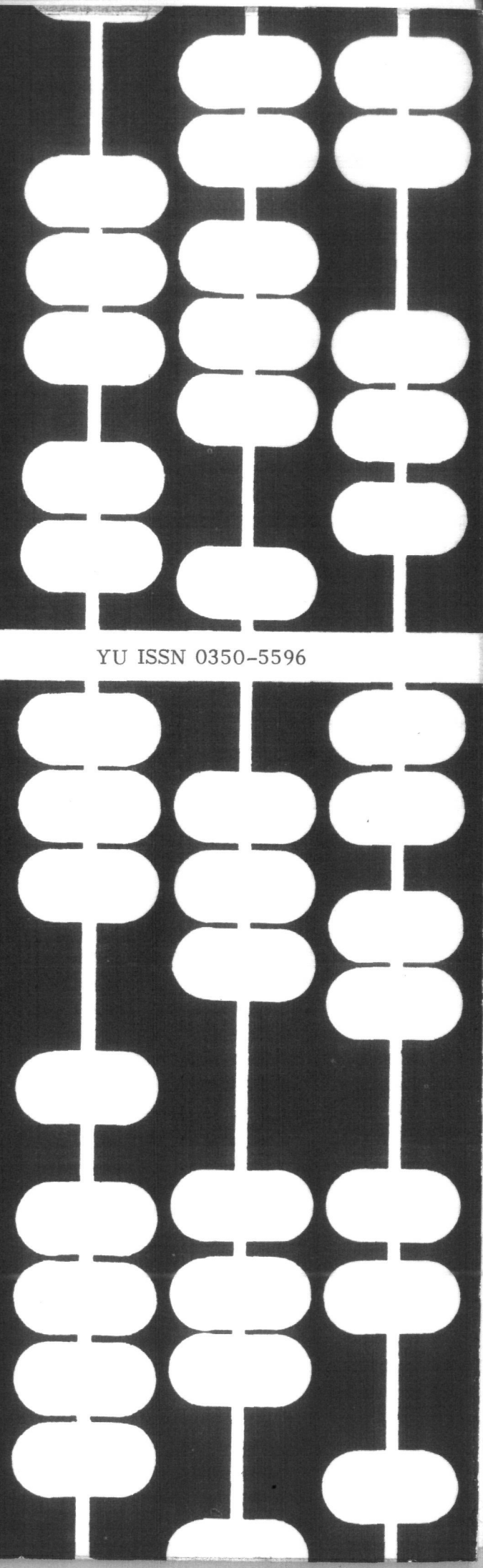


INFORMATICA



YU ISSN 0350-5596

4

1977

ŽELITE ?

... svetovalsko pomoč s področja sistemskih programov, valjinske obdelave podatkov in razširitve sistema?

... že izdelane in preizkušene programe, ki obdelujejo podatke posameznih podsistemov podjetja?

... v vaši organizaciji uvesti obdelavo podatkov, pa nimate svojega računalniškega sistema?

... imeti rešene statistične, transportne, linearne ali simulacijske probleme ?

... programe, ki bodo napisani za vašo delovno organizacijo?

... nasvet, kako dobiti od svojega računalniškega sistema kvalitetnejše rezultate s področja poslovne informatike?

... popoln servis obdelave podatkov v vaši delovni organizaciji?

KLICITE NAS !

UNIVERZA v LJUBLJANI



INSTITUT "JOŽEF STEFAN" odsek za uporabno matematiko

61001 Ljubljana jamova 39 jugoslavija

tel. (061) 63-261 tlx: 31-296 yu jostin

INFORMATICA

časopis za tehnologijo računalništva in
probleme informatike
časopis za računalnsku tehnologiju i pro-
bleme informatike
spisanie za tehnologija na smetanjeto i
problemi od oblasta na informatikata

Časopis Izdaja-Slovensko društvo INFORMATIKA,
61000 Ljubljana, Jamova 39, Jugoslavija

YU ISSN 0350-5596

UREDNIŠKI ODBOR:

Člani: T. Aleksić, Beograd, D. Bitrakov, Skopje, P. Dra-
gojlović, Reka, S. Hodžar, Ljubljana, B. Horvat,
Maribor, A. Mandžić, Sarajevo, S. Mihalić, Va-
raždin, S. Turk, Zagreb.

Glavni in odgovorni urednik: A.P. Železnikar

TEHNIČNI ODBOR:

Uredniki področij:

- V. Batagelj - programiranje
- I. Bratko - umetna inteligenca
- D. Čečez-Kecmanović - informacijski sistemi
- M. Exel - operacijski sistemi
- A. Jerman-Blažič - novice založništva
- B. Jerman-Blažič-Džonova - literatura in srečanja
- L. Lenart - procesna informatika
- D. Novak - mikro računalniki
- N. Papić - študentska vprašanja
- L. Pipan - terminologija
- B. Popović - novice in zanimivosti
- V. Rajković - vzgoja in izobraževanje
- M. Špegel, M. Vukobratović - robotika
- P. Tancig - računalništvo v humanističnih in družbe-
nih vedah
- S. Turk - materialna oprema

Tehnični urednik: R. Murn

ZALOŽNIŠKI SVET

- T. Banovec, Zavod SR Slovenije za družbeno planiranje,
Ljubljana
- A. Jerman-Blažič, Slovensko društvo Informatika,
Ljubljana
- B. Klemenčič, ISKRA, Elektromehanika, Kranj
- S. Saksida, Institut za sociologijo in filozofijo pri
Univerzi v Ljubljani
- J. Virant, Fakulteta za elektrotehniko, Univerza v
Ljubljani

Uredništvo in uprava: 61000 Ljubljana, Institut "Jožef
Stefan", Jamova 39, telefon (061) 63 261, 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: 50101-678-51841

Stališče uredništva se lahko razlikuje od mnenja avtorjev.

Na podlagi mnenja Republiškega sekretariata za prosveto
in kulturo št. 4210-151/77 z dne 4.5.1977, je časopis
INFORMATICA strokovni časopis, ki je oproščen temelj-
nega davka od prometa proizvodov.

Tisk: Tiskarna Kresija, Ljubljana

Grafična oprema: Tihomir Simončič

letnik 1. 1977 - št. 4.

VSEBINA

- R. Tomović 3 Sistemski prilaz razvoju i projekto-
vanju gradskih informacionih sistema
- A.P. Železnikar 9 Razvoj dinamičnih pomnilnikov za
M. Kovačević mikro računalnike
D. Novak
- J. Zupan 22 Kriptografija na prelomu
- S. Nikolić 29 Jedan postupak za mikroprocesorsko
Ž. Tošić ostvarenje prekidačkih funkcija
- V. Zgaga 32 TV - Terminal za mikroročunalno
- D.V. Glušac 37 Neki aspekti uvođenja komunikacione
mreže on-line real-time sistema
Beogradske banke
- N. Papić 42 Univerzalni serijsko vhodno/izhodni
M. Kovačević modul
- V. Batagelj 48 Kako narišemo pravilni mnogokotnik?
- O.B. Popov 53 Procesor za aritmetično-logični
izrazi
- L.N. Djordjević 60 On N-th Root Evaluation by Iterative
Methods with Parameters
- M. Kovačević 63 BUS Standardi
- I. Bratko 66 Drugo svetovno šahovsko prvenstvo
računalnikov
- 68 Tendence v razvoju mikro procesorjev
- 68 Novice in zanimivosti
- 73 Literatura in srečanja
- 76 Avtorji

INFORMATICA

Journal of Computing and Informatics

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

EDITORIAL BOARD:

T. Aleksić, Beograd, D. Bitrakov, Skopje, P. Dragojlović, Reka, S. Hodžar, Ljubljana, B. Horvat, Maribor, A. Mandžić, Sarajevo, S. Mihalić, Varaždin, S. Turk, Zagreb.

EDITOR-IN-CHIEF :

A. P. Železnikar

TECHNICAL DEPARTMENTS EDITORS :

V. Batagelj - Programming
I. Bratko - Artificial Intelligence
D. Čeček-Kecmanović - Information Systems
M. Exel - Operating Systems
A. Jerman-Blažič - Publishers News
B. Jerman-Blažič-Džonova - Literature and Meetings
L. Lenart - Process Informatics
D. Novak - Microcomputers
N. Papić - Student Matters
L. Pipan - Terminology
B. Popovič - News
V. Rajkovič - Education
M. Špegel, M. Vukobratović - Robotics
P. Tancig - Computing in Humanities and Social Sciences
S. Turk - Hardware

EXECUTIVE EDITOR :

R. Murn

PUBLISHING COUNCIL

T. Banovec, Zavod SR Slovenije za družbeno planiranje, Ljubljana
A. Jerman-Blažič, Slovensko društvo Informatika, Ljubljana
B. Klemenčič, ISKRA Elektromehanika, Kranj
S. Saksida, Institut za sociologijo in filozofijo pri Univerzi v Ljubljani
J. Virant, Fakulteta za elektrotehniko, Univerza v Ljubljani

Headquarters : 61000 Ljubljana, Institut "Jožef Stefan", Jamova 39, Phone: (061)63 261, Cable: JOSTIN Ljubljana, Telex:31 269 YU JOSTIN

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

Opinions expressed in the contributions are not necessarily shared by the Editorial Board.

Printed by: Tiskarna Kresija, Ljubljana

DESIGN: Tihomir Simončič

volume 1. 1977 - n° 4.

CONTENTS

R. Tomović	3	A System Approach to the Development and Design of Town Information Systems
A.P. Železnikar M. Kovačević D. Novak	9	Development of Dynamic Memory Modules for Microcomputer
J. Zupan	22	New Trends in Cryptography
S. Nikolić Ž. Tošić	29	A Process of Switching Functions Implementation by Microprocessors
V. Zgaga	32	TV-Terminal for a Microcomputer
D.V. Glušac	37	Some Aspects of Communication Network Design for On-line Real-Time System of Beogradska banka
N. Papić M. Kovačević	42	Universal, serial INPUT/OUTPUT Module
V. Batagelj	48	How to Draw a Regular Polygon?
O.B. Popov	43	Processor for Arithmetical and Logical Expressions
L.N. Djordjević	60	On N-th Root Evaluation by Iterative Methods with Parameters
M. Kovačević	63	BUS Standards
I. Bratko	66	Second Computer Chess Championship
	68	Micro News
	68	News
	73	Literature and Meetings
	76	Contributors

systemski prilaz razvoju i projektovanju gradskih informacionih sistema

r. tomović

UDK 681.3:659.21

Elektrotehnički fakultet Beograd

U djelu se nakazuje na osnovne efekte uvođenja upravljačkih informacionih sistema, naročito u odnosu na građane. Time počinje rasprava o konačnom domašaju razvoja upravljačkog informacionog sistema, koji znatno nadmašuje neposredne efekte uvođenja ovog sistema na prava građanja. Radi se o ukupnim promjenama u društveno-političkom sistemu grada, gde izgradnja informacionog sistema proširuje informacionu podlogu za procese delegatskog odlučivanja.

A SYSTEM APPROACH TO THE DEVELOPMENT AND DESIGN OF TOWN INFORMATION SYSTEMS. This paper deals with basic effects of management information system introduction (encompassing) the town population. This is a start-point of a discussion of the final goal of the information system development which drastically exceeds the effects of this system implementation with respect to people's rights. The development of the information system extends the information basis for the process of the delegate's decision-making leading to overall changes in the socio-political system of the town.

1. I Uvod

Gradski informacioni sistemi odnose se na organizacije sa upravljanjem na više nivoa.

Kao što je poznato, formalna, odnosno matematička teorija sistema nije uspjela da razvije efikasne postupke za projektovanje upravljačkih informacionih sistema u složenim organizacijama. Osim toga, i mnoga druga značajna pitanja upravljanja složenim sistemima, kao što su dekompozicija, agregacija i optimizacija – još su nerazrešena. O razlozima takvog stanja, pisano je u zasebnom radu [1]. U ovom tekstu korišćene su odatle samo definicije pojmova, kao što su ulaz, izlaz, sistem, podsystem, optimalna odluka i drugo. U aneksu A date su definicije osnovnih pojmova formalne teorije sistema koji se koriste u tekstu. Ova studija ima u tom pogledu strogu ograničenu namenu. Ona ne razmatra problem ni sa pozicija banki podataka, ni jedinstvenosti informacionog sistema, ni organizacije računarske mreže.

Cilj studije je da ispita vrednost formalne teorije sistema u vezi sa razvojem i projektovanjem informacionih sistema za upravljanje gradovima. Pri tom se pojmovima sistem i upravljanje daje tačno određeno značenje.

Imajući sve ovo u vidu, logično se postavlja nekoliko pitanja. Pre svega, ako je tačna teza da su formalne metode prilikom proučavanja složenih sistema ograničenog dometa, treba na praktičnom primeru pokazati koliki je stvarni doprinos formalne

teorije upravljanja u slučaju koji se ovde razmatra. Ako ovaj put ne pruža puni odgovor, treba ukazati koji se drugi naučni prilaz može primeniti kako bi se zadatak uspešno rešio.

Rešenje zadatka treba, po mišljenju autora, tražiti u konsekventnom multidisciplinarnom načinu projektovanja i realizacije informacionog sistema. Pored teorije sistema i informatike, u pomoć se moraju pozvati i određene grane pravnih i organizacionih nauka, zatim operaciona istraživanja i pojedine oblasti ekonomskih nauka.

Osnovni zadatak ovog rada je da na primeru projektovanja i realizacije upravljačkog informacionog sistema (UIS), za grad Beograd, pokaže stvarnu upotrebljivost systemske analize pri proučavanju složenih organizacija. Pored toga, istaci će se i značaj multidisciplinarnog rešavanja zadataka ove vrste.

Termin upravljački informacioni sistem koristi se radi izbegavanja eventualnih nesporazuma. Ako bi se upotrebio samo izraz informacioni sistem, moglo bi da se shvati da se radi o automatizaciji administrativnih poslova (platni spiskovi, knjigovodstvo, bankarske operacije). Naravno i automatizacija poslova ove vrste sačinjava deo razvoja gradskih informacionih sistema. Međutim, baš stavljanje naglaska na upravljački informacioni sistem pokazuje da je primena automatizacije, o kojoj je u ovom tekstu reč, okrenuta modernizaciji rada uprave i povećanju uticaja građana i ostalih subjekata na donošenje odluka. Automatizacija administrativnih poslova samo je deo opšte modernizacije uprave, koja treba da se uklopi u širu realizaciju ciljeva društveno-političkog karaktera. Istina, termin informacioni sistem sve se više koristi u širem smislu, uključujući i upravljanje, tako da će vremenom nestati mogućnosti pogrešne interpretacije.

Na samom početku, treba ukazati na još nekoliko principijelnih pitanja u vezi s razvojem upravljačkih informacionih sistema koja bi se mogla interpretirati na razne načine.

Pre svega, sam pojam gradske uprave shvata se često vrlo usko. Funkcije uprave ograničavaju se ponekad na rad gradske administracije, što očigledno nije tačno. Važna funkcija uprave je operativno upravljanje svim vidovima gradskog života, kao

* Knjigu "Upravljački informacioni sistem Beograda" dr. Rajka Tomovića je, u okviru biblioteke "Masovne komunikacije i društvo" i edicije "Informatika i javna uprava", izdalo štampalo štamparsko-izdavačko preduzeće GLAS iz Beograda, juna 1977 godine. Zanimljiva i aktualna tematika te način obrade problema dali su povoda da uz dozvolu pisca i izdavača objavimo prvi dio knjige pod naslovom "Systemski prilaz razvoju i projektovanju gradskih informacionih sistema".

što su saobraćaj, zdravstvo, obrazovanje, komunalne službe i drugo. I svi vidovi planiranja razvoja grada spadaju u domen lokalne uprave. Najzad, društveni razvoj i političko odlučivanje suštinska su potreba svake gradske zajednice.

Jasno je da obim i struktura gradske uprave zavise prvenstveno od društvenog i državnog sistema čiji je deo gradska zajednica. U zavisnosti od ovog okruženja dolaze do većeg ili manjeg izražaja pomenute komponente gradske uprave.

U slučaju Beograda grad je deo samoupravnog socijalističkog društvenog i državnog uređenja tako da ima jedinstvene karakteristike u odnosu na gradsku upravu u drugim zemljama. Stoga su iskustva dobijena razvojem upravljačkog informacionog sistema Beograda u mnogo čemu specifična. O tome će biti reči u drugom delu ovog rada. U ovom, prvom, istaći će se samo oni aspekti razvoja gradskih informacionih sistema koji bi mogli da budu od opšteg interesa.

Ugao gledanja u ovom radu razlikuje se od čisto inženjerskih prilaza. Polazna misao je da se u razvoju upravljačkih informacionih sistema, u stvari, radi o automatizaciji specifične prirode, s tim što u ovom slučaju objekat nije proizvodni proces već gradska organizacija, a dinamika objekta ne opisuje se jedinačama stanja već informacionom bazom. Osim toga, ciljevi automatizacije u gradu, nisu tehničko-ekonomskog, već prvenstveno društveno-političkog karaktera. Ovakvo gledište provlači se kroz ceo rad, a ovdje se posebno ističe zbog jednog pitanja o kome se neposredno diskutuje kad je reč o razvoju automatizovanih informacionih sistema. Radi se, naime, o prednostima i nedostacima tzv. jedinstvenih informacionih sistema.

Problem jedinstvenog informacionog sistema^o nametnut je i izveden, posebno u našim uslovima. Naime, ako se polazi od činjenice da informacioni sistem sam za sebe ne znači ništa već da je u službi opštih ciljeva organizacije čije se poslovanje, automatizuje, za izbor njegove koncepcije i organizacije odlučujuće je da li je način upravljanja centralizovan, sa hijerarhijom odlučivanja, ili je, pak, samoupravnog socijalističkog tipa, kao u našem gradu. Prema tome, društveni karakter upravljanja gradom opredeljuje stepen centralizacije ili decentralizacije informacionog sistema.

To nikako ne znači da automatizacija ne menja strukturu postojeće neautomatizovane ili parcijalno automatizovane baze podataka u smislu njene integracije^o. Kao što kompleksna automatizacija u industriji uspostavlja nove veze unutar krupnih proizvodnih potencijala tako i intenzivna automatizacija obrade podataka bitno menja organizaciju baze podataka.

Prema tome, izraz jedinstveni informacioni sistem, u našem slučaju, ne može nikako značiti *centralizaciju upravljanja i baze podataka*. Sa gledišta korisnika, jedinstvenost informacionog sistema znači:

- a) jedinstvenost ažuriranja – svaki novogenerisani podatak ulazi samo na jednom mestu u informacioni sistem i automatski se raspoređuje u sve relevantne lokacije;
- b) pristup svih (ovlašćenih) korisnika izabranim podacima, nezavisno od resorskih i drugih ograničenja;
- c) cirkulaciju podataka kroz ceo informacioni sistem, nezavisno od stepena automatizacije;
- d) znatnu redukciju redundantnosti podataka, što omogućuje kompleksna automatizacija.

Drugim rečima, ako priroda to zahteva, korisnici ne smeju osetiti resorske barijere niti podelu informacione baze na automatizovani i neautomatizovani deo.

Pitanje jedinstvenosti informacionog sistema pojavljuje se sa gledišta organizacije koja uvodi automatizaciju još u jednom vidu. Naime, u jednom trenutku razvoja informacione baze složene organizacije, potrebno je celovito i na duži period sagledati sve efekte automatizacije, odnosno promene tehnike i tehnologije obrade podataka. Zahtev je, svakako, potpuno opravdan, jer inače dolazi do nepredviđenih teškoća i neefikasnog

rada prilikom svakog sledećeg koraka ka kompleksoj automatizaciji poslovanja.

Interpretacija jedinstvenosti informacionog sistema u smislu centralizacije baze podataka nekorektno je i sa naučne tačke gledišta. Naime, sam pojam sistem, pa prema tome i informacioni sistem, znači da se radi o međusobno povezanim objektima. Naglašavanje jedinstvenosti informacionog sistema ima zato smisla samo u odnosu na agregaciju relativno nezavisnih i redundantnih baza podataka u složeni objekat.

Problem je i određivanje podsystematske strukture novog, manje ili više, utomatizovanog složenog sistema. Da li će se novoformirani objekat organizovati bez podsystema ili sa više njih, stvar je dekompozicije, koja je poseban problem svakog složenog sistema.

Da bi se izbegli nesporazumi, možda je bolje koristiti termin integrirani ili povezani informacioni sistem umesto jedinstveni. Time se želi naglasiti da se sa izolovanih baza podataka prešlo na agrerirani sistem u smislu navedenih tačaka. U ovom tekstu koristiće se prosto izraz informacioni sistem u obzir da se radi o upravljanju velikim gradom koji ima sve karakteristike složene organizacije.

1. 2 Informacioni sistem

Rukovodstvo složene organizacije koje donosi odluku o uvođenju automatizacije mora da se opredeli između nekoliko mogućnosti. Neophodno je razjasniti da li će se automatizovati samo pojedine aktivnosti složene organizacije ili je dugoročni cilj modernizacija organizacije u celini. Pri tom se ne misli da se sve aktivnosti složene organizacije moraju automatizovati, već je reč o nečem drugom. Suština sistemskog prilaza je u tome što se prelaz na automatizaciju poslovanja mora ispitati u odnosu na sve sektore aktivnosti pojedinačno i na performanse objekta u celini.

Sasvim je drugo pitanje treba li mašinske metode primeniti na sve aktivnosti. Suština nije, dakle, u forsiranju mašinske obrade, po svaku cenu, već u integralnom sagledavanju efekata automatizacije na rad složenog sistema kao celine.

Kada se prihvati orijentacija na sistemsko sagledavanje efekata automatizacije, pojavljuje se niz mogućnosti između kojih treba izvršiti izbor.

Jedan od mogućih puteva za izgradnju informacionog sistema je metod parcijalne automatizacije. U ovom slučaju prioritet prilikom automatizacije dobijaju one aktivnosti složene organizacije koje, prema određenoj proceni organa upravljanja, predstavljaju uska grla ili su od posebnog značaja. Na primer, pored aktivnosti vezanih za masovnu obradu podataka, kao što je evidencija pebivališta građana, u gradskoj upravi često se počinje sa parcijalnom automatizacijom službi prihoda, komunalnih usluga i sličnog. Evolucija zadatka složene organizacije i proširenje obima njene aktivnosti dovode, pre ili posle, do zahteva za integracijom parcijalno automatizovanih sistema u celinu višeg reda. Za rešavanje ovog problema koriste se prethodno iskustvo i neposredni zahtevi za povezivanjem parcijalno automatizovanih sistema. Po pravilu, integracija parcijalno automatizovanih sistema zahteva, pored dodatnih projektantskih napora, i znatna ulaganja u novu opremu. Ovakav proces integracije se, očevidno, može nastavljati do željenog nivoa.

Drugi metodološki pristup realizaciji (integriranog) informacionog sistema oslanja se na agregaciju. U ovom slučaju se ne polazi od parcijalne automatizacije određenih aktivnosti već kao podloga procesa automatizacije služi vertikalna dekompozicija složene organizacije. Kao što je poznato, upravljački sistemi sa više nivoa uvek se konstituišu na principu piramide. Naime, obilje informacija o stanju složene organizacije dovodi do zahteva za integracijom parcijalno automatizovanih sistema. Radi prevazilaženja informacione barijere koristi se organizaciju upravljanja sa više nivoa. Na taj način, stanje realnog sistema opisuje se na svakom višem nivou sve manjim brojem sintetičkih podataka, a broj upravljačkih punktova, idući naviše, smanjuje se.

Imajući sve ovo u vidu automatizacija složene organizacije može početi od najnižih i najbrojnijih organizacionih jedinica i retirati ih kao podsysteme sa svojom parcijalnom bazom pod-

^o Termin „jedinstveni informacioni sistem“ koristi se u stručnoj literaturi i u sledećem smislu: formiranje zajedničkog informacionog sistema za sve aktivnosti organizacije, bez obzira na sektore rada i njenu strukturu. Ovdje se misli na ovu interpretaciju.

^o Termini informaciona baza i baza podataka koriste se kao ravnopravni. Odnose se na sve registrovane podatke, sa ručnim ili automatizovanim pristupom, koje gradska uprava koristi za donošenje odluka.

taka, aktivnostima, ciljevima. Naravno, pri tom se moraju imati u vidu i eksterni, horizontalni i vertikalni ulazi i izlazi.

Proces integracije ovakvih podсистema u više celine poznat je kao agregacija. Treba naglasiti da se zbog hitnosti posla automatizacija najčešće ne prošire kao talas odozdo nagore, već nezavisno počinje na nekoliko nivoa. Integracija metodom agregacije zahteva u jednoj od kasnijih faza realizacije upravljačkog informacionih sistema izgradnju telekomunikaciono-računarske mreže, ukoliko su podsystemi teritorijalno razmešteni, što predstavlja poseban istraživačko-projektni zadatak.

Informacioni sistem može se od početka projektovati tako što se kao objekat automatizacije uzima složena organizacija u celini. Ovakav način projektovanja vuče koren iz teorije upravljanja dinamičkim sistemima. U daljem tekstu biće ukazano na prednosti ovakvog pristupa razvoju upravljačkih i informacionih sistema. Naravno, ni on se ne može primeniti u čistom vidu jer zahteva dugotrajne i obimne analize poslovanja složene organizacije, pa se obično kombinuje sa drugim metodama.

1. 3 Razvoj i projektovanje upravljačkih informacionih sistema

Izraz projektovanje upravljačkih informacionih sistema treba ovde prihvatiti sa određenom rezervom. U tehničkim naukama on ima precizno značenje kako u pogledu načina rešavanja postavljenog zadatka tako i u pogledu njegovog sadržaja. Već je rečeno da se kod upravljačkih informacionih sistema samo delimično radi o problemu tehničke prirode. Zadatak upravljanja i modernizacije ljudskih organizacija zadire u mnoge druge naučne discipline, uključujući, velikim delom, društvene i političke aspekte. Stoga, metodi izgradnje tehničkih objekata mogu samo delimično da posluže za razvoj upravljačkih informacionih sistema. To objašnjava i zašto je naslov ovog odeljka »razvoj i projektovanje upravljačkih informacionih sistema«.

Iz ovog proizlazi da nije moguće propisati krute, detaljne šeme za razvoj ovih sistema. Međutim, izvesna opšta pitanja i dileme i određeni metodološki i projektni postupci mogu se prepoznati kao zajednički faktor u mnogim zadacima ove vrste. To pogotovo važi ako se polje posmatranja suzi na uže klase zadataka, kao što su gradski informacioni sistemi.

U tom smislu učinjen je napor da se zajednička pitanja razvoja i realizacije upravljačkih informacionih sistema prikažu na jednom mestu, bez pretenzije na generalno i definitivno rešenje. Pri tom su kao zajednička tačka posmatranja poslužili neki osnovni pojmovi teorije sistema. Na osnovu toga se mogu izdvojiti i neki opšti problemi vezani za razvoj i realizaciju upravljačkih informacionih sistema.

a) Ciljevi modernizacije složene organizacije

Ovo je izuzetno odgovoran posao jer zahteva eksplicitno i precizno opisivanje kratkoročnih i dugoročnih ciljeva modernizacije. Pri tom treba posebno voditi računa da se u obzir uzmu kako interni tako i eksterni ciljevi modernizacije javne uprave. Naime, svaka služba ili organ uprave koji nabavlja opremu za automatizaciju postavlja »sopstvene« ciljeve modernizacije, ne vodeći uvek dovoljno računa kako će se to odraziti na druge delove sistema ili na njegove krajnje korisnike. Na primer, sasvim je moguće automatizovati administraciju zdravstvene ustanove, a da se to ne odrazi, znatnije, na vreme čekanja pacijenata i kvalitet lečenja. Stoga se u fazi definisanja ciljeva modernizacije mora podjednako voditi računa kako o internim tako i o eksternim ciljevima poboljšano načina rada.

Mada svi ovi stavovi izgledaju evidentni, mogu se naći brojni primeri automatizacije javne uprave koji su doneli minimalne koristi krajnjim korisnicima iako oni, u osnovi, finansiraju sve ove investicije.

Podelu ciljeva modernizacije na interne i eksterne treba fleksibilno shvatiti. Već je naglašeno da uprava ima veoma složene zadatke, počev od administracije do političkog informisanja građana i njihovih predstavnika. Ukoliko uprava izražava stvarni interes građana, utoliko pre, svaka njena modernizacija donosi zajedničke koristi svima – i internim i eksternim subjektima.

Već je naglašeno da se ni modernizacija uprave ne može smatrati cijem razvoja upravljačkih informacionih sistema. Polaznu tačku treba tražiti u društveno-političkim ciljevima kojima

treba da služi automatizacija. U tom pogledu nalazimo se u jedinstvenoj situaciji jer razvoj upravljačkog informacionog sistema Beograda treba, ustvari, da se ugradi u potpuno novu organizaciju gradske uprave, u kojoj ne postoje subordinirani nivoi upravljanja u klasičnom smislu.

Očevidno je da se prva iskustva o funkcionisanju upravljačkih informacionih sistema, u uslovima samoupravne transformacije uprave, ne mogu dobiti na drugim stranama, pa je, stoga, potrebno pažljivo pratiti rezultate razvoja gradskih informacionih sistema u našoj zemlji. Ovaj tekst pokušava upravo da osvetli neke fundamentalne probleme razvoja upravljačkih informacionih sistema u našim uslovima.

b) Adaptacija organizacije novom načinu rada

Na osnovu postavljenih ciljeva modernizacije treba pristupiti utvrđivanju programa organizacionih izmena i promena načina rada složenog sistema da bi se postigli željeni efekti. Obim izmena ide od proceduralnih pitanja, formata i sadržaja dokumenata do formiranja novih organizacionih jedinica i donošenja novih propisa, pa i zakonskih odluka. Posebna pažnja mora se posvetiti dodatnom obrazovanju, inovaciji znanja i novim kadrovskim profilima koji će eksploatisati i usavršavati upravljački informacioni sistem, jer se ovaj poduhvat ne može ograničiti samo na tehničke stručnjake.

c) Identifikacija postojećih informacione baze

Identifikacija postojećih informacione baze složene organizacije, uključujući naravno dinamiku i može da se po važnosti izjednači sa identifikacijom vektora stanja i jedinačnima stanja dinamičkih objekata u formalnoj teoriji upravljanja. Sa gledišta krajnje funkcije, informaciona baza složene organizacije, ima iste bitne funkcije kao i modeli dinamičkih objekata u prostoru stanja. Ona opisuje aktuelno stanje i dinamiku složene organizacije. Na osnovu podataka sadržanih u bazi podataka donose se upravljačke odluke i obavljaju aktivnosti složene organizacije.

Postupak identifikacije kompletne postojećih informacione baze ima specifičnosti koje zavise od prirode složene organizacije (gradska uprava, zdravstvena ustanova, robna kuća i drugo). Međutim, postoje u svim slučajevima i bitne zajedničke crte koje mogu poslužiti kao podloga jedne opštije metodologije.

Identifikacijom informacione baze složene organizacije treba da se dođe do sledećih elemenata:

1) Popis postojećih datoteka – Pojam datoteka u ovom tekstu koristi se u širem značenju. On označava svaki blok informacija smešten na mediju sa ručnim ili automatskim pristupom koji služi za realizaciju određenih aktivnosti složene organizacije. Drugim rečima, pomoću datoteke se razmenjuju ulazni i izlazni podaci. Ona ima svoj unutrašnji format smeštaja podataka i koristi se za obavljanje određenih aktivnosti složene organizacije. Ovakva interpretacija pojma datoteke odgovara opštim potrebama identifikacije informacione baze, jer na ovom nivou projektovanja nije uvek neophodno praviti razliku između termina kakvi su kartoteka, evidencija, popis i slično.

2) Identifikacija datoteka – Zadatak identifikacije datoteke je da formira kompletan skup deskriptivnih i numeričkih podataka potrebnih za:

- prelaz na automatsku obradu,
- povezivanje sa drugim delovima informacione baze i
- povezivanje sa relevantnim upravnim poslovima i zadacima.

Sa sistemске tačke gledišta, neophodno je odrediti opšti model i postupak koji se može primeniti za identifikaciju svih datoteka složene organizacije, bez obzira na njihov tip i namenu. Postupak identifikacije datoteka uslovljen je, pri tom, svrhom informacionog sistema. Ako se radi o administrativnoj obradi podataka i o automatizaciji izolovane aktivnosti složene organizacije, glavni identifikacioni podaci odnose se na samu datoteku. Prilikom razvoja upravljačkog informacionog sistema identifikacioni postupak mora da, pored datoteke, obuhvati i skup aktivnosti organizacije koje se u potpunosti ili delimično oslanjaju na posmatranu datoteku.

U toku projektovanja upravljačkog informacionog sistema Beograda, razvijen je sopstveni postupak identifikacije datoteke, koji će biti prikazan u drugom delu rada.

3) Tipovi datoteka – Osnovna karakteristika neintegrisane informacione baze je što se ona formira, ažurira i koristi u lokalnim (resorskim) okvirima. Drugim rečima, svaki subjekat određenog podskupa aktivnosti složene organizacije operiše in-

formacionim blokom u sopstvenoj nadležnosti, ne vodeći računa o stanju istih ili sličnih podataka u okviru cele složene organizacije. U takvim uslovima jedni isti podaci pa čak i čitave datoteke ponavljaju se više puta. Stoga je prilikom analize postojeće informacione baze nužno voditi računa o primarnim i izvedenim datotekama. Utvrđivanjem karaktera datoteka može se razlučiti osnovna neredundantna baza podataka od redundantnih i izvedenih podataka. Ovo je važan parametar za projektovanje informacionog sistema jer se izvedeni podaci najčešće ne moraju fizički generisati, već se to radi programskim paketima. Naravno, ovo je samo prvi korak ka realizaciji automatizovane informacione baze. U celini ona može da se isprojektuje jedino u funkciji zahteva korisnika.

d) Projektovanje banke podataka

Termin banka podataka koristiće se za označavanje delova informacione baze sa automatskim pristupom i automatskom obradom. Ako je reč o bankama višeg reda, sa izvedenim podacima, atribut »automatski« pristup nije neophodan.

Bitan predušlov za projektovanje banke podataka je poznavanje stanja, obima i strukture postojeće, polazne informacione baze. Polazna baza podataka je često neautomatizovana, sa izolovanim redundantnim delovima. Međutim, ona, takođe, može da već delom bude automatizovana tako da termin postojeća (polazna) informaciona baza ne treba izjednačiti sa nulnim ili niskim nivoom automatizacije.

Drugi važan prethodni elemenat za projektovanje banke podataka je poznavanje aktivnosti složene organizacije i zahteva korisnika. U stvari, projektanti banke podataka, bez obzira što je njihov finalni proizvod inženjerske prirode (oprema, programski paketi), moraju da budu detaljno upoznati sa ciljevima modernizacije i novim kriterijumima optimiziranja aktivnosti složene organizacije, inače može doći do ozbiljnog nesklada između tehničkih potencijala i načina rada složene organizacije, što značajno može da smanji opšti efekat automatizacije. Stoga je neophodno, u ovoj fazi projektovanja upravljačkih informacionih sistema, stalno paziti da se obezbedi puna saradnja stručnjaka iz oblasti informatike, pravnih i organizacionih nauka, operacionih istraživanja i drugih disciplina.

Između ostalog, u glavnom projektu banke podataka treba da budu zastupljeni sledeći elementi;

- *Osnovni parametri opreme za smeštaj i obradu podataka* - Izraz osnovni parametri znači da se na osnovu fiksiranih karakteristika informacione baze može prići traženju ponude za kupovinu opreme.

- *Osnovni parametri za realizaciju distribuiranog informacionog sistema* (terminalni, punktovi decentralizovane obrade podataka, telekomunikaciona mreža). - Pri današnjem stanju računarske tehnologije jedva da može biti reči o strogo centralizovanim informacionim sistemima složenih organizacija. Ovakva rešenja posebno su neadekvatna u našim društvenim uslovima gde odlučivanje polazi od neposrednih proizvođača i kreće se ka udruženom radu.

- *Programski deo*. - Pod ovim terminom podrazumeva se čitav sistem programskih paketa u više nivoa koji omogućavaju smeštaj, ažuriranje, proveru tačnosti podataka i vezu između fizičke i logičke, odnosno potrebne organizacije podataka. Kao i u slučaju opreme, u ovom delu projekta, definišu se osnovni parametri za nabavku sistema programske podrške koja realizuje postavljene zahteve korisnika banke podataka.

- *Propisivanje ulazno-izlaznih dokumenata* - Terminalni (ulazno-izlazni) punktovi upravljačkog informacionog sistema predstavljaju, kao i kod dinamičkih sistema upravljanja, bitne elemente njihove definicije. Kod organizacionih sistema ulazi i izlazi fizički se realizuju preko sadržaja i formata odgovarajućih dokumenata, poruka, instrukcija. S obzirom na karakter javne uprave, opštenje sa korisnicima je, po pravilu, u obliku fiksnog tipa dokumenata. Iz ovih razloga, neophodno je, u glavnom projektu banke podataka precizirati parametre ulazno-izlaznih dokumenata. Time se u velikoj meri stvara podloga za modernizaciju rada. Međutim, treba jasno reći da se samim projektom i usvajanjem zahteva korisnika ni izdaleka ne iscrpljuje *problem modernizacije i poboljšanja rada uprave*. Ovaj deo posla treba da prate operaciona istraživanja sadašnje i nove automatizovane tehnologije rada, nove sheme obavljanja aktivnosti složene organizacije, odgovarajući interni propisi i administrativne i zakonske promene. To je vrlo složen posao, koji po obimu ne zaoštuje iza projekta banke podataka u užem smislu.

e) Analiza efikasnosti ulaganja

Prilikom projektovanja upravljačkih informacionih sistema postoji opasna tendencija da se između informacionog sistema i potpune automatizacije stavi znak jednakosti. U tom pogledu nije naodmet imati stalno pred očima efekte automatizacije u industriji. Pored svih tehničkih dostignuća, kompletna automatizacija proizvodnje, uključujući procesne računare, ne odvija se bez teškoća. Visokoautomatizovana proizvodnja postavlja vrlo oštre zahteve počev od kvaliteta sirovina, ažurnosti kooperacije, transporta, do kompletnog ambijenta u kome je fabrika locirana.

Slični problemi javljaju se i pri visokoj automatizaciji upravljačkih informacionih sistema. Iz tih razloga, moraju se uvek temeljno ispitati zahtevi korisnika u svetlosti investicionih i eksploatacionih troškova. Kao i svi projektantski poslovi, tako i ovaj zadatak podrazumeva savesno ispitivanje raznih alternativa, počev od maskimalnih pa do minimalnih zahteva korisnika. Na osnovu ovakvih analiza treba u funkciji opštih ciljeva modernizacije složene organizacije doneti konačnu odluku o stepenu automatizacije odgovarajućih aktivnosti.

f) Veza između automatizovanog i neautomatizovanog dela informacione baze

Imajući u vidu da se ne može očekivati da će upravljački informacioni sistem uvek biti kompletno automatizovan, u projektu banke podataka mora se predvideti postupak kojim će se obezbediti veza između automatizovanog i neautomatizovanog dela baze podataka. Ako se tako ne postupi, preći opasnost da se, sa gledišta korisnika, informacioni sistem raspadne na dva dela, bez mogućnosti efikasne interakcije.

Veze između automatizovanog i neautomatizovanog dela informacionog sistema rešava se na specifičan način u svakom datom slučaju. O tome će biti više reči prilikom opisivanja Upravljačkog informacionog sistema Skupštine grada Beograda. Stoga će ovde biti navedeni samo najopštiji mehanizmi pomoću kojih se obezbeđuje efikasna interakcija automatizovanih i neautomatizovanih delova informacione baze. U tu svrhu koristi se adekvatno razvijeni sistem šifara, kao, na primer, matični broj građanina. Važnu ulogu ima detaljno isplanirana Metodoteka, Azbučni katalog pojmova i Nomenklatura upravnih poslova i zadataka (odeljak 2. 3, drugi deo rada).

Naravno, problemi prelaza podataka kroz graničnu liniju automatizovanog i neautomatizovanog dela informacione baze moraju se, takođe, rešavati organizacionim merama i precizno utvrđenim proceduralnim shemama.

1. 4 Opšti problemi

U prethodnom tekstu ukazano je na okvire sistemskog prilaza razvoju upravljačkog informacionog sistema. Na jednom polu procesa počinje se postavljanjem ciljeva modernizacije i optimizacije složene organizacije, a na drugom kraju završava se sa projektom banke podataka, organizacionim rešenjima, operacionim istraživanjima i analizom efikasnosti ulaganja.

Deoba procesa razvoja i projektovanja upravljačkog informacionog sistema na faze ne bi smela nijednog trenutka da dovede do zabune. Naime, podela na faze uslovnog je karaktera. U praksi se projektovanje i realizacija upravljačkog informacionog sistema odvijaju u velikoj meri paralelno, sa fazama koje se preklapaju. U ovom pogledu se izgradnja upravljačkog informacionog sistema bitno razlikuje od konstrukcija tehničkih objekata gde upravo sekvencijalni karakter projektnog i izvođačkog procesa pruža garanciju za kvalitetnu i pravovremenu realizaciju.

Sistemski prilaz razvoju upravljačkog informacionog sistema otvara i neka druga pitanja teorijskog i praktičnog karaktera. Naime, u dosadašnjem ulaganju implicitno je pretpostavljeno da je upravljanje složenom organizacijom ekvivalentno upravljanju izolovanim objektom. Već u ovoj osnovnoj pretpostavci krije se opasnost od ozbiljnih promašaja, jer se u formalnoj teoriji objekat, definisan kao sistem, posmatra kao izolovano celina.

Ako između više realnih objekata postoje značajni interakcioni članovi, oni se posmatraju kao sistem sastavljen od više pod sistema. Tako se dolazi do modela složenog sistema, sa ulazno-izlaznim parovima bez interakcionih članova prema okolini. Međutim, u kojim uslovima se jedan objekat može tretirati kao izolovana celina, a kada ova pretpostavka nije prihvatljiva, ostaje otvoreno pitanje u teoriji sistema i mora se rešavati pojedinačno, na početku razvoja upravljačkog informacionog sistema.

Prilikom preliminarnih razmatranja o razvoju upravljačkog informacionog sistema za grad, polaznu tačku za primenu sistemske analize treba tražiti u stepenu decentralizacije javne uprave. Ukoliko se odlučivanje o nizu funkcija javne uprave (planiranje, obrazovanje, zdravstvo, finansije i drugo) obavlja na lokalnom nivou, ima smisla tretirati grad kao sistem, a ne kao pod sistem centralizovane složene organizacije. U takvim uslovima vrednost sistemske analize dolazi do punog izražaja.

Važno fundamentalno pitanje svake teorije složenih sistema je dekompozicija na pod sisteme. Veliki grad se razbija na pod sisteme, na manje ili veće urbane, administrativne i upravne celine. Stepenn decentralizacije vlasti zavisi od društvenog sistema u celini, tako da se problem dekompozicije mora posmatrati u tom svetlu. Sa druge strane, srodne aktivnosti javne uprave (unutrašnji poslovi, finansije, komunalne službe i drugo) grupišu se po sektorima. Ovo takođe predstavlja specifičan vid dekompozicije složenog sistema po vrsti delatnosti. U našem smislu termin dekompozicija se odnosi na razlaganje samog upravljačkog informacionog sistema na pod sisteme i njihovo ukupanje u postojeću strukturu lokalne uprave. Pošto je dekompozicija lokalne uprave već uslovljena društveno-organizacionim i drugim faktorima, ovde će isključivo biti reči o problemu dekompozicije upravljačkog informacionog sistema unutar zadate strukture gradske uprave.

Dekompozicija upravljačkog informacionog sistema izuzetno je težak zadatak. Sa gledišta teorije složenih sistema bilo bi nužno da se ustanove opšti kriterijumi i postupci dekompozicije informacionih sistema tako da se naučno mogu praviti komparacije između različitih načina dekompozicije u okviru zadatih ograničenja. Zbog složene prirode urbanih organizacija, nema teorijskih rešenja dekompozicije upravljačkog informacionog sistema na pod sisteme. Projektantima upravljačkog informacionog sistema ostaje da zadatak rešavaju multidisciplinarno i heuristički, a da usvojene varijante proveravaju u praksi.

Navešćemo neke vrlo opšte kriterijume za dekompoziciju upravljačkog informacionog sistema. Jedno od mogućih rešenja je tzv. *sektorska dekompozicija*, prema prirodi aktivnosti organa uprave (unutrašnji poslovi, obrazovanje, zdravstvo, zaštita okoline, komunalne službe itd.). Dosljedna primena ovog kriterijuma ima više principijelnih nedostataka. Usled većeg broja pod sistema (reda 10 do 12) pojavljuju se mnogostruki redundantni podaci, komuniciranje između pod sistemskih baza podataka je složeno, a opasnost od zatvaranja organa uprave u resorske okvire nije smanjena.

Drugi način dekompozicije upravljačkog informacionog sistema sledi *tokove informacija* unutar gradske uprave. Termin gradska uprava, odnosno upravljanje gradom u ovom kontekstu ne treba usko shvatiti.

Termin označava sve institucije koje se staraju o razvoju grada i zadovoljavanju potreba njegovih stanovnika. Na primer, tokovi podataka između organa unutrašnjih poslova, istražnih i pravosudnih organa, organa za saobraćaj, su po prirodi aktivnosti mnogo intenzivniji nego sa nekim drugim institucijama. Slično je sa tokovima podataka između domova zdravlja, bolnica, zdravstvenih institucija, organizacija zdravstvenog i socijalnog osiguranja. Na osnovu analize tokova podataka mogu se formirati odgovarajući pod sistemi. Prednost ovakve podele na pod sisteme u odnosu na prethodni kriterijum, očigledna je. Broj pod sistema dobijen primenom ovakvog kriterijuma znatno je manji, a osnovni tokovi podataka kruže u okviru funkcionalnih pod sistema. Nedostatak ovakve podele je što i dalje ostaje sektorska, iako u znatno manjoj meri. Pojavljuju se ponovo nepotrebnii paralelni (redundantni) podaci. Ažuriranje se obavlja na više mesta, što povećava verovatnoće greški i netačnog praćenja realnog stanja. Nedostatke ovakve dekompozicije načelno je moguće otkloniti uvođenjem dodatnih kriterijuma podele na pod sisteme.

Posebnu podelu upravljačkog informacionog sistema na pod sisteme moguće je izvršiti na osnovu dekompozicije prema tzv. generatorima podataka. U ovom prilazu traže se osnovni subjekti koji generiraju podatke relevantne za upravljanje gradom. U krajnjoj liniji, stanje grada kao objekta upravljanja može se pratiti na osnovu podataka koje generiraju svojom egzistencijom i aktivnošću građani, sve vrste organizacija koje deluju na njegovoj teritoriji, kao i stanje gradskog prostora (porvšinske parcele, podzmeni prostor, ekoprostor). Za svaki od ova tri generatora informacija vezuju se mnogobrojni parametri raspoređeni u datoteke baze podataka. U principu, postoji mogućnost da se svi parametri koji opisuju stanje građanina u odnosu na gradsku upravu (identifikacioni blok, imovinska stanja, stambena situacija, zdravstvene karakteristike i drugo) grupišu u jednom nizu oko generatora podataka. Istina, može se postaviti pitanje racionalnosti ovako glomaznih blokova obeležja, ali suština problema nije u tome. Ovakva vrsta dekompozicije, isključivo prema generatorima podataka, vodi suviše malo računa o prirodnoj strukturi složene organizacije. Na taj način pristup i tokovi podataka postaju isuviše složeni u odnosu na ustaljenu organizacionu strukturu gradske uprave.

Do sada je bilo govora o monokriterijumskom načinu dekompozicije upravljačkog informacionog sistema. Radi otklancjanja pomenutih nedostataka ovog načina dekompozicije mogu se koristiti višenamenski kriterijumi podele na pod sisteme. U tom pogledu poseban interes predstavlja složeni kriterijum dekompozicije »generator podataka plus tokovi podataka«. Na taj način se formiraju tri osnovna pod sistema upravljačkog informacionog sistema – banka podataka o građanima, o gradskom prostoru i organizacijama, ali samo sa ključnim parametrima koji se jedinstveno ažuriraju. Ostali podaci, od stalnog interesa za pojedine grupe korisnika (zdravstvo, imovinska stanja, prosveta i drugo) smešteni su u funkcionalnim pod sistemima. Ovakvim *hibridnim načinom dekompozicije* dobija se efikasan pristup informacionoj bazi, a odgovarajući blokovi podataka vezuju za najčešće korisnike i njihove načine obrade.

Ovo izlaganje nije imalo za cilj da detaljno analizira *optimalna rešenja dekompozicije* upravljačkog informacionog sistema. Kao što je rečeno, pri današnjem stanju teorije i prakse gradskih informacionih sistema, optimalna dekompozicija je stvar lokalnih uslova i heurističkih metoda. Cilj je bio da se ukaže na neke moguće kriterijume dekompozicije upravljačkih informacionih sistema, kao i njihove glavne odlike.

Završavajući uvodni deo o sistemskom prilazu razvoju gradskih upravljačkih informacionih sistema treba posebno naglasiti značaj jealnog faktora čiji se uticaj često neopravdano potcenjuje. Reč je o *rukovođenju* projektovanjem i realizacijom upravljačkih informacionih sistema. Uloga ovog faktora je višestruka. Pre svega, rukovođenje je bitan uslov uspeha svakog *multidisciplinarnog posla*, pogotovo tamo gde u projektovanju učestvuju stručnjaci vrlo raznovrsnih profila. Osim toga, interakcija između tehnike, načina rada organizacije, kadrovskih problema i ciljeva modernizacije uprave je tako tesna da zanemarivanje bilo koje karlike u lancu dovodi do neefikasnosti, pa i neupseha celog poduhvata. Ni najbolja oprema informacionog sistema ne može da, sama po sebi, izmeni statutarne i zakonske propise koji moraju da prate proces promene tehnologije rada i načina odlučivanja gradske uprave.

Imajući u vidu sve ove činjenice može se istaći nekoliko opštih principa za rukovođenje projektovanjem i realizacijom upravljačkih informacionih sistema. Neracionalno je da sve faze i sve aspekte posla, obavlja sama organizacija koja se automatizuje. Ona mora da koristi stručnjake i institucije sa strane. Međutim, neophodno je, ako se želi sistemski prilaz razvoju upravljačkog informacionog sistema da se, već u početnoj fazi, formira posebno stručno-upravno telo u okviru gradske uprave koje će preuzeti rukovođenje celim projektom. Ovo telo postavlja zadatke projektantskim grupama, saraduje s njima, prati evoluciju koncepcija, a s druge strane, priprema i sve uslove unutar same organizacije za prelaz na novu tehnologiju rada.

