;

(2) DA SE BO VSAKA PRAZNA VRSTICA (NPR. ZAPOREDJE SPACE-CR-LF POJAVILO V IZPISU KOT POSLEDICA PRAZNE (VMESNE) VRSTICE;

(3) DA BO PRESLEDEK (SPACE) NA ZAČETKU VRSTICE POVZROČIL NOVO VRSTICO (ZAČETEK ODSTAVKA), KJER BO V CELOTI UPOŠTEVANO ŠTEVILO VSEH ZAČETNIH PRESLEDKOV KOT UMAKNITEV ZAČETKA ODSTAVKA.

UPORABNIK ZAČETNEGA PROCESORJA MORA TAKO OBVLADATI LE TOČKI (2) IN (3) PRI UREJANJU TEKSTOV!

SEVEDA PA RAZPOLAGA PROCESOR TEKSTA Z BOGATO ZALOGO UKAZOV, KI OMOGOČA VSAKO OBLIKOVANJE TEKSTOV IN VISOKO STOPNJO AVTOMATIZACIJE, SAJ LAHKO IZ UKAZOV PROCESORJA SESTAVIMO MAKROJE, TO JE PROGRAME Z MOŽNOSTJO NJIHOVEGA KLICANJA, KI OPRAVLJAJO POLJUBNO ZAPLETENE OBLIKOVALNE NALOGE.

3. REGISTRI PROCESORJA TEKSTA

PROCESOR TEKSTA IMA SVOJE REGISTRE, KI JIH LAHKO NASTAVLJAMO IN ODČITAVAMO TER JIH PROCESOR UPORABLJA PRI SVOJEM DELOVANJU. TEH REGISTROV JE 27 IN SO POIMENOVANI Z VSEMI ČRKAMI ANGLEŠKE ABECEDE IN Z ZNAKOM PROCENTA. VSI REGISTRI SO HEKSADECIMALNO DVOMESTNI IN NJIHOVE VREDNOSTI SO V INTERVALU (0, 255). SPISEK REGISTROV PA JE TALE:

%	ŠTEVILKA STRANI (JE 1 V ZAČETKU)
A-B	UPORABNIŠKA REGISTRA
D	DAN V MESECU
E-F	UPORABNIŠKA REGISTRA
G	ŠTEVILO ZNAKOV PRI .GI UKAZU
H	UPORABNIŠKI REGISTER
I	TRENUTNA UMAKNITEV TEKSTA (PRESLEDKI)
J-K	UPORABNIŠKA REGISTRA
L	TRENUTNA DOLŽINA VRSTICE (65 V ZAČETKU)
M	MESEC V LETU
N	ŠTEVNIK VRSTIC NA STRANI

O	TRENUTNI LEVI ROB
P	TRENUTNA DOLŽINA STRANI (66 V ZAČETKU)
Q-U	UPORABNIŠKI REGISTRI
V	ŠTEVNIK ZADNJE DIVERZNE VRSTICE
W-X	UPORABNIŠKA REGISTRA
Y	LETO (SAMO DVE ŠTEVILKI)
Z	UPORABNIŠKI REGISTER

IZPIS VREDNOSTI REGISTROV IMA LAHKO TUDI OBLIKO RIMSKIH ŠTEVIL (Z MALIMI ALI VELIKIMI ČRKAMI); TA PRIMER BOMO OPISALI KASNEJE.

REGISTER X IZ MNOŽICE REGISTROV A, B, ... , Y, Z LAHKO NAVAJAMO NA DVA NAČINA, IN SICER KOT

#X IN
#+X,

KJER JE '#' OZNAČEVALNIK REGISTRA IN X IME REGISTRA. REGISTER % NAVEDEMO PREPROSTO KAR Z % (BREZ PREDZNAKA #). ZNAK + V IZRAZU #+X DOLOČA, DA BO REGISTER X AVTOMATIČNO POVEČAN (INKREMENTIRAN ALI DEKREMENTIRAN) VSAKOKRAT PRED SVOJO UPORABO ZA DOLOČENO VREDNOST (GLEJ UKAZ .AU KASNEJE). IZPIS VREDNOSTI REGISTRA X IMA LAHKO OBLIKO Z ARABSKIMI ŠTEVILKAMI ALI PA TUDI Z VELIKIMI ALI MALIMI RIMSKIMI ŠTEVILKAMI (GLEJ UKAZE .AR, .CR IN ŠE .SR KASNEJE). KONVERZIJA V RIMSKE ŠTEVILKE SE IZVRŠI OB IZPISU (PRI OBDELAVI BESEDILA).

Z REGISTRI LAHKO OBLIKUJEMO TUDI IZRAZE, KI SO DELI UKAZOV; V IZRAZIH SE LAHKO POJAVLJAJO TUDI CELA ŠTEVILA. IZRAZI SE IZRAČUNAVAJO OD LEVE PROTI DESNI, TODA DOPUSTNA OPERATORJA STA LE '+' IN '-'. NPR. V IZRAZU

.IF #N-#P-#W .CH FO #N+1

KAR POMENI: ČE JE N-P-W > 0, SE BO IZVRŠIL UKAZ .CH FO #N+1, KI POMENI, SPREMENI LOKACIJO PASTI, DOLOČENO Z MAKROJEM 'FO' PRI N+1.

OGLEJMO SI ŠE REGISTRSKE UKAZE:

.NR X +N — POVZROČI PRIREDITEV POZITIVNE, NEGATIVNE ALI ABSOLUTNE VREDNOSTI N REGISTRU X. CE JE -N, DOBIMO 256-N. TA UKAZ NAJ BI SE PRAVILOMA UPORABLJAL LE ZA NASTAVLJANJE UPORABNIŠKIH REGISTROV A, B, E, F, H, J, K, Q, R, S, T, U, W, X, Z.

.AU +N — NASTAVI AVTOMATIČNO PRIRASTEK NA VREDNOST N. VSAKOKRAT, KO SE NAVEDE REGISTER X Z NAVEDBO OBLIKE

#+X,

SE PRED UPORABO REGISTRA X POVEČA NJEGOVA VREDNOST NA (X)-N. ČE JE -N, DOBIMO DEKREMENTIRANJE, TOREJ (X)-N.

.AR — ARABSKE ŠTEVILKE (V ZAČETKU SO PRAV TE NASTAVLJENE).

.CR — VELIKE RIMSKE ŠTEVILKE.

.SR — MALE RIMSKE ŠTEVILKE. VSEBINE REGISTROV X (%, A, B, ... , Y, Z) SE LAHKO PRETVORIJO V ARABSKO, MALO- ALI VELIKOČRKOVNO RIMSKO OBLIKO. NA ZAČETKU JE IZPIS V ARABSKI OBLIKI IN PODOBNO LAHKO SPREMINJAMO TUDI OBLIKO IZPISA REGISTRA %.

SLIKA 5 KAŽE TIPIČEN PRIMER NAVAJANJA IN INKREMENTIRANJA REGISTRA A. IZRAZI SE IZRAČUNAVAJO LE V UKAZIH, V TEKSTU SE PA SAMO INTERPRETIRAJO, KOT JE RAZVIDNO IZ SLIKE 5.

REGISTER %, KI HRANI ŠTEVILKO TEKOČE STRANI, SE LAHKO TUDI NASTAVI Z UKAZOM:

.PN +N — NASTAVI ŠTEVILKO STRANI NA +N (TO JE +N, -N ALI N, KJER DOBIMO ZA -N VREDNOST % = 256 - N). REGISTER % MORAMO NASTAVITI PRED PRVIM LOMOM STRANI, NJEGOVA ZAČETNA VREDNOST JE 1, MAKSIMALNA PA 255.

4. UKAZNE VRSTICE V UREJENEM TEKSTU

ŽE VECKRAT SMO POKAZALI, DA SO UKAZNE VRSTICE V UREJENEM TEKSTU POMEŠANE Z VRSTICAMI T.I. ČISTEGA ALI IZVIRNEGA TEKSTA. VSAKA UKAZNA VRSTICA MORA ZACENJATI Z ZNAKOM

'.' ALI ':'

V PRVEM STOLPCU VRSTICE, KI JI MORA TAKOJ (BREZ PRESLEDKA) SLEDITI UKAZ.

VSAK UKAZ JE SESTAVLJEN IZ DVOČRKOVNEGA OPERATORJA, KI MU LAHKO SLEDI ŠE OPERANDNO POLJE. TUDI POZIVI MAKROJEV SO VSELEJ DVOČRKOVNI. UKAZ SE ENOSTAVNO PRESKOČI, ČE NJEGOV OPERATORSKI DEL NI VELJAVEN UKAZ PROCESORJA ALI IME DEFINIRANEGA MAKROJA.

MED ZNAKOMA '.' IN ':' PA JE TALE RAZLIKA: ZNAK '.' POVZROČI HKRATI IN PRAVILOMA ZAČETEK NOVE VRSTICE PRI IZPISU, TOREJ LOM VRSTICE (TODA NE PRI VSEH UKAZIH), DOČIM ZNAK ':' VRSTICE NE LOMI, TEMVEČ SE IZPIS NADALJUJE DO KONCA TEKOČE IZPISANE VRSTICE. PROTI LOMLJENJU VRSTIC SE TAKO ZAVARUJEMO, ČE UPORABIMO ZNAK ':' NAMESTO ZNAKA '.'. KER PA TUDI Z ZNAKOM '.' NE DOBIMO VSELEJ ZLOMITVE VRSTICE, MORAMO POGLEDATI V UKAZNI SEZNAM IN UGOTOVITI ALI UKAZ Z ZNAKOM '.' POVZROČI LOM ALI NE. NPR. UKAZI

ORIGINAL

```
.NR A 88
.AU -3
A = #+A, #+A, #+A,  IZRAZ = #A+#A+#A+A+A,
IZRAZ 1 = #A+#A+#+A;
```

```
A = 85, 82, 79,  IZRAZ = 79+79+79+A+A, IZRAZ 1 = 79+79+76;
```

PREVOD

SLIKA 5. PRIMER NAVAJANJA REGISTRA A IN IZRAZOV. V TEKSTU SE #+A RES INKREMENTIRA, IZRAZ IN IZRAZ 1 PA SE NE IZRAČUNATA IN SE LE INTERPRETIRATA, KOT JE RAZVIDNO IZ SLIKE POD ČRTO.

.NR, .AU, .AR, .CR IN .SR,

KI SMO JIH ZE OPISALI, NE POVZROCAJO LOMA.

VSAKA UKAZNA VRSTICA LAHKO VSEBUJE LE EN UKAZ.

NUMERIČNI ARGUMENT (OPERAND) ±N SE POJAVLJA V UKAZIH, KOT SMO OPISALI ŽE V PREJŠNJEM POGLAVJU. ZA ARGUMENT, KI JE OZNAČEN Z ±N, VELJA

$$\pm N = \begin{cases} +N, & \text{POZITIVNA VREDNOST} \\ -N, & \text{NEGATIVNA VREDNOST} \\ N, & \text{ABSOLUTNA VREDNOST} \end{cases}$$

TOREJ SO VELJAVNI ARGUMENTI ZA ±N NPR. +10, -19 IN 26. V UKAZU ZA INKREMENTIRANJE BO -N POVZROCIL DEKREMENTIRANJE ZA N PO MODULU 256. PRI NASTAVITVI .NR A -1 BOMO DOBILI VREDNOST 255 ZA A ITN. SEVEDA PA NI PRIPOROČLJIVO, DA PREIZKUSAMO RAZNE NESMISELNE KOMBINACIJE, KER PROCESOR TEKSTA LAHKO ODGOVORI S STRANSKIMI UČINKI, KO SE IZVAJANJE PROCESORJA NADALJUJE V NEPREDVIDENIH PROGRAMSKIH SEGMENTIH IN LAHKO POVZROČI POŠKODBO PROGRAMA IN TEKSTA.

5. OBLIKOVANJE STRANI

OBLIKOVANJE STRANI OBSEGA FIZIČNO STRAN, IN SICER NJENO DOLŽINO, ŠTEVILO STRANI, ROBOVE STRANI, OSTEVILČENJE STRANI ITN. VRH IN SPODNJA STRAN STRANI SE LAHKO OBLIKUJETA Z MAKROJI OZIROMA UKAZNIMI ZAPOREDJI IN TO BO OPISANO KASNEJE. OGLEJMO SI ZNAČILNE UKAZE ZA OBLIKOVANJE STRANI:

.PL ±N DOLŽINA STRANI JE N VRSTIC. ZAČETNA VREDNOST 66 VRSTIC JE ŽE V REGISTRU P IN S TEM UKAZOM NASTAVIMO TAKO TUDI VREDNOST REGISTRA P (DRUG UKAZ BI BIL .NR P ±N). LOM VRSTICE PRI TEM UKAZU NE NASTOPI IN N IMA MAKSIMALNO VREDNOST 255.

.PG ±N IZDAJ (SKOČI NA) NASLEDNJO STRAN. PRI DANEM N SE NASTAVI NOVA ŠTEVILKA STRANI (%) = N. ČE NIMAMO ARGUMENTA N, IMAMO AVTOMATIČNO INKREMENTIRANJE (+1). TA UKAZ POVZROČI LOM VRSTICE IN NAJVEČJI N JE 255. PRI TEM SE IZDAJO SEVEDA ŠE VSE PRAZNE VRSTICE, KI MANJKAJO DO KONCA STRANI.

.PN ±N NASTAVI ŠTEVILKO STRANI NA ±N. ČE SE TA UKAZ POJAVI PRED PRVIM LOMOM VRSTICE ALI PRED TEKSTOM, BO ŠTEVILKA STRANI ENAKA 1. TA UKAZ NE POVZROČI LOMA IN N IMA MAKSIMALNO VREDNOST 255.

.LM ±N LEVI ROB ZNAŠA N ZNAKOV. VSE VRSTICE IZPISANEGA TEKSTA BODO POMAKNJENE ZA N PRESLEDKOV V DESNO. V ZAČETKU JE N ENAKO Ø (NIČ) IN VREDNOST REGISTRA '0' JE NIČ. LOM VRSTICE NE NASTOPI IN N NAJ BI NE BIL VEČJI OD 100.

.NL N POTREBUJEMO N PRAZNIH VRSTIC NA STRANI (NPR. ZA SLIKO ALI DIAGRAM). ČE JE RAZDALJA DO KONCA STRANI PREMAJHNA (MANJŠA KOT N VRSTIC), SE BO TA PROSTOR REZERVIRAL PRI NASLEDNJI PASTI, KO BO IZDANIH N PRAZNIH VRSTIC.

NASTAVITEV DOLŽINE STRANI Z UKAZOM .PL ±N JE SMISELNA ZLASTI V POVEZAVI Z NASTAVLJANJEM PASTI, KI JIH LAHKO POSTAVLJAMO TUDI OD KONCA STRANI PROTI SREDINI IN DOSEŽEMO TAKO ŽELENE UČINKE, NPR. Z UPORABO UKAZA .PG V POSEBNIH MAKROJIH. UPORABA NAŠTETIH UKAZOV V TEM POGLAVJU BO PRIKAZANA V POVEZAVI Z MAKROJI IN PASTMI.

6. TEKSTNO NAPOLNJEVANJE, PORAVNAVA IN CENTRIRANJE VRSTIC

UKAZI V TEM POGLAVJU SE BODO NANAŠALI NA VRSTICO, KO SE NAVAJATA DVA BISTVENA PARAMETRA, IN SICER NAPOLNITEV IN PORAVNAVA. POD NAPOLNITVIJO RAZUMEMO NAPOLNITEV VRSTICE S TEKSTOM, TAKO DA JE DOSEŽENA DOLOČENA (VERTIKALNA) PORAVNAVA TEKSTA OZIROMA VRSTIC, NPR. LEVA, DESNA, OBOJESTRANSKA (STOLPNA) ALI CENTRIRANA, ČE OPAZUJEMO OZIROMA GENERIRAMO TEKST KOT NEKE VRSTE STOLPEC, KI JE SESTAVLJEN IZ VRSTIC.

NAPOLNITVENI NAČIN (STANJE) NAJ BI NAPOLNIL VRSTICO S ČIM VEČJIM ŠTEVILOM BESED TAKO, DA NI PRESEŽENA DOLŽINA VRSTICE. ODVEČNE BESEDE, KI SE PRI TEM LAHKO POJAVIJO, SE SHRANIJO ZA IZPIS V NASLEDNJI VRSTICI.

BESEDA JE DOLOČENA KOT POLJUBEN NIZ ZNAKOV, KI SO LOČENI OD DRUGIH BESED Z ENIM ALI VEČ PRESLEDKI. ČE STA NPR. DVE BESEDI LOČENI S PRESLEDKOM, TODA JU NE ŽELIMO POSTAVITI V DVE LOČENI VRSTICI (TOREJ JU ŽELIMO IMETI V VRSTICI SKUPAJ), POTEM BO TO ŽELJO LAHKO UPOŠTEVALA PORAVNALNA SUBRUTINA, ČE BOMO UPORABILI NAMESTO PRESLEDKA RAZPOREK '\'.

POZNAMO LEVO IN DESNO PORAVNAVO VRSTIC V STOLPCU, PA TUDI OBOJESTRANSKO ALI STOLPNO PORAVNAVO. KADAR UPORABLJENE BESEDE V VRSTICI NE ZAPOLNIJO VRSTICE NATANKO OD NJENEGA LEVEGA DO DESNEGA ROBA, SE VSTAVLJAJO MED BESEDE DODATNI PRESLEDKI, TAKO DA SE VSELEJ DOSEŽE NATANČNA DOLŽINA VRSTICE. VSTAVLJANJE PRESLEDKOV SE OPRAVLJA V SMERI OD OBEH ROBOV VRSTICE, TAKO DA SE NE BI POJAVILA PREDOLGA ZAPOREDJA PRESLEDKOV (BELA POLJA). ČIM DALJŠA JE VRSTICA IN ČIM KRAJŠE SO BESEDE V NJEJ, TEMBOLJ ENAKOMERNO BODO LAHKO RAZPOREJENI DODATNI PRESLEDKI V VRSTICI. ANGLEŠČINA JE V TEM OZIRU PRAV GOTOVO PRIMERNEJŠA OD SLOVENŠČINE. METODA, KI JO TUKAJ PREDSTAVLJAMO, NAMREČ ŠE NE UPORABLJA DELITVE BESED. SEVEDA PA IMAJO NEKATERI PROCESORJI TEKSTA TUDI DELITVENE SUBRUTINE, KAR SEVEDA IZBOLJŠA OBLIKO IZPISANEGA TEKSTA, ŠE POSEBEJ, KO IMAMO KRATKE VRSTICE.

V STANJU POLNENJA MORA BITI PROCESOR, KADAR ŽELIMO IMETI STOLPNO PORAVNAVO. ZAČETNI PROCESOR TEKSTA (SLIKA 3) JE ŽE V TEM STANJU, SAJ VSEBUJE STOLPNO PORAVNAVO. ČE PROCESOR NI V STANJU POLNENJA, SE ZNAKI IZPISUJEJO NATANKO TAKO, KOT SO BILI VSTAVLJENI V VHODNO TEKSTNO ZBIRKO.

OGLEJMO SI SEDAJ POMEN NASLEDNJIH UKAZOV:

.BR ZLOMI VRSTICO, KI SE POLNI V VMESNIKU. TA VRSTICA SE IZPIŠE SKLADNO Z DOLOČENO PORAVNAVO, TODA BREZ NADALJNEGA POLNENJA ALI VSTAVLJANJA PRESLEDKOV. LOM VRSTICE POVZROČIJO TUDI PRESLEDKI V ZAČETKU VRSTICE IN PRAZNE VRSTICE (KOT JE BILO ŽE OPISANO).

.FI S TEM UKAZOM UVEDEMO STANJE POLNENJA IN NADALJNE VRSTICE SE BODO POLNILE. TA UKAZ POVZROČI LOM VRSTICE.

.NF S TEM UKAZOM PREKINEMO STANJE POLNENJA IN NASLEDNJE VRSTICE NE BODO NITI POLNENE NITI PORAVNANE TER SE BODO IZPISOVALE NATANKO TAKO, KOT SO BILE VPISANE, NEGLEDE NA TRENUTNO POSTAVLJENO DOLŽINO VRSTICE. TA UKAZ POVZROČI LOM VRSTICE.

.JU X IMAMO STANJE VRSTIČNE PORAVNAVE. ČE PRI TEM NIMAMO ŠE STANJA POLNENJA, NE BO PORAVNAVE VRSTIC, DOKLER NE UVEDEMO

STANJA POLNENJA. PRI TEM VELJA ŠE:
X = N ALI R ALI C

X = N POMENI: IMAMO NORMALNO OZIROMA STOLPNO PORAVNAVO.

X = R POMENI: IMAMO DESNO PORAVNAVO (PORAVNANI SO DESNI KONCI VRSTIC).

X = C POMENI: IMAMO PORAVNAVO VRSTIČNIH SREDIŠČ (STOLPEC JE SIMETRIČNO ZVIJUGAN). ČE X NE IZPIŠEMO (PUSTIMO PRAZNO), SE Z UKAZOM .JU UVEDE STANJE PORAVNAVE, KI JE BILO AKTIVNO PRED NJEGOVO UKINITVIJO. TA UKAZ NE POVZROČI LOMA VRSTICE.

.NJ UKINI PORAVNAVO. ČE IMAMO PRI TEM STANJE POLNITVE, BO IZPISANA VRSTICA LEVO PORAVNANA, DESNI ROB STOLPCA BO ZVIJUGAN. LOMA VRSTICE TU NI IN PORAVNALNI TIP (REGISTER) OSTANE OHRANJEN.

.CE +N CENTRIRAJ NASLEDNJIH N VHODNIH VRSTIC. LOM NASTOPI PRED UKAZOM IN POTEM ŠE AVTOMATIČNO PO VSAKI ZAPISANI VRSTICI. ČE JE VSTAVLJENA VRSTICA DALJŠA OD DANE DOLŽINE VRSTICE, BO IZPISANA VRSTICA LEVO PORAVNANA. MAKSIMALNO ŠTEVILO CENTRIRANIH VRSTIC BO 255.

7. VSTAVLJANJE PRAZNIH VRSTIC (PROSTORA)

VSTAVLJANJE PRAZNIH VRSTIC PREDSTAVLJA V BISTVU VSTAVLJANJE PRAZNEGA PROSTORA V VERTIKALNI SMERI STRANI, KI JO UREJAMO. SKOK V NOVO VRSTICO, V KATERI SE NADALJUJE PISANJE TEKSTA, POMENI VSTAVITEV PRAZNEGA PROSTORA Z DOLŽINO NIČ. PRAVI PRAZEN PROSTOR JE MOGOČE VSTAVITI Z UKAZOM .MS N. PRI PARAMETRU N V TEM UKAZU SE VSTAVI N-1 PRAZNIH VRSTIC PO VSAKI NEPRAZNI VRSTICI. S POMOČJO PASTI, KI JIH BOMO OBRAVNAVALI KASNEJE, LAHKO VPLIVAMO NA VSTAVITEV PRAZNEGA PROSTORA. PRAZNI PROSTOR LAHKO TUDI USTREZNO REZERVIRAMO SE Z DRUGIMI UKAZI.

OGLEJMO SI TIPIČNE UKAZE ZA VSTAVLJANJE PROSTORA IN Z NJIMI POVEZANA STANJA. IMAMO:

.MS N NASTAVI VEČKRATNI PRESLEDEK MED VRSTICAMI NA N. TU SE N NANAŠA TUDI NA NEPRAZNO VRSTICO, TAKO DA JE PRAZNIH VRSTIC ŠE N-1. TA UKAZ NE POVZROČI LOMA VRSTICE IN ČE ARGUMENT N IZPUSTIMO, IMAMO N = 2 (MED VSAKO IZPISANO VRSTICO JE ENA PRAZNA VRSTICA). NAJVEČJA VREDNOST ZA N JE 255.

.SS S TEM UKAZOM SE NASTAVI STANJE BREZ PRAZNIH VRSTIC. IZPISANE VRSTICE SI SLEDIJO STRNJENO EN ZA DRUGO IN SAM UKAZ NE POVZROČI LOMA VRSTICE.

.SP N S TEM UKAZOM VSTAVIMO ENKRATNI PRAZNI PROSTOR, KI OBSEGA N PRAZNIH VRSTIC. ŠTEVILO IZDANIH PRAZNIH VRSTIC JE LAHKO OMEJENO Z BLIŽNJO PASTJO ALI Z DNOM STRANI. ČE JE PROCESOR V REŽIMU, KO SE NE IZDAJA PROSTOR (UKAZ .NS), SE PRAZNE VRSTICE NE IZPISUJEJO. ČE N NE NAVEDEMO V UKAZU .SP, POMENI TO N = 1. UKAZ .SP POVZROČI LOM VRSTICE.

.SV N REZERVIRAJ (REŠI) PROSTOR Z N PRAZNIMI VRSTICAMI. ČE JE RAZDALJA DO NASLEDNJE PASTI ALI DO DNA STRANI VEČJA KOT N, SE TAKOJ IZPIŠE N PRAZNIH VRSTIC, SICER PA SE TE VRSTICE NE IZPIŠEJO TAKOJ, VENDAR SE N SHRANI ZA KASNEJŠI IZPIS PRAZNIH VRSTIC (GLEJ UKAZ .OS). TRENUTNI UKAZ .SV N IZNIČI VSAK PREJE POSTAVLJENI N. UKAZ .NS NE VPLIVA NA TA UKAZ. UKAZ .SV NE POVZROČI LOMA VRSTICE IN ČE N NE NAVEDEMO, POMENI, DA JE N = 1.

.OS IZDAJ REZERVIRANI PRAZNI PROSTOR (N VRSTIC). TA UKAZ SE UPORABLJA ZA IZPIS PREJE REZERVIRANEGA PROSTORA Z ZAHTEVO, KI SMO JO POSTAVILI Z UKAZOM .SV N. ZAPOMNJENI N SE PRI TEM ZBRIŠE (S POZIVOM UKAZA .OS) IN UKAZ .NS NIMA VPLIVA. LOM VRSTCE NE NASTOPI.

.NS UVEDE SE STANJE (REŽIM), KO SE NE IZDAJAJO PRAZNE VRSTICE (Z GORNJIMI IZJEMAMI). TA UKAZ PREPREČUJE ZAHTEVE, KI SO BILE POSTAVLJENE Z UKAZOMA .SP IN .PG, ČE UKAZ .PG NI IMEL NAVEDENE NASLEDNJE STRANI (BREZ N). TO STANJE SE AVTOMATIČNO UKINE PO IZPISU VRSTICE TEKSTA. LOM VRSTICE TU NE NASTOPI.

.RS UVEDE SE STANJE IZPISA PRAZNIH VRSTIC. ČE IMAMO REŽIM UKAZA .NS, GA S TEM UKAZOM UKINEMO. LOM VRSTICE NE NASTOPI.

8. DOLŽINA IN UMAKNITEV VRSTICE

Z DOLOČENIMI UKAZI JE MOGOČE NASTAVITI DOLŽINO VRSTICE IN NEKATERE OBLIKE VRSTIČNE UMAKNITVE. DOLŽINA VRSTICE UPOŠTEVA TUDI PRESLEDKE, KATERIH ŠTEVILO JE DOLOČENO Z UMAKNITVIJO VRSTICE. V DOLŽINI VRSTICE PA NI VSEBOVAN LEVI ROB, KI GA NASTAVIMO S POSEBNIM UKAZOM. V STANJU POLNENJA VRSTICE JE IZPISANA VRSTICA KRAJŠA ALI ENAKA VSTAVLJENI DOLŽINI VRSTICE, VENDAR SKRAJŠANA ZA TRENUTNO VELJAVNO UMAKNITEV. DOLŽINE VRSTIC, KI SO MANJŠE OD 6 (ZNAKOV), NISO DOPUSTNE. UKAZI PA SO TILE:

.LN +N NASTAVIMO DOLŽINO VRSTICE NA N. ZAČETNA VREDNOST (BREZ NASTAVITVE) JE 65 IN TA UKAZ NE POVZROČI LOMA VRSTICE. N MORA BITI V INTERVALU (6,255).