U određenoj fazi posla mora se, takođe, pričti *vertikalnoj organizaciji* rukovođenja izgradnjom upravljačkog informacionog sistema. Naime, u trenutku kada prevođenje uprave na novi način rada zahvati više sektora, a nova tehnička baza upravljačkog informacionog sistema počne da se pušta u rad, pogodno je pored ostalih tela formirati i operativno rukovodstvo na najvišem nivou organizacije, koje može da kompetentno i brzo rešava sva stručna, organizaciona i druga pitanja u vezi sa realizacijom upravljačkog informacionog sistema.

Jasno je da samim navođenjem elemenata sistematike analize upravljačkog informacionog sistema ne rešava pitanje celishodnosti i relativne prednosti formalne teorije sistema u ovoj oblasti primene. Ocena o vrednosti sistemske analize u određenoj oblasti primene može se dati samo na osnovu proučavanja niza karakterističnih slučajeva. Stoga su u drugom delu izlaganja izneti rezultati višegodišnjeg rada na razvoju upravljačkog informacionog sistema Skupštine grada Beograda. Međutim, i pre iznošenja praktičnih rezultata, treba primetiti da se, ako ništa drugo, sistemskim prilazom obezbeđuje jasnije sagledavanje svih relevantnih multidisciplinarnih i društvenih faktora od kojih zavisi trajniji uspeh celog poduhvata. Sistemski prilaz ukazuje, takođe, na osnovne, principijelne i metodološke alternative koje se javljaju u razvoju ovog sistema.

Već i sam izraz projektovanje i izgradnja upravljačkog informacionog sistema ima u nekim svojim crtama sasvim drugačiji karakter nego u tehnici. Kod tehničkih objekata jasno se razlikuju *uzastopne faze* idejnog projekta, glavnog projekta i izgradnje. Međutim, prilikom realizacije upravljačkog informacionog sistema ne mogu se jasno odvojiti faze projektovanja, modernizacije, uprave i automatizacije. Štaviše, one se prepliću u većoj ili manjoj meri. Najzad, ne može se govoriti ni o završetku izgradnje ovakvog sistema. Grad, kao i sve ljudske organizacije, stalno evoluiraju, tako da se ciljevi modernizacije i način upravljanja ne mogu fiksirati jednom zauvek. Prema tome, termini projektovanje i izgradnja upravljačkog informacionog sistema odnose se na uvođenje određenog stepena automatizacije aktivnosti složene organizacije, s tim što se u znatno većoj meri nego pri izgradnji tehničkih objekata ovde otvara proces kontinualnog prilagodavanja evoluciji grada.

monografija nema veće ambicije nego što je rečeno. Pogotovu ne treba smatrati da se njome izriču bilo kakve konačne ocene o upravljačkom informacionom sistemu Beograda. Međutim, ono što je do sada urađeno ukazuje da se išlo ispravnim putem i da su učinjeni originalni doprinosi razvoju gradskih informacionih sistema u našim uslovima. Pri oceni uspeha i neuspeha u izgradnji upravljačkog informacionog sistema Beograda treba imati na umu da je bilo moguće vrlo malo ili nikako istoričiti tuđa iskustva, jer je grad Beograd zajednica opština i poseban samoupravna društveno-politička zajednica.

Reference

1. Tomović R; Kerpluo W. J.: *Control of Large Systems: trends and perspectives*. University of California, Los Angeles, UCLA-ENG-7701, January 1977.

razvoj dinamičnih pomnilnikov za mikro računalnike

a. p. železnikar
m. kovačević
d. novak

UDK 681.3 - 181.4.06

Odsek za računalništvo in informatiko
Institut Jožef Stefan, Ljubljana

Članek se ukvarja s splošnimi problemi razvoja modulov z dinamičnimi pomnilniki, ki uporabljajo način multipleksiranja naslovov ter prikazuje konkretne rešitve za mikroračunalniške sisteme s procesorji 8080, 6800 in Z 80. Najprej je opisana osnovna zgradba 16 pinskega dinamičnega pomnilnika za 4k in 16k bitov in prikazani so časovni diagrami teh dinamičnih pomnilnikov z njihovimi kritičnimi točkami. Članek obravnava usklajevanje časovnih diagramov oziroma signalov, ko imamo na eni strani zahteve mikroračunalniškega sistema, na drugi strani pa karakteristike pomnilnih čipov. Podrobna krmilna vezja modulov za sisteme z 8080, 6800 in Z 80 so prikazana na slikah 5, 6 in 7 in opisana je natančna shema razvoja pomnilnega modula. Dan je tudi testirni program za sisteme z Z 80, s katerim je mogoče diverzno preizkusiti poljubni pomnilni blok. Posebni napotki za konstruktorja z analizo kritičnih mest so zbrani na koncu članka.

Development of Dynamic Memory Modules for Microcomputers. This article deals with general problems of dynamic memory modules development using chips with address multiplexing and presents some practical solutions for systems basing on 8080, 6800 and Z 80 microprocessors. First, the structure of a 16 pin dynamic memory for 4k and 16k bytes is described and time diagrams for signals of these memories are analysed. The article deals with according of time diagrams and signals, when one has the microcomputer demands on one and the characteristics of memory chips on the other hand. Detailed control circuits of memory modules for 8080, 6800 and Z 80 systems are presented in Fig. 5, 6 and 7 and an exact scheme of module development is described by pseudocode. A testing program for memories in Z 80 systems is given which enables a diversity test of a given memory block. Particular instructions for module engineering with analyses of critical problems are gathered at the end of this article.

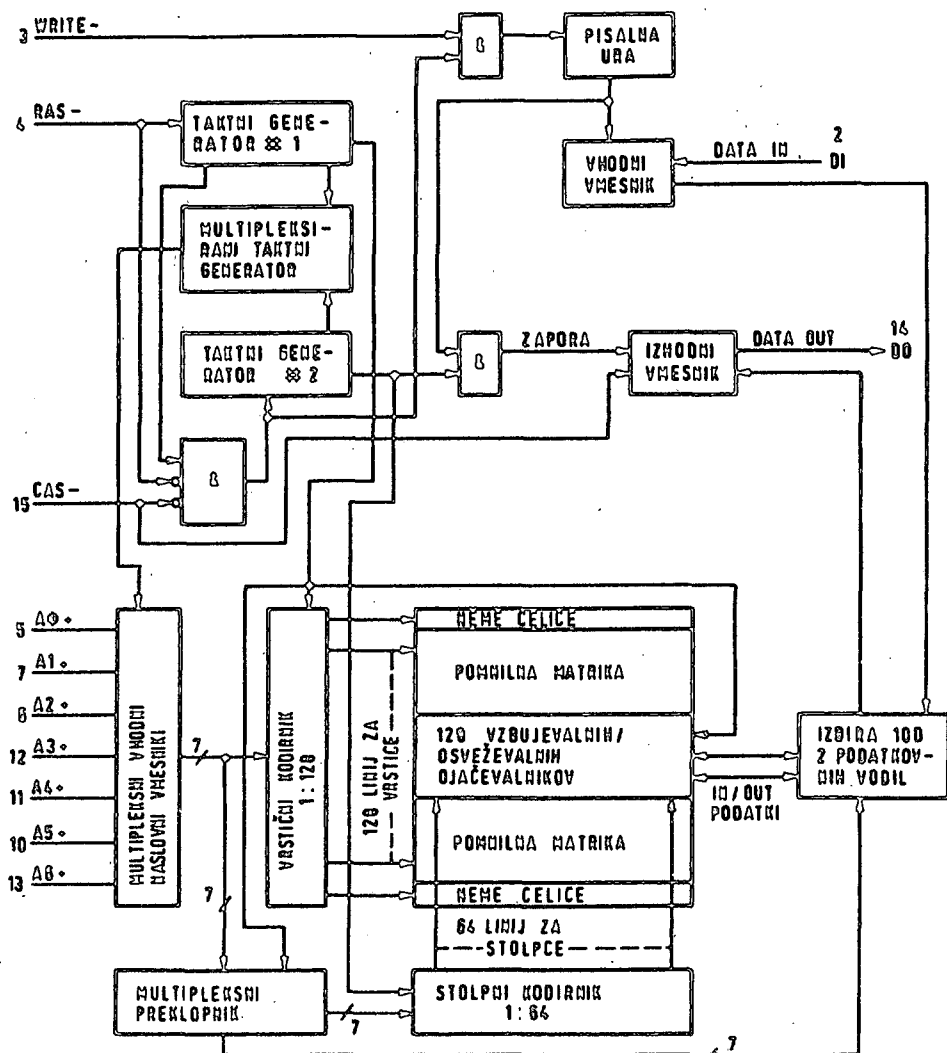
1. Uvod

Integrirana dinamična pomnilna vezja se intenzivno uporabljajo v različnih mikroračunalniških sistemih. Ta vezja (kratko pomnilni čipi ali okrajšano PC), ki jih uporabljamo pri gradnji posebnih pomnilnih sestavov, tako imenovanih pomnilniških modulov, imajo naslovljive celice, v katerih se shranjujejo stanja posameznih bitov. Število celic oziroma bitnih lokacij bo v vseh naših primerih, tj. v enem PC, vedno ali 4096 ali 16384; v prvem primeru bomo govorili o 4k PC, v drugem primeru pa o 16k PC (črka "k" pomeni "kilo"). Večkrat bomo govorili kar o dinamičnih 4k ali 16k RAM (Random Access Memory) pomnilnikih.

V članku obravnavana snov je skrbno izbrana in je posledica izkušenj pri lastnem razvoju vrste pomnilniških modulov z dinamičnimi PC. Pri tem velja poudariti, da bodo izbrani koncepti pomnilniških modulov močno odvisni od uporabe določenega mikro procesorja (npr. 8080, Intel 8080, Z 80, MC 6800 itn.); ti mikro procesorji imajo različne kontrolne signale, ki krmilijo delovanje pomnilniških modulov. Nekateri procesorji imajo že dovršen del t.i. osveževalnega vezja za dinamične pomnilnike (npr. Z 80), s katerimi se izvaja osveževanje vsebine v dinamičnih celicah PC. Druge mikro-

procesorske družine razpolagajo s posebnimi vmesniki za dinamične pomnilnike (npr. integrirano vezje DMI v družini 8080), ki izvajajo sicer dokaj zapleteno koordinacijo osveževanja. Pri procesorju MC 6800 je možnih več načinov osveževanja PC, npr. s podaljševanjem polperiode sistemskega takta oziroma z izvajanjem osveževanja v "neaktivni" polperiodi sistemskega takta (skrito osveževanje). Prva metoda zahteva t.i. način zaviranja takta, druga pa dovolj hitre PC oziroma krmilno vezje, ki lahko izkoristi pomnilni cikl PC. Pri procesorju 8080 (in tudi pri 6800) je moč uporabiti posebno, univerzalno integrirano vezje, ki generirajo osveževalni naslov v osveževalni fazi in opravljajo hkrati funkcijo multipleksiranja naslovov v vpisovalnih in čitalnih fazah.

V naših razmerah dostikrat ne razpolagamo s temi posebnimi, povezovalnimi integriranimi vezji. V takih primerih lahko zgradimo povezovalni del med naslovnim, podatkovnim in kontrolnim vodilom na eni strani in takolimenovano matriko PC na drugi strani, z navadnimi TTL in MOS vezji. To pa zahteva obvladanje načrtovanja digitalnih vezij in določeno eksperimentiranje, saj moramo razen logičnih zahtev realizirati tudi časovne pogoje, ki jih postavlja na eni strani dani mikro procesor, na drugi strani pa izbrani tip dinamičnega PC.



Slika 1. Zgradba 16k-bitnega pomnilnega čipa (npr. MK 4116); pri 4k pomnilnem čipu je na sponki 13 signal CS-

Članek prikazuje najprej osnovne podatke o zgradbi dinamičnega pomnilnika (PC), analizira prikladnost časovnih diagramov in pogoje za zanesljivo delovanje pomnilnika, pokaže osnovne probleme usklajevanja zahtev mikro procesorja in PC, obravnava konkretna vezja za dinamične pomnilnike v povezavi z mikroprocesorskimi družinami P8, 6800 in Z 80, opisuje preizkušanje izgotovljenih pomnilniških modulov (z osciloskopom in testirno programsko opremo) ter navaja napotke konstrukterju, ki mu pomagajo do uspešne izdelave specifičnega dinamičnega pomnilnega modula v mikroročunalniškem sistemu.

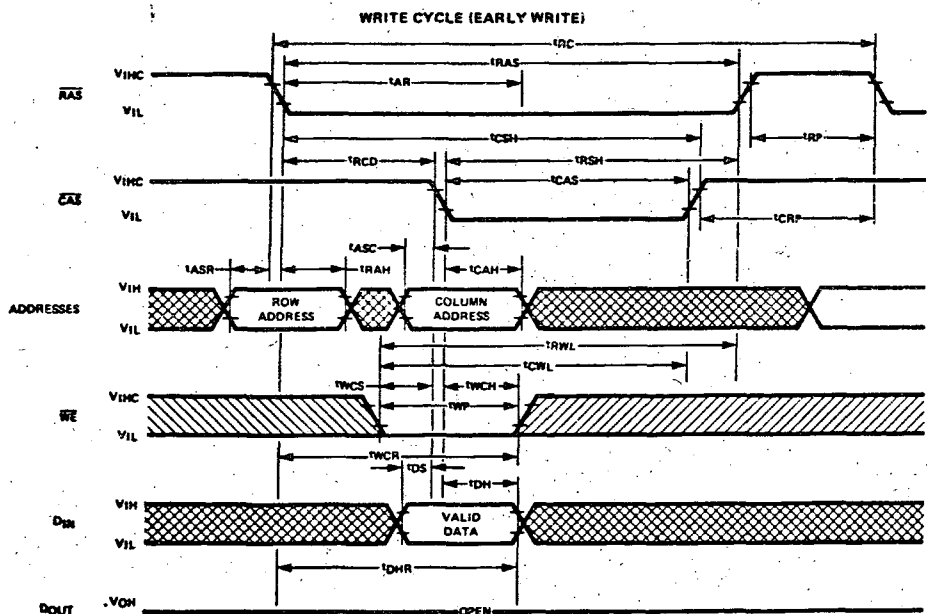
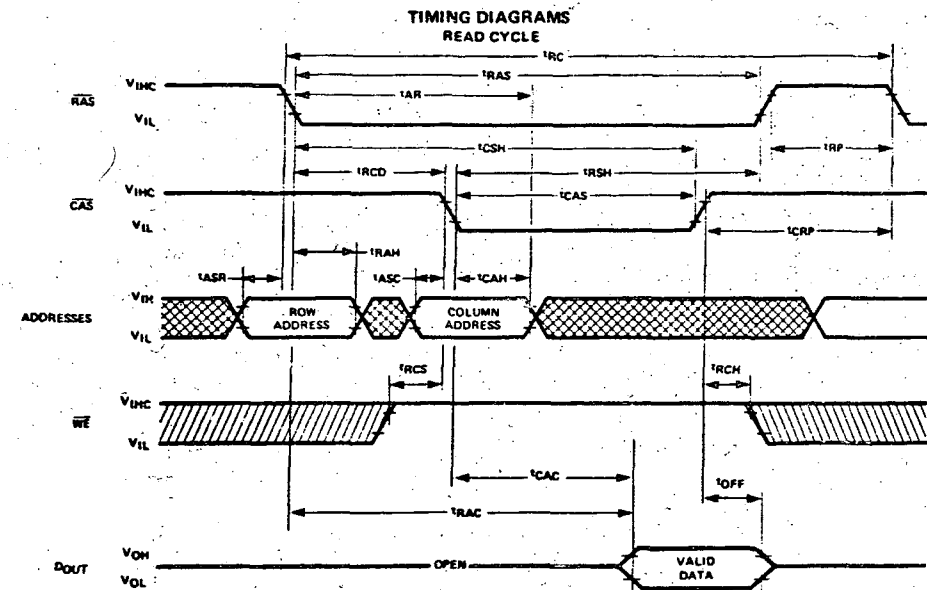
2. Osnovna zgradba dinamičnega pomnilnika

Zgradbo PC oziroma njegovo bločno shemo si bomo ogledali z namenom, da bi lažje razumeli delovanje PC, in sicer tako logično kot časovno. Razumevanje časovnega delovanja bo zlasti pomembno pri izbiri različnih alternativ generiranja krmilnih signalov in s tem konkretnega delovanja modula. Poudarimo takoj, da je moč realizirati pravilno delovanje PC z različnimi časovnimi zaporedji (različnimi kombinacijami) in časovnimi oblikami krmilnih signalov pri danem PC.

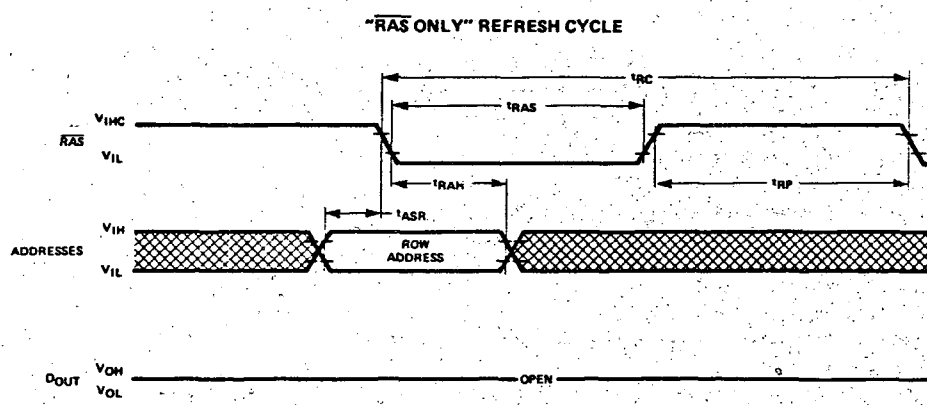
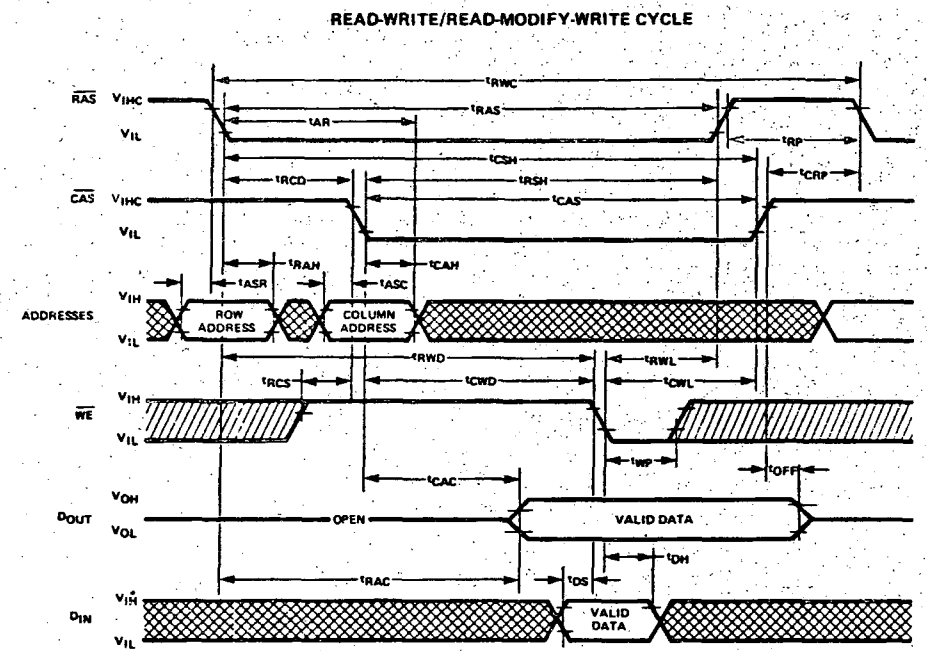
Ogledali si bomo le bločno zgradbo 16k dinamičnega pomnilnika (slika 1), ki je integrirano vezje s 16 nožicami oziroma 16 pini

(glej sliko 2). Tako vezje uporablja način nastavljanja naslovov z multipleksiranjem. Pri 4k pomnilnikih pošljemo najprej 6 naslovnih bitov in za njimi nadaljnjih 6 bitov preko 6 pinov (sponk) v vezje. Pri 16k pomnilnikih imamo dva zaporedna naslovna paketa po 7 bitov. 4k oziroma 16k RAM PC ima te krmilne signale (znak minus, tj. "-" za imenom signala pomeni, da je logična vrednost "0" aktivna vrednost, "+" za imenom pa, da je logična vrednost "1" aktivna vrednost):

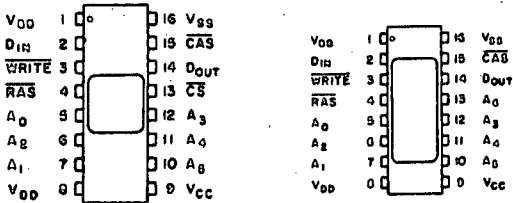
- RAS- (pomen: Row Address Strobe; negativna fronta signala RAS- začne prenašati prvih 6 ali 7 naslovnih bitov ali t.j. vrstični naslov v poseben register PC; signal RAS- traja do konca vpisovalne, izpisovalne in osvežitvene operacije)
- CAS- (pomen: Column Address Strobe; negativna fronta signala CAS- začne prenašati naslednjih 6 ali 7 naslovnih bitov ali t.j. stolpni naslov v poseben register PC; signal CAS- mora trajati pri klasičnih pomnilnih vezjih praktično natančno tako dolgo kot signal RAS-, tj. v mejah ozkih časovnih toleranc)
- CS- (pomen: Chip Select; nizka vrednost signala CS- povzroči aktiviranje 4k PC pri vpisu in čitanju; 16k PC ima na tem mestu naslovni pin A6+)



= May change in either direction
 = May change HIGH-to-LOW
 = May change LOW-to-HIGH



Slika 3. Časovni diagrami za 16k pomnilni čip, pri štirih tipičnih režimih



Slika 2. Razpored signalov na pomnilnikih čipih za 4k in 16k bitov

- WR-** (pomen: Write; z nizko vrednostjo signala WR- se opravi vpis v PC, z visoko vrednostjo pa čitanje vsebine iz PC)
- A0+** do **A6+** (pomen: Address Bit z ustreznim indeksom; zunanje vezje poskrbi, da je ob pojavu signala RAS- na razpolago prvih šest oz. sedem naslovnih bitov, ob pojavu signala CAS- pa še preostalih šest oz. sedem naslovnih bitov)
- DI+** (pomen: Data In; signal DI+ prihaja iz procesorja po podatkovnem vodilu in bit DI+ se začne vpisovati v PC z negativno fronto signala WR- ali signala CAS-, tj. s tisto od obeh front, ki se je pojavila kot zadnja, seveda ob prisotnosti ostalih signalov, tj. RAS-, CS-)
- DO+** (pomen: Data Out; signal DO+ prihaja iz PC in se v ustreznem trenutku preko posebnega vmesnika odda na podatkovno vodilo mikro računalnika, seveda pri izpolnjenih ostalih pogojih)

Signali

RAS-, CAS-, CS-, WR-, A0+ ... A6+, DI+ in DO+ so standardni za dinamične RAM pomnilnike, ki uporabljajo metodo adresnega multipleksiranja. Pri dinamičnih 16k RAM pomnilnikih nimamo signala CS+, ker je ta nožica PC potrebna za nadaljni naslovni bit.

Na sliki 1 je prikazana le groba zgradba PC za 16k bitov. Iz te slike lahko na določen način pojasnimo delovanje PC, saj je iz sheme razvidna groba logična odvisnost krmilnih, naslovnih in podatkovnih signalov. Razlika med bločnima vezjavama za 4k in 16k PC je minimalna. Večjo razliko imamo med dvema tipoma 4k PC, in sicer med t.i. klasičnim in starejšim (MCM 6604, MK 4096, F 4096 DC) ter t.i. razširljivim in novejšim 4k RAM pomnilnikom (MK 4027); slednji tip je namreč razširljiv oziroma zamenljiv s 16k RAM pomnilnikom (npr. MK 4116, F 16 K itn.), če ustrezno spremenimo povezavo na nožico A6+. Novejša pomnilna vezja z mikro računalnikih so navadno tako konstruirana, da je z minimalnim ožičevanjem možno uporabiti 4k ali 16k PC.

3. Časovni diagrami dinamičnih pomnilnikov

Poznavanje časovnih diagramov signalov PC oziroma njihovih odvisnosti je odločilno pri izbiri različnih signalnih kombinacij v konkretnih primerih uporabe, ko povezujemo pomnilna polja z mikroročunalniškimi sistemom. Časovna zaporednost, logična odvisnost signalov in njihovo trajanje nudijo vrsto možnosti, kako se najceneje dosežejo sistemske zahteve. V večjih sistemih je pri načrtovanju in razvoju dinamičnih pomnilnih vezij potrebno upoštevati tudi probleme motenj in z njimi povezano zanesljivo delovanje dinamičnega pomnilnika.

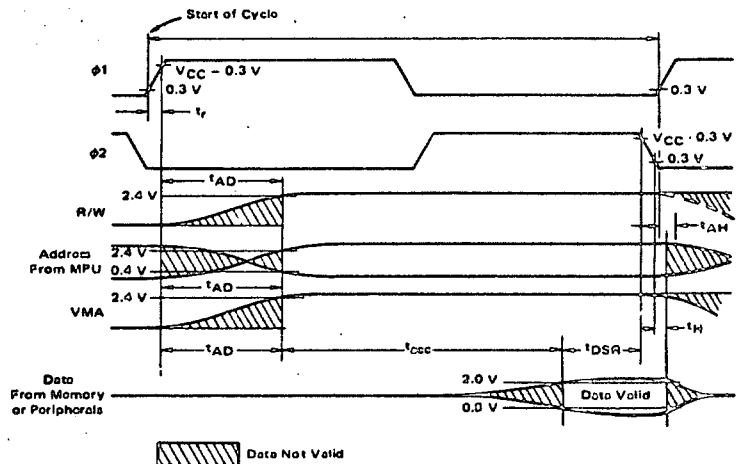
V časovnih diagramih moramo pri določeni uporabi upoštevati zaporednost signalov RAS-,

MUX+, CAS-, CS-, WR- in še nekaterih pomožnih signalov (npr. DOE-) ter njihovo vzročnost. Navadno se pri določenih zunanjih pogojih generira najprej signal RAS-, ta pa sproži skladno s časovnimi karakteristikami PC in zahtevami sistema nekatere od preje naštetih signalov. Tu imamo namreč na voljo več možnosti. Npr. pri določenih pogojih sme biti signal CS- (izbira PC pri 4k RAM) ves čas aktiven, pa tudi signal CAS- je lahko prisoten v vsaki sistemski taktični periodi, v kateri ni osveževanja PC. Podobno lahko velja tudi za signal MUX+ oziroma MUX-. Pri zaporedju signalov CAS- in WR- pa moramo biti previdni. Kadar zahtevamo kratek pomnilni cikel, se mora signal WR- pri vpisovanju podatka v PC pojaviti pred ali kvečjemu istočasno s signalom CAS-. To pa vselej ni mogoče, npr. tedaj, ko se podatkovni signali iz centralnega procesorja pojavijo proti koncu zadevne periode; v tem primeru je celotni pomnilni cikel daljši od cikla z obratnim vrstnim redom nastanka signalov CAS- in WR-.

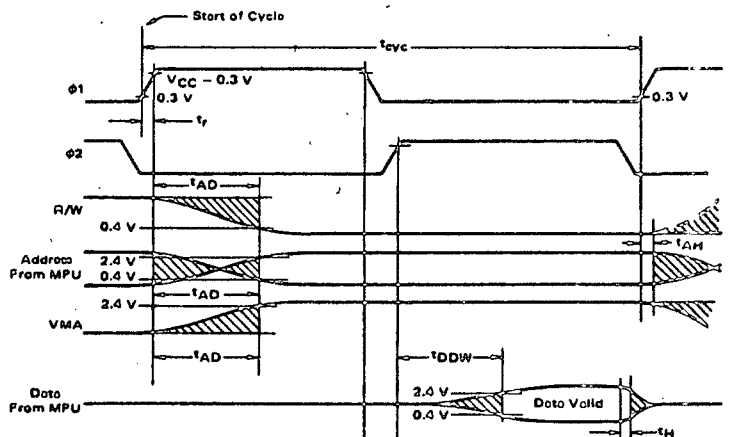
Na sliki 3 imamo časovne diagrame za 16k PC (F 16 K), in sicer za štiri tipične režime: čitalni, pisalni (z zgodnjim vpisom), čitalno-pisalni oziroma modifikirni in osveževalni cikel z RAS- kot edinim signalom. Pri tem velja opozoriti na tele kritične, značilne situacije:

1. Obstaja bistvena, kvalitativna signalna razlika med 4k klasičnimi, razširljivimi in 16k PC. Ta razlika se zrcali v cenejši izvedbi krmilnega vezja za 16k in 4k razširljive PC (neglede na mikro procesor) ter v njihovem zanesljivejšem obratovanju zaradi manj kritičnih pogojev ob nastopanju signalnih front.

READ DATA FROM MEMORY OR PERIPHERALS



WRITE IN MEMORY OR PERIPHERALS



Slika 4. Časovni diagrami za čitalni in pisalni cikel mikro procesorja

2. Klasičen 4k PČ je s stališča krmilnega vezja najbolj zahteven. Ena izmed najbolj kritičnih zahtev je, da morata biti pozitivni fronti (končni fronti) signalov RAS- in CAS- praktično poravnani. Pri nekaterih PČ (npr. Fairchild) sme biti dopustna toleranca le enosmerna (-20 ns), pri večini PČ pa se dopušča toleranca med -50 in +50 ns. Krmilno vezje mora tako poskrbeti za čimbolj "istočasno" ugasnitev signalov RAS- in CAS-.

3. Pri razširljivih 4k PČ obstaja znatna razlika med ciklom z zgodnjim (t_{RC}) in poznim (t_{RWC}) pisalnim signalom WR- (npr. pri MK 4027-4); ta razlika znese tudi do 30%, kar pomeni, da je PČ s poznim čitalnim signalom precej počasnejši. Pri 16k PČ je razlika med t_{RC} in t_{RWC} tudi tolikšna (npr. za MK 4116-4, F 16 K-4 in F 16 K-5). Ta podatek je pomemben takrat, ko primanjkuje razpoložljivega časa in moramo zaradi tega izbrati cikel z zgodnjim pisalnim signalom. Ta nevednost pa odpade pri kvalitetnejših in dražjih 16k PČ ter v celoti tudi pri klasičnih 4k PČ ($t_{RC} = t_{RWC}$ pri MK 4116-2, MK 4116-3, F 16 K-2, F 16 K-3, MK 4096).

4. Osveževanje s signalom RAS- kot edinim signalom je seveda ceno in je dopustno pri vseh 4k razširljivih in 16k PČ. Pri nekaterih klasičnih 4k PČ (tip 4096) pa tako osveževanje v delovnih režimih ni vedno dopustno, ker mora prvemu čitalnemu ali pisalnemu ciklu (npr. po daljšem obdobju samega osveževanja) nujno predhoditi par signalov (RAS-, CAS-); pri teh PČ je tovrstno osveževanje dopustno le v režimu mirovanja, ko želimo ohraniti podatke v dinamičnemu pomnilniku (npr. pri MK 4096) za daljše obdobje (npr. preko noči, ko sistem ne obratuje).

4. Dinamični pomnilnik in mikro procesor

Dinamični pomnilni modul lahko razdelimo na dva bistvena dela: kontrolno vezje in pomnilna matrika. Pri izgradnji oziroma razvoju kontrolnega vezja moramo upoštevati dvoje: sistemске zahteve in lastnosti izbranega PČ v pomnilni matriki. Kontrolno vezje je tedaj vmesnik med kompleksnim sistemskim vodilom (in njegovimi lastnostmi) na eni ter izbrano pomnilno matriko na drugi strani. Karakteristike sistema vodila so v veliki meri odvisne od izbranega mikro procesorja, saj je t.i. kontrolno vodilo, ki je del celotnega vodila, speljano iz mikro procesorja.

Pri načrtovanju kontrolnega vezja je tedaj smiselno govoriti o usklajevanju zahtev sistema oziroma mikro procesorja in dinamične pomnilne matrike. To usklajevanje je največkrat vezano na dva tipa časovnih diagramov: diagrami mikro procesorja (čitalni in pisalni cikel) in diagrami izbranega dinamičnega PČ (npr. diagrami na sliki 3).

Oglejmo si primer z mikro procesorjem M 6800, ko imamo časovne diagrame na sliki 4. Iz čitalnega diagrama je razvidno, da morajo biti podatki iz pomnilnika na razpolago za čitanje v mikro procesor vsaj v času $T_{DSR} = 100$ ns pred koncem taktne faze $\emptyset 2+$ ter morajo trajati še v času $T_H = 30$ ns po prenehanju negativne fronte signala $\emptyset 2+$. S to zahtevo je tedaj potrebno uskladiti pojav signala D_{OUT} na sliki 3 za čitalni cikel (Read Cycle). To pa pomeni, da se mora negativna fronta signala CAS- pojaviti vsaj v času t_{CAC} (npr. = 195 ns) pred časom t_{DSR} (tj. v skupnem času $t_{CAC} + T_{DSR}$) pred negativno fronto taktne faze $\emptyset 2+$.

Ob vsem tem moramo seveda upoštevati, da bo signal RAS- nastal le pri naslednjih izpolnjenih pogojih:

1. ko je izbran (selektiran) ustrezeni segment pomnilnega prostora (dekodiranje iz naslovnih bitov);

2. ko se je pojavil signal VMA+ (Valid Memory Address), ki pove, da je na naslovnem vodilu veljaven naslov (to pri M 6800 časovno ni vedno res tedaj, ko se ukaz nadaljuje preko več ciklov faze $\emptyset 2+$);

3. ko se je že pojavila negativna fronta (lahko tudi pozitivna) signala $\emptyset 2+$ (ta pogoj je potreben zaradi izjem v točki 2);

4. vsakokrat (neodvisno od točk 1, 2 in 3), ko se pojavi signal odobritve za osveževanje (Refresh Grant) dinamičnega pomnilnika.

Nekoliko bolj kritičen primer lahko nastopi v pisalnem ciklu. v tem ciklu se podatek iz mikro procesorja vpisuje v pomnilnik. Eden od dodatnih pogojev je v tem primeru, da se pojavi signal R/W+, in sicer njegova negativna fronta. Torej imamo še tole:

5. ko je pri čitanju R/W+ = 1 ter pri pisanju R/W+ = 0.

Podatek prispe do pomnilnika šele po času $T_{DDW} = 200$ ns (glej sliko 4) za pozitivno fronto faze $\emptyset 2+$; to pomeni, da moramo ta podatek prevzeti v pomnilnik ali z negativno fronto signala CAS- ali z negativno fronto signala WR-, pač v odvisnosti od tega, katera od obeh front se je pojavila kasneje. Tu imamo tudi možnost izbire zgodnjega (vobče krajšega cikla) ali poznega signala WR- (vobče daljšega pisalnega cikla). Ob tem moramo vedeti, da se prevzem podatka opravi s signalno fronto in da mora biti zaradi tega podatek v celoti prisoten ob prevzemu.

Pri razpravi o usklajevanju signalnih zahtev mikro procesorja in PČ velja poudariti, da moramo časovne diagrame (npr. na sliki 3 in sliki 4) predhodno uskladiti, tj. narisati časovni potek signalov za konkreten primer še pred načrtom kontrolnega logičnega vezja.

Primer usklajevanja časovnih diagramov z mikro računalnikom 6800 in danim PČ bi bil tudi najenostavnejši. Drugi mikro procesorji uporabljajo čitalne in pisalne cikle, ki se raztezajo preko več period taktne faze $\emptyset 4+$ (npr. F8 in Z 80); torej moramo s posebno pozornostjo upoštevati časovne diagrame takih mikro procesorjev.

5. Vezja za povezavo dinamičnih pomnilnikov z mikroprocesorskimi sistemi

Pomnilni modul, ki je sestavljen iz kontrolnega (krmilnega) vezja in pomnilne matrike, uporablja naslednje osnovne signale:

RAS-, MUX- in/ali MUX+, CAS-, WR-, CS-

Tem signalom moramo dodati še nekatere specifične interne signale modula:

DOE- ali DOE+, RAS \emptyset -, ..., RAS \emptyset +, REF-

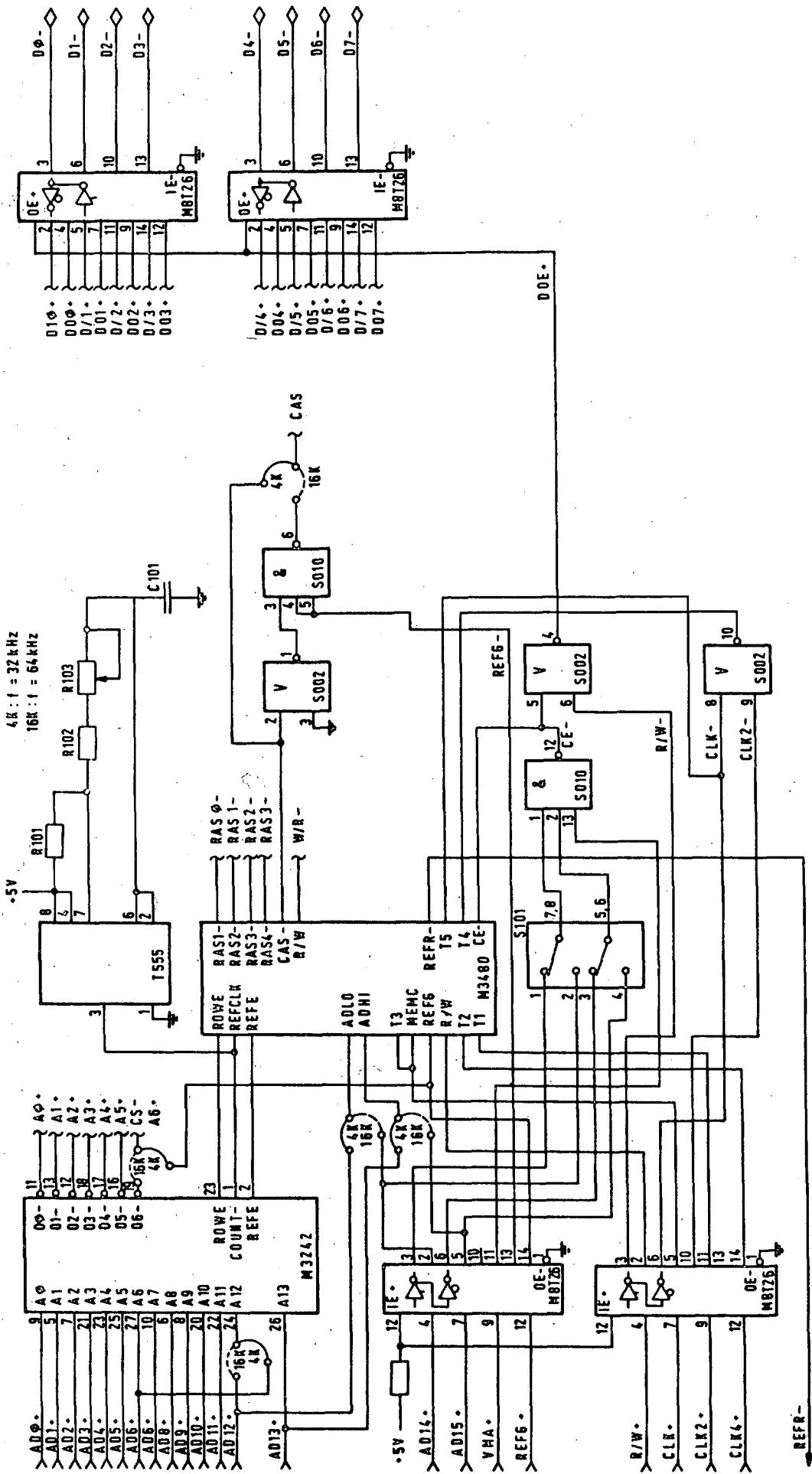
itn. Tu je DOE- (pomen: Data Output Enable) signal, ki praviloma odpira ojačevalnike s tremi stanji na podatkovno vodilo sistema. K tem signalom moramo dodati še krmilne signale mikroprocesorskega sistema; in sicer imamo:

za MC 6800: VMA+, R/W+, CLK+ (lahko tudi več derivacij), REFG+, REFR-, AD12+, ..., AD15+;

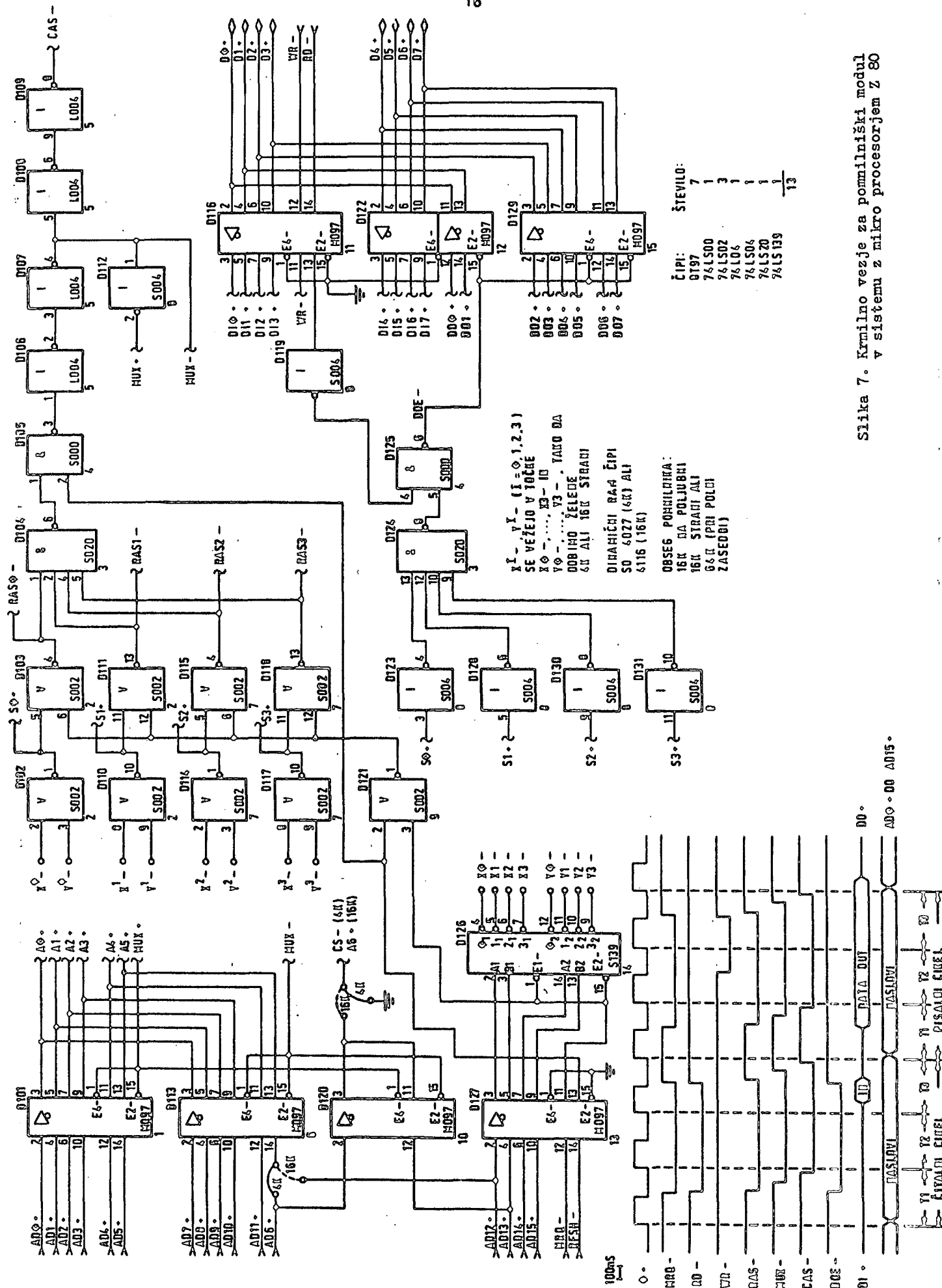
za FB : \emptyset +, W+, ROM \emptyset +, ..., ROMC4+;

za Z 80 : MRQ-, RFSH-, RD-, WR-, AD12+, ..., AD15+

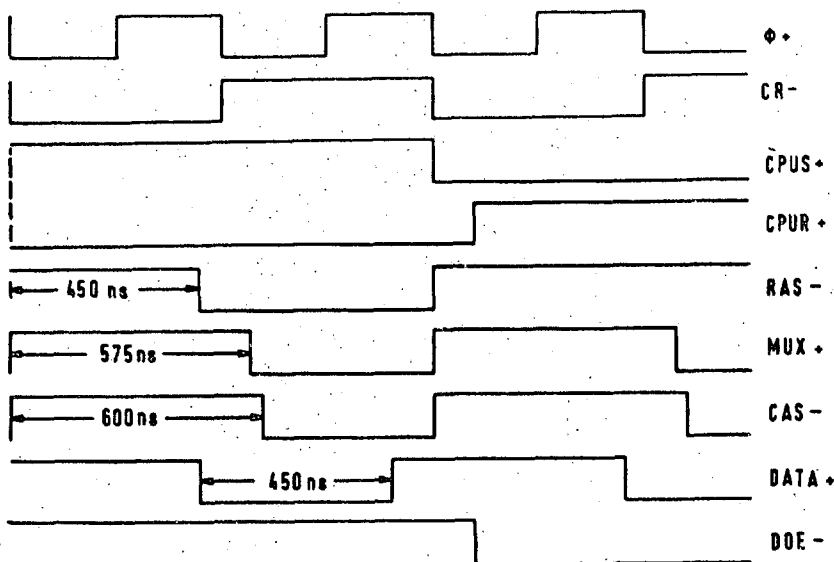
Prikazana vezja na slikah 5, 6 in 7 so konstruirana za uporabo dinamičnih 4k ali 16k PČ; z enostavnim ožičevanjem nekaj točk v teh vezjih dobimo tako osembitne (zložne) pomnilne module za 16k ali 64k zlogov. Pri tem velja tale



Slika 6. Krmilno vezje za pomnilniški modul v sistemu s procesorjem MC 6800



Slika 7. Krmilno vezje za pomnilniški modul v sistemu z mikro procesorjem Z 80

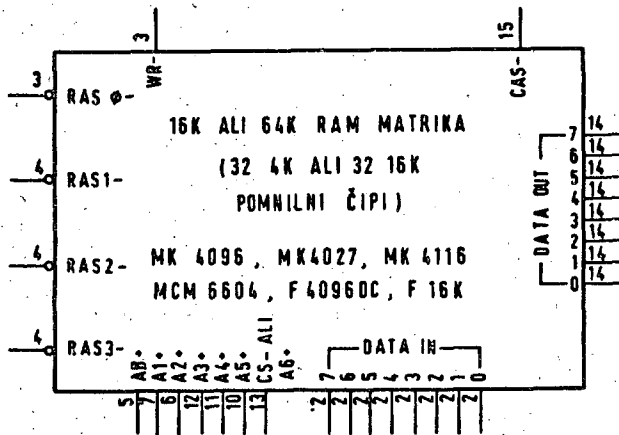


Slika 8. Časovni diagram signalov krmilnega vezja na sliki 5 v sistemu z F8 ter simbolični prikaz priključkov univerzalne matrice

R101 · C101 :: 450 ns : R102 · C102 :: 575 ns :
 R103 · C103 :: 600 ns : R104 · C104 :: 450 ns :

ČIPI : ŠTEVILO :

MK 3852	1
74LS157	2
74LS123	2
74LS 04	1
74LS 00	3
74LS 33	1
74LS 75	1
74LS 74	1
74LS83	1
74LS173	2
8T97	2
4096	32
SKUPNO	49



splošna ugotovitev: če smo konstruirali vezje za uporabo klasičnih 4k PC (npr. MK 4096) najslabše kvalitete (ujemanje pozitivnih front signalov RAS- in CAS-), tedaj bo vezje delovalo z vsemi 4k in 16k PC.

5.1. Krmilno vezje pri procesorju F8

Oglejmo si najprej pomnilno vezje v sistemu F8 na sliki 5. Za to vezje imamo časovne diagrame na sliki 8. Osnovna taktna frekvenca sistema F8 je 2MHz in s to frekvenco se pojavlja signal ϕ . Z dinamičnim vmesnikom DMI (oznaka MK 3852) se iz sistemskih signalov ϕ +, W+, ROMC ϕ +, ..., ROMC4+, D ϕ +, ..., D7+ pripravijo vsi interni naslovni signali A0+, ..., A15+ ter ostali signali za pogon in osveževanje dinamičnih PC v pomnilni matrici. Podatki D0 ϕ , ..., D07 se pri čitanju (čitanje glede na mikro procesor) vključujejo na dvosmerno podatkovno vodilo preko vmesnih registrov s tremi izhodnimi stanji (74LS173) in s pomočjo signala DOE-. Vezje ima štiri monostabilne multivibratorje (D103, ..., D106: 74LS123), ki oblikujejo časovno osnovo za generiranje signalov RAS-, MUX+, CAS- in DATA+; s slednjim signalom se prevzemajo podatki v registra D130 in D138 (glej spredaj). Časovne konstante za te štiri signale so prikazane na sliki 8; tu moramo ustrezno izbrati RC pare (R101, C101), ..., (R104, C104). Multipleksiranje se opravi s hitrim preklopnikom D102 in D113 (74LS157). Izbira 4k segmentov je določena

z nastavitvijo stikala S101 ter s seštevalnikom D124 (74LS83) in izbiralnikom D125 (74LS42). Izbiranje 16k segmentov (pri 64k modulu) praktično ni potrebno, vendar lahko zasedemo modul s 16k pomnilnimi čipi samo delno: v tem primeru lahko izbiramo tudi 16k strani oziroma segmente.

Osveževanje se izvaja avtomatično z vmesnikom D101, v katerem se nahaja osveževalni števec. Osveževanje se izvaja enkrat na štiri cikle, kar v vseh primerih zadostuje.

5.2. Krmilno vezje pri procesorju MC 6800

Na sliki 6 imamo do skrajnosti poenostavljeno krmilno vezje za sistem z mikro procesorjem MC 6800. To enostavnost omogočata dve MOS integrirani vezji: 1. naslovni multipleksor in osveževalni števec za 16k dinamične RAM PC (Intel 3242) in 2. kontroler za dinamične pomnilnike D104 (MC 3480); ti integrirani vezji poskrbita za vse potrebne krmilne signale. Osveževalna taktna ura je realizirana z vezjem D102 ter z ustreznimi časovnimi konstantami. Signal DOE+ nastane iz izbirnega signala modula, VMA+ in R/W+. V vezju so dosledno uporabljeni vmesni ojačevalniki s tremi izhodnimi stanji (tip 8T26). Posebnost tega krmilnega vezja je uporaba signalov CLK2+ in CLK4+, ki nista standardna in predstavljata takt z dvojno in štirikratno frekvenco osnovnega takta CLK+.