.IN +N UMAKNITEV VRSTICE ZA N PRESLEDKOV. PRI DOLŽINI VRSTICE L IN PRI UMAKNITVI N SE IZPIŠE NAJPREJ N PRESLEDKOV IN

```
.LN 40
.IN 10
.PI .AT -N XX
TA UKAZ DOLOČA,DA SE
V VRSTICI N AKTIVIRA MAKRO
XX. KATERIKOLI DRUG MAKRO,
KI JE BIL PREJE PRIREJEN
TEJ VRSTICI, SE ZAMENJA
Z MAKROJEM XX.
.IN Ø
TO JE OPIS UKAZA .AT .
```

(A)

(A) št.Č.

```
.AT -N XX TA UKAZ DOLOČA,DA SE V VRSTICI
          N AKTIVIRA MAKRO XX.
          KATERIKOLI DRUG MAKRO, KI JE
          BIL PREJE PRIREJEN TEJ
          VRSTICI, SE ZAMENJA Z MAKROJEM
          XX.
TO JE OPIS UKAZA .AT .
```

(B)

SLIKA 6. PRIMER UPORABE UKAZOV .IN IN .PI, KJER VIDIMO V (B) REZULTAT. UKAZ .IN Ø UKINE PREJSNJE STANJE UMAKNITVE ZA 10 PRESLEDKOV.

NATO PREOSTALIH L-N MEST V VRSTICI (TO JE TEKST). V ZAČETKU JE UMAKNITEV ENAKA NIČ IN TA UKAZ POVZROČI LOM VRSTICE.

.SI +N ENKRATNA UMAKNITEV VRSTICE ZA N PRESLEDKOV. SAMO ENA VRSTICA ZA TEM UKAZOM BO UMAKNJENA ZA N MEST. PRI OBSTOJEČI UMAKNITVI SE LAHKO UMAKNITEV ZMANJŠA Z UPORABO UKAZA .SI -M, TAKO DA JE TRENUTNA UMAKNITEV N-M. UKAZA .IN IN .SI STA KUMULATIVNA, VENDAR KONČNA VREDNOST ARGUMENTA NE SME BITI NEGATIVNA. TA UKAZ POVZROČI LOM VRSTICE.

.PI ST VSTAVI NIZ 'ST' V POLJE UMAKNITVE. NIZ 'ST', V KATEREM SE NE UPOŠTEVAJO VODEČI PRESLEDKI, SE VSTAVI V UMAKNITVENO POLJE (TAKOJ NA ZAČETKU VRSTICE), KI JE SICER NAPOLNJENO S PRESLEDKI. ČE JE NIZ DALJŠI OD UMAKNITVENEGA POLJA, SE TA NIZ OB IZPISU ODREŽE, TAKO DA NE PRESEŽE DANE DOLŽINE UMAKNITVE.

SLIKA 6 KAŽE PRIMER UMAKNITVE IN VSTAVITVE NIZA. TA PRIMER SE POJAVLJA TUDI PRI PISANJU TEGA TEKSTA (ČLANKA), KO OPISUJEMO POSAMEZNE UKAZE (ZGORAJ).

9. MAKROJI, ODVRNITVE IN VRSTIČNE PASTI

MAKRO JE ZAPOREDJE UKAZOV ALI TEKST, KI MU LAHKO PRIREDIMO IME TER GA KASNEJE KLIČEMO Z IMENOM. MAKROJSKA IMENA IMAJO DVA ZNAKA IN SE MORAJO RAZLIKOVATI OD OBSTOJEČIH IMEN UKAZOV IN DRUGIH MAKROJEV. MAKROJE DOLOČAMO ALI JIH SPREMINJAMO Z UPORABO UKAZA .DM ALI Z UPORABO IZHODNEGA ODVRNITVENEGA UKAZA .DI . OBSTOJEČIM MAKROJEM LAHKO ŠE KAJ PRIPNEMO (DODAMO, PRIDRUŽIMO) Z UPORABO UKAZOV .AM IN .DA . ČE JE MAKRO POIMENOVAN Z XX, GA POKLIČEMO OZIROMA RAZŠIRIMO V IZVAJANJE S POZIVOM OZIROMA Z NAVEDBO .XX .

PAST LAHKO NASTAVIMO GLEDE NA VRSTICO (POLOŽAJ NA STRANI) TAKO, DA SE AVTOMATIČNO V TEJ VRSTICI IZVRŠI DOLOČEN MAKRO, KO UPORABIMO UKAZ .AT . V SAMI DEFINICIJI MAKROJA REGISTRI NISO RAZŠIRJENI DO NUMERIČNIH VREDNOSTI; TO SE ZGODI ŠELE PRI IZVRŠEVANJU MAKROJA. V DEFINICIJI MAKROJA SE POSEBNI ZNAKI ŠE NE PREVEDEJO (NPR. RAZŠIRITEV TABULIRANJA ITN.).

MAKROJI SO LAHKO POLJUBNO ZAPOREDJE UKAZOV, MAKROJSKIH POZIVOV IN TEKSTA, TODA SAM MAKRO NE MORE DEFINIRATI DRUGEGA MAKROJA, LAHKO PA OBLIKUJE ODVRNITEV.

ODVRNITEV JE NEKE VRSTE MAKRO PO IZVRŠITVI, VENDAR JE DRUGAČE OBLIKOVANA. PROCESIRANI IZHOD JE LAHKO ODVRNJEN (PREUSMERJEN) V MAKROJSKI PROSTOR ZA POSEBNE NAMENE, KOT SO NPR. OBDELAVA OPOMB NA DNU STRANI ALI DOLOČITEV OBSEGA PROSTORA NA STRANI ZA POGOJNO SPREMINJANJE PARAMETROV STRANI (REGISTER 'V' VSEBUJE ŠTEVILKO ZADNJE ODVRNJENE VRSTICE) ITN. MED ODVRNITVIJO SE PROCESIRANJE TEKSTA OPRAVLJA NORMALNO, IZJEMA JE LE PROCESIRANJE LEVEGA ROBA. STANDARDEN NAČIN JE, DA JE ODVRNJENI TEKST ČITA Z REŽIMOM 'NEPROCESIRANJA' IN SE TAKO IZOGNEMO NADALJNEMU PROCESIRANJU VRSTICE.

KADAR PRIDE V POSTOPKU DEFINIRANJA MAKROJEV ALI OBLIKOVANJA ODVRNITEV DO PRESTOPA MAKROJSKEGA (REZERVIRANEGA) OBMOČJA, SE GENERIRA SPOROČILO O TEJ NAPAKI IN PROCESIRANJE SE USTAVI. NOBEN MAKROJSKI POZIV NE POVZROČI NOVE VRSTICE IN VRSTICA SE POLNI.

OGLEJMO SI NA KRATKO POMEN NASLEDNJIH UKAZOV:

.DM XX DOLOČI ALI SPREMENI DEFINICIJO MAKROJA Z IMENOM XX. DEJANSKI MAKRO (ALI NJEGOVO TELO) SE ZAČNE Z NASLEDNJO VRSTICO. MAKROJSKA DEFINICIJA SE IZVRŠUJE, DOKLER SE NE POJAVI ZNAK NJENEGA KONCA, KI JE '..' V ZAČETKU VRSTICE. MAKROJI NE SMEJO VSEBOVATI UKAZOV .DM, LAHKO PA OBLIKUJEJO ODVRNITVE.

.AM XX PRIPNI (PRIDRUŽI) NEKAJ K MAKROJU XX. TA UKAZ DELUJE TAKO KOT UKAZ .DM, LE DA SO SE NADALJNE VHODNE VRSTICE (ZA UKAZOM .AM) PRITAKNJENE K MAKROJU XX IN JE TAKO POTREBNO MANJ DODATNEGA PROSTORA, KOT ČE BI OBLIKOVALI NOV MAKRO.

.RM XX UKINI (ALI ZBRIŠI) MAKRO ALI ODVRNITEV Z IMENOM XX. S TEM UKAZOM JE BIL MAKRO XX ČRTAN IZ LISTE IMEN IN POZIVI TEGA IMENA NE BODO IMELI VEČ UČINKA.

.DI XX ODVRNI IZHOD V MAKROJSKI PROSTOR Z IMENOM XX. MAKRO Z IMENOM XX SE DEFINIRA ALI MODIFICIRA V TEJ TOČKI. MED ODVRNITVIJO SE NADALJUJE NADALJNE PROCESIRANJE TEKSTA NORMALNO, IZJEMA JE LE IZVAJANJE UKAZA LEVEGA ROBA (.LM). ODVRNITVENI POSTOPEK SE KONČA Z VČITANJEM DRUGEGA UKAZA .DI ALI .DA. ODVRNITEV NI MOGOČE VGNEZDITI. ŠTEVILO ODVRNJENIH VRSTIC SE HRANI V REGISTRU 'V', KI GA LAHKO KASNEJE NAVEDEMO.

.DA XX ODVRNI PRIPETO RAZLIČICO IZ UKAZA .DI. ENAKA PRAVILA VELJAJO ZA TA IN PREJŠNJI UKAZ.

.. TO JE KONEC DEFINICIJE MAKROJA.

.AT -N XX V VRSTICI N AKTIVIRAJ MAKRO Z IMENOM XX. VSAK MAKRO, KI JE BIL PREDHODNO PREDVIDEN ZA VRSTICO -N, SE ZAMENJA Z XX. TU ŠTEJEMO N OD ZGORAJ NAVZDOL NA STRANI (INDEKSA Ø IN 1 SE UPORABLJATA ZA VRH STRANI) IN -N OD SPODAJ NAVZGOR (ČE JE NPR. DOLŽINA STRANI 66, JE VRSTICA -1 PRAV VRSTICA 66). ČE NE NAVEDEMO MAKROJSKEGA IMENA V UKAZU, SE UMAKNE PAST IZ VRSTICE -N, ČE JE BILA TAM NASTAVLJENA.

.CH -N -M PRESTAVI PAST. PRESTAVI PAST, KI JE BILA NASTAVLJENA V VRSTICI -N V VRSTICO -M. ČE PRI -N NI PASTI, SE ZAHTEVA NE UPOŠTEVA.

.CH XX -M PRESTAVI PAST ZA MAKRO XX V VRSTICO -M.

OGLEJMO SI NASLEDNJI PRIMER. !MEJMO TALE VHODNI TEKST:

```
.SP
.AU 1
.NR A 13
.PL 10 ———— število vrstic na strani
.PN 27 ———— številka strani
.LN 47
.DM HD
:CE 2
POJASNILO #+A          makro HD (naslov)
---------------
..
.AT 1 HD ———— past za HD v 1. vrstici
.DM FO
:CE 1
STRAN --- % ---        makro FO (stran)
:PG
```

```
..
.AT -3 FO        past za FO v 8. vrstici
.HD              poziv makroja HD
    TA PRIMER KAŽE UPORABO MAKROJEV IN NJIHOVIH PASTI.
    ...
    MAKRO .HD OBLIKUJE NASLOV NA VSAKI STRANI, TAKO DA SE CENTRIRANO IZPIŠE
'POJASNILO' S TEKOČO ŠTEVILKO, KI SE ŠE SIMETRIČNO PODČRTA.
    ...
    MAKRO .FO IZPIŠE TEKOČO STRAN V SREDINI SPODNJEGA DELA STRANI.
    ...
    STRAN OBSEGA 10 VRSTIC IN V PRVI VRSTICI JE NASTAVLJENA PAST ZA MAKRO .HD;
V OSMI VRSTICI (ARGUMENT -3) PA IMAMO PAST ZA .FO.
    ...
    POZIV .HD V POSEBNI VRSTICI JE POTREBEN, DA SE IZPIŠE NASLOV ZE NA PRVI TEKOČI STRANI.
                                                    tekst za izpis
```

PROCESIRANJE TEKSTA Z GORNJIM ZAPOREDJEM UKAZOV OZIROMA Z MAKROJI IN PASTMI POVZROČI REZULTAT, KI JE RAZVIDEN IZ SPODNJEGA IZPISA, KO IMAMO ŠTIRI STRANI (OSTEVILČENE S 27, 28, 29, ZADNJA STRAN PA NI OSTEVILČENA), IN SICER:

POJASNILO 14

TA PRIMER KAŽE UPORABO MAKROJEV IN NJIHOVIH PASTI.

...

MAKRO .HD OBLIKUJE NASLOV NA VSAKI STRANI, TAKO DA SE CENTRIRANO IZPIŠE 'POJASNILO' S

STRAN --- 27 ---

POJASNILO 15

TEKOČO ŠTEVILKO, KI SE ŠE SIMETRIČNO PODČRTA.

...

MAKRO .FO IZPIŠE TEKOČO STRAN V SREDINI SPODNJEGA DELA STRANI.

...

STRAN --- 28 ---

POJASNILO 16

STRAN OBSEGA 10 VRSTIC IN V PRVI VRSTICI JE NASTAVLJENA PAST ZA MAKRO .HD; V OSMI VRSTICI (ARGUMENT -3) PA IMAMO PAST ZA .FO.

...

POZIV .HD V POSEBNI VRSTICI JE POTREBEN,

STRAN --- 29 ---

POJASNILO 17

DA SE IZPIŠE NASLOV ZE NA PRVI TEKOČI STRANI.

A.Ž.

10. TABULIRANJE

PRI TABULIRANJU SE TRENUTNO DOLOČENI HORIZONTALNI TABULIRNI KARAKTER ZAMENJA Z USTREZNIM ŠTEVILOM POLNILNIH KARAKTERJEV, TAKO DA SE ZAPOLNI PROSTOR DO NASLEDNJEGA USTAVITVENEGA TABULIRNEGA STOLPCA (V VRSTICI, KI SE TRENUTNO POLNI). POLNILNI KARAKTER JE NAVADNO PRESLEDEK, LAHKO PA GA TUDI POSEBEJ DEFINIRAMO Z UKAZOM .TF . DOLOČIMO LAHKO NAJVEČ 20 TABULIRNIH USTAVITEV IN JIH POSTAVIMO V NARAŠČAJOCE ZAPOREDJE. V ZAČETKU NIMAMO TABULIRNIH USTAVITEV IN TABULIRNI KARAKTER JE ASCII NIČLA (ØØH). TABULIRNI KARAKTER JE LAHKO VSAK NEALFANUMERIČNI ZNAK. V REŽIMU POLNENJA SE LAHKO ZGODI, DA BO UPORABA TABULIRANJA POVZROČILA NEUSTREZNA TABULIRNA POLJA.

IMAMO TELE UKAZE:

.TA N,.. NASTAVITEV TABULIRNIH USTAVITEV. MANJKAJOČE TABULIRNE USTAVITVE SO NIČNE (KOT DA JIH NI) IN 20 JIH LAHKO NASTAVIMO. USTAVITVENE VREDNOSTI LOČIMO S PRESLEDKI, VEJICAMI ALI Z DRUGIMI, NENUMERIČNIMI ZNAKI. TAKO IMAMO NPR. TABULIRNI UKAZ .TA 10,20,35,40.

.TF C NASTAVI TABULIRNI POLNILNI KARAKTER. TA JE NAVADNO PRESLEDEK, LAHKO PA GA DEFINIRAMO S POLJUBNIM NENUMERIČNIM PISNIM ZNAKOM. ČE 'C' NI DOLOČENO, IMAMO PRESLEDEK, SICER PA ZNAK, KI JE ZA 'C' VSTAVLJEN.

.TC C DEFINIRAJ TABULIRNI KARAKTER. V ZAČETKU JE TABULIRNI ZNAK NIČLA (GA NI), LAHKO PA GA DEFINIRAMO KOT POLJUBEN NENUMERIČNI PISNI ZNAK. ČE 'C' NI SPECIFICIRAN, POSTANE TABULIRNI ZNAK ZOPET NIČLA (GA UKINEMO).

11. TRODELNI NASLOVI

Z UKAZOM .TL OBLIKUJEMO NASLAVLJANJE (PISANJE NASLOVOV STRANI, POGLAVIJ), KI OBSEGA TRI POLJA: LEVO, SREDIŠČNO IN DESNO. LEVI IN DESNI NASLOV STA LEVO IN DESNO PORAVNANA (ZAPISANA V LEVI IN DESNI ROB STRANI ALI STOLPCA), SREDIŠČNI NASLOV PA JE CENTRIRAN (IZPISAN SIMETRIČNO). LAHKO UPORABIMO VSA TRI POLJA ALI PA NJIHOVO POLJUBNO KOMBINACIJO. PORAVNAVA SE OPRAVI GLEDE NA DOLŽINO NASLOVA, KI JE NEODVISNA OD DEFINIRANE DOLŽINE VRSTICE (TOREJ MORA BITI POSEBEJ DOLOČENA). V ZAČETKU JE DOLŽINA VRSTICE 65 ZNAKOV IN JE ENAKA DOLŽINI NASLOVA. UPORABA UKAZA .TL NE VPLIVA NA LOM VRSTICE IN PO IZPISU NASLOVA TEČE IZPIS VRSTIC NEMOTENO NAPREJ. UKAZ .TL SE UPORABLJA ZLASTI V MAKROJIH ZA IZPIS NASLOVOV (NAVADNO KOMPLEKSNIH) IN V MAKROJIH ZA IZPIS OPOMB NA DNU STRANI. NPR. .TL ''%'' POVZROČI IZPIS ŠTEVILKE STRANI V SREDIŠČU VRSTICE.

IMAMO LE DVA UKAZA:

.TL 'LEVO'SREDINA'DESNO'
POSTAVI NASLOVE PORAVNANO GLEDE NA POLJE (LEVO, CENTRIRANO, DESNO PORAVNANO). NIZI UKAZA 'LEVO', 'SREDINA' IN 'DESNO' SO TAKO LEVO, CENTRIRANO IN DESNO PORAVNANI GLEDE NA TRENUTNO DOLŽINO NASLOVA. VSAKO OD TEH POLJ JE LAHKO PRAZNO IN POLJUBEN NENUMERIČNI ZNAK SE LAHKO UPORABI NAMESTO OMEJEVALNIKA POLJA "'". ZNAK % SE ZAMENJA S TRENUTNO ŠTEVILKO STRANI, KO IMAMO ARABSKO ALI RIMSKO PREDSTAVITEV.

.LT -N NASTAVI DOLŽINO NASLOVA. DOLŽINA NASLOVA IN DOLŽINA VRSTICE STA LOČENA PARAMETRA. UMAKNITVE VRSTIC SE NE NANAŠAJO NA NASLOVE, LEVI ROB PA SE.

12. POGOJNI VHODNI UKAZI

VHODNI UKAZI IN MAKROJSKI POZIVI SE LAHKO UPORABLJAJO TUDI POGOJNO, NPR. V OBLIKI .IF

#A IF #B .XX . IMAMO:

```
.IF C UKAZ
.IF !C UKAZ
.IF N UKAZ
.IF !N UKAZ
```

'IF' JE POGOJNI UKAZ IN 'UKAZ' JE LAHKO VSAK UKAZ SISTEMA ALI IME MAKROJA. 'C' JE POGOJNI KOD IN JE LAHKO 'O' (LIHO) ALI 'E' (SODO) GLEDE NA LIHO ALI SODO ŠTEVILKO STRANI. 'N' JE POLJUBNO ŠTEVILO, REGISTER ALI IZRAZ S SEŠTEVANJEM ALI ODŠTEVANJEM. ČE JE POGOJ PRAVILEN (TA POGOJ JE IZPOLNJEN, ČE JE VREDNOST IZRAZA, REGISTRA ALI ŠTEVILA VEČJA OD NIČ), SE UKAZ ALI MAKRO IZVRŠI, SICER PA SE NE UPOŠTEVA. ČE JE PRED 'C' ALI 'N' ZNAK '!' (NEGACIJA), SE UKAZ IZVRŠI PRI NEIZPOLNJENEM POGOJU OZIROMA, KO JE VREDNOST IZRAZA, REGISTRA, ŠTEVILA MANJŠA ALI ENAKA NIČ.

13. PREKLAPLJANJE OKOLICE

PROCESOR TEKSTA IMA VRSTO PARAMETROV, KI KRMILIJO OBDELAVO TEKSTA IN TE PARAMETRE IMENUJEMO OKOLICA. PRAV TE PARAMETRE LAHKO SPREMENIMO VSE HKRATI Z UPORABO UKAZA PREKLOPA. IMAMO DVE OKOLICI, IN SICER Ø IN 1. OBE OKOLICI IMATA ENAKE ZAČETNE VREDNOSTI ZA VSE PARAMETRE. PARAMETRI TEH DVEH OKOLIC PA SO:

DOLŽINA VRSTICE,
UMAKNITEV
PORAVNAVA,
POLNENJE,
DOLŽINA NASLOVA,
VERTIKALNI VRSTIČNI PROSTOR,
ŠTEVNIK CENTRA,
AVTOMATIČNO INKREMENTIRANJE,
DELNO ZBRANE BESEDE IN
DELNO ZBRANE VRSTICE.

VSI OSTALI PARAMETRI SO GLOBALNI IN JIH NE MOREMO PREKLOPITI. PRIMERI GLOBALNIH PARAMETROV SO:

LEVI ROB,
ŠTEVILO STRANI,
TRENUTNO ŠTEVILO VRSTIC,
REGISTRI,
TABELE PASTI IN
MAKROJSKE DEFINICIJE.

KER SO DELNO ZBRANE BESEDE IN DELNO ZBRANE VRSTICE V OKOLICI, PREKLOP OKOLICE NE POVZROČI PREKINITVE (LOMA) IN OSTANKI (BESED, VRSTIC) SE OHRANIJO. IMAMO:

.EV N — SPREMENI OKOLICO N, KJER JE N ENAKO Ø ALI 1. ČE N NE NAVEDEMO, JE IZBRANA OKOLICA Ø.

14. POSEBNI KRMILNI UKAZI

OGLEJMO SI VRSTO UKAZOV ZA NAJRAZLIČNEJŠE FUNKCIJE. IMAMO:

.CP — UVEDI REŽIM VELIKIH ČRK. TA UKAZ OMOGOČA, DA UPORABIMO TERMINAL, KI IMA SAMO VELIKE ČRKE, DA Z NJIM OBLIKUJEMO TEKST ZA IZPIS, V KATEREM SE BODO POJAVLJALE MALE IN VELIKE ČRKE (SEVEDA NA USTREZNEM IZPISNEM TERMINALU). VSAKA ČRKA JE V NORMALNEM STANJU MAJHNA, VELIKA PA POSTANE, ČE PRED NJO POSTAVIMO ZNAK '@'. NIZI ČRK BODO VELIKE ČRKE, ČE JIH POSTAVIMO MED ZNAKA '↑'. ZNAK '@' DELUJE TU KOT POMIK IZ MALE NA VELIKO ČRKO TER ZNAK '↑' KOT FIKSIRANI POMIK, DOKLER GA Z DRUGIM ZNAKOM '↑' NE UKINEMO.

.NC — UKINI REŽIM VELIKIH ČRK. V ZAČETKU NIMAMO TEGA STANJA IN ZNAK POMIKA '@' TER ZNAK '↑' SE NE UPOŠTEVATA.

.ST — TA UKAZ JE USTAVITEV IN POVZROČI ZAČASNO PRENEHANJE PROCESIRANJA, NA ZASLON TERMINALA PA SE IZPIŠE BESEDA 'STOP'. ČE VTIPKAMO ČRKO 'S', BO SLEDIL IZSTOP IZ PROCESORJA. ČE VTIPKAMO KATERIKOLI ZNAK RAZLIČEN OD 'S', SE BO PROCESIRANJE NADALJEVALO. USTAVITVENI UKAZ POVZROČI LOM VRSTICE.

.EX — IZSTOPI IZ PROCESORJA. TU SE PROCESIRANJE USTAVI TAKO, KOT DA BI BIL CELOTEN VHODNI TEKST OBDELAN. TA UKAZ SE UPORABLJA ZLASTI V POVEZAVI Z UKAZOM .IF .

.PS — IZDAJ CELOTEN TEKST, VKLJUČNO Z UKAZI, NA IZHOD. TA UKAZ RABI KOT PRIPOMOČEK ZA POPRAVLJANJE TEKSTA, SAJ SE NA IZHOD PRENESEJO VSE VRSTICE, TOREJ TUDI UKAZNE. PRI TEM SE UKAZI NE INTERPRETIRAJO (IZVAJAJO) IN TAKO NIMAMO PROCESIRANJA TEKSTA. V TEM REŽIMU SE IZDA CELOTEN TEKST DO KONCA VHODNE PODATKOVNE ZBIRKE.

.RP — PONOVI PROCESIRANJE ZBIRKE (ALI BOLJE PONAVLJAJ). TA UKAZ POVZROČI PONAVLJAJOČE PROCESIRANJE ZBIRKE. TO STANJE JE UPORABLJIVO NPR. PRI PISANJU PISEM, V KATERIH SE POJAVLJAJO RAZLIČNI NASLOVNIKI, PREOSTALI TEKST PISMA PA JE ZA VSE NASLOVNIKE ENAK.

.U1 — UPORABNIŠKO DOLOČLJIVI UKAZ ŠTEVILKA 1. TA UKAZ MORA PROGRAMIRATI UPORABNIK IN TRENUTNO NIMA NOBENE FUNKCIJE.

.U2 — UPORABNIŠKO DOLOČLJIVI UKAZ 2. TU VELJA PODOBNO KOT PRI UKAZU .U1 .

.U3 — UPORABNIŠKO DOLOČLJIVI UKAZ 3. TU VELJA PODOBNO KOT ZA .U1 .

UKAZI .U1, .U2 IN .U3 POVZROČIJO IZVRŠITEV POLJUBNE UPORABNIŠKE FUNKCIJE, KI JO DOLOČIMO Z USTREZNIM PROGRAMOM. S TEMI UKAZI LAHKO UPOŠTEVAMO IN UPORABIMO POSEBNE LASTNOSTI TISKALNIKA, KI GA IMAMO ZA IZPIS. TAKŠNE LASTNOSTI SO: SPREMEMBA VELIKOSTI PISAVE ALI RAZMAKA MED ČRKAMI, POVRATEK V PREJŠNJO VRSTICO, DRUGE KRMILNE FUNKCIJE PISALNIKA ITN.

15. ZUNANJA KOMUNIKACIJA

IMAMO DVA UKAZA ZA KOMUNICIRANJE MED PROCESORJEM IN UPORABNIKOM V ČASU PROCESIRANJA TEKSTA. UKAZ .TM SE UPORABLJA ZA IZDAJANJE POSEBNIH NAVODIL NA TERMINALNI ZASLON; TAKA NAVODILA SO VSTAVITEV ALI NASTAVITEV PAPIRJA (NPR. DRUGI STOLPEC), ZAMENJAVA PISALNE GLAVE ITN. DRUGI UKAZ .GI PA SE LAHKO UPORABI PRI PISANJU PISEM, KO VSTAVLJAMO POSEBEN TEKST (NPR. NASLOVNIKE) V ČASU PROCESIRANJA TEKSTA. TAKO IMAMO:

.TM ST — IZDAJ SPOROČILO 'ST' NA TERMINAL. TU JE 'ST' POLJUBEN NIZ ZNAKOV OZIROMA BESED. VODEČI PRESLEDKI SE NE UPOŠTEVAJO. SPOROČILO SE POŠLJE NA TERMINAL IN TA UKAZ SE LAHKO UPORABI PRED UKAZOM USTAVITVE (.ST), TAKO DA DOBIMO USTREZNO NAVODILO NA ZASLON TERMINALA, KO SE PROCESIRANJE TEKSTA USTAVI.

.GI ST SPREJMI TEKST 'ST' IZ TERMINALA. TEKST 'ST', KI JE POLJUBEN NIZ ZNAKOV, SE IZPISUJE NA TERMINAL IN HKRATI SE ZNAKI, KI SESTAVLJAJO 'ST', S TEM UKAZOM VSTAVLJAJO V VHODNI NIZ ZA PROCESIRANJE. TA UKAZ SE UPORABLJA ZA VSTAVLJANJE IMEN IN NASLOVOV PRI PISANJU PISEM. UKAZ .GI SE KONČA Z ZNAKOM 'CR' (POVRATEK VALJA) IN TAKO LAHKO VPIŠEMO LE ENO VRSTICO Z ENIM UKAZOM .GI . PO IZVRŠITVI TEGA UKAZA VSEBUJE REGISTER G ŠTEVILO ZNAKOV, KI SO BILI VTIPKANI V NIZ (BREZ ZNAKA 'CR').