Vendar velja tu omeniti, da novo sistemsko taktno vezje IC 6875 (Motorola) generira tudi ti dve visoki taktno fazi; z uporabo tega vezja se hkrati poenostavi tudi sistemska ura v sistemu z mikro procesorjem MC 6800.

5.3. Krmilno vezje pri procesorju Z 80

Slika 7 kaže krmilno vezje za pomnilniški modul, ki se nahaja v sistemu z mikro procesorjem Z 80. Tu imamo štiri bistvene krmilne signale: MRQ- (pomen: Memory Request), RFSH- (pomen: Refresh), RD- (pomen: Read) in WR- (pomen: Write). To vezje je namenjeno le uporabi 4k PC razširljivega tipa (npr. MK 4027) ali 16k PC. Iz slike 7 je razvidno, da se v časovnih diagramih pozitivni fronti signalov RAS- in CAS- ne ujemata. Torej klasičnih 4k PC ne moremo uporabiti. Krmilno vezje ima zakasnilno verigo počasnih negatorjev D106, ..., D109 (74L04), s katero se doseže zelena zakasnitev med signali RAS-, MUX- in CAS-. Multipleksiranje naslovov se tu doseže z ojačevalniki s tremi stanji D101, D113 in D120 (tip 8T97). Ti ojačevalniki so hkrati vmesniki med naslovnim vodilom in krmilnim vezjem. Izbiranje pomnilniških strani se opravi preko dela ojačevalnika D127 in selektorja D126 (74LS139) tako, da se dvojice (pari) (x^I , y^I), $I = \emptyset, 1, 2, 3$ na vezjih D102, D110, D114 in D117 povežejo na sponke XK- in YJ-, $K, J = \emptyset, 1, 2, 3$, ki so izhodi iz vezja D126. Na ta način lahko 4k pomnilne strani razmečemo poljubno v okviru 64k pomnilnega prostora. Generiranje signala DOE- je standardno: ta signal nastane, ko je hkrati izbran modul (visoke naslovne linije) ter je prisoten signal RD-. Signal WR- je tu identičen s signalom WR- za pomnilno matriko. Vezje za prenos podatkov iz in na dvosmerni podatkovni vodilo je realizirano z enakimi ojačevalniki, kot jih imamo pri prevzemu naslovov iz naslovnega vodila, tj. z ojačevalniki tipa 8T97 (D116, D122 in D129). Časovni diagrami na sliki 7 (levo spodaj) kažejo medsebojno časovno signalno odvisnost.

Slika 8 kaže grobo strukturo pomnilne matrike, ki je v vseh naših prejšnjih primerih enaka. Seznan čipov in časovni diagrami pa se nanašajo na krmilno vezje v sistemu z F8, torej na sliko 5.

6. Načrtovanje in preizkušanje pomnilnih vezij

Osnovno vprašanje je, kako načrtujemo pomnilno vezje in kako ga preizkušamo. Tukaj velja poudariti, da sta načrtovanje in preizkušanje povezana postopka, ki bi ju preprosto lahko imenovali kar razvoj pomnilnega vezja. Razvoj predpostavlja tu določen načrt, ki upošteva specifikacije, eksperimentiranje, izpopolnjevanje načrta (revizijo), odpravljanje logičnih in vezalnih (tudi ožičevalnih) napak ter preizkušanje vezja, in sicer z elektronskimi aparaturami pa tudi s posebnimi testirnimi programi, ki se izvajajo kar na gostiteljskem sistemu.

Razvoj pomnilnega modula lahko tedaj poteka po tej shemi:

DEVELOPMENT din ram modul

- preglej in spoznaj specifikacije uporabnika dinamičnega modula;
- preglej in spoznaj zahteve primerne integriranega pomnilnega vezja (PC);
- preglej in spoznaj zahteve gostiteljskega sistema z mikro procesorjem;
- napiši in preizkusi program za testiranje delovanja pomnilnika v gostiteljskem mikroročunalniškem sistemu;
- DOUNTIL (modul deluje, tako kot je zahtevano) nariši časovne diagrame kontrolnih signalov modula, upošteva zahteve integriranega dinamičnega pomnilnega vezja in zahteve sistema z mikro procesorjem;
- načrtaj logično vezje z vsemi podrobnostmi

(specifikacijami, tehničnimi zahtevami), upošteva časovne diagrame; realiziraj logično vezje na plošči s tehnično ovijanja žic ali podobno metodo; nastavi s pomočjo osciloskopa krmilne signale modula, tako kot jih predvidevajo

časovni diagrami (zakasnitve, signalne fronte, logični pogoji); preizkusi modul v realnem sistemu z uporabo sistemkega monitorja (vpis in čitanje podatkov iz pomnilnika); preizkusi vse pomnilne celice modula z dovolj kompleksnim in zahtevnim preizkusnim programom, ki daje sporočila o naravi napak;

ugotovi napake (logične, vezalne, tehnološke v čipih) in predvidi njihovo odpravo;

ENDDO

nariši tiskano vezje (dvostransko) skladno s preizkušenim in pravilno delujočim vezjem in ga daj v pilotsko izdelavo;

izdelaj podrobno tehnično dokumentacijo (logični vezalni načrt, kosovnica, načrt z lego elementov na tiskanem vezju, opis delovanja modula, napotek za aparaturno in programsko preizkušanje modula);

sestavi vezje na tiskani plošči ter ga dokončno preizkusi (popravi risalne napake, če te obstajajo);

ENDEVELOPMENT

Ta razvojna shema tudi nakazuje preizkus pomnilnega modula npr. z osciloskopom ter skladno s časovnimi diagrami (npr. tistimi na slikah 7 in 8). Podrobneje pa si sedaj ogledjmo še dovolj kompleksen program za testiranje modula v konkretnem mikroročunalniškem sistemu.

Obstoječi sistem Z 80-MCB razširjamo z novim 16k modulom. Dodani modul želimo preizkušati kar s tem mikroročunalniškim sistemom. Strategija preizkušanja naj bo tale: pomnilni blok napolnimo vsakokrat z določeno vsebino, nato pa v naslednjem koraku preizkusimo to vsebino (kako se je vpisala v prejšnjem koraku). Vsak zlog pomnilnega bloka (8 bitov) pretestiramo na vse možne kombinacije, tj. z vpisom 00, 01, 02, ..., FF. Pri tem vpišemo v dva sosednja pomnilna zloga vselej komplementarni osembitni vrednosti; s tem testiramo tudi dovolj goste menjavo signalnih vrednosti na podatkovnem vodilu.

V dani blok (glej vrstice spodaj) vpišujemo zaporedoma naslednje vzorce:

```

OO, FF, OO, FF, ...
FF, OO, FF, OO, ...
FE, 01, FE, 01, ...
FD, 02, FD, 02, ...
. . . . .
O2, FD, O2, FD, ...
O1, FE, O1, FE, ...

```

Po vsaki vpisani vrstici pomnilnik testiramo. Če se vsebina osembitne celice ne ujema s primerjalno vrednostjo, tiskamo naslov celice, vsebino te celice in primerjalno vrednost. Iz razlike sklepamo, kateri bit deluje napačno, iz naslova pa, v katerem pomnilnem segmentu; tako ugotovimo, kateri pomnilni čip ali mesto v vezju je pokvarjeno. Ta program uporabljamo tudi za občasno preizkušanje pomnilnika, če sklopamo, da se je morda pojavila napaka kjer koli v pomnilnem vezju.

Imejmo najprej tolo osnovno strukturo programa:

```
PROGRAM RAMTEST
```

```
INCLUDE INIT
```

```
DOUNTIL (B = 0)
```

```
LD HL, AEBEG
```

```
LD A, B
```

```
INCLUDE STORE
```

```
LD HL, AEBEG
```

```
LD A, B
```

INCLUDE COMPAREENDDOINCLUDE SUCCENDPROGRAM

Tu je AEBEG začetek pomnilnega bloka, ki ga bomo preizkušali. V tem psevdo kodu smo uporabili že ukaze procesorja Z 80; podoben zapis bomo uporabljali tudi v segmentih INIT, STORE, COMPARE in SUCC. Imamo tele segmente gornjega programa:

SEGMENT INIT

INIT:

```
LD IX,AEBEG ;set IX to area begin
LD B,00H ;reset B
LD (IX+04H),B ;reset error flag
LD (IX+05H),B ;reset line count
LD (IX+06H),B ;reset heading flag
LD DE,AEBEG+02;set DE to area end
```

ENDSEGMENTSEGMENT STORE

STORE:

```
LD (HL),A ;store (A)
OR A ;reset carry flag
PUSH HL ;save (HL)
SBC HL,DE ;(HL)-(DE) negative?
POP HL ;reload (HL)
INC HL ;increment (HL)
CPL A ;complement (A)
JR C,STORE-$ ;if negative repeat
```

ENDSEGMENTSEGMENT COMPARE

COMPARE:

```
CP (HL) ;compare ((HL))
CALL NZ,ERMES ;if NZ go ERMES
OR A ;reset carry flag
PUSH HL ;save (HL)
SBC HL,DE ;(HL)-(DE) negative?
POP HL ;reload (HL)
INC HL ;increment (HL)
CPL A ;complement (A)
JR C,COMPARE-$ ;if negative repeat
```

ENDSEGMENTSEGMENT SUCC

SUCC:

```
LD HL,MESS ;common message
CALL MSG ;print it
LD HL,(AEBEG) ;print the memory
CALL PUTHL ; block interval
LD HL,ANDMES ;
CALL MSG ;
LD HL,(AEBEG+02);
CALL PUTHL ;
XOR A ;reset A
CP (IX+04H) ;if no failure
JR Z,OKMES ; output success
LD HL,FAILME ; otherwise print
CALL MSG ; error statement
JR END-$ ;
```

OKMES:

```
LD HL,RIGHT ;success message
CALL MSG ;
```

END:

```
JP MONITR ;program end
```

ENDSEGMENT

V segmentu COMPARE se pojavlja subrutina ERMES, ki opravi tole: natisne glavo sporočila o napakah, in sicer v zaporedju "naslov", "vsebina tega naslova" in "primerjalna vrednost". Torej ADDR MY AC Potem tiska ta subrutina pod temi naslovi zaporedoma naslove napak z vsakokratno vrednostjo MY in AC. Imamo:

SUBROUTINE ERMES

```
ERMES:
LD (IX+04H),OFFH ;set error flag
LD C,A ;save compar. value AC
```

```
XOR A ;reset acc
CP (IX+06H) ;test heading flag
JR NZ,TYPE-$ ;if NZ type error data
HEAD: ;otherwise
PUSH HL ; save (HL)
LD HL,HEADMS ; and print
CALL MSG ; heading row
POP HL ;reload (HL)
LD (IX+06H),OFFH ;set heading flag
TYPE:
INC (IX+05H) ;increment line count
CALL PUTHL ;print error address
LD A,SPACE ;
RST TTYWR ;print space
LD A,(HL) ;((HL)) to acc
CALL PUTA ;print mem.contents MY
LD A,SPACE ;
RST TTYWR ;print space
LD A,C ;reload compar. value AC
CALL PUTA ;print AC
LD A,SPACE ;
RST TTYWR ;
RST TTYWR ;print double space
LD A,06H ;test line count
CP (IX+05H) ; for end of line
JR NZ,EERM-$ ;if not end go EERM
LD (IX+05H),00H ; otherwise clear line
CALL CRLF ; count, print CR and LF
EERM:
LD A,C ;reload compar. value
RET ;return
ENDSUBROUTINE
```

```
>D 12A9 102
12A9 0D 0A 41 44 44 52 20 4D 59 20 41 43 20 20 41 44
12B9 44 52 20 4D 59 20 41 43 20 20 41 44 44 52 20 4D
12C9 59 20 41 42 20 20 41 44 44 52 20 4D 59 20 41 43
12D9 20 20 41 44 44 52 20 4D 59 20 41 43 20 20 41 44
12E9 44 52 20 4D 59 20 41 43 0D 0A 00 0D 0A 54 48 45
12F9 20 54 45 53 54 45 44 20 4D 45 4D 4F 52 59 20 42
1309 4C 4F 43 4B 20 42 45 54 57 45 45 4E 0D 0A 54 48
1319 45 20 41 44 44 52 45 53 53 45 53 20 00 20 41 4E
1329 44 20 00 0D 0A 49 53 20 44 45 46 45 43 54 49 56
1339 45 2E 20 47 4F 20 41 4E 41 4C 59 5A 45 20 54 48
1349 45 20 46 41 49 4C 55 52 45 20 4D 45 53 53 45 41
1359 47 45 2E 00 0D 0A 49 53 20 57 4F 52 4B 49 4E 47
1369 20 50 52 4F 50 45 52 4C 59 2E 20 54 48 45 20 46
1379 41 49 4C 55 52 45 2C 0D 0A 49 46 20 41 4E 59 2C
1389 20 49 53 20 4F 55 54 53 49 44 45 20 54 48 49 53
1399 20 4D 45 4D 4F 52 59 0D 0A 42 4C 4F 43 4B 2E 2E
13A9 2E 00
```

Slika 9. Tabela sporočil programa RAMTEST, izpisana v ASCII

V programu RAMTEST se pojavljajo standardne monitorske subrutine mikro računalnika Z 80-MCB, in sicer:

MSG : naslov 04D1; ta subrutina natisne sporočilo, ki začena na naslovu (HL) ter se v pomnilniku končuje z vrednostjo 00H oziroma z ASCII znakom "NULL";

PUTHL: naslov 01BE; ta subrutina natisne vsebino registra HL v obliki 4 heksadecimalnih znakov;

TTYWR: naslov 0020; ta subrutina natisne ASCII znak, ki je v akumulatorju A;

PUTA : naslov 01C3; ta subrutina natisne vsebino akumulatorja A v dveh heksadecimalnih znakih;

CRLF : naslov 04CA; ta subrutina natisne CR in LF (pomik glave v levo in nova vrstica).

Na sliki 9 imamo tabelo sporočil programa RAMTEST. Vsako sporočilo se razteza od danega naslovnega začetka do vrednosti 00H v tabeli. Sporočila so:

1207	DD	21	00	12	1257	28	08
120B	06	00			1259	21	2C 13
120D	DD	70	04		125C	CD	D1 04
1210	DD	70	05		125F	18	06
1213	DD	70	06		1261	21	5D 13
1216	ED	5B	02 12		1264	CD	D1 04
121A	2A	00	12		1267	C3	60 00
121D	78				126A	DD	36 04 FF
121E	77				126E	4F	
121F	B7				126F	AF	
1220	E5				1270	DD	BE 06
1221	ED	52			1273	20	0C
1223	E1				1275	E5	
1224	23				1276	21	A9 12
1225	2F				1279	CD	D1 04
1226	38	F6			127C	E1	
1228	2A	00 12			127D	DD	36 06 FF
122B	78				1281	DD	34 05
122C	BE				1284	CD	BE 01
122D	C4	6A 12			1287	3E	20
1230	B7				1289	E7	
1231	E5				128A	7E	
1232	ED	52			128B	CD	C3 01
1234	E1				128E	3E	20
1235	23				1290	E7	
1236	2F				1291	79	
1237	38	F3			1292	CD	C3 01
1239	10	DF			1295	3E	20
123B	21	F4 12			1297	E7	
123E	CD	D1 04			1298	E7	
1241	2A	00 12			1299	3E	06
1244	CD	BE 01			129B	DD	BE 05
1247	21	26 13			129E	20	07
124A	CD	D1 04			12A0	DD	36 05 00
124D	2A	02 12			12A4	CD	CA 04
1250	CD	BE 01			12A7	79	
1253	AF				12A8	C9	
1254	DD	BE 04					

Slika 10. Kodni format programa RAMTEST

SUCMES: naslov 12F4; vsebina: "CR""LF"THE TESTED MEMORY BLOCK BETWEEN"CR""LF" THE_ADDRESSES_"NULL"

ANDMES: naslov 1326; vsebina: _AND_"NULL"

FAILME: naslov 132C; vsebina: "CR""LF"IS DEFECTIVE. GO ANALYZE THE FAILURE_MESSEAGE_"NULL"

RIGHT: naslov 135D; vsebina: "CR""LF"IS WORKING PROPERLY. THE FAILURE_"CR""LF" IF ANY, IS OUTSIDE THIS MEMORY"CR""LF" BLOCK..."NULL"

HEADMS: naslov 12A9; vsebina: "CR""LF"ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC"CR""LF"NULL

Tu smo uporabili znak "_" za presledek.

```
THE TESTED MEMORY BLOCK BETWEEN
THE ADDRESSES 1800 AND 18FF
IS WORKING PROPERLY. THE FAILURE,
IF ANY, IS OUTSIDE THIS MEMORY
BLOCK...>
```

```
ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC ADDR MY AC
0FFF 7E 0E 0FFF 7E 0D 0FFF 7E 0C 0FFF 7E 0B 0FFF 7E 0A 0FFF 7E 09
0FFF 7E 08 0FFF 7E 07 0FFF 7E 06 0FFF 7E 05 0FFF 7E 04 0FFF 7E 03
0FFF 7E 02 0FFF 7E 01
THE TESTED MEMORY BLOCK BETWEEN
THE ADDRESSES 0FFF AND 1003
IS DEFECTIVE. GO ANALYZE THE FAILURE MESSEAGE.>
```

Na sliki 10 imamo program RAMTEST zapisan v kodnem (ukaznem) formatu, tako da ga je moč skupaj s tabelo na sliki 9 naložiti v pomnilnik ter uporabiti. Slika 11 prikazuje testirne primere s tem programom, kjer vidimo obliko sporočila pri uspehu ter pri napakah.

7. Posebni napotki: kritična mesta

Ko vstavimo izgotovljeni modul v mikroračunalniški sistem, se lahko zgodi, da delovanje tega modula ne bo zanesljivo. Krajši testirni programi npr. ne bodo pokazali napake, vendar bomo lahko po določenem času opazili nepredvidene in nedopustne spremembe vsebin na nekaterih pomnilnih lokacijah. Za zanesljivo delovanje pomnilnega modula je pomemben predvsem dovolj nizek nivo motenj, ki nastajajo pretežno zaradi pojavljanja signalov RAS- in CAS- pri čitanju, vpisovanju in še posebej pri osveževanju.

Diagram na sliki 12 je tipičen za 4k PC klasične vrste (MK 4096). Podobne diagrame poznamo tudi pri 4k razširljivih in 16k PC. Iz diagramov na sliki 12 je razvidno, da je nivo motenj najnižji tudi tik pred nastopom negativne fronte (začetka) signala RAS-. Vprašanje, ki si ga tu postavimo, je, kako se je moč izogniti motilnim vplivom napajalnih tokov in doseči zanesljivo delovanje PC skozi poljubno dolgo obdobje.

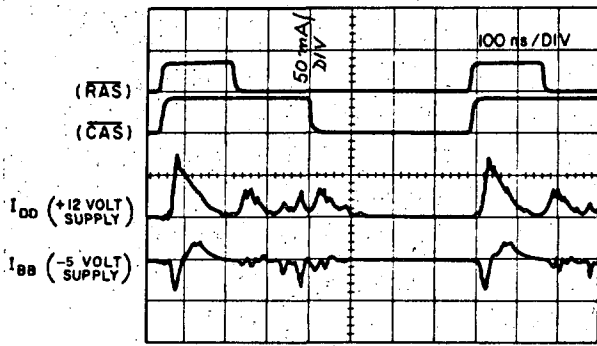
Poznamo dve učinkoviti metodi za zanesljivo delovanje PC, ki ju lahko uporabimo posamično ali pa tudi obe hkrati. Najprej moramo zmanjšati tokovne impulze napajanja na vodih (+12V, +5V in -5V) tako, da ustrezno blokiramo (premostimo s kondenzatorji) določene točke. Zahtevamo tudi nizkoinduktivne vode (palice) za preje našete tokovne vode. Uporabimo pa tudi nizkoinduktivne kondenzatorje, in sicer najmanj 10 nF na dva pomnilna čipa in najmanj 6,8 uF na osem PC.

Dodatno zanesljivost dosežemo z uporabo vzorčevanja naslovnih signalov. Npr. tik pred nastopom signala RAS- (negativne fronte) pobereмо naslovne bite v register (latch) in to informacijo iz registra (ne iz naslovnega vodila) uporabljamo potem do konca pomnilnega cikla. Dogaja se namreč lahko, da motnje iz naslovnega vodila sistema udarjajo preko vmesnih ojačevalnikov modula in povzročajo moteče posledice.

Nadaljni korak pri kontroli zanesljivega obratovanja dinamičnega pomnilnega modula je uvedba bita za kontrolo parnosti (deveti bit), s katerim lahko zaznamo eno napako v osembitnem zlogu. Ta način pa zahteva dodatni bit v pomnilniku in vezje za signalizacijo napake, ko npr. sproži prekinitve uporabniškega programa s sporočilom, kje je napaka nastala.

Nazadnje velja še omeniti, da moramo pri 4k klasičnih PC v mejah dopustnih toleranc poravnati pozitivne fronte signalov RAS- in CAS-. Ker praktično ne moremo doseči popolno 1-

Slika 11. Primer sporočila programa RAMTEST pri uspehu in pri napakah



Slika 12. Tokovni motilni signali pri generiranju signalov RAS- in CAS-

stočasnosti teh front (razlika je mimogrede nekaj 10 ns), je pred dokončno izdelavo modula obvezna kontrola z osciloskopom.

Pri logičnem načrtovanju vezja moramo izhajati iz časovnih diagramov, ki smo jih narisali z upoštevanjem karakteristik PC in zahtev gostiteljskega mikroračunalniškega sistema. Pri vseh naštetih izpolnjenih pogojih za zniževanje nivoja motenj lahko dosežemo pri dinamičnem pomnilnem modulu enako stopnjo zanesljivosti, kot jo imamo pri uporabi statičnih PC.

K programu RAMTEST pa povejmo tole: kadar želimo dati pomnilni blok med heksadecimalnimi naslovi $X_1X_2X_3X_4$ in $Y_1Y_2Y_3Y_4$ testirati s programom RAMTEST, vstavimo ta naslova v celice 1200H, ... ,1203H, in sicer tako, da imamo pre-

1200	X_3X_4
1201	X_1X_2
1202	Y_3Y_4
1203	Y_1Y_2

Z monitorsko direktivo J 1207 poženemo program. Če je pomnilni blok dolg (npr. 16k zlogov), bo trajalo testiranje tudi nekaj minut. Pri uspehu se bo sporočilo pojavilo povsem na koncu izvajanja programa, pri napaki pa ob nastanku vsake napake. Če je iz doslej generiranega sporočila o napakah razvidna fatalna napaka pomnilnika, prekinemo program s pritiskom na gumb za sistemski "reset", saj bi se sicer izpisovanje napak lahko povečalo preko razumne meje.

Program RAMTEST je moč spremeniti tako, da dobimo namesto sporočila o numerični napaki kar sporočilo o defektnem elementu, tj. o lokaciji, na kateri se pokvarjeni čip nahaja. Takšen način testiranja je tedaj primeren predvsem v proizvodnji, tj. tam, kjer pomnilne module sestavljamo.

Literatura

- (1) 1977 Memory Products Catalog, Mostek Corp., Texas 1977.
- (2) A.P.Železnikar, I.Ozimek, M.Kovačević, D. Novak, Programiranje mikro računalnikov s procesorjem Z80, Informatica 1 (1977), št. 2, str. 5-12.

kriptografija na prelomu

j. zupan

UDK 681.3.06:003.6

Kemijski institut "Boris Kidrič"
Ljubljana

Opisanih in razloženih je nekaj metod, ki se uporabljajo za šifriranje podatkov in sporočil. Navedena in diskutirana so merila za kvaliteto šifrirnih algoritmov ter smernice razvoja modernih kriptografskih metod s pomočjo računalnikov.

NEW TRENDS IN CRYPTOGRAPHY. Some simple cryptography methods and algorithms along with new directions of research in the field of modern computer cryptography are described and discussed.

UVOD

Kriptografija, šifriranje poročil, je nedvomno staro toliko kot pisava, s precejšnjo gotovostjo bi lahko sklepali, da še več, saj so verjetno tudi že stari narodi skušali s ponarejenimi znaki ali signali zakriti pravi pomen sporočil. Dokazano so prve šifre uporabljali Fenečani, za njimi Grki, Rimljani in še kasneje Benečani (1).

S prodorom računalniške tehnologije, posebno v zadnjih letih, na skoro vsa področja gospodarskih, socioloških, političnih, vojaških ter še mnogo drugih udejstvovanj, se je predvsem povečal pretok podatkov med posameznimi avtomatiziranimi enotami. Problemi varovanja zaupnosti, točnosti in avtentičnosti poslanih oz. sprejetih podatkov so postali osnovne zahteve vseh uporabnikov. Varnost vsebine podatkov ni samo stvar vojaškega ali političnega pomena, prav tako se tiče bank, industrijskih obratov in laboratorijev, trgovskih mrež, patentnih pisarn, zasebnih telefonskih pogovorov ter še mnogo drugih uporabnikov sodobnih komunikacijskih sistemov. Poleg zaščite vsebine, ki je nedvomno najpo-

membnejši dejavnik vsakega šifriranega sistema, je tu še problem, ki je navadno odrinjen na stran, a ni zaradi tega nič manj važen, to je avtentičnost podatkov. Avtentično je sporočilo s posebnim "pečatom" odposiljatelja, ki prejemniku nedvomno dokazuje, da je bilo poslano iz pravega naslova. Lahko se zgodi, da vsebina niti ni tolike pomembna, kot izvor podatkov. To je npr. problem pri letaliških radarjih, zelo obremenjenih letališč, kjer je natančno določanje izvora informacij bistveno; da o vojaških potrebah niti ne govorimo.

Seveda so na voljo tudi bolj enostavni in vsakdanji primeri, pri katerih je zaščita podatkov nujna. Vzemimo samo srednje velik računalnik, ki ga skupno uporablja nekaj ljudi. Vse njihove datoteke morajo biti načeloma zaščitene pred nedovoljenim kopiranjem, spreminjanjem, ali kakim drugačnim zlorabi. Še bolj morajo biti zaščiteni statistični podatki, npr. o bolnikih in njihovih boleznih v bolnicah, življenjepisi, in podobno.

Pri vse splošnem naraščanju zmogljivosti računalniške tehnologije seveda ne smemo pozabiti, da postaja tudi odkrivanje šifer vedno lažje.

Postavlja se vprašanje, kdo bo zmagal v tej tekmi? Če predpostavljamo, da pri ogromni množici šifriranih podatkov nikakor ne moremo zahtevati, da ostanejo vsa šifrirana sporočila večno tajna, moramo istočasno postaviti zahtevo, da mora biti šifrirni algoritem sestavljen tako, da se ga ne da razkriti, tudi, če je na voljo poljubno dolg šifriran tekst z znano vsebino.

Če sedaj na hitro strnemo, kar smo povedali, dobimo naslednje zahteve, ki jim mora ustrezati sodoben kriptografski sistem oz. postopek:

- šifrirni algoritem mora biti tak, da ga z najhitrejšimi znanimi algoritmi in z naj-sodobnejšo računalniško opremo ne moremo "zlomiti" v doglednem času (npr. v 1000 letih),
- algoritem mora biti nerešljiv tudi v primeru "plain text" testa, tj. pri predpostavki, da poznamo poljubno dolg šifrirani tekst, in
- šifrirano sporočilo mora biti avtorizirano, vsebovati mora avtorjev "pečat".

Vsem tem zahtevam nikakor ni lahko ustreči in šele v zadnjih letih (1976/77) se je z deli Diffie-a in Hellman-a (2) ter Rivesta (3) pokazala realna možnost, da bi v tem uspeli.

SUBSTITUCIJSKE ŠIFRE

Šem sodijo vsi šifrirni sistemi, ki zamenjujejo posamezne črke znanega teksta z drugimi znaki v skladu z naprej določenimi pravili ali ključem. Če so ti, drugi znaki, zopet črke abecede (to pot seveda z drugim pomenom) ni tajnost šifre nič manjša, znatno pa se olajša prenos šifriranega teksta z računalniki ali teleprinterji.

Med najenostavnejše substitucijske metode sodi nedvomno monoalfabetska substitucija s cikličnim premikom črk, znana pod imenom Cezarjeva šifra (1,4). Ključ K je v tem primeru ciklični premik vrste števil:

0, 1, 2, 3, 4, 23, 24, 25,

ki ustrezajo 26 znakom naše abecede:

presledek, A, B, C, Č, V, Z, Ž.

Če i -ti znak teksta označimo s $t(i)$, ustrezni šifrirani znak pa z $x(i)$, lahko šifrirni in dešifrirni postopek s Cezarjevo šifro napišemo v obliki:

šifriranje: $x(i) = (t(i) + K) \bmod 26$ (1a)

dešifriranje: $t(i) = (x(i) - K) \bmod 26$ (1b)

Pri tem smo funkcijo x mod y definirali kot:

$$x \bmod y = x - y \cdot \lfloor x/y \rfloor$$

Znak $\lfloor a \rfloor$ pomeni največje celo število, ki je še manjše od a .

Kljub temu, da je Cezarjeva šifra izredno enostavna in lahko zlomljiva, saj ne spremeni frekvenčne porazdelitve znakov znanega teksta, ampak jo samo ciklično premakne, je vseeno zanimiva zaradi osnovnega pristopa, ki se v izboljšani obliki uporablja tudi pri teže rešljivih polialfabetskih substitucijah.

Osnovni način šifriranja s polialfabetsko substitucijo je podan v Tabeli I za primer ključa s tremi črkami. To je 3-črkovna substitucija, kjer je ključ beseda "TRI". Ključ K v tem primeru ni konstanta, ampak končna množica $\{K\}$ p elementov $k(i)$, $i = 0, 1, 2, \dots, p-1$. Izraza (1a) in (1b) se ustrezno izpremenita:

$$x(i) = (t(i) + k(i \bmod p)) \bmod 26 \quad (2a)$$

$$t(i) = (x(i) - k(i \bmod p)) \bmod 26 \quad (2b)$$

Tabela I

Šifriranje s 3-črkovno substitucijo. Ključ je beseda "TRI".

Sporočilo	Ključ	Šifriranje	Šifra
O 16	T 21	$(16+21) \bmod 26 = 11$	11 J
B 2	R 18	$(2+18) \bmod 26 = 20$	20 Š
Z 24	I 10	$(24+10) \bmod 26 = 8$	8 G
O 16	T 21	$(16+21) \bmod 26 = 11$	11 J
R 18	R 18	$(18+18) \bmod 26 = 10$	10 I
N 15	I 10	$(15+10) \bmod 26 = 25$	25 Ž
I 10	T 21	$(10+21) \bmod 25 = 5$	5 D
K 12	R 18	$(12+18) \bmod 26 = 4$	4 Č

Polialfabetska substitucija, ali z drugim imenom Vigenereova šifra je tem teže rešljiva, čim večji je p , oz. čim daljši je ključ $\{K\}$. Tuckerman (5) je pokazal, da je Vigenereova šifra vedno rešljiva, če imamo na voljo najmanj 20 p znakov šifriranega teksta.

V primeru, da je :

- a) perioda p daljša od teksta,
- b) da je ključ $\{K\}$ uporabljen samo enkrat in
- c) da znani tekst ne pride v roke dešifrantu,

je Vigenereova šifra teoretično nerešljiva (6) in se imenuje Vernamov sistem. Vernamov sistem se veliko uporablja za prenos izjemno važnih informacij. Ker morata potovati šifrirani tekst in ključ do prejemnika po dveh ločenih poteh, je šifra za široko uporabo

zelo neprikladna.

Ideja, ki se ponuja sama od sebe, je konstrukcija neskončno dolgega ključa $\{K(i), i \rightarrow \infty\}$. Za to je navadno potreben generator slučajnostnih števil (GSS). Dobri algoritmi za izračun slučajnostnih števil ne dajo samo števil, ki so "slučajnostno" porazdeljene med 1 in neko maksimalno vrednostjo m , ampak poleg tega generirajo vsa ta števila, preden se ponove, na tak način, da izgleda izbor slučajnost. Algoritma, ki generira dovolj dolgo vrsto slučajnostnih števil, da jo lahko v praksi imenujemo "neskončna", nikakor ni lahko konstruirati (7). Knuth (7) je pokazal, da generira linearni GSS v obliki:

$$T(i+1) = (aT(i) + c) \bmod m \quad (3)$$

vrsto s periodo $m = 2^k$, takrat in le takrat, ko je c lih in $a \bmod 4 = 1$. Vrednost $T(0)$ imenujemo kal generatorja. Ker je k navadno izbran tako, da je malo manjši od števila bitov v računalniški besedi je vrsta $T(i)$, preden se začne ponavljati zelo dolga. S 60-bitno besedo naredimo lahko 10^{19} elementov dolgo "slučajnostno" vrsto, preden se začne ponavljati.

V nadaljevanju je opisan verjetno najboljši (6) algoritem šifriranja z neskončnim ključem, ki ga je sestavil Carroll (8) in izboljšal McCarthy (9). Algoritem je osnovan na dvostopenjskem kongruentnem GSS in izmenično generira dve vrsti: $S(i)$, ali vrsto kali ter $K(i)$ ali ključ. Vrednosti konstant, ki so navedene v algoritmu A, so bile uporabljene pri kodiranju Turjaške Rozamunde (Tabela II in III). Kal je bila kratica OZMIF - Obzornik za matematiko in fiziko, interpretirana v 7-bitni ASCII codi. Vsi izračuni so bili narejeni na računalniku DEC 10.

Algoritem A

- A1. (inicializacija) $m \leftarrow 2^{30}$, $a \leftarrow 2^{15}$, $c \leftarrow 3$
 A2. (vstavimo kal) $S(0) \leftarrow \text{"OZMIF"}$
 A3. (postavimo L) $L \leftarrow 20_{10} + S(0) \cdot \text{AND} \cdot 77_{16}$
 A4. (izračun S) $S(i+1) \leftarrow (a^i S(i) + c) \bmod m$,
 če je $L \leq i \leq L$
 A5. (postavimo N) $N \leftarrow 10000_{10} + S(L) \cdot \text{AND} \cdot 7777_{16}$
 A6. (izračun K) $K(i) \leftarrow S(i)$,
 če je $L \leq i \leq L$
 A7. (nova kal) $K(i) \leftarrow K(i-1) + K(i-1-L)$,
 če je $L < i \leq N$
 A8. (gramo k A3.) $S(0) \leftarrow K(N)$

Zanko A3. do A8. lahko ponavljamo poljubno dolgo. Elemente $K(i)$ uporabljamo za šifriranje ali dešifriranje v skladu z enačbama (2a) in (2b).

V tabeli II so podane frekvenčne porazdelitve nekaj manj kot 2600 znakov dolge Prešernove Turjaške Rozamunde, kodirane s Cezarjevo ($K=2$), 3-črkovno (ključ "TRI") in "neskončno" dolgo (Algoritem A) substitucijo. V zadnjem stolpcu je za primerjavo podana še frekvenčna porazdelitev 100000 enakih znakov (presledkov), šifriranih z algoritmom A. Teoretično enakomerna porazdelitev 26-ih znakov je 3,85% na znak in se, kot je razvidno iz tabele II, precej dobro ujema s porazdelitvijo črk kriptograma. Za primerjavo so v tabeli III navedeni še prvi 4 verzi omenjene pesmi, kodirane na vse tri načine.

Tabela II

Frekvenčne porazdelitve črk Prešernove Turjaške Rozamunde šifrirane na tri različne načine. Število vseh znakov (s presledki, a brez ločil) je 2565.

Znak	Znani tekst	Cezarjeva šifra	3-črkovna substitucija	Rozamunda	Algoritem A 100000 presl.
presl.	17.32	1.52	4.81	4.03	3.76
A	8.01	0.86	1.92	4.81	3.77
B	1.92	17.32	3.83	4.15	3.91
C	0.63	8.01	2.78	3.09	3.96
D	1.29	1.92	3.87	4.07	3.83
E	2.97	0.63	3.40	3.25	3.91
F	11.02	1.29	2.89	3.87	3.88
G	0.00	2.97	3.21	3.87	3.69
H	1.41	11.02	3.36	4.11	3.79
I	1.06	0.00	7.74	3.91	3.81
J	5.00	1.41	6.57	3.68	3.82
K	4.50	1.06	3.82	3.95	3.84
L	2.93	5.00	3.05	3.40	3.72
M	3.56	4.50	2.81	3.56	3.86
N	2.03	2.93	2.27	3.76	3.85
O	4.65	3.56	5.55	3.83	3.96
P	7.23	2.03	1.65	3.64	3.93
Q	2.46	4.65	2.19	4.34	3.79
R	5.08	7.23	7.19	3.99	3.90
S	4.30	2.46	4.65	4.34	3.93
T	1.60	5.08	7.35	3.91	3.79
U	3.52	4.30	3.56	3.87	3.86
V	1.95	1.60	2.19	3.72	3.80
W	3.17	3.52	1.64	3.99	3.86
X	1.52	1.95	5.43	3.17	3.88
Y	0.86	3.17	3.21	3.68	3.88

Se namig za dešifriranje 3-črkovne substitucije: vsota frekvenc črk T, R in I (Tabe-

la II, stolpec 4) je natanko enaka frekvenci presledka pri znanem tekstu (stolpec 2, $3.56 + 7.19 + 6.57 = 17.32$).

Tabela II

Prvi štirje verzi Turjaške Rozamunde (a) kodirani s tremi šiframi, (b) Cezarjeva šifra, (c) 3-črkovna substitucija in (d) algoritem A.

a HRAST STOJI V TURJAŠKEM DVORU
b JŠCTVBTBRLVKBZBVZŠLCUMGOBFŽRŠZB
c CIILLHLLZDBHPRČOIŠTKT EHZNZKMH
d CTJUATL OČGLJZPŽ IAFUŽ CJOGSRA

a VRH VZDIGUJE SVOJ V OBLAKE
b ŽŠJBŽ FKIZLGBTZRLBZBRČNMGB
c PIRŠNFZBPOCNŠJEICHPRŽUDIEZH
d AJACTCATŠSDFŠOTASPFNŠSC EDO

a V SENCI PRI KAMNITNI MIZI
b ŽBTGPKBSZKBMCOFKVPKBOK KB
c PRB FKČR KBHESVHBČHBHGBRBA
d MOŠBRVFIKIHRFJVHŽCBKCBTOFM

a ZBOR SEDI GOSPODE ŽLAHTNE
b ČRŠBTGFKBIRTSRFBANCVJVPGB
c RŠZKRB VSŠ ŽLHZZHSDICZL R
d LČN AJALPOSEKŽVVŠSIABTČDNK

DES (Data Encryption Standard) ŠIFRA

Leta 1973 je NBS (National Bureau of Standards) pripravil smernice (6) za izdelavo standardnega šifrirnega algoritma, katerega tajnost naj ne bi bila pogojena s tajnostjo algoritma, pač pa s ključem, ki bi nudil toliko možnosti, da ga ne bi bilo mogoče odkriti, če bi hoteli preizkusiti vse možnosti. Smernice so vsebovale tudi stališče, da mora biti algoritem dostopen vsem zainteresiranim uporabnikom. Po dveh letih je bil v Federal Registru U.S. (10, 11) objavljen ustrezeni algoritem, za katerega avtorji (IBM) trdijo, da ustreza vsem sodobnim zahtevam uporabnikov, kakor tudi zahtevam NBS.

DES algoritem deluje na 64-bitnem (2 32-bitni besedi) bloku teksta (8 ASCII-1968 8-bitnih znakov), in ga v skladu s 56 bitnim ključem predela v 64-bitni blok kodiranega teksta. Vsi operatorji (s, c, p, ls, e), ki nastopajo v algoritmu so standardizirani in si jih lahko vsak ogleda v literaturi (12-14).

Algoritem DES. (Spodnji indeks pomeni število bitov v besedi)

- DES01. (start) $B_{56} \leftarrow s(\text{ključ: } A_{64})$
 $i \leftarrow 0, j \leftarrow 0$
- DES02. (cepitev) $C(0)_{28}, D(0)_{28} \leftarrow c(p^0(B_{56}))$
- DES03. (naslednji korak) $i \leftarrow i + 1$, če je $i > 16$, gremo k DES07.
- DES04. (i-ti ciklični premik)
 $C(i)_{28} \leftarrow ls^i(C(i-1)_{28})$
 $D(i)_{28} \leftarrow ls^i(D(i-1)_{28})$
- DES05. (skrčitev)
 $K(i)_{56} \leftarrow s(C(i)_{28}, D(i)_{28})$
- DES06. (gremo k DES03.)
- DES07. (nov 64-bitni blok)
 $j \leftarrow j + 1, i \leftarrow 0$,
če ni več teksta STOP
- DES08. (permutacija p) $B'_{64} \leftarrow p^1(B_{64})$
- DES09. (cepitev) $L(0)_{32}, R(0)_{32} \leftarrow c(B'_{64})$
- DES10. (naslednji korak)
 $i \leftarrow i + 1$,
če je $i > 16$ gremo k DES14.
- DES11. (izračun L(i)) $L(i)_{32} \leftarrow R(i-1)_{32}$
- DES12. (izračun R(i))
 $R(i) \leftarrow \text{algoritem } R(R(i-1), L(i), K(i))$
- DES13. (gremo k DES10.)
- DES14. (inverzna permut.)
 $X(j)_{64} \leftarrow p^{-1}(R(16)_{32}, L(16)_{32})$
- DES15. (gremo k DES07.)

Algoritem R(R,L,K).

(Spodnji indeks pomeni št. bitov v besedi)

- R1. (vhodni podatki) $R_{32}, L_{32}, K_{48}, i \leftarrow 0$
- R2. (razširitev) $A_{48} \leftarrow e(R_{32})$
- R3. (logični .XOR.) $B_{48} \leftarrow A_{48} \cdot \text{XOR} \cdot K_{48}$
- R4. (cepitev) $C(1)_6, C(2)_6, \dots, C(8)_6 \leftarrow c(B_{48})$
- R5. (skrčitev) $D(i)_4 \leftarrow s(C(i)_6)$,
za vse $1 \leq i \leq 8$
- R6. (sestavitev)
 $F_{32} \leftarrow (D(1)_4, D(2)_4, \dots, D(8)_4)$
- R7. (permutacija) $F_{32} \leftarrow p(F_{32})$
- R8. (izračun rezultata) $R_{32} \leftarrow F_{32} \cdot \text{XOR} \cdot F_{32}$
- R9. (return R)

Algoritma DES in R lahko primerjamo s privzdigovanjem in mešanjem kart. DES1. DES6. pripravi 16 "kupčkov" za primešanje v skladu s ključem (K(i) so vedno enaki). V zanki DES7. DES13. tekst, "kupček" 64-ih kart, 16 krat "privzdignemo" na dva dela L(i) in R(i). Algoritem R poskrbi za i-to zmešanje obeh

polovic skupaj z ustreznim "kupčkom" $K(i)$. Po vsakem mešanju v algoritmu R, celotni razširjeni "kup" še skrčimo na pravo velikost.

Morda bolj zanimive, kot sam algoritem, so bile reakcije in kritike, ki jih je sprožila njegova objava v Federal Registru. Najbolj očitno je bilo predvsem, da stoji za algoritmom IBM, ki je s tem zopet dosegel eno od prednosti na boju za trg. Vse firme v ZDA, ki bodo vgrajevale šifrirni software v svojo opremo, bodo "vezane" na 32-bitni blok, kar je zopet v prid IBM. Drugi, bolj resni, ugovor pa zadeva zlomljivost šifre. Marca 1976 sta Diffie in Hellman (12) kritično vzela v precep izbiro 56-bitnega ključa in postavila vprašanje, zakaj je prišlo do te odločitve, ko bi 94 ali 128-bitni ključ z gotovostjo pokopal vse upe dešifrantov. Sestavila sta predračun hipotetičnega računalnika, ki bi imel vgrajenih 10^6 specialnih chipov za paralelno preizkušanje vseh $2^{56} \approx 10^{17}$ možnih ključev. S tehnologijo iz leta 1976 je možno izdelati specialni chip, ki bi preizkusil 10^6 ključev v sekundi, kar bi pomenilo, da bi celotno vezje lahko pregledalo vseh 10^{17} možnosti v 10^5 sekundah, tj. v približno enem dnevju. Pri ceni 10 \$ na chip in pri 5 letni amortizaciji celotnega sistema, ki je obremenjeno še s 100 % stroški za energijo in kontrolo, bi to pomenilo ceno 10000 \$ za eno rešitev. Če upoštevamo, da je v povprečju mogoče priti do rešitve v pol dneva, je cena še manjša.

Julija 1977 je G.B. Kolata v Science (13) objavila zanimivo informacijo, da je nad razvojem DES ves čas bedela NSA (National Security Agency), ki jo je predvsem zanimal tisti del smernic NBS, ki so zagotavljale dobavo šifrirnega sistema vsem potencialnim uporabnikom. Članek citira kritike DES-a, ki trdijo, da NSA gotovo ni zainteresirana za to, da bi ZDA izvažale šifrirni algoritem, ki ga ni mogoče razkriti. Cena dešifrirnega računalnika, ki sta jo navedla Diffie in Hellman je ravno tako visoka, da si jo lahko privoščiti vlada, ne pa privatna firma. Zanimiv je še podatek, da je predstavnik Bell Telephone Company izjavil, da je DES preveč nezanesljiv, da bi ga firma uporabljala.

Delno na pritisk Diffiea in Hellmana, delno pa na rastoče zanimanje drugih računalniških strokovnjakov, sta NBS in IBM organizirali avgusta in septembra 1976 dva semi-

narja, na katerih so diskutirali o varnosti DES algoritma. Medtem ko so na prvem sestanku vztrajali, da dešifrirnega računalnika, kot ga opisujeta Diffie in Hellman ne bo mogoče izdelati do 1990 leta, so strokovnjaki IBM na drugem simpoziju izjavili, da tudi IBM raziskuje možnost izdelave takega sistema do leta 1981, ki pa bi stal 10 krat toliko, kot predvidevata Diffie in Hellman, tj. 200 milijonov dolarjev. Dodatni zaključek obeh sestankov je bil še, da je algoritem DES, kljub nekaterimi pomanjkljivostim, trenutno (1976) najboljši komercialno dobavljivi šifrirni sistem.

ŠIFRIRNI SISTEM ZA JAVNO UPORABO

Komaj slabo leto po tej ugotovitvi je Rivest s sodelavci (3) objavil izpopolnjeno inačico javnega šifrirnega sistema, kot sta ga zasnovala Diffie in Hellman (2,4). To in pa napoved, da bo MIT (Massachusetts Institute of Technology) pričel izdelovati integrirano vezje za šifriranje podatkov, ki bo delovalo na omenjenem postopku, je privedeno do takšnih posledic, da si je sistem, ki mu vsi priznavajo visoko kvaliteto, vredno ogledati. Pri zamisli Diffiea in Hellmana lahko tak sistem primerjamo z javnim telefonskim omrežjem. Vsak, ki bi želel sprejemati in oddajati šifrirana sporočila bi dobil "številko" in "telefon". "Številka" vsakega uporabnika bi bila, skupaj z njegovim imenom in naslovom, natisnjena in objavljena v javnem "telefonskem" imeniku. "Številka" so vsi parametri šifrirne funkcije, s katero lahko šifriramo poljubno sporočilo. Vsak, ki želi poslati šifrirano sporočilo nekomu, ki je naveden v "imeniku", pogleda njegovo "številko" (šifro) in kodira ter pošlje sporočilo. V primeru, da sam nima "telefona" lahko uporabi "javno govorilnico", ker je tudi funkcija, ali algoritem za šifriranje javen. Še več, vsak, ki ima "številko" oz. "telefonski priključek", lahko sporočilu doda avtoriziran "pečat", ki dokazuje, da je pošiljatelj res on. Tak, na prvi pogled, skoro nemogoči šifrirni sistem, je osnovan na tipu algoritma, ki ga Diffie in Hellman imenujeta "one-way trap function" (OWT), ali po naše enosmerna past. OWT funkcija ima naslednje lastnosti:

- je enolična nad prostorom ključev $\{K\}$:

$$y = f^k(x), k \in \{K\},$$

- ima inverzno funkcijo g, tako je:

$$g^k(f^k(x)) = f^k(g^k(x)) = x,$$

- eksistira algoritem Z , s katerim lahko za vsak k iz $\{K\}$ izračunamo f^k in g^k ,
- računalniško nemogoče (v doglednem času) je najti ali določiti funkcijo g^k , ali njej ekvivalentni algoritem, če poznamo samo f^k in algoritem Z .

Pri predpostavki, da poznamo tak algoritem Z , ki nam določa pare f^k in g^k , ki skoro za vsak k iz $\{K\}$ ustrezajo zgornjim pogojem, bi javni šifrirni sistem deloval takole: vsak uporabnik i ima dva algoritma f^i in g^i . Vsi f^i so objavljeni v "imeniku", vse g^i obdrže uporabniki tajne. Z uporabo f^i lahko vsak, tudi tisti, ki nima lastne šifre, spremeni sporočilo t v šifro x in jo pošlje uporabniku i :

$$x = f^i(t)$$

Ker je zadnji pogoj izpolnjen se ni bati, da bi kdorkoli razen lastnika funkcije g^i , lahko dešifriral sporočilo:

$$t = g^i(x) = g^i(f^i(t)) = t$$

Z dvojnimi šifriranjem lahko dodatno pošiljatelj j (lastnik "številke j ") pošlje uporabniku i tudi avtorizirano sporočilo t :

$$z = g^j(t); \quad x = f^i(z) = f^i(g^j(t))$$

Ko prejemnik i dobi sporočilo x in ga prvič dešifrira z uporabo lastne funkcije g^i , dobi nerazumljivo sporočilo z , ki postane smiselno šele, ko uporabi pošiljateljevo javno "številko" f^j . Dejstvo, da dobi smiselno sporočilo po uporabi prave f^j , mu nedvomno dokazuje, da je sporočilo lahko poslal le uporabnik j , ki je prvič šifriral sporočilo z lastnim g^j .

$$z = g^i(x) = g^i(f^i(z)); \quad t = f^j(z) = f^j(g^j(t))$$

Vse bi bilo lepo in prav, če bi bilo OWT funkcijo lahko določiti in najti algoritem Z . Diffie in Hellman ugotavljata, da kljub nekaterim sugestijam (2), nista našla primerne rešitve zastavljene naloge.