16. RAZLIČNI ZNAKI IN OPOMBE

PROCESOR TEKSTA LAHKO UPORABLJA POSEBEN UKAZ ZA VSTAVLJANJE KOMENTARJEV. IMAMO:

.* UKAZ KOMENTARSKEGA POLJA SE UPORABLJA ZA VSTAVITEV KOMENTARJA V VHODNI TEKST. TA KOMENTAR SE NE UPOŠTEVA S PROCESORJEM IN Z NJIM SE NE OBLIKUJE IZHOD IN SAM KOMENTAR SE NE IZDA.

IMAMO TUDI VRSTO POSEBNIH ZNAKOV IN NEKATERE SMO ŽE PREJE OPISALI:

\ ZNAK '\' SE UPORABLJA ZA UKINITEV POMENA NEKEGA ZNAKA, KI GA IMA TA ZNAK V PROCESORJU. ČE ŽELIMO TISKATI ZNAK % (NE PA ŠTEVILKO STRANI), MORAMO POSTAVITI \% . ZA TISKANJE ZNAKA '\' BI MORALI POSTAVITI \\ , ZA TISKANJE ZNAKA @ PA \@ V REŽIMU VELIKIH ČRK ITN.

@ ZNAK @ POMENI VELIKO ČRKO V REŽIMU KAPITALIZACIJE (UKAZ .CP). NJEGOV UČINEK JE POMIK NA VELIKO ČRKO.

↑ OMEJEVALNIK ZA NIZE VELIKIH ČRK. TA ZNAK IMA UČINEK FIKSIRANEGA POMIKA NA VELIKE ČRKE IN POMEN UKINITVE TEGA POMIKA. PROCESOR MORA BITI PRI TEM V STANJU KAPITALIZACIJE (UKAZ .CP).

OZNAČEVALNIK REGISTROV A, B, ... , Z.

. PIKA '.' JE KRMILNI ZNAK UKAZA, ČE SE NAHAJA V ZAČETKU VRSTICE. UKAZ JE DVOČRKOVNO IME, KI JE LAHKO TUDI MAKRO.

: DVOPIČJE JE KRMILNI ZNAK UKAZA, KO NIMAMO LOMA VRSTICE (SEVEDA SAMO, ČE JE V ZAČETKU VRSTICE). UČINEK JE PODOBEN UČINKU ZNAKA '.', VENDAR SE TU LOM VRSTICE ZADUŠI.

% TO JE ZNAK ŠTEVILKE STRANI. KJERKOLI SE V TEKSTU NAHAJA ZNAK '%', SE NAMESTO NJEGA NATISNE ŠTEVILKA TEKOČE STRANI.

CAN TO JE ZNAK 'CONTROL X' NA TASTATURI (18H), KI POVZROČI BRISANJE VRSTICE, KI SMO JO TIPKALI V PROCESOR.

PROCESOR TEKSTA AVTOMATIČNO VSTAVLJA PO DVA PRESLEDKA ZA ZNAKI '.', '!' IN '?', ČE JE ZA NJIMI EDEN ALI VEČ PRESLEDKOV. TO PA SE NE ZGODI, ČE ZA TEMI ZNAKI NI PRESLEDKA.

17. SKLEP

V TEM ČLANKU SMO NA KRATKO OPISALI UKAZE PROCESORJA TEKSTA IN NAKAZALI NJEGOVO UPORABNOST. V DRUGEM DELU ČLANKA BOMO OPISALI KONKRETNE MAKROJE, ZGRADBO PROCESORJA IN NADALJNE PRIMERE. PREDVSEM BOMO V NADALJEVANJU POKAZALI RABO PROCESORJA ZA UPORABNIKA, KI MU NI POTREBNO PODROBNO ZNANJE O UKAZIH, TEMVEČ LAHKO UPORABLJA LE NEKAJ IZBRANIH IN NJEGOVIM POTREBAM PRIREJENIH MAKROJEV. NA TA NAČIN POSTANE PROCESOR TEKSTA ŠIROKO UPORABLJIV IN MASOVNI INSTRUMENT V ADMINISTRACIJI IN DOKUMENTACIJI.

V NADALJEVANJU ČLANKA BOMO OPISALI TUDI TIPIČNE MIKRORAČUNALNIŠKE KONFIGURACIJE, KI SO SMISELNE ZA PROCESIRANJE TEKSTA, IN SICER OD CENENIH DO BOGATO OPREMLJENIH. DODALI BOMO TUDI KRATEK SEZNAM VSEH UKAZOV.

INFORMATICA 3/1979

PRIMENA AKKMR METODE U ANALIZI KRUŽNE MREŽE RAČUNARA ZA PODRŠKU DISTRIBUIRANE BAZE PODATAKA

GOJKO A. BABIĆ

UDK: 681.3 : 336.712

ELEKTROTEHNIČKI FAKULTET U SARAJEVU, ODSJEK ZA INFORMATIKU

U ovom članku, AKKMR metod je primenjen u analizi kružne mreže midi/mini-računara za podršku procesiranja transakcija u distribuiranoj bazi podataka. Ta mreža računara uključuje deset radnih računara i komunikacionu mrežu koja koristi DLCN-ov ili Pierce-ov mehanizam za transmisiju poruka. Izvedeni su analitički izrazi za iskorišćenost procesora, prosečnu dužinu redova čekanja i prosečno vreme odziva u sistemu, kao funkcije lokaliteta referenci, opterećenja sistema i intenziteta servisiranja. Opisani su i odgovarajući simulacioni modeli. Rezultati dobijeni iz analitičkih i simulacionih modela su poredjeni i njihovo slaganje je dobro.

APPLICATION OF AKKMR METHOD TO ANALYZE LOOP COMPUTER NETWORK TO SUPPORT DISTRIBUTED DATA BASE: In this paper, AKKMR method is used to analyze the loop network of midi/mini-computers to support transaction processing against a distributed data base. This computer network consists of ten hosts and a communication network using DLCN or Pierce message transmission mechanism. We derive analytical expressions for calculating the utilization of processors, the average queue lengths and the average response time in the system, as functions of locality of references, system load, and service rates. Also, corresponding simulation models are described. Results obtained from analytical and simulation models are compared and agreement between these results is very good.

1. UVOD

U članku /BAB78/ dati su opis i osnovni principi metoda za analizu karakteristika kružne mreže računara - AKKMR metod. (Hardverska struktura kružne mreže računara je data na slici 1.) Metod se sastoji od dva osnovna koncepta: a)modeliranja čitave komunikacione mreže s jednim poslužiocem i b) analiziranje modela redova čekanja za mrežu računara iterativno, razmatrajući samo jednu komponentu, tj. jedan radni računar (host), u toku svake iteracije. Metod je veoma generalan i jednostavan, a može se koristiti u analizi heterogenih kružnih mreža računara koje imaju proizvoljan broj radnih računara i koje koriste komunikacione mreže sa bilo kojim transmisionim mehanizmom. Postoje dva osnovna razloga koji su uslovili AKKMR metod:problem inkopatibilnosti i problem kompleksnosti. Metod daje rešenje za oba problema, a njegova opštost sugeriše da bi se sa malim modifikacijama mogao koristiti i za analizu mreža računara sa proizvoljnom topologijom. U istom članku dati su neki rezultati primene AKKMR metode u analizi kružne mreže računara sa relativno jednostavnom strukturom radnih računara.

U ovom članku, AKKMR metod je primenjen u analizi jedne realnije kružne mreže računara i čitav postupak analize je opisan detaljno. Sistem koji je analiziran je mreža midi/mini-računara za podršku procesiranja transakcija (upita) u distribuiranoj bazi podataka. Ta mreža računara uključuje deset radnih računara, a razmatrane su komunikacione mreže koje koriste DLCN-ov /REA75/ ili Pierce-ov /PIE72/ mehanizam za transmisiju poruka.

Ovaj članak je organizovan na sledeći način. U drugom poglavlju su dati opisi sistema koji se analizira i mreže redova čekanja koja se dobije kao odgovarajući analitički model. U trećem poglavlju, korišćenjem AKKMR metode, izvedeni su analitički izrazi za iskorišćenost procesora, prosečne dužine redova čekanja i prosečno vreme odziva u sistemu, kao funkcije lokaliteta referenci, opterećenja sistema i intenziteta servisiranja. Da bi se potvrdila tačnost metode i dobijenih analitičkih izraza, u četvrtom poglavlju prvo je opisan simulacioni model, a zatim su poredjeni analitički i simulacioni rezultati. U zadnjem poglavlju sumirani su najznačajniji rezultati i date su motivacije korišćenja kružne komunikacione mreže kao komunikacionog sistema za lokalne mreže računara.

2. OPIS I MODEL SISTEMA

Sistem koji se ispituje je kružna mreža midi/mini-računara za podršku procesiranja transakcija u distribuiranoj bazi podataka, identičan sistemu koji je razmatran u /LAB77/. Model svakog radnog računara sastoji se od dva dela: komunikacionog procesora (KP) i sistema za podršku diskova (SD). Takodje je izvestan broj terminala vezan za svaki komunikacioni procesor (slika 2).

Komunikacioni procesor prihvata transakcije iz terminala, predprocesira ih i šalje na procesiranje lokalnom sistemu za podršku diskova, ako se transakcije mogu zadovoljiti lokalno (lokalne transakcije) ili u komunikacionu mrežu ako se transakcije mogu zadovoljiti jedino kod nekog od udaljenih sistema za podršku diskova (daljinske transakcije). Da bi se transakcija procesirala, tj. dobio odziv može biti potrebno da se napravi nekoliko pristupa diskovima. Odgovori na lokalne

Rad je referiran na XIV.Simp.Informatica,Bled,okt.1979

transakcije se posle postprocesiranja na lokalnom komunikacionom računaru prosleđuju do terminala. Kada stignu do svog odredišta daljinske transakcije se predprocesiraju u tom radnom računaru. Odzivi na ove transakcije posle postprocesiranja na udaljenom komunikacionom računaru se šalju kroz komunikacionu mrežu do lokalnog radnog računara i prosleđuju do terminala.

Na slici 3. je dat model mreže redova čekanja koji se razmatra u toku svake iteracije AKKMR metode pri analiziranju datog sistema. Sada su detaljno opisana kretanja transakcija i odziva kroz model. Transakcije ulaze u i-ti radni računar iz terminala brzinom L(i) i čekaju na predprocesiranje kod lokalnog komunikacionog procesora (poslužilac 2 i-te komponente koji opslužuje brzinom M(2,i)). One tamo konkurišu za opsluživanje sa odzivima na lokalne i daljinske transakcije, koje zahtevaju postprocesiranje i sa daljinskim transakcijama (poruke niza A(i)), koje zahtevaju predprocesiranje.

Ako transakcije mogu da budu zadovoljene lokalno, one se šalju lokalnom sistemu za podršku diskova (poslužilac 3 i-te komponente koji opslužuje brzinom M(3,i)). Tu one konkurišu za procesiranje daljinskim transakcijama ostalih priključenih komponenti. Posle jednog procesiranja transakcija može da se sa odredjenom verovatnoćom pridruži redu čekanja poslužioca 3 radi dodatnog procesiranja, jer više pristupa disku može biti neophodno da bi se transakcija zadovoljila. Posle konačnog procesiranja odgovarajući odzivi odlaze do lokalnog komunikacionog procesora na postprocesiranje. Ovog puta oni konkurišu za opsluživanje lokalnim i daljinskim transakcijama, koje zahtevaju predprocesiranje i sa odzivima na daljinske transakcije, koje zahtevaju postprocesiranje. Posle postprocesiranja odzivi na lokalne transakcije se šalju nazad do terminala.

Ako transakcije sa terminala vezanih za i-ti radni računar trebaju biti procesirane na nekom udaljenom radnom računaru, posle predprocesiranja one se šalju u komunikacionu mrežu (poslužilac 3 i-te komponente koji opslužuje brzinom M(1,i)). One tu konkurišu za procesiranje sa odzivima na daljinske transakcije iz drugih komponenti (odzivi na transakcije iz niza A(i), koje su zadovoljene u i-tom radnom računaru). Daljinske transakcije i-tog radnog računara odlaze do drugih radnih računara (kao deo niza C(i)) kroz komunikacionu mrežu. Posle dolaska u udaljeni, recimo j-ti, radni računar, svaka transakcija se ponovo predprocesira kod j-tog komunikacionog procesora (poslužilac 2 j-te komponente sa brzinom M(2,j)). Posle procesiranja (poslužilac 3 j-te komponente sa brzinom M(3,j)), vrši se postprocesiranje kod j-tog komunikacionog procesora. Odzivi se zatim šalju kroz komunikacionu mrežu (poslužilac 1 j-te komponentesa brzinom M(1,j)), da bi posle dolaska u lokalni radni računar bili direktno prosleđjeni do terminala odakle su transakcije bile poslane (poruke niza B(i)).

3. ANALITICKI REZULTATI

U ovom poglavlju će biti izvedeni analitički izrazi za prosečno vreme odziva, iskorišćenost procesora i prosečne dužine redova čekanja, kao funkcije lokaliteta referenci, opterećenja sistema i intenziteta procesiranja.

Definicije i predpostavke. Predpostavlja se da je sistem u stacionarnom režimu i da nema padova delova sistema ili transmisionih grešaka. Takodje se predpostavlja da su sledeći parametri poznati i dati:

1. matrica servisiranja zahteva, $\mathcal{F}=(f(i,j))$, gde je $f(i,j)$, $i,j=1,2,\ldots,N$ (N je broj radnih računara u sistemu), deo zahteva koji su generisani sa terminala vezanih za i-ti radni računar i trebaju biti zadovoljeni u j-tom radnom računaru;

2. brzina dolazaka L(i), i=1,2,...,N, je brzina generisanja zahteva sa terminala vezanih na i-ti radni računar (pretpostavlja se Poissonov proces);

3. brzine opsluživanja M(2,i) i M(3,i), i=1,2,...,N, su brzine opsluživanja poslužioca 2, odnosno poslužioca 3, i-tog radnog računara (predpostavlja se eksponencijalna distribucija);

4. c(i), i=1,2,...,N, je prosečan broj pristupa disku potreban da se zadovolji jedna transakcija(pretpostavlja se geometriska distribucija);

5. 1/k(i), i=1,2,...,N, je prosečna dužina poruke poslane iz i-tog radnog računara (predpostavlja se eksponencijalna distribucija);

6. C je kapacitet komunikacionog kanala;

7. B je dužina adresnog polja u poruci.

Sada se sledeća dva parametra mogu izračunati:

1. F(i), i=1,2,...,N, je frakcija transakcija koje trebaju biti zadovoljene u nekom udaljenom radnom računaru,

$$F(i) = \sum_{\substack{j=1 \\ j\neq i}}^{N} f(i,j) \qquad (1)$$

Zapazi da je brzina dolazaka poruka u nizu B(i) jednaka F(i)L(i);

2. S(i), i=1,2,...,N, je brzina dolazaka daljinskih transakcija u i-ti radni računar,

$$S(i) = \sum_{\substack{j=1 \\ j\neq i}}^{N} f(j,i)L(j) \qquad (2)$$

Zapazi da su brzine dolazaka poruka u nizovima A(i) i C(i) jednake S(i), odnosno S(i)+F(i)L(i).

Računanje verovatnoća prelaza. Definisane su sledeće verovatnoće prelaza:

1. q(1,i), i=1,2,...,N, verovatnoća da posle opsluživanja kod poslužioca 2 i-tog radnog računara transakcija ide do poslužioca 3 istog radnog računara;

2. q(2,i), i=1,2,...,N, verovatnoća da posle opsluživanja kod poslužioca 2 i-tog radnog računara transakcija ide do poslužioca 1 istog radnog računara;

3. q(3,i), i=1,2,...,N, verovatnoća da posle opsluživanja kod poslužioca 3 i-tog radnog računara transakcija ide do poslužioca 2 istog radnog računara.

Verovatnoća prelaza iz čvora X u čvor Y se dobije kao odnos brzine dolazaka zahteva u čvor Y iz čvora X i ukupne brzine odlazaka zahteva iz čvora X. Kako se izlaz iz poslužioca 2 i-tog radnog računara sastoji od sledećih nizova:

1. niz a, daljinske transakcije koje odlaze brzinom F(i)L(i);

2. niz b, lokalne transakcije koje odlaze brzinom (1 - F(i))L(i);

3. niz A(i), brzina odlazaka je S(i);

4. niz c, odzivi na lokalne transakcije koji odlaze brzinom (1 - F(i))L(i);

5. niz d, odzivi na daljinske transakcije koji dolaze brzinom S(i);

a od ovih nizova jedino nizovi b i A(i) odlaze do poslužioca 3, tada za q(1,i) imamo

$$q(1,i) = \frac{(1-F(i))L(i)+S(i)}{F(i)L(i)+(1-F(i))L(i)+S(i)+(1-F(i))L(i)+S(i)} = \frac{(1-F(i))L(i)+S(i)}{(2-F(i))L(i)+2S(i)} \qquad (3)$$

Slično, nizovi a i d odlaze do poslužioca 1, pa za q(2,i) imamo

$$q(2,i) = \frac{F(i)L(i)+S(i)}{(2-F(i))L(i)+2S(i)} \qquad (4)$$

Za dobijanje izraza za q(3,i) koristi se drugačiji pristup. Kako svaki c(i)-ti izlaz iz SD ide u KP tada je

$$q(3,i) = \frac{1}{c(i)} \qquad (5)$$

Računanje izraza za a(i,j). Kako je mreža redova

čekanja koju razmatramo Jakson-ovog tipa, koristeći rezultate Jackson-ove teoreme /JAC57/, može se formirati sledeći sistem linearnih jednačina u matričnoj formi koji odgovara mreži redova čekanja na slici 3,

$$\begin{bmatrix} a(1,i) \\ a(2,i) \\ a(3,i) \end{bmatrix} = \begin{bmatrix} 0 \\ L(i)+S(i) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & q(2,i) & 0 \\ 0 & 0 & q(3,i) \\ 0 & q(1,i) & 1-q(3,i) \end{bmatrix} \begin{bmatrix} a(1,i) \\ a(2,i) \\ a(3,i) \end{bmatrix} \quad (6)$$

gde je a(r,i), r=1,2,3, brzina dolazaka kod r-tog poslužioca. Rešenje za ovaj sistem je:

$$\begin{aligned} a(1,i) &= S(i) + F(i)L(i) \\ a(2,i) &= (2 - F(i))L(i) + 2S(i) \quad (7) \\ a(3,i) &= c(i)((1 - F(i))L(i) + S(i)) \end{aligned}$$

Kako se pretpostavlja da svi radni računari u mreži imaju identične konfiguracije, ali ne i iste parametre, izrazi (3)-(7) važe za sve i=1,2,...,N.

Kada su konfiguracije radnih računara različite (što je sasvim normalno u realnim situacijama), svaki radni računar se razmatra posebno. To znači da se verovatnoće prelaza i rešenje za sistem linearnih jednačina računaju za svaki radni računar zasebno. Veličina sistema lineranih jednačina je R(i), gde je R(i) broj poslužilaca kod i-tog radnog računara. Kada se nebi koristio AKKMR metod, tada bi sistem linearnih jednačina koji odgovara mreži redova čekanja koja modelira čitav sistem, imao $\sum_i R(i)$ jednačina.

U simetričnom slučaju svi parametri su isti, npa

$$\begin{aligned} q(1) &= \frac{1}{2+F} \\ q(2) &= \frac{2F}{2+F} \quad (8) \\ q(3) &= \frac{1}{c} \end{aligned}$$

$$\begin{aligned} a(1) &= 2FL \\ a(2) &= (2 + F)L \quad (9) \\ a(3) &= cL \end{aligned}$$

gde je F=F(i), q(k)=q(k,i), a(k)=a(k,i), L=L(i) i c=c(i) za k=1,2,3 i i=1,2,...,N.
Može se lako proveriti da su izrazi za parametre dobijene u /LAB77/ koji odgovaraju izrazima za a(i,j) identični izrazima (9). Medjutim može se uočiti da je AKKMR metod jednostavniji, a i generalniji jer važi kako za simetrične, tako i za asimetrične slučajeve.

Računanje parametara komunikacione mreže. Kao što je već rečeno čitava komunikaciona mreža se aproksimira eksponencijalnim poslužiocem, čiju prosečnu brzinu opsluživanja treba izračunati. Pored pretpostavke o eksponencijalnoj dužini poruka, pretpostavlja se da su poruke iz radnih računara generisane poisson-ovim procesima. Parametri za te dve distribucije su poznati: a(1,i) (koji je izračunat) i 1/k(i) (koji je dat). Da bi se mogla izvršiti analiza kružne komunikacione mreže, pored kapaciteta C, dužine adresnog polja B, potrebno je odrediti i matricu prometa P = (P(i,j)), gde P(i,j), i,j=1,2,...,N, označava frakciju prometa generisanog iz i-tog radnog računara čije je odredište j-ti radni računar /LIU77, HAY71/. P(i,j) je dato sa izrazom:

$$P(i,j) = \begin{cases} \dfrac{f(i,j)L(i) + f(j,i)L(j)}{F(i)L(i) + S(i)} & i \neq j \\ 0 & i = j \end{cases} \quad (10)$$

Zapazi da je izraz (10) opšti i ne ovisi od konfiguracija radnih računara ili transmisionog mehanizma koji se koristi.

Sada su svi parametri koji se traže za analizu kružne komunikacione mreže definisani i izračunati. Da bi se dobilo prosečno vreme kašnjenja poruke T koristiće se odgovarajuća analiza. Tako na primer, u slučaju DLCN sistema koristiće se analiza iz /LIU77/, dok se za Pierce-ovu mrežu koriste rezultati iz /HAY71/. Tada se može izračunati prosečno vreme opsluživanja poslužioca koji aproksimira komunikacionu mrežu M(1,i), i=1,2,...,N, kao

$$M(1,i) = \frac{1}{T} + a(1,i) \quad (11)$$

Uoči da ako je a(i,j) ≥ M(i,j) za bilo koje i,j=1,2,...,N, sistem je nestabilan i u tom slučaju vreme kašnjenja i dužine redova rastu bez limita.

Kod mreža redova čekanja Jackson-ovog tipa, svaki čvor se ponaša kao neovisan M/M/1 sistem čekanja, pa za računanje parametara koji nas interesuju poslužiće poznato rešenje za taj sistem. Ako bi model uključivao više raznih klasa transakcija i više tipova poslužilaca tada bi rezultati iz /BAS75/ bili korišćeni.

Iskorišćenost procesora. Iskorišćenost procesora je važan projektni parametar. (Iskorišćenost komunikacione mreže se dobije iz analize odgovarajeće mreže, pa neće biti razmatrana.) Za iskorišćenost procesora imamo sledeći izraz

$$U(i,j) = \frac{a(i,j)}{M(i,j)} \quad (12)$$

gde je U(i,j), i=2,3 i j=1,2,...,N, iskorišćenost i-tog poslužioca j-tog radnog računara.

Dužine redova čekanja. Prosečna dužina redova čekanja, tj. prosečan broj poruka (transakcija i njihovih odziva) u redovima čekanja, su takodje važni projektni parametri, koji se koriste za odredjivanje veličine odgovarajućih bafera u sistemu. Prosečan broj poruka kod i-tog poslužioca j-tog radnog računara N(i,j), i=1,2,3, i j=1,2,...,N. su dati sa

$$N(i,j) = \frac{a(i,j)}{M(i,j) - a(i,j)} \quad (13)$$

Vremena odziva. Korisnik računarskog sistema je najviše zainteresovan za odziv na njegov zahtev (transakciju), koji se definiše kao vreme koje protekne izmedju dolaska transakcije u lokalni KP i povratka odgovarajućeg odziva do terminala. Prosečno vreme čekanja kod i-tog poslužioca j-tog radnog računara TQ(i,j), i=1,2,3 i j=1,2,...,N, je dato sa

$$TQ(i,j) = \frac{1}{M(i,j) - a(i,j)} \quad (14)$$

Uoči da je T=TQ(1,j) za j=1,2,...,N. Pomoću izraza (14) može se izračunati prosečno vrme odziva za daljinske transakcije iz i-tog radnog računara koje se zadovoljavaju u j-tom radnom računaru TR(i,j), i≠j i i,j=1,2,...,N. TR(i,j) se sastoji od:

1. TQ(2,i), prosečno vreme predprocesiranja u lokalnom, i-tom, radnom računaru;
2. T, prosečno vreme kašnjenja poruke u komunikacionoj mreži od i-tog do j-tog računara;
3. TQ(2,j), prosečno vreme predprocesiranja u udaljenom, j-tom radnom računaru;
4. c(j)TQ(3,j), prosečno vreme procesiranja u j-tom radnom računaru;
5. TQ(2,j), prosečno vreme postprocesiranja u j-tom radnom računaru;
6. T, prosečno vreme kašnjenja poruke u komunikacionoj mreži od j-tog do i-tog računara;

TR(i,j) je suma svih ovih vremena i dato sa

$$TR(i,j)=TQ(2,i)+2T+2TQ(2,j)+c(j)TQ(3,j) \quad (15)$$

dok je prosečno vreme odziva za daljinske transakcije iz i-tog radnog računara TR(i), i=1,2,...,N, dato sa

$$TR(i) = \frac{1}{F(i)} \sum_{\substack{j=1 \\ j \neq i}}^{N} TR(i,j)f(i,j) \quad (16)$$

Prosečno vreme odziva za daljinske transakcije u sistemu TR, je dato sa

$$TR = \frac{\sum_{i=1}^{N} F(i)L(i)TR(i)}{\sum_{i=1}^{N} F(i)L(i)} \quad (17)$$

Prosečno vreme odziva za lokalne transakcije TL(i), i=1,2,...,N, sastoji se od:

1. TQ(2,i), prosečno vreme predprocesiranja u i-tom radnom računaru;
2. c(i)TQ(3,i), prosečno vreme procesiranja u i-tom radnom računaru;
3. TQ(2,i), prosečno vreme postprocesiranja u i-tom radnom računaru.

Kako je TL(i) suma svih tih vremena, imam

$$TL(i) = 2TQ(2,i) + c(i)TQ(3,i) \quad (18)$$

Prosečno vreme odziva za lokalne transakcije u sistemu TL, je dato sa

$$TL = \frac{\sum_{i=1}^{N} (1 - F(i))L(i)TL(i)}{\sum_{i=1}^{N} (1 - F(i))L(i)} \quad (19)$$

Konačno, prosečno vreme odziva da se zadovolje transakcije sa terminala vezanih za i-ti radni računar TD(i), i=1,2,...,N, je dato sa

$$TD(i) = F(i)TR(i) + (1 - F(i))TL(i) \quad (20)$$

a prosečno vreme odziva u sistemu TD, je dato sa

$$TD = \frac{\sum_{i=1}^{N} L(i)TD(i)}{\sum_{i=1}^{N} L(i)} \quad (21)$$

Uoči da su izrazi (16),(17),(19),(20) i (21) opšti i ne ovise o konfiguraciji računara.

Sa ovim je završeno izvodjenje analitičkih izraza za više projektnih parametara za datu kružnu mrežu računara. U sledećem poglavlju su poredjeni simulacioni i analitički rezultati dobijeni iz izraza izvedenih u ovom poglavlju.

4. SIMULACIONI MODELI I POREDJENJE ANALITIČKIH I SIMULACIONIH REZULTATA

U toku predhodne analize napravljeno je više predpostavki i aproksimacija. Da bi još jednom verifikovali AKKMR metod i dobijene analitičke izraze izvršena su simuliranja kružne mreže računarakoja se razmatra, pa su se analitički i simulacioni rezultati mogli porediti. Simulacioni modeli su konstruisani da oponašaju što je moguće tačnije stvarno funkcionisanje mreže računara. U modeliranju je korišćen GPSS simulacioni programski jezik, a simulacije su izvršene na IBM 370/168 računaru.