Kot sem že omenil že prej, so Rivest in sodelavci iz MIT lani prišli do originalne rešitve, temelječe na teoriji praštevil, ki omogoča izvedbo omenjenega sistema v vseh podrobnostih. Še več, avgusta je bilo v reviji Scientific American objavljeno (4), da je Rivestovo poročilo (64), brezplačno na voljo vsem zainteresiranim. Istočasno je navedeno tudi, da bo integrirano vezje, ki ga namerava izdelovati MIT slonelo na 128 bit-

nem ključu, ki bo zanesljivo jamčil za varnost šifre. Prav tako je bilo objavljeno, da bo na Simpoziju o kriptologiji, napovedanem za oktober 1977 na Cornell University, pod pokroviteljstvom IEEE (Institute of Electrical and Electronics Engineers), diskusija o Rivestovem algoritmu. S tem se je stvar zapletla. NSA je posvarila IEEE, da je javna objava takih raziskav v nasprotju z Zakonom o zaščiti zaupnih tehničnih podatkov ... atomskega orožja in kriptografije (15). IEEE je predal obvestilo MIT in Stanfordu, kjer so se odvetniki takoj spravili na delo in izjavili, da sodijo, da je objava še v mejah zakona. Kljub temu je bilo razpošiljanje Rivestovega poročila ustavljeno. Reakcija na vmešavanje NSA v raziskovalno dejavnost je bila zelo hitra in Science (16) se je vprašala do kod segajo meje svobodnih raziskav in celo The New York Times (17) je na prvi strani postavil isto vprašanje.

Za konec lahko navedem samo grobi obris Rivestove metode, v kolikor je bila objavljena (4, 14), saj sem tudi sam med tistimi, ki niso dobili njegovega poročila.

Vsak uporabnik i objavi dve števili s^i in r^i , ki služita za šifriranje do 20 znakov dolgega bloka teksta t :

$$x = t^{s^i} \pmod{r^i}$$

Dešifriranje poteka na isti način:

$$t = x^{w^i} \pmod{r^i},$$

pri čemer igra potencia w^i vlogo tajne inverzne funkcije g^i ; r^i je produkt dveh praštevil p^i in q^i , ki morata biti večji od 2^{63} . Obe števili s^i in w^i sta tudi funkciji p^i in q^i . Da bi zlomili šifro, moramo najti potenco w^i oziroma obe števili p^i in q^i , ki določata vse tri parametre šifre: s^i , r^i in w^i . Rivest ocenjuje, da je čas potreben za določitev dveh 63-bitnih praštevil, ki tvorita določen 126 bitni produkt, z najboljšim znanim algoritmom in z najzmoglivejšim računalnikom $4 \cdot 10^{25}$ let.

Vsekakor dovolj.

Literatura:

1. D.Kahn, The Codebreakers, Macmillan, New York, 1967
2. W.Diffie, M.E. Hellman, IEEE Trans.Inform. Theory 22, 644, (1976).
3. R.L. Rivest, A.Shamir, L.Adleman, On Digital Signatures and Public-Key Cryptosys-

- tem, Technical Memo 82, MIT, April 1977.
4. M. Gardner, *Scient. Am.* (8), 120, (1977).
 5. B. Tuckerman, IBM Report RC 2879, May (1970).
 6. L.J. Hoffman, *Modern Methods for Computer Security and Privacy*, Prentice Hall 1977, strani 49, 59, 70-86.
 7. D.E.Knuth, *The Art of Computer Programming*, Sec. Ed., Addison Wesley, Reading, 1973, Vol. II, s. 78, 155.
 8. J.M. Carroll, P.M. Leland, *Security and Privacy in Computer Systems*, Ed. L.J. Hoffman, Melville Publ. Co., Los Angeles, 1973.
 9. J.McCarthy, *Comm. ACM* 18, 131, (1975).
 10. *Federal Register*, 40 (52), 12134, (1975).
 11. *Federal Register*, 40 (149), 32395, (1975).
 12. W.Diffie, M.E.Hellman, *Comm. ACM*, 19, 164, (1976).
 13. G.B. Kolata, *Science*, 197, 438, (1977), July 29.
 14. G.B. Kolata, *Science*, 197, 747, (1977), August 19.
 15. *International Traffic in Arms Regulations*, 22 CFR, 125.01 +ITAR, 125.01, Category XIII (b).
 16. G.B. Kolata, *Science*, 197, 1345, (1977).
 17. N.W. Brown, *The New York Times*, 19. Oktober 1977, stran 1.

jedan postupak za mikro- procesorsko ostvarenje prekidačkih funkcija

s. nikolić
ž. tošić

UDK 681.3 - 181.4

Elektronski fakultet, Niš

U radu je dat jedan način za predstavljanje prekidačkih funkcija pogodan za njihovo mikroprocesorsko ostvarenje. Svaki član disjunktivne normalne forme određen je parom (M, D) . Ovaj par pruža mogućnost određivanja vrednosti funkcije za date vrednosti nezavisno promenljivih. Postupak za ovo određivanje je efikasniji od postupka koji je izložen u radu [1].

A PROCESS OF SWITCHING FUNCTIONS IMPLEMENTATION BY MICROPROCESSORS. One way of representing switching functions suitable for their microprocessor implementation is given in this paper. Each member of a disjunctive normal form is determined by the pair (M, D) . This pair makes possible determining values of a function for the given values of the variables. This determining procedure is more effective than that explained in paper [1].

1. UVOD

Mikroprocesorsko ostvarenje prekidačkih funkcija je sledeći problem. Data je prekidačka funkcija, a traži se da se ona na pogodan način predstavi u mikroprocesorskoj memoriji. Jednovremeno se postavlja zahtev da taj način predstavljanja omogućí efikasan postupak za određivanje vrednosti prekidačke funkcije za date vrednosti nezavisno promenljivih. Rešenje se pri tome ocenjuje na osnovu veličine potrebnog memorijskog prostora i brzine određivanja vrednosti funkcije za date vrednosti nezavisno promenljivih.

U radu [1] je dato rešenje na osnovu minimalnih normalnih formi prekidačkih funkcija. Svaki član normalne forme predstavljen je sa minimalnim i maksimalnim elementom odgovarajućeg kuba. Na osnovu relacije uređenja temena kuba ispituje se da li data vrednost nezavisno promenljivih pripada kubu. Shodno tome se određuje vrednost prekidačke funkcije.

U ovom radu se razmatra poboljšanje postupka za mikroprocesorsko ostvarenje prekidačkih funkcija koji je izložen u radu [1]. Poboljšanje se sastoji u većoj brzini određivanja vrednosti funkcije za datu vrednost nezavisno promenljivih pri istoj veličini potrebnog memorijskog prostora.

2. ELEMENTI POSTUPKA

Ovo poglavlje ukazuje na jednu mogućnost predstavljanja proizvoda promenljivih,

potpunih ili skraćenih, u mikroprocesorskoj memoriji, kao i na elemente postupka za ostvarenje prekidačkih funkcija.

Definicija 1. Proizvodu nezavisno promenljivih

$$P = \tilde{x}_1 \dots \tilde{x}_i \dots \tilde{x}_n, \tilde{x}_i \in \{x_i, \bar{x}_i, 1\}, i=1, \dots, n \quad (1)$$

pridružuje se niz

$$M = m_1 \dots m_i \dots m_n, m_i \in \{0, 1\}, i=1, \dots, n$$

na sledeći način

$$m_i = \begin{cases} 0, & \tilde{x}_i = 1 \\ 1, & \tilde{x}_i \in \{x_i, \bar{x}_i\}, i=1, \dots, n \end{cases}$$

i naziva se maskom.

Definicija 2. Proizvodu nezavisno promenljivih tipa (1) pridružuje se niz

$$M^* = m_1^* \dots m_i^* \dots m_n^*, m_i^* \in \{0, 1\}, i=1, \dots, n$$

na sledeći način

$$m_i^* = \begin{cases} 0, & \tilde{x}_i \in \{x_i, \bar{x}_i\} \\ 1, & \tilde{x}_i = 1 \quad i=1, \dots, n \end{cases}$$

i takodje se naziva maskom.

Proizvodu tipa (1) može se pridružiti kub $C = (v_1, \dots, v_i, \dots, v_n | v_i \in \{0, 1\}, i=1, \dots, n)$ [1] pri čemu važi

$$\begin{aligned} \tilde{x}_i = 1 &= v_i \in \{0, 1\} \\ \tilde{x}_i = x_i &= > v_i = 1 \\ \tilde{x}_i = \bar{x}_i &= > v_i = 0 \end{aligned} \quad i=1, \dots, n$$

Definicija 3. Maksimalni element kuba C jeste teme DeC za koje važi:

$$\begin{aligned} \tilde{x}_i = 1 & \Rightarrow d_i = 0 \\ \tilde{x}_i = x_i & \Rightarrow d_i = 1 \\ \tilde{x}_i = \bar{x}_i & \Rightarrow d_i = 0 \end{aligned} \quad i=1, \dots, n$$

Definicija 4. Maksimalni element kuba C jeste teme $G \in C$ za koje važi:

$$\begin{aligned} \tilde{x}_i = 1 & \Rightarrow g_i = 1 \\ \tilde{x}_i = x_i & \Rightarrow g_i = 1 \\ \tilde{x}_i = \bar{x}_i & \Rightarrow g_i = 0 \end{aligned} \quad i=1, \dots, n$$

Teorema 1. Za $v = v_1 \dots v_i \dots v_n$, $v_i \in \{0, 1\}$, $i=1, \dots, n$ važi $v \in C$ ako je

$$v \wedge M = D \quad (2)$$

Pri tome je M pridruženo proizvodu tipa (1) kome odgovara kub C, a \wedge označava pokomponentnu I operaciju.

Dokaz. Ako se proizvod tipa (1) važi $\tilde{x}_i \neq 1$, $i=1, \dots, n$ tada na osnovu Definicija 1 i 3, važi da $v \wedge M$ i D imaju i-te komponente jednake 0.

Ako za proizvod tipa (1) važi $\tilde{x}_i \neq 1$, $i=1, \dots, n$, onda je na osnovu Definicije 1, $m_i = 1$, pa je i-ta komponenta $v \wedge M$ jednaka $v_i \in \{0, 1\}$. I i-ta komponenta u D je jednaka v_i , jer sva temena kuba C imaju iste i-te komponente za koje važi $\tilde{x}_i \neq 1$. U suprotnom $v \notin C$.

Teorema 2. Za $v = v_1 \dots v_i \dots v_n$, $v_i \in \{0, 1\}$, $i=1, \dots, n$ važi $v \in C$ ako i samo ako je

$$v \vee M^* = G \quad (3)$$

pri tome je M^* , na osnovu Definicije 2, pridruženo proizvodu tipa (1) kome odgovara kub C, a \vee označava pokomponentnu ILI operaciju.

Dokaz. Ako za proizvod tipa (1) važi $\tilde{x}_i = 1$, $i=1, \dots, n$, tada na osnovu Definicije 2 i 4, važi da $v \vee M^*$ i G imaju i-te komponente jednake 1.

Ako za proizvod tipa (1) važi $\tilde{x}_i \neq 1$, $i=1, \dots, n$, onda je na osnovu Definicije 2, $m_i^* = 0$, pa je i-ta komponenta $v \vee M^*$ jednaka $v_i \in \{0, 1\}$. Takodje je i-ta komponenta G jednaka $v_i \in \{0, 1\}$. U suprotnom $v \notin C$.

Prethodne dve teoreme ukazuju na način ispitivanja da li je $v \in C$. Dovoljno je da je ispunjena jedna od relacija (2) ili (3), pa se može tvrditi da $v \in C$.

Teorema 3. Par $(M, D) ((M^*, G))$ jednoznačno određuje je proizvod tipa (1).

Dokaz. Na osnovu Definicije 1 (Definicije 2) $M(M^*)$ svojim 1 (0) -komponentama ukazuje

na promenljive koje su zastupljene u ovom proizvodu. Na osnovu Definicije 3 (Definicije 4) D(G) vrednošću svojih komponenta određuje stanje komplementiranja ovih promenljivih.

Teorema 4. Ako M sadrži q - komponenta 1, onda se odnosi na 2^q proizvoda tipa (1).

Dokaz. Bez uticaja na opštost dokaza, neka je

$$\begin{aligned} M &= m_1 \dots m_i \dots m_q m_{q+1} \dots m_n \\ m_i &= 1, \quad i=1, \dots, q \\ m_i &= 0, \quad i=q+1, \dots, n \end{aligned}$$

Na osnovu Definicije 1, ako je $m_i = 1$ u proizvodu tipa (1) promenljiva $\tilde{x}_i \in \{x_i, \bar{x}_i\}$ proizvoljno. Tome odgovaraju dva različita proizvoda tipa (1). Pošto svakom $m_i = 1$ ($i=1, \dots, q$) odgovaraju nezavisno po dva proizvoda ukupan broj različitih proizvoda je očigledno 2^q .

3. POSTUPAK ZA OSTVARENJE PREKIDAČKIH FUNKCIJA

Prekidačke funkcije $y = f(x_1, \dots, x_i, \dots, x_n)$ predstavljena je minimalnom disjunktivnom normalnom formom

$$y = p_1 v \dots v p_i v \dots v p_r$$

Pri tome je p_i ($i=1, \dots, r$) proizvod tipa (1). Na osnovu Teoreme 3 svaki član p_i ($i=1, \dots, r$) je određen parom (M_i, D_i) . Na osnovu Teoreme 1 i parova $(M_1, D_1), \dots, (M_i, D_i), \dots, (M_r, D_r)$ koji određuju svaki član disjunktivne normalne forme može da se odredi vrednost prekidačke funkcije za zadatu vrednost nezavisno promenljivih $v = v_1 \dots v_i \dots v_n$, $v_i \in \{0, 1\}$, $i=1, \dots, n$. Pri tome se koristi sledeći postupak.

1. Na osnovu para (M_i, D_i) počev od $i=1$ ispituje se da li je ispunjen uslov

$$v \wedge M_i = D_i \quad (4)$$

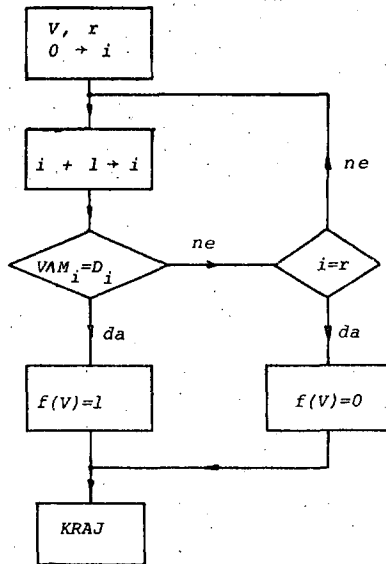
Ako je uslov ispunjen $f(v) = 1$. Ako uslov nije ispunjen prelazi se na korak 2.

2. Korak 1 se ponavlja dok se za neko $i=1, \dots, r$ ne utvrdi da važi jednakost (4).

3. Ako ni za jedno $i=1, \dots, r$ ne važi (4), onda je $f(v) = 0$.

Na Sl. 1 je prikazan dijagram toka ovog postupka.

Ovde se pretpostavlja da su parovi podataka (M_i, D_i) $i=1, \dots, r$, smešteni u mikroproceorskoj memoriji. Pokomponenta I operacija može se ostvariti pomoću logičke AND naredbe mikroprocesora.



Sl. 1

Sličan dijagram toka može da se formira na osnovu Teoreme 2.

Na osnovu Teoreme 4, ovde može da se postigne, u izvesnim slučajevima, ušteda memorijskog prostora mikroprocesorske memorije, zajedničkim korišćenjem maske M za više proizvoda u disjunktivnoj normalnoj formi.

4. ZAKLJUČAK

Dat je jedan postupak za određivanje vrednosti prekidačke funkcije za zadatu posebnu vrednost nezavisno promenljivih. Svaki član disjunktivne normalne forme prekidačke funkcije opisan je parom podataka (M, D) . Pri tome M opisuje koje su promenljive zastupljene

u odgovarajućem članu, a D opisuje stanje njihovog komplementiranja. Na osnovu para (M, D) ispituje se da li posebna vrednost nezavisno promenljivih pripada kubu koji odgovara datom članu normalne forme. Shodno tome se određuje vrednost prekidačke funkcije.

Izloženi postupak je pogodan kod mikroprocesorskog ostvarenja prekidačkih funkcija. Veličina potrebnog memorijskog prostora iznosi

$$P = 2nr \text{ (bita)}$$

gde je: n - broj promenljivih prekidačke funkcije,

r - broj članova disjunktivne normalne funkcije.

Zajedničkim korišćenjem M ovde se može postići izvesna ušteda memorijskog prostora. Prosečno vreme T potrebno za određivanje vrednosti funkcije je

$$T = \frac{r}{2} t$$

gde je t prosečno vreme izvršenja mikroprocesorskih naredbi potrebnih u ovom postupku. Ova vrednost prosečnog vremena važi pod uslovom da je svaki element para (M, D) smešten u poseban registar mikroprocesorske memorije.

5. LITERATURA

- [1] A. Albau, J. Figueras: AN ALGORITHM FOR BOOLEAN FUNCTION IMPLEMENTATION USING A MICROCOMPUTER, *Microarchitecture of Computer Systems*, R.W. Hartenstein and R. Zaks, (eds.), *Euromicro*, North-Holland Publishing Company, 1975.
- [2] T. Aleksić: Logička sinteza digitalnih sistema, *Prograđ*, "Naučna knjiga", 1971.

tv - terminal za mikroračunalo

V. Zgaga

UDK 681.3 - 181.4:681.327.13

Institut "Rudjer Bošković" Zagreb

Opisana je izvedba sklopa za prikaz alfanumeričkih podataka na standardnom TV prijemniku. Novi način generiranja video-signala i novi način adresiranja memorije znakova bitno smanjuju broj potrebnih komponentata, a ujedno proširuju mogućnosti terminala: uz dodatak potrebne memorije dobije se repetitivna grafika, a sklopovi dodani generatoru znakova omogućuju ispis slova č, š, ž, ć. "Cursor" se generira hardware-ski. Znakovi mogu biti bijeli na crnoj podlozi, ili invertirani - crni na bijeloj podlozi. Terminal je predviđen za rad pod kontrolom procesora 8080 ili Z 80.

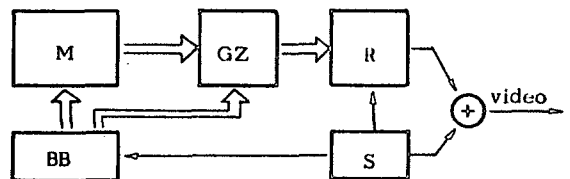
TV-TERMINAL FOR A MICROCOMPUTER: A hardware realization for the display of alphanumeric data on a standard TV receiver is described. A new way of video-signal generating and a new way of character memory addressing considerably reduce the number of components required. At the same time they increase terminal capacities. When necessary memory is added, the repetitive graphic terminal can be achieved, and by some circuits added to the character generator the display of letters is accomplished. Cursor is generated by hardware. Characters can be displayed as white on a black background, or vice-versa. The terminal is designed to work under the control of microcomputer based on 8080 (or Z80) processor.

UVOD

Razvoj mikroračunarskih sistema zahtjeva i paralelno razvijanje uređaja za ulaz/izlaz podataka. Ispis slova brojeva i specijalnih znakova na ekranu katodne cijevi (alfanumerički prikaz) svakako je vrlo pogodan način komuniciranja sa korisnikom, pogotovo ako postoji i mogućnost "crtanja" po ekranu (grafički prikaz). Za male računarske sisteme također je bitna i jednostavnost izvedbe takvog terminala, jednostavnost povezivanja sa procesorom, i, naravno, cijena.

Kako kod jednostavnijih alfanumeričkih terminala veći dio troškova otpada na samu katodnu cijev i sklopove za otklon zrake, opravdano je kao jedinicu za prikaz upotrijebiti obični TV prijemnik. Kompleksnost elektroničkih sklopova ovisi o funkcijama koje se žele postići: rad u mikroračunarskom sistemu obično dozvoljava direktnu višezilnu vezu između terminala i računala, pa otpada potreba za paralelno-serijskim pretvorbama. Paralelnim vezama na male udaljenosti mogu se postići vrlo velike brzine prijenosa podataka između računala i terminala, pa se neke funkcije terminala (čak i generiranje cursora) mogu izvesti programom. Ovakav pristup omogućuje izradu "inteligentnog" terminala malih dimenzija sa relativno jednostavnim elektroničkim sklopovima, koji se može ubaciti kao modul u kutiju samog mikroračunala (time naziv "terminal" u stvari gubi smisao). Najjednostavniji sklop za prikaz podataka na TV prijemniku sadrži, dakle, samo memoriju znakova M, generator znakova GZ, registar za paralelno-serijsku pretvorbu R, binarno brojiło za adresiranje me-

morije BB, te sklop za generiranje sinhro-impulsa S. Video signal dobije se zbrajanjem TTL-izlaza iz posmačnog registra i sinhro-impulsa. Broj dijelova ne može biti manji, jer je prijenos podataka između bilo koja dva bloka sa sl.1 pre brz za današnje mikroprocesore.



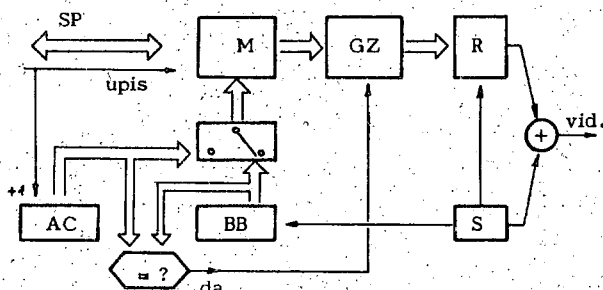
Sl.1. Minimalna konfiguracija

Promjena ispisa na ekranu televizora postiže se promjenom sadržaja memorije znakova. U memoriji su znakovi jedan po jedan spremljeni u ASCII kodu. Sklop za sinhronizaciju sinhronizira otklon zrake u TV prijemniku sa adresiranjem memorije znakova, tako da se za određeni položaj zrake u TV prijemniku uvijek adresira ista lokacija u memoriji. ASCII znak upisan u toj lokaciji pomoću generatora znakova i posmačnog registra pretvara se u niz TTL impulsa, koje prijemnik ispisuje kao niz svijetlih i tamnih točaka (postupak generiranja video-signala detaljno je opisan u slijedećem poglavlju). Ako se želi na nekome mjestu ekrana ispisati neki drugi znak, a ne onaj koji se na tom mjestu upravo prikazuje, potrebno je

promijeniti ASCII kod u odgovarajućoj memorijskoj lokaciji. Da bi mikroručunalo to moglo izvršiti, potrebno je prema sl.1. :

- privremeno adresirati memoriju iz mikroručunala (npr. preko sabirnice adrese)
 - u adresiranu lokaciju ispisati novi ASCII kod.
- Kako se redovno ispisuje znak do znaka, zgodno je imati brojilo koje uvijek pokazuje na slijedeću memorijsku lokaciju (upisom novog podatka u tu lokaciju brojilo se inkrementira). Sad mikroručunalo treba, da bi upisalo novi znak (desno do znaka koji je zadnji bio upisan), učiniti slijedeće :
- prebaciti adresiranje memorije na brojilo "AC" (sl.2)
 - upisati novi znak u memoriju (adresa je zadana iz "AC")
 - inkrementirati brojilo AC
 - vratiti adresiranje na brojilo BB.

Korisniku je potrebno znati adresu brojila AC, kako bi znao na kom mjestu će se upisati slijedeći znak. Zato se na ekranu na tom mjestu (to je trenutak kad je sadržaj AC jednak sadržaju BB) generira posebni znak, tzv. "cursor". Brojilo AC sadrži, dakle, adresu cursora. Blok shema ovakvog uređaja je na sl.2. Veza sa računalom je ostvarena preko sabirnice podataka procesora SP i kontrolne linije za upis podataka sa sabirnicu u izlaznu jedinicu (signal I/OW za procesor 8080).

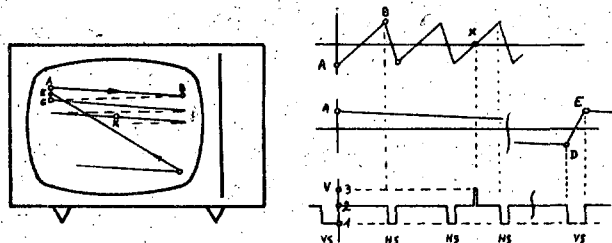


Sl.2. Dodatak "cursora"

Konfiguracija sa sl.2. je još uvijek relativno jednostavna za izvedbu, ali sa bitno većim mogućnostima nego ona sa sl.1. Brojilo AC i komparator između AC i BB mogli bi se riješiti i programom, no to bi značilo da vrlo velik dio raspoloživog vremena procesora (oko 50%) otpada samo na simuliranje rada tih sklopova. Zato su ti sklopovi izvedeni hardvarski - konfiguracija sa sl.2. izabrana je kao temelj za izgradnju ovdje opisanog TV terminala.

GENERATOR SINHRO-IMPULSA

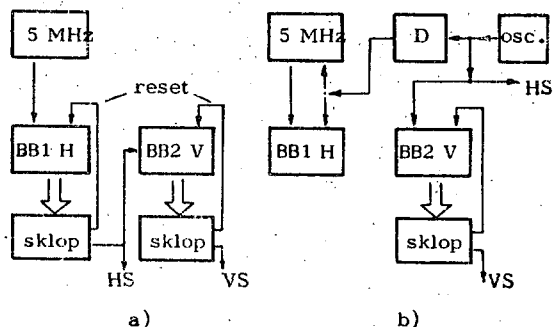
Ako se kao jedinica za prikaz koristi obični TV prijemnik, potrebno je signalu koji prenosi podatke o svjetlni točki dodati i impulse za sinhronizaciju, pomoću kojih prijemnik sinhronizira svoje sklopove za otklon zrake sa predajnikom. Upravljanje prijemnikom pomoću sinhro impulsa prikazano je na sl.3.



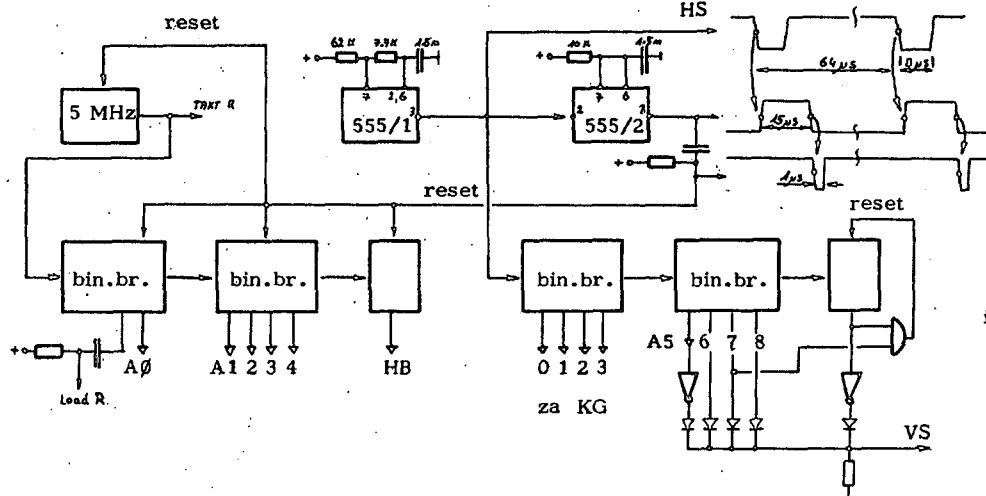
Sl.3. Rad TV prijemnika

Najgornji graf prikazuje amplitudu horizontalnog otklona zrake u TV prijemniku, srednji (U_V) amplitudu vertikalnog otklona, a donji prikazuje video signal (V) koji je prouzročio oba otklona. Na video signalu se mogu razlučiti tri osnovna naponska nivoa. Najniži nivo (1) je nivo sinhro impulsa, (2) je nivo "crnoga", a (3) je nivo "bijeloga". Za vrijeme kretanja zrake duž jedne linije (npr. od A do B) video signal modulira svjetlinu točke : ako duž cijelog puta od A do B točka treba biti nevidljiva, amplituda video signala treba za to vrijeme biti na nivou "crnoga". U točki B predajnik treba prijemniku narediti "vрати zraku na lijevi rub". To se radi kratkim impulsom na nivou 1, koji se zato zove horizontalni sinhro-impuls (HS). Taj impuls će u prijemniku upravljati horizontalnim oscilatorom. Po završetku HS impulsa zraka će se naći na lijevom rubu ekrana, pa će početi ispisivanje nove linije. Ako na sredini treće linije treba ispisati svijetlu točku, video signal će izgledati kao što je prikazano na slici. Na kraju zadnjeg reda otpremnik treba prijemniku narediti "vрати se u početni položaj". To se opet izvodi impulsom na nivou 1, tzv. vertikalnim sinhro impulsom (VS). Taj impuls traje bitno duže od impulsa HS, pa se u prijemniku obično raspoznaje sklopom sa RC spojem i komparatorom.

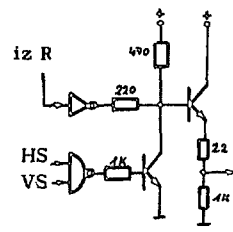
U Evropskoj TV mreži prikazuje se 293 vidljive linije za vrijeme jedne vertikalne periode. Svaki drugi VS impuls postavlja zraku u točku E, za pola razmaka među linijama ispod točke A. Time se u stvari dobiva prikaz sa $2 \cdot 293 = 586$ vidljivih linija. Svaki VS impuls traje koliko i ispis 19,5 linija (nevidljivih!), što ukupno daje 625 redova, tj. jednu "sliku". Frekvencija vertikalnog otklona izabrana je da bude baš frekvencija mreže (50 Hz). To daje za frekvenciju horizontalnog otklona $50 \cdot 312,5 = 15625$ Hz. Za frekventni opseg video signala izabrano je od DC do 5 MHz, što dozvoljava do 530 elemenata slike (točaka) po svakoj liniji. Za ispis znakova na TV prijemnik zgodno je upotrijebiti samo polje od 256×256 točaka ($2^8 = 256$). Prikaz na 256 linija se postiže zanemarivanjem razlike između svakog drugog vertikalnog otklona, a preostalih 293-256 = 37 vidljivih linija se zanemaruje (treba ih učiniti nevidljivim). Uobičajeni način generiranja sinhro-impulsa je dekodiranje stanja dva binarna brojila (sl.4a). Prvo brojilo pogoni se oscilatorom od 5 MHz, a njegova binarna vrijednost odgovara amplitudi horizontalnog pomaka zrake. Od stanja brojila 0 do 255 ispisuju se znakovi, zatim se u širini 5 znakova (255-295) ispisuje nivo crnoga, pa u širini 6 znakova (295-343) HS impuls, te opet 5 znakova crnoga. HS impuls inkrementira drugo brojilo, koje sadrži redni broj retka koji se ispisuje, tj. iznos vertikalnog otklona zrake. Od stanja 0 do 255 ispisuju se znakovi, od 256 do 293 ispisuje se nivo crnoga, a od 293 do 312,5 ispisuje se VS impuls. Oba brojila resetiraju se kad postignu određeni sadržaj (npr. drugo brojilo resetira se nakon što postigne vrijednost 313).



Sl.4. Mogućnosti za dobivanje HS i VS impulsa



Sl. 5. Prvi dio sheme terminala



Sl. 6.

Nedostatak upravo opisanog načina generiranja sinhro-impulsa je u složenosti sklopova za dekodiranje stanja brojlara. Ti sklopovi mogu se bitno pojednostavniti tako da se HS impulsi generiraju astabilom. Na sl. 4a je prikazan klasičan način generiranja sinhro impulsa, a na sl. 4b je način sa astabilom. Na sl. 5 je shema astabila koji generira HS impuls, monostabila kojim se podešava trenutak početka ispisa znakova (udaljenost znakova od lijevog ruba ekrana), te oscilatora od 5 MHz koji prije ispisa svakog novog reda treba biti resetiran - inače bi ispis pojedine točke bio nedefiniran za trajanje jedne periode oscilatora, tj. baš za širinu jedne točke.

Brojilo za horizontalni otklon realizirano je sa dva 4-bitna binarna brojlara i jednim bistabilom. Nakon 256 točaka izlaz bistabila postane "1": to je signal koji "gasi" zraku u TV prijemniku. Ovdje više nisu potrebni nikakvi dodatni sklopovi. Binarno brojilo za vertikalni otklon također je realizirano sa dva 4-bitna binarna brojlara i jednim bistabilom. Upotrebljeni generator znakova pretvara ASCII kod u znak od 7x9 točaka. Iskorištena je mogućnost upotrebljenog generatora znakova da se znakovi ispisuju na 8x16 točaka (8 točaka po svakoj horizontalnoj liniji, a 16 linija) - mjesta izvan polja 7x9 kod ispisa ostaju zatamnjena. Zbog 16 linija po znaku očito prvo brojilo treba adresirati jednu od 16 linija pojedinog znaka, a drugo 4-bitno brojilo određuje redak znakova koji se ispisuje (na ovaj način može se na ekranu ispisati najviše 16 redaka teksta, i najviše 32 znaka u svakom redu). Iz stanja tog drugog 4-bitnog brojlara i bistabila jednostavno se može dobiti impuls za vertikalnu sinhronizaciju (sl. 5). Pogodnim miješanjem oba sinhro impulsa (VS i HS) sa serijskim izlazom iz posmačnog registra (sl. 6) dobije se video izlaz. Tim signalom se može amplitudno modulirati VF oscilator, čiji izlaz se onda može direktno dovesti na antenski ulaz TV prijemnika.

ADRESIRANJE MEMORIJE ZNAKOVA

Za pamćenje znakova koji se prikazuju upotrebljena je memorija 1024x8 bita. Znakovi su u ASCII kodu, a 8-i bit određuje da li će dotični znak biti prikazan bijelo na crnoj podlozi ili obrnuto ("ekskluzivno ili" funkcija između serijskog izlaza iz posmačnog registra i 8-og bita). Kako je kapacitet memorije 1024 znaka, a na ekranu se može prikazati 16x32 = 512 znakova, memorija može pamtit i prošlih 16 redaka, tzv. "drugu stranicu". Za adresiranje 1024 znaka potrebno je 10 bitova. Najnižih 5 označuju znak u redu, slijedeća 4 predstavljaju redni broj reda, a najviši adresni bit određuje da li se prikazuje

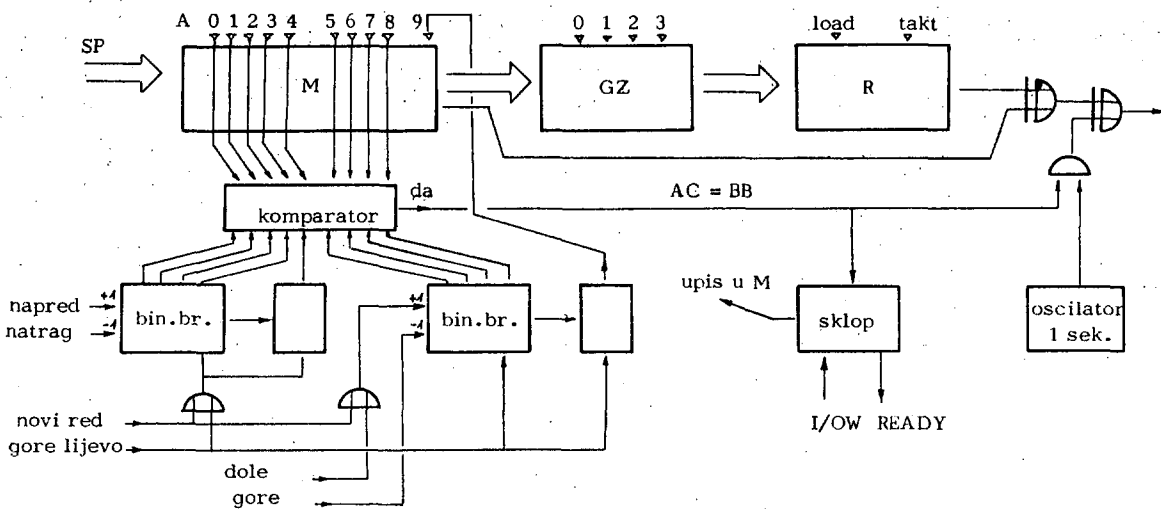
tekuća ili prošla stranica. Adresa znaka u redu je zapravo sadržaj prvog binarnog brojlara sa sl. 5. : prva 3 bita određuju koja od 8 točaka u jednoj od 16 linija pojedinog znaka se upravo ispisuje, a slijedećih 5 bitova definira jedan od 32 znaka u jednom od 16 redova. Drugo binarno brojilo sa sl. 5 pokazuje vertikalni otklon kod ispisa. Sa prva 4 bita određuje jednu od 16 linija unutar pojedinog znaka, a sa druga 4 bita definira jedan od 16 redova.

Ovakva organizacija adresiranja memorije znakova je vrlo jednostavna, a ujedno dopušta i izvedbu grafičkog terminala rastera 256x256. U tom slučaju treba adresirati pojedinu točku, a ne znak. Adresa točke se sastoji od 8+8 bitova (za horizontalni i vertikalni otklon). Potrebna je memorija kapaciteta 2^{16} bita. Takva memorija direktno se može adresirati sa dva 8-bitna brojlara sa sl. 5. Radi lakše komunikacije sa procesorom bolje je memoriju organizirati kao $2^{13} \times 8$ bita (8Kx8 bita). U tom slučaju opet se koristi posmačni registar za ispis 8 po 8 bitova, kao i u alfanumeričkom terminalu.

RAD POD KONTROLOM MIKRORAČUNALA

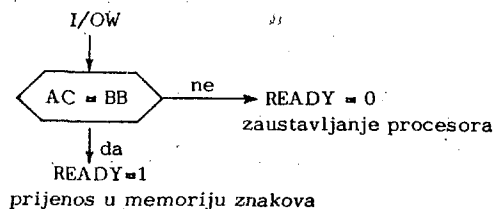
Klasični alfanumerički terminal redovno sadrži i tastaturu, preko koje se znakovi mogu direktno upisivati u memoriju znakova. Ovdje opisani terminal nema priključak za tastaturu. Prijenos podataka od bilo kojeg izvora znakova do memorije znakova u terminalu mora se vršiti preko mikroracunala. Za sve kontrolne funkcije (brisanje memorije znakova, invertiranje pojedinih znakova), kao i za kontrolu cursora (cursor gore, dole, lijevo, desno, u lijevi gornji kut) potrebna je programska podrška. Brojilo koje sadržava adresu cursora izvedeno je hardware-ski (sl. 7.). To brojilo sastoji se iz dva dijela : oba se nezavisno mogu inkrementirati i dekrementirati. Prvi dio (5 bita) definira jedan od 32 znaka u redu, a drugi (4+1 bit) definira jedan od 16 redaka i stranicu. Promjena stanja ovih brojlara vrši se izlaznom naredbom mikroracunala : npr. za skok na početak novog reda potrebno je 1) resetirati prvi dio i 2) inkrementirati drugi dio. Između brojlara za adresu cursora i binarnih brojlara koja adresiraju memoriju znakova postoje komparatori. Kad su obje adrese jednake znači da je na ekranu TV prijemnika upravo u toku ispisivanje znaka na čijem mjestu se nalazi i cursor. Na tom mjestu se tada ispisuje znak cursora (npr. invertira se postojeći znak, kao na sl. 7.).

Kad računalo treba u terminal upisati slijedeći znak (na adresu zadanu cursorom) potrebno je memoriju



Sl. 7. Drugi dio sheme terminala

znakova privremeno adresirati iz brojila za adresu cursora, kao što je bilo prikazano na sl.2. To zahtijeva ugradnju 10-bitnih digitalnih sklopki. Ovaj dodatak može se potpuno izbjeći na slijedeći način: kad procesor počne sa izvršavanjem slijedeće naredbe (I/OW) za upis novog podatka u memoriju znakova, pomoću postojećeg komparatora se testira da li je adresa na memoriji znakova jednaka adresi cursora (sl.8). Ako nije, pomoću READY ulaza u mikroprocesor privremeno se zaustavlja rad mikrorachunala, sve dok se te dvije adrese ne izjednače. Tada se ponovo dozvoli rad procesoru, pa se i započeta izlazna naredba kompletira.



Sl. 8.

Maksimalno vrijeme koje procesor može biti zaustavljen radi upisa pojedinog znaka je približno vrijeme ispisa jedne poluslike na TV prijemniku, dakle 1/50 sekunde. Interesantno je da je vrijeme potrebno za izmjenu svih 16 redova na stranici, sa do 16 znakova u redu, također samo 1/50 sekunde. (Svaki red znakova sastoji se iz 16 linija koje se u TV prijemniku ispisuju jedna ispod druge, a za ispis svake linije potrebno je oko 50μs. U komparator ne ulazi podatak o broju linije unutar istog reda, pa se ispisom svake od 16 linija jednog reda može postići upis novog znaka bilo gdje u tom redu).

UPIS SLOVA Č, Š, Ž I Ć

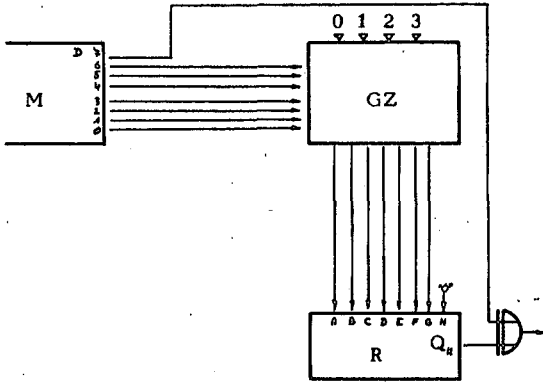
Da bi se terminal napravio pogodnim za rad s našim jezikom, generatoru znakova su dogradjeni sklopovi za transformiranje nekih rijetkih ASCII znakova u znakove C, S i Z, uz istovremeno generiranje znaka "ˇ". Modifikacija u standardnom ASCII kodu može se vidjeti iz tabele na sl.9.

ASCII KOD :	ASCII ZNAK:	NOVI ZNAK:	novi kod za KG :	generirani znak :	dobrani znak:
1x0 0011	C	-	-	C	-
1x1 0011	S	-	-	S	-
1x1 1010	Z	-	-	Z	-
1x1 1011	{, {	Č	1x0 0011	C	ˇ
1x1 1100	\, }	Š	1x1 0011	S	ˇ
1x1 1101], }	Ž	1x1 1010	Z	ˇ
1x1 1110	†, ~	Ć (*)	1x0 0011	C	ˇ

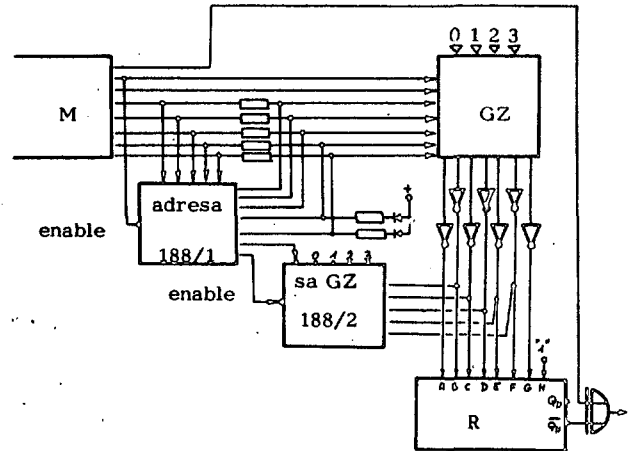
(*) Modifikacija Č samo u terminalima gdje znak † ne znači potenciranje !

Sl. 9. Izmjene u ASCII kodu

Na sl. 10a prikazano je povezivanje memorije znakova M, generatora znakova GZ i posmačnog registra R u terminalu bez navedene modifikacije. Prikazane su samo linije podataka. Na sl. 10b prikazane su izmjene koje je potrebno unijeti : prvo treba transformirati kod ASCII znakova [, \,], † u kod za ASCII znakove C, S, Z, Ć respektivno. To je najlakše ostvarivo upotrebom PROM memorije (SN 74188 : 32x8 bita, otvoreni kolektori na izlazu, može se programirati). Taj PROM (na shemi : 188/1) adresira se kodom iz memorije znakova. Djelovanjem izlaznih bitova PROM-a treba se na ulazu u generator znakova dobiti ASCII kod modificiran prema tabeli na sl.8. Jedna mogućnost izvedbe dana je na sl. 10b. Nakon što je KG primio modificirani ASCII kod potrebno je na dobivena slova C, Z i S dodati znak "ˇ". GZ na svom izlazu paralelno daje 7 bita jedne od 16 linija adresiranog znaka. Linija se izabere dovodjenjem 4-bitne adrese u GZ, a od brojila za vertikalni pomak (sl.5). Jednostavan način za ubacivanje u ispis jednog od znakova C, S, Z prikazan je na sl.10b. Druga PROM memorija istog tipa (188/2) adresira se sa ista 4 bita kao i GZ, adresa linije. Ulaz "enable" spojen je na izlaz 188/1, a to je u stvari kontrolni signal "ovom znaku treba dodati znak"ˇ". Izlazi iz 188/2 tada u "žičano ili" funkciji sa izlazima iz GZ formiraju kompletan znak Č, Š, Ž, ili Ć.



SI. 10a



SI. 10b

ZAKLJUČAK

Opisan je princip rada alfanumeričkog Terminala koji se po konstrukciji bitno razlikuje od dosadašnjih takvih uređaja. Osnovna prednost mu je jednostavna hardware-ska izvedba i niska cijena, a osnovni nedostatak je potreba za programskom podrškom, što ga čini nesamostalnim. Predviđen je za rad u malim mikroracunarskim sistemima, priključen direktno na sabirnice mikroracunala. Broj potrebnih komponenata je toliko reduciran da cijeli terminal

stane na jednu karticu standardnog evropskog formata (16x10 cm). To ga čini pogodnom izlaznom jedinicom u modularno gradjenom mikroracunalima. Svakako je interesantna i mogućnost ispisa slova Č, Š, Ž, Ć, te mogućnost nadogradnje terminala na repetitivnu grafiku, što bitno proširuje područja primjene.

neki aspekti uvođenja komunikacione mreže on-line real-time sistema beogradske banke

d. v. glušac

UDK 621.395.74:681.3:336.712(497.11 Bgd)

ELEKTRONSKI RAČUNSKI CENTAR BEOGRADSKJE BANKE
Grupa za razvoj komunikacionog dela ON-LINE REAL-TIME Sistema

U ovom članku ćemo diskutovati o razvoju komunikacione mreže i analizirati kriterijume oko izbora modema, kvalitet iznajmljenih linija i pouzdanost prenosa podataka. Takođe, ovde možete naći detaljan proračun pri prenosu podataka u slučaju 2-žične odnosno 4-žične veze.

SOME ASPECTS OF COMMUNICATION NETWORK DESIGN FOR ON-LINE REAL-TIME SYSTEM OF BEOGRADSKA BANKA. In this article we shall discuss the development of communication network and analyzing some criteria for selecting modems, quality of leased lines and reliability of data transfer. Also, you can find here very detailed calculating of line facilities for data transfer concerning 2-wires and 4-wires mode.

1.1. Opšte o komunikacionoj mreži

U skladu napora za uvođenje u rad ON-LINE REAL-TIME sistema za potrebe Beogradske banke, iz niza problema koje u tom pravcu treba rešiti, ovde ćemo se osvrnuti na aspekte razrešavanja problema oko ostvarenja prenosnih puteva, tj. izgradnje komunikacione mreže pomenutog ON-LINE REAL-TIME sistema. Pod prenosnim putem se smatra veza: modem - linija - modem, i u većim zemljama on se kao takav iznajmljuje od nacionalnih PTT ustanova, što kod nas nije slučaj. Iz prethodnog slede problemi izbora vrste prenosnog puta, vrste modema, kao i obezbeđenje pouzdanosti prenosa, koji se kao problemi nabacuju radnoj organizaciji koja uvodi takav ON-LINE sistem. Ovde ćemo razmatrati nekoliko važnijih problema i dati svoj doprinos ovoj problematici, jer se očekuje da sve više radnih organizacija pridje uvođenju ON-LINE sistema i svi će se susretati sa ovim problemima.

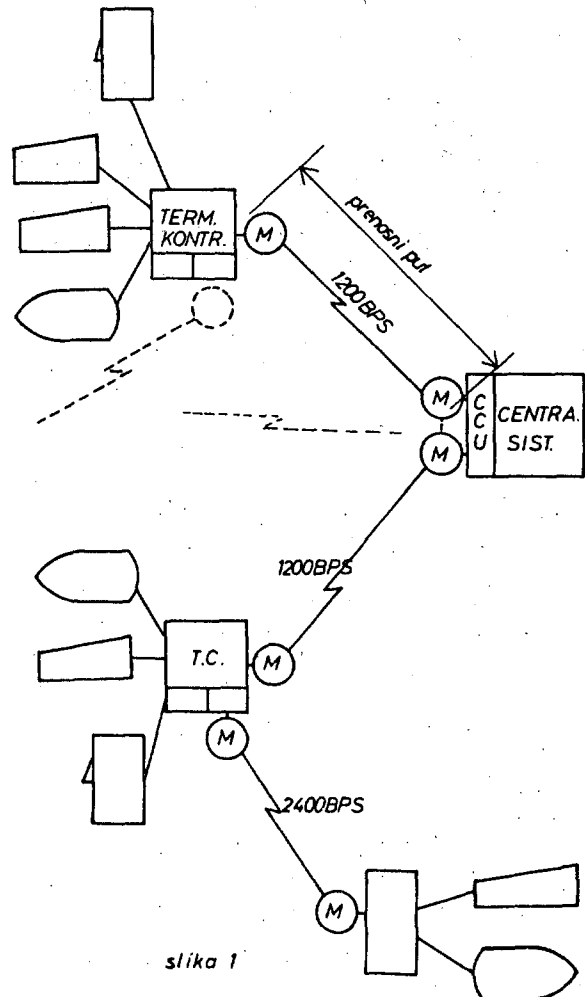
1.2 Prikaz mreže ON-LINE REAL-TIME Sistema

Ovaj prikaz mreže (slika 1.) tj. dela mreže ON-LINE REAL-TIME sistem smatrajmo kao tipičan, i uočimo samo prenosne puteve i vrste. Imamo 2 vrste prenosnog puta:

- prenosni put od 1200 BPS (oko 60 linija)
- prenosni put od 2400 BPS (oko 60 linija)

gde je vrsta prenosa pri brzini 1200 BPS asinhroni, a kod 2400 BPS sinhroni način. Mreža je zvezdastog tipa ("POINT-TO-POINT"), a ON-LINE Sistem je "sistem upita i odgovora ("ENQUIRY SYSTEM").

Na slici 1 pod T. C. se smatra terminal kontroler, M - modem, CCU - komunikaciona kontrolna jedinica pri centralnom procesaru.



slika 1

Pri uvođenju mreže ovog sistema javlja se nekoliko problema koje treba rešiti i to:

- definisanje vrste veze - iznajmljena ili komutirana
- koje primarne karakteristike mora veza da poseduje za prenos podataka
- izbor 2-žične odnosno 4-žične veze
- izbor modema,
- obežbeđenje potrebne pouzdanosti.

2.1 Izbor vrste veze

Dilema oko izbora vrste veze je ipak mala, tj. svi sistemi u "realnom vremenu" ("REAL-TIME") isključivo traže iznajmljene veze. Mi ćemo ovde napraviti prikaz u kome će se videti razlika između ove dve vrste veza, da bi se videla "cena" uvođenja jednog sistema u "realnom vremenu".