Opis simulacionih modela. Iako je bit preciznija mera za dužinu poruke da bi se ostvarile efikasnije i ekonomičnije simulacije, korišćen je znak (1 znak = 8 bita) umesto bita kao jedinica dužine poruke. Tako su sva vremena izražena u proizvoljnim znak-vremenskim jedinicama. Simulacioni modeli su se sastojali od deset radnih računara priključenih na komunikacioni kanal kapaciteta C = 1o ili 50 Kbit/sec. Svaka poruka koja je multipleksirana u komunikacioni kanal sastojala se od konstantnog dela (100 znakova) i eksponencijalnog dela sa srednjom vrednošću od 60 znakova (maksimalna dužina eksponencijalnog dela je 600 znakova). Propagaciono kašnjenje je ignorisano, dok je svaki komunikacioni interfejs doprinosio kašnjenju od 2 vremenske jedinice za proveru adrese u adresnom polju poruke.

Komunikacione mreže su koristile DLCN-ov ili Pierce-ov mehanizam za transmisiju poruka. Za modele Pierce-ove komunikacione mreže, veličina paketa od 90 znakova je izabranaza datu prosečnu dužinu poruka. Odlučeno je da se samo jedan kompletan paket može staviti u komunikacioni kanal, kao što je bilo uradjeno i u nekim drugim simulacijama /HAY71/. Kašnjenje od 70 jedinica vremena je postavljeno izmedju poslednjeg i prvog interfejsa. Kako svih deset komunikacionih interfejsa zajedno unose kašnjenje od 20 jedinica vremena, na taj način kašnjenje od 90 jedinica vremena je formirano, što odgovara potrebnom vremenu da se transmituje jedan paket od 90 znaka.

25 terminala je vezano na svaki radni računar sa prosečnim vremenom razmišljanja (vreme koje protekne izmedju dolaska zadnjeg odziva do terminala i slanja sledeće transakcije iz istog terminala) jednakom 90 sekundi. Zapazi da smo u analitičkoj analizi predpostavili da su zahtevi generisani poisson-ovim procesima brzinom 90/25= 3,6 transakcija u sekundi.

Analizirani su simetrični i asimetrični slučajevi. U simetričnom slučaju razmatrane su homogene mreže računara, gde svi radni računari imaju ista prosečna vremena predprocesiranja, procesiranja i postprocesiranjajednaka 3,2 msec, 32 msec, odnosno 3,2 msec (sva vremena su eksponencijalno distribuirana), a daljinske transakcije jednog radnog računara su uniformno distribuirane naostalih devet radnih računara. Broj pristupa disku po jednoj transakciji je uniformno distribuiran izmedju 1 i 29, sa srednjom vrednošću c(i) jednakom 15. Za ovaj sistem uniformna distribucija je relativno tačna i realistična predpostavka, koja je korišćena i u /LAB77/. Podsetimo se da smo u analitičkoj analizi uzeli da je q(3,i)= 1/c(i)=1/15 verovatnoća da zahtev neće ponovo otići u red čekanja SD-a. To implicira geometrisku distribuciju za broj pristupa disku. Poznato je da je varijansa ove distribucije veća od varijanse uniformne distribucije. Konsekventno, analitički model je konzervativniji (kašnjenja su veća) od simulacionog modela. Naglašavamo da nije poznat metod za inkorporiranje uniformne distribucije u model mreža redova čekanja.

U asimetričnom slučaju razmatrana je mreža koja se sastoji od jednog velikog sistema i devet manjih. Karakteristike manjih radnih računara su identične onima u simetričnom slučaju, osim što se sve daljinske transakcije adresiraju na veliki radni računar. Prosečna vremena predprocesiranja, procesiranja i postprocesiranja su 1,6 msec, 12,8 msec, odnosno 1,6 msec. Transakcije sa terminala vezanih za veliki radni računar su uvek lokalno zadovoljene.

Radi ograničenog prostora detalji simulacionog procesa i analiziranje datog modela nisu uključeni u ovaj članak. Članak sa opštim opisom simulacionog procesa i analiziranje simulacionog modela (odredjivanje trenutka početka uzimanja rezultata iz simulacionog modela, dužina trajanja simulacije, analiziranje dobijenih rezultata, itd) sa opisom autorovih iskustava u vezi ovih simulacionih modela je u pripremi /BAB80/.

Poredjenje analitičkih i simulacionih rezultata. Na slikama 4-8 dati su neki analitički rezultati (pune linije) i simulacioni rezultati (specijalni znakovi). Na svim grafovima x-osa je procenat daljinskih transakcija PDZ. Na slikama 4 i 5 prikazani su rezultati za DLCN-ov sistem za prosečno vreme odziva u sistemu TD (simetrični i asimetrični slučajevi). Na slikama 6 i 7 su grafovi za Pierce-ovu mrežu računara za iste parametre. Slika 8 prikazuje iskorišćenost SD-ova U(3,i) za asimetrični slučaj. Zapazi da U(3,i) ne zavisi od kapaciteta komunikacionog kanala i mehanizama za transmisiju poruka.

Može se uočiti da je slaganje izmedju analitičkih i simulacionih rezultata veoma dobro, naročito za veće vrednosti C.

5. ZAKLJUČAK

U ovom članku AKKMR metod je primenjen u analizi kružne mreže računara za podršku distribuirane baze podataka. Dobijeni su analitički rezultati za prosečno vreme odziva u sistemu, dužine redova čekanja i iskorišćenje procesora, kao funkcije lokaliteta referenci, opterećenja sistema i brzine opsluživanja. Citav postupak je opisan detaljno tako da bi se mogao lako prilagoditi i primenti u analizi mreža računara sa drugačijom konfiguracijom radnih računara.

U toku razvijanja analitičkog modela uvedeno je više predpostavki i aproksimacija. Da bi se verifikovali analitički rezultati izvršena je simulacija korišćenjem GPSS simulacionog jezika. Slaganje analitičkih i simulacionih rezultata je veoma dobro.

Na kraju bi želeli da istaknemo pogodnosti kružnih mreža računara kao lokalnih mreža. Razvoj midi/mini računarskih sistema koji su relativno jeftini sugeriše njihovu integraciju u snažnije sisteme. Kako je sa današnjom komunikacionom tehnologijom moguće imati ekonomski opravdane komunikacione kanale velikih kapaciteta, reda Mbit/sec, samo na malim udaljenostima, u zadnje vreme lokalne mreže računara privlače pažnju mnogih istraživača i postoji značajan trend u tom pravcu. Tako naprimer, IFIP radna grupa 6.4 za lokalne mreže računara (IFIP WG on Local Computer Networks) je osnovana ove godine.

Kružne komunikacione mreže kao komunikacioni sistem za lokalne mreže računara radi svoje strukture, funkcionisanja i tehnološkog razvoja nude mnoge prednosti. Najznačajnije prednosti su jeftin i jednostavan komunikacioni interfejs (manje od $500), niska konstrukciona cena (potrebno je samo dodati novi komunikacioni interfejs bez ozbiljnijeg uticaja na ostatak sistema), velike brzine prenosa (1-10 Mbit/sec), lako maršutiranje (postoji jedinstven put između bilo koja dva čvora), jednostavno emitovanje poruka - "broadcasting" (poruka se kopira u svakom komunikacionom interfejsu, a odstranjuje se kada dodje do interfejsa koji je izvršio transmisiju) i laka kontrola protoka (komunikacioni interfejs samo čeka na slobodan i neiskorišćen deo komunikacionog kanala da izvrši transmisiju). Pregled kružnih komunikacionih mreža sa distribuiranom kontrolom i njihovo poredjenje su dati u /BAB79/.

REFERENCE

BAB78 G. A. Babić, "Analiza karakteristika kružnih mreža računara (opis AKKMR metode)," Informatica 78, Bled, oktobar 1978.

BAB79 G. A. Babić, "Lokalne mreže računara: Kružne komunikacione mreže," Jugoslovenski simpozijum o informacionim sistemima, Beograd, maj 1979.

BAB80 G. A. Babić, " Simulacija kao metod za ispitivanje karakteristika računarskih sistema," u pripremi.

BAS75 F. Baskett i autori, "Open, Closed, and Mixed Networks of Queues with Different Classes of Customers," JACM, april 1975.

HAY71 J. F. Hayes i D. N. Sherman, "Traffic Analysis of a Ring Switched Data Transmission System," Bell System Tech. Journal, novembar 1971.

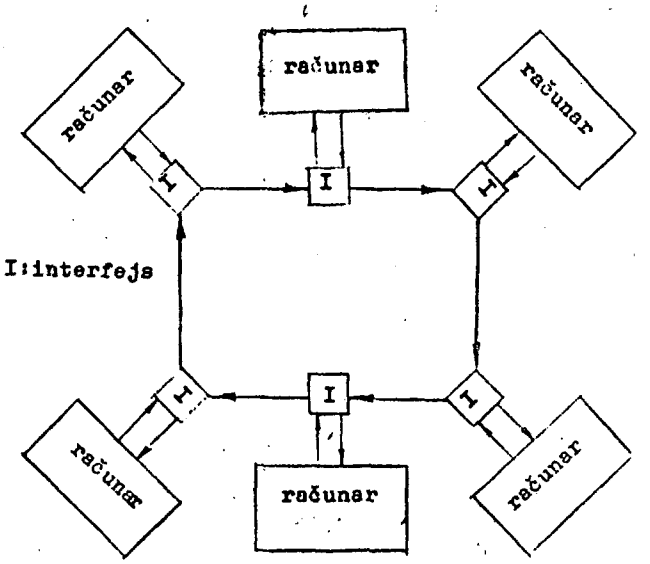
JAC57 J. R. Jackson, "Network of Waiting Lines" Operations Research, 1957.

LAB77 J. Labetoulle i autori, "A Homogeneous Comp. Networks: Analysis and Simulation" Computer Networks, maj 1977.

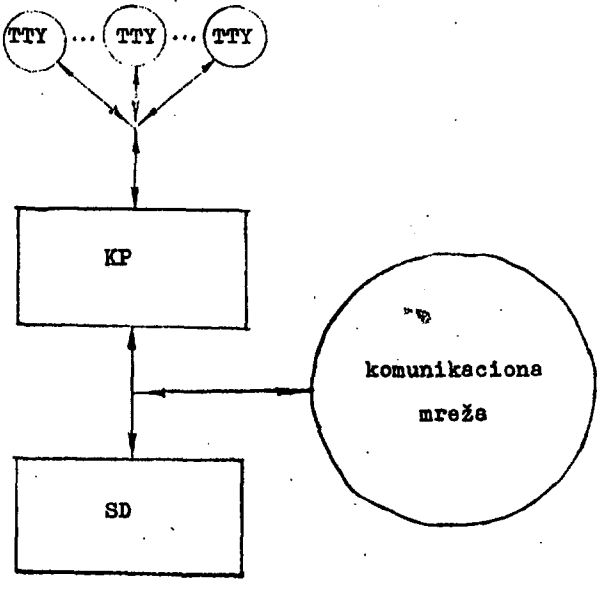
LIU77 M. T. Liu, G. A. Babić i R. Pardo, "Traffic Analysis of the Distributed Loop Computer Network (DLCN)," National Telecomunication Conf., Los Angeles, S.A.D. decembar 1977.

PIE72 J. R. Pierce, "Network for Block Switching of Data," Bell System Tech. Journal, jul/avgust 1972.

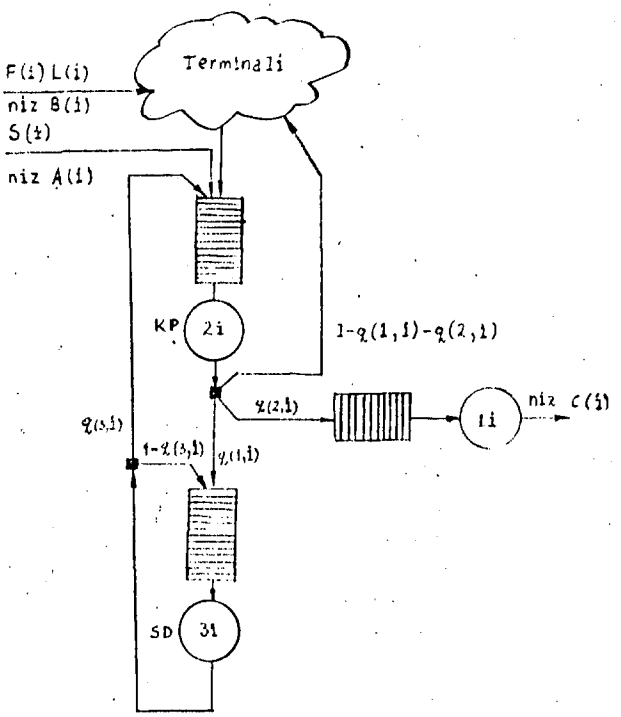
REA75 C. C. Reames i M. T. Liu, "A Loop Network for Simulteneous Transmission of Variable-Length Messages," 2. Annual Symp. on Comp. Architecture, januar 1975.



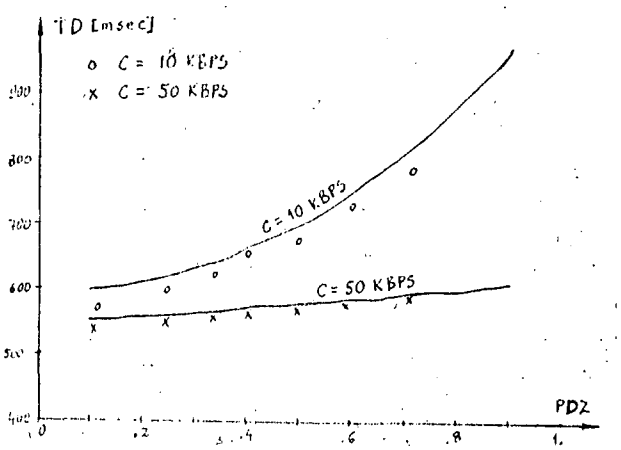
Slika 1. Hardverska struktura kružne mreže računara



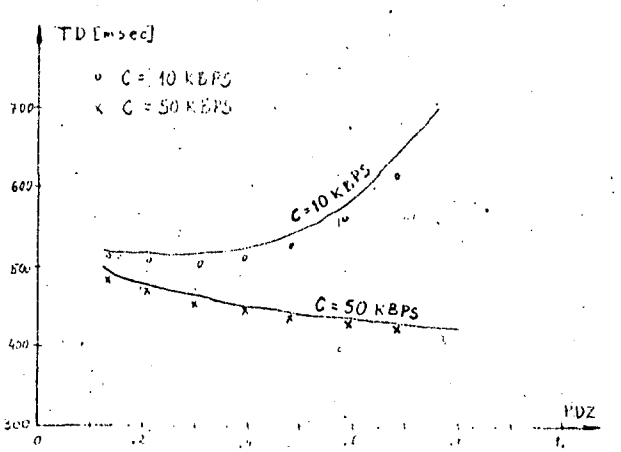
Slika 2. Model radnog računara



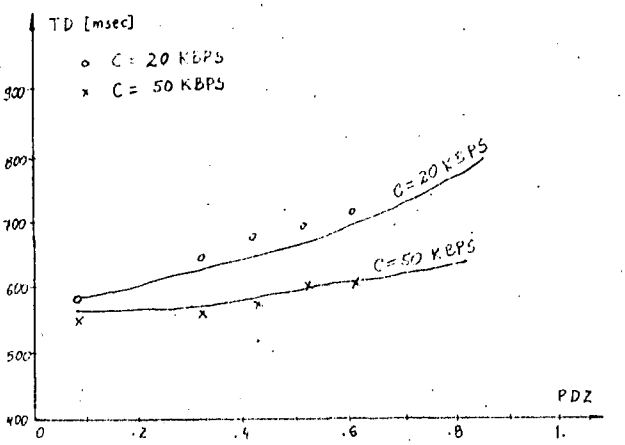
Slika 3. Mreže redova čekanja koji se razmatra u toku svake iteracije



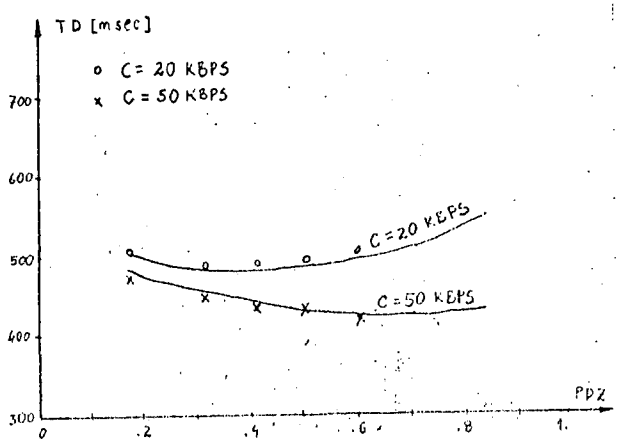
Slika 4. DLCN-ova mreža, simetrični slučaj
Prosečno vreme odziva



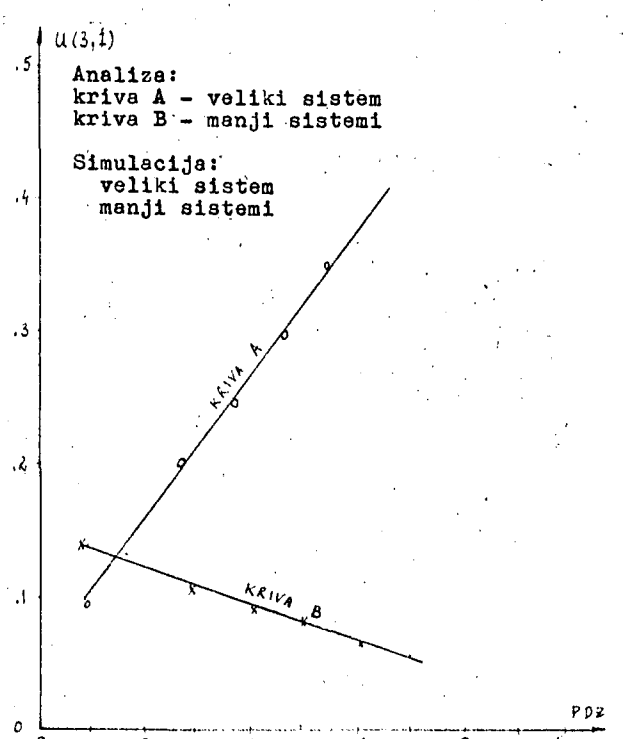
Slika 5. DLCN-ova mreža, asimetrični slučaj
Prosečno vreme odziva



Slika 6. Pierce-ova mreža, simetrični slučaj
Prosečno vreme odziva



Slika 7. Pierce-ova mreža, asimetrični slučaj
Prosečno vreme odziva



Slika 8. Asimetrični slučaj, iskorišćenost sistema za podršku diskova

INFORMATICA 3/1979

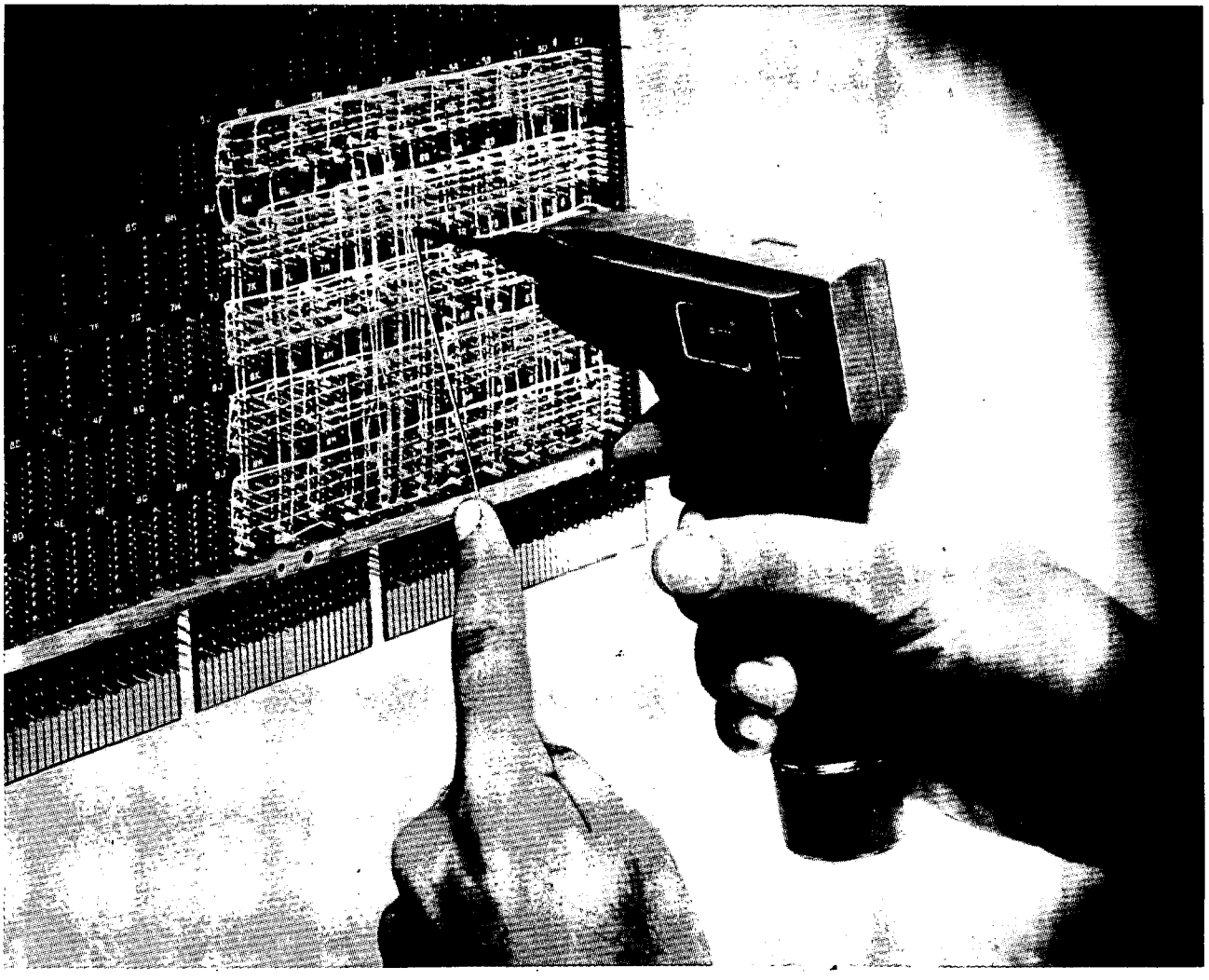
TEHNOLOGIJA OŽIČEVANJA I.

R. ČOP
M. KOVAČEVIĆ

UDK: 621.791

Članek je dvodelen in obravnava tehnologijo ožičevanja z uporabniškega stališča. V članku je podana primerjava z ostalimi metodami povezovanja vezij v elektroniki glede na ekonomičnost in vrsto uporabe. Nadalje je opisana spremljajaoča materialna oprema kot so: trni, povetovalne plošče, žico in njene lastnosti, izolacijo in njeno snemanje glede na kvaliteto kontaktnega spoja. V tem delu je tudi opisana razlika med modificiranim in običajnim ovijanjem in orodje, ki to omogoča. Prvi del je posvečen predvsem osnovni tehnologiji ožičevanja za vsakega uporabnika. Drugi del pa je namenjen industrijskemu orodju za ožičevanje in testiranje kvalitete.

WIRE WRAPPING TECHNOLOGY
The article presents in two parts the technology of wire wrapping from the users point of view. Comparison between different methods of connecting electric circuits regarding economy and alternative connection methods is presented. All material support like: pins, boards, wire and isolation and the difference between modified and regular wrapping are also discused.



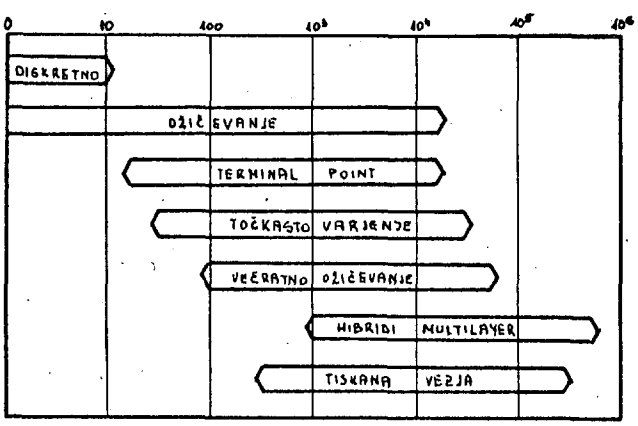
1) UVOD

Osnovno vprašanje, ki si ga postavljamo je, kdaj in kako uporabljati ožičevanje. Imamo več metod za povezovanje kontaktov, vsaka ima določene prednosti in slabosti. Osnovno vodilo pri izbiri določene metode je prostor, ki ga zavzemajo komponenete na vezni plošči. Odločitev za dločeno metodo je včasih težavna. (sl.1)

2) PRIMERJAVA METOD POVEZOVANJA ELEMENTOV

Za izhodišče bomo vzeli sl.1. Na tej sliki so prikazani tehnološki postopki, ki se glede na navedene količine uporabljajo za izdelavo vezij.

Primernost izdelave vezij s pomočjo točkastega varjenja, kontaktnega spoja in večkratnega ožičevanja se nanaša samo na manjše proizvodne serije. Vse zgoraj naštete metode pa so dodatno omejene tudi z višjo ceno razvoja, ceno orodja in omejitvami pri konstrukcijskih možnostih. Ožičevanje in diskretno spajkanje uporabljamo za izdelavo prototipov, ali majhnih serij, vendar je samo ožičevanje zelo primrerno za izdelovanje večji serij.



Sl. 1. Tehnika povezav glede na količino in vrsto tehnologije.

V poprečnem proizvodnem obsegu pomeni ožičevanje priporočljivo alternativo za večino aplikacij. Med drugim so tudi naslednje prednosti: preprosta tehnika, hitra proizvodnja (od prototipa do izdelka cca. 7 dni), ekonomičnost, kompaktnost (dovoljuje veliko gostoto elementov na površini vezja - enako kot pri dvostranskem tiskanem vezju), fleksibilnost (lahko mu kadarkoli spremenimo obliko in zasnovo vezja). Zaradi enakega delovnega medija, mnogokrat nimamo dodatnih stroškov pri prehodu iz prototipne v proizvodnjo serijo.

Spajkanje dopušča zelo visoko gostoto elementov po površini vezja. Vendar pa še vedno potrebujemo izredno drago opremo in posebno predpripravo in pripravo vezja. Spremembe in popravki so mnogokrat težko izvedljivi (zaradi ponavljanja vseh faz priprave). Spajkanje se zato priporoča pri količini 1000 enot vezij ob nizki proizvodni ceni glede na enoto. Pri serijah od 1000 enot vezij navzgor postanejo tiskana vezja, hibridna vezja in večslojna vezja cenejša od ožičevanja. Toda tudi pri tej količini se lahko uporablja ožičevanje, skoraj do spodnjega dela visoko količinske proizvodnje oz. do mej kjer je to rentabilno.

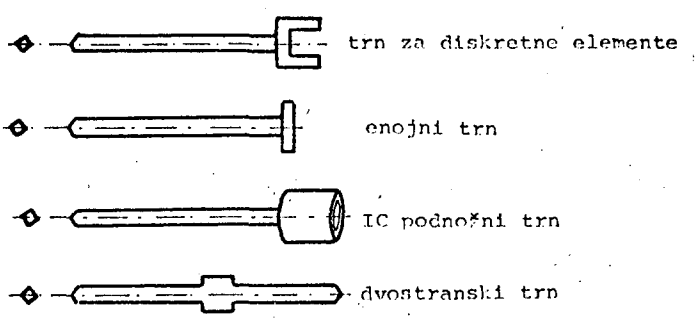
Vsekakor omejitev ne obstaja samo za ožičevanje pri visokih proizvodnih količinah, ampak tudi pri številu povezav na enoto. V primeru, da se število povezav na enoto povečuje se premikajo na levo tudi diagrami na sl.1.