- troškovi iznajmljivanja jedne komutirane veze čine oko 13% cene jedne iznajmljene veze (2),
- stepen iskorišćenja iznajmljene veze je veoma nizak, ali je opravdan značajem jednog sistema u "realnom vremenu",
- dozvoljen odnos grešaka na iznajmljenoj vezi je oko $1 \cdot 10^{-5}$, a komutirane veze oko $1 \cdot 10^{-3}$,
- u uslovima naše PTT mesečna cena iznajmljene 2-žične veze je 900 NDIN, ako veza prelazi područje jedne telefonske centrale, a 1800 NDIN ako prelazi dve telefonske centrale, itd.

3.1 Izbor primarnih karakteristika linija

U cilju sagledavanja primarnih karakteristika linija koje će se koristiti za prenos podataka, na zahtev Beogradske banke, naše PTT preduzeće je izvršilo merenja primarnih karakteristika nekih 12 linija (koje su sad komutirane veze i koriste za telefonski saobraćaj) i ovde ćemo izvući najbitnije karakteristike iz detaljnog tehničkog izveštaja.

Od ukupno 12 pravaca, što lokalnih što međujunskih, uzećemo u analizu 3 pravca u području grada, gde se očekuje preko 90% pravaca mreže. Za data tri pravca, prema slici 2, uvedene su skraćene oznake R_{1-3} .

R_1	relacija: V. MIŠIĆA 39 - A. NIKOLIĆA 2
R_2	relacija: NJEGOŠEVA 7 - KOLARČEVA 5
R_3	relacija: NJEGOŠEVA 7 - B. REVOLUCIJE 24

slika 2

Sva merenja se dele u dve grupe:

- merenja telefonskog tipa
 - slabljenja na 800 Hz,
 - izobličenje slabljenja u opsegu 300 - 3400 Hz,
 - izobličenje kašnjenja grupe u opsegu 300-3400 Hz,
 - impulsni šum,
- merenja telegrafskog tipa
 - stepen greške na bitovima,
 - stepen greške na blokovima

su vršena prema CCITT - V53 (poglavlje 2.1 gde je za 1200 BPS odnos greške po bitu $5 \cdot 10^{-5}$).

Merenja telegrafskog tipa su vršena na brzini od 1200 BPS, modem je tipa "ITT 1200", interval merenja je 15 min, izlazni nivo je - 6 dBm.

	vreme merenja	broj pogrešno primljenih		broj prenetih		stepen greške	
		bitova	blokova	bitova	blokova	bitovi	blokovi
R_1	10 ²⁵ -10 ⁴⁰	0	0	108·10 ⁶	2113	$\geq 1 \cdot 10^{-6}$	$\geq 1 \cdot 10^{-4}$
R_2	08 ⁴⁵ -09 ⁰⁰	0	0	108·10 ⁶	2113	$\geq 1 \cdot 10^{-6}$	$\geq 1 \cdot 10^{-4}$
R_3	11 ⁰⁵ -11 ²⁰	0	0	108·10 ⁶	2113	$\geq 1 \cdot 10^{-6}$	$\geq 1 \cdot 10^{-4}$

slika 3

I na brzini od 2400 BPS, modem je tipa "TTT 2400", interval merenja 15 minuta, a izlazni nivo je - 6 dBm.

	vreme merenja	broj pogrešno primljenih		broj prenetih		stepen greške	
		bitova	blokova	bitova	blokova	bitovi	blokovi
R_1	10 ⁵⁵ -11 ¹⁰	0	0	2,16·10 ⁶	4227	$\geq 1 \cdot 10^{-6}$	$\geq 1 \cdot 10^{-4}$
R_2	09 ³⁰ -09 ⁴⁵	0	0	2,16·10 ⁶	4227	$\geq 1 \cdot 10^{-6}$	$\geq 1 \cdot 10^{-4}$
R_3	10 ⁰⁵ -10 ²⁰	0	0	2,16·10 ⁶	4227	$\geq 1 \cdot 10^{-6}$	$\geq 1 \cdot 10^{-4}$

slika 4

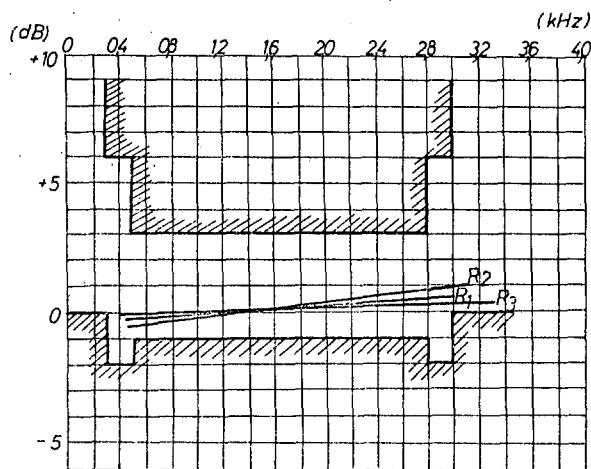
Merenja telefonskog tipa

a) merenje impulsnog šuma u intervalu 15 minuta.

		PRAG MERENJA U (dBm)	
		-50	-40
R_1	vreme merenja	12 ³⁰ -12 ⁴⁵	12 ⁴⁵ -13 ⁰⁰
	broj impulsa šuma	208	17
R_2	vreme merenja	11 ³⁰ -11 ⁴⁵	10 ⁴⁵ -11 ⁰⁰
	broj impulsa šuma	120	320
R_3	vreme merenja	11 ⁴⁰ -11 ⁵⁵	11 ⁵⁵ -12 ¹⁰
	broj impulsa šuma	37	0

slika 5

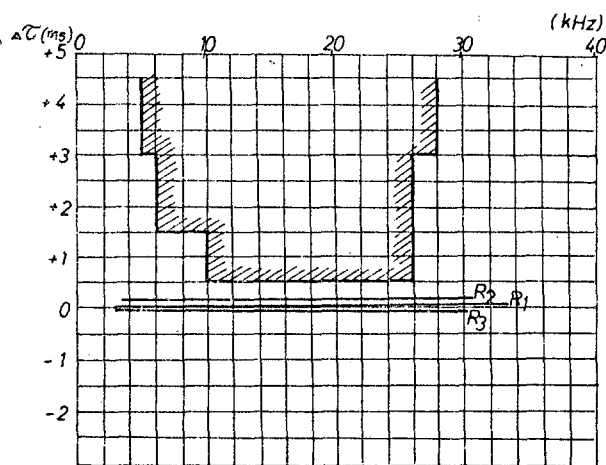
b) Merenje promene slabljenja



slika 6

Vidi se iz slike 6 da su merene vrednosti u okviru gabarita propisanih sa CCITT - M102, prijemni nivo je -3 dB, opseg merenja od 300 - 3400 Hz, referentna frekvencija je 2 KHz.

c) Merenje izobličenja kašnjenja grupe



slika 7

3.2 Analiza rezultata merenja

Može se konstatovati da su svi primarni parametri linijskih karakteristika u okviru CCITT propisa (M-102). Odnos greške na bitovima, jedan od glavnih parametara za ocenu kvaliteta voda koji se koristi za prenos podataka, takodje je zadovoljavajući prema V. 53 CCITT propisa. Stanje iznajmljenih linija kod naše PTT se može smatrati dobrim za brzine prenosa do 1200 i 2400 BPS!

3.3 Dodatna merenja

Pored merenja koje PTT preduzeće vrši pred iznajmljivanje linija, korisno bi bilo meriti još:

- beli šum,
- kratkotrajne prekide,
- nagle promene nivoa,

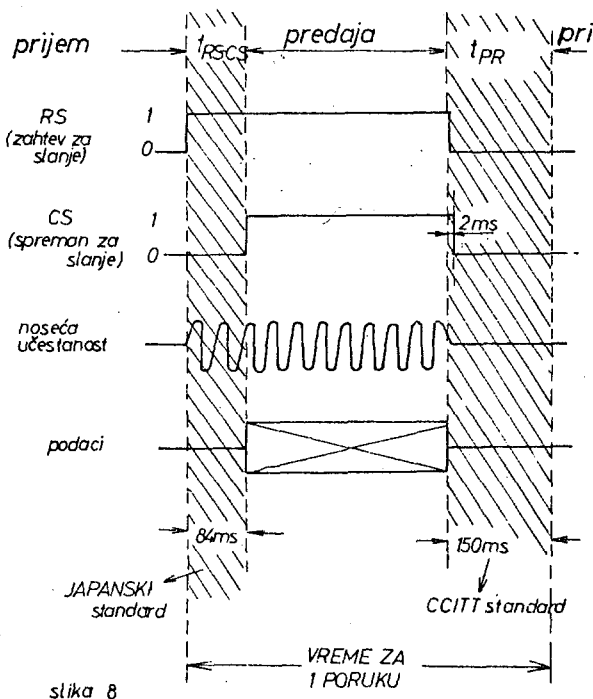
Kod merenja impulsnog šuma je uzeto najnepogodnije vreme merenja od 10 - 13 časova, jer su tada i najveći impulsní šumovi (od raznih električnih mašina, uređaja itd.).

4. Dileme kod izbora 2-žične odnosno 4-žične veze

Sama dilema kod izbora 2-žične odnosno 4-žične veze je u stvari i dilema prenosa podataka u polu-dupleksu odnosno punom dupleksu. Kao polaznu osnovu uzimamo da oprema ON-LINE sistema pruža mogućnost izbora ova dva načina rada i da naredne analize i procene treba da nas dovedu do odluke.

4.1. Vremenski dijagram prenosa podataka u 2-žičnoj vezi

Analizirajući vreme potrebno za prenos jednog broja karaktera u 2-žičnoj odnosno 4-žičnoj vezi, dolazimo do osnovne razlike koja govori u prilog 4-žične veze. Ova razlika nastaje kod 2-žične veze zbog potrebe da se "menja smer prenosa" i da je za to potrebno određeno vreme, dok kod 4-žične veze imamo izdvojene pravce prenosa: prijem i predaja. Pokušajmo doći do tog vremena preko kraće analize prenosa: npr. 720 transakcija /čas u oba načina rada. I ovde treba imati da je analiza vezana za jedan određeni sistem, ovde je to oprema ON-LINE REAL-TIME sistema japanske firme "FUJITSU".



slika 8

Posmatrajući vremenski dijagram prenosa podataka vidimo 2 šrafirane zone:

1) vreme između uspostavljanja signala RS (poznat u CCITT standardima kao "Zahtev za slanje" ili na engleskom jeziku "REQUEST TO SEND") i signala CS ("Spreman za slanje" - "CLEAR TO SEND") usvojeno je od strane japanske PTT na 84 msec.

2) Vreme promene smera prenosa koje po CCITT standardu (konferencija V. 23 - 8. f) usvojeno na 150 msec (u literaturi se sreće kao "SQUELCH TIME").

Dakle, vreme potrebno za prenos proizvoljne poruke se uvećava za 234 ms (84 + 150), za slučaj 2-žične veze u odnosu na 4-žičnu, koja to vreme ne traži. Nadjimo koliko se karaktera može preneti za ovo vreme, pri brzini rada od 1200 BPS, što je potrebno za dalju analizu. Imajući u vidu da pomenuti sistem radi u asinhronom načinu (1 karakter sadrži 10 bitova)

dolazimo do brzine od 120 kar/sek (1200 BPS/10 = 120 kar/s. Neka sa N označimo traženi broj karaktera i uspostavimo relaciju:

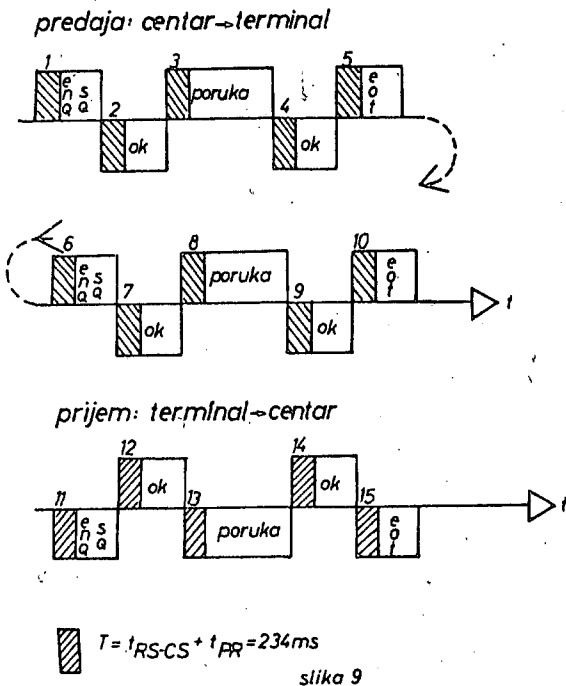
$$\frac{120 \text{ kar}}{1000 \text{ ms}} = \frac{N \text{ kar}}{234 \text{ ms}}$$

$$N = \frac{120}{1000} \cdot 234 = 24,08 \text{ kar (za 234 ms)}$$

Znači, povećanje vremena prenosa poruke zbog korišćenja 2-žične veze (odnosno polu- dupleks veze) iznosi 234 ms ili 28,08 karaktera.

4.2 Dijagram komunikacione procedure po jednoj transakciji

U cilju obavljanja jedne transakcije, za sistem koji obradjuje, koristi se sledeća komunikaciona procedura data na sl. 9.



Po ovoj komunikacionoj proceduri iz centra se šalju dva puta poruku terminalu, a prima samo jedanput uz obavezne kontrolne poruke (ovde se nećemo u njih udubljivati). Osnovne poruke uz svaku od poruka (ima ih ukupno 15 i svaka je obeležena rednim brojem) su povećanje vremena zbog 2-žične veze (u prethodnom paragrafu detaljno objašnjeno) od 234 ms. Dakle, za celu transakciju se gubi $15 \times 234 \text{ ms} = 3510 \text{ ms}$, za koje se vreme može preneti 421,2 karaktera pri brzini od 1200 BPS!

4.3 Poredbeni primer upotrebe 2-žične odnosno 4-žične veze

U tabeli sa slike 10. vidimo rezultate ove analize, čiju smo postavku dali na prenosu 720 kar/sat pri brzini od 1200 BPS.

Uradimo proračun linijskog faktora opterećenja za slučaj 4-žične veze, dok je za 2-žičnu identičan postupak. Znamo da je potrebno preneti 720 transakcija na sat, a ako pretpostavimo srednji broj karaktera po transakciji, 270 karaktera dobijamo da treba preneti 194.400 karaktera na sat za 720 transakcija.

Linijski faktor opterećenja definisan je sa:

$$L. F. O. = \frac{\text{broj kar. koje prenosimo na sat}}{\text{maks. br. kar. za datu brzinu prenosa}} \times 100 (\%)$$

I potražimo maksimalni broj karaktera za brzinu od 1200 BPS (označimo je sa T)

$$T = 1200 \text{ BPS} = \frac{1200 \text{ BPS}}{10} = 120 \text{ kar/sek}$$

da bi dobili koliko karaktera prenesemo na sat pomnožimo:

$$120 \frac{\text{Kar}}{\text{sek}} \times 3600 = 432.000 \frac{\text{kar}}{\text{sek}}$$

I dalje sledi:

$$L. F. O. = \frac{194.400}{432.000} = 0,452 \times 100 = 45,2\%$$

Ovaj faktor nam govori da je 4-žična veza iskorišćena za dati broj karaktera i datu brzinu sa oko 45% svoje ukupno propusne moći.

Za slučaj 2-žične veze, uz isti broj karaktera na sat i brzinu prenosa, povećava se srednji broj karaktera na sat sa 270 na 690 kar (zbog dodatnog vremena za promenu smeru prenosa pri jednoj transakciji od 3510 ms za koje vreme se prenese 420 karaktera - vidi poglavlje 2. 2)

Ukupan broj karaktera koje treba preneti na sat iznosi:

$$(270 + 420) \times 720 = 497.325 \text{ kar/sat}$$

a odatle sledi:

$$L. F. O. = \frac{497.325}{432.000} \times 100 = 115,0\%$$

Znači, dolazimo do nemogućeg slučaja, jer LFO mora biti manji od 1 (odnosno 100%).

	4- žična	2- žična
Broj transakcija na sat	720 TR/sat	720 TR/sat
Sred. br. kart. po transakc.	270 kar/tr	270+420 = 690 kar/tr
Ukup. br. kart. na sat	194.400	497.325
Linijski faktor opterećenja	45,2%	115,0%

← nemoguć slučaj

Dakle, u cilju dovođenja LFO u normalne granice moramo ili smanjiti broj transakcija koje želimo preneti na sat ili povećati brzinu prenosa na liniji.

4.4 Testiranje prenosnog puta

Upotreba 4-žične veze daje još jednu prednost nad 2-žičnom, a to je mogućnost testiranja linije i udaljenog modema u slučaju 4-žične veze, što nije slučaj u 2-žičnoj vezi.

5.1 Izbor modema

Izbor modema je nešto lakši deo posla, jer su modemi, kao telekomunikacioni uređaji, veoma striktno definisani karakteristikama, funkcijama kao i svih ulazno-izlaznih žila. Dakle, modem za brzinu prenosa treba od 1200 BPS da zadovolji:

- tehničke uslove za modem 600/1200 BPS (objavljene u PTT Vesniku, br. 15, 1976),
- da omogućuje jednostavno održavanje,
- da je pouzdan u tehnološkom smislu,
- da poseduje atest PTT preduzeća.

Modem za brzinu od 2400 BPS treba uglavnom da zadovolji iste uslove, a da odgovara tehničkim uslovima ("PTT Ves-

nik" broj 5, 1977).

5.2 Dizajn modema

U našem slučaju primene od opcionih sklopova koje svaki modem može da ima, mi smo se odlučili na sledeći dizajn modema:

- modem ne poseduje povratni kanal od 75 BPS (sistem prenosa ga ne zahteva),
- modem ne poseduje "eho supresor" (potreba za njim javlja se na većim brzinama prenosa),
- modem poseduje "detektor kvaliteta linije",
- modem ima mogućnost jednostavne ugradnje "kompromisnog korektora amplitudne i fazne karakteristike".

6.1 Pouzdanost prenosnog puta

Ovde ćemo pobrojati osnovne činioce pouzdanosti prenosnog puta:

- proizvesti (izvršiti izbor) modem visoke pouzdanosti kako komponenti tako i celog uređaja,
- obezbediti dobro testirane linije od strane PTT,
- koristiti 4-žičnu vezu (manje smetnje),
- ako je moguće obezbediti rezervne linije,
- realizovati testiranje i snimanje linijskih karakteristika na

jednostavan način (lakše je grešku lokalizovati "na nižem nivou složenosti", tj. lakše je liniju lokalizovati kao izvor greške, nego istu tražiti u složenim ispitivanjima sa celim sistemom).

Kao poseban parametar pouzdanosti u literaturi (1) se sreće podatak da se dozvoljava "ispad" linije do 2% vremena korišćenja linije.

7.1 Zaključak

Problemi koje smo ovde analizirali su ti sa kojima će svako da suočava pri uvodjenju komunikacione mreže svog ON-LINE sistem, pa se nadam da će u ovom radu naći poneki odgovor za sebe.

Literatura:

1. JAMES MARTIN: "Design of real-time computer systems", Prentice - Hall, New Jersey
2. M. ŽIVKOVIĆ: "Prenos podataka u Jugoslaviji preko PTT linija" - "Praksa" broj 9, 1972. godine
3. PTT Vesnik: "Tehnički uslovi za MODEM 2400 BPS" PTT Vesnik, broj 5, 1977.
4. PTT Vesnik: "Tehnički uslovi za MODEM 1200 BPS" PTT Vesnik, broj 15, 1976.
5. J. OBA: "Technical report about data transmission" "FUJITSU", Papers, Nov. 1977.

univerzalni serijsko vhodno izhodni modul

n. papić
m. kovačević

UDK 681.3 - 181.4:681.327.2

Institutt "Jožef Stefan" Ljubljana

Članek opisuje praktične prijeme pri projekti ranju vhodno-izhodnih modulov za mikroročunalnike. Podana je električna shema modula za priključitev Baudotovega printerja na mikro računalnik Iskradata 1680. Natančno je opisano funkcioniranje modula in programska oprema ki je potrebna za vključitev omenjene omenjene periferne naprave v obstoječo programsko opremo tega mikro računalnika.

UNIVERSAL, SERIAL INPUT/OUTPUT MODULE - In this paper some particular solutions for the problems which appear during the development and realisation of I/O interfaces for microprocessors are outlined. Figure 9 shows a module for Baudot printer and microprocessor Iskra Data 1680. We have explained the structure for interface logic. An interface software package which is compatible with the existing monitor programme for this microprocessor is included.

1. UVOD

Mikro računalnik (kratko uR) potrebuje za komunikacijo z uporabnikom ustrezno periferijo (izhodno-vhodne enote). Prejkoslej bodo uR postali del našega vsakdanjega življenja, zato je potrebno omogočiti uporabnost uR na različnih mestih in nalogah z uporabo najbolj enostavne ali obstoječe periferije. Izbor perifernih enot je pomemben, ker pogosto odločilno vpliva na ceno mikroročunalniškega sistema. Zato smo tudi prišli na idejo, da uporabimo navadni printer z Baudot-ovim kodom za vhodno-izhodno enoto.

Običajne periferne enote uR so n.pr. D/A in A/D pretvorniki, čitalniki in luknjalniki papirnega traku, video zasloni, magnetno-tračne enote (kasete) in diskete. Da bi informacije, ki jih vnašamo, ali dobimo, bile uporabne, za človeka in tudi za stroj, je med periferno enoto in uR potreben vmesnik. Ta prenaša tudi podatek o možnosti prenosa informacije in uravnava razliko med časom posredovanja informacije uR in periferne enote.

Vmesnik ima štiri osnovne funkcije: dekodiranje ukaza pri enotah, kjer imamo razen čitanja tudi druge operacije, n.pr. premikanje traku; izbiranje periferne enote, pri več perifernih enotah z dekodiranjem naslova; sinhronizacijo izmenjave informacij med uR in periferno enoto; za vse te tri funkcije uravnavanje časa in kontrolo.

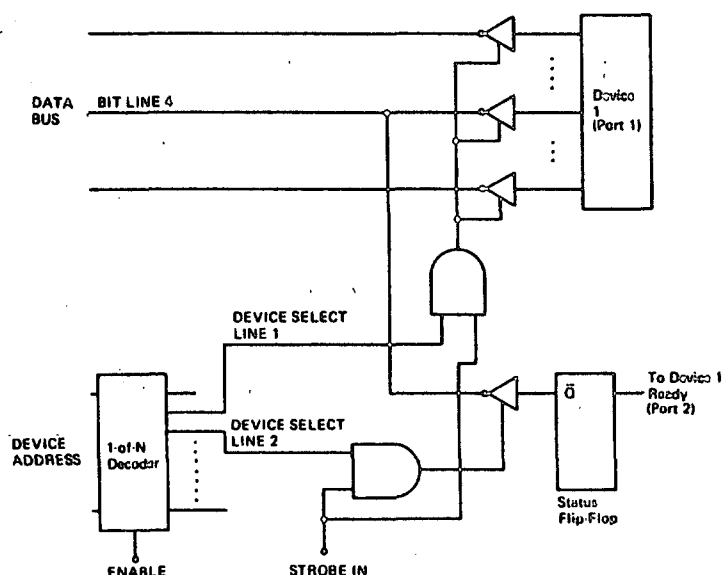
Izmenjava informacij se lahko opravi na dva načina - z mikroročunalniškim programom kontroliramo prenos ene besede v trenutku, ali z direktnim dostopom v pomnilnik (DMA), ki ga kontrolira periferne enote in omogoča prenos večjega bloka informacij.

Prvi način je mogoč pri vseh mikro računalnikih. Kontroliran je s programom in je lahko sinhronski, asinhronski ali s programskimi prekinitvami.

Sinhronski način se uporablja pri perifernih enotah, ki imajo točno določen čas prenosa. Periferne enote mora biti pripravljena za prenos in izvrševati prenos informacij brez posebnega določevanja časa instrukcije procesov.

Asinhronski prenos uporabljamo, ko je potrebno preveriti stanje periferne enote pred izvajanjem vhodno-izhodne operacije. Potrebujemo štiri korake:

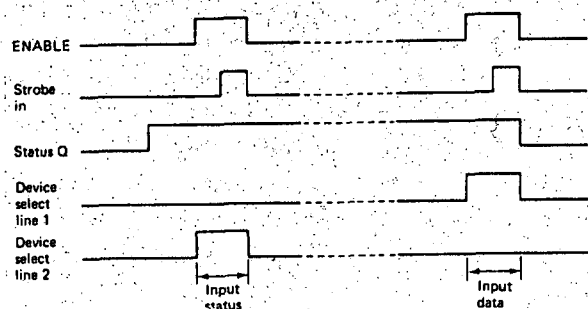
1. Kontrola stanja periferne enote
2. Aktiviranje periferne enote, če je pripravljena
3. Izvrši se prenos (vhod-izhod)
4. Deaktiviranje periferne enote.



Slika 1. Ponazoritev povezave periferne enote na vodilo

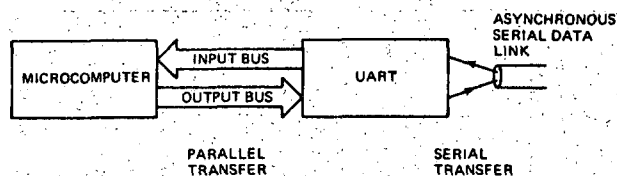
Ponazoritev povezave periferne enote na vodila je dana na sliki 1.

Slika 2 je tipičen časovni diagram za asinhronski prenos.



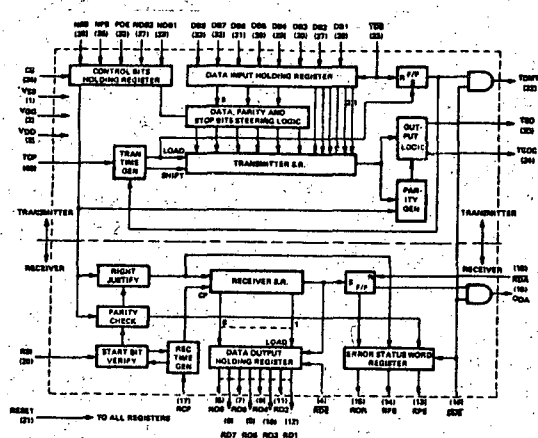
Slika 2. Časovni diagram za asinhronski prenos

Asinhronski prenos je uporaben v primerih, ko obstaja časovno nesorazmerje med uR in periferno enoto. uR najverjetneje čaka, da bo periferna enota PRIPRAVLJENA. Zato tak prenos ne uporabljamo pri kontroli procesov, kjer bi morali zadrževati kontrolni signal, dokler čakamo na signal.



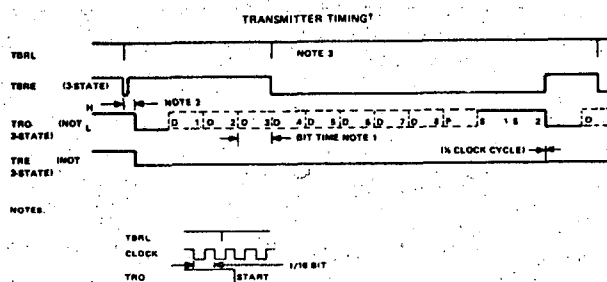
Slika 3. Blok diagram uporabe UART

Periferne enote sprejemajo ali oddajajo podatke v vzporedni ali zaporedni obliki. V našem primeru periferna enota prenaša podatke v zaporedni obliki. Zato je bilo potrebno v vmesniku uporabiti vzporedno-zaporedni pretvornik signalov. Posebno integrirano vezje za take transformacije je UART (Universal Asynchronous Receiver-Transmitter), tj. univerzalni asinhronski pretvornik. Bločni diagram je prikazan na sliki 3. Na sliki 4. pa vidimo diagram sistemske uporabe.



Slika 4. Blok diagram UART S1883

Zaporedna beseda ima lahko startni bit, 5-8 podatkovnih bitov, kontrolni bit za parnost, enega in pol ali dva ustavitvena bita. Število podatkovnih bitov je programirano. Kontrola parnosti se lahko nastavi ali pa ne upošteva. Število ustavitvenih bitov tudi lahko izberemo. Dela za pretvorbo vzporednih v zaporedne in zaporednih v vzporedne podatke sta ločena in delujeta simultano ali ločeno eden za drugim. Kontrolni del je skupen za zaporedni in vzporedni pretvornik. Prenosni del sprejema vhodne podatke na vhodih D 1 -D 8. Podatek se shrani v prenosni vhodni register in nato v prenosni register, kar se kontrolira s signalom TDS. Serijski podatek se prenese na izhod TSO ob impulzu TCP (impulz takta). Pri tem se nastavi tudi kontrolna bita TBMT in TEOC, ki nam povesta, ali je podatek prenesen iz vhodnega prenosnega registra v prenosni register in ali je prenos zaključen, ko se lahko vstavlja že novi podatek. Časovni potek tega prenosa je podan na sliki 5.

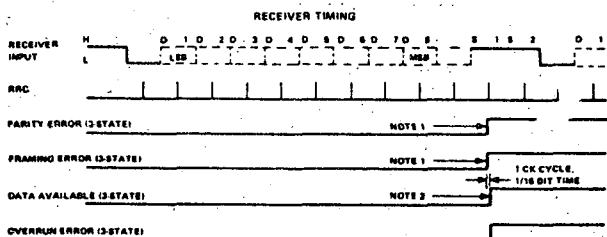


Slika 5. Časovni potek prenašanja

Sprejemni del sprejema podatke iz serijskega vhoda RSI. Podatki, ki se shranijo z 1/16 periode takta RCP, se prenašajo iz sprejemnega registra v sprejemni izhodni register (D01 - D08). Bit RDE omogoči izhod podatkov iz tega registra.

V sprejemnem delu se generira več kontrolnih bitov: RPE za kontrolo parnosti, RFE kontrolira zaključitev sprejema podatkov s STOP bitom in ROR, ki se nastavi, ko je nov znak sprejet v sprejemni izhodni register dokler je prejšnji bil še v registru. To se zgodi, če bit ODA (ki ugotavlja ali je cel znak sprejet in prenesen v sprejemni izhodni register) ni v pravem trenutku spremenil stanja, kar spet povzroča bit RDA (uči podatek, da je znak dostopen).

Časovni potek sprejema je podan na sliki 6.



Slika 6. Časovni potek sprejemanja

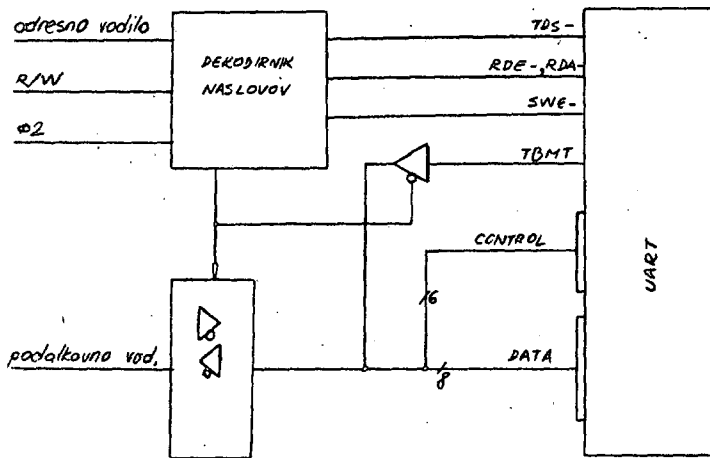
Status periferne enote je določen z že opisanimi biti TBMT, TEOC, RFE, ROR, ODA.

Krmiljenje operacij periferne enote se opravlja preko kontrolnih bitov: SWE - omogoča čitanje statusa periferne enote in s tem tudi sproži odločitev o sprejemu ali prenašanju; RDE - ob prihodu veljavnega signala na ta vhod je omogočen sprejem (podatek gre v izhodni sprejemni register); TDS - ki časovno krmili prehod podatkov v prenosni vhodni register (sprejem vzporednih podatkov v UART). Vsi ti kontrolni signali se v vmesniku generirajo na poseben način, ki je določen z logično zasnovno vmesnika. RESET signal se pripelje na UART mikroračunalniško direktno. TCP in RCP signala generirata potrebne periode. Serijski vhod SI in serijski izhod SO sta zaključena z 20 mA tokovno zanko.

2. LOGIČNA ZASNOVA VMESNIKA

Vhodni podatki vmesnika so signali z adresnega in podatkovnega vodila, časovna signala Φ_2 in CLK1, VMA (Valid Memory Address), RESET, R/W in serijski vhod za podatke s teleprinterja. Z dekodiranjem adrese izberemo signal za čitanje statusa, ali za pisanje ali čitanje podatkov. V našem primeru sodeluje DATA signal pri formiranju signala, ki povzroči pisanje ali čitanje podatkov. Signal CNT sodeluje pri formiranju signala za čitanje statusa. V konjunkciji teh signalov z R/W in Φ_2 dobimo končno RDA (RDE ob katerem se izvede čitanje podatkov; TDSrabi za pisanje podatkov in SWE za odčitavanje statusa.

Prenos vzporednih podatkov na podatkovno vodilo krmilimo v prvem primeru s pogojem, ki nastopi ob istočasnem prihodu signalov Φ_2 , VMA in R/W in drugič s signalom iz UART, ki daje obvestilo o tem, ali je končana oddaja znaka in ustavitvenih bitov.



Slika 7. Blok shema vmesnika za priključitev navadnega teleprinterja na mikro računalnik

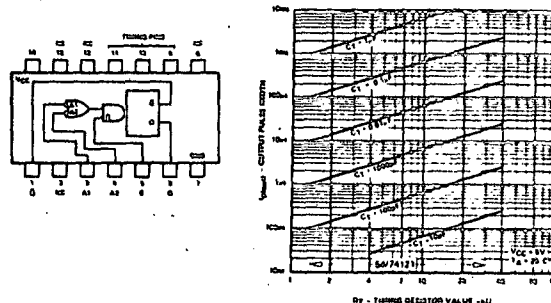
3. ELEKTRIČNA SHEMA VMESNIKA

Pri realizaciji nastanejo težave zaradi neusklojenosti različnih signalov; zato je bil uporabljen en multi vibrator za "podaljšanje" signala pri pisanju podatkov (S74121).

Na sliki 8 je logični diagram tega multivibratorja in časovni diagram. Vhoda A1 in A2 sta vhoda, ki sprožita aktivnost, ko eden od obeh ali oba preideta v logično 0, ko je B na logični 1. Zahtevani impulz dobimo, ko vezemo kondenzator C_T med pine 10 in 11. Zunanjo upornost R_T vezemo med pine 9 in 14. Eno od teh vrednosti določimo poljubno (za C_T so meje med 10pF in 10 uF, za upornost pa 2-40 K) drugo pa izračunamo iz enačbe za časovno dolžino impulza :

$$t_p = 0.693 \cdot C_T R_T$$

Za hitrost 45,45 baudov popravljamo dolžino impulza na $1,3 \times 10^{-3}$ s. Zaradi uporabe teleprinterjev z različnimi hitrostmi smo uporabili za R_T nastavljivi potenciometer in že vezali preko stikal priključitev na printerje s hitrostmi 50 in 100 baudov.



Slika 8. Logični in časovni diagram multivibratorja

Ker mora biti vhodni takt oddajnika (transmitter) 16x hitrejši, kot je zahtevana hitrost v baudih v serijskem izhodu, je za to funkcijo uporabljen časovnik.

To je astabilni multivibrator, oscilator, ki ima frekvenco neobremenjenega cikla in delovni cikle kontroliran z zunanjo upornostjo in kondenzatorjem. Z izbiranjem dveh vrednosti je bilo treba natančno določiti frekvenco takta po enačbi

$$f = \frac{1.44}{(R_A + 2R_B) \cdot C} = \frac{1}{T}$$

kjer sta R_A in R_B zunanji upornosti in C zunanja kapacitivnost.

Električna shema celega vmesnika je podana na sliki 9.

4. PROGRAMSKI DEL VMESNIKA

Programski del je izveden strukturirano. Narejeni so podprogrami za čitanje in izpisovanje Baudotovega serijskega znaka ter za konverzijo ASCII v Baudotov kod in obratno. Konverzijski rutini omogočata prilagoditev sistema z monitorjem, pisanim za ASCII teleprinter na konfiguracijo z Baudotovim teleprinterjem kot vhodno-izhodno enoto.

Komentarji so pisani ob posameznih korakih programa, ki je priložen.


```

0000 B6 FC FC INCRB LDA A STATUS
0001 28 FB FB INCRB BHI INCRB
0002 84 E9 FF LDA A DATA
0003 84 E4 E4 CMP A SINUM
0004 27 09 SEC SWICH
0005 27 05 CMP A SIALF
0006 80 50 BEQ TSTA
0007 2D 0A BSR OUTCHB
0008 39 40 RTS
0009 77 40 SWICH STA A
0010 80 05 BSR OUTCHB
0011 20 05 BSR INCRB
0012 20 E5 BSR INCRB
0013 80 00 A2 OUTCHB TATB
0014 76 FC OUTCHB PSH A
0015 85 FB FC LDA A STATUS
0016 87 40 AND A STEADY
0017 25 F9 BNE
0018 7C F9 PUL A
0019 27 F9 STA A DATA
0020 29 F9 RTS
*****
* SEND CODE TABLE
*****
0021 F7 CE PCH
0022 F0 LF PCH
0023 F8 BLNK PCH
0024 F4 PCH
0025 F3 PCH
0026 F1 PCH
*****
* ASCII CODE TABLE FOR NUMERIC CHR.
*****
0027 3D $2F
0028 3F $3F
0029 3A $3A
0030 24 $24
0031 33 $33
0032 21 $21
0033 26 $26
0034 23 $23
0035 27 $27
0036 28 $28
0037 2D $2D
0038 29 $29
0039 2C $2C
0040 2E $2E
0041 20 $20
0042 24 $24
0043 27 $27
0044 28 $28
0045 2D $2D
0046 2A $2A
0047 2E $2E
*****
* ASCII CODE TABLE FOR NUMERIC CHR.
*****
0048 2D $2D
0049 3F $3F
004A 3A $3A
004B 24 $24
004C 33 $33
004D 21 $21
004E 26 $26
004F 23 $23
0050 27 $27
0051 28 $28
0052 2D $2D
0053 29 $29
0054 2C $2C
0055 2E $2E
0056 20 $20
0057 24 $24
0058 27 $27
0059 28 $28
005A 2D $2D
005B 2A $2A
005C 2E $2E
005D 20 $20
005E 24 $24
005F 27 $27
0060 28 $28
0061 2D $2D
0062 2A $2A
0063 2E $2E
0064 20 $20

```

```

*****
* INPUT CHR. FROM BAUDOT TTY
*****
* INTERFACE INCLUDES UART; TDS-ARDE- AND RDA-
* STORED ON $E001 SWE- ENABLED ON $E5FC
*****
* INITIALIZATION
TREADY EQU $40
NSW EQU $4D
YHI EQU $54
XHL0W EQU $55
IALF EQU $5D
INUM EQU $54
DATA EQU $E0F9
STATUS EQU $E0FC
FC INCRB LDA A STATUS
BHI INCRB
LDA A DATA
CMP A SINUM
SEC SWICH
CMP A SIALF
BEQ TSTA
BSR OUTCHB
RTS
SWICH STA A
BSR OUTCHB
BSR INCRB
BSR INCRB
TATB
PSH A
LDA A STATUS
AND A STEADY
BNE
PUL A
STA A DATA
RTS
*****
* SEND CODE TABLE
*****
CE PCH
LF PCH
BLNK PCH
PCH
PCH
*****
* ASCII CODE TABLE FOR NUMERIC CHR.
*****
$2F
$3F
$3A
$24
$33
$21
$26
$23
$27
$28
$2D
$29
$2C
$2E
$20
$24
$27
$28
$2D
$2A
$2E
*****
* ASCII CODE TABLE FOR NUMERIC CHR.
*****
$2D
$3F
$3A
$24
$33
$21
$26
$23
$27
$28
$2D
$29
$2C
$2E
$20
$24
$27
$28
$2D
$2A
$2E
*****
* ASCII CODE TABLE FOR NUMERIC CHR.
*****
$2D
$3F
$3A
$24
$33
$21
$26
$23
$27
$28
$2D
$29
$2C
$2E
$20
$24
$27
$28
$2D
$2A
$2E
*****
* ASCII CODE TABLE FOR NUMERIC CHR.
*****
$2D
$3F
$3A
$24
$33
$21
$26
$23
$27
$28
$2D
$29
$2C
$2E
$20
$24
$27
$28
$2D
$2A
$2E

```


kako narišemo pravilni mnogokotnik ?

v. batagelj

UDK 681.3:513

FNT, VTO matematika in mehanika Ljubljana

Namen sestavka je, ob reševanju navidez enostavne naloge: narisati pravilni mnogokotnik z vsemi diagonalami, seznaniti bralca z nekaj osnovnimi značilnostmi uporabe risalne naprave (plotterja). V sestavku se srečamo tudi s teorijo grafov (Eulerjev izrek), z uporabo rekurzije pri programiranju in s prevedbo rekurzivne rešitve v iterativno.

HOW TO DRAW A REGULAR POLYGON ? - In the paper some basic characteristics of plotter-programming are illustrated by solving the problem: write a program for drawing the regular polygon with all diagonals. The use of recursion and the transformation of the recursive solution to an iterative one are also included.

Naloga:

Sestavi program, ki bo na risalni napravi (plotterju) narisal pravilni mnogokotnik z vsemi diagonalami;

kateri je posvečen pričujoči sestavek, sodi v tisto področje programiranja, ki mu radi pravimo "igračkanje na računalniku"; nekateri pa celo: "kako lahko z računalnikom upravljamo čas (in denar) ?!".

Toda to je le ena plat medalje. Poglejmo še za hip na drugo stran. Programiranje je precej razgibana dejavnost in zahteva od programerjev nenehno učenje. Kajti, če hočemo nekaj uporabljati, moramo to tudi dobro poznati in obvladati. In prav tu se rado pojavi "igračkanje", v katerem se prepletajo:

- preizkušanje in iskanje (za programerja) novih programirnih tehnik in konceptov;
- zanimivost in "čistost" naloge;
- kratkost rešitve (programa) - od nekaj 10 do nekaj 100 vrstic;

... seveda, nove tehnike in koncepte lahko ponavadi veliko hitreje in temeljiteje preizkusimo s posebej v ta namen pripravljenimi testnimi programi ... ali, če povzamemo za [8]: "Čemu zabresti v morje težav, ki se pojavijo pri računanju števila π , recimo, na milijon deci-

malnih mest natančno? Nedvomno prvih sto decimal, znanih že stoletja, zadostuje za reševanje kateregakoli realnega problema, kakršne srečamo v fiziki ali astronomiji. Čemu torej tolikšni naporji za nepotrebno natančnost?" ...

In vendar so računalniki mleli podobne probleme ure in ure. Omenimo le še, da je naš rojak baron Jurij Vega že leta 1794 (brez računalnika) izračunal π na 140 mest natančno.

Večina programerjev, ko se sreča z interaktivnim delom, piše programe za razne igrice; ko pa se srečamo z rekurzijo, skoraj ne moremo mimo naloge o osmih kraljicah [5, 6, 7, 8, 9, 10].

Toliko v pojasnilo na morebitni ZAKAJ ?

S problemom uporabe risalne naprave sem se srečal, ko sem se odločil, da bom v CLUSE [1] vključil podprogram za risanje dendrogramov (dendrogram = grafični prikaz drevesa združevanja). Preden sem se spoprijel s tem problemom, sem se moral privaditi rutinam, ki omogočajo delo z risalno napravo. Slučajno sem tedaj v neki knjigi iz teorije grafov zagledal sliko polnega grafa (= pravilni mnogokotnik z vsemi diagonalami) in seveda sem se nemudoma vprašal: Ali bi ga znal narisati ?

Preden nadaljujemo, naj za bralca, ki ni domač

z risalno napravo, povem, da je osnovna akcija, ki jo omogočajo rutine za risanje, premik peresa po premici iz točke, v kateri se trenutno nahaja, v izbrano točko. Pri tem je pero ali spuščeno (= risanje) ali pa dvignjeno (= pomik brez risanja).

Za začetek našega ubadanja z nalogo poskusimo odgovoriti na vprašanje: Kako bi nalogo rešili sami z ravnilom, šestilom in kotomerom?

Recimo, da želimo narisati pravilni n -kotnik. Teda j narišemo najprej krožnico in jo z n točkami T_1, T_2, \dots, T_n (oglišča mnogokotnika) razdelimo na n enakih lokov. Nato povežemo vse točke eno z drugo z daljicami (diagonalami). To lahko povzamemo v prvo zamisel rešitve:

1. določi koordinate točk T_i , $i = 1, \dots, n$
2. za vsak $(i, j) : 1 \leq i, j \leq n$ ponovi
- 2.1. poveži točki T_i in T_j z daljico

Ta rešitev vsebuje dva spodrsaljaja:

- če je $i = j$, poskušamo z diagonalno povezati točko T_i samo s sabo;
- vsako diagonalno rišemo dvakrat,

ki pa ju zlahka odpravimo, če se odločimo, da mora biti indeks j vedno večji od indeksa i . Tako dobimo drugo zamisel:

1. določi koordinate točk T_i , $i = 1, \dots, n$
2. za vsak $(i, j) : 1 \leq i < j \leq n$ ponovi
- 2.1. poveži točki T_i in T_j z daljico

Vrstica 2 v tej rešitvi nam pušča proste roke v izbiri vrstnega reda prebora množice $\{(i, j) : 1 \leq i < j \leq n\}$. Ena izmed najpreprostejših določitev vrstnega reda je podana v naslednji zamisli rešitve:

1. določi koordinate točk T_i , $i = 1, \dots, n$
2. za $i = 1, \dots, n-1$ ponovi
za $j = i+1, \dots, n$ ponovi
- 2.1. prestavi dvignjeno pero v točko T_i
- 2.2. nariši daljico (od T_i) do točke T_j

Toda v tej rešitvi nismo niti poskusili upoštevati ene od osnovnih zahtev pri programiranju dela risalne naprave:

celotna pot, ki jo pri risanju opravi pero risalne naprave, naj bo kar se da kratka.

To je predvsem zahteva po (časovno) učinkoviti rabi risalne naprave.

Ali lahko pridemo do boljše rešitve?

Dolžina vsake poti peresa, katere rezultat je neka izbrana risba, je navzdol omejena z vsoto dolžin dejansko narisanih črt na tej risbi. Potemtakem lahko dobimo boljše rešitev le tako, da zmanjšamo skupno dolžino premikov dvignjene-ga peresa. Najugodnejše je seveda, če znamo ris-

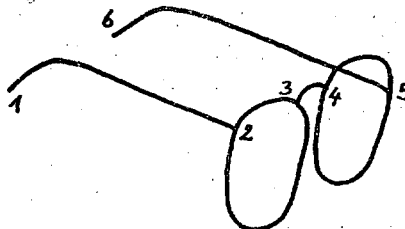
bo narisati v "eni potezi" - ne da bi dvignili pero.

Tu nam priskoči na pomoč Eulerjev izrek iz teorije grafov [4]. Za razumevanje tega izreka pojasnimo najprej dva pojma: Rekli bomo, da je risba povezana, če lahko iz vsake točke risbe pridemo po risbi v vsako drugo. Točka risbe je liha natanko takrat, ko je ali prosto krajišče črte, ki pripada risbi, ali pa se v njej sreča liho število črt; drugače je soda.

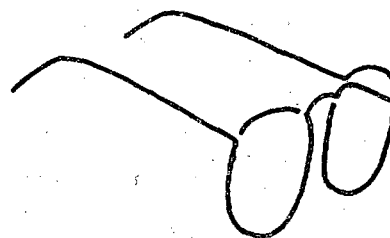
Sedaj že lahko prevedemo Eulerjev izrek v terminologijo našega problema:

Če je na povezani risbi $2k$ lihih točk (število lihih točk je vedno soda), potem lahko risbo narišemo v k potezah. Vsaka od teh potez se začne in konča v lihi točki. Povezano risbo brez lihih točk lahko narišemo v eni potezi s sklenjeno črto (risanje končamo v točki, v kateri smo ga začeli).

Za primer si oglejmo risbo:



Na njej je šest lihih točk, ki so oštevilčene s številkami od 1 do 6. Torej jo lahko narišemo v treh potezah. Na primer takole:



Povrnimo se na našo nalogo! Ali lahko kak (vsak?) n -kotnik z vsami diagonalami narišemo v eni potezi?

Ker so presečišča diagonal sode točke, nam lahko povzročajo težave le oglišča mnogokotnika. Vsako oglišče je povezano z diagonalami z vsemi ostalimi; to pa so tudi vse črte, ki se v njem stikajo. Zato bodo oglišča liha/soda, če bo n sodo/liho število. Torej lahko v eni potezi narišemo le lihe mnogokotnike (in 2-kotnik = daljica).

Toda, KAKO?

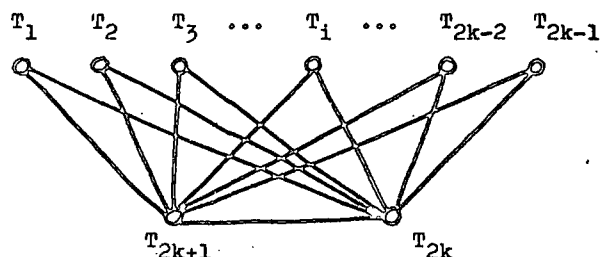
Recimo, da želimo narisati $2k+1$ - kotnik. Kakor vemo, ga je mogoče narisati s sklenjeno črto v eni potezi. To velja tudi za $2k-1$ - kotnik. Ali bi znali narisati $2k+1$ - kotnik, če že vemo, kako narišemo $2k-1$ kotnik? Če to znamo, potem znamo narisati poljuben lihi mnogokotnik; saj, ker znamo narisati trikotnik, bi znali narisati tudi petkotnik; ker znamo tega, bi znali tudi sedemkotnik; ...

Poglejmo, kam nas pripelje ta zamisel postopka risanja $2k+1$ - kotnika:

1. nariši $2k-1$ - kotnik
2. nariši ostanek: diagonale, ki imajo eno krajišče v točki T_{2k} ali točki T_{2k+1}

Kako narisati ostanek?

Tudi vse točke ostanka so sode. Torej ga lahko narišemo s sklenjeno črto v eni potezi.



Ponuja nam se nekaj "smiselnih" risanj ostanka; toda paziti moramo še, da bosta risanje $2k-1$ - kotnika in risanje ostanka vključena: točka, v kateri začnemo (in zaradi sklenjenosti tudi končamo) risati $2k-1$ - kotnik, mora biti začetek (in konec) risanja ostanka. Najbrž je najugodnejše, če izberemo za to točko kar točko T_1 . Teda lahko narišemo ostanek recimo takole (na začetku risanja se nahajamo v točki T_1):

2. "nariši ostanek"
 - 2.1. za $i = 2, 3, \dots, 2k$ ponovi
 - 2.1.1. če je i sod potem
 - 2.1.1.1. nariši daljico do T_{2k+1} drugače
 - 2.1.1.2. nariši daljico do T_{2k}
 - 2.1.2. nariši daljico do T_i
 - 2.2. nariši daljico do T_1

Če jezik, v katerega programiramo, pozna rekurzivne podprograme, smo s tem risanje lihega mnogokotnika že ugnali.