3) SPREMLJAJOČA MATERIALNA OPREMA

Za ožičevanje potrebujemo samo ploščo opremljeno z enojnimi DIP (dual inline package) podnožji za elemente, ki imajo ožičevalne trne. V večji konfiguraciji imamo lahko več medseboj povezanih plošč, ali povezovalno polje z različnimi velikostmi trnov in različnimi medsebojnimi razdaljami, vse to pa povečuje konstrukcijske zmogljivosti.

3a) Trni za ovijanje

Edina omejitev, ki nastopa pri trnih za ovijanje je potreba po vsaj dveh ostrih robovih, da se dobi ustrezen električni kontakt. Običajna pošča za ožičevanje vsebuje kontaktne trne 0.91 mm (0.025 cole) na 2.4 mm (0.100 cole) mreži. Material iz katerega so trni narejeni je največkrat fosfor - bron, posrebren ali pozlačen. Na razpolago je več oblik trnov, ki se na tržišču dobijo v velikih količinah. (sl. 2)

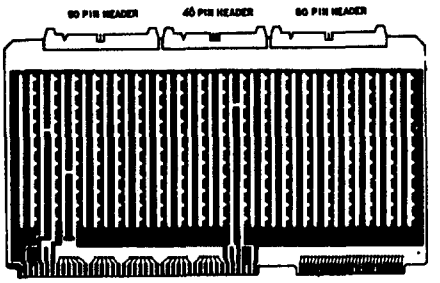


Sl. 2. Najbolj pogoste oblike trnov.

3b) Povezovalne plošče s tiskano mrežo

Plošče so narejene iz različnih materialov, kot so: epoksi, polyester, termoplastik, eno ali dvostransko tiskane z Cu slojem. Uporabnik lahko izbira različne izvedbe plošč, ki se razlikujejo po funkcionalnosti. Najbolj pogosto se uporabljajo plošče z ali brez V/I vodnikov, že vstavljenimi podnožnimi trni, DIP podnožji, razporejenimi vodili za Vcc in GND, kondenzatorji za blokiranje napetosti, vsemi vrstami konektorjev in ostalimi možnostmi za izbiro. Uporabnik lahko izbira med mnogimi velikostnimi formati, oblikami konektorjev in ohišij, (sl.3)

Na tržišču so na voljo univerzalne ožičevalne plošče za mini in mikro računalniške sisteme kot so: AMI 680MDC, AMI EVK300, Altair, CAMAC, Data General, DEC PDP-8, DEC PDP-11, DEC LSI11, Hewlet Packard, Heatkit H11, IMSAI 8080, INTEL 8080, INTEL SBC, Mostek SDB, Motorola 6800, National SC/MP, Prolog, POLY 88, RCA Cosmac, Textronix 8001, Texas Instruments 990/960, Zilog Z80-MCB. Vse te polšče za ožičevanje proizvaja tvrdka Garry Manufacturing.



Sl. 3. Univerzalna plošča z rasterjem za DIP podnožje, napetostnimi vodniki, I/O konektorji za INTEL SBC 80/10.

Cena narašča od 10 dinarjev za DIP podnožje do 500 din za majhno logično kartico. Cena ožičevalnega polja za vodilo srednje velikosti je približno 6000 dinarjev in preko nekaj sto tisoč dinarjev za več povezanih plošč z vsemi materialnimi dodtaki. Vendar tu ne moremo podati dobre primerjave zaradi različnih proizvajalcev in cen.

3c) Žica za ožičevanje

Žico izberemo glede na vrsto uporabe. Običajno se uporablja. žica od 22 - 30 AWG (0.65 - 0.25 mm). V telekomunikacijah se je uveljavil standard za debelino žice od 22 - 24 AWG (0.65 - 0.50 mm) v elektroniki pa standard za debelino žice 30 AWG (0.25 mm), vendar pa v principu ni omejitev pri izbiri debeline žice.

Najvažnejša lastnost žice je električna prevodnost. Druga zahteva je elastičnost. Žica je med procesom ovijanja izpostavljena upogibanju in raznim mehanskim napetostim. Pri takih pogojih uporabe pa običajno poči krhka žica. V razredu 18 - 22 AWG (1.02 - 0.65 mm) mora žica zdržati minimalno 20 procentno podaljšanje. Za razred 24 - 32 AWG (0.50 - 0.20 mm) se priporoča vsaj 15 procentno podaljšanje žice.

V vsakem primeru se ne priporoča aluminijasta žica, ki teh lastnosti nima . Uporabnik lahko izbira med navadnim bakrenim vodnikom, pocinjenim ali posrebrenim vodnikom. Posrebreni vodniki so priporočljivi zaradi odpornosti proti koroziji in podaljšujejo življenjsko dobo orodju. Ožičevanje je osnovano na čistem kovinsko kovinskem kontaktu med žico in trnom, tako da posrebrena žica ne nudi dodatnih prednosti.

3d) Izolacija žice

Najbolj znana izolacija za žico je KYNLAR in ostali PVC tipi. Izolacija mora imeti visoko dialektričnost, nizko ceno in različne barve.

Izolacija se ne sme nikoli tesno sprijeti z žico, zradi lažjega snemanja. Za uporabo pri višjih teperaturah in vibracijah se priporoča uporaba TEFLON ali MIL-ENE izolacije. Te vrste izolacije imajo veliko elastičnost in termično stabilnost, so odporne proti kemičnim vplivom v primeru, da jih uporabljamo pri prisotnosti topil ali korozivnih plinov. Slabost teh vrst izolacije je visoka cena in relativno težko odstranjevanje z vodnika.

Kynlar izolacija se uporablja predvsem pri avtomatičnemu snemanju. izolacije iz vodnika z "Cut - Strip - Wrapp" orodji. Pri tej vrsti orodja so snemalne sile preračunane za ta tip izolacije, tako da se ta tip orodja NE da uporabljati za ostale tipe izolacije. To orodje se tudi NE priporoča za ovijanje žice debeline 30 AWG (0.25 mm).

4) DOLŽINA SNETE IZOLACIJE IN IZBIRA OVIJANJA

Glede na ceno in kvaliteto ovijanja moramo upoštevati več parametrov. Izbrati moramo žico in dolžino snete izolacije. Dolžina snete izolacije se direktno pozna pri št. ovojev. Kolikor daljši je konec žice brez izolacije, tolikokrat se žica lahko ovije okoli trna. Nekaj osnovnih pravil za dolžino žice, kjer snemamo izolacijo je prikazanih na sl.4 v odvisnosti od debeline žice.

WIRE SIZE		"SHINER" LENGTH OF STRIPPED WIRE FROM		TO	
AWG	MM	INCHES	MM	INCHES	MM
20 GA.	0,80	1-5/16"	33,33	1-11/16"	42,86
22 GA.	0,65	1-5/16"	33,33	1-9/16"	39,68
22-24 GA.	0,65-0,50	1-5/16"	33,33	1-9/16"	39,68
24 GA.	0,50	1-5/16"	33,33	1-9/16"	39,68
26 GA.	0,40	1-5/16"	33,33	1-11/16"	42,86
28 GA.	0,32	7/8"	22,22	1-1/8"	28,57
30 GA.	0,25	7/8"	22,22	1-1/8"	28,57

Sl. 4. Dolžina snete izolacije za ožičevanje glede na debelino žice.

Ovijalna sila pritiska žico na trn in število ovojev se končno odraža kot kvaliteta kontakta. Z naraščanjem števila ovojev narašča tudi število kontaktnih točk, oziroma pada upornost celotnega spoja. Z določitvijo ovijalne sile, ki jo lahko kontroliramo tudi med delovnim procesom se omogoči ustrezen pritisk med žico in trnom, kar zagotavlja ustrezen kovinsko kovinski spoj med žico in trnom. Nakratko, če narašča dolžina snete izolacije se izboljšujejo tudi električne in mehanske lastnosti, to pa tudi zviša ceno. Končno je tudi upornost kontakta, ki jo dosežemo z minimalnimi zahtevami že v razredu 10-E4 Ohma, tako da dodajanje navojev ne poveča prevodnosti.

Samo okroglo žico se lahko ovije okoli trna. Tako dobimo običajen tip ovijanja. Lahko pa se še navije dodatnih 1 1/2 ovoja izolacije, stem se dobi modificirani tip ovijanja. Izolacija ovita okoli trna služi za kompenziranje mahanske stabilnosti in ublažitve pritiska žice ob prvi rob trna, vendar samo pri manjših debelinah žice. (sl. 5)

Sl. 5. Povečana slika trna ovitega z žico
a) običajno ovijanje
b) modificirano ovijanje

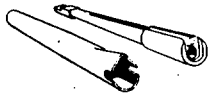
Pri debelini žice 18 - 24 AWG (1,02 - 0.5 mm) mehanska stabilnost običajnega ovijanja zadovoljuje zahtevam. Toda pri debelini žice 24 AWG (0.5 mm) se pojavljajo različne vibracije, tako da se priporoča modificirano ovijanje. Izkaže se tudi, da je modificirano ovijanje bolj prikladno za uporabo. Tako se je primerno v praksi ustalila uporaba naslednje debeline žice:

* V telekomunikacijah 22 - 24 AWG (0.5 - 0.6 mm)
* V elektroniki 30AWG (0.25 mm)

za način ovijanja

* Za premer žice od 18 - 24 AWG (1.02 - 0.5 mm) je tipično običajno ovijanje
* Za premer žice od 26 - 32 AWG (0.4 - 0.2 mm) je tipično modificirano ovijanje.

Sl. 6. Tulec (levo) in vodilo.

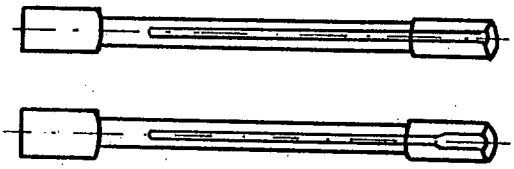


5) OPREMA IN ORODJE ZA OŽIČEVANJE

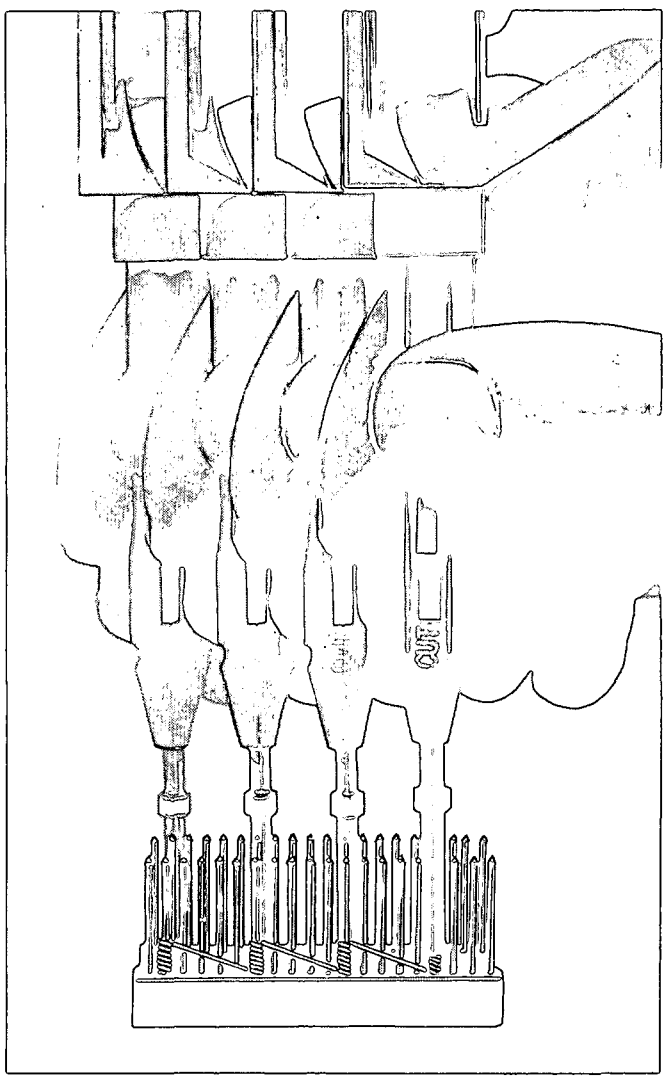
Izbira tipa ovijanja je preprosta funkcija, ki je odvisna samo od ovijalnega vodila. Bistveni del celotnega postopka ožičevanja sta "BIT" in "SLEEVE", VODILO in TULEC. Pri izbiri vodila pa moramo upoštevati naslednje lastnosti (sl. 6)

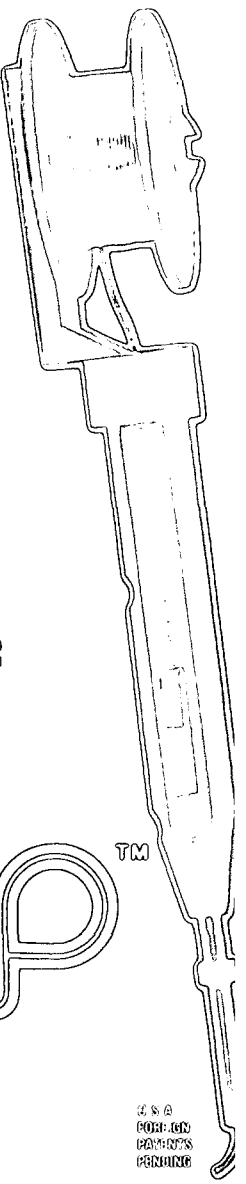
a) Odprtina za žico mora biti dovolj velika, da sprejme žico katero želimo ovijati.
b) Kontaktna luknja vodila mora biti dovolj dolga, da prekrije kontakt katerega želimo ovijati.
c) Vodilo mora biti konstruirano za določen tip ovijanja, katerega želimo uporabljati,(običajno, modificirano) (sl. 7)

Sl. 7. Vodilo za običajno in modificirano ovijanje. Vodilo za modificirano ovijanje ima na začetku večjo odprtino za 1 1/2 ovoja izolacije.



NEW!





WHY CUT?
WHY STRIP?
WHY SLIT?

WHY NOT...

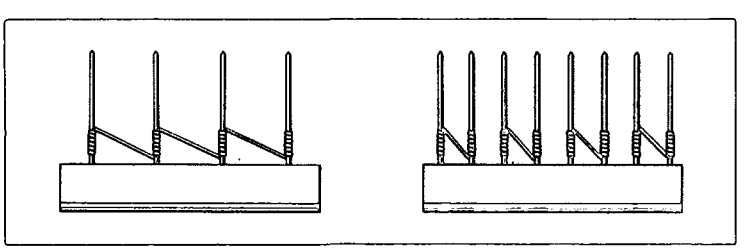
JUST WRAP™

WIRE-WRAPPING TOOL

U.S.A.
FOREIGN
PATENTS
PENDING

- AWG 30 Wire
- .025" Square Posts
- Daisy Chain or Point To Point
- No Stripping or Slitting Required
  ....JUST WRAP™....
- Built In Cut Off
- Easy Loading of Wire
- Available Wire Colors:
  Blue, White, Red & Yellow

	COLOR	PART NO.
JUST WRAP TOOL WITH ONE 50 FT. ROLL OF WIRE	BLUE	JW-1-B
	WHITE	JW-1-W
	YELLOW	JW-1-Y
	RED	JW-1-R
REPLACEMENT ROLL OF WIRE 50 FT.	BLUE	R-JW-B
	WHITE	R-JW-W
	YELLOW	R-JW-Y
	RED	R-JW-R



DAISY CHAIN

POINT TO POINT

OK MACHINE & TOOL CORPORATION 3455 CONNER ST., BRONX, N.Y. 10475 (212) 994-6600/TELEX 125091

NOVICE IN ZANIMIVOSTI

FEDIDA - IZUMITELJ VIDEOTEKSTA

EN SAM INŽENIR JE V POZNIH SESTDESETIH LETIH PRI BRITANSKI POŠTI SANJARIL O SISTEMU, IZ KATEREGA SE JE KASNEJE RAZVILA SODOBNA NOVA INDUSTRIJA. TA SISTEM JE DOBIL IME PRESTEL TER JE POMENIL TEHNIKO GLEDANJA PODATKOV (VIEWDATA), TO JE POVEZAVE TELEVIZIJSKEGA TERMINALA PREKO TELEFONSKEGA OMREŽJA Z RAČUNALNIŠKO BAZO PODATKOV. TA SISTEM JE BIL TAKO PREDHODNIK DANAŠNJE USLUŽNOSTNE INFORMACIJSKE DEJAVNOSTI, KI JE V BRITANIJI ZNANA POD IMENOM VIEWDATA (GLEDANJE BESEDIL OZIROMA INFORMACIJ). OTVORITEV NOVEGA RAČUNALNIŠKEGA CENTRA V ANGLIJI, KI ODDAJA BREZŽIČNO INFORMACIJE ŽE NEKAJ LET, JE POTRDILA UPRAVIČENOST SISTEMA PRESTEL. ŠE TRIJE KONKURENČNI SISTEMI ZA VIDEOTEKSTNE USLUGE SO SE POJAVILI V ZAPADNIH DRŽAVAH (IMENA KOT SO TELETEKST, VIDEOTEKST, VIEWDATA, PRESTEL ITN. PREDSTAVLJAJO RAZLIČNE SISTEME ZA PRENOS IN PRIKAZOVANJE VIDNEGA TEKSTA OZIROMA INFORMACIJE).

SAM FEDIDA SE JE PO USPEŠNI RAZISKOVALNI KARIERI V LETU 1968 ZAPOSLIL PRI BRITANSKI POŠTI. TU JE BILA NJEGOVA NALOGA ZGOLJ V UPORABI RAČUNALNIKOV V TELEKOMUNIKACIJAH. SAM PRAVI: "V TISTEM ČASU SEM BIL EDINI ČLAN MOJEGA ODDELKA IN TO JE BILO LEPO, KER SEM IMEL ČAS ZA RAZMIŠLJANJE." TA ČAS JE FEDIDA DOBRO IZRABIL IN JE ŽE V NEKAJ MESECIH OBLIKOVAL CELOTEN KONCEPT IN STRATEGIJO RAZVOJA SISTEMA ZA GLEDANJE TEKSTOV.

FEDIDA JE KMALU OGOTOVIL, DA MORAJO BITI UPORABA RAČUNALNIŠKIH BAZ PODATKOV, DOPOLNJEVANJE BAZ Z NOVIMI PODATKI TER DOSTOPNOST V BAZE POSAMEZNIKOM ALI PODJETJEM ZELO RAZNOVRSTNE. PRI TEM MORAJO OSTATI STROŠKI UPORABE NIZKI, UPORABA MORA BITI ENOSTAVNA IN DOSTOPNA VSAKOMUR, TUDI TISTEMU, KI NIMA NOBENIH PREDSTAV O DELOVANJU ZAPLETENEGA RAČUNALNIŠKEGA SISTEMA.

SEVEDA NI BILO TEŽAVNO DOBITI RAČUNALNIK IN NEKAJ STRANI INFORMACIJE NA ZASLONU. PROBLEMI SO SE POJAVILI PRI NAČRTOVANJU SISTEMA, KI BI BIL UPORABEN ZA CELOTNO PODROČJE VELIKE BRITANIJE. NA TEJ RAVNI SO BILE POTREBNE INŽENIRSKE ODLOČITVE, KJE VSTAVITI RAČUNALNIKE, KAKŠNO PROGRAMSKO OPREMO RAZVIJATI ZA VZDRŽEVANJE IN KNJIGOVODSTVO SISTEMA, KI BO UPOŠTEVALA OBRAČUNAVANJE USLUG, DOBAVO ZAHTEVANIH INFORMACIJ, MOŽNOSTI IZBIRE IN UREJEVANJA INFORMACIJ TER KONČNO POSREDOVANJE INFORMACIJ NAROČNIKOM.

FEDIDA SE JE ODLOČIL ZA DECENTRALIZIRANI SISTEM, V KATEREM SO RAČUNALNIŠKE BAZE PODATKOV PONOVLJENE V DOLOČENIH TOČKAH OMREŽJA PO VSEJ DRŽAVI. TAK KONCEPT NAJ BI OMOGOČIL, DA DOBI NAROČNIK SVOJO USLUGO Z LOKALNIM TELEFONSKIM POZIVOM, TOREJ PO NAJNIŽJI MOŽNI CENI.

FEDIDOVA INOVACIJA ZA GLEDANJE TEKSTOV JE TIPIČEN PRIMER PRAVEGA STROKOVNJAKA NA PRAVEM MESTU V PRAVEM TRENUTKU. FEDIDA JE BIL PRED TEM RAZISKOVALNI INŽENIR IN TU JE DOSEGEL STOPNJO MAGISTRA RAČUNALNIŠKIH ZNANOSTI. KASNEJE JE BIL NA POLOŽAJU DIREKTORJA ZA PODROČJE TELEKOMUNIKACIJ. NA BRITANSKO POŠTO JE PRIŠEL S TEMELJITIMI IZKUŠNJAMI TER S PRIMERNIM STROKOVNIM OZADJEM.

A.P.ŽELEZNIKAR

O NOVIH MIKROPROCESORJIH

GENERACIJA MIKROPROCESORJEV, KI UPORABLJA 16-BITNE ENOTE, JE SEVEDA MOČNEJŠA OD STARE 8-BITNE GENERACIJE. POVEČALI SO SE NASLOVNI PROSTOR IN TAKTNE HITROSTI, SPREMEMBA ARHITEKTURE TEH PROCESORJEV OMOGOČA OPERACIJE NAD BITI, ZLOGI, 16- IN 32-BITNIMI BESEDAMI IN NAD PODATKOVNIMI NIZI. NEKATERI NOVEJŠI PROIZVODI PA SO LE HIBRIDNI PROCESORJI, KOT JE NPR. MOTOROLIN 6809, KI DELUJE NAD DVOJNIMI BESEDAMI TUDI NA SVOJEM ZUNANJEM VODILU.

SODOBNI MIKROPROCESORJI UPORABLJAJO PREDHODNO JEMANJE UKAZOV IZ POMNILNIKA IN PARALELNO POVEZOVANJE. NJIHOVE KRMILNE ENOTE SO VEČKRAT IZDATNO MIKROKODIRANE (8086 IN 68000). MC 68000 UPORABLJA TAKO VERTIKALNE IN HORIZONTALNE MIKROUKAZE, KI JIH MOTOROLA IMENUJE NANOUKAZI. S TAKO TEHNIKO JE MOGOČE V PRIHODNOSTI IZVAJATI MODIFIKACIJE IN MENJATI UKAZE. TAKO BO MOGOČE K OBSTOJEČEMU PROCESORJU DODATI ARITMETIKO S POMIČNO VEJICO IN OPERACIJE NAD NIZI.

V NASPROTJU Z MC 68000 PA JE Z-8000 NAREJEN Z NAKLJUČNO LOGIKO TER VSEBUJE MANJ MIKROKODA KOT RAČUNALNIK V ENEM VEZJU Z-8. NATIONALOV INS 8070 PA PRAKTIČNO NIMA NAKLJUČNE LOGIKE.

NOVI PROCESORJI SO RAZDELJENI NA DELOVNE ENOTE. I 8086 IMA IZVRŠILNO IN POVEZOVALNO (NA VODILO) ENOTO. UPORABLJA UKAZNO VRSTO Z VNAPREJŠNJIM JEMANJEM UKAZOV IZ POMNILNIKA. IZVRŠILNA ENOTA JEMLJE UKAZE IZ VRSTE, TAKO DA NI VEČ ČAKALNIH PERIOD. Z-8000 JE MOGOČE KONFIGURIRATI NA DVA NAČINA: Z MATERIALNO IN PROGRAMSKO OPREMO. DVE RAZLIČICI STA NA VOLJO: 48-NOŽIČNI Z-8001 ZA SEGMENTIRANI POMNILNIK Z SOUPORABO POMNILNIŠKEGA UPRAVLJALNEGA VEZJA Z-8010 TER 40-NOŽIČNI Z-8002 ZA DIREKTNO NASLAVLJANJE ENEGA SAMEGA POMNILNIŠKEGA SEGMENTA ZA 64K ZLOGOV. OBE RAZLIČICI LAHKO DELUJETA V SISTEMSKEM IN NORMALNEM NAČINU S SVOJIM SKLADNIM KAZALCEM. MC 68000 IMA ŠE T.I. NADZORNIŠKI NAČIN, KI SE GA IZBERE Z ENIM BITOM V STATUSNEM REGISTRU. V TEM NAČINU IMA PROGRAMER NA RAZPOLAGO POSEBNE UKAZE, KOT JE USTAVITEV IN NOVA NASTAVITEV POMNILNIŠKE UPRAVLJALNE ENOTE.

TUDI NATIONAL SEMICONDUCTORS PRIPRAVLJA NOVO PROCESORSKO DRUŽINO NS 16000, KI BO IMELA VEZ RAZLIČIC CENTRALNIH PROCESNIH ENOT (PODOBNO KOT ZILOG). 16008 BO IMEL 8-BITNO MULTIPLEKSIRANO VODILO, 16016 BO IMEL 16-BITNO VODILO IN 16032 BO IMEL 32-BITNO ARITMETIČNO IN LOGIČNO ENOTO TER REGISTRE. NEKATERE CPE BODO IMELE TUDI UKAZNE NADMNOŽICE PROCESORJEV 8080A IN Z-80. TI PROCESORJI BODO IMELI ŠE T.I. PODPROCESORJE ZA ARITMETIKO S PLAVAJOČO VEJICO, ZA UPRAVLJANJE POMNILNIKA ITD.

NOVI PROCESORJI BODO DELOVALI V RAZLIČNIH OKOLJIH: V NOVIH OPERACIJSKIH SISTEMIH, ZA VEČ UPORABNIKOV HKRATI, Z VISOKIMI JEZIKI.

VEČJA SPOSOBNOST MIKROPROCESORJEV ZA PROGRAMIRANJE V VISOKIH JEZIKIH SE DOSEŽE S T.I. REGULARNO (KONSISTENTNO, ORTOGONALNO) ARHITEKTURO IN Z VELIKIM ŠTEVILOM ADRESIRNIH NAČINOV, Z UKAZNO RAZNOVRSTNOSTJO IN Z DODATNIMI PERIFERNIMI VEZJI. UKAZNA ZALOGA PROCESORJA 8086 ŠE NI PRILAGOJENA TEMU CILJU, KER IMA PREVEČ SPECIFIČNE REGISTRE, VEZANE NA DOLOČENE FUNKCIJE. TAK NAČIN POVZROČA TEŽAVE PRI GENERIRANJU KODA (PRI PREVAJANJU), KER JE POTREBNO PREVEČ PREMATAVATI PODATKE IZ ENEGA REGISTRA V DRUGEGA. ZAMISEL ORTOGONALNOSTI JE BILA MNOGO BOLJ UPOŠTEVANA PRI PROCESORJIH Z-8000 IN 68000, SAJ STA SE KASNEJE VKLJUČILA V IGRO. Z-8000 IMA 16 POVSEM SPLOŠNIH REGISTROV, KI SO UPORABNI KOT AKUMULATORJI ZA ZLOGE,

16-BITNE BESEDE IN 32-BITNE OPERACIJE. TEMU PRAVIMO REGULARNA REGISTRSKA ZBIRKA.

ZBIRKA SPLOŠNIH REGISTROV, RAZLIČNIH NASLAVLJALNIH NAČINOV IN UKAZNIH TIPOV OMOGOČA PREVAJALNIKU IN PROGRAMERJU UČINKOVITO GENERIRANJE KODA. MC 68000 IMA NPR. POSTINKREMENTNI IN PREDEKREMENTNI NASLAVLJALNI NAČIN, S KATERIM SE SEŠTEVA ALI ODŠTEVA ZLOGE, BESEDE IN DOLGE BESEDE Z ALI OD 8-, 16- ALI 32-BITNIH REGISTROV S PROŽNO SKLADNO IN VRSTNO MANIPULACIJO.

ZA UKAZNO ZALOGO JE MOČ POSAMEZNIH UKAZOV POMEMBNEJŠA OD ŠTEVILA UKAZOV. MC 68000 IMA DVA UKAZA (LINK IN UNLINK), KI DODELJUJETA OZIROMA ODVZEMATA LOKALNI POMNILNIK ZA SKLAD, TAKO DA JE MOGOČE IZVAJATI PROCEDURE. PROCEDURE SO TU OSNOVNI BLOKI KODA. TA PROCESO IMA ŠE DRUGE UKAZE ZA SHRANJEVANJE IN VEČKRATNI KLIC REGISTROV PRED PROCEDURNIM POZIVOM.

JEZIK PASCAL JE V POSLEDNJEM LETU DOSEGEL VIŠEK SVOJE POPULARNOSTI. PROCESORJA 68000 IN 16000 STA BILA NAČRTOVANA Z UPOŠTEVANJEM TEGA JEZIKA IN PASCALSKI MIKRORAČUNALNIK (WESTERN DIGITAL) JE BIL RAZVIT ZA IZKLJUČNO IZVAJANJE PASCALSKIH PROGRAMOV. PODOBEN MIKRORAČUNALNIK JE TUDI MK-16 (MIKROS SYSTEMS CORP.), KI UPORABLJA REZINASTO STRUKTURO TER IZVAJA P-KOD, TJ. INTERPRETIVNI JEZIK, KI GA GENERIRAJO NEKATERI PASCALSKI PREVAJALNIKI.

RAZEN VISOKOH PROGRAMIRNIH JEZIKOV JE ZA NOVE PROCESORJE POMEMBNA TUDI HITRA ARITMETIKA IN HITRA LOGIKA. AMD IMA DVE TAKI VEZJI (9511 IN 9512), KI JIH IZDELUJE TUDI INTEL (DRUGI VIR). INTEL IMA TUDI SVOJ MATEMATIČNI PROCESOR 8087 ZA DRUŽINO PROCESORJA 8086 IN PODOBNI PODPROCESOR 80 IZDELOVAL TUDI NATIONAL ZA SVOJO DRUŽINO 16000.

VHODNA/IZHODNA VEZJA SO BILA V POSLEDNJIH LETIH PRIZORIŠČE PRAVEGA TEKMOVANJA MED RAZLIČNIMI PROIZVAJALCI. I 8089 V/I PROCESOR JE TAK NOV DOSEŽEK. TO JE VISOK PROCESOR ZA DMA (DIREKTEN POMNILNIŠKI DOSTOP), KI IMA LASTNOSTI PROCESORJA ZA PRENOS PROTOKOLOV (NPR. KANALNI PROCESOR IBM). IMA DVA NEODVISNA KANALA TER SI GA LAHKO (ČASOVNO) DELIJO POČASNEJŠE PERIFERNE ENOTE. TAKTNA HITROST PROCESORJA 8089 JE 5 MHZ, S KATERO SE DOSEŽE PRENOSNO RAZMERJE 1,25 MEGAZLOGOV NA SEKUNDO. DA BI OLAJŠAL RAZVOJ NAPRAV S TEM PROCESORJEM JE INTEL IZDAL ZBIRNIK Z OZNAKO ASM 89.

REALNI SVET IMA TUDI ANALOGNE SIGNALE. INTEL 2920 JE ANALOGNI MIKRORAČUNALNIK IN S2811 JE PERIFERNO VEZJE ZA PROCESIRANJE SIGNALOV (AMI). VEZJI 2811 IN 2920 STA 28-NOŽIČNI IN STA NAMENJENI DELOVANJU V ANALOGNEM PROSTORU. VEZJE 2920 VSEBUJE A/D IN D/A PRETVORNIKE, VEZJE 2811 PA IMA ŠE 16-BITNI MULTIPLIKATOR S HITROSTJO MNOŽENJA 300 NANOSEKUND. ZA RAZVOJ PROGRAMOV IMA INTEL PROGRAMSKI SIMULATOR, AMI PA IMA EMULATOR V REALNEM ČASU.

POSEBNA ZANIMIVOST JE TUDI MIKROPROCESOR MAC-4 (BELL LABORATORIES). TA PROCESOR IMA POSEBNE ZMOGLJIVOSTI ZA MANIPULACIJO Z BITI, JE NAREJEN V C-MOS TEHNOLOGIJI IN PORABI LE 200 MILIVATOV. S POSEBNIM UKAZOM GA JE MOGOČE IZKLJUČITI IZ SISTEMA TER GA POSTAVITI V STANJE NIZKE PORABE (NEKAJ MIKROVATOV). TAKO LAHKO KONDENZATOR DO 2 MIKROFARADA DRŽI SHRANJENO TELEFONSKO ŠTEVILKO V VEZJU TUDI VEČ DNI.

SPECIALIZIRANI IN PERIFERNI PROCESORJI POSTAJAJO TAKO ZAPLETENI IN ZMOGLJIVI, DA DOSTIKRAT PRESEGAJO CELO SAME MIKROPROCESORJE. TAKO POSTAJA RAZLIKA MED POSEBNIMI IN MIKROPROCESORJI VSE MANJŠA. KER POTREBUJEJO TAKI PROCESORJI TUDI ŠE VRSTO REGISTROV, POMNILNIK IN DRUGE MATERIALNE PRIPOMOČKE, JIH JE NA TAK PROCESOR NAJLAŽJE POVEZATI S POMOČJO MIKROPROCESORJA.

A.P.ŽELEZNIKAR

N. WIRTH, OČE PASCALA

KDO NE POZNA WIRTHA, BI SE LAHKO VPRAŠALI. ŠTUDENTJE RAČUNALNIŠKIH SMERI PO VSEM SVETU SE 'MORAJO' UČITI NJEGOV JEZIK PASCAL, KI NAJ BI PREDVSEM NAVAJAL PROGRAMERJE K OBLIKOVANJU DOBRIH PROGRAMOV. SAM WIRTH PA PRAVI: "NAŠ DOPRINOS K RAZVOJU RAČUNALNIŠKIH JEZIKOV JE BIL NEZNATEN V PRIMERJAVI S TEM, KAR JE OSTALO, DA SE ŠE NAREDI." TAKO GLEDA NIKLAUS WIRTH V BISTVU NAZAJ NA SVOJE DELO, KO JE RAZVIJAL JEZIK PASCAL IN DRUGE VISOKE PROGRAMIRNE JEZIKE.

WIRTH VODI DANES ODDELEK ZA RAČUNALNIŠKE ZNANOSTI PRI SVIČARSKEM ZVEZNEM INŠTITUTU ZA TEHNOLIGIJO V ZUERICHU. PRAVI, DA JE ZADOVOLJEN, KER JE METODOLOGIJA, ZA KATERO SI JE DOLGO PRIZADEVAL, BILA RAZPOZNANA KOT DRAGOCENO ORODJE V PROCESIRANJU PODATKOV. PRI TEM NE GRE TOLIKO ZA JEZIK PASCAL, TEMVEČ PREDVSEM ZA DISCIPLINIRAN IN STRUKTURIRAN PRISTOP PRI OBLIKOVANJU PROGRAMOV. PASCAL NAJ BI PRIBLIŽAL PROGRAMERJA TEMU CILJU OZIROMA NAČINU DELA.

WIRTHOVO DELO NA PASCALU SE JE ZAČELO V LETU 1965, KO JE V OKVIRU POSEBNEGA DELOVNEGA ODBORA V IFIP SODELOVAL PRI RAZVOJU NASLEDNIKA JEZIKA ALGOL 60. V LETU 1966 JE SODELOVAL PRI IMPLEMENTACIJI JEZIKA ALGOL W NA STANFORDSKI UNIVERZI, KJER JE BIL PROFESOR NA NOVOUSTANOVLJENEM ODDELKU ZA RAČUNALNIŠKE ZNANOSTI.

LETA 1967 SE JE WIRTH VRNIL V ŠVICO, KJER SE JE DOKONČNO POSLOVIL OD KOMPROMISARSKEGA IZHODIŠČA IFIPOVEGA ODBORA TER JE IZ JEZIKA ALGOL W RAZVIL NOV JEZIK. TA JEZIK JE DOBIL IME PASCAL PO FRANCOSKEMU MATEMATIKU IN FIZIKU 17. STOLETJA BLAISU PASCALU, KI JE V LETU 1642 ZGRADIL RAČUNALNI STROJ ZA POMOČ PRI ZBIRANJU DAVKOV.

WIRTH POUDARJA RAZLIKO MED STROJEM IN JEZIKOM: "BISTVO JEZIKA JE V TEM, DA OMOGOČI UPORABNIKU RAČUNALNIKA OBLIKOVANJE NJEGOVIH PROGRAMOV NA VIŠJI ABSTRAKTNI RAVNINI, DA SE TAKO BOLJE REŠI DANI PROBLEM. JEZIK SAM NIMA NAMENA POSPEŠEVATI RAZVOJ RAČUNALNIKOV." TA IZJAVA SEVEDA VSE TEŽJE KLUBUJE SODOBNEMU RAZVOJU, SAJ POZNAMO DANES ŽE PASCALSKE PROCESORJE (NPR. WESTERN DIGITAL).

WIRTH JE KONČAL SVOJE DELO NA PASCALU V LETU 1974 IN POSVETIL SVOJO SKRB JEZIKU MODULA, KI NAJ BI BIL JEZIK ZA PROGRAMIRANJE POSEBNIH (NPR. OPERACIJSKIH) SISTEMOV IN MALIH RAČUNALNIKOV (NPR. MIKRORAČUNALNIKOV). ČEPRAV MODULA NI NEPOSREDEN NASLEDNIK PASCALA, UPORABLJA PASCALSKE ELEMENTE. RAZEN NAVADNE STRUKTURE BLOKA IMA MODULA ŠE 'MODULSKO' STRUKTURO. MODUL IMA T.I. PREGLED IMEN, KI PROGRAMERJU OMOGOČA SKRIVANJE PODATKOV PRED DRUGIMI PROGRAMSKIMI SEGMENTI.

PRVA RAZLIČICA JEZIKA Z IMENOM MODULA I ŠE NI BILA POPOLN JEZIK ZA PROGRAMIRANJE SISTEMOV; TO UGOTAVLJA SAM WIRTH. PAČ PA JE MODULA II TISTI JEZIK, KI JE NAMENJEN OSEBNEMU RAČUNALNIKU, KATEREGA RAZVIJAJO SEDAJ V ZUERIŠKEM INŠTITUTU. NAJVEČJA PREDNOST MODULE II NAD PASCALOM JE REALIZACIJA KONCEPTA MODULA, KI BO POSTAL V PRIHODNOSTI OSNOVNI POGOJ ZA PROGRAMIRANJE IN POVEZOVANJE PROGRAMOV. TA UGOTOVITEV SEVEDA NI NOVA, SAJ POZNAMO PODOBEN

NAČIN PROGRAMIRANJA ŽE V ZBIRNIH JEZIKIH, KATERIH ZBIRNIKI DOPUŠČAJO DEFINICIJO MODULA OZIROMA DEFINICIJO GLOBALNIH SPREMENLJIVK, KI SO SESTAVNI DEL PREMESTLJIVEGA RELATIVNEGA STROJNEGA KODA.

WIRTH POVDARJA, DA JEZIK NE SME POSTATI STANDARD, ČE NJEGOVA UPORABNOST NI BILA DOKAZANA V VEČLETNI PRAKTIČNI RABI. ZA PASCAL MENI, DA JE ŠELE NJEGOV PRIROČNIK IZ LETA 1975 MOGOČE VZETI KOT STANDARDNO DEFINICIJO.

A.P.ŽELEZNIKAR

INFORMACIJSKI SISTEMI ZA DOM

PODJETJE AMERICAN TELEPHONE AND TELEGRAPH (AT&T) JE ZACELO PREIZKUŠATI SVOJ INFORMACIJSKI SISTEM ZA DOM, KI JE PODOBEN SISTEMU VIEWDATA V VELIKI BRITANIJI. MED RAZLIČNIMI SISTEMI, KI SE PREIZKUŠAJO JE T.I. KNIGHT-RIDDERJEV SISTEM (RAZVIL GA JE MC DONNELL DOUGLAS) IN SISTEM PODJETJA BELL LABORATORIES. V OBEH SISTEMIH SE BODO ODDAJALE INFORMACIJE S PODROČJA NOVIC (POROČIL), ŠPORTNIH REZULTATOV, VREMENA IN JAVNIH INFORMACIJ, UPORABNIK BO POKLICAL POSEBNO TELEFONSKO ŠTEVILKO IN NATO VTIPKAL ŠE POSEBEN KLJUČ S TASTATURO, INFORMACIJO PA BO DOBIL V ZVOČNI OBLIKI.

A.P.ŽELEZNIKAR

RAČUNALNISKI CENTER ZA DOM

JAPONSKO PODJETJE SHARP ELECTRONICS JE PRIKAZALO PRENOSNO ENOTO, KI IMA VELIKOST TIPIČNE PRENOSNE STEREO NAPRAVE IN VSEBUJE TOLE: TELEVIZIJSKI SPREJEMNIK Z ZASLONOM 11,5 CM, RADIJSKI SPREJEMNIK ZA AM IN FM, STEREOKASETNO NAPRAVO, DIGITALNO URO, KALKULATOR IN OSEBNI RAČUNALNIK. RAČUNALNIK IMA ZLOŽLJIVO TASTATURO Z 48 TIPKAMI. VIDOEZASLON TELEVIZORJA SE UPORABLJA ZA PRIKAZOVANJE RAČUNALNISKIH SPOROČIL IN PODATKOV, KASETNA NAPRAVA PA ZA SHRANJEVANJE PODATKOV IN PROGRAMOV. RAČUNALNIK UPORABLJA JEZIK BASIC, IMA TUDI GRAFIČNE MOŽNOSTI TER GA JE MOČ ŠIRITI.

A.P.ŽELEZNIKAR

O PORAZDELJENIH INFORMACIJSKIH SISTEMIH

VEČKRAT GOVORIMO O T.I. PORAZDELJENIH ALI DISTRIBUIRANIH INFORMACIJSKIH SISTEMIH (PIS), PRI ČEMER JE BESEDA 'DISTRIBUIRAN' RAZUMLJENA UPORABNIŠKO, TJ. V ODVISNOSTI OD KONKRETNE UPORABE. PORAZDELJENO POMENI, DA IMAMO LOGIČNO INTEGRIRANI INFORMACIJSKI SISTEM, KI PA IMA FIZIČNO PORAZDELJENE PODATKE PREK DVEH ALI VEČ RAČUNALNISKIH ENOT. UPORABNIK IMA V TAKEM SISTEMU DOSTOP DO PODATKOV IZ VSAKE 'UDELEŽENE' RAČUNALNISKE ENOTE PODOBNO, KOT BI GA IMEL V CENTRALIZIRANEM INFORMACIJSKEM SISTEMU. V ORGANIZACIJSKEM POGLEDU POMENI PIS TUDI, DA SE NAHAJA PROCESIRNA MOČ TAM, KJER JE POTREBNA.

KER JE V PIS TUDI INFORMACIJA PORAZDELJENA, MORA TAK SISTEM RAZPOLAGATI S KOMUNIKACIJSKIM SISTEMOM ZA PRENOS PODATKOV, S SEZNAMI, KATALOGI IN IMENIKI ZA IDENTIFIKACIJO IN LOKALIZACIJO PODATKOV TER S PODATKOVNIM UPRAVLJALNIM SISTEMOM ZA SINHRONIZACIJO, INTEGRITETO, SKLADNOSTJO, PRIVATNOSTJO, VARNOSTJO IN NAMESTITEV PODATKOV. OGLEJMO SI ŠE ENO IZMED DEFINICIJ PORAZDELJENEGA INFORMACIJSKEGA SISTEMA (PIS):

VOZLIŠČE NAJ BO EN PROCESOR ALI VEČ PROCESORJEV S SKUPNIM OPERACIJSKIM SISTEMOM. PORAZDELJENI SISTEM JE KOMPLEKSNA POVEZAVA SORODNIH ALI RAZNOVRSTNIH VOZLIŠČ, KI

IZMENJUJEJO PODATKE,
SI DELIJO VIRE IN
IZVAJAJO RAZNOVRSTNE, DOBRO OPREDELJENE IN MEDSEBOJNO ODVISNE PROGRAMSKE KOMPONENTE (PROGRAME) V ENEM ALI V VEČ VOZLIŠČIH.

BISTVENA RAZLIKA MED PORAZDELJENIM IN NEPORAZDELJENIM SISTEMOM JE V PORAZDELITVI UPRAVLJANJA V MREŽI (V PORAZDELJENEM SISTEMU) IN V NEPOMNILNIŠKIH POVEZAVAH MED PROGRAMSKIMI KOMPONENTAMI. PORAZDELITEV UPRAVLJANJA POMENI DODELJEVANJE, ISKANJE IN DOSTOP DO MREŽNIH VIROV TER SINHRONIZACIJO USLUG, KOT STA OBNAVLJANJE PODATKOV (AŽURIRANJE) IN VZDRŽEVANJE. NEPOMNILNIŠKE KOMUNIKACIJE PREK KANALOV IN LINIJ PA ZAHTEVAJO ZAPLETENE PODATKOVNE PROTOKOLE ZA ZANESLJIV PRENOS.

A.P.ŽELEZNIKAR

ANALOGNI MIKRORAČUNALNIKI

INTELOVA NOVA INTEGRIRANA KOMPONENTA JE ANALOGNI MIKRORAČUNALNIK 2920, KI IMA NOTRANJO A/D IN D/A PRETVORBO, SPREJEMA STIMULE IZ OKOLICE, JIH DIGITALNO PROCESIRA V REALNEM ČASU TER REZULTATE PREVEDE V ANALOGNO OBLIKO. NOVI PROCESOR JE ZLASTI PRIMEREN ZA UPORABO V TELEKOMUNIKACIJAH, IN SICER V TELEFONSKIH SISTEMIH.

INTEL NAPOVEDUJE NOVI VAL ANALOGNEGA RAČUNALNIŠTVA, KI BO NASTOPIL S PRODAJO TEGA PROCESORJA, SAJ JE S TEM PROCESORJEM KONČNO NASTOPIL TRENUTEK DIGITALNE OBDELAVE ANALOGNIH SIGNALOV. TA NOVA TEHNOLOGIJA BO TAKO ŽE V KRATKEM DOSTOPNA NAČRTOVALCEM IN INŽENIRJEM ZA NAJŠIRŠO UPORABO V PROCESNI TEHNIKI.

ANALOGNO RAČUNALNIŠTVO JE DANES ŠELE V DOBI SVOJEGA OTROŠTVA IN POJAVITI SE MORAJO ŠE DRUGI SPECIALIZIRANI PROCESORJI ANALOGNIH SIGNALOV, KOT SO NPR. PROGRAMIRLJIVI FILTRI, MEŠALNIKI, MODEMI, TONSKI SPREJEMNIKI, FREKVENČNI GENERATORJI, VISOKOINTEGRIRANI VMESNIKI ZA AKTUATORJE ITN., KI SO ZA PROCESNO REGULACIJO, KOMUNIKACIJO IN VISOKO AVTOMATIZACIJO OSNOVNEGA POMENA. PROCESOR 2920 VSEBUJE DVOSMERNE PRETVORNIKE, DIGITALNI PROCESOR TER MIKRORAČUNALNIK S POMNILNIKOM RAM IN EPROM ZA UPORABNIŠKE UKAZE.

A.P.ŽELEZNIKAR

16-BITNI A/D PRETVORNIK

INTERSIL JE DAL V PRODAJO SVOJ NOVI 16-BITNI A/D PRETVORNIK ZA CENO US$ 29,00. LASTNOSTI TEGA PRETVORNIKA SO:

- 16-BITNI, BINARNI: 1 DELEC V 65536;
- 1 DELEC NELINEARNOSTI V CELOTNEM OBMOČJU;
- LOČLJIVOST MANJŠA OD 10 MIKROVOLTOV NA KORAK;
- NAPETOSTNI ŠUM JE CCA. 2 MIKROVOLTA;
- DRIFT NIČLE JE 0,5 MIKROVOLT/STOPINJA CELZIJA IN
- DIREKTNA PRIKLJUČITEV NA UART ALI MIKROPROCESOR.

TA NOVI PRETVORNIK JE SESTAVLJEN IZ DVEH VEZIJ IN IMA OZNAKO ICL 7104/8068.

A.P.ŽELEZNIKAR

LITERATURA IN SREČANJA

30 jan.-1 feb., Monterey, California, ZDA

INTERNATIONAL SYMPOSIUM ON MICROCOMPUTERS AND THEIR APPLICATIONS

Organizator: ISMM
Informacije: Secretary MIMI-80 (Monterey) Box 2481 Anaheim, Ca 92804, ZDA

4-6 februar, Bombay, India

INTERNATIONAL SYMPOSIUM ON DATA COMMUNICATION AND COMPUTER NETWORKS

Organizator: Computer Society of India, IFIP TC6
Informacije: Networks 80, c/o CMC, World Trade Center, Cuffe Parade, Bombay 400 005, India

9-13 februar, Mailand, Italija

INTEL - MEDNARODNA RAZSTAVA ELEKTRONIKE

12-14 februar, Kansas City, ZDA

ACM COMPUTER SCIENCE CONFERENCE

Informacije: Conf. chm.Earl J.Schweppe, Dept. of Computer Science, University of Kansas, Lawrence, KS 66044

13-15 februar, San Francisco, ZDA

IEEE INTERNATIONAL SOLID STATE CIRCUITS CONFERENCE

Organizator: IEEE, Solid State Circuits Council, IEEE San Francisco Sect., University of Pennsylvania, Contact
Informacije: Winner, 301 Almeria Ave., Box 343788, Coral Gables, FL 33134; 305 466-8193

14-15 februar, Kansas City, ZDA

ACM SIGCSE TECHNICAL SYMPOSIUM ON COMPUTER SCIENCE EDUCATION

Organizator: ACM, SIGCSE
Informacije: William Bulgren, Dept. of Computer Science, The University of Kansas, Lawrence, KS 66044; 913 864-4482

20-24 februar, Dortmund, ZRN

RAZSTAVA HOBBYTRONIC 80

25-28 februar, San Francisco, ZDA

COMPCON 80 SPRING

Organizator: IEEE-CS
Informacije: COMPCON 80 Spring, Box 639, Silver Spring, MD 20901

25-29 februar, Birminghaw , Velika Britanija

IEA IN ELECTREX, MEDNARODNA RAZSTAVA ZA INSTRUMENTACIJO, ELEKTRONIKO IN AVTOMATIKO

11 februar, München, ZRN

SEMINAR TVRTKE ICS: MIKROPROCESORSKA IN MIKRORAČUNALNIŠKA

Informacije: ICS, München, ZRN

13-15 februar, Esslingen, ZRN

SEMINAR: OBDELAVO DVODIMENZIONALNIH PODATKOV ZA PODROČJE MEDICINE

Organizator: Tehnična akademija, Esslingen, ZRN
Informacija: Tehnična akademija, Esslingen, ZRN

17-22 februar, Atlanta, ZDA

SVETOVNI SEJEM ZA IZMENJAVO TEHNOLOGIJE

3-7 marec, Florida Atlantic University Boca Raton, ZDA

ELEVENTH SOUTHEASTERN CONFERENCE ON COMBINATORIES, GRAPH THEORY, AND COMPUTING

Organizator: Florida Atlantic University Conf.
Informacije: Frederick Hoffman, Dept. of Mathematics, Florida Atlantic University, Boca Raton, FL 33431; 305 395-5100 x2756

3-5 marec, Georgia, Atlanta, ZDA

NCC OFFICE AUTOMATION CONFERENCE

Organizator: AFIPS, ACM, DPMA, IEE-CS, SCS
Informacije: Jerry Chriffriller, AFIPS, 210 Summit Ave., Montvale, NJ 07645

4-6 marec Zürich, Švica

1980 INTERNATIONAL ZURICH SEMINAR ON DIGITAL COMMUNICATIONS

Organizator: IEEE, Switzerland Chapter on Digital Communication Systems in cooperation IEEE, Switzerland Section, ACM, EUREL, SEV, NTG, AEI, IEE, ETHZ
Informacije: Secretariat 1980 International Zürich Seminar, D.Hug, Dept. ENF, BBC Brown, Boweri and Co. Ltd., CH-5401 Baden, Switzerland

9-11 marec, Santa Clara, California, ZDA

CONFERENCE ON APPLICATION DEVELOPMENT SYSTEMS

Organizator: BDP, UCLA Graduate School of Management, IBM San Jose Research Laboratory, M.I.T. Sloan School of Management
Informacije: Eric D.Carlson, IBM Research Div. K52/282, 5600 Cottle Road, San Jose, Ca 95193; 408 256-6431 or 7582

11-14 marec, Pacific Grove, California, ZDA

5th WORKSHOP ON COMPUTER ARCHITECTURE FOR NON-NUMERIC PROCESSING

Organizator: ACM Sigarch, Sigir and Sigmod
Informacije: W.F.King and S.P.Ghosh, IBM Research Laboratory, 5600 Cottle Road, San Jose, CA 95193; 408 256-6746 and 7649

12-14 marec, Kiel, ZRN

GI-NTG CONFERENCE ON COMPUTER ARCHITECTURE AND OPERATING SYSTEMS

Organizator: GI. NTG.
Informacije: G.Mimmermann, Institut für Informatik und Prakt.Math. Universitäet Kiel, D-2300 Kiel, Germany

12-14 marec, Versailles, Francija

INTERNATIONAL SYMPOSIUM ON DISTRIBUTED DATA BASES

Organizator: IRIA
Informacije: IRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78150 Le Chesnay, France

17-19 marec, Philadelphia, ZDA

IECI 80, SIXTH ANNUAL CONFERENCE AND EXHIBIT ON INDUSTRIAL AND CONTROL APPLICATIONS OF MICROPROCESSORS

Organizator: Industrial Electronic and Control Instrumentation Society, IEEE
Informacije: Paul M.Russo, RCA Laboratories, Princeton, NJ 0540; 609 452-2700

17-21 marec, Dunaj, Avstrija

6TH INTERNATIONAL CONGRESS ON DATA PROCESING IN EUROPE

Organizator: Arbeitgemeinschaft für Datenverbeitung
Informacije: Sekretariat, 6 Internationaler Kongress Datenverbeitung im Europäischen Raum, c/o Interconvention, P.O.Box 35, A-1095 Wien, Austria

18-20 marec, Seattle, Washington, ZDA

ELECTRIC POWER PROBLEMS: THE MATHEMATICAL CHALLENGE

Informacije: Albert M.Erisman, Boeing Computer Services Co., 565 Andover Park West, M/S 9C-01, Tukwila, WA 98188

19-21 marec, Tampa, ZDA

13th ANNUAL SIMULATION SYMPOSIUM

Organizator: ACM, SIGSIM, IEEE-CS, SCS
Informacije: Harvey Fisher, Alcan Products, Box 511, Warren, OH 44482; 216 841-3416

24-26 marec, Tallahasse, Fla., ZDA

ACM SE 80, EIGHTEENTH ANNUAL CONFERENCE

Organizator: ACM, South Region
Informacije: E.P.Miles, Jr., Dept. of Mathematics, Florida State University, Tallahaussee, FL 32306; 904 644-1587

24-28 marec, Jahorina, Jugoslavija

IV. BOSANSKOHERCEGOVAČKI SIMPOZIJUM IZ INFORMATIKE "JAHORINA 80"

Organizator: Elektrotehnički fakultet Sarajevo
Informacije: Elektrotehnički fakultet Sarajevo, Odsjek za informatiku, za simpozijum, 71000 Sarajevo, Toplička cesta bb, telef. 071 521-677/132

27-28 marec, ST. Louis, Mo., ZDA

SEVENTH ANNUAL ACM SIGUCC COMPUTER CENTER MANAGEMENT SYMPOSIUM

Organizator: ACM, SIGNCC
Informacije: Ralph E.Lee, Director of Computer Center, University of Missouri, Rolla, Rolla, MO 65401; 314 341-4841

28 marec-3 april, Cambridge, Velika Britanija

SIXTH INTERNATIONAL ALLC SYMPOSIUM ON COMPUTERS IN LITERARY AND LINGUISTIC RESEARCH

Organizator: Association for Literary and Linguistic Research
Informacije: J.L.Dawson, Secretary, 1980 Symposium, Literary and Linguistic Computing Centre, Sidgewick Site, Cambridge CB3 9DA, England

31 marec-2 april Brighton, Velika Britanija

CAD 80 - 4th INTERNATIONAL CONFERENCE ON COMPUTERS IN ENGINEERING AND BUILDING DESIGN

Organizator: Inst. Of Mechanical Engineers, Inst. of Structural Engineers, Inst. of Civil Engineers, CAD Centre and other in cooperation with ACM SIGDA.
Informacije: Conf.chm. Gareth Jones, IPC Science and Technology Press Ltd., 32 High st., Guildford, Surrey, England GU1 3EW

8-11 april Dunaj, Avstrija

5th EUROPEAN MEETING ON CYBERNETICS AND SYSTEMS RESEARCH

Organizator: Austrian Society for Cybernetic Studies
Informacije: Austrian Society for Cybernetic Studies, Schottengasse 3, A-1010 Vienna, Austria

13-16 april, St. Louis, Mo., ZDA

AEDS ANNUAL CONVENTION

Organizator: AEDS
Informacije: Ralph Lee, University of Missouri-Rolla, Rolla, MO 65559

14-16 april, Schloss Retzhaf, Leibnitz, Austrija

1980 IFAC/IFIP WORKSHOP ON REAL TIME PROGRAMMING

Organizator: IFAC TC on Computers, IFIP TC on Computer Applications in Technology
Informacije: V.H. Haase, Institut für Informatik, Steyrergasse 17, A-8010 Graz, Austria

15-17 april, Hertford, Velika Britanija

IFAC WORK SHOP ON MANAGEMENT CONTROL SYSTEMS

Organizator: IFAC
Informacija: M.J.Yates, Institute of Measurement and control, 20 Peel Street, London, W8, JPD, Great Britain

16-18 April, Lougborough University of Technology, Velika Britanija

CONFERENCE ON THE THEORETICAL ASPECTS OF DISTRIBUTED COMPUTING

Organizator: BCS specialist group on Formal Aspects of Computing Science supported by Science Research Council
Informacije: D.J.Cooke, Dept.of Computer Studies, Loughborough University of Technology, Loughborough, Leics, LE11 3TU, U.K.