Recimo, da je dana procedura $crtado(i)$, ki nariše daljico iz točke, v kateri se pero trenutno nahaja, do oglišča T_i , potem bi risanje lihega mnogokotnika opisali v Pascalu z naslednjo rekurzivno proceduro:

```

procedure risilih( m: integer );
  { m - red mnogokotnika; liho število }
  var i: integer;
  begin
    if m > 1 then begin
      risilih( m-2 );
      for i := 2 to m-1 do begin
        crtado( m - i mod 2 ); crtado( i )
      end;
      crtado( 1 )
    end
  end { risilih } ;

```

Kaj pa, če jezik, v katerega programiramo, ne pozna rekurzije? Običajno se moramo tedaj zateči k uporabi sklada; vendar to v našem primeru ni potrebno, ker postopek vsebuje samo en rekurzivni klic. Take procedure ne zahtevajo "drevesnega mehanizma" in jih zato lahko pogosto brez posebnih težav prepisemo v iterativno obliko. Proceduro $risilih$ bi tako zapisali v Structranu [3]:

```

SUBROUTINE RISI LIH ( M )
  N = 3
  WHILE ( N .LE. M ) DO
    NM = N - 1
    FOR ( I = 2, NM ) DO
      CALL CRTA DO ( N - MOD(I,2) )
      CALL CRTA DO ( I )
    ENDFOR
    CALL CRTA DO ( 1 )
    NI = N + 2
  ENDWHILE
  RETURN
END

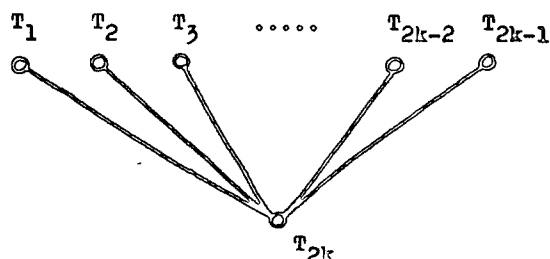
```

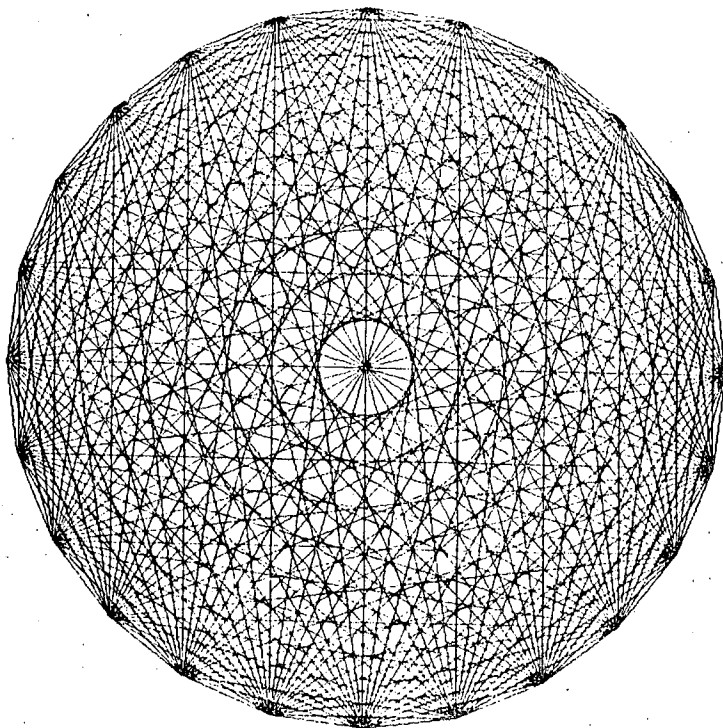
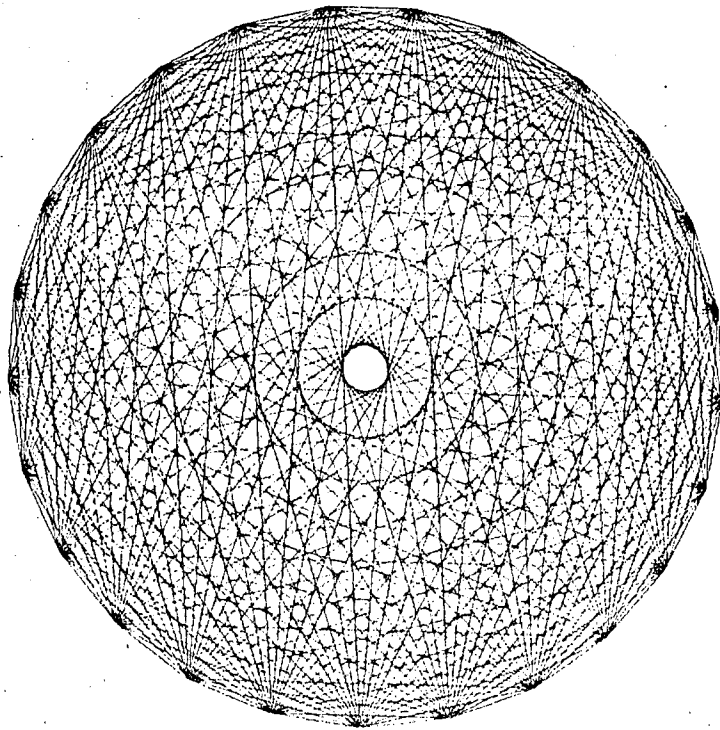
Tako, lihe mnogokotnike znamo narisati v eni potezi. Kako pa je s sodimi?

Recimo, da želimo narisati $2k$ - kotnik. Potem nam Eulerjev izrek pove, da bomo morali vsaj k krat pomakniti pero "po zraku" v drugo točko; pa tudi, da to zadostuje. Podobno kot prej, lahko problem razbijemo na dva podproblema:

- nariši $2k-1$ - kotnik
- in
- nariši ostanek: diagonale, ki imajo eno krajišče v točki T_{2k} .

Prvi podproblem je naš stari znanec - risanje lihega mnogokotnika. Torej moramo premisliti le še risanje ostanka. Ta sedaj izgleda takole:





Ker mora biti risanje $2k-1$ - kotnika in ostan-
ka vsklajeno, je najenostavneje začeti risanje
v točki T_1 . Tako dobimo naslednji postopek:

1. pomakni se v točko T_1 in nariši
 $2k-1$ - kotnik
2. za $i = 1, 2, \dots, k-1$ ponovi
 - 2.1. nariši daljico do T_{2k} ,
 - 2.2. nariši daljico do T_{2i} ,
 - 2.3. pomakni se v točko T_{2i+1}
3. nariši daljico do T_{2k}

S tem smo obdelali obe možnosti (lihe in sode
mnogokotnike). Preostane nam le še, da ju zdru-
žimo v enoten program za risanje poljubnega
mnogokotnika:

```
PROGRAM TETIVE( INPUT, TAPE14=INPUT )
COMMON / TOCKE / X(50), Y(50)
DATA PI / 3.14159 26536 /
CALLI DDPRI
```

◦ PREBEREMO RED M IN POLMER R MNOGOKOTNIKA

```
READ( 14, * ) M, R
```

◦ DOLOČIMO KOORDINATE OGLIŠE MNOGOKOTNIKA

```
PI = 2 * PI / M
FOR ( I = 1, M ) DO
  X(I) = R * COS(I*PI)
  Y(I) = R * SIN(I*PI)
ENDFOR
```

◦ NARIŠEMO MNOGOKOTNIK

```
CALLI POMIK V ( 1 )
IFI ( MOD(M;2) .EQ. 1 ) THEN
  CALL RISI LIH ( M )
ELSEI
  MM = M - 1
  CALL RISI LIH ( MM )
  FOR ( I = 2, MM, 2 ) DO
    CALL CRTA DO ( M )
    CALL CRTA DO ( I )
    CALL POMIK V ( I+1 )
  ENDFOR
  CALL CRTA DO ( M )
ENDIF
CALLI ZAPRI
END
```

V programu smo poleg podprograma risilih, ki
ga že poznamo, uporabili nekaj podprogramov, ki
omogočajo delo z risalno napravo. Ti so odvisni
od osnovnih rutin za risanje, ki se od računal-
nika do računalnika spreminjajo. Za računalnik
Cyber na RRC v Ljubljani izgledajo takole [1]:

```
SUBROUTINE CRTA DO ( I )
COMMON / TOCKE / X(50), Y(50)
CALLI PLINE ( X(I), Y(I), 1 )
RETURN
END
```

```
SUBROUTINE POMIK V ( I )
COMMON / TOCKE / X(50), Y(50)
CALLI PLINE ( X(I), Y(I), 0 )
RETURN
END
```

```
SUBROUTINE DDPRI
CALLI POPEV ( 11, 6LTETIVE )
CALLI PSCALE ( 11, 0, 9, 0, 11, 0, 11, 0,
              -1, 0, 1, 0, -1, 0, 1, 0 )
```

```
RETURN
END
```

```
SUBROUTINE ZAPRI
CALLI PCLOSE
RETURN
END
```

Opomba: Bralec, ki ima dostop do Cybera, lahko
zahteva izpis priročnikov za delo z risalnimi
rutinami in za Structran s Scope karticama:

BEGIN,MINI.
MANUAL,PLOTTER,STRUCTRAN.

Na koncu naj omenim soroden in nekoliko "upora-
bnejši" (vsaj za tiste, ki se ukvarjajo z grafi)
problem, pri reševanju katerega bi nam nabrane
izkušnje precej koristile. To je: sestavi pro-
gram, ki bo na podoben način, kot smo ga upora-
bili pri risanju mnogokotnikov, narisal dani
neusmerjeni graf, v katerem poljubni dve točki
veže največ ena povezava. Nekaj koristnih idej
v tej smoti je najti na zadnjih straneh sestav-
ka [2]. Problem si lahko dodatno zabelimo z
zahtevo, naj bodo točke grafa razvrščene po
krožnici tako, da se bodo povezave sekale čim
manjkrat. To pa je že zelo težek problem.

LITERATURA

- [1] V.Batagelj: CLUSE - program za hierarhično
razvrščanje v skupine; priročnik, Ljubljana,
1977
- [2] V.Batagelj, T.Pisanski: Delno usmerjeni Eu-
lerjevi grafi; (rokopis) seminar za računal-
niško in numerično matematiko, Ljubljana,
1977
- [3] V.Batagelj, E.Zakrajšek: Structran, Infor-
matica 75, Bled, 1975
- [4] D.Cvetković, M.Milić: Teorija grafova i nje-
ne primene; Naučna knjiga, Beograd, 1977
- [5] O.J.Dahl, E.W.Dijkstra, C.A.R.Hoare: Struct-
ured programming; Academic Press, London,
New York, 1972
- [6] F.Gruenberger, G.Jaffray: Problems for Com-
puter Solution; John Wiley & Sons, New York,
1965
- [7] R.B.Kieburzt: Structured programming and
problem solving with PL/1; Prentice-Hall,
Englewood Cliffs, New Jersey, 1977
- [8] J.Nievergelt, J.C.Farrar, E.M.Reingold: Com-
puter approaches to mathematical problems,
Prentice-Hall, Eng. Cliffs, N.Jersey, 1974
- [9] M.B.Wells: Elements of Combinatorial Compu-
ting; Pergamon Press, Oxford, 1971
- [10] N.Wirth: Algorithms+Data Structures=Programs;
Prentice-Hall, Eng.Cliffs, N.Jersey, 1976
- [11] E.Zakrajšek: Uporaba risalne naprave v RRC;
priročnik, Ljubljana, 1975

procesor za aritmetično - - logični izrazi

o. b. popov

UDK 681.3 - 181.4.055.06

Univerzitet "Kiril Metodij" Skopje

Se pogolemata primena na mikroprocesorite na site področja od naukata za smetački mašini uslovi razvivanje na soodvetna programska oprema. Prirodna na programskata oprema se zasniva na principot na izvrševanje na istat, odnosno dali taa će bide izveduvana na samiot smetač ili na nekoj proizvolen sistem koj od svoja strana nudi izvesni prenosti. Močna i kompleksna aritmetika mu ovozmožuju na sistemot da go proširi poletu na svoeto dejstvovanje, kako vo rešavanjeto na soodvetni matematičko-prirodni problemi, taka i za simulacija na tehnični procesi. Procesorot za aritmetičko-logični izrazi pretstavuva eden mal preveduvač, koj ovozmožuju preveduvanjeto na programite napišani vo programski jazik od povisok red da bide izveduvano na uslužen smetački sistem i so toa se zgolemat možnostite što gi podava mikroprocesorskiot sistem MC6800.

PROCESSOR FOR ARITHMETICAL AND LOGICAL EXPRESSIONS. The purpose of this article is to enlighten some ideas in the development of software for microprocessors. The top-down method of vertical fragmentation has been employed in the construction of the processor for arithmetical and logical expressions for the actual microprocessor MC6800. The structure of the processor resembles both functionally and structurally, that one of a mini-compiler. Powerfull and convenient arithmetics broadens the field for the application; it also enables us to develop the associate software before the actual microprocessor has been purchased. Finally, some efforts have been explained in the construction of processor that will employ the principle of machine independence.

UVOD

Pojavata na mikroprocesorot pretstavuva revolucija vo razvojt na naukata za smetački mašini. Razvlenata tehnologija, masovnoto proizvodstvo i potrošivačka, sigurnata i efitna izработка se pričini koji uslovlja da mikroprocesorot deneska naoga primena vo site področja na naukata za smetački mašini.

Vistinata da e materijalnata oprema na mikroprocesorot ednostavna i modularna vo odnos na osnovniot sistem, često pati e zamaglena so vistinata da e programiranje naporno i skapo. Množestvoto na mikroprocesori koe deneska e prisutno na tehnološkiot pazār, uslovi razvivanje na soodvetna programska oprema na tri nivoja:

- programiranje vo heksadecimalna koda
- programiranje vo sobiren jazik
- programiranje vo jazik od povisok red

Sekoj od ovie tri načina na programiranje nosi so sebe odredeni prednosti i nedostaci. Trgnuvajči od čovekot kako osnoven faktor za kogo treba da bide olesneta upotreba na mikroprocesorot, se odlučime za razvivanje na programska oprema koja ovozmožuju upotreba na jazik za programiranje od povisok red.

Vo odnos na izvrševanjeto na programskata oprema, istata ja delime na: sistemsko zavisna programska oprema i sistemsko nezavisna programska oprema. Sistemsko zavisnata

programska oprema sodrži programi koji se napišani vo množestvoto naredbi koji pripagaat na mikroprocesorot i se izvršuju na samiot mikroprocesor. Sistemsko nezavisnata programska oprema sodrži programi koji se izveduvaat na uslužen smetački sistem, ne bara poznavanje na jazikot na mikroprocesorot, so cel da se izraboti programska oprema za konkreten mikroprocesor. Razvivanjeto na sistemsko nezavisnata programska oprema ni ovozmožuju da gi koristime site prednosti što gi poddava eden golem smetački sistem. Pri toa vremenskata kompleksnost na programite vo obrabotka e relativno mala. Nedostatak e nesoodvetnata prostorna kompleksnost, zatoa što programite napišani vo jazik od povisok red generiraat dva pati poveće mašinska koda, otkolku programite koji se napišani vo sobiren jazik; znači upotreben e dvapati pogolem prostor vo memorijata na smetačot.

KONCEPTUALEN MODEL NA PROCESOROT

Procesorot za aritmetičko-logični izrazi po svojata konceptualna struktura potsetuva na mini-preveduvač za jazik što generira aritmetičko-logični izrazi. Procesorot e vertikalno fragmentiran; podelen e na poveće fazna struktura koja go izveduva procesot na preveduvanje posledovatelno.

Sostaven e od dva osnovni modula: modul na analiza i modul na sinteza. Implementacijata na procesorot će bide realizirana na proizvolen smetački sistem, koj vo napiot slučaj e imenuvan uslužben smetač. Izlez na procesorot e niza od naredbi vo sobiren jazik, koj od svoja strana pretstavuva izvoren jazik za mikroprocesorskiot sistem MC6800.

Procesorot treba da ovozmoži:

- koristnikot pri razmisluvanjeto i izrabotkata na programi i algoritmi da ostane na logično nivo; arhitekturata, osobinute i možnostite na objektniot smetač ne go zasegnuvaat.
- procesot na prepoznavanje i popravanje na greškite da bide poednostaven
- zgolemuvanje na univerzalnosta na mikroprocesorot so promena na jazikot i ednostavni modifikacii na odredeni subroutini
- poednostavna modularnost vo programiranjeto i dokumentacija

Modulot na analiza gi sodrži slednite funkcionalni edinici:

a. leksičen analizator: izgraduva vnatrešen oblik na pretstavuvanje na sekoj izvoren znak; go pregleduva izvornite zborovi i gradi simbolni tabeli vo kol se naoga informacijata koja se odnesuva na semantičното značenje na izvornite simboli i koja ja upotrebuva procesorot vo procesot na generiranje na koda.

b. sintaksen analizator: opredeluva dali jazičnata struktura na izvornata rečenica se sovpa so sintaksната definicija na jazikot; gradi sintaksno drvo za sekoja rečenica

c. translator na programot vo preoden jazik: nastanuva linearizacija na sintaksното drvo, pri što dobivame struktura pogodna za generacija na kodata.

Modulot na sinteza gi isodrži slednite funkcionalni edinici:

a. generator na kodata: gi presmetuva site adresi zavisni od sistemot (objektniot smetač), pridružuva registri na operandite i generira niza sobirni naredbi kol ovozmožuvaat izvršuvanje na soodvetnite aritmetično-logični operacii.

b. optimalizator na kodata: ja optimalizira sobirnat koda na smetačot; pri toa se namaluva prostornata i vremenskata kompleksnost na izvorniot program.

Pokraj spomenatite funkcionalni edinic porazdeleni vo dvata osnovni moduli na procesorot, postojat dve funkcionalni edinici kol procesorot gi gradi vo modulot na analiza i gi upotrebuva niz celiot proces na preveduvanje. Toa se: simbolnata tabela i edinicata za otkrivanje i popravanje na greškite.

Izvorniot jazik e element na množestvoto programski jazici od povisok red. Toj e tipski jazik; znači da atributite kol gi pridružuvaat operandite eksplicitno ili implicitno, vo zavisnot od samata implementacija na izvorniot jazik. Jazikot za aritmetično-logični izrazi (JAL vo ponatamošnata soдрžina) e generiran od kontekstno-slobodna gramatika. Prvobitniot oblik na ovaa gramatika soдрži leva rekurzija od čisto semantični pričini. Pri konstrukcijata na sintaksniot analizator kontekstno-slobodnata gramatika ja preveduvame vo regularna gramatika, (ovoj oblik i go podavame).

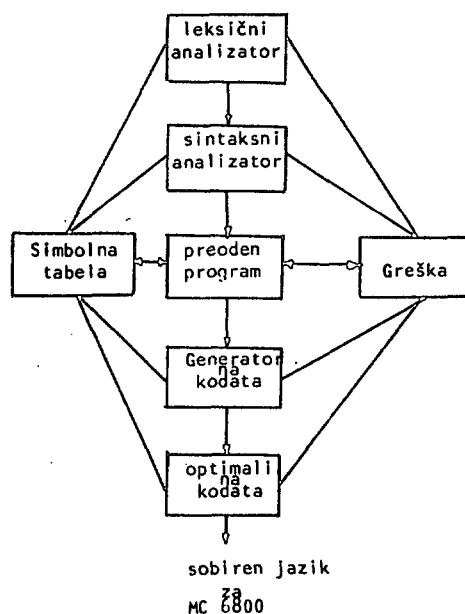
Strukturata na JAL e opredelena so slednite ograničuvanja

i pravila:

- site identifikatori se sostaveni samo od bukvi
- maksimalnata dolžina na sekoj identifikator e šest znaka
- site realni konstanti soдрžat decimalna točka
- promenlivata od levata strana na rečenicata na pridružuvanje mora da bide od ednakov tip kako i izrazot na desna strana
- operandi se promenlivi i konstanti
- operatorite se ednomestni, dvomestni i povecemestni
- rezultat na aritmetična operacija so celobrojni vrednosti e sekogaš celobrojna vrednost
- rezultat na aritmetična operacija so realni vrednosti e sekogaš realna vrednost
- vo slučaj da imame operandi od različen tip pri aritmetičnite operacii, rezultatot e realna vrednost
- vo logičnite izrazi operandite moraat da imaat logična (boolean) vrednost
- rezultat na logični operacii e sekogaš logična vrednost
- rezultat na relaciski operacii e sekogaš logična vrednost

Sintaksni edinici na JAL se: operandi, operatori i poseben znak, imenuvan "simbol na krajot na izrazot" (gi razdeluva izrazite megu sebe) i go označuvame so (" ; ").

Napomena: Programite kol gi pretstavuvaat soodvetnite funkcionalni edinici na procesorot za JAL ne se dadeni vo celost. Veruvame da od delovite kol se pretstaveni, osnovnite principii i metodi kol se upotrebeni se jasni.



LEKSIČEN ANALIZATOR

Vlez na leksičniot analizator e niza od znaci od izvorniot program, kol gi pregleduva i sozdava edinstvena logična pretstava za sekoj znak imenuvana "simbol". Leksičniot analizator e imenuvan REDAKTOR, zatoa što osnovna negova dejnost e redakcija na izvorniot tekst (program) i preureduvanje za ponatamošna obrabotka. REDAKTOR e račno napišan leksični

analizator, koj gi izvršuje slednite operaciji:

1. Čitanje na znacite na izvorniot program
 2. Prepoznavanje na soodvetnite sintaksni edinici
 3. Gradnje na tabeli na vrednosti (realni, celobrojni vrednosti, kako i logični konstanti)
 4. Gradnje na tabeli na identifikatori
 5. Odstranuvanje na tekstualni elementi, koji ne se potrebni vo ponatamošniot proces na preveduvanje (prazni mesta)
- Sekoja sintaksna edinica po redakcijata e pretstavena so deskriptor koj opišuje:

- prirodna na sintaksna edinica (broj, identifikator, operator)

- položbata na sintaksna edinica (adresata)

Programot REDAKTOR gi upotrebuva slednite rutini:

- PROČITAJGOSLEDNIOTZNAK : go pročituje sledniot znak vo programot napišan vo izvoren jazyk

- NAPRAVISIMBOL : izgradi soodvetna vnatrešna predstava za osnovnite simboli

- REČNIK : ima dva parametra: broj ili identifikator, kako i redniot broj na simbolot

- TABELA : vo nea gi smestuvame atributite i nivnite soodvetni adresi

- TEST : proveruje dali daden identifikator e rezerviran zbor

program REDAKTOR

```

dowhile ( sledniot znak = kraj na datotekata )
  case vidnasimbolot
    case1 ( vidnasimbolot = identifikator )
      call PROČITAJGOSLEDNIOTZNAK ;
      dowhile ( sledniot znak vo množestvoto A...Z )
        call PROČITAJGOSLEDNIOTZNAK ;
      enddo ;
    call NAPRAVISIMBOL ;
    call TEST ;
    call REČNIK ;
    call TABELA ;
  case2 ( vidnasimbolot = broj )
    call PROČITAJGOSLEDNIOTZNAK ;
    dowhile ( sledniot znak vo množestvoto 0...9 )
      call PROČITAJGOSLEDNIOTZNAK ;
    enddo ;
    call NAPRAVISIMBOL ;
    call REČNIK ;
    call TABELA ;
  case3 ( vidnasimbolot = praznomesto )
    call PROČITAJGOSLEDNIOTZNAK ;
    dowhile ( sledniot znak = " " )
      call PROČITAJGOSLEDNIOTZNAK ;
    enddo ;
  endcase ;
if ( sledniot znak = " : " )
  then call PROČITAJGOSLEDNIOTZNAK ;
  if ( sledniot znak = " = " )
    then call NAPRAVISIMBOL ;
    call TABELA ;
    call PROČITAJGOSLEDNIOTZNAK ;

```

```

else call GREŠKA
endif ;
else prodolži
endif ;
if ( sledniot znak = " + " )
  then call NAPRAVISIMBOL ;
  call TABELA ;
  call PROČITAJGOSLEDNIOTZNAK ;
else prodolži
endif ;

```

endprogram co kraj na programot REDAKTOR co

SINTAKSEN ANALIZATOR

Sledna faza vo procesot na preveduvanja na JAL e sintaksna analiza. Tvrdime da e rečenicata r od jazykot J(G) za proizvolna kontekstno-slobodna gramatika sintaksno analizirana ako poznavame barem edno nejzino sintaksno drvo. Sintaksno drvo e ekvivalentna grafična pretstava na rečenicata i struktura koja se obiduvame da ja izgradime pri sintaksna analiza.

Od množestvoto metodi koji postojat za realizacija na sintaksniot analizator se opredelime za rekurzivnata metoda. Ednostavnosta i popularnosta na ovaa metoda ne osloboduvaat od podetalno objasnuvanje na istata, koja e del od sekoj sodremen kurs za pišuvanje na preveduvači. Metodata postavuva dva uslova: gramatična (formalna) definicija na izvorniot jazyk i rekurziven programski jazyk vo koj će bide rekurzivnata metoda realizirana.

Samata gramatika ni služi kako blok-dijagram pri pišuvanje na sintaksniot analizator. Sekoj neterminalen znak na soodvetnata gramatika go pretvorame vo rutina. Analizata na neterminalniot znak taka stanuva povik na odredena rutina. Rekurzivnata metoda, kako element na množestvoto gramatiki LL (k) ima pogodna prostorna i vremenska kompleksnost: taa e od redot n, pri obrabotka na niza od simboli so dolžina n. Pri rekurzivnata metoda, skladot go gradime implicitno. Sintaksniot analizator gi upotrebuva slednite rutini:

- REDAKTOR : toa e vsušnost leksičniot analizator ; povikan e koga e procesiran pravi terminalen simbol ili pravi niza od terminalni simboli

- SIMBOL : sekogaš go soдрži sledniot simbol za procesiranje

- GREŠKA : ako terminalniot simbol ne se sovpaga so sintaksniot konstrukt na jazykot, se povikuje rutinata GREŠKA koja generira za vidot na sintaksna greška vo izvorniot program

program SINTAKSNAANALIZA

```

co ja procesirame rečenicata na pridružuvanje co
call PROMENLIVA ;
if ( call SIMBOL = " := " )
  then call REDAKTOR ;

```

```

    else call GREŠKA
endif ;
call IZRAZ ;
if ( call SIMBOL = " ; " )
    then call REDAKTOR
    else call GREŠKA
endif ;
endprogram co kraj na programot SINTAKSNA ANALIZA co
subroutine IZRAZ
    co karakterot na izrazot go poddava atributot na
    promenlivata na levata strana od izrazot co
    case karakternaizrazot
        case1 ( karakternaizrazot = aritmetični izraz )
            call ARITMETIČENIZRAZ ;
        case2 ( karakternaizrazot = logičen izraz )
            call LOGIČENIZRAZ ;
    endcase ;
endsubroutine
subroutine ARITMETIČENIZRAZ
    co proveruvame dali kako operator se javuva ednomesten
    minus co
    if ( call SIMBOL = " - " )
        then call REDAKTOR ;
        call ČLEN
    else call ČLEN
    endif ;
    co procesirame niza od operatori na sobiranje co
    dowhile ( call SIMBOL = "-" ili "+" )
        call REDAKTOR ;
        call ČLEN ;
    enddo ;
endsubroutine
subroutine ČLEN
    co proveruvame dali kako operator se pojavuva
    simbolot za stepenuvanje co
    if ( call SIMBOL = " " )
        then call REDAKTOR ;
        call FAKTOR
    else call FAKTOR
    endif ;
    co procesirame niza od operatori na množenje co
    dowhile ( call SIMBOL = "*" ili "/" )
        call REDAKTOR ;
        call FAKTOR ;
    enddo ;
endsubroutine
subroutine FAKTOR
    co proveruvame dali postojat podizrazi vo mali
    zagradi ( okrugli zagradi ) co
    if ( call SIMBOL = " ( " )
        then call REDAKTOR ;
        call ARITMETIČENIZRAZ ;
        if ( call SIMBOL = " ) " )
            then call REDAKTOR ;
            else call GREŠKA

```

```

    endif ;
    else if ( call SIMBOL vo množestvoto A...Z )
        then call PROMENLIVA
        else call KONSTANTA
    endif ;
endif ;
endsubroutine
subroutine LOGIČENIZRAZ
    call IMPLIKACIJA ;
    co procesirame niza od operatori na ekvivalencija co
    dowhile ( call SIMBOL = "EQUI" )
        call REDAKTOR ;
        call IMPLIKACIJA ;
    enddo ;
endsubroutine
subroutine IMPLIKACIJA
    call LOGIČENČLEN ;
    co procesirame niza od operatori na implikacija co
    dowhile ( call SIMBOL = "IMPL" )
        call REDAKTOR ;
        call LOGIČENČLEN ;
    enddo ;
endsubroutine
subroutine LOGIČENČLEN
    call LOGIČENFAKTOR ;
    co procesirame niza od operatori "ILI " co
    dowhile ( call SIMBOL = "OR" )
        call REDAKTOR ;
        call LOGIČENFAKTOR ;
    enddo ;
endsubroutine
subroutine LOGIČENFAKTOR
    call LOGIČENKONSTITUENT ;
    co procesirame niza od operatori na "I" co
    dowhile ( call SIMBOL = "AND" )
        call REDAKTOR ;
        call LOGIČENKONSTITUENT ;
    enddo ;
endsubroutine
subroutine LOGIČNIKONSTITUENT
    co proveruvame dali kako operator se pojavuva
    negacija "NE" co
    if ( call SIMBOL = "NOT" )
        then call REDAKTOR ;
        call LOGIČNAPRIMITIVA
    else call LOGIČNAPRIMITIVA
    endif ;
endsubroutine
subroutine LOGIČNAPRIMITIVA
    co proveruvame dali postoi logičenpodizraz vo
    mali zagradi co
    if ( call SIMBOL = " ( " )
        then call REDAKTOR ;
        call LOGIČENIZRAZ ;
        if ( call SIMBOL = " ) " )

```

```

    then call REDAKTOR ;
    else call GREŠKA
endif ;
endif ;
else if ( call SIMBOL vo množestvo A ... Z )
    then call LOGIČNAVREDNOST
    else if ( call SIMBOL = identifikator )
        then call PROMENLIVA
        else call ODNOS
    endif ;
endif ;
endif ;
endsubroutine

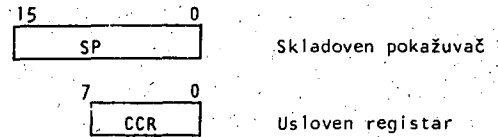
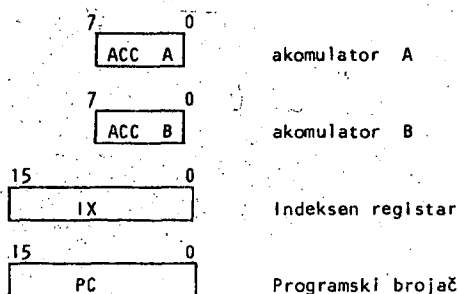
```

Pravilnoto delovanje na sekoj proces e zavisno od vlezot. Vleznoto struktura može da ne se sovpaga so strukturata koja go izvršuva procesot ili da ne postojat točni specifikaciji na vleznite podatoci, zboruvame respektivno za logična i formalna greška. Pri realizacijata na procesorot za aritmetično-logični izrazi treba da se posveti određen napor i vnanie za implementacija na rutinata GREŠKA. Realizacijata na rutinata treba da se zasniva na slednite pravila:

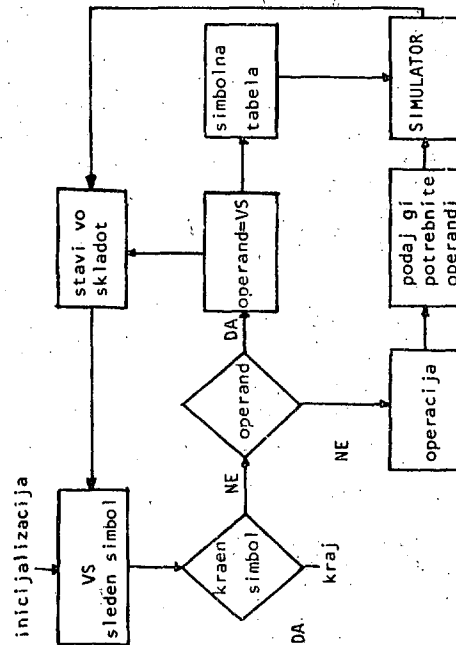
- da e nasočena kon koristnikot ; da ja otkriva greškata na koristnikot-programer, a ne na procesorot
 - da ja lokalizira greškata i da bide što podetalen opisot na prirodata na greškata (može da soodrži i poseben simbol koj ke ukažuva na mestoto na greškata)
 - da porakata za greškata se zasniva na izvorniot tekst
- Za pretstavuvanje na izvorniot program vo preoden jazik, ja izbravme postfiksna polska notacija. Algoritmot na translacija vodi smetka za odnapred defoniranata hierarhiška prednost na aritmetičnite i logičnite operatori. Kodata koja ja generirame od programot vo preoden jazik vo postfiksna, polska notacija e relativno optimalna. Algoritmot na translacija e podaden podetalno vo (15).

GENERATOR NA KODATA

Generiranje na kodata pretstavuva preslikuvanje na programot vo preoden jazik vo nekoja niza od simboli. Preslikuvanjeto go razgleduvame kako operator definiran nad sintaksnoto drvo i informacijata vo simbolnata tabela. Pri generacija na kodata se obiduvame da so pomoš na množestvo primitivni operatori gi razbleme složenite konstrukti na izvorniot jazik. Definirame okolina na objektniot sistem, koja ja tvorat organizacijata na registrite i strukturata na adresiranje. Za konkretniot primer okolina na sistemot e programskata struktura na mikroprocesorot IC 6800:



Generatorot na kodata e sostaven od dve funkcionalni ednici: skladoven avtomat i simulator. Skladovniot avtomat gi dostavuva soodvetnite operandi i operatori na simulatorot. Simulatorot generira soodvetna koda i ja menuva okolinata na sistemot. Za da ja izvrši ta zadata simulatorot soodrži opis na site vrednosti koji se naogaat vo procesot na obrabotka, toa e slika na vrednosta, kako i opis na okolina na objektniot sistem, imenuvan slika na sistemot. Vrednosta, proizvolna, e pretstavena vo sistemot na poveče načini, zadata na slika na vrednosta da ja poddava momentalnata pretstava na vrednosta. Registrite na objektniot sistem soodržat poveče vrednosti za vreme na izvršuvanje na programot zadata na slikata na sistemot da ja dava momentalnata soodržina na sekoj programski registar. Vrednosta e pod kontrola na generatorot na koda, koga skladovniot avtomat bara simulacija na simbol koja e operand ; momentalnata sostojba na avtomatot e argument na simulacijata. Generatorot ja prekinuva vrskata pomegu slikata na vrednosta i slikata na sistemot, koga vrednosta e upotrevena kako operand.



Generatorot na koda gi upotrebuva slednite rutini:

- EDNOMESTNAOPERACIJA : ima dva parametra : znak na operacijata i operandot koja se naoga na vrvot od skladot. Po izvršenata generacija na soodvetnata koda, rezultatot se vraća nazad vo skladot.
- DVOMESTNAOPERACIJA : ima tri parametri: znakot na operacijata i dvata operandi. Pri toa prviot operand veče se naoga vo akumulorot na objektniot sistem, vtoriot operand se naoga na poslednata vremenska memoriska lokacija. Rezultatot se vraća nazad na poslednata lokacija od skladot.

- GENERIRAJ : go opredeluje opisot na registrite i slikata na sistemot. ma tri parametri: registar koj se odnesuva na site programski registri na MC6800, operacija koja e pretstavena so soodveten mmonik od sobirniot jazik na sistemot, i rezultat koe pretstavuva lokacija na na koja se sočuvuva rezultatot na operacijata.

- CONVERSION : pretvara celobrojna vrednost vo realna vrednost.

- REOPERACIJA : rutina za generacija na soodvetnite naredbi za izvršuvanje na reacijski operaciji.

- LOGOPERACIJA : rutina za generacija na soodvetnite naredbi za izvršuvanje na logičnite operaciji.

Sekoja od rutinite za generacija na naredbi za izvršuvanje na aritmetični operaciji sodrži soodvetni naredbi za testiranje na prekoračuvanja pri aritmetičnite operaciji. Celobrojnite vrednosti zavzemaat eden zbor od memorijata, realnite vrednosti dva zbora. Sekoja realna vrednost e pretstavena so mantisa i eksponent.

zemi go sledniot simbol za obrabotka vo izrazot :

enddo ;

endprogram

subroutine EDNOMESTNAOPERACIJA

if (znak na operacijata = " - ")

then call CHSIGN ;

call GENERIRAJ ;

skladnaoperandite ← rezultat ;

else call COMPLEMENT ;

call GENERIRAJ ;

skladnaoperandite ← rezultat ;

endif ;

endsubroutine

subroutine DVOMESTNAOPERACIJA

case operator

case1 (operator = " + ")

if (vrednost na operandite = realna)

then call FPADD ;

call GENERIRAJ ;

skladnaoperandite ← rezultat ;

else prodolži

endif ;

if (vrednost na operandite = celobrojna)

then call FIXPADD ;

call GENERIRAJ ;

skladnaoperandite ← rezultat

else prodolži

endif ;

if (vrednost na operandite = različna)

then call CONVERSION ; call FPADD ;

call GENERIRAJ ;

skladnaoperandite ← rezultat

else call GREŠKA

endif ;

case2 (operator = " - ")

⋮

endsubroutine

program GENERACIJANAKODATA

inicijalizacija ;

co go inicijalizirame skladot na operandi, koj e vsušnost kontrolen sklad na skladovniot avtomat; go inicijalizirame poletoto vo koe se naoga aritmetično-logičniot izraz vo postfiksen polski oblik, go inicijalizirame poletoto na vremenski lokaciji co

dowhile (IZRAZ = " ; ")

if (A registar = zafaten)

then call GENERIRAJ ;

skladnaoperandite ← temaddress

else prodolži

endif ;

if (B registar = zafaten)

then call GENERIRAJ ;

skladnaoperandite ← temaddress

else prodolži

endif ;

if (IZRAZ = operand)

then call POBARAJVOTABELATA ;

call POBARAJVOREČNIKOT ;

case vidnaoperandot

case1 (vidnaoperandot = konstanta)

call GENERIRAJ ;

skladnaoperandite ← IZRAZ ;

case2 (vidnaoperandot = promenliva)

call GENERIRAJ ;

skladnaoperandite ← IZRAZ ;

endcase ;

else prodolži

endif ;

if (IZRAZ = znak na operacija)

then if (znak na operacija = ednomestna)

then call EDNOMESTNAOPERACIJA

else call DVOMESTNAOPERACIJA

endif ;

else call GREŠKA

endif ;

SISTEMSKO NEZAVISEN PROCESOR

Rešenieto na sekoj problem teži kon univerzalnost. Pri konstrukcijata na procesorot za aritmetično-logični izrazi se obidovme da se do nivoto na sinteza procesorot bide nezavisean od objektnio sistem. Procesorot e sistemsko nezavisean na nivo na generacija na kodata, koga ne vodi smetka za opisot na sistemot i generira abstrakten preoden jazik, koj podocna lesno može da se pretransformira vo sobiren jazik za konkreten mikroprocesorski sistem.

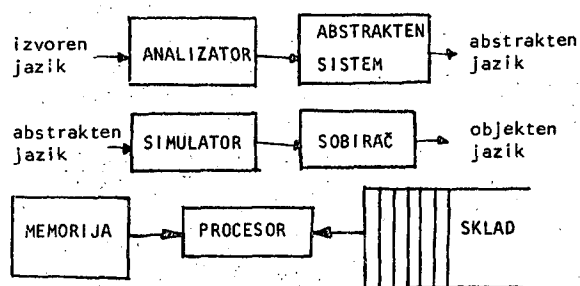
Konceptot na abstrakten sistem, koj što go voveduvame, kako model za proučuvanje i razvivanje na sistemsko nezavisna programska oprema, pretstavuva eden vid na preod po megu čovekot i realniot sistem. Abstraktni sistem go pretstavuvame so ednostavna arhitektura i malo množestvo na primitivni naredbi.

Primitivni naredbi se:

- LOAD i STORE : formatite na ovie naredbi se edno-adresni. Koga procesorot emitira naredba LOAD, operandot e staven na vrhov od skladot, ostanata soдрžina e potisnata nadolu. Pri naredbata STORE operandot e odstranet od vrhov na skladot, ostanata soдрžina e potisnata nagore.

- OPERACII : operatorite nemaat adresa (eden vid na na sistem so nulta adresa): Ednomestnite operatori dejstvuvaat na vrhov od skladot na samo eden operand, dvomestnite na gornite dva operanda.

Simulatorot pretstavuva množestvo na rutini kol se napišani so najgolema možna opštost za odredeni semejstva mikroprocesori kol se razdeleni vo odredeni ekvivalentni klasi. Na toj način simulatorot se pišuva za soodvetniot pretstavnik na ekvivalentnite klasi.



ZAKLJUČOK

Procesori-preveduvači deneska e možno da se implementiraat vo skoro site programski jazici; postojat i posebno napišani jazici za implementacija na preveduvači. Izborot na programskiot jazik se zasniva na minimalizacijata na naporite pri realizacijata kako i najgolem možen kvalitet vo upotreblivost i univerzalnost na preveduvačot.

Jazičnite strukturi se napolno formalizirani vo vremeto po Chomsky, so što e olesneta realizacijata na odredeni jazici. Leksikalnite i sintaksnite analizatori se napolno definirani i verojatno najizučeni oblasti od naukata za konstrukcija na preveduvači.

Generacijata i optimalizacijata na kodata se pred se zavisni od objektniot sistem. Optimalizacijata na jazikot JAL, treba da bide nasočena kon optimalizacija na zaedničkite aritmetični i logični izrazi. Edno od možnite rešehija, sekako treba so vnimanie da se razgleda i izuči, e i voveduvanje na redundantni zagradi za sekoja n-torka zaednički podizrazi.

* * *

Mal rečnik na ključni zborovi i frazi (anglisko-makedonski)

hardware = materijalna oprema
 software = programska oprema
 compiler = preveduvač
 assembler = sobiren jazik
 source language = izvoren jazik

resident software = sistemsko zavisna programska oprema

non-resident software = sistemsko nezavisna programska oprema

high-level programming language = programski jazik od povisok red

LITERATURA

- (1) A.P. ZELEZNIKAR
Prevaljniki, Univerza v Ljubljani 1977
- (2) A.P. ZELEZNIKAR
Pseudokodiranje: sintaksa in uporaba
Elektrotehniški vestnik, Letnik 43, št. 5, 1976
- (3) D. GRIES
Compiler Construction for Digital Computers
John Wiley & Sons, Inc., 1971
- (4) F.L. BAUER & J. EICKEL
Compiler Construction: An Advanced Course
Springer-Verlag, 1976
- (5) J.S. ROHL
An Introduction to Compiler Writing
Macdonald & Jane's, London, 1975
- (6) F. GENUYS
Programming Languages
Academic Press London & New York, 1968
- (7) A.V. AHO & J.D. ULLMAN
The Theory of Parsing, Translation & Compiling
Volume I : Parsing ; Volume II : Compiling
Prentice Hall 1972
- (8) H.S. STONE
Introduction to Computer Architecture
Science Research Associates, Inc. 1975
- (9) F.R. HOPGOOD
Compiling Techniques
Macdonald London, 1974.
- (10) D. QUEYSAC
Understanding Microprocessors
Motorola Inc., Semiconductors Products Division
- (11) W. MARTIN
Microcomputer in der Prozesdatenverarbeitung
Carl Hansen Verlag, Munchen, 1977
- (12) It's getting easier to program uPs with the new software design aids
Electronic Design, Volume 25, No: 2, January 18, 1977
- (13) Solve software problems step by step
Electronics Design, Volume 25, No: 2, January 18, 1977
- (14) R.T. YEH
Applied Computation Theory: Analysis, Design & Modeling
Prentice Hall, Inc., 1976
- (15) O.B. POPOV, Diplomaska naloga, FE-Univerza v Ljubljani

on n-th root evaluation by iterative methods with parameters

I. N. Djordjević

UDK 681.3:51

Faculty of Electronic Engineering
University of Niš, Yugoslavia

Introducing parameters new iterative formulae for n-th root evaluation are obtained. Choosing parameter values an iterative higher-order convergence formula is obtained, as a linear combination of iterative lower-order formulae. Thus one can achieve parallelization of an iterative procedure.

O IZRAČUNAVANJU N-TOG KORENA ITERATIVNIM METODAMA SA PARAMETRIMA. Uvodjenje parametara dobijene su nove iterativne formule za izračunavanje n-tog korena. Izborom vrednosti parametara dobija se iterativna formula višeg reda konvergencije, kao linearna kombinacija iterativnih formula nižih redova. Tim se postiže paralelizacija iterativnog procesa.

1. INTRODUCTION

Ever increasing role of digital computers in various practical fields requires an increased speed in square, cube and n-th root evaluation. That requires the generation of the new iterative algorithms of a high-order convergence.

To evaluate a simple real root of equation $f(x)=0$, the use in [1] has been taken of formulae

$$(1) \quad x_{i+1} = \phi(x_i) \quad (i=0, 1, 2, \dots),$$

where $\phi(x) = D_2(x)/\Delta_2(x)$, and $D_2(x)$ and $\Delta_2(x)$ are the following determinantes

$$D_2 = \begin{vmatrix} f(x) & f'(x) & \dots & f^{(n-1)}(x) \\ (f(x))' & (f(x))'' & \dots & (f(x))^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ (f(x))^{(n-1)} & (f(x))^{(n)} & \dots & (f(x))^{(n)} \end{vmatrix} \quad \Delta_2 = \begin{vmatrix} (f(x))' & (f(x))'' & \dots & (f(x))^{(n)} \\ (f(x))'' & (f(x))''' & \dots & (f(x))^{(n+1)} \\ \vdots & \vdots & \ddots & \vdots \\ (f(x))^{(n-1)} & (f(x))^{(n)} & \dots & (f(x))^{(n+1)} \end{vmatrix}$$

It was also shown in [1] that (1) has the convergence order $\mathcal{L}+2$. In this paper, using (1), iterative methods with parameters for the real, positive root evaluation of the following equation will be generated:

$$(2) \quad x^n - a = 0, \quad a > 0$$

where n is a natural number.

2. METHODS WITH A SINGLE PARAMETER

Formula (1) for $\mathcal{L}=0$, $f(x) = x^n$ and $f(x) = x^n - a$ obtains the form

$$(3) \quad x_{i+1} = x_i \left[1 - \frac{x_i^n - a}{n x_i^n + p(x_i^n - a)} \right]$$

If parameter p in (3) takes in turn the values 0 , $-n+1$, $-n$, one obtains the well-known formulae

$$(4) \quad x_{i+1} = \frac{1}{n} \left[(n-1)x_i + \frac{a}{x_i^{n-1}} \right]$$

$$(5) \quad x_{i+1} = x_i \frac{n a}{x_i^n + (n-1)a}$$

$$(6) \quad x_{i+1} = \frac{1}{na} x_i \left[(n+1)a - x_i^n \right]$$

Let x_i be a found approximate value $\sqrt[n]{a}$ by means of (3), and δ_i a relative error: $\delta_i = (x_i - \sqrt[n]{a}) / \sqrt[n]{a}$. Then the relative error δ_{i+1} expressed through δ_i , can satisfy the relation

$$(7) \quad \delta_{i+1} = (1 + \delta_i) \frac{(n-1)(1 + \delta_i)^n + p [(1 + \delta_i)^n - 1] + 1}{n(1 + \delta_i)^n + p [(1 + \delta_i)^n - 1]}$$

On applying a binomial formula, (7) becomes

$$(8) \quad \delta_{i+1} = \delta_i^2 \frac{\sum_{l=0}^{n-1} \binom{n-1}{l} [(1+l)n-1 + (l+2)p] \delta_i^l}{1 + \sum_{l=0}^{n-1} \binom{n-1}{l} \frac{n+p}{l+1} \delta_i^{l+1}}$$

Developing the right hand side of (8) in Mac Laurin's series with δ_i , and disregarding exponents higher than $m+2$ one obtains

$$(9) \quad \delta_{i+1} = \delta_i^2 [b_1(p) + b_2(p) \delta_i + b_3(p) \delta_i^2 + \dots + b_m(p) \delta_i^{m-1} + b_{m+1}(p) \delta_i^m]$$

where $b_i(p)$ ($i=1, 2, 3, \dots, m, m+1$) are given by

$$b_1(p) = p + \frac{n-1}{2}$$

$$(10) \quad b_2(p) = -p^2 - \frac{(n-1)(n+1)}{\delta}$$

$$b_3(p) = p^3 + \frac{3n+1}{2} p^2 + \frac{(7n-1)(n+1)p}{12} + \frac{(n-1)(n+1)(n+2)}{24} \text{ etc.}$$

Let us suppose that in (3) parameter p takes m mutually different values $p_i (i=1, 2, \dots, m)$, and that $x_{i+1,1}$ are correspondent approximate values $\sqrt[n]{a}$ estimated by means of (3) for the given value of x_i . Let S_{i+1} be an arithmetical mean value of those m approximate values $x_{i+1,1}$. Then it holds

$$(11) \quad \delta(a_{i+1}) = \frac{\delta^2}{m} \left[\sum_{j=1}^m b_1(p_j) + \delta_1 \sum_{j=1}^m b_2(p_j) + \dots + \delta_1^{m-1} \sum_{j=1}^m b_m(p_j) + \delta_1^m \sum_{j=1}^m b_{m+1}(p_j) \right].$$

Necessary and sufficient condition that S_{i+1} has convergence order $m+2$ is:

$$(12) \quad \begin{aligned} & \sum_{j=1}^m b_1(p_j) = 0, \\ & \sum_{j=1}^m b_2(p_j) = 0, \\ & \vdots \\ & \sum_{j=1}^m b_m(p_j) = 0, \\ & \sum_{j=1}^m b_{m+1}(p_j) \neq 0. \end{aligned}$$

Condition (12) determines the system m of algebraic equations by $p_i (i=1, 2, \dots, m)$ where the first equation is linear, the second quadratic and m -th equation is of degree m . Eliminating the unknown $m-1$ one can obtain an equation with p .

$$(13) \quad p^m + \frac{m(n-1)}{2} p^{m-1} + \frac{m(n-1)}{24} [(3m-4)n-3m+2] p^{m-2} + \frac{m(n-1)}{144} [3(m-2)^2 n^2 - 2(3m^2 - 9m+5)n + 3m^2 - 6m+2] p^{m-3} + \dots = 0.$$

The roots of equation (13) are the values of parameter p_i for which

$$(14) \quad \delta(a_{i+1}) = \frac{C(n)}{m} \delta_1^{m+2},$$

where $C(n)$ is a sum $\sum_{j=1}^m b_{m+1}(p_j)$ for the found values of p_i .