16-20 april Skopje, Jugoslavija

BIOMEDICINSKA KIBERNETIKA 1980

Organizator: Društvo za biokibernetiko
Informacije: M. Kon-Popovska, Matematički fakultet, pp 504, 91000 Skopje

21-22 april, West Lafayette, Ind., ZDA

WORKSHOP ON INTERCONNECTION NETWORKS FOR PARALLEL AND DISTRIBUTED PROCESSING

Organizator: ACM SIGARCH, IEEE-CS, TCCA, TCDP in cooperation with Purdue University School of EE
Informacije: H.J.Siegel, School of Electrical Engineering, Purdue University, W.Lafayette, IN 47907; 317 493-9252

22-24 april, Pariz, Francija

4th INTERNATIONAL SYMPOSIUM ON PROGRAMMING

Organizator: C.N.R.S. Universite Pierre et Marie Curie
Informacije: B.Robinet, Institut de Programmation, Universite Pierre et Marie Curie, 4 Place Jussieu, Paris 75005, France

21-23 april, Montreal, Kanada

INTERNATIONAL SYMPOSIUM ON COMPUTER NETWORK INTERCONNECTION

Organizator: Bell C.
Informacije: D.Tuyver, Bell Northern Research, P.O.Box 3511, Station C, Otawa, Canada K14 4A7

22-25 april, London, Velika Britanija

INTERNATIONAL CONFERENCE ON THE ELECTRONIC OFFICE

Organizator: IERE, IEE, IEEE, CIBS, BCS, ORS
Informacije: Conference Secretariat, IERE, 99 Gower St., London WC1E6AZ, England

28-30 april, Houston, Texas, ZDA

IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS

Organizator: IEE, Circuits and Systems Society
Informacije: Steven C.Bass, School of EE, Purdue University, W.Lafayette, In 47907; 317 493-2133

28-30 april, Los Angeles, Calif., ZDA

TWELFTH ANNUAL ACM SYMPOSIUM ON THEORY OF COMPUTING

Organizator: ACM, SIGACT and University of Southern California
Informacije: Richard J.Lipton, College of Engineering, University of California, Berkeley, CA 94720

28-30 april, Champaign. Ill., ZDA

SIXTH ILLINOIS CONFERENCE ON MEDICAL INFORMATION SYSTEMS

Organizator: University of Illinois, Society for Computer Medicine and others.
Informacije: Saundra Wheeler, Regional Health Resource Center, 1408 W. University Urbana, IL 61801

28 april - 1 maj, Linz, Austrija

IFIP WORKING CONFERENCE, FIRMWARE, MICROPROGRAMMING AND RESTRUCTURABLE HARDWARE

Organizator: IFIP, TC2 (Programming) and TC10 (Digital systems Design), EUROMICRO, IEEE TCMICRO, OCG in cooperation with ACM SIGMICRO
Informacije: Jörg Mühlbacher, Institut f. Statistik u. Informatik, Kepler University Linz, A-4045 Linz, Austria

30 april - 2 maj Pittsburgh, ZDA

11th ANNUAL PITTSBURGH CONFERENCE ON MODELING AND SIMULATION

Organizator: School of Engineering, University of Pittsburgh
Informacije: William G.Vogt or Marlin H. Mickle, Modeling and Simulation Conference, 348 Benedum Engineering Hal University of Pittsburgh, Pittsburgh, Pennsylvania 15269, USA

1-2 maj, Washington, ZDA

SECOND SYMPOSIUM ON MATHEMATICAL PROGRAMMING WITH DATA PERTURBATION

Organizator: The George Washington University
Informacije: Anthony V.Fiacco, Dept of Operations Research, School of Engineering and Applied Science,The George Washington Univ., Washington,DC 20052; 202 676-7511

2 maj, New York City, ZDA

ROLE OF DOCUMENTATION IN THE PROJECT LIFE CYCLE

Organizator: ACM, SIGDOC, SIGCOSIM
Informacije: Belden Menkus, Box 85, Middleville, NJ 07855; 201 383-3928

5-7 maj, Washington, ZDA

TIMS/ORSA NATIONAL MEETING

Organizator: TIMS/ORSA
Informacije: Donald Gross, School of Engineering, George Washington University, Washington, DC 20052

6-8 maj, La Baule, Francija

7th INTERNATIONAL SYMPOSIUM ON COMPUTER ARCHITECTURE

Organizator: IRISA, ACM French Chapter, ACM SIGARCH, IEEE CS, TCCA, IRIA
Informacije: Jaques Lenfant, IRISA, Universite de Rennes, Campus de Beaulieu, 35042 Rennes cedex, France

6-10 maj, Toronto, Kanada

1980 CAIS/ACSI CONFERENCE

Organizator: Canadian Association for Information
Informacije: John Wilson, 706-35 Wynford Heights, Don Mills, Ontatio, Canada M3C 1K9

8-9 maj, Niš, Jugoslavija

JUGOSLOVENSKO SAVETOVANJE O MIKROELEKTRONIKI MIEL-80

Organizator: ETAN, EI Niš
Informacije: Sekretarijat Elektrotehniške zveze Slovenije, Titova 50 GR, Ljubljana, Jugoslavija

11-14 maj, New Orleans, ZDA

ASM 1980 ANNUAL CONFERENCE

Organizator: Association for Systems Management
Informacije: R.B.McCaffrey, ASM, 24587 Bagley Road, Cleveland, OH 44138

12-14 maj, Victoria, B.Colum, Kanada

SESSION 80, CIPS ANNUAL CONFERENCE

Organizator: Canadian Information Processing Society
Informacije: Canadian Information Processing Society, 243 College St., Fifth Floor, Toronto, Ont., Canada M5T 2Y1

14-16 maj, Los Angeles, ZDA

1980 ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA

Organizator: ACM, SIGMOD
Informacije: Clay Sprowls, University of California Graduate School of Management, 405 Hilgard Ave., Los Angeles, Ca 90024; 213 825-7730

14-16 maj, Victoria, B.Colum., Kanada

THIRD NATIONAL CONFERENCE OF THE CANADIAN SOCIETY FOR COMPUTATIONAL STUDIES OF INTELIGENCE

Organizator: CSCI-SCEIO in coop. with other Canadian societies and University of Victoria
Informacije: L.K.Schubert, Dept. of Computing Science, U. of Alberta, Edmonton, Alberta, Canada P6G 2H1

13-16 maj, Toronto, Kanada

SRE-80 SYMPOSIUM ON RELIABILITY, MAINTAINABILITY, AND SAFELY OF TRANSPORTATION, INFORMATION, AND ENERGY SYSTEMS

Organizator: Society of Reliability Engineers
Informacije: I.N.McRae, SRE, Toronto, Chapter, Suite 1, 732 Wilson Ave., Downsview, Ontario M3K 1E2, Canada

19-21 maj, Atlanta, Ga., ZDA

ANNUAL TECHNICAL CONFERENCE OF AMERICAN SOCIETY FOR QUALITY CONTROL

Organizator: ASQC
Informacije: Darlene C.Schmidt, ASQC, 161 West Wisconsin Ave., Milwaukee, Wl 53203

19-22 maj, Anaheim, California, ZDA

NATIONAL COMPUTER CONFERENCE 80

Organizator: AFIPS
Informacije: Jerry Chriffriller, AFIPS, 210 Summit Ave., Montvale, NJ 07645; 201 391-9810

20-22 maj, Quebec City, Kanada

1980 CORS CONFERENCE

Organizator: Canadian Operational Research Society
Informacije: G.D'Avignon, Faculte Administration, Universite Laval, Quebec, Que., Canada G1K 7P4

21-22 maj, Montreal, Kanada

OPTIMIZATION DAYS 1980

Organizator: SIAM, IEEE Control Systems Society
Informacije: Alain Haurie, Dept. Methodes Quantitatives, Ecole des Hautes Etudes Commerciales, 5255

26-28 maj, Fort Lauderdale, Flo., ZDA

INTERNATIONAL WORKSKOP ON HIGH-LEVEL LANGUAGE COMPUTER ARCHITECTURE

Organizator: University of Maryland with support of Office of Naval Research
Informacije: Yaohan Chu, Dept. of Computer Science, University of Maryland, College Park, MD 20742

28-30 maj, Shiraz, Iran

IFAC/IFIP CONFERENCE ON SYSTEM APPROACH AND COMPUTER APPLICATIONS FOR DEVELOPMENT

Organizator: Iran Society of Automatic Control Engineers
Informacije: Secretary of IFAC/IFIP Conference, Iran 1980, PO BOX 737, Shiraz, Iran

28-30 maj, Toronto, Canada

PERFORMANCE 80, INTERNATIONAL SYMPOSIUM ON COMPUTER PERFORMANCE MODELING, MEASUREMENT AND EVALUATION

Organizator: IFIP W.G. 7.3. ACM Sigmetrics Conf.
Informacije: Kenneth C.Sevcik, Computer Systems Research Group, University of Toronto, Toronto, Canada M5S 1A1; 416 978-6219

3-5 junij, North western University Evanston, ZDA

TENTH INTERNATIONAL SYMPOSIUM ON MULTIPLE-VALUED LOGIC

Organizator: IEEE CS
Informacije: Jon T. Butler, Dept. of EE and Computer Science, Northwestern University, Evanston, IL 60201

3-6 junij, Chent, Belgija

4th INTERNATIONAL IFAC CONFERENCE ON INSTRUMENTATION AND AUTOMATION IN THE PAPER, RUBBER, PLASTICS, AND POLYMERIZATION INDUSTRIES

Organizator: IFAC
Informacije: IFAC-P.R.P. Automation Conference, Jan Van Rijswijeklaan, 58, B-2000 Antwerp, Belgium

16-18 junij, Trondheim, Norveška

IFAC/IFIP SYMPOSIUM ON AUTOMATION FOR SAFETY IN SHIPPING AND OFFSHORE OPERATIONS

Organizator: IFAC, IFIP, SINTEF, Norwegian Petroleum Directorate
Informacije: SINTEF, Automatic Control Division, 7034 Trondheim-NTH, Norway

16-18 junij, Seattle, Wash., ZDA

1980 INTERNATIONAL CONFERENCE ON COMMUNICATIONS

Organizator: 1980 International Conference on Communications
Informacije: P.R. Metz, ICC 80, Box 88465, Seattle, Wa. 98188, ZDA

16-19 junij, Warsaw, Poljska

3rd IFAC/IFORS CONFERENCE ON MODELLING AND CONTROL OF NATIONAL ECONOMIES

Organizator: Systems Research Institute - Polish Academy of Sciences
Informacije: Dr. M.Lipiec, Systems Research Institute Polish Academy of Sciences, 6 Newelska st., 01-477 Warsaw, Poland

15-20 junij, Bad Honnef, Bonn, ZRN

SIXTH WORKSHOP ON GRAPTHEORETIC CONCEPTS IN COMPUTER SCIENCE (WG80)

Organizator: European Association for Theoretical Computer Science
Informacije: Hartmut Nottemeier, Lehrstuhl für Informatik III, RWTH Aachen, Büchel 29/31, D-5100 Aachen, Germany

16-20 junij, Montreal, Kanada

SECOND INTERNATIONAL SYMPOSIUM ON INNOVATIVE NUMERICAL ANALYSIS IN APPLIED ENGINEERING SCIENCE

Organizator: I.Cruse, EB-3S3, Pratt & Whitney Aircraft, East Hartford, CT 06108
Informacije: I.Cruse, EB-3S3, Pratt & Whitney Aircraft, East Hartford, CT 06108

16-21 junij, Hong Kong

RECENT ADVANCES IN MATHEMATICS AND ITS APPLICATIONS

Organizator: Southeast Asian Mathematical Society
Informacije: SAMS, Baptist College Hong Kong, or Polytechnic, Department of Mathematics, Hong Kong

17-19 junij, Tulsa, Oklahoma, ZDA

COMPUTERIZED ENERGY MANAGEMENT CONTROL SYSTEMS CONFERENCE

Organizator: Oklahoma State University, Dept. of Energy
Informacije: Wayne C.Turner or Phillip M. Wolfe, School of Industrial Engineering and Management, 322 Engineering North, OSU, Stillwater, OK 74074

18-21 junij, Philadelphia, ZDA

ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS

Organizaotr: ACL
Informacije: Don Walker, Artificial Intelligence Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025; 415 326-6200 x3071

23-25 junij, Minneapolis, ZDA

SEVENTEENTH DESIGN AUTOMATION CONFERENCE

Organizator: ACM, SIGDA, IEEE-CS, DATC
Informacije: Edwin B.Hassler, Texas Instruments Inc. Box 225621, M.S. 3907, Dallas, TX 75265; 214 238-5781

AVTORJI IN SODELAVCI

Wanda Jurišić-Kette. Diplomirala Prirodoslovno matematički fakultet u Zagrebu, diplomirala Elektrotehnički fakultet u Zagrebu te magistrirala na istom fakultetu. Radila na Elektrotehničkom fakultetu, Institutu Rudjer Bošković, Sveučilišnom računskom centru i Jugoturbini-EAB. Sada radi u RIZ-OD, Zagreb. Viši predavać je na katedri za informatiku filozofskog fakulteta u Zagrebu. U zadnje vrijeme rad je usmerjen na primjenu malih sistema za obradu poslovanja te uvodjenje i primjenu informacijskih sistema.

Zarko Nozica (1947), diplomirao je 1971. na Elektrotehničkom fakultetu u Zagrebu, smjer Elektronika-usmjerenje Automatika. Nakon diplomiranja zaposlio se na Zavodu za elektroniku Elektrotehničkog fakulteta u Zagrebu, gdje radi i danas kao znanstveni asistent. Magistrirao je 1974. na području modeliranja i analize primjenom računala. Objavio je veći broj radova. Bavi se analizom i projektiranjem pomoću računala, te ostvarivanjem sistema s visokim zahtjevima za pouzdanim radom.

Tomaž Kalin (1936) diplomiral na Fakulteti za naravoslovje in tehnologijo in doktoriral prav tako na FNT s področja nevtronske fizike. Od leta 1972 dela kot analitik sistema na Republiškem računskem centru, kjer je odgovoren za spremljanje delovanja sistema in planiranje razvoja konfiguracije. Sodeluje v mednarodnem projektu Evropska računalniška mreža, kjer je bil 1977/78 tehnični pomočnik direktorja projekta. Koncem leta 1979 je bil izvoljen za docenta na Fakulteti za elektrotehniko v Ljubljani. Je avtor ali koavtor vrste del.

RAZISKOVALNE NALOGE, PRIJAVLJENE NA RSS V LETU 1979

V tej rubriki objavljamo kratke povzetke raziskovalnih nalog, ki jih financira Področna raziskovalna skupnost za avtomatiko, računalništvo in informatiko, ki so s področja računalništva in informatike.

Naslov naloge: Regulacija vrtilne hitrosti enosmernih motorjev s spreminjanjem polja, 2.faza

Projekt: Krmilja in regulacije

Nosilec naloge: Rafael Cajhen, Fakulteta za elektrotehniko, Ljubljana

Program raziskave:

Okvirni program 2-letne raziskave:

1. leto: Teoretična in tehnološka problematika pri krmiljenju vrtilne hitrosti enosmernega motorja preko vzbujanja: motor, merilna in regulacijska oprema, regulacijsko-tehnične variante.

2. leto: Optimalno vodenje, problematika adaptivnosti regulacije, kombinirana regulacija preko rotorske in vzbujalne napetosti, projektne smernice.

Naslov naloge: Računalniški model temperaturnega polja izohorne ekstruzije

Projekt: Računalniško projektiranje

Nosilec naloge: Peter Leš, VTŠ, VTO strojništvo, Maribor

Program raziskave:

Na osnovi naših dosedanjih izkušenj na področjih preoblikovanja, metalurgije, mehanike, termotehnike in računalništva bomo 1979 (oz. v 12 mesecih od odobritve naloge) izdelali računalniški program, ki bo izvrednotil temperaturne razmere v preoblikovalnem območju. Uporabili bomo metodo končnih razlik (ki za toplotne izračune daje enako ugodne rezultate kot bolj moderna metoda končnih elementov). V naslednjih letih bi lahko obdelali tudi problem določitve končnih elementov za obe področji dela (tj. termo-plastifikacija).

Naslov naloge: Materialna/programska oprema za računalniško industrijo

Projekt: Računalniška tehnika in proizvodnja

Nosilec naloge: Anton P.Železnikar, IJS, Ljubljana

Program raziskave:

Za pogodbeno leto 1979 je predviden naslednji okvirni program dela:

1. Metodologija načrtovanja mikroračunalniških sistemov
2. Raziskave podsistemov s centralnimi procesorskimi enotami, krmilniki za diske, za prekinjanje, za časovne funkcije, za direkten pomnilniški dostop, za prikazovanje, za serijski in paralelni vhod/izhod.
3. Raziskave mikroračunalniške sistemske programske opreme (operacijski sistemi, zbirniki, prevajalniki, urejevalniki, nalagalniki), in sicer tako rezidentne kot prečne programske opreme.
4. Raziskave multimikroprogramskih sistemov in paralelnega programiranja mikro sistemov.
5. Teoretične raziskave distribuiranih informacijskih sistemov, multi uporabniški sistemi.
6. Samorazširljivi in samoorganizacijski programirni koncepti in metode kot sredstvo za pospešeno proizvodnjo mikroračunalniške programske opreme.
7. Vrednotenje dinamike programov in procesorjev (meritve izvajalnih časov, lokalibilnosti, pojavnosti spremenljivk, odkrivanje napak, regeneracija programa).
8. Ocenjevanje najnovejše tehnologije na področju mikroračunalniške matrialne in programske opreme.
9. Osnovni eksperimenti z mikroračunalniki in primitivnimi roboti: operacijski sistem za gibajoči robot z dvema rokama.

Naslov naloge: Simulacija velikih sistemov na mikroprocesorskih strukturah

Projekt: Krmilja in regulacije

Nosilec naloge: Karel Jezernik, VTŠ, VTO Elektrotehnika, Maribor

Program raziskave:

Nalogo prijavljamo kot triletno s tem, da bomo vsako raziskovalno leto zaključili z elaboratom.

Pri raziskavi bomo paralelno razvijali metodo simulacije na procesnem računalniku s ciljem aplikacije v mikroračunalniku. Za reševanje naših aplikacij bomo raziskali propustnost in kapaciteto mikroračunalniških sistemov. Zgradili bomo mikroračunalnik v strukturi sposobni delovanja v realnem času. Razvili bomo mikroračunalniške programe za karakteristične aplikacije elektroenergetskih naprav in regulacijskih podsistemov. Razviti računalniški programi v fortramu nam bodo služili pri izdelavi diagramov potekov obravnavanih aplikacij.

V prvem raziskovalnem letu bomo zgradili mikroprocesor s potrebnimi vhodno izhodnimi enotami. Razvili bomo programe za simulacijo regulacije vzbujanje in frekvence sinhronskega stroja. Kompliciran dinamični sistem sinhronskega stroja bomo simulirali na procesnem računalniku. Raziskali bomo aplikacije mikroprocesorja v elektrarni.

Naslov naloge: Grafični sistemi za računalniško projektiranje

Projekt: Računalniško projektiranje

Nosilec naloge: Anton Jezernik, VTŠ, Maribor

Program raziskave:

Program raziskave je tro-leten s konkretnimi rezultati ob koncu vsakega leta. V prvem letu bo možno izdelati en del grafičnega sistema za generacijo podatkov in risanje rezultatov in vskladiti oz. vpeljati nekatere računske programe za konstrukcije na miniračunalnik. Večji konstrukcijski programi (SAP, BERSAFE) se bodo še vnaprej uporabljali na večjih računalniških sistemih n. pr. CYBER 70-72 povezava (interface) pa bo ustvarjena z grafičnim sistemom preko magnetnih trakov.

Naslov naloge: Razpoznavanje rokopisov,tipkopisov in prstnih odtisov (Razpoznavalni sistem celih in delnih prstnih odtisov ter sledi)

Projekt: Individualne naloge

Nosilec naloge: Ludvik Gyergyek, Fakulteta za elektrotehniko, Ljubljana

Program raziskave:

- Pretvorba slikovnih vzorcev v digitalno obliko in shranjevanje na masovni pomnilnik.
- Določitev in računalniška implementacija postopka predobdelave.
- Določitev monitorskih ukazov za delo s sistemom.
- Določitev postopkov za vnašanje značilnih točk v računalnik preko grafičnega zaslona.
- Prireditev in povezava že razvitih postopkov za avtomatsko razpoznavanje prstnih odtisov na računalniku PDP 11.
- Preizkus razpoznavalnega sistema s testnimi vzorci.

Naslov naloge: Uporaba numeričnih metod v analizi in sintezi regulacijskih sistemov

Projekt: Krmilja in regulacije

Nosilec naloge: Edvard Kiker, VTŠ, Maribor

Program raziskave:

V programu raziskovalnega dela za leto 1979 je smotrna razčlenitev programskega paketa z ozirom na naloge, ki jih mora le-ta opravljati. Določili bomo zgradbo celotnega programskega paketa, ki ga bomo izgrajevali po posameznih fazah, predvidene bodo vstopne točke in oblika izhodnih poročil oziroma diagramov. Programski paket mora zadoščati zahtevi, da opravi analizo in sintezo regulacijskega sistema korakoma, uporabljajoč rezultate predhodnih korakov (programov), razen tega pa morajo biti posamezne analize dostopne tudi ločeno. Med te spadajo: izračun časovnega poteka, frekvenčna karakteristika, Bodejev diagram, Nyquistov diagram, Nicholsov diagram, diagram lege korenov in fazni diagram. Vhodi in izhodi teh programskih modulov naj bodo posebej dostopni. Modul za parametrsko optimizacijo pa je nasprotno odvisen od izsledkov predhodnih faz. Druga zahteva, ki jo bomo pri zgradbi programskega paketa upoštevali je standardizacija vhodnih signalov oziroma motenj (upoštevali bomo tudi r ljučne motnje) ter standardiza-

cija najpogosteje uporabljanih nelinearnih karakteristik; le-te bomo v programe permanentno vgradili. Predvideli bomo omejitve, s katerimi bo program računal.

Naslov naloge: Model risalnika krmiljenega z računalnikom

Projekt: Računalniška tehnika in proizvodnja

Nosilec naloge: Zdravko Ženko, VTŠ, Maribor

Program raziskave:

V okviru raziskovalne naloge bomo izdelali model risalnika krmiljenega z mikroprocesorjem in opravili v zvezi s tem osnovne raziskave. Raziskava bo omogočila analizo vseh tistih parametrov na osnovi katerih bo nato mogoče izdelati prototip kvalitetnega risalnika, na osnovi katerega bo možno pričeti serijsko proizvodnjo.

NAVODILO ZA PRIPRAVO ČLANKA

Avtorje prosimo, da pošljejo uredništvu naslov in kratek povzetek članka ter navedejo približen obseg članka (število strani A 4 formata). Uredništvo bo nato poslalo avtorjem ustrezno število formularjev z navodilom.

Članek tipkajte na priložene dvokolonske formularje. Če potrebujete dodatne formularje, lahko uporabite bel papir istih dimenzij. Pri tem pa se morate držati predpisanega formata, vendar pa ga ne vrišite na papir.

Bodite natančni pri tipkanju in temeljiti pri korigiranju. Vaš članek bo s foto postopkom pomanjšan in pripravljen za tisk brez kakršnihkoli dodatnik korektur.

Uporabljajte kvaliteten pisalni stroj. Če le tekst dopušča uporabljajte enojni presledek. Črni trak je obvezen.

Članek tipkajte v prostor obrobljen z modrimi črtami. Tipkajte do črt - ne preko njih. Odstavek ločite z dvojnim presledkom in brez zamikanja prve vrstice novega odstavka.

Prva stran članka :

a) v sredino zgornjega okvira na prvi strani napišite naslov članka z velikimi črkami;
b) v sredino pod naslov članka napišite imena avtorjev, ime podjetja, mesto, državo;
c) na označenem mestu čez oba stolpca napišite povzetek članka v jeziku, v katerem je napisan članek. Povzetek naj ne bo daljši od 10 vrst.
d) če članek ni v angleščini, ampak v katerem od jugoslovanskih jezikov izpustite 2 cm in napišite povzetek tudi v angleščini. Pred povzetkom napišite angleški naslov članka z velikimi črkami. Povzetek naj ne bo daljši od 10 vrst. Če je članek v tujem jeziku napišite povzetek tudi v enem od jugoslovanskih jezikov;
e) izpustite 2 cm in pričnite v levo kolono pisati članek.

Druga in naslednje strani članka:
Kot je označeno na formularju začnite tipkati tekst druge in naslednjih strani v zgornjem levem kotu,

Naslovi poglavij:
naslove ločuje od ostalega teksta dvojni presledek.

Če nekaterih znakov ne morete vpisati s strojem jih čitljivo vpišite s črnim črnilom ali svinčnikom. Ne uporabljajte modrega črnila, ker se z njim napisani znaki ne bodo preslikali.

Ilustracije morajo biti ostre, jasne in črno bele. Če jih vključite v tekst, se morajo skladati s predpisanim formatom. Lahko pa jih vstavite tudi na konec članka, vendar morajo v tem primeru ostati v mejah skupnega dvokolonskega formata. Vse ilustracije morate (nalepiti) vstaviti sami na ustrezno mesto.

Napake pri tipkanju se lahko popravljajo s korekcijsko folijo ali belim tušem. Napačne besede, stavke ali odstavke pa lahko ponovno natipkate na neprozoren papir in ga pazljivo nalepite na mesto napake.

V zgornjem desnem kotu izven modro označenega roba oštevilčite strani članka s svinčnikom, tako da jih je mogoče zbrisati.

Časopis INFORMATICA
Uredništvo, Institut Jožef Stefan, Jamova 39, Ljubljana

Naročam se na časopis INFORMATICA. Predplačilo bom izvršil po prejemu vaše položnice.

Cenik: letna naročnina za delovne organizacije 300,00 din, za posameznika 100,00 din.

Časopis mi pošiljajte na naslov ☐ stanovanja ☐ delovne organizacije.

Priimek.....

Ime.....

Naslov stanovanja

Ulica.....

Poštna številka_____ Kraj.....

Naslov delovne organizacije

Delovna organizacija.....

.....

Ulica.....

Poštna številka_____ Kraj.....

Datum..... Podpis:

.....

INSTRUCTIONS FOR PREPARATION OF A MANUSCRIPT

Authors are invited to send in the address and short summary of their articles and indicate the opproximate size of their contributions (in terms of A 4 paper). Subsequently they will receive the outor's kits.

Type your manuscript on the enclosed two-column-format manuscript paper. If you require additional manuscript paper you can use similar-size white paper and keep the proposed format but in that case please do not draw the format limits on the paper.

Be accurate in your typing and through in your proof reading. This manuscript will be photographically reduced for reproduction without any proof reading or corrections before printing.

Use a good typewriter. If the text allows it, use single spacing. Use a black ribbon only.

Keep your copy within the blue margin lines on the paper, typing to the lines, but not beyond them. Double space between paragraphs.

First page manuscript:

a) Give title of the paper in the upper box on the first page. Use block letters.
b) Under the title give author's names, company name, city and state - all centered.
c) As it is marked, begin the abstract of the paper. Type over both the columns. The abstract should be written in the language of the paper and should not excesed 10 lines.
d) If the paper is not in English, drop 2 cm after having written the abstract in the language of the paper and write the abstract in English as well. In front of the abstract put the English title of the paper. Use block letters for the title. The lenght of the abstract should not be greater than 10 lines.
e) Drop 2 cm and begin the text of the paper in the left column.

Second and succeeding pages of the manuscript:
As it is marked on the paper, begin the text of the second and succeeding pages in the left upper corner.

Format of the subject headings:
Headings are separated from text by double spacing.

If some characters are not available on your typwriter write them legibly in black ink or with a pencil. Do not use blue ink, because it snows poorly.

Illustrations must be black and white, sharp and clear. If you incorporate your illustrations into the text keep the proposed format. Illustration can also be placed at the end of all text material provided, however, that they are kept within the margin lines of the full size two-column format. All illustrations must be placed into appropriate positons in the text by the author.

Typing errors may be corrected by using white correction paint or by retyping the word, sentence or paragraph on a piece of apaque, white paper and pasting it nearly over errors

Use pencil to number each page on the upper-right-hand corner of the manuscript, outside the blue margin lines so that the numbers may be erased.

Časopis INFORMATICA
Uredništvo, Institut Jožef Stefan, Jamova 39, Ljubljana

Please enter my subscription to INFORMATICA and send me the bill.

Annual subscription price: companies 300,00 din (for abroad US $ 18), individuals 100,00 din (for abroad US $ 6)

Send journal to my ☐ home address ☐ company's address.

Surname.....

Name.....

Home address

Street.....

Postal code _____ City.....

Company address

Company.....

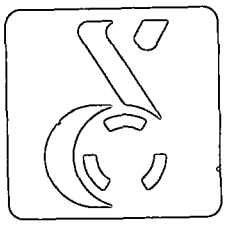
.....

Street.....

Postal code _____ City.....

Date..... Signature

.....



delta sistemi

ELEKTROTEHNA LJUBLJANA, TOZD za računalništvo Digital proizvaja in prodaja naslednje standardne računalniške konfiguracije:

DELTA 700/80

- DELTA 700 centralna procesna enota
- 512 KByte centralni pomnilnik s paritetno kontrolo,ki se lahko razširi do 4 MBytov
- 2 KByte vmesni pomnilnik spomina (cache)
- ura realnega časa
- konzolni terminal s kontrolno enoto
- dve diskovni enoti s kapaciteto po 80 MByte s kontrolno enoto
- dve magnetni tračni enoti 800/1600 b/i, 45 i/s, 9 kanalni zapis s kontrolno enoto
- asinhroni komunikacijski vmesnik
  (8 linij: EIA/CCITT modemski izhod)
  (8 linij: 20 mA tokovna zanka)
- 600 linijski tiskalnik
- KOPA 1000 alfanumerični video display terminal (2 kom)

DELTA 340/80

- DELTA 340 centralna procesna enota
- 256 KByte centralni pomnilnik s paritetno kontrolo
- 2 KByte vmesni pomnilnik (cache)
- ura realnega časa
- konzolni terminal s kontrolno enoto
- enota za baterijsko napajanje pomnilnika
- procesor s plavajočo vejico (floating point processor)
- dve diskovni enoti s kapaciteto po 80 MByte s kontrolno enoto
- dve magnetni tračni enoti (1600 b/i, 75 i/s, 9 kanalni zapis), s kontrolno enoto
- asinhroni komunikacijski vmesnik
  (8 linij EIA/CCITT modemski izhod)
  (8 linij 20 mA tokovne zanke)
- 600 linijski tiskalnik
- KOPA 1000 alafanumerični video display terminal
  (2 kom.)

DELTA 340/5

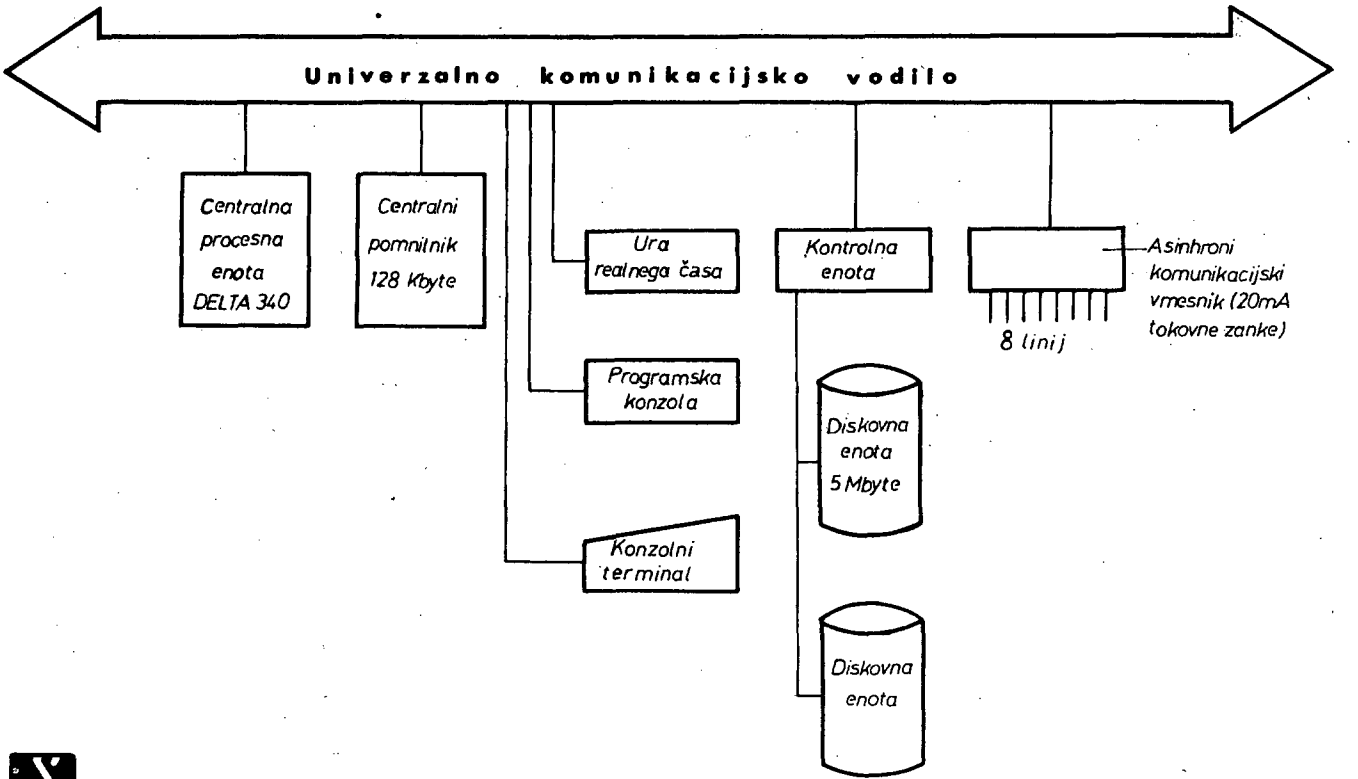
- DELTA 340 centralna procesna enota
- 128 KByte centralni pomnilnik s paritetno kontrolo, ki se lahko razširi do 256 KBytov
- ure realnega časa
- konzolni terminal s kontrolno enoto
- dve diskovni enoti s kapaciteto po 5 MByte s kontrolno enoto
- asinhroni komunikacijski vmesnik
  (8 linij: 20 mA tokovne zanke)

DELTA 340/40

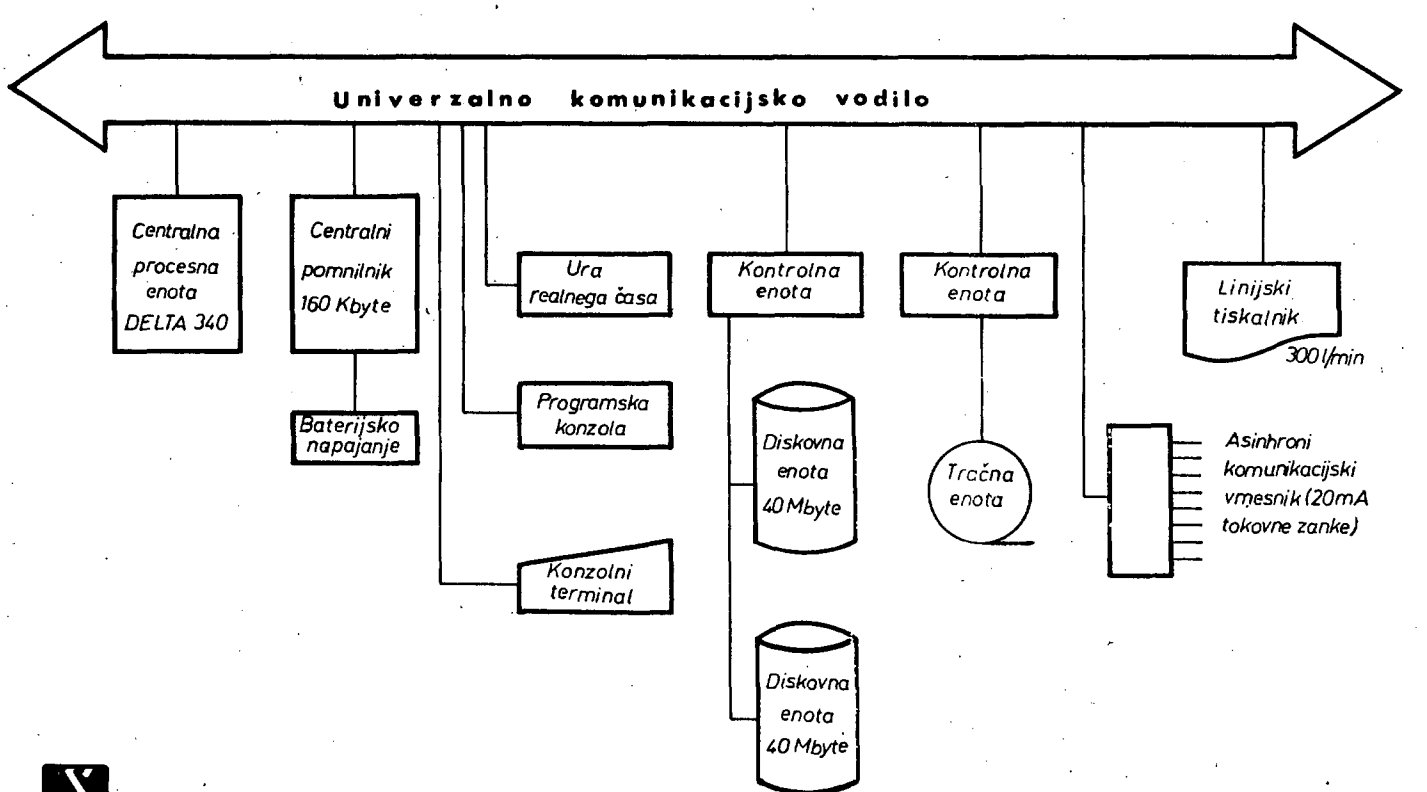
- DELTA 340 centralna procesna enota
- 160 KByte centralni pomnilnik s paritetno kontrolo do 256 KBytov
- ure realnega časa
- konzolni terminal s kontrolno enoto
- enota za baterijsko napajanje pomnilnika
- dve diskovni enoti s kapaciteto po 40 MByte s kontrolno enoto
- ena magnetna tračna enota (1600 b/i, 75 i/s, 9 kanalni zapis) s kontrolno enoto
- asinhroni kumunikacijski vmesnik
  (8 linij: 20 mA tokovne zanke)
- 300 linijski tiskalnik

NAŠTETE STANDARDNE KONFIGURACIJE LAHKO RAZŠIRITE S PRIKLJUČEVANJEM NOVIH VHODNO-IZHODNO ENOT, POVEČANJEM POMNILNIKA IPD.

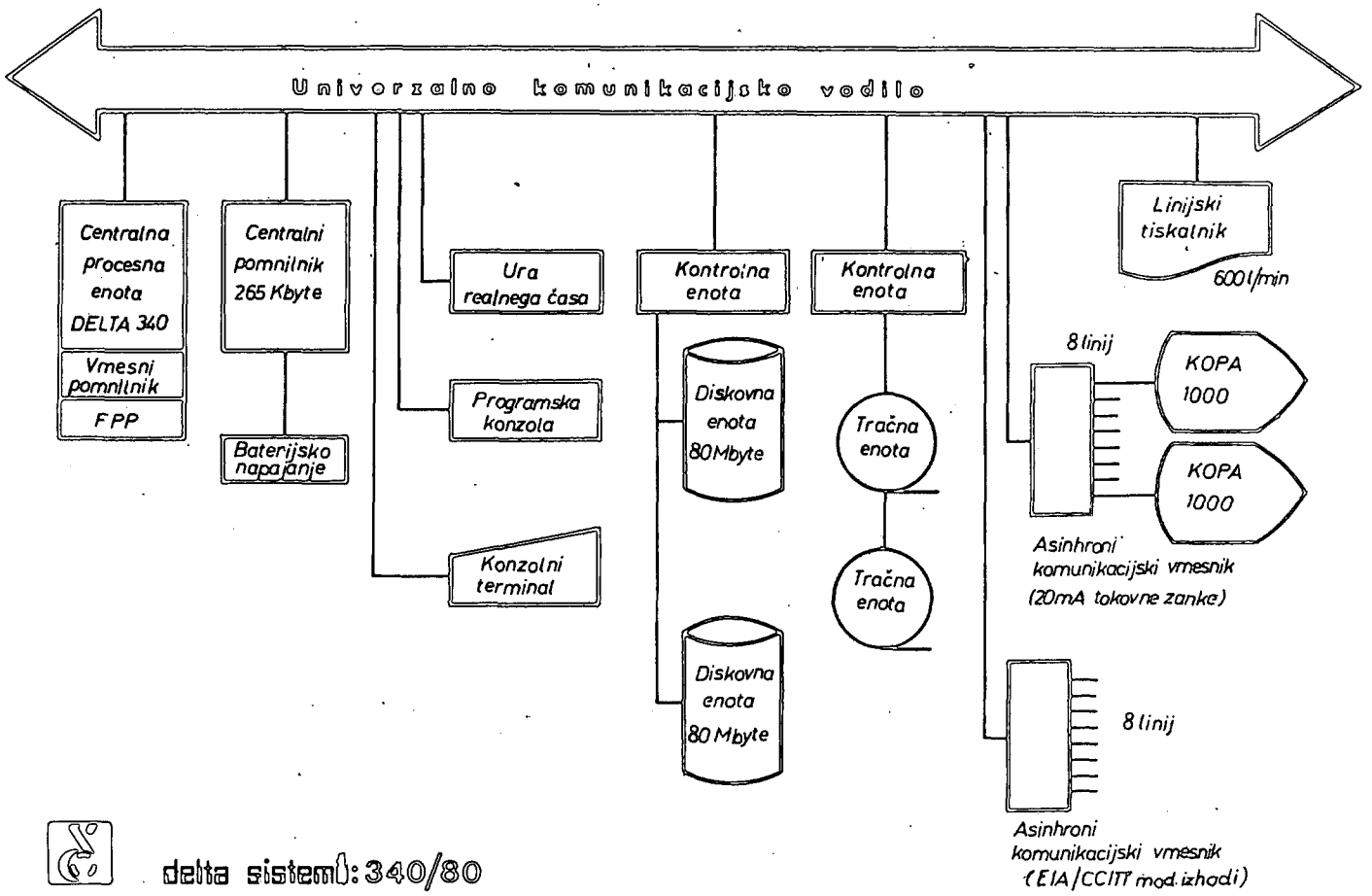
SISTEMSKI PAKETI DELTA 700/80, 340/80, 340/40 IN 340/5 VKLJUČUJEJO TUDI: OPERACIJSKI SISTEM DELTA/M S PREVAJALNIKI IN APLIKATIVNIMI PROGRAMI, ŠOLANJE V LASTNEM IZOBRAŽEVALNEM CENTRU, POMOČ PRI UVAJANJU PROGRAMSKE OPREME, INSTALACIJO RAČUNALNIŠKEGA SISTEMA IN ENOLETNO GARANCIJO ZA STROJNO IN PROGRAMSKO OPREMO.



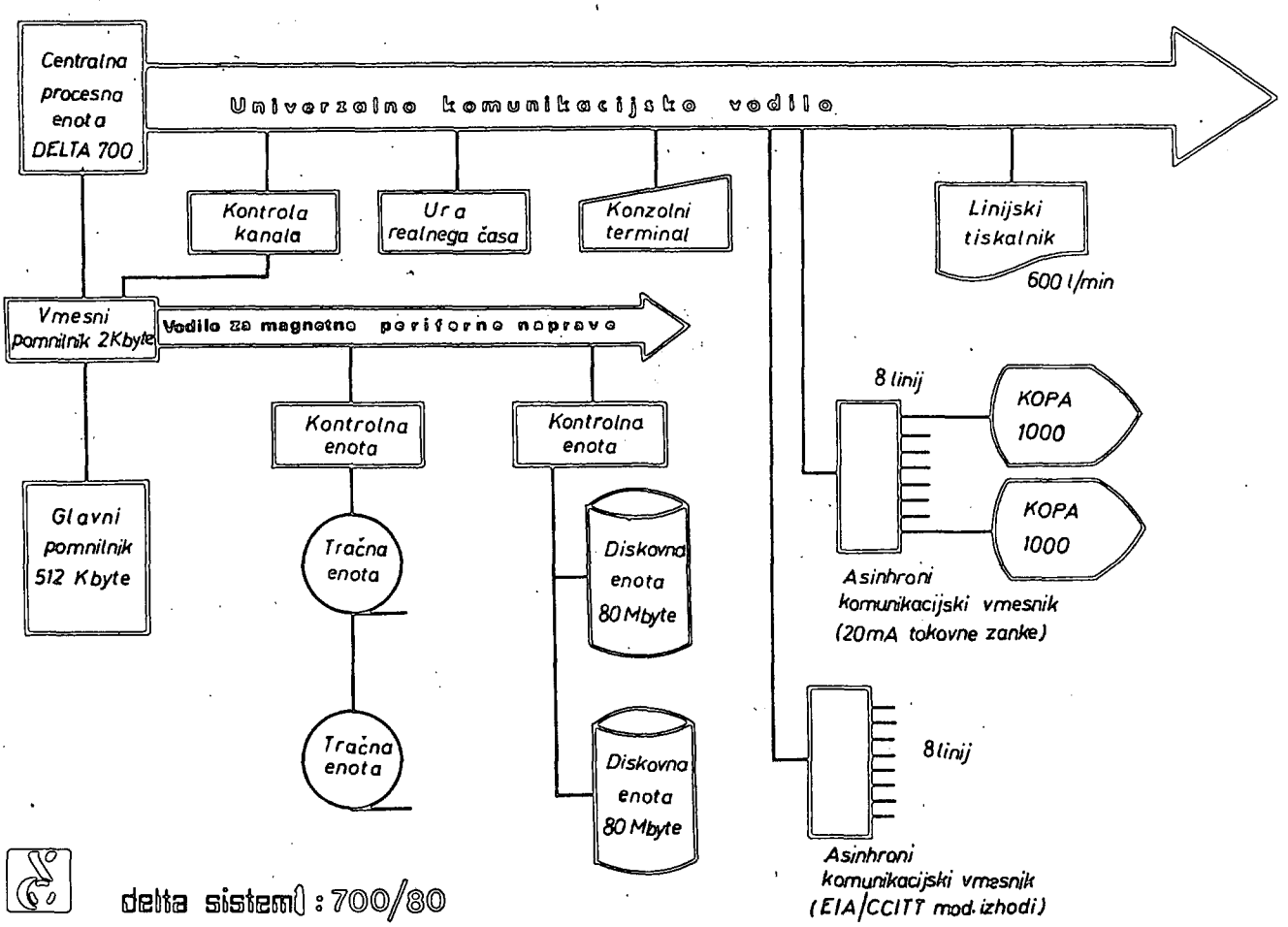
delta sistem: 340/5



delta sistem: 340/40



delta sistem: 340/80



delta sistem: 700/80

PROGRAMSKA OPREMA DELTA SISTEMOV

Osnovo sistemske programske opreme predstavlja DELTA/M operacijski sistem, ki je namenjen za delo v realnem času in časovno dodeljevanje resursov do 256 uporabnikom, ki lahko istočasno uporabljajo sistem. Glavna karakteristika DELTA/M sistema je interaktivnost. Človek in sistem komunicirata preko posebne enote, ki je običajno video terminal. Vsak monitorski ukaz se lahko vnese preko poljubnega terminala, če le uporabnikovo geslo zadošča ustrezni stopnji tajnosti. To pomeni z vidika uporabnika enake možnosti, kot da bi delal sam na sistemu.

Večuporabniško okolje zahteva zaščito med uporabniki samimi, saj bi lahko napaka enega uporabnika povzročila težave vsem drugim. Zaradi tega obstaja med uporabniki zaščita na nivoju programske opreme in na nivoju strojne opreme. Vsak disk je razdeljen v več logičnih področij, od katerih jih vsak uporabnik lahko nekaj uporablja . Praktično to pomeni, da lahko briše samo svoje nize in bere nize drugih uporabnikov, če mu le-ti to dovolijo. Elektronsko pa je zaščiten adresni prostor programov in uporaba instrukcij, ki bi lahko porušile integriteto sistema. Te lahko uporablja samo izvajalni sistem.

Multiprogramiranje je realizirano na nivoju sistema kot celote in na nivoju posameznega terminala. Tako ima lahko vsak uporabnik lastni multiprograming. To je važno predvsem za programerje, saj lahko istočasno razvijajo (prevajalnik, povezovalnik) in testirajo (izvajajo) programe.

Velika hitrost procesorja in perifernih enot ter učinkovito oblikovana programska oprema omogočata gospodarno uporabo vseh komponent DELTA računalnika.

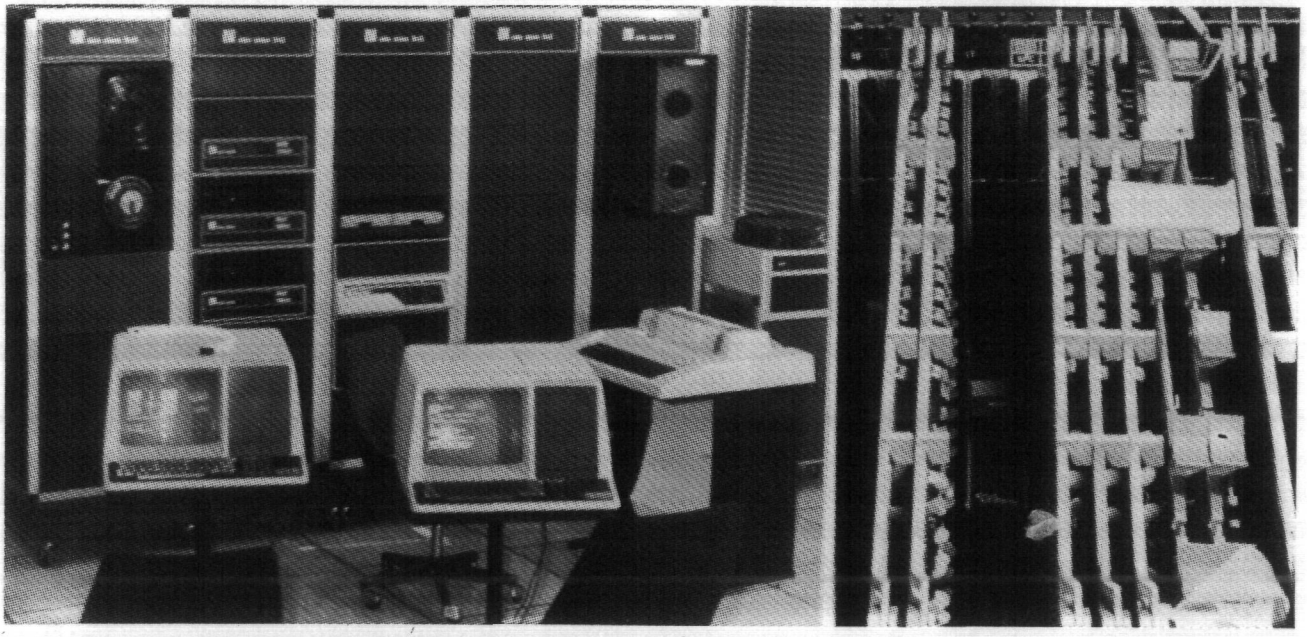
Sistem lahko istočasno upravlja industrijski proces(visoka prioriteta - realni čas), interaktivne poslovne aplikacije(srednja prioriteta), razvoj novih programov v poljubnih programskih jezikih(standardna prioriteta) in paketne obdelave (nizka prioriteta).

Aplikacijski programi se lahko pišejo v MACRO zbirnem ali enem od višjih programskih jezikov:

- FORTRAN IV
- FORTRAN IV PLUS
- BASIC 11
- RPG II
- COBOL (ANSI 74 standard)
- BASIC-PLUS-2
- PASCAL
- DATATRIEVE 11

Na tržišču ugotavljamo velike potrebe po kvalitetni komunikacijski opremi, zato posvečamo veliko pozornost prav temu področju. Komunikacijska programska oprema na DELTA/M je eden od poslov, ki se odvija v multiprogramingu in omogoča povezavo z računalniki: DELTA, PDP-11, VAX, DEC-10, DEC-20, CDC-6600, IBM 360/370, UNIVAC-11.

DELTA sistemi so namenjeni splošni uporabi. Zato je v osnovni paket vedno vključena samo tista programska oprema, ki je potrebna vsem uporabnikom. Vsak pa si lahko izbere dodatno sistemsko ali aplikativno programsko opremo. DELTA/M namreč ohranja popolno kompatibilnost navzdol z RSX-11/M operacijskim sistemom firme DEC. Ta operacijski sistem je zelo razširjen, zato je tudi ponudba ELEKTROTEHNE, DEC-a in drugih proizvajalcev zelo velika.



PODROBNE INFORMACIJE O NAKUPU DELTA SISTEMOV NUDI ELEKTROTEHNA LJUBLJANA, TOZD ZA RAČUNALNIŠTVO DIGITAL:

LJUBLJANA
Linhartova 62a
tel. (061) 323-585

ZAGREB
Aleja Borisa Kidriča 2
tel. (041) 516-690

BEOGRAD
Karadorđev trg 13
tel. (011) 694-537