Further, the cases $m=1$ and $m=2$ will be studied. For $m=1$

$$p + \frac{n-1}{2} = 0, \quad \delta(a_{i+1}) = \frac{n-1}{6} \delta_1^3.$$

For $m=2$ equation (13) becomes

$$(15) \quad 6p^2 + 6(n-1)p + (n-1)(n-2) = 0,$$

the roots of which are

$$p_{1,2} = -\frac{n-1}{2} \pm \frac{\sqrt{3(n^2-1)}}{6}.$$

Substituting the values p_1 and p_2 in (14) one obtains

$$(16) \quad \delta(a_{i+1}) = \frac{n-1}{24} \delta_1^4.$$

Depending on that which value of parameter p is taken, (3) gives two values for the following approximate root value

$$(17) \quad x_{i+1,1} = x_i - \frac{x_i(x_i^n - a)}{nx_i^n + p_1(x_i^n - a)}, \quad x_{i+1,2} = x_i - \frac{x_i(x_i^n - a)}{nx_i^n + p_2(x_i^n - a)}$$

where $x_{i+1,1}$ is the value of $\sqrt[n]{a}$ above, and $x_{i+1,2}$ is the value of $\sqrt[n]{a}$ below the exact value. Then, for the next approximate value is taken

$$(18) \quad x_{i+1} = (x_{i+1,1} + x_{i+1,2})/2.$$

If in (18) are directly substituted $x_{i+1,1}$ and $x_{i+1,2}$ from (17), and the use is taken of connections $p_1+p_2 = 1-n$, $p_1 p_2 = (n-1)(n-2)/6$ it is obtained

$$(19) \quad x_{i+1} = x_i \frac{(n^2-1)x_i^{2n} + 2(2n^2+1)ax_i^n + (n^2-1)a^2}{(n+1)(n+2)x_i^{2n} + 4(n^2-1)ax_i^n + (n-2)(n-1)a^2}.$$

Note that in [2], for the next approximate value of $\sqrt[n]{a}$, the arithmetic mean of (4) and (5) ($p_1 = 0$, $p_2 = -n+1$), is taken. The convergence order of so obtained method is three (3) for $n \geq 3$, that is, for one order lower than the convergence order of formula (18).

3. METHODS WITH TWO PARAMETERS

In this case $z=1$, $\phi_0(x) = x^p$, $\phi_1(x) = x^q$, $f(x) = x^n - a$, and formula (1) after evident simplifications and substitutions $p+q=u$, $pq=v$ acquires the form

$$(20) \quad x_{i+1} = x_i \frac{[(n-1)u+v+(n-1)^2] x_i^{2n} + [(2-n)u-2v+n^2+2n-2] ax_i^n + (v-u+1)a^2}{(nu+vn^2) x_i^{2n} + (-nu-2v+n^2) ax_i^n + v \cdot a^2}$$

Relative error δ_{i+1} expressed through δ_i satisfies the relation

$$(21) \quad \delta_{i+1} = \frac{A(1+\delta_i)^{2n+1} + B(1+\delta_i)^{n+1} + C(1+\delta_i)}{D(1+\delta_i)^{2n} + E(1+\delta_i)^n + v}$$

where $A = (n-1)u+v+(n-1)^2$, $B = (2-n)u-2v+n^2+2n-2$,
 $C = -u+v+1$, $D = nu+vn^2$, $E = -nu-2v+n^2$.

Formula (20) has the convergence order 3, and there are two independent parameters u and v in it. By connection $v = -(n-1)(3u+n-2)/6$ convergence order of (20) is increased to 4, and only one parameter stays in (20). So the problem is reduced to the previous case where instead of p there exists u .

Now:

$$(22) \quad \delta_{i+1} = \frac{n-1}{24} \delta_i^4 (b_1(u) + b_2(u)\delta_i + b_3(u)\delta_i^2 + b_4(u)\delta_i^3 + b_5(u)\delta_i^4 + \dots)$$

where $b_i(u) (i=1, 2, 3, 4, 5, \dots)$ are given by

$$b_1(u) = u+n-1$$

$$b_2(u) = - [15u^2 + 30(n+1)u + 11n^2 + 30n - 44] / 30$$

$$b_3(u) = [3u^3 + (9n+6)u^2 + (7n^2 + 12n+14)u + n^3 + 4n^2 + 14n - 22] / 12$$

$$(23) \quad b_4(u) = - [63u^4 + (252n+126)u^3 + (336n^2 + 378n+294)u^2 + (168n^3 + 210n^2 + 588n+672)u + 29n^4 - 42n^3 + 218n^2 + 672n - 1084] / 504$$

Here, too, the cases $m=1$ and $m=2$ will be considered. For $m=1$, $u+n-1=0$ and (20) is reduced to

$$(24) \quad x_{i+1} = x_i \left[1 - \frac{6n(x_i^{2n} - a^2)}{(n+1)(2n+1)x_i^{2n} + 2(2n-1)(2n+1)ax_i^n + (n-1)(2n-1)a^2} \right]$$

The relative error (24) is

$$(25) \quad \delta(a_{i+1}) = \frac{(n^2-1)(4n^2-1)}{720} \delta_i^5$$

Formula (24) has the convergence order 5 and simpler structure than (19) whose convergence order is 4.

For $n = 2$, the values of parameters u_1 and u_2 satisfy the equation

$$(26) \quad 15u^2 + 30(n-1)u + (n-2)(11n-8) = 0.$$

Substituting u_1 and u_2 in $[b_1(u_1) + b_1(u_2)] / 2$, ($i=3,4,5$), for the relative error of the arithmetical mean a_{i+1} obtained are the terms

$$(27) \quad \delta(a_{i+1}) = \begin{cases} \frac{\delta_1^8}{128}, & n=2 \\ -\frac{(n^2-1)(n^2-4)(4n^2-1)}{37800} \delta_1^7, & n \geq 3. \end{cases}$$

Considering (27) one may conclude that the combined method has the convergence order 8 for $n=2$ and 7 for $n \geq 3$ using u_1 and u_2 from (26), and not 6 as may be expected by analogy with results from (2).

For $u = 2-n$, $v = (n-1)(n-2)/3$ (19) is obtained from (20), and for $u = -3n$ and $v = 2n^2$ is obtained

$$(28) \quad x_{i+1} = x_i \left\{ 1 + \frac{(x_i^n - a) [(n+1)x_i^n - (3n+1)a]}{2n^2 a^2} \right\}$$

the convergence order of which is two and in which only one division is necessary at the beginning of the iterative process. Introducing three parameters, an iterative formula may be obtained whose convergence order is 7 or more.

Numerical check of convergence speed of the described methods has confirmed the correctness of the exposed studies. The check was done for an interval $[\frac{1}{16}, 1]$ and for $n = 2 \div 10$.

4. CONCLUSION

The mentioned methods can be used in evaluation of $\sqrt[n]{x}$ by means of standard programme package for realization on microprocessors in float point arithmetics.

REFERENCES

- /1/ Varjuhin V.A., Kasjanjuk S.A., Ob iteracionnyh metodah utocnenija kornej uravnenij, Ž. vychisl. matem. i matem. fiz., 1969, 9, No 3, 684-687.
- /2/ Priladijka B.K., O vychislenii kornej s naturalnym pokazatelem, Matematika v škole, 1972, No5, 11-16.

bus standardi

m. kovačević

UDK 681.3 (083.7)

Institut "Jožef Stefan" Ljubljana

Standardizacija na svim područjima tehnike je proces koji je star skoro toliko koliko i tehnika sama, a razvija se i širi, osvajajući nova područja, zajedno sa razvojem tehnike. Tako i na području računarske materialne opreme (hardware) uočavamo sve veća i brojnija nastojanja na području standardizacije elemenata pa čak i modula. Jedan od značajnijih posreda na području standardizacije materialne računarske opreme predstavljaju pokušaji usvajanja jedinstvenog vodila (bus). U ovom broju našeg časopisa vam predstavljamo najpopularnije vodilo na području mikroracunarskih sistema S-100.

pin	symbol	ime	funkcija
1	+ 8 V	+ 8 Volts	Nestabiliziran napon na basu koji po stabilizaciji na ploči štampanog kola dobija vrijednost + 5 V.
2	+ 18 V	+ 18 Volts	Predstabiliziran pozitivni napon.
3	XRDY	EXTERNAL READY	Vanjski status ulaz u CPU modul
4	V/0	Vectored Interrupt Line 0	
5	V/1	Vectored Interrupt Line 1	
6	V/2	Vectored Interrupt Line 2	
7	V/3	Vectored Interrupt Line 3	
8	V/4	Vectored Interrupt Line 4	
9	V/5	Vectored Interrupt Line 5	
10	V/6	Vectored Interrupt Line 6	
11	V/7	Vectored Interrupt Line 7	
12	XRDY 2	EXTERNAL READY 2	Drugi vanjski status ulaz sličan XRDY
13-17			Nije definisano
18	STAT DSB	STATUS DISABLE	Otvara/zatvara medjuspojeve za 8 status linija
19	C/C DSB	COMMAND/CONTROL DISABLE	Otvara/zatvara medjuspojeve za 6 izlaznih komandno/kontrolnih linija
20	UNPROT	UNPROTECT	Ulaz na memorijski zaštitni flip-flop na memorijskom modulu
21	SS	SINGLE STEP	Označava da računar djeluje u režimu korak po korak (SS flip-flop je postavljen)
22	ADD DSB	ADDRESS DISABLE	Otvara/zatvara medjuspojeve za 16 adresnih linija
23	DO DSB	DATA OUT DISABLE	Otvara/zatvara medjuspojeve za 8 izlaznih podatkovnih linija
24	Φ 2	PHASE 2 CLOCK	
25	Φ 1	PHASE 1 CLOCK	
26	PHLDA	HOLD ACKNOWLEDGE	Procesorski komandno/kontrolni signal koji se pojavi kao odgovor na HOLD signal po kompletiranju tekuće naredbe; označava da podatkovni i adresni bus prelaze u stanje visoke impedanse.
27	PWAIT	WAIT	Procesorski komandno/kontrolni signal koji se pojavi kao odgovor na skok READY signala na niski nivo; označava da procesor generira seriju WAIT stanja trajanja 0.5 us, dok READY ponovo predje u visoki nivo.
28	PINTE	INTERRUPT ENABLE	Procesorski komandno/kontrolni izlazni signal koji označava dozvolu za prekide u skladu sa stanjem CPU flip-flopa za prekide. Ako je flip-flop setiran, CPU prihvata prekide, inače ne.

pin	symbol	ime	funkcija
29	A 5	Address line 5	
30	A 4	Address line 4	
31	A 3	Address line 3	
32	A 15	Address line 15	
33	A 12	Address line 12	
34	A 9	Address line 9	
35	DO 1	Data Out Line 1	
36	DO 0	Data Out Line 0	
37	A 7	Address line 7	
38	DO 4	Data Out line 4	
39	DO 5	Data Out line 5	
40	DO 6	Data Out line 6	
41	DI 2	Data In line 2	
42	DI 3	Data In line 3	
43	DI 7	Data In line 7	
44	SM 1	MACHINE CYCLE 1	Statusni izlazni signal koji označava da je procesor u feteč ciklu prvog bajta instrukcije
45	SOUT	OUTPUT	Statusni izlazni signal koji označava da je na adresnom basu adresa izlazne naprave i da će podatkovni bas sa- držati izlazni podatak čim PWR postane aktivan.
46	SINP	INPUT	Statusni izlazni signal koji označava da je na adresnom basu adresa ulazne naprave, te da ulazni podatak može biti postavljen na podatkovni bas čim PDBIN postane aktivan.
47	SMEMR	MEMORY READ	Statusni izlazni signal koji pokazuje da će podatkovni bas biti upotrijebljen za čitanje memorijskih podataka.
48	SHLTA	HALT	Statusni izlazni signal koji se odnosi na HALT naredbu.
49	CLOCK	CLOCK	Invertiran izlaz 0 2.
50	GND	GROUND	
51	+ 8 V	+ 8 Volts	Nestabiliziran ulaz za 5-Voltne stabilizatorje.
52	- 18 V	- 18 Volts	Predstabilizirani negativni napon
53	SSWI	SENSE SWITCH INPUT	Označava prenos ulaznih podataka sa senzorskih tipki. Ovaj signal se upotrebljava pri Display/Control logiki za a) uključivanje drajvera senzorskih tipki b) uključivanje Display/Control Board drajvera podatkov- nih ulaza (FD10 - FD17) c) Isključivanje CPU Board drajvera podatkovnih ulaza (D10 - D17)
54	EXT CLR	EXTERNAL CLEAR	Signal čišćenja za I/O naprave
55	RTC	REAL TIME CLOCK	60 Hz signal je upotrijebljen kao vremenska referenca za Real-Time clock/Vectorred Interrupt Board.
56	STSTB	STATUS TSROBE	Izlazni strob signal generiran u 8224 clock generatoru.
57	DIG1	DATA INPUT GATE 1	Izlazni signal sa Display/Control logike koji određuje koji podatkovno ulazni drajveri imaju kontrolu nad CPU dvosmernim basom. Ako je DIG1 na visokom nivou kon- trolu imaju CPU drajveri; ako je DIG1 na niskom nivou kontrolu imaju Display/Control drajveri.
58	FRDY	FRONT PANEL READY	Izlazni signal sa D/C logike koji omogućava kontrolu READY linija sa frontne ploče.
59-67			Nije definisano
68	MWRITE	MEMORY WRITE	Označava da podatci koji su trenutno na podatkovnom basu moraju biti upisani u memorijsku lokaciju koju određuje trenutno stanje adresnog basa.
69	PS	PROTECT STATUS	Označava stanje memorijskog zaštitnog flip-flopa na me- morijskom modulu koji je trenutno adresiran.
70	PROT	PROTECT	Ulaz u memorijski zaštitni flip-flop na memorijskom mo- dulu koji je trenutno adresiran.
71	RUN	RUN	Označava da je 64/RUN flip-flop resetiran-računar izvo- di program.
72	PRDY	PROCESSOR READY	Memorijski I I/O ulazi u CPU čekaju u zanki.
73	PINT	INTERRUPT REQUEST	Procesor prepoznaje prekidni zahtjev na ovoj liniji po kompletiranju tekuće instrukcije ili ako je zaustavljen. Ako je procesor u HOLD stanju ili ako je prekidni flip-flop resetiran, zahtjev za prekidom će biti ignorisan.
74	PHOLD	HOLD	Procesorski komandno/kontrolni ulazni signal koji zahti- jeva procesorsko HOLD stanje.
75	PRESET	RESET	Procesorski komandno/kontrolni ulaz; čim je aktiviran

pin	symbol	ime	funkcija
76	PSYNC	SYNC	očisti se sadržina programskog brojača te instrukcijskog registra u CPU.
77	PWR	WRITE	Procesorski komandno/kontrolni izlaz; označava početak svakog mašinskog ciklusa.
78	DDBIN	DATA BUS IN	Procesorski komandno/kontrolni izlaz; upotrebljava se pri upisivanju u memorijsku ili za I/O izlaznu kontrolu. Podatci na podatkovnom basu su stabilni dok je PWR aktivan.
79	A0	Address Line 0	Procesorski komandno/kontrolni izlaz; označava da je podatkovni bas u ulaznom načinu.
80	A1	Address Line 1	
81	A2	Address Line 2	
82	A6	Address Line 6	
83	A7	Address Line 7	
84	A8	Address Line 8	
85	A13	Address Line 13	
86	A14	Address Line 14	
87	A11	Address Line 11	
88	DO2	Data Out Line 2	
89	DO3	Data Out Line 3	
90	DO7	Data Out Line 7	
91	DI4	Data In Line 4	
92	DI5	Data In Line 5	
93	DI6	Data In Line 6	
94	DI1	Data In Line 1	
95	DI0	Data In Line 0	
96	SINTA	INTERRUPT ACKNOWLEDGE	Statusni izlazni signal; formira signal za prekidni zahtjev.
97	SWO	WRITE OUT	Statusni izlazni signal; označava da će operacija u tekućem mašinskom ciklusu biti WRITE memory ili izlazna funkcija.
98	SSTACK	STACK	Statusni izlazni signal; pokazuje da adresni bas drži sadržaj pokazivača steka(stack printer).
99	POC	POWER-ON CLEAR	
100	GND	GROUND	

drugo svetovno šahovsko prvenstvo računalnikov

i. bratko

UDK 794.1.092.2:681.3

Fakulteta za elektrotehniko, Institut "Jožef Stefan"
Univerza v Ljubljani

Med kongresom IFIP 77 avgusta 1977 v Torontu je potekalo tudi drugo svetovno šahovsko prvenstvo računalnikov, oz. natančneje šahovskih programov. Nastopilo je 16 programov, ki so bili na prvenstvo uvrščeni po uspehih na drugih turnirjih ali pa po predhodnih izločilnih kvalifikacijah.

Glavna favorita za prvo mesto sta bila pred začetkom prvo in drugouvrščeni s prvega svetovnega prvenstva l. 1974 v Stockholmu, sovjetski program KAISSA in ameriški CHESS. Avtorji so za Toronto pripravili novi verziji svojih programov s številnimi izboljšavami. Zanimanje za to, ali bo v velikem obračunu programu CHESS 4.6 uspelo maščevanje za Stockholm, je bilo izredno, med drugim tudi zato, ker je CHESS 4.6 pred nekaj meseci nastopal na človeških turnirjih in beležil senzacionalne

uspehe. Med drugim je spomladi 1977 v brzopoteznih partijah premagal tudi mednarodne mojstre Levyja in Berlinerja, izgubil pa partijo pod turnirskimi pogoji (tempo 40 potez v 2,5 urah) z Levyjem.

Velika neznanka je bil program Mihaila Botvinnika, bivšega svetovnega šahovskega prvaka, ki je bil najavljen na startni listi nekaj mesecev pred turnirjem. Botvinnikov projekt je dosegel izjemno publiciteto, ne nazadnje je Botvinnik objavil dve knjigi o svojih konceptih o programiranju šaha. Toda Botvinnikov program se ni pojavil na startu v Torontu in vse kaže, da Botvinniku s svojo ekipo še 10 let po izidu njegove prve knjige ni uspelo implementirati svojih idej.

Turnir v Torontu je potekal po švicarskem sistemu 4-ih

Program	Računalnik	Avtorji	Programirni jezik	Potrebne pomnilne kapacitete	Povprečno število pregledanih pozicij na potezo	Osvojene točke
CHESS 4.6	CDC CYBER 176	D. Slate, L. Atkin Northwestern University Illinois, USA	assembler	7,5k besed (60 bitov) + Ext. core	400 000	4
DUCHESS	IBM 370/168	T. Truscott, B. Wright, E. Jensen Duke University, Durham, N.C., USA	PL/1 in assembler	300k	1200	3
KAISSA	IBM 370/168	M.V. Donskoj, V. Arlazarov Institut za študij sistemov Moskva, SSSR	assembler	250k	90 000	3
BELLE	PDP-11	K. Thompson, J. Condon, Belle Tel. Labs, Murray Hill, New Jersey USA	"C"	8k besed (16 bitov)	30 000	2,5
CHAOS	AMDAHL 470 V/6	M. Alexander Idr., University of Michigan, Ann Arbor, Michigan, USA	fortran	3 Mega byte	30 000	2,5

TABELA : Podatki o petih prvouvrščenih šahovskih programih s prvenstva v Torontu

kol. Pri tem sistemu igrajo vodilni udeleženci med seboj, o končnem vrstnem redu pa odloča celotni seštev ek osvojenih točk. Tempo igranja je bil 2,5 ure za 40 potez, kar ustreza običajnim turnirskim pravilom.

Že prvo kolo je prineslo senzacionalen poraz KAISSE z malo znanim ameriškim programom DUCHESS in s tem praktično že odločitev o prvem mestu. V derbiju tretjega kola je potem CHESS 4.6 odpravil DUCHESS, medtem ko so ti trije vodilni programi dobili vse preostale partije z drugimi nasprotniki. Tako je bil končni vrstni: 1. CHESS 4.6 (ZDA) 4 točke, 2-3. DUCHESS (ZDA) in KAISSA (SZ) po 3 točke, 4-5. BELLE in CHAOS (oba ZDA) po 2,5 točke, 6-10. MASTER (Vel. Britanija), BLACK KNIGHT (ZDA), DARK HORSE (Švedska), ELSA (ZR Nemčija) in WITA (Kanada) vsi po 2 točki itd.

V partiji DUCHESS-KAISSE je prišlo še do enega zanimivega dogodka, ki je povzročil številne nasprotujoče si komentarje. Nekje v toku partije je namreč KAISSA nastavila nebranjeno trdnjavo, ki jo je DUCHESS seveda vzela in potem z lahkoto slavila v nadaljevanju. Vsi prisotni so potezo KAISSE s trdnjavo ocenili kot nerazumljivo napako. Po splošnem prepričanju je bila to posledica kakšne banalne programerske napake, ki je ostala dotlej še neodkrita. Sovjetska ekipa je v paniki poizkušala čez noč pred naslednjo partijo odpraviti domnevno napako. Ven-

dar pa je računalniški izpis s podrobno analizo variant, ki jih je pregledala KAISSA pred usodno potezo, pokazal, da je bila poteza s trdnjavo v resnici briljantna žrtev, s katero je KAISSA preprečila mat in podaljšala partijo še za nekaj časa. V nasprotnem primeru bi bil namreč možen forsiran mat v petih potezah z nepričakovano žrtvijo kraljice! Tega mata pa nihče v dvorani med partijo ni videl, niti znani šahisti kot je Botvinnik, ki je bil med prisotnimi.

Programi, ki so bili prisotni v Torontu, temeljijo povečini na znani Shannonovi shemi s tem, da uporabljajo ocenitveno funkcijo za ocenjevanje pozicij in preizkušajo možna nadaljevanja v skladu z minimaks principom. Pri tem je minimaks postopek implementiran z znanim alfa-beta algoritmom (npr. N. Nilsson, Problem Solving Methods in Artificial Intelligence, Mc Graw-Hill (1971)), ki da isti rezultat kot minimaks, je pa veliko hitrejši. V tabeli so zbrani nekateri zanimivejši podatki o najboljših programih iz Toronta. Impresionira podatek, da četrtouvrščen program BELL teče na miniračunalniku PDP 11 in vsega 8k besed spomina. CHESS 4.6 preišče pri izboru poteze drevo, ki vsebuje v povprečju 400 000 pozicij. Da število preglednih pozicij ni nujno potreben pogoj za uspeh dokazuje DUCHESS s skromnimi 1200 pozicijami.

tendence v razvoju mikro procesorjev

Za marec 1978 je ZILOG objavil spisek Z8 krmilnikov in za mesec junij izdelavo 16 bitnega mikro procesorja Z8000. Osembitni procesor Z8 je namenjen široki uporabi predvsem za samostojne aplikacije in povsod tam, kjer ni zahtev po povečanem obsegu pomnilnika. V čip je vključen 2K zložni ROM in 96K zložni RAM. Čip ima 32 vhodnih/izhodnih linij. Te linije se lahko uporabljajo za zunanje adresiranje. Na tak način se pridobijo še nadaljnji zlogi, tj. omogočeno je 128K zlogov zunanjega pomnilnika, ki se deli na dva dela: podatkovnega in programskega.

V internem pomnilniku (beležki) je 76 zlogov RAM namenjenih za uporabniške programe, medtem ko ostalih 20 zlogov pomnilnika pripada štirim vhodno/izhodnim vratom, ki se oblikujejo za vhodno/izhodno vodilo. Med naštetimi prednostmi vsebuje Z8 še interni vektorsko nastavljeni večnivojski prekinitveni mehanizem, dva časovnika, števniki za štetje dogodkov in eno samo 5 V napajanje.

Za Z8000 (Z86) se pričakuje, da bo imel procesno zmogljivost, ki je primerljiva z PDP 11/70 mini računalnikom, uporabljal bo tudi podobno arhitekturo. Zbirni jezik za Z8000 bo razširitev zbirnega jezika za Z-80 in bo imel še dodatne ukaze, kot sta ukaza za množenje in deljenje. Ohišje bo v izvedbi s 40 ali 48 nogicami, kar pa še ni določeno. Z8000 ima tudi možnost direktne priključitve 8 M zlogov zunanjega pomnilnika.

Prihod novega 16 bitnega paralelnega procesorja Intel 8086 je bil za nekaj časa skrivnost. 16 bitni 8086, ki ima enake procesne zmogljivosti kot PDP 11/45, je na seznamu za proizvodnjo v prvem četrtletju tega leta. Procesor bo programsko in materialno kompatibilen s procesorjema 8080 in 8085, s tem, da je hitrost obdelave petkrat večja kot pri procesorju 8080. Da bi dosegli optimalne hitrosti v obdelavi podatkov priporoča Intel pretvorbo kodov. To bi izboljšalo lastnosti mikro procesorja pri obdelavi prekodiranih programov tudi do 15 krat. Načrtujejo pa tudi povečano število instrukcij. V zvezi s tem pa se pripravljajo tudi nova verzija Intelovega visokonivojskega jezika PL/M86.

Čeprav je na procesor 8086 mogoče priključiti že razvite periferne naprave bodo za boljše lastnosti pri komuniciranju z zunanji napravami razvili vhodno/izhodni procesor 8089, ki bo v prodaji konec letošnjega leta. V teku letošnjega leta napovedujejo izpeljavo procesorja 8086, ki bo poznan kot 8088. Imel bo enako notranjo zgradbo kot 8086 vendar 8 bitni zunanji podatkovni vod.

Na področju kontrolerjev je Intel znižal ceno za 8748 z EPROM pomnilniško varianto 8084. Znižanje cene bo ugodno vplivalo na povečano uporabo v industrijskih napravah, kjer bi se prvenstveno uporabljali kot del za razvoj. Ravno tako je nov proizvod procesor 8021, ki je poenostavljena izvedba procesorja 8084. Zgrajen je v izvedbi z 28 priključki. Vsebuje 1 K zložni ROM in 64 zložni RAM pomnilnik.

Tudi Motorola se pripravljajo, da na tržišču nastopi konec leta in sicer s 16 bitnim mikroprocesorjem. Imenoval se bo Motorola Advanced Computer System (MACS). Imel bo enako procesno moč kot PDP 11/45.

Proizvajali bodo tudi izboljšano verzijo procesorjev 6800 z oznako 6809. Ta verzija bo imela enak strojni kod kot 6800 vendar lahko operira s 16 bitnimi podatkovnimi besedami in naslavlja več registrov, kot da bi bil en sam register.

Kot samostojni procesor kontroler je Motorolin procesor 6801 z naslednjimi prednostmi: 1 K zložni ROM in 96 zložni RAM pomnilnik s prilagodljivo vhodno/izhodno strukturo, ki se naslanja na 28 vhodno-izhodnih linij. Tudi ta procesor bo programsko kompatibilen s 6800.

Iz Computer Products International priredil
R. Čop

novice in zanimivosti

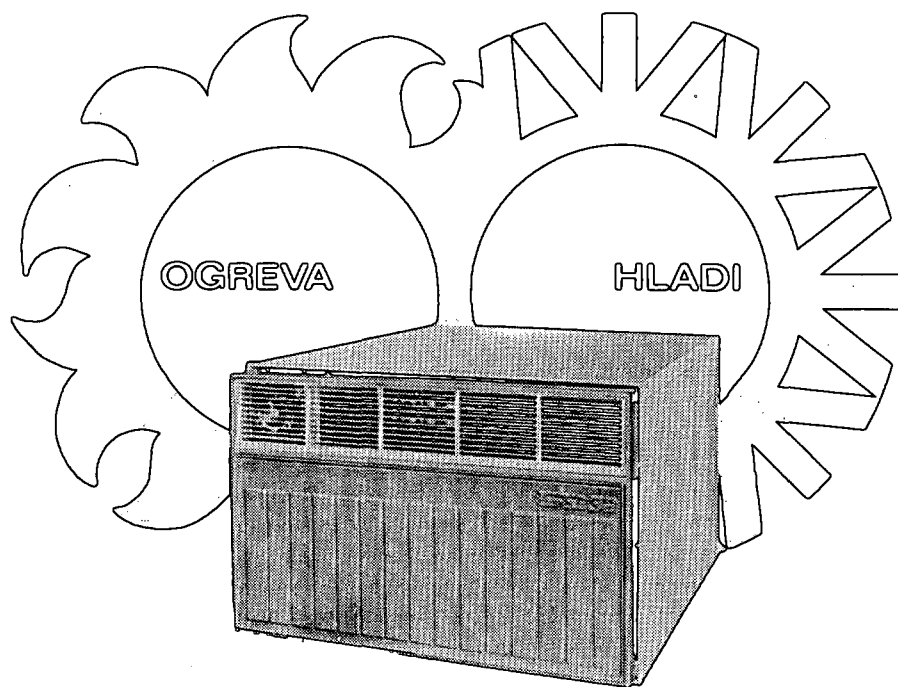
Klub uporabnikov sistemov s procesorjem 6800

Ta klub zanima vse tisto, kar je povezano z uporabo in gradnjo mikro računalnikov s procesorjem 6800. Amaterji v tem klubu razvijajo programsko in materialno opremo ter iščejo stike z vsemi, ki se kreativno ukvarjajo z razvojem sistemov s procesorjem 6800. Naslov kluba je: 6800 Computer Club, P.O.B. 18081, San Jose, CA 95118, USA.

Center za razvoj integriranih vezij v Avstriji

V bližnji prihodnosti je moč pričakovati povečano potrebo po integriranih vezjih (kratko IV) za posebno uporabo, ki se izdelujejo v manjših količinah. Velika podjetja, ki proizvajajo IV velikoserijsko, niso zainteresirana za takšno proizvodnjo! Spodnja meja maloserijske proizvodnje je med 50 do 100 tisoč kosi in stroški razvoja ne smejo preseči 10% celotnih stroškov. Maloserijska proizvodnja predvideva le nekaj sto kosov določenega IV, kjer smejo znašati stroški razvoja 50 do 70 %, pri 100 kosih pa celo 80 % celotnih stroškov. To pomeni, da morajo biti stroški proizvodnje zelo majhni, praktično enaki ničli. Razvoj posebnih IV naj bi v Avstriji prevzeli tisti inženirski biroji, ki se sedaj ukvarjajo z razvojem in maloserijsko proizvodnjo specialnih aparatov. Država bi ta razvoj podprla tako, da bi ustanovila poseben razvojni center, v katerem bi se izdelovali načrti in opravljala tudi proizvodnja. Difuzijski procesi naj bi se izvajali v sodelovanju s podjetji, ki razpolagajo z difuzijskimi napravami. Kasneje bi ustanovili majhno podjetje, ki bi se ukvarjalo samo s tovrstno, maloserijsko proizvodnjo. Tehnična univerza na Dunaju bi se vključila v ta razvoj tako, da bi dopolnila opremo svojega polprevodniškega laboratorija v iznosu 5 milijonov šilingov; s tem bi ta laboratorij lahko prevzel začasno funkcijo razvojnega centra za IV. Organizacija dela v razvojnem centru bi morala biti industrijska ali njej podobna. Stroški izgradnje sodobnega razvojnega centra za IV pa bi bili znatno višji in so bili ocenjeni na nekaj deset milijonov šilingov.

8K x 8ROM. Firma National Semiconductor prodaja 65 536-bitni statični ROM pomnilnik (INS8364), ki je



VEDNO PRIJETNO POČUTJE

Rešitev vseh problemov klimatiziranja:

KLIMATIZER TOBI 32

KLIMA OMARE KO

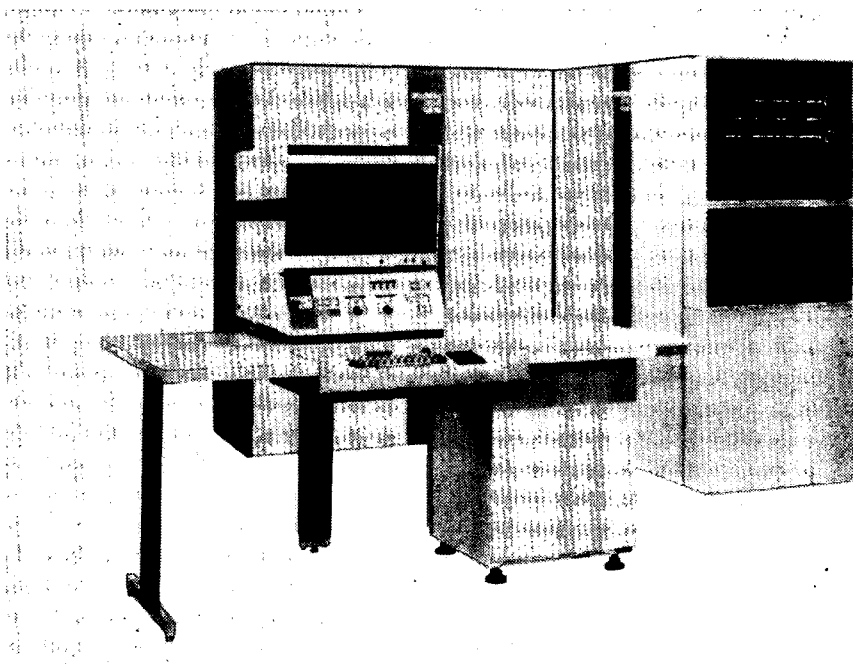
- Samodejno uravnavanje temperature v prostoru
- Enostavna montaža in vzdrževanje
- Servisna mreža in rezervni deli na celotnem področju SFRJ

INFORMACIJE O PRODAJI IN INŽENIRINGU:

EMO
63000 CELJE
Mariborska 86
tel.: (063) 23-921

ISKRA - TOZD INŽENIRINGI
61000 LJUBLJANA
Kotnikova 6
tel.: (061) 312-322

PORODICA FACOM FIRME FUJITSU



Radeći bez mnogo buke, ali marljivo posljednje četiri godine, FUJITSU je zajedno sa svojim zastupnikom ZPR-om (Zavod za primjenu elektroničkih računala) uspješno sklopila ugovore za više od 50 FACOM računala u Jugoslaviji.

Iznenadjeni? Ne morate biti. FUJITSU, povrh toga što je vodeći proizvođač sistema za elektroničku obradu podataka na Japanskom tržištu, vrlo brzo preuzima jedno od vodećih mjesta i na svjetskom tržištu. Tajna uspjeha firme FUJITSU je u tome što ima vodeću tehnologiju u kombinaciji sa velikom pouzdanošću sistema i dobro organiziranom službom za održavanje i stručnu pomoć. Ne smijemo zaboraviti ni konkurentne cijene koje će vam dati najbolji mogući odnos cijena/performance.

Ako razmišljate o uvođenju elektronske obrade podataka u vašoj organizaciji ili želite da poboljšate svoj sadašnji sistem, obratite se predstavnicima firme FUJITSU da vas upoznaju sa svim novostima.

FUJITSU proizvodi sve - sastavne dijelove, memorije, off i on-line uredjaje za prikupljanje podataka, inteligentne terminale, micro procesore i micro računala, malih, srednjih, velikih i super velikih kompjutera, uključujući najsnažnije kompjutere za opću svrhu koji se mogu kupiti na tržištu.

FUJITSU je poznata i u području telekomunikacija. To je razumljivo zbog toga što je FUJITSU jedan od vodećih proizvođača telefona i telekomunikacija u Japanu. FUJITSU je u Jugoslaviji izabrana da snabdije i pomogne kod razvoja najveće on-line real-time mreže koja je do sada ugovorena, uključujući oko 300 terminala. Mislimo da biste sebi i svojoj organizaciji učinili mnogo, ako saznate više o tome što Vam firma FUJITSU može ponuditi.

Servisni centri i uredi su u Ljubljani, Mariboru, Beogradu i Zagrebu

ZPR
ZAGREB, Savska 56
tel. 518-706
Telex 21689 YU ZPR FJ



LJUBLJANA, Topniška 45
tel. 311-059

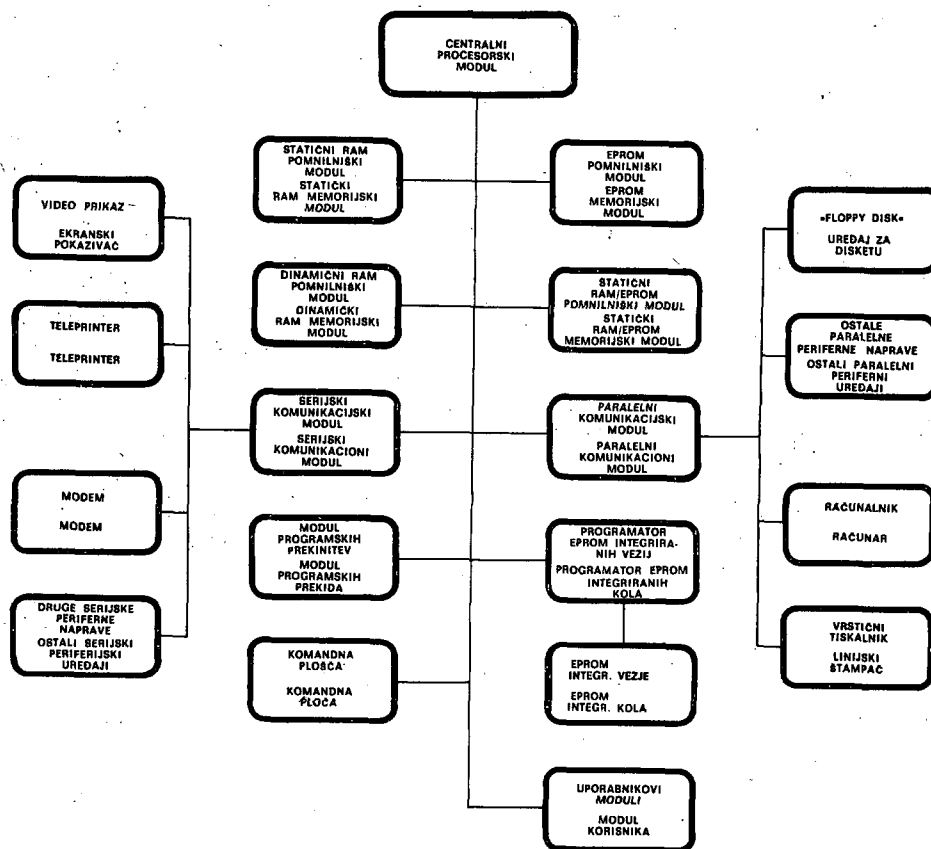
FUJITSU LIMITED
Communications and Electronics



imamo, što ste tražili ! iskradata 1680 prvi jugoslovenski mikroračunar

Konfiguracija mikroračunalnika Iskradata 1680

Konfiguracija mikroračunara Iskradata 1680



Mikroračunar Iskradata 1680 je zasnovan na mikroprocesorskoj seriji MOTOROLA 6800. Sistem je izradjen u modularnoj tehnici, tako da svaki modul (ploča sa štampanim kolima, format dvojna evropa) predstavlja električni zatvorenu celinu. Zajednički sistem vodjica za ploče, na koji dolaze signali zajednički svim modulima, omogućava korisniku proširivanje osnovnog sistema, odnosno slobodan izbor konfiguracije mikroračunara Iskradata 1680.

Ulazno/izlazni moduli povezuju uređaj sa proizvoljnim perifernim uređajima i time otvaraju široke mogućnosti komuniciranja.

Programski deo mikroračunara Iskradata 1680 obezbeđuje korisniku jednostavno pisanje programa i vezu sa programskim komandama.

Mikroračunar Iskradata 1680 sačinjavaju moduli sa zajedničkim vodjocima zbog toga mogu zauzimati proizvoljno mesto u uređaju. Moduli nemaju fizičke adrese i zato mogu dobiti proizvoljnu adresu u okviru celog adresnog područja. U svojoj strukturi svaki modul ima relativnu adresu, koja se uključuje na samom povezivanjem ili pomoću preklopnika.

Software mikroračunara Iskradata 1680 omogućava korisniku jednostavno programiranje sistema. Kontrolni program (monitor) sadrži sve potrebne rutine (naredbe) potrebne za upravljanje sa mikroračunarom (zapamti, pokaži itd). Kontrolni program je upisan na EPROM integrisanom kolu.

Rezidentni asemblerski prevodilac omogućava korisniku jednostavno pisanje programa u mnemoničkim rečima. (source program).

Makro assembler/disassemblerski prevodilac služi za pisanje programa u makro assembleru, odnosno za pretvaranje mnemoničkih naredbi u heksadecimalni kod.

Korisnik ima takodje mogućnost programiranja u višem programskom jeziku BASIC.

Korisnik može izabrati prevedeni program upisan u EPROM memoriji, odnosno na bušenoj papirnoj traci, koji potom sam svaki put unosi u RAM memoriju.

U budućnosti bićemo u mogućnosti da isporučimo pored mikroračunara Iskradata 1680 još i potrebne periferne jedinice (ekranski pokazivač, teleprinter, uređaj za disketu, linijski štampač itd.)

Pored modula prikazanih na slici, imamo i :
Brisač EPROM-a je u samostalno kućište ugradjena ultravioletna lampa za brisanje EPROM-a sa vremenskim uključenjem brisanja.

U pripremi su još sledeći moduli: časovnik realnog vremena, DMA moduli, D/A i A/D konverter, koji će omogućiti još veću univerzalnost i sveukupnost sistema mikroračunara Iskradata 1680,

ISKRA, Industrija za telekomunikacije, elektroniku in elektromehaniku, Kranj, TOZD Računalniki, 64000 Kranj PE Ljubljana, 61000 Ljubljana, Titova 81 - telefon: (061) 326-367.

INFORMATICA 78

Bled, 2. - 7. oktober 1978

simpozij

13. jugoslovanski mednarodni simpozij o obravnavanju podatkov

Bled, 2. - 7. oktober 1978

seminarji

izbrana poglavja računalniških znanosti

Bled, 3. - 6. oktober 1978

razstava

računalniška oprema in literatura

Bled, 1. - 7. oktober 1978

organizator

Slovensko društvo INFORMATIKA v sodelovanju z Institutom Jožef Stefan in Fakulteto za elektrotehniko, Ljubljana

roki

20. marec 1978 - zadnji rok za sprejem formularja s prijavo in 2 izvodov razširjenega povzetka

30. april 1978 - pošiljanje rezultatov recenzije in avtorskega kompleta

20. junij 1978 - zadnji rok za sprejem končnega teksta prispevka

nadaljnje informacije:

INFORMATICA 78

Institut Jožef Stefan, 6001 Ljubljana, pp. 199

telefon : (061) 263 261, telegram : JOSTIN, Ljubljana

telex : 31 296 YU JOSTIN

S tradicionalnim posvetovanjem "Simpozij in seminarji INFORMATICA" slovensko društvo INFORMATIKA nadaljuje aktivnost Zveznega strokovnega odbora za obravnavanje podatkov pri Jugoslovanskem komiteju za ETAN.

To posvetovanje je postalo tako po udeležbi kot po tehtnosti objavljenih del osrednje jugoslovanske srečanje teoretikov in praktikov s področja obravnavanja podatkov.

Lanskemu simpoziju, na katerem je bilo predstavljenih 247 tujih in domačih del, je prisostvovalo 443 strokovnjakov. Zaradi hitrega vzpona znanstvenih in strokovnih računalniških moči v Jugoslaviji in sosednjih deželah pričakujemo, da bo na letošnjem simpoziju in seminarjih sodelovalo še večje število predavateljev in poslušalcev. Tudi letošnji simpozij bo mednarodnega značaja. V vabljenih uvodnih predavanjih in na seminarjih, ki bodo potekali vzporedno s simpozijem, bodo priznani tuji in domači strokovnjaki pregledno predstavili najnovejše dosežke iz izbranih področij računalništva in informatike.

Za naše strokovnjake bo simpozij priložnost, da v teku uradnega dela simpozija in v neformalnih srečanjih podvržejo svoja dognanja kritični oceni priznanih tujih in domačih strokovnjakov. Upravičeno smemo pričakovati, da bo simpozij pomemben prispevek k medsebojnemu povezovanju ter izmenjavi izkušenj na področju računalništva.

Vabimo vas, da se s prispevkom aktivno udeležite letošnjega posvetovanja.

jezik simpozija

Zaradi mednarodnega značaja simpozija vabimo avtorje, da predložijo in predstavijo svoja dela v angleškem jeziku, kar zagotavlja večji krog poslušalcev in bralcev. Seveda bodo v program uvrščena tudi dela, napisana v kateremkoli izmed jugoslovanskih jezikov.

zbornik del

Vsak udeleženec bo prejel zbornik del ob prihodu na simpozij, vsak avtor pa bo poleg tega prejel še pet kopij svojega prispevka.

prijavnina

Za simpozij : 2.000 din

Za seminarje : 500 din (velja za cikel treh seminarjev)

Za študente velja 50% popust

Avtorji plačajo enak znesek kot drugi udeleženci.

INFORMATICA 78

Bled, October 2 - 7, 1978

Symposium

13th Yugoslav International Symposium on Information Processing

Bled, October 2 - 7, 1978

Seminars

Selected Topics in Computer Science

Bled, October 3 - 6, 1978

Exhibition

Computer Equipment and Literature

Bled, October 1 - 7, 1978

Organizer

INFORMATIKA, Slovene Computer Society in co-operation with Jožef Stefan Institute, and Faculty of Electrical Engineering, University of Ljubljana

Deadlines

March 20, 1978 - submission of the application form and 2 copies of extended summary

April 30, 1978 - mailing out of the summary reviews and author kits

June 20, 1978 - submission of the full text of contribution

Further information

INFORMATICA 78

Institut Jožef Stefan, 61001 Ljubljana, pp. 199, Yugoslavia

Phone : (061) 263 261, Cable: JOSTIN Ljubljana

Telex : 31 296 YU JOSTIN

The traditional Bled computer conference, "Symposium and Seminars INFORMATICA", carries on its role as the major Yugoslav meeting of professionals in the field of computers and information processing

Last year, 443 experts from Yugoslavia and abroad attended the meeting and presented 247 papers.

In view of the recent upsurge of computer-related activities in Yugoslavia and in its neighbouring countries, even stronger participation is expected this year.

As in previous years, this will be an international symposium. In their invited papers and at the seminars, eminent foreign experts will present surveys of the latest achievements in selected fields of information science and technology.

The sponsors and the organizing committee believe that INFORMATICA 78 will be a good opportunity for the exchange of ideas and experiences in the field of information processing. As a meeting of people from East and West it will also be an opportunity to further mutual understanding and co-operation.

You are invited to take part in the symposium with submitted papers.

General Information

In view of the international character of the symposium, authors are invited to write and present their contributions in English in order to benefit from broader audiences. However, papers written in any of the Yugoslav languages will also be included in the program.

Proceedings

Participants will receive the proceedings upon arrival. Each author will also receive five copies of his paper.

Registration Fee

For symposium : 2.000 din

For seminars : 500 din (for cycle of 3 seminars)

Student registration will be half price.

Authors pay full registration fee.

organiziran v obliki 8192 8-bitnih besed. Podatkovni izhodi treh stanj in vhodi za izbiro čipa, ki jih je mogoče programirati, omogočajo širjenje pomnilnika. Vsi vhodi in izhodi so TTL kompatibilni. INS8364 je mogoče dobiti tudi kot INS8298 8080A LL "BASIC" interpreter in Hex Debugger. Hitrost dostopa je 450 nsec, čip pa potrebuje samo eno napajalno napetost +5 V.

ALGOL, SNOBOL za sistem s procesorjem 8080

Uporabniki sistemov z vodilom S-100 lahko sedaj kupijo simulator, s katerim je mogoče simulirati skoraj vse programske pakete, ki so pisani za PDP-8, na sistemih s procesorji 8080/8085/Z-80. Brez modifikacije je mogoče uporabljati tudi višje programirne jezike kot so: FOCAL, Fortran II, ALGOL, LISA in SNOBOL ter asemblerski jezik, editor in programski paket za aritmetiko s plavajočo vejico. Edina osvojitvev je hitrost pri BASIC-u in FOCAL-u, sicer pa se izvaja točna simulacija procesorja PDP-8. Uporabnik programira svoj sistem, kot da bi delal z računalnikom PDP-8. Za kompatibilni sistem 8080/Z-80, ki simulira PDP-8 je potrebno: 12K RAM pomnilnik, CPU, vmesnik za kasetni pogon, čitalnik luknjanega traku ali floppy disc in običajen V-24 terminal. Za 89.000 Sch je že mogoče dobiti razvojni sistem IMSAI 8080 in PDP-8 simulatorski sistem. Nekoliko dražji je sistem z Z-80 - CPU. Prednost te ponudbe je v obliki programskih paketov za PDP-8, ki jih je mogoče dobiti po ugodnih cenah pri Digital Equipment ali DECUS-u. Simulator je mogoče kupiti pri Margreiter GesmbH, Dunaj.

Nekateri raziskovalni in razvojni projekti s področja mikro računalnikov in obdelava podatkov na Japonskem.

V spisku so navedeni nekateri projekti, ki se izvajajo na japonskih univerzah ter podjetjih, in so zanimivi tudi za nas. Podatki se nanašajo na stanje v septembru 1977. Spisek je tale (naslov projekta:nosilec:naslov ustanove):

1/ Lisp machine using a micr computer as a control unit : Prof. Kotaro Mano; College of Science and Eng. Aoyama Gakuin University, 6-16-1 Chitosedai Setagaya-ku, Tokyo.

2/ A new model of distributed type associative memory : Kenji Murakami; Faculty of Engineering, Ehime University; Matsuyama-shi, 790 Japan.

3/ Microprogram Generator : Takanobu Baba; Department of Computer Science University of Electro-Communications; 1-5-1 Chofuga-oka, Chofu-shi, Tokyo 182 Japan.

4/ BASIC-S Project : Mitsuo Kobayashi; Department of Computer Science, University of Electro-Communications; 1-5-1 Chofuga-oka, Chofu-shi, Tokyo 182 Japan.

5/ On a question-answering for fuzzy informations: Division of Information Engineering, Graduate School of Engineering, Hokkaido University; Kita-ku, Kita 13-jo Nishi 8-chome, Sapporo, 060 Japan.

6/ Implementation of PASCAL Compiler by Bootstrapping Technique : Eiichi Miyamoto; Division of Information Engineering, Graduate School of Engineering, Hokkaido University.

7/ SL/M : A Microcomputer-Oriented System Description Language : Keiji Makino; Dept. of Electronic

Engineering, Faculty of Engineering, Hokkaido University.

8/ uCAD (Computer Aided Design for Microprogrammable Microprocessor Application Systems): Mario Tokoro, Ph. D.; AISO Research Group, Dept. of Electrical Engineering, Keio University, 832 Hiyoshicho, Kohoku-ku, Yokohama, Japan.

9/ The Changeable Floppy Disk Operating System depending upon the Diskette : M. Kitagawa; Faculty of Engineering, Keio University.

10/ Simulation of computer system with buffer memory: Tomoo Nakamura; Department of Engineering, Kyoto University; Yoshida Honmachi, Sakyo-ku, Kyoto, Japan.

11/ Development of a Universal Assembler : Prof. Toshiyuki Sakai; Dept. of Information Science, Faculty of Engineering, Kyoto University.

12/ An Interactive Picture Processing System on a Minicomputer : Makota Nagao; Dept. of Electrical Eng., Kyoto University.

13/ Implementation of Associative Data Processing System a Multi-computer : Terumasa Hosomi; Dept. of Information and Computer Sciences, Osaka University. Machikaneyama, Toyonaka, 560 Japan.

14/ Shortening the length of programs by using subroutines : Yoshiki Katsuyama; Dept. of Information and Computer Sciences, Faculty of Engineering Science, Osaka University, Toyonaka, Osaka 560, Japan.

15/ Networking for inter-university computer centers in Japan : Hiroshi Inose, Director, Computer Center, University of Tokyo; Dept. of Information Science, Faculty of Science, University of Tokyo; Yayoi 2-11-16, Bunkyo-ku, Tokyo, 112, Japan.

16/ Job Management System of Computer Network TECNET : Hidekido Tanaka; Moto-oka Lab., Dept. of Electrical Engineering, Faculty of Engineering, The University of Tokyo; 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan.

17/ Implementation and Application of Concurrent Pascal : Masaru Watanabe; Institute of Industrial Science, University of Tokyo; Roppongi 7-22-1, Minato-ku, Tokyo, Japan

18/ Implementation of a compiler for the programming language Pascal : Teruo Hikita, Computer Center, Univ. of Tokyo

19/ Implementation of a compiler for the programming language Pascal : Kiyoshi Ishihata; Dept. of Information Science, Faculty of Science, University of Tokyo; 2-11-16 Yayoi, Bankyo-ku, Tokyo 113, Japan.

20/ Address Mapping with Software : Morio Onoe, Institute of Industrial Science, Univ. of Tokyo; Roppongi 7-22-1 Minato-ku, Tokyo 106 Japan.

21/ Pattern recognition of the handwritten numerals : Toyoshi Shimaoka; Junior College Dept., Yamaguchi University; Ogushi Ube City Yamaguchi.

22/ Development of Multi-Micro Processor System : Makoto Ito; Faculty of Engineering, Yamanashi University; 4-3-11 Takeda, Kofu, Yamanashi, 400, Japan.

23/ System Program for Multi-Micro-Processors : Makoto Ito; Faculty of Engineering, Yamanashi Univ., 4-3-11 Takeda, Kofu, Yamanashi, 400, Japan.

24/ Hierarchical Programming Language System : Makoto Arisawa; Computer Science Dept. Yamanashi University.

25/ Parallel Processing System for Multioperand Arithmetic Operations : Hiroyoshi Ohara; School of Science and Engineering, Waseda Univ., 4-170 Nishi-Ohkubo, Shinjuku-ku, Tokyo 160 Japan.

26/ Development of a high-performance microprocessor architecture : H. Iizuka; Electrotechnical Lab., 6-1, 2-chome, Nagata-cho, Chiyoda-ku, Tokyo, Japan.

27/ A model and analysis of bus-connected multi-processor systems : T. Furutani; Electrotechnical Lab., 6-1, 2-chome, Nagata-cho, Chiyoda-ku, Tokyo, Japan.

28/ A Design of Mutual Exclusion Control Module for a Multiprocessor System; Electrotechnical Lab., 6-1, 2-chome, Nagata-cho, Chiyoda-ku, Tokyo Japan.

29/ Personal LISP Machine : Toshio Shimada; Electrotechnical Lab., 6-1, 2-chome, Nagata-cho, Chiyoda-ku, Tokyo, Japan.

30/ Study on Simulation Program System for Various Type of Arithmetic Instructions of Aerospace Computer: Masao Naka; National Aerospace Laboratory; 1880 Jindaiji-machi, Chofu, Tokyo 182, Japan.

Japonski mikroročunalniški klub. Število članov tega kluba je že v letu 1976 preseglo 1000. Klub je zainteresiran za izmenjavo izkušenj, časopisov, informacij, itn. Naslov je : JAPAN MICROCOMPUTER CLUB, Mr. Koji Yada, Manager, Computer Center, Electrotechnical Laboratory, Tanaski Branch, 5-4-1 Mukodi-Machi, Tanaski-shi, Tokyo, Japan.

Amaterski računalniški klub Essex Anglija obstaja že pet let in ima več kot tisoč članov, tako iz Velike Britanije kot iz oddaljenih krajev sveta. Člani kluba so povezani s časopisom, ki izhaja šestkrat letno. Vsebina časopisa je tako iz hardverskega kot iz softverskega področja. Letna naročnina znaša 2 funta. Naslov kluba je : Amateur Computer Club at 7 Dordells, Basildon, Essex, England.

Tvrdka Burr-Brown je najavila nov čip z oznako MP20, t.j. mikroprocesorski analogni vhod z 16 kanalnim analognim multiprocesorjem, ojačevalnikom z visokim ojačanjem, 8 bitnim analogno-digitalnim pretvornikom in potrebnimi vmesniki vodil.

MP20 je časovno in logično združljiv s procesorji 8080A in 8008. Ne zahteva se zunanja logika, Absolutna točnost je boljša od $\pm 0,4\%$ pri $\pm 5V$ ali napetostnem območju od 0 do $+5V$.

Nizko nivojske signale kot so termosklopljeni izhodi lahko uporabimo neposredno. Ojačevalnik lahko programiramo od zunaj s pomočjo zunanjega upora, tako da lahko uporabimo vhodne signale velikosti $\pm 10mV$.

Dejansko lahko smatramo MP20 kot pomnilnik. Vsak analogni vhodni signal zasede eno pomnilniško lokacijo. Pomnilniška referenčna instrukcija se lahko smatra kot dostopen podatek. Tako s pomočjo LDA prenesemo vhodni podatek iz enega kanala v akumulator.

Potrební enosmerni napetosti sta $\pm 15V$ in $+5V$. Za podrobnejše informacije pišite na naslov: Burr-Brown, International Airport Industrial Park, Tucson AZ 85734.

Texas Instruments ima na tržištu podatkovni terminal

Silent 700 sa memorijom sa magnetnim mjehurićima. To je prva poznata aplikacija memorija sa magnetnim mjehurićima. U standardnoj verziji ima taj terminal 20K bajtova memorije sa magnetnim mjehurićima, što je moguće proširiti na 80K bajtova. Memorijski podatkovni terminali udružuju iskustva povezana sa TI Silent 700 terminalima i prednosti magnetnih mjehurića za gradnju podatkovne memorije za terminale koji inače zahtijevaju mnogo skuplje memorijske medije kao na primjer kasete, papirne trake, floppy diske itd. 763 ali 765 terminal može doseći bilo koji zapis u memoriji za 15 ms, što je mnogo manje u odnosu na nekoliko sekundi do nekoliko minuta za kasetne sistem. TI označava da aplikacije za novi terminal uključuju "timesharing", novinarsko reporterstvo, bankovno poslovanje te poslovanje u osiguravajućim društvima. TI također znači da terminal može biti upotrijebljen za bilo koju aplikaciju u toku cijelog dana u režimu offline. Zapamćeni podatci mogu, u odgovarajućem trenutku, biti poslani u glavni računar. Ovako off-line unošenje podataka reducira vrijeme prenosa i zauzetost telefonske linije preko koje se komunicira. Sa samo 17 funti, model 765 Portable Memorijski terminal ima potpunu ASCII tastaturu, tihi termički printer, velike editorske sposobnosti itd. Model 765 Portable Memory Terminal stane 2995 \$, model 763 Memory Send-Receive Terminal stane 2695 \$, oba sa osnovnom konfiguracijom 20K. Dodatnih 20K memorije sa magnetnim mjehurićima stane 500 \$.

Američka firma AMI Microsystems objavljuje novu tehnologiju na području LSI tehnike koji nosi oznaku VMOS. Ta tehnologija bazira na trodimenzionalnim tranzistorima u silicijon substratu. Tehnologija daje ekstremno brza kola sa velikom gustošću i niskom cijenom.

Novi VMOS produkti i tipična vremena dostupa pri toj tehnologiji su sljedeći :

4K fully static RAM	(1K x 4)	80 ns
4K fully static RAM	(4K x 1)	80 ns
4K fully static low power	(4K x 1)	45 ns
8K fully static RAM	(1K x 8)	125 ns
16K ROM	(2K x 8)	100 ns
64K ROM	(8K x 8)	250 ns
16K EPROM	(2K x 8)	200 ns

Pregledovalnik paličnega koda. Firma Micro Scan Corporation, P.O.Box 705, Natick, MA 01760, USA prodaja pregledovalnik (čitalnik) paličnega koda (bar code). V paličnem kodu se npr. objavljuje programski pakseti v strokovnih časopisih (npr. v BYTE). S čitalnikom je moč čitati fotostatične kopije s hitrostjo 25 do 91 cm na sekundo. K vsakemu čitalniku je priložena dokumentacija nalagalnih programov za procesorje 6800, 6502, 8080 in Z-80. Čitalnik se lahko napaja z nestabilizirano napetostjo od 11,5 do 18 V, cena čitalnika je \$ 97,50.

Izboljšava procesorja SC/MP. Tvrdka National Semiconductor Corp. proizvaja novi procesor SC/MP-II, ki je od starega procesorja SC/MP dvakrat hitrejši in troši manj kot 200 mV; razen tega potrebuje eno samo napajalno napetost $+5V$ (prej $+5V$ in $-7V$) ter ga je moč povezati s TTL in CMOS vezji. Tvrdka prodaja t.i. nadomestni kit (\$ 18,50), ki vsebuje novi CPU, kristal za 2 MHz, priročnik in shemo.

Programator za EPROM-e. Podjetje Micro Pheripherals, 24 Matford Close, Westbury on Trym, Bristol BS10 6LR, England, GB izdeluje programator za popularne izbrisljive ROM, kot so 2704, 2708 in 2716. Naprava programira (vpiše) podatke v 1K čip v 2,5 minute; v tem času je vključena tudi verifikacija vpisnega vzorca. Programator se uporablja v povezavi s sistemom uporabnika, in sicer kot periferna naprava v več variantah. Osnovni mode programatorja je namenjen vpisovanju vzorcev v čipe 2704 (0,5K zlogov) in 2708 (1K zlogov) ter se napaja iz uporabniškega sistema; ta varianta stane \$ 199. Večji modeli imajo lasten usmernik, ročno nastavljanje ter čitanje z LE diodami.

Klub za uporabnike mikro procesorjev F8. Skupina amaterjev je ustanovila klub za uporabnike mikro računalnikov s procesorjem F8. Klub razpolaga s tremi F8-sistemi s po 5K in 16K pomnilniki. Člani kluba so zainteresirani za izmenjavo informacij in programov. Naslov je : G.W. Hemphill, 132 Scott Swamp Rd., Farmington, CT 06032, USA.

Mikrošahovski turnir. V avgustu 1978 bo v Louisville, KY, ZDA mikrošahovski turnir, ki ga organizira "Louisville Area Computer Club", z naslednjimi pravili : uporabljajo se samo 8-bitni mikro računalniki z maksimalno 16K pomnilnikom; programi so lahko napisani v strojnem ali visokem jeziku; dovoljeni so amaterski in komercialni računalniki; računalniki se naložijo iz poljubnega medija, vendar je potrebno po naložitvi zunanjo napravo odklopiti; turnir je časovno omejen; skupina sodnikov bo skrbela za pravi potek turnirja.

V letu 1979 bodo dovolili pomnilnike do 32K in v letu 1980 do 64K. Prvo, drugo in tretje mesto bo potrjeno z diplomom. Naslov organizatorja : Louisville Area Computer Club, 3028 Hunsinger Ln, Louisville, KY 40220, USA.

Tvrdba Norden Division of United Technologies je predstavila nov mikroročunalnik, LSI-11M popolnoma podrejen vojaškim zahtevam. LSI-11M je 16 bitni mikro računalnik, arhitektura pa je na osnovi mikro računalnika DIGITAL LSI-11, vendar je LSI-11M 4 krat hitrejši. LSI-11M vsebuje vodilo z visoko hitrostjo, dvosmerni interface s CPU-jem, spomin in I/O. Naprave priključene na vodilo lahko medsebojno komunicirajo z lastno hitrostjo, brez vpliva CPU-ja. I/O prenaša 833K besed na sekundo. Hitrost prenosa je neodvisna od zgradbe vodila. Za dodatne informacije pisati na : Norden Division of United Technologies, Norwalk, CT 06856.

Proizvodnja LSI integriranih vezij je poenostavljena z tehnologijo, ki jo je uvedla Hughes Aircraft Company. Gre za večslojno tankoplastno metalizacijo na monolitu. Nova tehnologija se imenuje SCAT (Hughes Schottky Cell Array TEchnology) omogoča gradnjo LSI vezij na osnovi knjižnice standardnih MSI in SSI Schottky TTL elementov. Rezultat SCAT tehnologije je zmanjšana poraba prostora in izboljšana zanesljivost.

Tvrdba Texas Instruments je dala na tržišče nov kalkulator namanjen programerjem. Kalkulator vsebuje osmiško, heksadecimalno in decimalno aritmetiko in pretvorbo iz ene baze v drugo. Tipične aplikacije kalkulatorja so npr. : pretvorba heksadecimalnih adres v decimalne, izračun relativne adrese ipd. Izrazi imajo lahko meša-

no bazo, maksimalna globina oklepajev pa je štiri. Negativna števila so prikazana na prikazovalniku v eniškem ali dvojiškem komplementu; velja za heksadecimalno in oktavno bazo.

Kalkulator izvaja operacije AND, OK, XOR, OR in SHIFT bit za bitom nad heksadecimalnimi in oktavnimi števili. Za popolnejšo informacijo pišite: Texas Instruments Inc., Inquiry Answering Service POB 5012, M/584 (ATT: TIP), Dallas TX 75222.

Z novo tehnologijo LSI je tehnologija komponent naraščala, z njo pa tudi gostota integracije logičnih vezij v čipu. S postavitvijo modernih tekočih trakov za množično proizvodnjo lahko za enako ceno podvojimo količino elementov ali pa jim izboljšamo zmogljivosti.

Vse to vpliva na proizvajalce, ki se ukvarjajo z razvojem materialne opreme. Za svoj obstoj morajo podvojiti proizvodnjo elementov ali pa izboljšati zmogljivosti.

Na osnovi teh teženj je INTEL-ov predstavnik najavil 16 bitni paralelni mikro procesor v enem čipu, INTEL 8086. Najavljen je za konec četrtega. Trg in cena se približujeta osem bitnemu 8080 A z ceno 30 \$ za čip v količinah nad 100 komadov. To pomeni povečanje števila uporabnikov družine 8080.

V sredini leta lahko pričakujemo, da bo tudi Zilog najavil svoj novi 16 bitni mikroprocesor Z86. Za konec leta pa se najavlja tudi Motorola s 16 bitnim AMPS. Predvideva se, da bo tudi njihova politika cen gibala skladno z Intelovo.

Toda INTEL 8086 ni prvi 16 bitni procesor v enem čipu na tržišču. Že pred 18 meseci so se najavljali proizvajalci kot so: Ferranti, General Automation, Texas Instruments, Data General in Fujitsu. Vendar so bili vsi ti čipi proizvedeni v majhnih količinah.

Predvideva se, da se bo z visoko proizvodnjo 16 bitnih mikro procesorjev približala do konca leta 1978 cena vseh 16 bitnih mikro rprocesorjev na nivo 30 \$. To znižanje cen se bo poznalo predvsem pri produkciji mikroročunalniških modulov in pri izdelavi celotnih sistemov za razvoj programske opreme.

Klub uporabnikov Cosmac Elf. Klub izdaja mesečnik s članki, ki se na našajo na mikro računalnik firme RCA, Cosmac Elf. Naslov kluba : C.E. Manry, 2102 Williamsburg Ct S, League City, TX 77573, USA.

Ura, koledar in kalkulator v enem vezju. Na tiskanem vezju za vodilo S-100 se nahaja ura realnega časa, koledar in 40 funkcij kalkulatorja. Uporaba ure in koledarja obsega tiskanje izhoda za čas in datum ter sprožitev alarma in časovniških operacij s pomočjo dveh posebnih števnikov na tiskanem vezju. Vezje se lahko napaja iz akumulatorja. 40 funkcij kalkulatorja obsega aritmetične funkcije s plavajočo vejico ter tigonometrične, algebrične in druge osnovne matematične funkcije. Vezje se proizvaja v treh vrstah sestavljenk : sestavljenka za čas, datum in kalkulator stane \$ 199; sestavljenka za čas in datum stane \$ 165; sestavljenka za kalkulator velja \$ 149. Naslov proizvajalca : Compu Time, POB 417, Huntington Beach, CA 92648, USA.

Testiranje osebja za obdelavo podatkov. Na Zapadu se je med drugimi preizkusi človeških sposobnosti uveljavilo tudi psihologično testiranje osebja, ki je zaposleno pri obdelavi podatkov. Pri teh testih gre za preizkuse obnašanja ter za ugotavljanje t.i. vedenjskih konstant posameznikov. Lastnosti, ki jih želijo preizkusiti s takimi testi posameznikov so: splošna inteligenca, logično-kombinatorijsko razmišljanje, analogijsko razmišljanje, sposobnost abstrahiranja, jezikovna inteligenca, analitične sposobnosti, koncentracija in delovna dinamika. Seveda ni mogoče vseh teh lastnosti ugotavljati z enim samim testom. Vendar velja omeniti, da je osem gornjih lastnosti preizkusljivih s kombinacijami drugih testov, kot so: luknjičavi, slikovni, računski/kombinatorični, besedni, izdržljivostni in obremenilni, slučajnostni preizkus in preizkus prostorske predstavljalivosti. Vsak preizkus se lahko nadomesti tudi s psihodiagnostičnim razgovorom.

Naklada časopisa BYTE. V mesecu februarju 1978 je naklada popularnega časopisa za majhne računalniške sisteme BYTE presegla število 110 000. Urednica časopisa obljublja, da se bo poslovanje časopisa izboljšalo, tako da ne bo več pripomb glede na ekspanzivnost obravnave želja in zahtev naročnikov.

Nizozemski računalniški klub. T klub ima 230 članov, tretjina njih ima osebne računalnike. Klub priporoča zlasti uporabo procesorjev 6800. Naslov kluba je: HCC, Delftsekade 12, 2266 AJ Leidschendam, Netherlands.

Kasetni zapisovalnik/čitalnik za 4800 Baud. Zapis je asinhronski NRZ tip, zapisovalnik omogoča tudi start/stop motorja. Razmerje napak je 10^8 pri 4800 Baud. Zapisovalnik/čitalnik se lahko uporablja za hitrosti med 110 do 4800 Baud preko elementov UART in PIA ter dobiva oz. oddaja signale na sponke RS232 ali TTL; tako ne uporablja nizkofrekvenčnega vmesnika. Ustavljanje in zagon motorja sta ročna ali s TTL in RS 232 signali. Hitrost traku je 8 cm/sek ali 4 cm/sek. (s posebno izbiro). Uporabljajo se lahko visokokvalitetne kasete Philips ali pa digitalne kasete. Pogonske napetosti so: 110 V, 220 V izmenične ali 12 V enosmerne. Tipične obdelave v zvezi s tem zapisovalnikom/čitalnikom so: zbiranje podatkov procesiranje tekstov, shranjevanje računalniških programov (nalaganje/shranjevanje) in podatkov. Programska oprema za obdelave na sistemih s procesorji 6800 in 8080 je na razpolago. Cene tega modela (Model CC-8) znaša \$ 175, naslov proizvajalca je: National Multiplex Corp., 3474 Rand Avenue, South Plainfield, NJ 07080, Box 288, USA.

Čas in datum za LSI-11. Tiskano vezje (za LSI-11) daje datum in razpolaga s funkcijami realnega časa za računalnik PDP-11, Model TCU-50 je namenjen LSI-11 in ima te lastnosti: daje mesec in dan ter čas v urah, minutah in sekundah potem, ko je bil sprožen posebni čitalni ukaz. Model TCU-100 je namenjen uporabi v računalniku PDP-11 ter ima še prekinitvene mehanizme, tako da lahko prekine sistem v določenih časih ali pa v regularnih intervalih. Vezje se lahko napaja iz akumulatorja. Cena teh enot je \$ 325 za TCU-50 in \$ 495 za TCU-100, naslov

proizvajalca je: Digital Pathways Inc., 4151 Middlefield Rd, Palo Alto, CA 94306, USA.

Mini/mikro računalnik RCA Cosmac. Elf II je 8 bitni mikro računalnik z možnostjo naslavljanja 64K zlogov, DMA, prekinitiv, 16 registri, ALU, 256 zlogi RAM-a, kompletno heksadecimalno tastaturo, dvomestnim heksadecimalnim prikazovalnikom, s petimi podnožji za razširitev sistema na glavni plošči tiskanega vezja Elf II. Ima kristalni takti oscilator in RCA 1861 video integrirano vezje za prikazovanje poljubnega pomnilniškega segmenta na video monitorju ali navadnem televizorju. Cena sestavljenke je \$ 99,95, naslov proizvajalca je: Netronics R D Ltd., Dept. BY2, 33 Litchfield Road, New Milford, CT 06776, USA.

POJMOVI I SKRAČENICE NA PODRUČJU RAČUNARSKE TEHNIKE

- Flow chart - Grafična predstava logičnog toka programa.
- Hibrid computer (Hibridni računar) - Računarski sistem u kojem su digitalne i analogne komponente kombinirane.
- Interpreter - Program koji naredbe u visokom programskom jeziku pretvara u mašinske naredbe i neposredno zatim ih izvršava.
- Interrupt (Prekid) - Suspenzija normalnog izvršavanja programa s ciljem da se izvrši određeni servis zahtjevan iz periferije. Poslije kompletiranja servisa nastavlja se sa izvođenjem osnovnog programa.
- Interrupt handler (Kontrolni program za prekinde) - Program koji ima kontrolu nad prekidima. Registri i statusi prekinutog programa se zapamte prije nego što se kontrola prenese na kontrolni program za prekinde.
- Interrupt level (Nivo prekida) - Oznaka prioriteta nekog prekida u odnosu na druge.
- Label (Oznaka) - Simbolična reprezentacija adrese koja se sastoji od jednog alfanumeričkog znaka ili niza znakova. Oznaka identificira instrukciju, program, konstantu, variablen ili podatkovno područje.
- Link Loader (Nalagač koji povezuje) - Nalagač koji omogućava povezivanje programa i subrutina u jedan program.
- Loader (Nalagač) - Program koji u memoriju prenese program u mašinskom kodu.
- Macro - Pseudo naredba koja se upotrebljava pri generiranju skupina mašinskih instrukcija koje izvršavaju određene zadatke.
- Mask - Uzorak bitova koji se upotrebljava za izvajanje dijela riječi (ili byte-a) tako da se izvrši logična funkcija AND sa tom riječi.
- Microcomputer - Mali računar projektiran prije svega za OEM aplikacije, često sa hard-wired programima.
- Micro instruction - Instrukcija u CPU koja operira na nižem nivou od mašinskih instrukcija.
- Micro program - Logični niz mikro instrukcija koji je zapamćen u CPU jedinici.

- Monitor - Program koji nadzire, kontrolira i verificira operiranje računarskog sistema.
- Overlay - Upotreba istog dijela glavne memorije za nekoliko različitih programskih segmenata ili zbirki podataka koji se vremenski ne poklapaju. Tu tehniku upotrebljavamo ako su potrebe za radnom memorijom veće od raspoloživih kapaciteta.
- Page - Memorijski segment koji može biti direktno adresiran jednom naredbom.
- Page register - Registar koji sadrži broj stranica koju želimo adresirati.
- Polling - Procedura testiranja statusa I/O naprava jedne za drugom s namjenom da se odredi ona koja zahtijeva servis.
- Port - I/O dement koji omogućava povezivanje perifernih naprava sa računarom dozvoljavajući visoku brzinu prenosa tehnikom direktnog dostupa do memorije.
- Program counter (Programski brojač) - Registar koji drži adresu sljedeće instrukcije koju treba da prihvati CPU.
- Programmed I/O - I/O tehnika pri kojoj CPU direktno kontrolira prenos podataka.
- Pseudo instruction - Naredba koju interpretira assembler i koja nema ekvivalenta u objektnom kodu.
- Radix - Broj različitih znakova za digit u određenom brojnom sistemu.

Hilberg, W.; Piloty, R (Hrsg.): Mikroprozessoren und ihre Anwendungen; München: Oldenbourg 1977, 284 S., 151 Abb., 7 Tab., DM 58.-

Hoffmann, R.: Rechenwerke und Mikroprogrammierung; München: Oldenbourg 1977, 197 S., 80 Abb, 17 Tab., DM 48.-

Kohonen, T.: Associative Memory A System Theoretical Approach; Berlin: Springer 1977; 176 S., 54 Abb., 7 Tab., DM 48.-

Martin, W.: Mikrocomputer in der Prozedatenverarbeitung; Aufbau und Einsatz der Mikrocomputer zur Überwachung, Steuerung und Regelung (Applied Computer Science 5); München: Hanser 1977, VIII, 205 S., 85 Abb., 13 Tab., DM 38.-

Pieper, F.: Einführung in die Programmierung paralleler Prozesse; München: Oldenbourg 1977, 133 S., 52 Abb., 35 Alg., DM 38.-

Seitzer, D.: Elektronische Analog-Digital-Umsetzer; Verfahren, Bauelemente, Beispiele; Berlin: Springer 1977, 143 S., DM 39.-

Uticaj primene elektronskih računara i automatizacije na poboljšanje rezultata poslovanja u privredi.

Izdavač: Privredna komora Jugoslavije, Sekretarijat PKJ za istraživanje i razvoj, Beograd, 1978.

Sadržaj:

1. Povezivanje na nivou jedne složene organizacije udruženog rada na primeru poljoprivrednog kombinata "Beograd".
2. Upotreba procesnih (mini) računara u čeličani, Uvodjenje i korišćenje NC-tehnologije i NC-mašina alatnih u Željezari Ravne.
3. Prednosti vođenja tehnoloških procesa i obrade podataka u industriji elektronskim računarom na primeru Železarne Ravne - Ravne na Koroškem.
4. Korišćenje elektronskih računara u ZP "Jugoturbina"
5. Programski proizvod CPPS sa mogućnošću primene u industriji nafte i petrohemijskoj industriji na primeru Rafinerije nafte Pančevo.
6. Mogućnosti primene elektronskih računara u elektroprivredi na primeru SOUR "Elektroprivreda BiH"
7. Osnovne informacije o primeni elektronskih računara u industriji sa primerom Topionica bakra RTB - Bor.
8. Prednosti primene elektronskih računara za vođenje procesa elektrolize na primeru Kombinata aluminija - Titograd.
9. Neki pogledi na primenu elektronskih računara u proizvodnim organizacijama, Operativni sistemi održavanja opreme elektronskim računarima.
10. Primjena elektronskih računara u SOUR "Uljanik" - Pula.
11. Numerička kontrola alatnih mašina u Fabrici ayiona "Utva"- Pančevo.
12. Ekonomija primene NC upravljanja.
13. Uvodjenje NC tehnologije u organizacije udruženog rada.

literatura in srečanja

Tvrdka Osborne and Associates, Inc., je ena vodilnih s područja izdavanja knjig iz mikroručalništva. Pogledajmo neka iz ponudbe:

1/ "An Introduction to Microcomputers: Vol. I - Basic Concepts by Adam Osborne; 350 strani, cena \$ 7,50; Book No.: 2001.

2/ "An Introduction to Microcomputers: Vol. II - Some Real Products" by Adam Osborne; 120 strani, cena \$ 15.00, Book No.: 3001.

3/ "Some Common Basic Programs", by Lon Poole and Mary Borchers; 200 strani, cena \$ 7,50, Book No.: 21002.

4/ "8080A in 8085 Assembly Language Programming" by Lance Leventhal; 450 strani, cena \$ 7,50; Book No.: 31003.

5/ "8080 Programming for Logic Design", by Adam Osborne, 300 strani, cena \$ 7,50, Book No. 4001.

6/ "6800 Programming for Logic Design" by Adam Osborne, 300 strani, cena \$ 7,50, Book No.: 5001.

7/ "Z80 Programming for Logic Design", by Adam Osborne and Susanna Jacobson, cena \$ 7,50, Book No.: 7001.

Osborne and Associates, Inc., P.O.Box 2036, Berkeley, CA 94702.

14. Primena elektronskih računara i automatizacije u baznoj hemijskoj industriji i ostalim procesnim tehnologijama.
15. Prednost vođenja tehnoloških procesa i obrade podataka u industriji na primeru Hemijske farmaceutske i kozmetičke industrije "Alkaloid" - Skopje.
16. Primena elektronskih računara u industriji konfekcije.
17. Tendencije razvoja sredstva za automatsku obradu podataka u složenim proizvodnim organizacijama.
18. Aktuelni problemi automatske obrade podataka u poljoprivredno-prehrambenom sistemu, Osijek.
19. Mikroročunari i mikroprocesori, Roboti.
20. Primena elektronskih računara i elektronike u grafičkoj i novinsko-izdavačkoj delatnosti.
21. Primena elektronskih računara u trgovini na primeru Preduzeće robnih kuća "Beograd".
22. Primena elektronskih računara u poslovnim bankama,
Primena elektronskih računara u Službi društvenog knjigovodstva,
Potreba za uključivanje Jugoslavije u SWIFT.
23. Prednosti vođenja procesa i obrade podataka na primeru PIK "Takovo" - Gornji Milanovac.

S. Alagić, M.A. Arbib : The Design of Well-Structured and Correct Programs, Springer-Verlag, Berlin-Heidelberg-New York, cena DM 27,90.

V februarju 1978 je v okviru zbirke "Texts and Monographs in Computer Science", ki jo urejmeta F.L. Bauer in D. Gries izšla knjiga priznanega jugoslovanskega strokovnjaka na področju računalniških znanosti v koavtorstvu s specialistom za umetno inteligenco M.A. Arbibom, ki obravnava razvoj dobro strukturiranih in korektnih programov. Na 260 straneh s 68 slikami je opisan navzdolnji razvoj programov, sestavljanje akcij in pravila dokazovanja, tipi podatkov, razvoj programov z dokazi korektnosti, procedure in funkcije, rekurzije ter programiranje z in brez stavkov go to. Avtorja uporabljata Pascal, primeri algoritmov pa so skrbno izbrani.

LETO 1978

14-17 marec, Toulouse, Francija

SYMPOSIUM ON MEDICAL INFORMATICS
Informacije: Prof. François Begon, CTI, Domaine de Vo-
lucéau, BP 105. 78150 Le Chesnay, France.

29-31 marec, Lancaster, Velika Britanija

COMPUTERS IN HIGHER EDUCATION
Informacije: Dr. E. Tagg, S Dept. of Mathematics,
University of Lancaster, Lancaster LA 1 4YI, UK.

3-5 april, Palo Alto, California, ZDA

FIFTH ANNUAL SYMPOSIUM ON COMPUTER ARCHITECTURE

Organizator: ACM-SIGARCH, IEEE-CS
Informacije: ACM HQ, 1133 Avenue of the Americas,
New York, NY 10036, USA.

11-13 april, Stocholm, Švedska

SEMINAR ON MINIS IN TERMINAL-BASED SYSTEMS
Organizator: On line Conference Limited
Informacije: Jean Seago, Online, Cleveland Road,
Uxbridge UB8 2DD, Middlesex, UK.

3-8 april, Paris, Francija

SALON INTERNATIONAL DES COMPOSANTS ELECTRONIQUES
Informacije: Französische Fachausstellungen Reisner-
strasse 50, 1030 Wien 3.

11-22 april, Madrid, Španija

INTERGOVERNMENTAL CONFERENCE ON STRATEGY AND
POLICIES FOR INFORMATICS
Organizator: IBI, UNESCO
Informacije: IBI Headquarters, POB 10253, 00144 Rome
Italy.

10-14 april, Zagreb

JUREMA
Informacije: Djura Salaja 5/VIII, PBO 398,
41000 Zagreb.

9-12 maj, London, Velika Britanija

EUROCOMP 78
Organizator: Online Conferences Limited
Informacije: EUROCOMP 78, Online, Cleveland Road,
Uxbridge UBB 2DD, England.

22-25 maj, Taormina, Sicilija

SIXTH INTERNATIONAL CODATE CONFERENCE
Organizator: CODATA
Informacije: CODATA Sekretarat, 51 Boulevard de Mont-
morency, 75016 Paris, France.

maj, Bled, Jugoslavija

CONFERENCE ON COMPUTERS IN BANKING AND FINANCE
Organizator: IAG
Informacije: IAG HQ, Paulus Potterstraat 40, Amsterdam
-1007, The Hetherlands.

29 maj - 3 junij, Dubrovnik, Jugoslavija

MEDJUNARNA ŠKOLA ELEKTRONIKE
Informacije: Institut za Nuklearne Nauke "Boris Kidrič",
POB 522, Beograd.

11-17 junij, Helsinki, Finska

IFAC CONGRESS
Organizator: Finish Society of Automatic Control
Informacije: Mr Olli Pezolanti, Hbyläämotie 18, 00380
Helsinki 38, Finland.

21-23 junij, Toulouse, Francija

1978 INTERNATIONAL SYMPOSIUM ON FAULT TOLERANT
COMPUTING-FTCS-8
Organizator: FTC Technical Committee of the Institute
for Electrical and Electronics Engineers Computer So-
ciety
Informacije: IEEE. 345 East 47th Street, New York, NY
10017, USA.

11-15 junij, Prague, Češkoslovaška

IFAC/IFIP 2nd INTERNATIONAL SYMPOSIUM "SOFTWARE FOR COMPUTER CONTROL"

Organizator in informacije: IFAC/IFIP TC 5

12-15 junij, Zürich, Švica

FOURTH INTERNATIONAL SYMPOSIUM AND EXHIBITION MINI- AND MICROCOMPUTERS AND THEIR APPLICATIONS-MIMI'78

Organizator: International Society of Mini-and Microcomputers

Informacije: Secretariat MIMI'78, Interconvention, c/o Swissair, Postfach, 8085, Zürich, Switzerland.

22-24 junij, London, UK

THE 1978 DO-IT-YOURSELF COMPUTER SHOW

Organizator: Online conferences limited

Informacija: Mrs. Lynne Swiatek, Cleveland road, Uxbridge UB 8 2DD, Middlesex, UK

22-23 junij, Gardone Riviera, Italija

ICPCI 78: INTERNATIONAL CONFERENCE ON THE PERFORMANCE OF COMPUTER INSTALLATIONS

Organizator: Politecnico di Milano, Università Cattolica del S.Cuore, Università Commerciale L. Bocconi, Università degli Studi di Milano, Università degli Studi di Pavia

Informacije: ICPCI 78 Conference Secretariat, CILEA, Via Raffaello Sanzio 4, 20090 Segrate (MI), Italia.

29-30 junij, Milano, Italija

INTERNATIONAL CONFERENCE ON MANAGEMENT DATA

Organizator: ACM Italian Chapter

Informacije: ACM HQ, 1133 Avenue of the Americas, New York, NY 10036, USA.

18-22 julij, Udine, Italija

FIFTH INTERNATIONAL CONFERENCE ON AUTOMATA LANGUAGES AND PROGRAMMING

Organizator: CISM, CNR

Informacije: CNR, Via Panciatchi 56/16, 50127 Florence, Italia.

24-26 julij, Newport Beach, California, ZDA

1978 SUMMER COMPUTER SIMULATION CONFERENCE

Informacije: Dr. Ralph C. Hunsinger, Comp. Science Dept., California State University - Chico, Chico, CA 95029, USA.

6-9 avgust, Jerusalem, Izrael

JERUSALEM CONFERENCE ON INFORMATION TECHNOLOGY

Informacije: Dr. A. Ralston, SUNY at Buffalo, 4226 Ridge Lea Road, Amherst, NY 14226, USA.

13-18 avgust, Rehovot, Izrael

MODELIZATION AND SIMULATION METHODS

Informacije: Bernard Zeigler, Dept. of Applied Mathematics, Weizmann Institute, Rehovot, Israel.

21-25 avgust, Amsterdam, Nizozemska

FOURTH INTERNATIONAL CONGRESS OF CYBERNETICS AND SYSTEMS

Informacije: Dr. J. Rose, c/o College of Technology, Feilden Street, Blackburn BB2 1LH, Lancs, UK; or Prof. van der Zouwen, Dept. of Research Methods, The Free University, de Boelelaan No 1115, Amsterdam,

Netherlands.

september, Berlin, ZRN

THIRD INTERNATIONAL CONGRESS ON ELECTRONIC INFORMATION PROCESSING (IKD)

Organizator: AMK-Berlin, IKD Professional Commission
Informacije: AMK-Berlin, Ausstellungs-Mess-Kongress-GmbH, Messedamm 22, D-1000, Berlin 19, Germany.

4-8 september, Manila, Filipini

SOUTH EAST ASIA REGIONAL COMPUTER CONFERENCE 1978 (SEARCC 78)

Organizator: Singapore Computer Society
Informacije: Robert Iau, President, Singapore Computer Society, c/o Central Provident Fund Board, Robinson Road, Singapore 1, Rep. of Singapore.

5-6 september, München, ZRN

SECOND INTERNATIONAL SYMPOSIUM ON DATA PROCESSING IN SOCIAL SECURITY

Organizator: International Social Security Association
Informacije: Dr. V. Velimsky, ISSA Data Processing Consultative Service, Boulevard de l'Empereur 7, B-1000 Brussels, Belgium.

11-15 september, London, Velika Britanija

MINI AND MICRO CONFERENCE

Organizator: Online Conferences Limited

Informacije: Jean Seago, Online, Cleveland Road, Uxbridge UB8 2DD, Middlesex, UK

13-15 september, West Berlin, ZRN

FOURTH INTERNATIONAL CONFERENCE ON VERY LARGE DATA BASES

Informacije: Herbert Weber (Gen Chm), Hahn-Meitner Institut, 1-Berlin 39, Postfach 390128, Federal Republic of Germany

18-22 september, Paris, Francija

CONVENTION INFORMATIQUE 78

Informacije: Convention Informatique, Secretariat, 6 place de Valois, 75001 Paris, France.

21-23 september, Bologna, Italija

INTERNATIONAL CONFERENCE ON INTERACTIVE TECHNIQUES IN COMPUTER-AIDED DESIGN

Informacije: Ira Cotton, Institute for Computer Science and Technology, National Bureau of Standards, Washington, DC 20234, USA.

10-12 oktober, Venezia, Italija

SECOND CONFERENCE OF THE EUROPEAN COOPERATION IN INFORMATICS ON INFORMATION SYSTEMS METHODOLOGY

Informacije: Prof. dr. P. Lockemann, Institut für Informatik II, Universität Karlsruhe, Postfach 6380, D-7500 Karlsruhe 1, Federal Republic of Germany; or Luciano Lippi, IBM Centro Scientifico, Dorsoduro 3228, 30124 Venice, Italia.

24-27 oktober, Copenhagen, Danska

COPENHAGEN CONFERENCE ON COMPUTER IMPACT

Organizator: Danish IAG
Informacije: CCCI-78, DIAG, The Danish EDP-Council, 58 Bredgade, DK-1260 Copenhagen K, Denmark.

30 oktober - 1 november, Washington, DC, ZDA

SYMPOSIUM ON FUTURE TRENDS IN COMPUTERIZED

STRUCTURAL ANALYSIS AND SYNTHESIS

Organizator: University of Maryland, George Washington University, NASA Langley Research Centre
 Informacije: University of Maryland, College Park, MD, USA.

7-10 november, Kyoto, Japonska

FOURTH INTERNATIONAL JOINT CONFERENCE ON PATTERN RECOGNITION

Informacije: Prof. T. Sakai, Kyoto University, Dept. of Information Science, Kyoto, Japan.

avtorji in sodelavci

Veljko ZGAGA (1955), diplomiral je 1977 godine na Elektrotehničkom fakultetu u Zagrebu, usmjerenje računarska tehnika i informatika. Zaposlen je na Institutu "Rudjer Bošković" u Zagrebu, gdje radi na primjeni mikroprocesora i razvoja mikroročunarskih sistema. Tokom studija bavio se problemima sinteze ljudskog glasa. Iz tog područja objavio je nekoliko radova u zemlji i jedan u inozemstvu.

Anton P. ŽELEZNIKAR (1928), diplomiral leta 1956 z diplomskim delom "Magnetostriksijska spominska zanka" na Fakulteti za elektrotehniko v Ljubljani. Magistriral leta 1966 in doktoriral leta 1967 s področja teorije algoritmov. Leta 1968 je prejel Kidričevo nagrado za dela s področja teorije algoritmov. Od leta 1965 je bil predsednik strokovnega odbora za obravnavanje podatkov (ETAN), organizator simpozijev Informatika na Bledu, kongresa IFIP 71 v Ljubljani. Od leta 1976 je predsednik Slovenskega društva Informatika in od leta 1977 izredni profesor na Fakulteti za elektrotehniko v Ljubljani. Objavil je vrsto del v jugoslovanskih časopisih.

Slavko TRIPKOVIĆ (1949), diplomiral na odsjeku za automatiku Elektrotehničkom fakulteta u Sarajevu 1973 godine. Magistriral na odsjeku za računarsku tehniku i informatiku, Elektrotehničkom fakulteta u Beogradu 1977 godine.

Od 1973 godine radi u Energoinvestu - Institut za računarske i informacione sisteme - na problemima primjene računara u upravljanju industrijskim procesima i programiranje u realnem vremenu. Sadašnje područje interesovanja i rada obuhvata distribuirane sisteme (realno vrijeme) na bazi mikro/mini računara.

Roman TROBEC (1953), diplomiral 1976 leta na Fakulteti za elektrotehniko v Ljubljani, smer industrijska elektronika, s temo diplomske naloge: "Matematični model za funkcionalno diagnostiko sistemov." Zaposlen v oddelku za elektroniko na Institutu "Jožef Stefan" v Ljubljani. Področje dela je odkrivanje metod za funkcionalno diagnostiko sistemov (mikroročunalnikov, računalniško vodenih sistemov) in uporaba mikroročunalnikov za potrebe industrije, kakor tudi izdelava programske in systemske opreme za te naloge.

vsebina letnika 1977

Alagić S., R. Jović, Dž. Ridjanović:
 Manipulisanje hijerarhijskom bazom podataka (št. 1, str. 10).

Alagić S.:
 Cleaning up Unstructured Algorithms Using Invariants (št. 2, str. 17).

Batagelj V.:
 Urejanje zaporedij (št. 1, str. 56).

Batagelj V.:
 Kako narišemo pravilni mnogokotnik (št. 4, str. 50).

Bratko I.:
 Drugo svetovno šahovsko prvenstvo računalnikov (št. 4, str. 67).

Čećez-Kecmanović D.:
 Okvir za izračunavanje informacijskih sistema (št. 1, str. 25).

Davčev D.:
 Memoire virtue'le et la technique d'optimisation adaptee a l'interpreteur APL sur Mitra 15 (št. 1, str. 50).

Davčev D.:
 Function de tri croissant et de tri decroissant APL Mitra 15 (št. 2, str. 62).

Djordjević L.N.:
 On N-th Root Evaluation by Iterative Methods with Parameters (št. 4, str. 61).

Doberlet V., M. Mrak, B. Popović, M. Gazvoda:
 Problematika izvedbe in vključevanja velikih programskih sistemov realnega časa (št. 3, str. 12).

Dujmović J.J.:
 A Programming System for Editing Annotated Bibliographies (št. 2, str. 43).

Dujmović J.J.:
 The Preference Scoring Method for Decision Making - Survey, Classification and Annotated Bibliography (št. 2, str. 26).

Exel M.:
 Komunikacija med sekvenčnimi procesi - pregled, del I (št. 1, str. 41).

Exel M.:
 Komunikacija med sekvenčnimi procesi - pregled, del II (št. 2, str. 50).

Faleskini R.:
 Računalniški center Univerze v Ljubljani (št. 1, str. 7)

Glušac D.:
 Pouzdanost prenosa podataka izmedju magnetnih medijuma i CRC-Kontrola (št. 3, str. 46).

Glušac D.:
 Neki aspekti uvodjenja komunikacione mreže on-line real-time sistema Beogradske banke (št. 4, str. 39)

Gojanović D.:
 Graphics Machine Independence with Help of Network Graphic Protocol (št. 3, str. 35).

Kalan J.:
 Primer koncepta računalniškega omrežja (št. 3, str. 40).

Kalin T., I. Lajovic, V. Rajković, D. Viličić:
 Osnovni parametri in usmeritev načrtovanja računalniške mreže v Sloveniji (št. 3, str. 15).

- Kette B.:
Specifičnosti mikroprocesorske izvedbe mjerila broja okretaja (št. 3, str. 27).
- Kodek D.:
Primer preprostega mikro računalnika z mikro procesorjem Motorola 6800 (št. 1, str. 46).
- Kolbezen P., B. Mihovilović, Z. Milavc:
Rešitve nekaterih problemov krmiljenja objekta z mikroprocesorjem v realnem času (št. 2, str. 35).
- Kovačević M., D. Novak, A.P. Železnikar:
Monitori za mikro sisteme sa procesorom 6800 (št. 2, str. 20).
- Kovačević M., D. Novak, J. Novak, A.P. Železnikar:
Preizkušanje mikroročunalniških modulov (št. 3, str. 32).
- Kovačević M.:
BUS standardi (št. 4, str. 64).
- Michie D.:
New Face of Artificial Intelligence (št. 3, str. 5).
- Muftić S.:
A Model of Secure Computer System (št. 1, str. 33).
- Nikolić S., Ž. Tošić:
Jedan postupak za mikroprocesorsko ostvaranje preki-dačkih funkcija (št. 4, str. 31).
- Novak D., M. Kovačević, A.P. Železnikar:
Priključitev hitrega čitalnika luknjanega traku na mikro računalnik (št. 3, str. 20).
- Papić N., M. Kovačević:
Univerzalni serijsko vhodno/izhodni modul (št. 4, str. 44).
- Popov O.B.:
Procesor za aritmetično-logični izrazi (št. 4, str. 54).
- Ponebšek B.:
The Microcomputer Families 8080, 6800, F8, Z80 (št. 1, str. 21).
- Popovič B., M. Exel, M. Mekinda:
Primjena monitorskog koncepta u izgradnji operacionog sistema za periodično aktiviranje programa (št. 2, str. 13).
- Sgall D.:
Linguistics and Automatic Processing of Texts (št. 2, str. 68).
- Tasič J., L. Lenart, B. Blatnik:
Ceneni mikroročunalnik s procesorjem SC/MP (št. 3, str. 50).
- Tomović R.:
Sistemski prilaz razvoju i projektovanju gradskih informacionih sistema (št. 4, str. 5).
- Trampuž C., A. Ferligoj:
Nekateri vidiki uporabe računalnikov v sociologiji in politologiji (št. 1, str. 17).
- Tireford H.:
Linking Fortran and Assembly Language Programs (št. 2, str. 60).
- Trobec R., J. Korenini, F. Novak:
Dinamični MOS pomnilniki (št. 2, str. 55).
- Tripković S., M. Aškrabič:
Kros-sembler i simulator za mikro/mini računare (št. 3, str. 23).
- Virant J.:
Reševanje problemov računalniškega kadra in kadra informatike v SR Sloveniji (št. 1, str. 38).
- Zgaga V.:
Upis kontinuiranih signala u mikro računalno (št. 2, str. 47).
- Zgaga V.:
TV-Terminal za mikroročunalno (št. 4, str. 34).
- Zupan J.:
Kriptografija na prelomu (št. 4, str. 24).
- Železnikar A.P.:
O časopisu Informatica (št. 1, str. 3).
- Železnikar A.P., Ozimek I., M. Kovačević, D. Novak:
Programiranje mikro računalnikov s procesorjem Z80 (št. 2, str. 5).
- Železnikar A.P., M. Kovačević, D. Novak:
Razvoj dinamičnih pomnilnikov za mikro računalnike (št. 4, str. 11).

CENIK OGLASOV

Ovitek - notranja stran (za letnik 1977)	
2 stran -----	16.000 din
3 stran -----	12.000 din
Vmesne strani (za letnik 1977)	
1/1 stran -----	8.000 din
1/2 strani -----	5.000 din
Vmesne strani (za posamezno številko)	
1/1 stran -----	3.000 din
1/2 strani -----	2.000 din
Oglas o potrebah po kadrih (za posamezno številko)	
-----	1.000 din

Razen oglasov v klasični obliki so zaželjene tudi krajše poslovne, strokovne in propagandne informacije in članki. Cena objave tovrstnega materiala se bo določala sporazumno.

ADVERTIZING RATES

Cover page (for all issues of 1977)	
2nd page -----	16.000 din
3rd page -----	12.000 din
Inside pages (for all issues of 1977)	
1/1 page -----	8.000 din
1/2 page -----	5.000 din
Inside pages (individual issues)	
1/1 page -----	3.000 din
1/2 page -----	2.000 din
Rates for classified advertizing:	
each ad -----	1.000 din

In addition to advertisements, we welcome short business or product news, notes and articles. The related charges are negotiable.

mikroračunalniki

**STE SE ŽE ODLOČILI ZA UPORABO MIKRORAČUNALNIKA V PROIZVODNJI,
ALI PA ŽE IMATE MIKRORAČUNALNIK IN ŽELITE IZVESTI REŠITVE PO
VAŠI ZAMISLI?**

NUDIMO VAM:

- NAJSODOBNEJŠE TEHNIČNE REŠITVE Z UPORABO NOVE TEHNOLOGIJE**
- PROJEKTIRANJE VEČJIH SISTEMOV (MULTIPROCESORSKIH)**
- DIAGNOSTICIRANJE PROCESOV**
- IZDELAVO IN TESTIRANJE UPORABNIŠKIH PROGRAMOV S POMOČJO
VELIKIH SISTEMOV**
- KONZULTACIJE ZA UPORABO MIKRORAČUNALNIŠKIH SISTEMOV V
INDUSTRIJI IN GOSPODARSTVU**
- IMAMO IZKUŠNJE Z UPORABO MIKROPROCESORJEV (Z 80, 6800,
F-8, 8080, 2650, PFL 16, SC/MP), DINAMIČNIH POMNILNIKOV, PERI-
FERIJE ITN.**

**Institut Jožef Stefan, Ljubljana, Jamova 39
ODSEK ZA RAČUNALNIŠTVO IN INFORMATIKO
Telefon (061)63-261 Int. 305**

MIKRORAČUNALNIKI